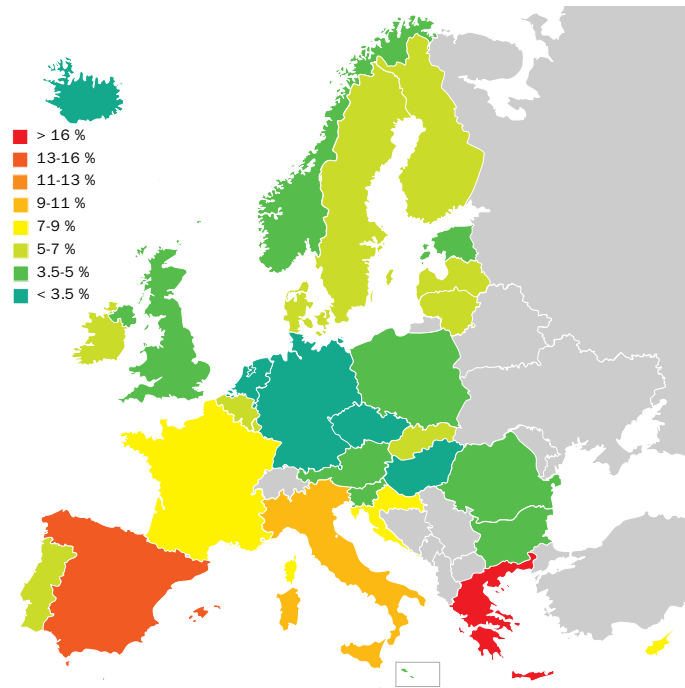


Case-Study zur Arbeitslosigkeit in Deutschland

Ziel der Case-Study

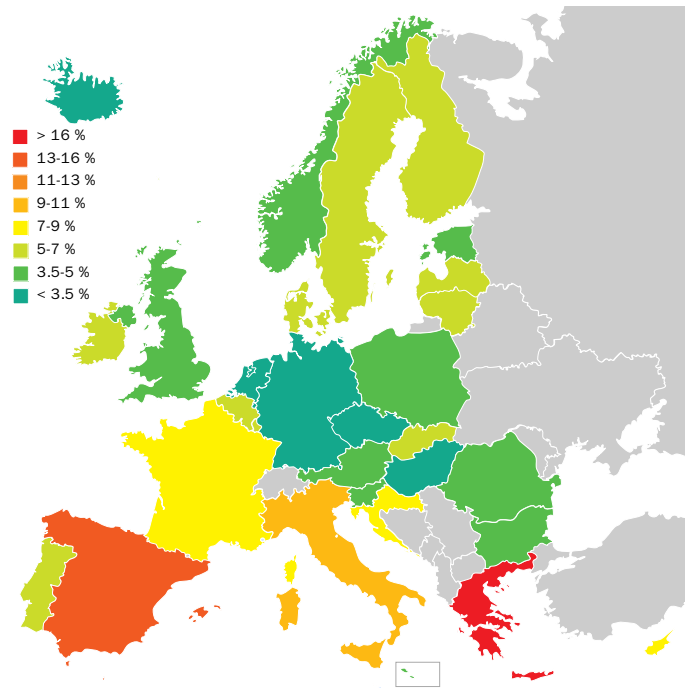
Deutschland hat europaweit eine der niedrigsten Arbeitslosenquoten:



Quelle: [Von Heycci - Daten von Eurostat, CC BY-SA 2.5](#)

Ziel der Case-Study

Deutschland hat europaweit eine der niedrigsten Arbeitslosenquoten:



Doch gilt dies für alle Regionen in Deutschland?
Warum ist die Arbeitslosenquote in manchen Regionen höher als in anderen?

Dem wollen wir in dieser Case-Study auf den Grund gehen.

Quelle: [Von Heycci - Daten von Eurostat, CC BY-SA 2.5](#)

Ziele der Case Study

Diese Case-Study besteht aus **mehreren Teilen** und wird Sie durch die komplette Vorlesung als **konkretes Anschauungsobjekt** begleiten.

Diese Case-Study dient als:

- + konkretes und umfangreiches Beispiel für ein Projekt
- + ökonomische und geographische Kenntnisse über Deutschland erhalten
- + Beispiel wie statistische und programmiertechnische Kenntnisse in der empirischen Arbeit eingesetzt werden können

Datensätze herunterladen

Ersten Teil der Case Study

- + Daten einlesen
- + Daten bearbeiten und in eine geeignete Form bringen (`tidy`)

Anwenden auf

- + Daten zur Arbeitslosenstatistik
- + Daten zur Verschuldung einzelner Landkreise bzw. Gemeinden
- + Daten zum BIP

Wichtig für die Datenbeschaffung

- + Zuverlässige und qualitativ hochwertige Datenquellen ausfindig machen
- + Automatisierten Download programmieren
- + Einlesen, verarbeiten und zusammenführen verschiedener Datensätze in R

Wichtig für die Datenbeschaffung

- + Zuverlässige und qualitativ hochwertige Datenquellen ausfindig machen
- + Automatisierten Download programmieren
- + Einlesen, verarbeiten und zusammenführen verschiedener Datensätze in R

Verbindung zum 2. RTutor Problem Set:

- + **Im Problem Set:** Kennzahlen zu verschiedenen Ländern der europäischen Union
- + **Hier:** Kennzahlen innerhalb Deutschlands

Wichtig für die Datenbeschaffung

- + Zuverlässige und qualitativ hochwertige Datenquellen ausfindig machen
- + Automatisierten Download programmieren
- + Einlesen, verarbeiten und zusammenführen verschiedener Datensätze in R

Verbindung zum 2. RTutor Problem Set:

- + **Im Problem Set:** Kennzahlen zu verschiedenen Ländern der europäischen Union
- + **Hier:** Kennzahlen innerhalb Deutschlands

Sowohl in der Case-Study als auch in den RTutor Problem Sets treffen Sie auf konkrete Probleme, die Sie mit ihren Kenntnissen aus der Vorlesung lösen sollen.

Daten beschaffen

Woher beziehen wir unsere Informationen?

Daten beschaffen

Woher beziehen wir unsere Informationen?

- ✚ Die Informationen über die Verschuldung der **Gemeinden** finden wir auf den Seiten des Statistischen Bundesamts im Report: [Integrierte Schulden der Gemeinden und Gemeindeverbände](#).
- ✚ Die Informationen zur Arbeitslosigkeit auf **Verwaltungsgemeinschaftsebene** finden wir auf den Seiten der [Bundesagentur für Arbeit](#).
- ✚ Die Informationen zum BIP auf **Landkreisebene** finden wir auf den Seiten der [Statistischen Ämter des Bundes und der Länder](#).

Daten beschaffen

Woher beziehen wir unsere Informationen?

- ✚ Die Informationen über die Verschuldung der **Gemeinden** finden wir auf den Seiten des Statistischen Bundesamts im Report: [Integrierte Schulden der Gemeinden und Gemeindeverbände](#).
- ✚ Die Informationen zur Arbeitslosigkeit auf **Verwaltungsgemeinschaftsebene** finden wir auf den Seiten der [Bundesagentur für Arbeit](#).
- ✚ Die Informationen zum BIP auf **Landkreisebene** finden wir auf den Seiten der [Statistischen Ämter des Bundes und der Länder](#).

Zuverlässige und qualitativ hochwertige Datenquellen ausfindig machen ✓

Nötige Pakete laden

```
library(readxl)
library(skimr)
library(tidyverse)
```

```
## — Attaching packages ————— tidyverse 1.3.0 —
```

```
## ✓ ggplot2 3.3.0    ✓ purrr  0.3.4
## ✓ tibble  3.0.4    ✓ dplyr  1.0.2
## ✓ tidyr   1.1.2    ✓ stringr 1.4.0
## ✓ readr   1.3.1    ✓ forcats 0.5.0
```

```
## — Conflicts ————— tidyverse_conflicts() —
```

```
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Daten herunterladen

- + Daten können von URLs mit Befehlen aus den Paketen `readxl` und `readr` direkt eingelesen werden
 - + Für Text und Excel-Dateien
- + Allerdings, wenn URL nicht mehr verfügbar, was dann?
 - + Daten immer mit `download.file()` herunterladen und in einem Unterordner `data` abspeichern!

Daten herunterladen

- + Daten können von URLs mit Befehlen aus den Paketen `readxl` und `readr` direkt eingelesen werden
 - + Für Text und Excel-Dateien
- + Allerdings, wenn URL nicht mehr verfügbar, was dann?
 - + Daten immer mit `download.file()` herunterladen und in einem Unterordner `data` abspeichern!

Automatisierten Download programmieren (wird in der ausformulierten Case-Study gemacht) (✓)

Daten herunterladen

- + Daten können von URLs mit Befehlen aus den Paketen `readxl` und `readr` direkt eingelesen werden
 - + Für Text und Excel-Dateien
- + Allerdings, wenn URL nicht mehr verfügbar, was dann?
 - + Daten immer mit `download.file()` herunterladen und in einem Unterordner `data` abspeichern!

Automatisierten Download programmieren (wird in der ausformulierten Case-Study gemacht) (✓)

Wir haben die Daten bereits im Github Repository `case-study-germany` heruntergeladen und abgespeichert. Klonen Sie dieses Repository von Github auf ihren PC!

05:00

Daten einlesen

Unterschiedliche Tabellenblätter, welches ist für uns interessant?

```
excel_sheets("./data/Arbeitslose_2017.xlsx")
```

```
## [1] "Deckblatt"      "Impressum"      "Inhalt"  
## [4] "Hinweis"        "5"              "6"  
## [7] "7"              "8"              "9"  
## [10] "Statistik-Infoseite"
```

Daten einlesen

Unterschiedliche Tabellenblätter, welches ist für uns interessant?

```
excel_sheets("./data/Arbeitslose_2017.xlsx")
```

```
## [1] "Deckblatt"      "Impressum"      "Inhalt"  
## [4] "Hinweis"        "5"              "6"  
## [7] "7"              "8"              "9"  
## [10] "Statistik-Infoseite"
```

Vermutung: Durch Tabellenblatt "Inhalt" könnten wir schlauer werden

```
alo_inhalt <- read_xlsx("./data/Arbeitslose_2017.xlsx", sheet = "Inhalt")
head(alo_inhalt, 15)
```

```
## # A tibble: 15 x 4
##   ...1          ...2          ...3 `Arbeitslose nach Gemeinden   Ja...
##   <chr>         <chr>         <lgl> <chr>
## 1 <NA>          <NA>          NA    <NA>
## 2 Inhaltsverzeichnis <NA>          NA    <NA>
## 3 <NA>          <NA>          NA    <NA>
## 4 Berichtsmonat: Jahre... <NA>          NA    <NA>
## 5 <NA>          <NA>          NA    <NA>
## 6 Arbeitslose nach Gem... <NA>          NA    <NA>
## 7 <NA>          Impressum     NA     2
## 8 <NA>          Inhaltsverze... NA     3
## 9 <NA>          Hinweise      NA     4
## 10 Zugang       <NA>          NA    <NA>
## 11 <NA>          Insgesamt     NA     5
## 12 Bestand     <NA>          NA    <NA>
## 13 <NA>          Insgesamt     NA     6
## 14 <NA>          Männer        NA     7
## 15 <NA>          Frauen        NA     8
```

```
alo_inhalt <- read_xlsx("./data/Arbeitslose_2017.xlsx", sheet = "Inhalt")
head(alo_inhalt, 15)
```

```
## # A tibble: 15 x 4
##   ...1          ...2          ...3 `Arbeitslose nach Gemeinden` Ja...
##   <chr>         <chr>         <lgl> <chr>
## 1 <NA>          <NA>          NA    <NA>
## 2 Inhaltsverzeichnis <NA>          NA    <NA>
## 3 <NA>          <NA>          NA    <NA>
## 4 Berichtsmonat: Jahre... <NA>          NA    <NA>
## 5 <NA>          <NA>          NA    <NA>
## 6 Arbeitslose nach Gem... <NA>          NA    <NA>
## 7 <NA>          Impressum     NA    2
## 8 <NA>          Inhaltsverze... NA    3
## 9 <NA>          Hinweise      NA    4
## 10 Zugang       <NA>          NA    <NA>
## 11 <NA>          Insgesamt     NA    5
## 12 Bestand      <NA>          NA    <NA>
## 13 <NA>          Insgesamt     NA    6
## 14 <NA>          Männer        NA    7
## 15 <NA>          Frauen        NA    8
```

```
alo_inhalt <- read_xlsx("./data/Arbeitslose_2017.xlsx", sheet = "Inhalt")
head(alo_inhalt, 15)
```

```
## # A tibble: 15 x 4
##   ...1          ...2          ...3 `Arbeitslose nach Gemeinden` Ja...
##   <chr>         <chr>         <lgl> <chr>
## 1 <NA>         <NA>         NA    <NA>
## 2 Inhaltsverzeichnis <NA>         NA    <NA>
## 3 <NA>         <NA>         NA    <NA>
## 4 Berichtsmonat: Jahre... <NA>         NA    <NA>
## 5 <NA>         <NA>         NA    <NA>
## 6 Arbeitslose nach Gem... <NA>         NA    <NA>
## 7 <NA>         Impressum    NA    2
## 8 <NA>         Inhaltsverze... NA    3
## 9 <NA>         Hinweise     NA    4
## 10 Zugang      <NA>         NA    <NA>
## 11 <NA>         Insgesamt    NA    5
## 12 Bestand     <NA>         NA    <NA>
## 13 <NA>         Insgesamt    NA    6
## 14 <NA>         Männer       NA    7
## 15 <NA>         Frauen       NA    8
```

Alternative: Schauen Sie sich die Excel-Datei in Excel oder LibreOffice an und entscheiden Sie dann, welches Tabellenblatt Sie einlesen möchten.

Spezifizieren welche Spalten eingelesen werden sollen

Welche Information benötigen wir aus der Tabelle

- ✚ Die Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam)
- ✚ Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis (z.B. nur SGB II)
- ✚ Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis und ein bestimmtes Alter (z.B. SGB II alle unter 25 Jahre)

Spezifizieren welche Spalten eingelesen werden sollen

Welche Information benötigen wir aus der Tabelle

- + Die Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis (z.B. nur SGB II)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis und ein bestimmtes Alter (z.B. SGB II alle unter 25 Jahre)

Spezifizieren welche Spalten eingelesen werden sollen

Welche Information benötigen wir aus der Tabelle

- + Die Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis (z.B. nur SGB II)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis und ein bestimmtes Alter (z.B. SGB II alle unter 25 Jahre)

Was ist hier eine Beobachtung?

Spezifizieren welche Spalten eingelesen werden sollen

Welche Information benötigen wir aus der Tabelle

- + Die Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis (z.B. nur SGB II)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis und ein bestimmtes Alter (z.B. SGB II alle unter 25 Jahre)

Was ist hier eine Beobachtung?

Weiterhin benötigen wir noch den `Schlüssel` und den Gemeindennamen.

Wie können wir die von uns benötigte Information möglichst einfach extrahieren?

Spezifizieren welche Spalten eingelesen werden sollen

Welche Information benötigen wir aus der Tabelle

- + Die Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis (z.B. nur SGB II)
- + Die Anzahl der Arbeitslosen pro Gemeinde für einen bestimmten Rechtskreis und ein bestimmtes Alter (z.B. SGB II alle unter 25 Jahre)

Was ist hier eine Beobachtung?

Weiterhin benötigen wir noch den `Schlüssel` und den Gemeinidenamen.

Wie können wir die von uns benötigte Information möglichst einfach extrahieren?

- + Der einfachste Weg: Die ersten acht Zeilen abzuschneiden und die Daten erst ab dort einzulesen.
- + Anschließend behalten wir nur die ersten 3 Spalten

```
alo_skip <- read_xlsx("./data/Arbeitslose_2017.xlsx")
```

```
alo_skip <- read_xlsx("./data/Arbeitslose_2017.xlsx")
```

```
alo_skip
```

```
## # A tibble: 11,346 x 32
##   Schlüssel Gemeinde   `1`   `2`   `3`   `4`   `5`   `6`   `7`   `8`
##   <chr>      <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 <NA>      Deutsch... 2.53e6 47840 230694 839545 529002 655249 14022 71133
## 2 01        Schlesw... 9.24e4 2072 9703 30138 18788 18688 437 2297
## 3 01001000 Flensbu... 4.51e3 108 566 1120 654 983 18 141
## 4 01002000 Kiel, L... 1.23e4 220 1100 3306 1842 3103 54 294
## 5 01003000 Lübeck,... 9.69e3 180 836 2902 1667 2050 41 204
## 6 01004000 Neumüns... 3.84e3 105 440 1134 685 724 10 45
## 7 01051    Dithmar... 4.63e3 147 634 1402 826 791 27 141
## 8 01051001 Albersd... 1.50e2 5 23 43 23 24 0 4
## 9 01051002 Arkebek 5.00e0 0 1 1 1 2 NA NA
## 10 01051003 Averlak 1.50e1 0 3 4 4 1 NA 0
## # ... with 11,336 more rows, and 22 more variables: `9` <dbl>, `10` <dbl>,
## # `11` <dbl>, `12` <dbl>, `13` <dbl>, `14` <dbl>, `15` <dbl>, `16` <dbl>,
## # `17` <dbl>, `18` <dbl>, `19` <dbl>, `20` <dbl>, `21` <dbl>, `22` <dbl>,
## # `23` <dbl>, `24` <dbl>, `25` <dbl>, `26` <dbl>, `27` <dbl>, `28` <dbl>,
## # `29` <dbl>, `30` <dbl>
```

```
alo_skip <- read_xlsx("./data/Arbeitslose_2017.xlsx")
alo_skip %>%
  select(c(`Schlüssel`, Gemeinde, `1`))
```

```
## # A tibble: 11,346 x 3
##   Schlüssel Gemeinde      `1`
##   <chr>      <chr>      <dbl>
## 1 <NA>      Deutschland 2532837
## 2 01        Schleswig-Holstein 92434
## 3 01001000 Flensburg, Stadt 4512
## 4 01002000 Kiel, Landeshauptstadt 12345
## 5 01003000 Lübeck, Hansestadt 9692
## 6 01004000 Neumünster, Stadt 3836
## 7 01051    Dithmarschen 4628
## 8 01051001 Albersdorf 150
## 9 01051002 Arkebek 5
## 10 01051003 Averlak 15
## # ... with 11,336 more rows
```

```
alo_skip <- read_xlsx("./data/Arbeitslose_2017.xlsx")
alo_skip %>%
  select(c(`Schlüssel`, Gemeinde, `1`)) %>%
  rename(Regionalschlüssel = `Schlüssel`,
         alo = `1`)
```

```
## # A tibble: 11,346 x 3
##   Regionalschlüssel Gemeinde      alo
##   <chr>              <chr>      <dbl>
## 1 <NA>              Deutschland 2532837
## 2 01                Schleswig-Holstein 92434
## 3 01001000         Flensburg, Stadt 4512
## 4 01002000         Kiel, Landeshauptstadt 12345
## 5 01003000         Lübeck, Hansestadt 9692
## 6 01004000         Neumünster, Stadt 3836
## 7 01051            Dithmarschen 4628
## 8 01051001         Albersdorf 150
## 9 01051002         Arkebek 5
## 10 01051003        Averlak 15
## # ... with 11,336 more rows
```



```
alo_skip <- read_xlsx("./data/Arbeitslose_2017.xlsx")

alo_skip %>%
  select(c(`Schlüssel`, Gemeinde, `1`)) %>%
  rename(Regionalschlüssel = `Schlüssel`,
         alo = `1`) %>%
  filter(!is.na(alo) & Gemeinde!= "Deutschland")
```

```
## # A tibble: 11,343 x 3
##   Regionalschlüssel Gemeinde      alo
##   <chr>             <chr>      <dbl>
## 1 01                 Schleswig-Holstein 92434
## 2 01001000          Flensburg, Stadt 4512
## 3 01002000          Kiel, Landeshauptstadt 12345
## 4 01003000          Lübeck, Hansestadt 9692
## 5 01004000          Neumünster, Stadt 3836
## 6 01051             Dithmarschen      4628
## 7 01051001          Albersdorf        150
## 8 01051002          Arkebek            5
## 9 01051003          Averlak            15
## 10 01051004         Bargenstedt       17
## # ... with 11,333 more rows
```

```
#Abspeichern als Datensatz data_alo
```

```
data_alo <- alo_skip %>%
  select(c(`Schlüssel`, Gemeinde, `1`)) %>%
  rename(Regionalschlüssel = `Schlüssel`,
         alo = `1`) %>%
  filter(!is.na(alo) & Gemeinde!= "Deutschland")
```

Konsistenzcheck

- + Machen die Angaben Sinn und sind die Daten in sich konsistent?
- + Externe Datenquelle suchen und intern auf Konsistenz prüfen.
- + Informationen aggregieren und mit anderen Quellen vergleichen

Konsistenzcheck

- + Machen die Angaben Sinn und sind die Daten in sich konsistent?
- + Externe Datenquelle suchen und intern auf Konsistenz prüfen.
- + Informationen aggregieren und mit anderen Quellen vergleichen
- + Zunächst: Anzahl an Arbeitslosen für jedes **Bundesland** in 2017.
 - + zweistelligen `Regionalschlüssel`
 - + "Buchstaben" für jeden `Regionalschlüssel` zählen (`nchar()` (number of characters))
- + **Alternative Datenquelle:** [Die Anzahl der Arbeitslosen für das Jahr 2017 unterteilt nach Ländern der Arbeitsagentur](#)
 - + Wichtig: Tabellenblatt 8

data_alo

```
## # A tibble: 11,343 x 3
##   Regionalschlüssel Gemeinde      alo
##   <chr>                <chr>    <dbl>
## 1 01                    Schleswig-Holstein 92434
## 2 01001000              Flensburg, Stadt 4512
## 3 01002000              Kiel, Landeshauptstadt 12345
## 4 01003000              Lübeck, Hansestadt 9692
## 5 01004000              Neumünster, Stadt 3836
## 6 01051                Dithmarschen      4628
## 7 01051001              Albersdorf         150
## 8 01051002              Arkebek            5
## 9 01051003              Averlak            15
## 10 01051004              Bargenstedt       17
## # ... with 11,333 more rows
```

```
data_alo %>%
```

```
  filter(nchar(Regionalschluessel) == 2)
```

```
## # A tibble: 16 x 3
```

```
##   Regionalschluessel Gemeinde      alo
##   <chr>              <chr>      <dbl>
## 1 01                 Schleswig-Holstein  92434
## 2 02                 Hamburg           69248
## 3 03                 Niedersachsen     244260
## 4 04                 Bremen            35687
## 5 05                 Nordrhein-Westfalen 701219
## 6 06                 Hessen            166286
## 7 07                 Rheinland-Pfalz   106299
## 8 08                 Baden-Württemberg 212837
## 9 09                 Bayern            231353
## 10 10                Saarland          34672
## 11 11                Berlin            168991
## 12 12                Brandenburg       92648
## 13 13                Mecklenburg-Vorpommern 70982
## 14 14                Sachsen           140348
## 15 15                Sachsen-Anhalt    96960
## 16 16                Thüringen         68614
```

```
data_alo %>%  
  filter(nchar(Regionalschlüssel) == 2) %>%  
  rename(bundesland = Regionalschlüssel)
```

```
## # A tibble: 16 x 3  
##   bundesland Gemeinde      alo  
##   <chr>      <chr>      <dbl>  
## 1 01      Schleswig-Holstein  92434  
## 2 02      Hamburg          69248  
## 3 03      Niedersachsen     244260  
## 4 04      Bremen            35687  
## 5 05      Nordrhein-Westfalen 701219  
## 6 06      Hessen            166286  
## 7 07      Rheinland-Pfalz    106299  
## 8 08      Baden-Württemberg  212837  
## 9 09      Bayern            231353  
## 10 10     Saarland           34672  
## 11 11     Berlin             168991  
## 12 12     Brandenburg         92648  
## 13 13     Mecklenburg-Vorpommern 70982  
## 14 14     Sachsen            140348  
## 15 15     Sachsen-Anhalt     96960  
## 16 16     Thüringen           68614
```

```
# Abspeichern als check_alo_bundesland
```

```
check_alo_bundesland <- data_alo %>%  
  filter(nchar(Regionalschlüssel) == 2) %>%  
  rename(bundesland = Regionalschlüssel)
```

```
check_alo_bundesland
```

```
## # A tibble: 16 x 3
##   bundesland Gemeinde      alo
##   <chr>      <chr>      <dbl>
## 1 01      Schleswig-Holstein  92434
## 2 02      Hamburg           69248
## 3 03      Niedersachsen     244260
## 4 04      Bremen             35687
## 5 05      Nordrhein-Westfalen 701219
## 6 06      Hessen            166286
## 7 07      Rheinland-Pfalz    106299
## 8 08      Baden-Württemberg  212837
## 9 09      Bayern            231353
## 10 10      Saarland           34672
## 11 11      Berlin            168991
## 12 12      Brandenburg        92648
## 13 13      Mecklenburg-Vorpommern 70982
## 14 14      Sachsen           140348
## 15 15      Sachsen-Anhalt     96960
## 16 16      Thüringen         68614
```

```
include_graphics("./figs/Alo_Laender.png")
```

Region	Bt	
	Insgesamt	
	absolut	Anteil in %
	1	2
Deutschland	2.532.837	100
Westdeutschland	1.894.294	74,8
Ostdeutschland	638.543	25,2
01 Schleswig-Holstein	92.434	3,6
02 Hamburg	69.248	2,7
03 Niedersachsen	244.260	9,6
04 Bremen	35.687	1,4
05 Nordrhein-Westfalen	701.219	27,7
06 Hessen	166.287	6,6
07 Rheinland-Pfalz	106.299	4,2
08 Baden-Württemberg	212.837	8,4
09 Bayern	231.353	9,1
10 Saarland	34.672	1,4
11 Berlin	168.991	6,7
12 Brandenburg	92.648	3,7
13 Mecklenburg-Vorpommern	70.982	2,8
14 Sachsen	140.348	5,5
15 Sachsen-Anhalt	96.960	3,8
16 Thüringen	68.614	2,7

```
check_alo_bundesland
```

```
## # A tibble: 16 x 3
##   bundesland Gemeinde      alo
##   <chr>      <chr>      <dbl>
## 1 01      Schleswig-Holstein  92434
## 2 02      Hamburg           69248
## 3 03      Niedersachsen     244260
## 4 04      Bremen             35687
## 5 05      Nordrhein-Westfalen 701219
## 6 06      Hessen            166286
## 7 07      Rheinland-Pfalz    106299
## 8 08      Baden-Württemberg  212837
## 9 09      Bayern            231353
## 10 10      Saarland           34672
## 11 11      Berlin            168991
## 12 12      Brandenburg        92648
## 13 13      Mecklenburg-Vorpommern 70982
## 14 14      Sachsen           140348
## 15 15      Sachsen-Anhalt     96960
## 16 16      Thüringen          68614
```

```
include_graphics("./figs/Alo_Laender.png")
```

Region	Bt	
	Insgesamt	
	absolut	Anteil in %
	1	2
Deutschland	2.532.837	100
Westdeutschland	1.894.294	74,8
Ostdeutschland	638.543	25,2
01 Schleswig-Holstein	92.434	3,6
02 Hamburg	69.248	2,7
03 Niedersachsen	244.260	9,6
04 Bremen	35.687	1,4
05 Nordrhein-Westfalen	701.219	27,7
06 Hessen	166.287	6,6
07 Rheinland-Pfalz	106.299	4,2
08 Baden-Württemberg	212.837	8,4
09 Bayern	231.353	9,1
10 Saarland	34.672	1,4
11 Berlin	168.991	6,7
12 Brandenburg	92.648	3,7
13 Mecklenburg-Vorpommern	70.982	2,8
14 Sachsen	140.348	5,5
15 Sachsen-Anhalt	96.960	3,8
16 Thüringen	68.614	2,7

Beide Datenreihen sind identisch (lediglich Hessen weicht um eine Person ab)

INTERNE KONSISTENZ ÜBERPRÜFEN

Berechne: Anzahl an Arbeitslosen für jedes Bundesland als Summe der Arbeitslosen einer Gemeinde.

```
# Nur Gemeindedaten nutzen, dann auf Bundeslandebende die Summe aus den Gemeindedaten berechnen
alo_meta <- data_alo %>%
  filter(nchar(Regionalschlüssel) == 8) %>%
  mutate(landkreis = str_extract(Regionalschlüssel, "^.{5}"),
         bundesland = str_extract(Regionalschlüssel, "^.{2}"))

alo_bundesland <- alo_meta %>%
  group_by(bundesland) %>%
  summarise(total_alo = sum(alo))

alo_landkreis <- alo_meta %>%
  group_by(landkreis) %>%
  summarise(total_alo = sum(alo)) %>%
  rename(Regionalschlüssel = landkreis)
```

data_alo

```
## # A tibble: 11,343 x 3
##   Regionalschlüssel Gemeinde      alo
##   <chr>                <chr>    <dbl>
## 1 01                    Schleswig-Holstein 92434
## 2 01001000             Flensburg, Stadt  4512
## 3 01002000             Kiel, Landeshauptstadt 12345
## 4 01003000             Lübeck, Hansestadt  9692
## 5 01004000             Neumünster, Stadt  3836
## 6 01051                Dithmarschen      4628
## 7 01051001            Albersdorf         150
## 8 01051002            Arkebek            5
## 9 01051003            Averlak            15
## 10 01051004            Bargenstedt       17
## # ... with 11,333 more rows
```

```
data_alo %>%
```

```
  filter(nchar(Regionalschlüssel) == 8)
```

```
## # A tibble: 11,010 x 3
```

```
##   Regionalschlüssel Gemeinde      alo
##   <chr>              <chr>      <dbl>
## 1 01001000           Flensburg, Stadt    4512
## 2 01002000           Kiel, Landeshauptstadt 12345
## 3 01003000           Lübeck, Hansestadt   9692
## 4 01004000           Neumünster, Stadt   3836
## 5 01051001           Albersdorf          150
## 6 01051002           Arkebek              5
## 7 01051003           Averlak              15
## 8 01051004           Bargenstedt         17
## 9 01051005           Barkenholm           3
## 10 01051006           Barlt                21
## # ... with 11,000 more rows
```

```
data_alo %>%
  filter(nchar(Regionalschlüssel) == 8) %>%
  mutate(landkreis = str_extract(Regionalschlüssel,
```

```
## # A tibble: 11,010 x 4
##   Regionalschlüssel Gemeinde      alo landkreis
##   <chr>              <chr>      <dbl> <chr>
## 1 01001000          Flensburg, Stadt  4512 01001
## 2 01002000          Kiel, Landeshauptstadt 12345 01002
## 3 01003000          Lübeck, Hansestadt  9692 01003
## 4 01004000          Neumünster, Stadt  3836 01004
## 5 01051001          Albersdorf        150 01051
## 6 01051002          Arkebek            5 01051
## 7 01051003          Averlak            15 01051
## 8 01051004          Bargenstedt       17 01051
## 9 01051005          Barkenholm         3 01051
## 10 01051006          Barlt             21 01051
## # ... with 11,000 more rows
```

```
data_alo %>%
  filter(nchar(Regionalschlüssel) == 8) %>%
  mutate(landkreis = str_extract(Regionalschlüssel,
  mutate(bundesland = str_extract(Regionalschlüssel
```

```
## # A tibble: 11,010 x 5
##   Regionalschlüssel Gemeinde      alo landkreis bundesland
##   <chr>                <chr>      <dbl> <chr>      <chr>
## 1 01001000            Flensburg, Stadt      4512 01001      01
## 2 01002000            Kiel, Landeshauptstadt 12345 01002      01
## 3 01003000            Lübeck, Hansestadt    9692 01003      01
## 4 01004000            Neumünster, Stadt    3836 01004      01
## 5 01051001            Albersdorf            150 01051      01
## 6 01051002            Arkebek                5 01051      01
## 7 01051003            Averlak                15 01051      01
## 8 01051004            Bargenstedt           17 01051      01
## 9 01051005            Barkenholm             3 01051      01
## 10 01051006            Barlt                  21 01051      01
## # ... with 11,000 more rows
```

```
data_alo %>%  
  filter(nchar(Regionalschlüssel) == 8) %>%  
  mutate(landkreis = str_extract(Regionalschlüssel,  
  mutate(bundesland = str_extract(Regionalschlüssel,  
alo_meta
```

alo_meta

```
## # A tibble: 11,010 x 5
##   Regionalschlüssel Gemeinde      alo landkreis bundesland
##   <chr>                <chr>      <dbl> <chr>      <chr>
## 1 01001000            Flensburg, Stadt    4512 01001      01
## 2 01002000            Kiel, Landeshauptstadt 12345 01002      01
## 3 01003000            Lübeck, Hansestadt   9692 01003      01
## 4 01004000            Neumünster, Stadt   3836 01004      01
## 5 01051001            Albersdorf          150 01051      01
## 6 01051002            Arkebek              5 01051      01
## 7 01051003            Averlak              15 01051      01
## 8 01051004            Bargenstedt         17 01051      01
## 9 01051005            Barkenholm           3 01051      01
## 10 01051006            Barlt                21 01051      01
## # ... with 11,000 more rows
```

```
alo_meta %>%
```

```
  group_by(bundesland)
```

```
## # A tibble: 11,010 x 5
## # Groups:   bundesland [16]
##   Regionalschlüssel Gemeinde      alo landkreis bundesland
##   <chr>                <chr>      <dbl> <chr>    <chr>
## 1 01001000      Flensburg, Stadt    4512 01001    01
## 2 01002000      Kiel, Landeshauptstadt 12345 01002    01
## 3 01003000      Lübeck, Hansestadt   9692 01003    01
## 4 01004000      Neumünster, Stadt   3836 01004    01
## 5 01051001      Albersdorf          150 01051    01
## 6 01051002      Arkebek              5 01051    01
## 7 01051003      Averlak              15 01051    01
## 8 01051004      Bargaenstedt         17 01051    01
## 9 01051005      Barkenholm           3 01051    01
## 10 01051006      Barlt                21 01051    01
## # ... with 11,000 more rows
```



```
alo_meta %>%  
  group_by(bundesland) %>%  
  summarise(total_alo = sum(alo))
```

```
## # A tibble: 16 x 2  
##   bundesland total_alo  
##   <chr>         <dbl>  
## 1 01             92449  
## 2 02             69248  
## 3 03            244277  
## 4 04             35687  
## 5 05            701212  
## 6 06            166296  
## 7 07            106287  
## 8 08            212835  
## 9 09            231355  
## 10 10            34675  
## 11 11            168991  
## 12 12            92644  
## 13 13            70989  
## 14 14            140348  
## 15 15            96960  
## 16 16            68609
```

```
alo_meta %>%  
  group_by(bundesland) %>%  
  summarise(total_alo = sum(alo)) ->  
alo_bundesland
```

alo_meta

```
## # A tibble: 11,010 x 5
##   Regionalschlüssel Gemeinde      alo landkreis bundesland
##   <chr>                <chr>      <dbl> <chr>      <chr>
## 1 01001000      Flensburg, Stadt    4512 01001      01
## 2 01002000      Kiel, Landeshauptstadt 12345 01002      01
## 3 01003000      Lübeck, Hansestadt   9692 01003      01
## 4 01004000      Neumünster, Stadt   3836 01004      01
## 5 01051001      Albersdorf          150 01051      01
## 6 01051002      Arkebek              5 01051      01
## 7 01051003      Averlak              15 01051      01
## 8 01051004      Bargenstedt         17 01051      01
## 9 01051005      Barkenholm           3 01051      01
## 10 01051006      Barlt                21 01051      01
## # ... with 11,000 more rows
```

```
alo_meta %>%
```

```
  group_by(landkreis)
```

```
## # A tibble: 11,010 x 5
## # Groups:   landkreis [401]
##   Regionalschlüssel Gemeinde          alo landkreis bundesland
##   <chr>                <chr>          <dbl> <chr>    <chr>
## 1 01001000             Flensburg, Stadt    4512 01001    01
## 2 01002000             Kiel, Landeshauptstadt 12345 01002    01
## 3 01003000             Lübeck, Hansestadt   9692 01003    01
## 4 01004000             Neumünster, Stadt   3836 01004    01
## 5 01051001             Albersdorf          150 01051    01
## 6 01051002             Arkebek              5 01051    01
## 7 01051003             Averlak              15 01051    01
## 8 01051004             Bargaenstedt        17 01051    01
## 9 01051005             Barkenholm           3 01051    01
## 10 01051006            Barlt                21 01051    01
## # ... with 11,000 more rows
```

```
alo_meta %>%  
  group_by(landkreis) %>%  
  summarise(total_alo = sum(alo))
```

```
## # A tibble: 401 x 2  
##   landkreis total_alo  
##   <chr>         <dbl>  
## 1 01001           4512  
## 2 01002          12345  
## 3 01003           9692  
## 4 01004           3836  
## 5 01051           4632  
## 6 01053           5592  
## 7 01054           5657  
## 8 01055           5748  
## 9 01056           8599  
## 10 01057          3264  
## # ... with 391 more rows
```

```
alo_meta %>%
  group_by(landkreis) %>%
  summarise(total_alo = sum(alo)) %>%
  rename(Regionalschluessel = landkreis)
```

```
## # A tibble: 401 x 2
##   Regionalschluessel total_alo
##   <chr>                <dbl>
## 1 01001                  4512
## 2 01002                 12345
## 3 01003                  9692
## 4 01004                  3836
## 5 01051                  4632
## 6 01053                  5592
## 7 01054                  5657
## 8 01055                  5748
## 9 01056                  8599
## 10 01057                 3264
## # ... with 391 more rows
```

```
alo_meta %>%  
  group_by(landkreis) %>%  
  summarise(total_alo = sum(alo)) %>%  
  rename(Regionalschlüssel = landkreis) ->  
alo_landkreis
```

INTERNE KONSISTENZ ÜBERPRÜFEN

Wir wollen nun die zwei Tabellen miteinander verbinden (besserer Überblick)

- + Datensatz `check_alo_bundesland`: Auf Bundesland aggregierte Zahlen der Arbeitslosigkeit aus den Gemeinden
- + Datensatz `alo_bundesland`: Die schon von der Arbeitsagentur aggregierte Zahlen in unserem Datensatz


```
left_join(check_alo_bundesland, alo_bundesland, by =
```

```
## # A tibble: 16 x 4
##   bundesland Gemeinde      alo total_alo
##   <chr>      <chr>      <dbl>    <dbl>
## 1 01      Schleswig-Holstein  92434    92449
## 2 02      Hamburg          69248    69248
## 3 03      Niedersachsen     244260   244277
## 4 04      Bremen            35687    35687
## 5 05      Nordrhein-Westfalen 701219   701212
## 6 06      Hessen            166286   166296
## 7 07      Rheinland-Pfalz    106299   106287
## 8 08      Baden-Württemberg  212837   212835
## 9 09      Bayern            231353   231355
## 10 10     Saarland           34672    34675
## 11 11     Berlin            168991   168991
## 12 12     Brandenburg        92648    92644
## 13 13     Mecklenburg-Vorpommern 70982    70989
## 14 14     Sachsen            140348   140348
## 15 15     Sachsen-Anhalt     96960    96960
## 16 16     Thüringen          68614    68609
```

```
left_join(check_alo_bundesland, alo_bundesland, by =  
  check_consistency
```

```
left_join(check_alo_bundesland, alo_bundesland, by =  
  check_consistency
```

```
check_consistency
```

```
## # A tibble: 16 x 4  
##   bundesland Gemeinde      alo total_alo  
##   <chr>      <chr>      <dbl>    <dbl>  
## 1 01      Schleswig-Holstein  92434    92449  
## 2 02      Hamburg          69248    69248  
## 3 03      Niedersachsen     244260   244277  
## 4 04      Bremen            35687    35687  
## 5 05      Nordrhein-Westfalen 701219   701212  
## 6 06      Hessen            166286   166296  
## 7 07      Rheinland-Pfalz    106299   106287  
## 8 08      Baden-Württemberg  212837   212835  
## 9 09      Bayern            231353   231355  
## 10 10     Saarland           34672    34675  
## 11 11     Berlin            168991   168991  
## 12 12     Brandenburg         92648    92644  
## 13 13     Mecklenburg-Vorpommern 70982    70989  
## 14 14     Sachsen            140348   140348  
## 15 15     Sachsen-Anhalt     96960    96960  
## 16 16     Thüringen          68614    68609
```

```
left_join(check_alo_bundesland, alo_bundesland, by =
  check_consistency
check_consistency %>%
  mutate(diff = alo - total_alo)
```

```
## # A tibble: 16 x 5
##   bundesland Gemeinde      alo total_alo  diff
##   <chr>      <chr>      <dbl>    <dbl> <dbl>
## 1 01      Schleswig-Holstein  92434    92449   -15
## 2 02      Hamburg          69248    69248    0
## 3 03      Niedersachsen    244260   244277   -17
## 4 04      Bremen            35687    35687    0
## 5 05      Nordrhein-Westfalen 701219   701212    7
## 6 06      Hessen            166286   166296   -10
## 7 07      Rheinland-Pfalz   106299   106287    12
## 8 08      Baden-Württemberg 212837   212835    2
## 9 09      Bayern            231353   231355   -2
## 10 10     Saarland          34672    34675   -3
## 11 11     Berlin            168991   168991    0
## 12 12     Brandenburg        92648    92644    4
## 13 13     Mecklenburg-Vorpommern 70982    70989   -7
## 14 14     Sachsen            140348   140348    0
## 15 15     Sachsen-Anhalt     96960    96960    0
## 16 16     Thüringen          68614    68609    5
```

```
left_join(check_alo_bundesland, alo_bundesland, by =  
  check_consistency  
  
check_consistency %>%  
  mutate(diff = alo - total_alo)
```

```
## # A tibble: 16 x 5  
##   bundesland Gemeinde      alo total_alo diff  
##   <chr>      <chr>      <dbl>    <dbl> <dbl>  
## 1 01      Schleswig-Holstein  92434    92449  -15  
## 2 02      Hamburg          69248    69248   0  
## 3 03      Niedersachsen    244260   244277  -17  
## 4 04      Bremen           35687    35687   0  
## 5 05      Nordrhein-Westfalen 701219   701212   7  
## 6 06      Hessen           166286   166296  -10  
## 7 07      Rheinland-Pfalz   106299   106287  12  
## 8 08      Baden-Württemberg 212837   212835   2  
## 9 09      Bayern           231353   231355  -2  
## 10 10     Saarland         34672    34675  -3  
## 11 11     Berlin           168991   168991   0  
## 12 12     Brandenburg       92648    92644   4  
## 13 13     Mecklenburg-Vorpommern 70982    70989  -7  
## 14 14     Sachsen           140348   140348   0  
## 15 15     Sachsen-Anhalt    96960    96960   0  
## 16 16     Thüringen         68614    68609   5
```

Es bestehen kleinere Unstimmigkeiten, jedoch sind diese marginal.

Pro-Kopf Verschuldung

Pro-Kopf Verschuldung auf Gemeindeebene

- + Auf Gemeindeebene aus dem Jahr 2017
- + Querschnittsdaten
- + Vom Statistischen Bundesamt direkt als Excel-Tabelle heruntergeladen (✓)

Welche Tabellenblätter sollten wir nutzen?

```
excel_sheets("./data/Schulden_2017.xlsx")
```

```
## [1] "Titel"           "Impressum"      "Inhalt"
## [4] "Abkürzungen"    "Erläuterungen" "SH"
## [7] "NI"             "NW"             "HE"
## [10] "RP"            "BW"             "BY"
## [13] "SL"            "BB"             "MV"
## [16] "SN"            "ST"             "TH"
## [19] "Statistische Ämter"
```

Mehrere Tabellenblätter einlesen

- + Nicht alle Informationen in **einem Tabellenblatt** enthalten
 - + Viele separate Tabellenblätter
 - + Hier kommt die `for`-Schleife zum Einsatz

Mehrere Tabellenblätter einlesen

- + Nicht alle Informationen in **einem Tabellenblatt** enthalten
 - + Viele separate Tabellenblätter
 - + Hier kommt die `for`-Schleife zum Einsatz

Zuerst schauen wir jedoch welche Informationen wir benötigen anhand eines Beispiels:

Mehrere Tabellenblätter einlesen

```
sh <- read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH")
head(sh, 20)
```

```
## # A tibble: 20 x 16
##   `Zurück zum Inh...` ...2 ...3 ...4 ...5 ...6 ...7 ...8 ...9 ...10 ...11
##   <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 2 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 3 "Tabelle 1: ... <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 4 "nach Höhe der ... <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 5 "Regional-\r\ns... Geme... Verw... "Ein... Schu... "Sch... Schu... <NA> <NA> <NA> <NA>
## 6 <NA> <NA> <NA> <NA> <NA> zusa... Schu... ante... <NA> <NA>
## 7 <NA> <NA> <NA> <NA> <NA> <NA> <NA> zusa... davo... <NA>
## 8 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> 100% 50% ...
## 9 <NA> <NA> <NA> <NA> EUR <NA> <NA> <NA> <NA> <NA> <NA>
## 10 <NA> <NA> <NA> <NA> 1 "2" 3 4 5 6 7
## 11 "010010000000" Flen... krei... "877... 5082... "579... 2979... 1126... 1853... 1822... 2906...
## 12 "010020000000" Kiel... krei... "247... 9488... "383... 5632... 5626... 5701... 4437... 0
## 13 "010030000000" Lübe... krei... "216... 1206... "556... 6014... 5938... 7606... 7027... 0
## 14 "010040000000" Neum... krei... "787... 4260... "540... 1233... 1152... 8176... 8103... 0
## 15 "01051" Krei... Krei... "{13... 6517... "487... 4309... 4302... 72975 0 0
```

Mehrere Tabellenblätter einlesen

Wir benötigen:

- + "Regionalschlüssel"
- + "Gemeindename"
- + "Einwohner"
- + "Schuldes des öffentlichen Bereichs insgesamt"
- + "Schulden je Einwohner"

Variablenbezeichnungen beginnen in Zeile 5, d.h. wir ignorieren die ersten 4 Zeilen beim Einlesen.

Was ist hier eine Beobachtung?

Mehrere Tabellenblätter einlesen

Der Übersicht halber wollen wir noch eine Spalte hinzufügen, welche den Namen des Tabellenblattes enthält, welches wir gerade eingelesen haben.

```
# Einlesen des Tabellenblattes "SH" ohne die ersten 5 Zeilen und nur die Spalten 1-6
schulden_individuell <- read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH", skip = 5)[1:6]
# Umbenennen der ersten 6 Spalten
colnames(schulden_individuell) <- c("Regionalschlüssel", "Gemeinde",
                                   "Verwaltungsform", "Einwohner", "Schulden_gesamt", "Schulden_pro_kopf")

# Zusätzliche Spalte hinzufügen mit dem Namen des Tabellenblattes
schulden_individuell$Bundesland <- "SH"
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten
read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH",
```

```
## # A tibble: 1,312 x 6
##   `Regional-\r\ns...` `Gemeinde/Gemei...` `Verwaltungsform` `Einwohner/in\r...`
##   <chr>                <chr>                <chr>                <chr>
## 1 <NA>                 <NA>                 <NA>                 <NA>
## 2 <NA>                 <NA>                 <NA>                 <NA>
## 3 <NA>                 <NA>                 <NA>                 <NA>
## 4 <NA>                 <NA>                 <NA>                 <NA>
## 5 <NA>                 <NA>                 <NA>                 <NA>
## 6 010010000000        Flensburg, Stadt kreisfreie Sta... 87770
## 7 010020000000        Kiel, Landeshau... kreisfreie Sta... 247135
## 8 010030000000        Lübeck, Hansest... kreisfreie Sta... 216739
## 9 010040000000        Neumünster, Sta... kreisfreie Sta... 78759
## 10 01051                Kreisverwaltung... Kreisverwaltung {133 684}
## # ... with 1,302 more rows, and 2 more variables: `Schulden des öffentlichen
## # Bereichs insgesamt` <chr>, `Schulden je \r\nEinwohner/in` <chr>
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten  
read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH",  
schulden_individuell
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten
read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH",
  schulden_individuell

# Umbenennen der ersten 6 Spalten
colnames(schulden_individuell) <- c("Regionalschlues
  "Verwaltungsform
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten
read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH",
  schulden_individuell

# Umbenennen der ersten 6 Spalten
colnames(schulden_individuell) <- c("Regionalschlues
  "Verwaltungsform

# Zusätzliche Spalte hinzufügen mit dem Namen des Ta
schulden_individuell$Bundesland <- "SH"
```



```

# Einlesen des Tabellenblattes "SH" ohne die ersten
read_xlsx("./data/Schulden_2017.xlsx", sheet = "SH",
  schulden_individuell

# Umbenennen der ersten 6 Spalten
colnames(schulden_individuell) <- c("Regionalschlues
  "Verwaltungsform

# Zusätzliche Spalte hinzufügen mit dem Namen des Ta
schulden_individuell$Bundesland <- "SH"

schulden_individuell

```

```

## # A tibble: 1,312 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>      <chr>
## 1 <NA>             <NA>      <NA>           <NA>      <NA>
## 2 <NA>             <NA>      <NA>           <NA>      <NA>
## 3 <NA>             <NA>      <NA>           <NA>      <NA>
## 4 <NA>             <NA>      <NA>           <NA>      EUR
## 5 <NA>             <NA>      <NA>           <NA>      1
## 6 010010000000    Flensbu... kreisfreie Sta... 87770    508281539
## 7 010020000000    Kiel, L... kreisfreie Sta... 247135   948848421
## 8 010030000000    Lübeck,... kreisfreie Sta... 216739   1206620094
## 9 010040000000    Neumüns... kreisfreie Sta... 78759    426019276
## 10 01051           Kreisve... Kreisverwaltung {133 684} 65179097
## # ... with 1,302 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>

```

Mehrere Tabellenblätter einlesen

Nun können wir genauso bei allen anderen Tabellenblättern vorgehen:

Mehrere Tabellenblätter einlesen

Nun können wir genauso bei allen anderen Tabellenblättern vorgehen:

```
# Daten mit for-Schleife einlesen (Struktur gleich wie im vorherigen Chunk)
sheet_names <- excel_sheets("./data/Schulden_2017.xlsx")
# Einlesen der Tabellenblätter 7-18 (alle Bundesländer)
sheet_read <- sheet_names[7:18]
for (i in 1:length(sheet_read)){
  tmp <- read_xlsx("./data/Schulden_2017.xlsx", sheet = sheet_read[i], skip = 5)[1:6]
  tmp$Bundesland <- sheet_read[i]
  colnames(tmp) <- c("Regionalschlüssel", "Gemeinde", "Verwaltungsform",
                    "Einwohner", "Schulden_gesamt", "Schulden_pro_kopf", "Bundesland")
  # Daten aller weiteren Tabellenblätter unter den aktuellen Datensatz anheften
  schulden_individuell <- bind_rows(schulden_individuell, tmp)
}
```

Mehrere Tabellenblätter einlesen

Nun können wir genauso bei allen anderen Tabellenblättern vorgehen:

```
# Daten mit for-Schleife einlesen (Struktur gleich wie im vorherigen Chunk)
sheet_names <- excel_sheets("./data/Schulden_2017.xlsx")
# Einlesen der Tabellenblätter 7-18 (alle Bundesländer)
sheet_read <- sheet_names[7:18]
for (i in 1:length(sheet_read)){
  tmp <- read_xlsx("./data/Schulden_2017.xlsx", sheet = sheet_read[i], skip = 5)[1:6]
  tmp$Bundesland <- sheet_read[i]
  colnames(tmp) <- c("Regionalschlüssel", "Gemeinde", "Verwaltungsform",
                    "Einwohner", "Schulden_gesamt", "Schulden_pro_kopf", "Bundesland")
  # Daten aller weiteren Tabellenblätter unter den aktuellen Datensatz anheften
  schulden_individuell <- bind_rows(schulden_individuell, tmp)
}
```

Mehrere Tabellenblätter einlesen

Nun können wir genauso bei allen anderen Tabellenblättern vorgehen:

```
# Daten mit for-Schleife einlesen (Struktur gleich wie im vorherigen Chunk)
sheet_names <- excel_sheets("./data/Schulden_2017.xlsx")
# Einlesen der Tabellenblätter 7-18 (alle Bundesländer)
sheet_read <- sheet_names[7:18]
for (i in 1:length(sheet_read)){
  tmp <- read_xlsx("./data/Schulden_2017.xlsx", sheet = sheet_read[i], skip = 5)[1:6]
  tmp$Bundesland <- sheet_read[i]
  colnames(tmp) <- c("Regionalschlüssel", "Gemeinde", "Verwaltungsform",
                    "Einwohner", "Schulden_gesamt", "Schulden_pro_kopf", "Bundesland")
  # Daten aller weiteren Tabellenblätter unter den aktuellen Datensatz anheften
  schulden_individuell <- bind_rows(schulden_individuell, tmp)
}
```

Eine zusätzliche Spalte generieren, welche die Information pro Bundesland enthält

```
# Daten mit for-Schleife einlesen (Struktur gleich w
```

```
# Daten mit for-Schleife einlesen (Struktur gleich w  
excel_sheets("./data/Schulden_2017.xlsx")
```

```
## [1] "Titel" "Impressum" "Inhalt"  
## [4] "Abkürzungen" "Erläuterungen" "SH"  
## [7] "NI" "NW" "HE"  
## [10] "RP" "BW" "BY"  
## [13] "SL" "BB" "MV"  
## [16] "SN" "ST" "TH"  
## [19] "Statistische Ämter"
```

```
# Daten mit for-Schleife einlesen (Struktur gleich w  
excel_sheets("./data/Schulden_2017.xlsx") ->  
sheet_names
```



```
# Daten mit for-Schleife einlesen (Struktur gleich w  
excel_sheets("./data/Schulden_2017.xlsx") ->  
  sheet_names
```

```
# Einlesen der Tabellenblätter 7-18 (alle Bundesländ
```

```
# Daten mit for-Schleife einlesen (Struktur gleich wie  
excel_sheets("./data/Schulden_2017.xlsx") ->  
  sheet_names
```

```
# Einlesen der Tabellenblätter 7-18 (alle Bundesländer)  
sheet_names[7:18]
```

```
## [1] "NI" "NW" "HE" "RP" "BW" "BY" "SL" "BB" "MV" "SN" "ST" "TH"
```

```
# Daten mit for-Schleife einlesen (Struktur gleich w  
excel_sheets("./data/Schulden_2017.xlsx") ->  
  sheet_names  
  
# Einlesen der Tabellenblätter 7-18 (alle Bundesländ  
sheet_names[7:18] ->  
  sheet_read
```

```
# Daten mit for-Schleife einlesen (Struktur gleich w  
excel_sheets("./data/Schulden_2017.xlsx") ->  
  sheet_names  
  
# Einlesen der Tabellenblätter 7-18 (alle Bundesländ  
sheet_names[7:18] ->  
  sheet_read  
  
length(sheet_read)
```

```
## [1] 12
```

```
# Daten mit for-Schleife einlesen (Struktur gleich w
excel_sheets("./data/Schulden_2017.xlsx") ->
  sheet_names

# Einlesen der Tabellenblätter 7-18 (alle Bundesländ
sheet_names[7:18] ->
  sheet_read

length(sheet_read)
```

```
for (i in 1:length(sheet_read)){
  tmp <- read_xlsx("./data/Schulden_2017.xlsx", shee
  tmp$Bundesland <- sheet_read[i]
  colnames(tmp) <- c("Regionalschlüssel", "Gemeinde
                    "Einwohner", "Schulden_gesamt",
# Daten aller weiteren Tabellenblätter unter den akt
  schulden_individuell <- bind_rows(schulden_individ
}
```

```
## [1] 12
```

```

# Daten mit for-Schleife einlesen (Struktur gleich w
excel_sheets("./data/Schulden_2017.xlsx") ->
  sheet_names

# Einlesen der Tabellenblätter 7-18 (alle Bundesländ
sheet_names[7:18] ->
  sheet_read

length(sheet_read)

for (i in 1:length(sheet_read)){
  tmp <- read_xlsx("./data/Schulden_2017.xlsx", shee
  tmp$Bundesland <- sheet_read[i]
  colnames(tmp) <- c("Regionalschluessel", "Gemeinde
    "Einwohner", "Schulden_gesamt",
# Daten aller weiteren Tabellenblätter unter den akt
schulden_individuell <- bind_rows(schulden_individ
}

```

```
schulden_individuell
```

```
## [1] 12
```

```
## # A tibble: 25,796 x 7
```

```

##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>      <chr>
## 1 <NA>              <NA>      <NA>           <NA>      <NA>
## 2 <NA>              <NA>      <NA>           <NA>      <NA>
## 3 <NA>              <NA>      <NA>           <NA>      <NA>
## 4 <NA>              <NA>      <NA>           <NA>      EUR
## 5 <NA>              <NA>      <NA>           <NA>      1
## 6 010010000000     Flensbu... kreisfreie Sta... 87770      508281539
## 7 010020000000     Kiel, L... kreisfreie Sta... 247135     948848421
## 8 010030000000     Lübeck,... kreisfreie Sta... 216739     1206620094
## 9 010040000000     Neumüns... kreisfreie Sta... 78759      426019276
## 10 01051            Kreisve... Kreisverwaltung {133 684} 65179097
## # ... with 25,786 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>

```

Variablen umformen

```
head(schulden_individuell, 15)
```

```
## # A tibble: 15 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>             <chr>      <chr>
## 1 <NA>             <NA>      <NA>             <NA>      <NA>
## 2 <NA>             <NA>      <NA>             <NA>      <NA>
## 3 <NA>             <NA>      <NA>             <NA>      <NA>
## 4 <NA>             <NA>      <NA>             <NA>      EUR
## 5 <NA>             <NA>      <NA>             <NA>      1
## 6 010010000000    Flensbu... kreisfreie Sta... 87770      508281539
## 7 010020000000    Kiel, L... kreisfreie Sta... 247135     948848421
## 8 010030000000    Lübeck,... kreisfreie Sta... 216739     1206620094
## 9 010040000000    Neumüns... kreisfreie Sta... 78759      426019276
## 10 01051           Kreisve... Kreisverwaltung {133 684} 65179097
## 11 010510011011    Brunsbü... amtsfreie Geme... 12781      34934876
## 12 010510044044    Heide, ... amtsfreie Geme... 21508      37056249
## 13 010515163       Amtsver... Amtsverwaltung  {15 653} 1047175
## 14 010515163003    Averlak  amtsangehörige... 576        1197761
## 15 010515163010    Brickeln amtsangehörige... 219        496264
## # ... with 2 more variables: Schulden_pro_kopf <chr>, Bundesland <chr>
```

Variablen umformen

Wir sehen, es gibt immer noch einige Probleme:

- + Die Werte unserer Variablen stehen nicht direkt unter dem Variablennamen
 - + Dies können wir am einfachsten bereinigen indem wir alle `NA`s im Regionalschlüssel entfernen

Variablen umformen

Wir sehen, es gibt immer noch einige Probleme:

- ✚ Die Werte unserer Variablen stehen nicht direkt unter dem Variablennamen
 - ✚ Dies können wir am einfachsten bereinigen indem wir alle `NA`s im Regionalschlüssel entfernen
- ✚ Die Variablen "Einwohner", "Schulden_gesamt" und "Schulden_pro_Kopf" sind alle als `character` hinterlegt (`<chr>` unter dem Variablennamen in der vorherigen Tabelle)
 - ✚ Beispiel warum Klasse `character` (Zeile 28): Es sind geschweifte Klammern enthalten

```
schulden_individuell[28,]
```

```
## # A tibble: 1 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>      <chr>
## 1 010515163_Summe... Burg-Sa... Amtsgebiet     {15 653}  {22 848 088}
## # ... with 2 more variables: Schulden_pro_kopf <chr>, Bundesland <chr>
```

Variablen umformen

Wir sehen, es gibt immer noch einige Probleme:

- ✚ Die Werte unserer Variablen stehen nicht direkt unter dem Variablennamen
 - ✚ Dies können wir am einfachsten bereinigen indem wir alle `NA`s im Regionalschlüssel entfernen
- ✚ Die Variablen "Einwohner", "Schulden_gesamt" und "Schulden_pro_Kopf" sind alle als `character` hinterlegt (`<chr>` unter dem Variablennamen in der vorherigen Tabelle)
 - ✚ Beispiel warum Klasse `character` (Zeile 28): Es sind geschweifte Klammern enthalten

```
schulden_individuell[28,]
```

```
## # A tibble: 1 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>      <chr>
## 1 010515163_Summe... Burg-Sa... Amtsgebiet     {15 653}  {22 848 088}
## # ... with 2 more variables: Schulden_pro_kopf <chr>, Bundesland <chr>
```

- ✚ Definition einer Variablen `landkreis`: Ersten 5 Zeichen im Regionalschlüssel

```
# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezogen
schulden_individuell
```

```
## # A tibble: 13,554 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>      <chr>
## 1 <NA>             <NA>      <NA>           <NA>      <NA>
## 2 <NA>             <NA>      <NA>           <NA>      <NA>
## 3 <NA>             <NA>      <NA>           <NA>      <NA>
## 4 <NA>             <NA>      <NA>           <NA>      EUR
## 5 <NA>             <NA>      <NA>           <NA>      1
## 6 010010000000    Flensbu... kreisfreie Sta... 87770    508281539
## 7 010020000000    Kiel, L... kreisfreie Sta... 247135   948848421
## 8 010030000000    Lübeck,... kreisfreie Sta... 216739   1206620094
## 9 010040000000    Neumüns... kreisfreie Sta... 78759    426019276
## 10 01051           Kreisve... Kreisverwaltung {133 684} 65179097
## # ... with 13,544 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>
```

```
# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
```

```
filter(!is.na(Regionalschlüssel))
```

```
## # A tibble: 13,450 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>           <chr>    <chr>
## 1 0100100000000    Flensbu... kreisfreie Sta... 87770    508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta... 247135   948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta... 216739   1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta... 78759    426019276
## 5 01051           Kreisve... Kreisverwaltung {133 684} 65179097
## 6 010510011011    Brunsbü... amtsfreie Geme... 12781    34934876
## 7 010510044044    Heide, ... amtsfreie Geme... 21508    37056249
## 8 010515163       Amtsver... Amtsverwaltung {15 653} 1047175
## 9 010515163003    Averlak  amtsangehörige... 576      1197761
## 10 010515163010    Brickeln amtsangehörige... 219      496264
## # ... with 13,440 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>
```

```
# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezogen
schulden_individuell %>%
```

```
  filter(!is.na(Regionalschlüssel)) %>%
```

```
  mutate(Schulden_gesamt = as.numeric(Schulden_gesamt))
```

```
## # A tibble: 13,450 x 7
```

```
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
```

```
##   <chr>             <chr>    <chr>           <chr>           <dbl>
```

```
## 1 0100100000000    Flensbu... kreisfreie Sta... 87770           508281539
```

```
## 2 0100200000000    Kiel, L... kreisfreie Sta... 247135          948848421
```

```
## 3 0100300000000    Lübeck,... kreisfreie Sta... 216739          1206620094
```

```
## 4 0100400000000    Neumüns... kreisfreie Sta... 78759           426019276
```

```
## 5 01051          Kreisve... Kreisverwaltung {133 684}      65179097
```

```
## 6 010510011011    Brunsbü... amtsfreie Geme... 12781           34934876
```

```
## 7 010510044044    Heide, ... amtsfreie Geme... 21508           37056249
```

```
## 8 010515163      Amtsver... Amtsverwaltung {15 653}      1047175
```

```
## 9 010515163003    Averlak  amtsangehörige... 576             1197761
```

```
## 10 010515163010    Brickeln amtsangehörige... 219             496264
```

```
## # ... with 13,440 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
```

```
## #   Bundesland <chr>
```

```
# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschlüssel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
mutate(Einwohner = as.numeric(Einwohner))
```

```
## # A tibble: 13,450 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>             <dbl>         <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta... 87770         508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta... 247135        948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta... 216739        1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta... 78759         426019276
## 5 01051           Kreisve... Kreisverwaltung    NA            65179097
## 6 010510011011    Brunsbü... amtsfreie Geme... 12781         34934876
## 7 010510044044    Heide, ... amtsfreie Geme... 21508         37056249
## 8 010515163       Amtsver... Amtsverwaltung    NA            1047175
## 9 010515163003    Averlak  amtsangehörige... 576           1197761
## 10 010515163010    Brickeln amtsangehörige... 219           496264
## # ... with 13,440 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>
```

```

# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschlüssel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
  mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro

```

```

## # A tibble: 13,450 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>                <dbl>         <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta...   87770         508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta...  247135         948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta...  216739         1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta...   78759         426019276
## 5 01051           Kreisve... Kreisverwaltung      NA             65179097
## 6 010510011011    Brunsbü... amtsfreie Geme...  12781          34934876
## 7 010510044044    Heide, ... amtsfreie Geme...  21508          37056249
## 8 010515163       Amtsver... Amtsverwaltung      NA             1047175
## 9 010515163003    Averlak   amtsangehörige...   576            1197761
## 10 010515163010    Brickeln amtsangehörige...  219             496264
## # ... with 13,440 more rows, and 2 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>

```

```

# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschlüssel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
  mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro
  mutate(landkreis = str_extract(Regionalschlüssel,

```

```

## # A tibble: 13,450 x 8
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>                <dbl>         <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta...   87770         508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta...  247135         948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta...  216739         1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta...   78759         426019276
## 5 01051           Kreisve... Kreisverwaltung      NA             65179097
## 6 010510011011    Brunsbü... amtsfreie Geme...   12781          34934876
## 7 010510044044    Heide, ... amtsfreie Geme...   21508          37056249
## 8 010515163       Amtsver... Amtsverwaltung      NA             1047175
## 9 010515163003    Averlak   amtsangehörige...    576            1197761
## 10 010515163010    Brickeln amtsangehörige...   219            496264
## # ... with 13,440 more rows, and 3 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, landkreis <chr>

```



```

# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschluessel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro
mutate(landkreis = str_extract(Regionalschluessel,
#bei der Gemeinde Selent wurde in der Excel Tabelle
mutate(landkreis = ifelse(landkreis == "10575", "0

```

```

## # A tibble: 13,450 x 8
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>             <dbl>         <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta... 87770         508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta... 247135        948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta... 216739        1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta... 78759         426019276
## 5 01051           Kreisve... Kreisverwaltung    NA            65179097
## 6 010510011011    Brunsbü... amtsfreie Geme... 12781         34934876
## 7 010510044044    Heide, ... amtsfreie Geme... 21508         37056249
## 8 010515163       Amtsver... Amtsverwaltung    NA            1047175
## 9 010515163003    Averlak  amtsangehörige... 576           1197761
## 10 010515163010    Brickeln amtsangehörige... 219           496264
## # ... with 13,440 more rows, and 3 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, landkreis <chr>

```

```

# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschlüssel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
  mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro
  mutate(landkreis = str_extract(Regionalschlüssel,
#bei der Gemeinde Selent wurde in der Excel Tabelle
  mutate(landkreis = ifelse(landkreis == "10575", "0
#manche Landkreise haben keine Infos zu den Einwohne
  filter( !is.na( Einwohner ))

```

```

## # A tibble: 11,050 x 8
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>                <dbl>         <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta...   87770         508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta...  247135         948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta...  216739         1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta...   78759         426019276
## 5 010510011011    Brunsbü... amtsfreie Geme...   12781          34934876
## 6 010510044044    Heide, ... amtsfreie Geme...   21508          37056249
## 7 010515163003    Averlak  amtsangehörige...    576            1197761
## 8 010515163010    Brickeln amtsangehörige...    219             496264
## 9 010515163012    Buchholz amtsangehörige...   1008            1229382
## 10 010515163016    Burg (D... amtsangehörige...   4114             7921959
## # ... with 11,040 more rows, and 3 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, landkreis <chr>

```

```
# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschluessel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
  mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro
  mutate(landkreis = str_extract(Regionalschluessel,
#bei der Gemeinde Selent wurde in der Excel Tabelle
  mutate(landkreis = ifelse(landkreis == "10575", "0
#manche Landkreise haben keine Infos zu den Einwohne
  filter( !is.na( Einwohner ) ) ->
schulden_bereinigt
```

```
# Die Daten wurden noch nicht schön eingelesen, in d
# waren die Variablennamen über mehrere Reihen gezog
schulden_individuell %>%
  filter(!is.na(Regionalschluessel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro
mutate(landkreis = str_extract(Regionalschluessel,
#bei der Gemeinde Selent wurde in der Excel Tabelle
mutate(landkreis = ifelse(landkreis == "10575", "0
#manche Landkreise haben keine Infos zu den Einwohne
  filter( !is.na( Einwohner )) ->
schulden_bereinigt
```

Konsistenzcheck zum Schulden- Datensatz

Interne Validität Schulden pro Kopf

- + Schulden_pro_Kopf_new von Hand berechnen
- + **Beachte:**
 - + Geschweiften Klammern entfernen bei Schulden_gesamt (mit `str_remove_all`), als auch die Leerzeichen innerhalb der Zahlen (z.B. 15 653), was wir mit `gsub("[:space:]")` erreichen.
 - + Tun wir das nicht, so würden wir wieder NAs im Datensatz erhalten
 - + Durch die `ifelse` Bedingung wird der Befehl `str_remove_all` nur angewendet, wenn tatsächlich geschweifte Klammern vorhanden sind

```
# Erstellen der Vergleichstabelle
schulden_consistency <- schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschluessel) ) %>%
  mutate(Schulden_gesamt = ifelse(is.na(as.numeric(Schulden_gesamt))==TRUE,
                                as.numeric(gsub("[:space:]", "", str_remove_all(Schulden_gesamt, "[{}]")),
                                as.numeric(Schulden_gesamt)),
  Schulden_pro_kopf = ifelse(is.na(as.numeric(Schulden_pro_kopf))==TRUE,
                              as.numeric(gsub("[:space:]", "", str_remove_all(Schulden_pro_kopf, "[{}]")),
                              as.numeric(Schulden_pro_kopf)),
  Einwohner_num = ifelse(is.na(as.numeric(Einwohner))==TRUE,
                          as.numeric(gsub("[:space:]", "", str_remove_all(Einwohner, "[{}]")),
                          as.numeric(Einwohner)),
  Schulden_pro_kopf_new = round(Schulden_gesamt / Einwohner_num,2) %>%
  mutate(landkreis = str_extract(Regionalschluessel, "^.{5}"),
         differenz = Schulden_pro_kopf - Schulden_pro_kopf_new)
```

```
# Erstellen der Vergleichstabelle
```

```
schulden_individuell
```

```
## # A tibble: 13,554 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>      <chr>
## 1 <NA>              <NA>      <NA>           <NA>      <NA>
## 2 <NA>              <NA>      <NA>           <NA>      <NA>
## 3 <NA>              <NA>      <NA>           <NA>      <NA>
## 4 <NA>              <NA>      <NA>           <NA>      EUR
## 5 <NA>              <NA>      <NA>           <NA>      1
## 6 0100100000000    Flensbu... kreisfreie Sta... 87770     508281539
## 7 0100200000000    Kiel, L... kreisfreie Sta... 247135    948848421
## 8 0100300000000    Lübeck,... kreisfreie Sta... 216739    1206620094
## 9 0100400000000    Neumüns... kreisfreie Sta... 78759     426019276
## 10 01051            Kreisve... Kreisverwaltung {133 684} 65179097
## # ... with 13,544 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>
```

```
# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues
```

```
## # A tibble: 13,421 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>                <chr>    <chr>                <chr>    <chr>
## 1 0100100000000      Flensbu... kreisfreie Sta... 87770    508281539
## 2 0100200000000      Kiel, L... kreisfreie Sta... 247135   948848421
## 3 0100300000000      Lübeck,... kreisfreie Sta... 216739   1206620094
## 4 0100400000000      Neumüns... kreisfreie Sta... 78759    426019276
## 5 01051              Kreisve... Kreisverwaltung {133 684} 65179097
## 6 010510011011      Brunsbü... amtsfreie Geme... 12781    34934876
## 7 010510044044      Heide, ... amtsfreie Geme... 21508    37056249
## 8 010515163          Amtsver... Amtsverwaltung {15 653} 1047175
## 9 010515163003      Averlak   amtsangehörige... 576      1197761
## 10 010515163010     Brickeln amtsangehörige... 219      496264
## # ... with 13,411 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>
```



```

# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues
mutate(Schulden_gesamt = ifelse(is.na(as.numeric(S
as.numeric(gsub("[
as.numeric(Schulde

```

```

## # A tibble: 13,421 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>           <chr>           <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta... 87770           508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta... 247135          948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta... 216739          1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta... 78759           426019276
## 5 01051           Kreisve... Kreisverwaltung {133 684}      65179097
## 6 010510011011    Brunsbü... amtsfreie Geme... 12781           34934876
## 7 010510044044    Heide, ... amtsfreie Geme... 21508           37056249
## 8 010515163       Amtsver... Amtsverwaltung {15 653}      1047175
## 9 010515163003    Averlak  amtsangehörige... 576             1197761
## 10 010515163010    Brickeln amtsangehörige... 219             496264
## # ... with 13,411 more rows, and 2 more variables: Schulden_pro_kopf <chr>,
## #   Bundesland <chr>

```

```

# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues
  mutate(Schulden_gesamt = ifelse(is.na(as.numeric(S
    as.numeric(gsub("[
    as.numeric(Schulde
mutate(Schulden_pro_kopf = ifelse(is.na(as.numeric
  as.numeric(gsub(
  as.numeric(Schul

```

```

## # A tibble: 13,421 x 7
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>                <chr>    <chr>          <chr>          <dbl>
## 1 0100100000000      Flensbu... kreisfreie Sta... 87770          508281539
## 2 0100200000000      Kiel, L... kreisfreie Sta... 247135         948848421
## 3 0100300000000      Lübeck,... kreisfreie Sta... 216739         1206620094
## 4 0100400000000      Neumüns... kreisfreie Sta... 78759          426019276
## 5 01051              Kreisve... Kreisverwaltung {133 684} 65179097
## 6 010510011011      Brunsbü... amtsfreie Geme... 12781          34934876
## 7 010510044044      Heide, ... amtsfreie Geme... 21508          37056249
## 8 010515163          Amtsver... Amtsverwaltung {15 653} 1047175
## 9 010515163003      Averlak  amtsangehörige... 576            1197761
## 10 010515163010      Brickeln amtsangehörige... 219            496264
## # ... with 13,411 more rows, and 2 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>

```

```

# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues) )
  mutate(Schulden_gesamt = ifelse(is.na(as.numeric(Schulden_gesamt)),
    as.numeric(gsub("[^0-9]", "", Schulden_gesamt)),
    as.numeric(Schulden_gesamt))
  mutate(Schulden_pro_kopf = ifelse(is.na(as.numeric(Schulden_pro_kopf)),
    as.numeric(gsub("[^0-9]", "", Schulden_pro_kopf)),
    as.numeric(Schulden_pro_kopf))
  mutate(Einwohner_num = ifelse(is.na(as.numeric(Einwohner_num)),
    as.numeric(gsub("[^0-9]", "", Einwohner_num)),
    as.numeric(Einwohner_num))

```

```

## # A tibble: 13,421 x 8
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>                <chr>    <chr>          <chr>          <dbl>
## 1 0100100000000      Flensbu... kreisfreie Sta... 87770          508281539
## 2 0100200000000      Kiel, L... kreisfreie Sta... 247135         948848421
## 3 0100300000000      Lübeck,... kreisfreie Sta... 216739         1206620094
## 4 0100400000000      Neumüns... kreisfreie Sta... 78759          426019276
## 5 01051              Kreisve... Kreisverwaltung {133 684} 65179097
## 6 010510011011      Brunsbü... amtsfreie Geme... 12781          34934876
## 7 010510044044      Heide, ... amtsfreie Geme... 21508          37056249
## 8 010515163          Amtsver... Amtsverwaltung {15 653} 1047175
## 9 010515163003      Averlak  amtsangehörige... 576            1197761
## 10 010515163010      Brickeln amtsangehörige... 219            496264
## # ... with 13,411 more rows, and 3 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, Einwohner_num <dbl>

```

```

# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues) )
  mutate(Schulden_gesamt = ifelse(is.na(as.numeric(Schulden_gesamt)),
    as.numeric(gsub("[^0-9]", "", Schulden_gesamt)),
    as.numeric(Schulden_gesamt))
  mutate(Schulden_pro_kopf = ifelse(is.na(as.numeric(Schulden_gesamt)),
    as.numeric(gsub("[^0-9]", "", Schulden_gesamt)),
    as.numeric(Schulden_gesamt) / as.numeric(Einwohner))
  mutate(Einwohner_num = ifelse(is.na(as.numeric(Einwohner)),
    as.numeric(gsub("[^0-9]", "", Einwohner)),
    as.numeric(Einwohner))
  mutate(Schulden_pro_kopf_new = round(Schulden_gesamt / Einwohner_num, 2))

```

```

## # A tibble: 13,421 x 9
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>                <chr>      <chr>          <chr>          <dbl>
## 1 0100100000000      Flensbu... kreisfreie Sta... 87770          508281539
## 2 0100200000000      Kiel, L... kreisfreie Sta... 247135         948848421
## 3 0100300000000      Lübeck,... kreisfreie Sta... 216739         1206620094
## 4 0100400000000      Neumüns... kreisfreie Sta... 78759          426019276
## 5 01051              Kreisve... Kreisverwaltung {133 684} 65179097
## 6 010510011011      Brunsbü... amtsfreie Geme... 12781          34934876
## 7 010510044044      Heide, ... amtsfreie Geme... 21508          37056249
## 8 010515163          Amtsver... Amtsverwaltung {15 653} 1047175
## 9 010515163003      Averlak  amtsangehörige... 576            1197761
## 10 010515163010      Brickeln amtsangehörige... 219            496264
## # ... with 13,411 more rows, and 4 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, Einwohner_num <dbl>, Schulden_pro_kopf_new <dbl>

```

```

# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues
  mutate(Schulden_gesamt = ifelse(is.na(as.numeric(S
    as.numeric(gsub("[
    as.numeric(Schulde
  mutate(Schulden_pro_kopf = ifelse(is.na(as.numeric
    as.numeric(gsub(
    as.numeric(Schul
  mutate(Einwohner_num = ifelse(is.na(as.numeric(Ein
    as.numeric(gsub("[[:
    as.numeric(Einwohner
  mutate(Schulden_pro_kopf_new = round(Schulden_gesa
  mutate(landkreis = str_extract(Regionalschluessel,
    differenz = Schulden_pro_kopf - Schulden_pr

```

```

## # A tibble: 13,421 x 11
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>      <chr>           <chr>           <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta... 87770           508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta... 247135          948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta... 216739          1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta... 78759           426019276
## 5 01051           Kreisve... Kreisverwaltung {133 684} 65179097
## 6 010510011011    Brunsbü... amtsfreie Geme... 12781           34934876
## 7 010510044044    Heide, ... amtsfreie Geme... 21508           37056249
## 8 010515163       Amtsver... Amtsverwaltung {15 653} 1047175
## 9 010515163003    Averlak  amtsangehörige... 576             1197761
## 10 010515163010    Brickeln amtsangehörige... 219             496264
## # ... with 13,411 more rows, and 6 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, Einwohner_num <dbl>, Schulden_pro_kopf_new <dbl>,
## #   landkreis <chr>, differenz <dbl>

```

```
# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter( !is.na(Einwohner) & !is.na(Regionalschlues
  mutate(Schulden_gesamt = ifelse(is.na(as.numeric(S
    as.numeric(gsub("[
    as.numeric(Schulde
  mutate(Schulden_pro_kopf = ifelse(is.na(as.numeric
    as.numeric(gsub(
    as.numeric(Schul
  mutate(Einwohner_num = ifelse(is.na(as.numeric(Ein
    as.numeric(gsub("[[:
    as.numeric(Einwohner
  mutate(Schulden_pro_kopf_new = round(Schulden_gesa
  mutate(landkreis = str_extract(Regionalschluessel,
    differenz = Schulden_pro_kopf - Schulden_pr
schulden_consistency
```

Interne Validität Schulden pro Kopf

```
range(schulden_consistency$differenz, na.rm=TRUE)
```

```
## [1] -0.49 0.50
```

Interne Validität Schulden pro Kopf

```
range(schulden_consistency$differenz, na.rm=TRUE)
```

```
## [1] -0.49 0.50
```

Die Differenzen liegen zwischen +/- 50 Cent

Interne Validität Schulden pro Kopf

Es gibt 12 nicht verfügbaren Werte:

```
filter(schulden_consistency, is.na(differenz))
```

```
## # A tibble: 12 x 11
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>              <chr>      <chr>          <chr>          <dbl>
## 1 033519501_Summe... Lohheid... Samtgemeindege... {768}          NA
## 2 033589501_Summe... Osterhe... Samtgemeindege... {2 884}        NA
## 3 052                Landsch... Landschaftsver... X              736700545
## 4 056                Landsch... Landschaftsver... X              671832895
## 5 058                Regiona... Kommunalverband X              132203729
## 6 067                Verwalt... Landeswohlfahr... X              30001419
## 7 074                Bezirks... Bezirksverband X              140470575
## 8 081a              Landesw... Landeswohlfahr... X              5323363
## 9 081b              Kommuna... Landeswohlfahr... X              0
## 10 091785127_Summe... Allersh... Verwaltungsgem... {7 190}        NA
## 11 095725512_Summe... Auracht... Verwaltungsgem... {4 368}        NA
## 12 144              Kommuna... Landeswohlfahr... X              10043701
## # ... with 6 more variables: Schulden_pro_kopf <dbl>, Bundesland <chr>,
## #   Einwohner_num <dbl>, Schulden_pro_kopf_new <dbl>, landkreis <chr>,
```

Bruttoinlandsprodukt

Informationen bzgl. des Bruttoinlandsprodukts

Nach dem Download bei den Statistischen Ämtern des Bundes und der Länder und einer ersten Betrachtung interessieren uns folgende Tabellenblätter:

- + Betrachten der Daten
 - + Tabellenblatt "1.1" ist für unsere Analyse ausschlaggebend (für das BIP)
 - + Tabellenblatt "3.1" ist für die Anzahl an Erwerbstätigen ausschlaggebend
 - + Tabellenblatt "5." ist für die Anzahl an Einwohnern ausschlaggebend
- + Die ersten vier Zeilen benötigen wir nicht
- + Die letzte Zeile enthält eine kurze Beschreibung die wir nicht benötigen
 - + **Lösung:** Behalte alle Zeilen, welche bei der Lfd. Nr. numerisch sind
- + Die folgenden Variablen benötigen wir nicht für unsere Analyse und können entfernt werden: Lfd. Nr., EU-Code, NUTS 1, NUTS 2, NUTS 3, Land, Gebietseinheit

Informationen bzgl. des Bruttoinlandsprodukts

```
# Blatt 1.1 einlesen und die ersten 4 Zeilen skippen
bip <- read_xlsx("./data/R2B1.xlsx", sheet="1.1", skip = 4)
erwerb <- read_xlsx("./data/R2B1.xlsx", sheet="3.1", skip = 4)
einwohner <- read_xlsx("./data/R2B1.xlsx", sheet = "5.", skip = 4)
```

```
# Zeile löschen in der die `Lfd. Nr.` nicht nummeris
# Zusätzliche Spalten löschen
bip
```

```
## # A tibble: 448 x 33
##   `Lfd. Nr.` `EU-Code` `Regional-schlü... Land `NUTS 1` `NUTS 2` `NUTS 3`
##   <chr>      <chr>      <chr>          <chr> <chr>      <chr>      <chr>
## 1 <NA>      <NA>      <NA>          <NA> <NA>      <NA>      <NA>
## 2 1         DE1        08            BW    1         <NA>      <NA>
## 3 2         DE11       081           BW    <NA>      2         <NA>
## 4 3         DE111      08111         BW    <NA>      <NA>      3
## 5 4         DE112      08115         BW    <NA>      <NA>      3
## 6 5         DE113      08116         BW    <NA>      <NA>      3
## 7 6         DE114      08117         BW    <NA>      <NA>      3
## 8 7         DE115      08118         BW    <NA>      <NA>      3
## 9 8         DE116      08119         BW    <NA>      <NA>      3
## 10 9        DE117      08121         BW    <NA>      <NA>      3
## # ... with 438 more rows, and 26 more variables: Gebietseinheit <chr>,
## # `1992` <chr>, `1994` <chr>, `1995` <chr>, `1996` <chr>, `1997` <chr>,
## # `1998` <chr>, `1999` <chr>, `2000` <dbl>, `2001` <dbl>, `2002` <dbl>,
## # `2003` <dbl>, `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>,
## # `2008` <dbl>, `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>,
## # `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

```
# Zeile löschen in der die `Lfd. Nr.` nicht nummeris
# Zusätzliche Spalten löschen
bip %>%
  filter(is.na(as.numeric(`Lfd. Nr.`))==FALSE)
```

```
## # A tibble: 445 x 33
##   `Lfd. Nr.` `EU-Code` `Regional-schlü... Land `NUTS 1` `NUTS 2` `NUTS 3`
##   <chr>      <chr>      <chr>          <chr> <chr>      <chr>      <chr>
## 1 1          DE1         08             BW     1          <NA>      <NA>
## 2 2          DE11        081            BW     <NA>      2          <NA>
## 3 3          DE111       08111          BW     <NA>      <NA>      3
## 4 4          DE112       08115          BW     <NA>      <NA>      3
## 5 5          DE113       08116          BW     <NA>      <NA>      3
## 6 6          DE114       08117          BW     <NA>      <NA>      3
## 7 7          DE115       08118          BW     <NA>      <NA>      3
## 8 8          DE116       08119          BW     <NA>      <NA>      3
## 9 9          DE117       08121          BW     <NA>      <NA>      3
## 10 10         DE118       08125          BW     <NA>      <NA>      3
## # ... with 435 more rows, and 26 more variables: Gebietseinheit <chr>,
## #   `1992` <chr>, `1994` <chr>, `1995` <chr>, `1996` <chr>, `1997` <chr>,
## #   `1998` <chr>, `1999` <chr>, `2000` <dbl>, `2001` <dbl>, `2002` <dbl>,
## #   `2003` <dbl>, `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>,
## #   `2008` <dbl>, `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>,
## #   `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

```
# Zeile löschen in der die `Lfd. Nr.` nicht nummeris
# Zusätzliche Spalten löschen
bip %>%
  filter(is.na(as.numeric(`Lfd. Nr.`))==FALSE) %>%
  select(-c(`Lfd. Nr.`, `EU-Code`, `NUTS 1`, `NUTS 2`))
```

```
## # A tibble: 445 x 26
##   `Regional-schlü...` `1992` `1994` `1995` `1996` `1997` `1998` `1999` `2000`
##   <chr>                <chr> <chr> <chr> <chr> <chr> <chr> <chr> <dbl>
## 1 08                    25493... 26274... 27240... 27745... 28310... 29167... 30138... 3.10e5
## 2 081                   10985... 11086... 11474... 11624... 12040... 12321... 12714... 1.30e5
## 3 08111                 32014... 30887... 31479... 31988... 33589... 32711... 34128... 3.43e4
## 4 08115                 11977... 11750... 11867... 12025... 13820... 13545... 14308... 1.38e4
## 5 08116                 12212... 12462... 12748... 13166... 13291... 13937... 14189... 1.44e4
## 6 08117                 5119.... 5264.... 5546.... 5746.7 5775.... 5944.... 6032.... 6.12e3
## 7 08118                 11486... 11965... 12571... 12707... 12962... 13324... 13663... 1.44e4
## 8 08119                 8487.... 8743.... 9358.... 8816.... 8974.... 9213.... 9749.... 1.04e4
## 9 08121                 4214.... 4401.... 4546.... 4534.... 4606.... 5682.... 5323.... 5.31e3
## 10 08125                6112.... 6195.... 6663.... 6900.... 7113.... 7737.... 8033.... 8.57e3
## # ... with 435 more rows, and 17 more variables: `2001` <dbl>, `2002` <dbl>,
## # `2003` <dbl>, `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>,
## # `2008` <dbl>, `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>,
## # `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

```
# Zeile löschen in der die `Lfd. Nr.` nicht nummeris
# Zusätzliche Spalten löschen
bip %>%
  filter(is.na(as.numeric(`Lfd. Nr.`))==FALSE) %>%
  select(-c(`Lfd. Nr.`, `EU-Code`, `NUTS 1`, `NUTS 2`))
  rename(Regionalschlüssel = `Regional-schlüssel`)
```

```
## # A tibble: 445 x 26
##   Regionalschlues... `1992` `1994` `1995` `1996` `1997` `1998` `1999` `2000`
##   <chr>             <chr> <chr> <chr> <chr> <chr> <chr> <chr> <dbl>
## 1 08                25493... 26274... 27240... 27745... 28310... 29167... 30138... 3.10e5
## 2 081               10985... 11086... 11474... 11624... 12040... 12321... 12714... 1.30e5
## 3 08111            32014... 30887... 31479... 31988... 33589... 32711... 34128... 3.43e4
## 4 08115            11977... 11750... 11867... 12025... 13820... 13545... 14308... 1.38e4
## 5 08116            12212... 12462... 12748... 13166... 13291... 13937... 14189... 1.44e4
## 6 08117            5119.... 5264.... 5546.... 5746.7 5775.... 5944.... 6032.... 6.12e3
## 7 08118            11486... 11965... 12571... 12707... 12962... 13324... 13663... 1.44e4
## 8 08119            8487.... 8743.... 9358.... 8816.... 8974.... 9213.... 9749.... 1.04e4
## 9 08121            4214.... 4401.... 4546.... 4534.... 4606.... 5682.... 5323.... 5.31e3
## 10 08125           6112.... 6195.... 6663.... 6900.... 7113.... 7737.... 8033.... 8.57e3
## # ... with 435 more rows, and 17 more variables: `2001` <dbl>, `2002` <dbl>,
## # `2003` <dbl>, `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>,
## # `2008` <dbl>, `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>,
## # `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```



```
# Zeile löschen in der die `Lfd. Nr.` nicht nummeris
# Zusätzliche Spalten löschen
bip %>%
  filter(is.na(as.numeric(`Lfd. Nr.`))==FALSE) %>%
  select(-c(`Lfd. Nr.`, `EU-Code`, `NUTS 1`, `NUTS 2`))
  rename(Regionalschlüssel = `Regional-schlüssel`)
bip_wide
```

Informationen bzgl. des Bruttoinlandsprodukts

Was ist hier eine Beobachtung?

Informationen bzgl. des Bruttoinlandsprodukts

Was ist hier eine Beobachtung?

Entsprechend können wir bei den Erwerbstätigen und den Einwohnern vorgehen:

```
# Zeile löschen in der die `Lfd. Nr.` nicht numerisch ist
# Zusätzliche Spalten löschen
erwerb_wide <- erwerb %>%
  filter(is.na(as.numeric(`Lfd. Nr.`))==FALSE) %>%
  select(-c(`Lfd. Nr.`, `EU-Code`, `NUTS 1`, `NUTS 2`, `NUTS 3`, Land, Gebietseinheit)) %>%
  rename(Regionalschlüssel = `Regional-schlüssel`)

einwohner_wide <- einwohner %>%
  filter(is.na(as.numeric(`Lfd. Nr.`))==FALSE) %>%
  select(-c(`Lfd. Nr.`, `EU-Code`, `NUTS 1`, `NUTS 2`, `NUTS 3`, Land, Gebietseinheit)) %>%
  rename(Regionalschlüssel = `Regional-schlüssel`)
```

Informationen bzgl. des Bruttoinlandsprodukts

Datensatz,

- ✚ ist ein Panel: Mehrere Jahre für mehrere Landkreise in Deutschland vorhanden
- ✚ ist im `wide` Format -> d.h. die Daten sind nicht `tidy`

```
head(bip_wide, 3)
```

```
## # A tibble: 3 x 26
##   Regionalschlues... `1992` `1994` `1995` `1996` `1997` `1998` `1999` `2000`
##   <chr>              <chr> <chr> <chr> <chr> <chr> <chr> <chr> <dbl>
## 1 08                 25493... 26274... 27240... 27745... 28310... 29167... 30138... 3.10e5
## 2 081                10985... 11086... 11474... 11624... 12040... 12321... 12714... 1.30e5
## 3 08111             32014... 30887... 31479... 31988... 33589... 32711... 34128... 3.43e4
## # ... with 17 more variables: `2001` <dbl>, `2002` <dbl>, `2003` <dbl>,
## #   `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>, `2008` <dbl>,
## #   `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>, `2013` <dbl>,
## #   `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

Informationen bzgl. des Bruttoinlandsprodukts

Datensatz,

- ✚ ist ein Panel: Mehrere Jahre für mehrere Landkreise in Deutschland vorhanden
- ✚ ist im `wide` Format -> d.h. die Daten sind nicht `tidy`

```
head(bip_wide, 3)
```

```
## # A tibble: 3 x 26
##   Regionalschlues... `1992` `1994` `1995` `1996` `1997` `1998` `1999` `2000`
##   <chr>              <chr> <chr> <chr> <chr> <chr> <chr> <chr> <dbl>
## 1 08                 25493... 26274... 27240... 27745... 28310... 29167... 30138... 3.10e5
## 2 081                10985... 11086... 11474... 11624... 12040... 12321... 12714... 1.30e5
## 3 08111              32014... 30887... 31479... 31988... 33589... 32711... 34128... 3.43e4
## # ... with 17 more variables: `2001` <dbl>, `2002` <dbl>, `2003` <dbl>,
## #   `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>, `2008` <dbl>,
## #   `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>, `2013` <dbl>,
## #   `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

Was sind die Bedingungen für einen `tidy` Datensatz?

Daten in das long-Format überführen

Datensatz ins long-Format überführen mit `pivot_longer`:

```
bip_long <- pivot_longer(bip_wide, cols = c("1992":"2017") , names_to = "Jahr", values_to = "BIP")
```

```
Fehler: No common type for `1992` <character> and `2000` <double>.
```

Daten in das `long`-Format überführen

BIP sollte normalerweise numerisch sein:

- + Klasse `double` sollte korrekt sein
- + umformatieren der Spalten `1992 - 1999`
- + mit `across()` kann der `mutate()`-Befehl über mehrere Spalten angewendet werden

#BIP von 1992 - 1999 umformen (als numerische Variab


```
#BIP von 1992 - 1999 umformen (als numerische Variab  
bip_wide
```

```
## # A tibble: 445 x 26  
##   Regionalschlues... `1992` `1994` `1995` `1996` `1997` `1998` `1999` `2000`  
##   <chr> <chr> <chr> <chr> <chr> <chr> <chr> <dbl>  
## 1 08 25493... 26274... 27240... 27745... 28310... 29167... 30138... 3.10e5  
## 2 081 10985... 11086... 11474... 11624... 12040... 12321... 12714... 1.30e5  
## 3 08111 32014... 30887... 31479... 31988... 33589... 32711... 34128... 3.43e4  
## 4 08115 11977... 11750... 11867... 12025... 13820... 13545... 14308... 1.38e4  
## 5 08116 12212... 12462... 12748... 13166... 13291... 13937... 14189... 1.44e4  
## 6 08117 5119.... 5264.... 5546.... 5746.7 5775.... 5944.... 6032.... 6.12e3  
## 7 08118 11486... 11965... 12571... 12707... 12962... 13324... 13663... 1.44e4  
## 8 08119 8487.... 8743.... 9358.... 8816.... 8974.... 9213.... 9749.... 1.04e4  
## 9 08121 4214.... 4401.... 4546.... 4534.... 4606.... 5682.... 5323.... 5.31e3  
## 10 08125 6112.... 6195.... 6663.... 6900.... 7113.... 7737.... 8033.... 8.57e3  
## # ... with 435 more rows, and 17 more variables: `2001` <dbl>, `2002` <dbl>,  
## # `2003` <dbl>, `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>,  
## # `2008` <dbl>, `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>,  
## # `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

```
#BIP von 1992 - 1999 umformen (als numerische Variab  
bip_wide %>%  
  select(`1992`:`1999`)
```

```
## # A tibble: 445 x 7  
##   `1992`      `1994`      `1995`      `1996`      `1997`      `1998`      `1999`  
##   <chr>      <chr>      <chr>      <chr>      <chr>      <chr>      <chr>  
## 1 254933.035... 262748.99... 272401.95... 277454.179 283105.929 291677.68 301388.3  
## 2 109854.755... 110868.36... 114745.16... 116240.28... 120409.49... 123210.43... 127147.0  
## 3 32014.3810... 30887.921... 31479.096... 31988.062... 33589.646... 32711.758... 34128.16  
## 4 11977.21      11750.939  11867.928  12025.593... 13820.067... 13545.587  14308.85  
## 5 12212.2720... 12462.454  12748.133  13166.955  13291.102... 13937.165... 14189.82  
## 6 5119.55299... 5264.9319... 5546.5290... 5746.7      5775.7439... 5944.8069... 6032.456  
## 7 11486.777     11965.535  12571.231  12707.628... 12962.808... 13324.504... 13663.91  
## 8 8487.68100... 8743.8909... 9358.4269... 8816.5249... 8974.5640... 9213.8649... 9749.792  
## 9 4214.17799... 4401.4340... 4546.1450... 4534.0709... 4606.848     5682.3159... 5323.235  
## 10 6112.03200... 6195.4309... 6663.0569... 6900.6459... 7113.1009... 7737.732     8033.988  
## # ... with 435 more rows
```

```
#BIP von 1992 - 1999 umformen (als numerische Variab  
bip_wide %>%  
  select(`1992`:`1999`) %>%  
  mutate(across(is.character, as.double))
```

```
## # A tibble: 445 x 7  
##   `1992` `1994` `1995` `1996` `1997` `1998` `1999`  
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 254933. 262749. 272402. 277454. 283106. 291678. 301388.  
## 2 109855. 110868. 114745. 116240. 120409. 123210. 127147.  
## 3 32014. 30888. 31479. 31988. 33590. 32712. 34128.  
## 4 11977. 11751. 11868. 12026. 13820. 13546. 14309.  
## 5 12212. 12462. 12748. 13167. 13291. 13937. 14190.  
## 6 5120. 5265. 5547. 5747. 5776. 5945. 6032.  
## 7 11487. 11966. 12571. 12708. 12963. 13325. 13664.  
## 8 8488. 8744. 9358. 8817. 8975. 9214. 9750.  
## 9 4214. 4401. 4546. 4534. 4607. 5682. 5323.  
## 10 6112. 6195. 6663. 6901. 7113. 7738. 8034.  
## # ... with 435 more rows
```

```
#BIP von 1992 - 1999 umformen (als numerische Variab  
bip_wide %>%  
  select(`1992`:`1999`) %>%  
  mutate(across(is.character, as.double)) ->  
bip_double
```

Entsprechend dann bei den Einwohnern und Erwerbstätigen:

Es wird eine Warnmeldung ausgegeben das NAs bei der Umwandlung erzeugt wurden. Warum?

```
# Erwerbstätige von 1992 - 1999 umformen (als numerische Variable)
erwerb_double <- erwerb_wide %>%
  select(`1992`:`1999`) %>%
  mutate(across(is.character, as.double))
```

```
## Warning: Problem with `mutate()` input `..1`.
## i NAs durch Umwandlung erzeugt
## i Input `..1` is `across(is.character, as.double)`.
```

```
## Warning in fn(col, ...): NAs durch Umwandlung erzeugt
```

```
## Warning: Problem with `mutate()` input `..1`.
## i NAs durch Umwandlung erzeugt
## i Input `..1` is `across(is.character, as.double)`.
```

```
## Warning in fn(col, ...): NAs durch Umwandlung erzeugt
```

```
## Warning: Problem with `mutate()` input `..1`.
## i NAs durch Umwandlung erzeugt
## i Input `..1` is `across(is.character, as.double)`.
```


Eine Umwandlung zu NA geschieht bei den Werten bei denen – eingetragen wurde. D.h. für uns ist es ok hier ein NA einzutragen. Somit können wir die Umwandlung in die Klasse `double` durchführen:

```
bip_wide <- bip_wide %>%
  select(-(`1992`:`1999`)) %>%
  bind_cols(bip_double)

erwerb_wide <- erwerb_wide %>%
  select(-(`1992`:`1999`)) %>%
  bind_cols(erwerb_double)

einwohner_wide <- einwohner_wide %>%
  select(-(`1992`:`1999`)) %>%
  bind_cols(einwohner_double)
```

Daten in das `long`-Format überführen

Nun können wir den Datensatz ins `long`-Format transferieren und nach dem Jahr sortieren.

- + Einwohner und Erwerbstätigen in 1000 Personen angegeben, daher Erwerbstätigen und Einwohner mit 1000 multiplizieren.
- + BIP ist in 1 Mio. Euro angegeben, daher die Multiplikation mit 1 Mio.


```
# BIP ins long-Format
pivot_longer(bip_wide, cols = c("2000":"1999"), nam
```

```
## # A tibble: 11,125 x 3
##   Regionalschlüssel Jahr      bip
##   <chr>                <chr>  <dbl>
## 1 08                    2000  309550.
## 2 08                    2001  323511.
## 3 08                    2002  325818.
## 4 08                    2003  328749.
## 5 08                    2004  332697.
## 6 08                    2005  336269.
## 7 08                    2006  356956.
## 8 08                    2007  377127.
## 9 08                    2008  381486.
## 10 08                   2009  355496.
## # ... with 11,115 more rows
```

```
# BIP ins long-Format
pivot_longer(bip_wide, cols = c("2000":"1999"), nam
mutate( Jahr = as.numeric(Jahr),
bip = bip * 1000000)
```

```
## # A tibble: 11,125 x 3
##   Regionalschlüssel Jahr      bip
##   <chr>             <dbl>   <dbl>
## 1 08                 2000 309550222000
## 2 08                 2001 323510871000
## 3 08                 2002 325817946000
## 4 08                 2003 328748524000
## 5 08                 2004 332697391000
## 6 08                 2005 336268560000
## 7 08                 2006 356956434000
## 8 08                 2007 377126811000
## 9 08                 2008 381485624000
## 10 08                2009 355496127000
## # ... with 11,115 more rows
```

```
# BIP ins long-Format
pivot_longer(bip_wide, cols = c("2000":"1999") , nam
  mutate( Jahr = as.numeric(Jahr),
          bip = bip * 1000000) %>%
  arrange( Jahr )
```

```
## # A tibble: 11,125 x 3
##   Regionalschlüssel Jahr      bip
##   <chr>             <dbl>   <dbl>
## 1 08                 1992 254933036000
## 2 081                1992 109854756000
## 3 08111              1992 32014381000
## 4 08115              1992 11977210000
## 5 08116              1992 12212272000
## 6 08117              1992 5119553000
## 7 08118              1992 11486777000
## 8 08119              1992 8487681000.
## 9 08121              1992 4214178000
## 10 08125             1992 6112032000
## # ... with 11,115 more rows
```

```
# BIP ins long-Format
pivot_longer(bip_wide, cols = c("2000":"1999") , nam
  mutate( Jahr = as.numeric(Jahr),
          bip = bip * 1000000) %>%
  arrange( Jahr ) ->
bip_long
```

Für die Erwerbstätigen und Einwohner entsprechend:

```
# Anzahl der Erwerbstätigen ins long-Format
erwerb_long <- pivot_longer(erwerb_wide, cols = c("2000":"1999") , names_to = "Jahr", values_to = "erw") %>%
  mutate( Jahr = as.numeric(Jahr),
           erw = erw * 1000) %>%
  arrange( Jahr )

# Anzahl der Einwohner ins long-Format
einwohner_long <- pivot_longer(einwohner_wide, cols = c("2000":"1999") , names_to = "Jahr", values_to = "einw") %>%
  mutate( Jahr = as.numeric(Jahr),
           einwohner = einwohner * 1000) %>%
  arrange( Jahr )
```

Konsistenzchecks

Hier sollten Sie selbst aktiv werden und die Daten auf Konsistenz prüfen:

Als Konsistenzcheck könnten Sie hier die Anzahl der Einwohner aus den verschiedenen Datensätzen vergleichen.

Kartenmaterial hinzufügen

Wir benötigen hier eine Karte von Deutschland mit den einzelnen Verwaltungsgrenzen als SHAPE-File und können diese mittels des `sf`-Pakets einlesen.

Das [OpenData Portal des Bundesamts für Kartographie und Geodäsie](#) stellt die nötigen Informationen kostenlos zur Verfügung.

[Die Dokumentation der Daten](#) sollten wir uns immer zuerst anschauen, bevor wir die Datenquelle herunterladen.

Dies gilt nicht nur für die Geodaten, sondern allgemein für alle Datenreihen.

Bitte versuchen Sie selbst die Daten herunterzuladen und anhand des Regionalschlüssels (RS) mit dem BIP, den Arbeitslosen und den Schulden zusammenzuführen.

Datensätze zusammenführen

Nun möchten wir die unterschiedlichen Datensätze noch zu einem zusammenfügen!

Zuerst müssen wir folgende Schritte unternehmen:

- + Informationen zur Verschuldung auf Landkreisebene aggregieren
- + Daten zum BIP auf das Jahr 2017 einschränken.
- + Datensätze anhand des Regionalschlüssels miteinander verbinden.

Weiterhin können wir die geografischen Daten separat abspeichern und bei Bedarf anhand des Regionalschlüssels zu unserem Datensatz hinzumergen.


```
# Schulden auf Landkreisebene
```

```
schulden_bereinigt
```

```
## # A tibble: 11,050 x 8
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>                <dbl>          <dbl>
## 1 0100100000000    Flensbu... kreisfreie Sta...  87770          508281539
## 2 0100200000000    Kiel, L... kreisfreie Sta...  247135         948848421
## 3 0100300000000    Lübeck,... kreisfreie Sta...  216739         1206620094
## 4 0100400000000    Neumüns... kreisfreie Sta...  78759          426019276
## 5 010510011011    Brunsbü... amtsfreie Geme...  12781          34934876
## 6 010510044044    Heide, ... amtsfreie Geme...  21508          37056249
## 7 010515163003    Averlak  amtsangehörige...   576           1197761
## 8 010515163010    Brickeln amtsangehörige...   219            496264
## 9 010515163012    Buchholz amtsangehörige...  1008           1229382
## 10 010515163016    Burg (D... amtsangehörige...   4114           7921959
## # ... with 11,040 more rows, and 3 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, landkreis <chr>
```

```
# Schulden auf Landkreisebene
schulden_bereinigt %>%
  group_by(landkreis)
```

```
## # A tibble: 11,050 x 8
## # Groups:   landkreis [397]
##   Regionalschlues... Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>             <chr>    <chr>                <dbl>         <dbl>
## 1 010010000000      Flensbu... kreisfreie Sta...   87770         508281539
## 2 010020000000      Kiel, L... kreisfreie Sta...  247135         948848421
## 3 010030000000      Lübeck,... kreisfreie Sta...  216739         1206620094
## 4 010040000000      Neumüns... kreisfreie Sta...   78759         426019276
## 5 010510011011      Brunsbü... amtsfreie Geme...   12781          34934876
## 6 010510044044      Heide, ... amtsfreie Geme...   21508          37056249
## 7 010515163003      Averlak  amtsangehörige...    576            1197761
## 8 010515163010      Brickeln amtsangehörige...   219             496264
## 9 010515163012      Buchholz amtsangehörige...   1008            1229382
## 10 010515163016      Burg (D... amtsangehörige...   4114             7921959
## # ... with 11,040 more rows, and 3 more variables: Schulden_pro_kopf <dbl>,
## #   Bundesland <chr>, landkreis <chr>
```

```
# Schulden auf Landkreisebene
schulden_bereinigt %>%
  group_by(landkreis) %>%
  summarise( Schulden_pro_kopf_lk = sum(Schulden_gesamt) /
            Einwohner = sum(Einwohner),
            Schulden_gesamt = sum(Schulden_gesamt))
```

```
## # A tibble: 397 x 4
##   landkreis Schulden_pro_kopf_lk Einwohner Schulden_gesamt
##   <chr>          <dbl>     <dbl>         <dbl>
## 1 01001          5791.     87770         508281539
## 2 01002          3839.    247135         948848421
## 3 01003          5567.    216739        1206620094
## 4 01004          5409.     78759         426019276
## 5 01051          1670.    133684         223191181
## 6 01053          1293.    195677         252944185
## 7 01054          2624.    165642         434624906
## 8 01055          1890.    200931         379698731
## 9 01056          2225.    311713         693581474
## 10 01057         1532.    128763         197203962
## # ... with 387 more rows
```

```
# Schulden auf Landkreisebene
schulden_bereinigt %>%
  group_by(landkreis) %>%
  summarise( Schulden_pro_kopf_lk = sum(Schulden_gesamt) /
            Einwohner = sum(Einwohner),
            Schulden_gesamt = sum(Schulden_gesamt))
rename(Regionalschlüssel = landkreis)
```

```
## # A tibble: 397 x 4
##   Regionalschlüssel Schulden_pro_kopf_lk Einwohner Schulden_gesamt
##   <chr>                <dbl>     <dbl>         <dbl>
## 1 01001                5791.     87770         508281539
## 2 01002                3839.     247135         948848421
## 3 01003                5567.     216739         1206620094
## 4 01004                5409.       78759         426019276
## 5 01051                1670.     133684         223191181
## 6 01053                1293.     195677         252944185
## 7 01054                2624.     165642         434624906
## 8 01055                1890.     200931         379698731
## 9 01056                2225.     311713         693581474
## 10 01057               1532.     128763         197203962
## # ... with 387 more rows
```

```
# Schulden auf Landkreisebene
schulden_bereinigt %>%
  group_by(landkreis) %>%
  summarise( Schulden_pro_kopf_lk = sum(Schulden_ges
    Einwohner = sum(Einwohner),
    Schulden_gesamt = sum(Schulden_gesamt))
  rename(Regionalschlüssel = landkreis) ->
schulden_kombi
```

```
# Anzahl an Erwerbstätigen für das Jahr 2017
```

```
erwerb_long
```

```
## # A tibble: 11,125 x 3
```

```
##   Regionalschlüssel Jahr   erw
```

```
##   <chr>             <dbl> <dbl>
```

```
## 1 08                 1992 5219305
```

```
## 2 081                1992 2042070
```

```
## 3 08111             1992 486056
```

```
## 4 08115            1992 188009
```

```
## 5 08116            1992 237005
```

```
## 6 08117            1992 117863
```

```
## 7 08118            1992 222503
```

```
## 8 08119            1992 176550
```

```
## 9 08121            1992 93574
```

```
## 10 08125           1992 115699
```

```
## # ... with 11,115 more rows
```

```
# Anzahl an Erwerbstätigen für das Jahr 2017
erwerb_long %>%
  filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
```

```
## # A tibble: 399 x 3
##   Regionalschlüssel Jahr     erw
##   <chr>             <dbl>  <dbl>
## 1 08111             2017 524127.
## 2 08115             2017 236642
## 3 08116             2017 282159
## 4 08117             2017 124248
## 5 08118             2017 265086
## 6 08119             2017 204669
## 7 08121             2017  96895
## 8 08125             2017 176665
## 9 08126             2017  71992
## 10 08127            2017 111521
## # ... with 389 more rows
```

```
# Anzahl an Erwerbstätigen für das Jahr 2017
erwerb_long %>%
  filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
  select(-Jahr)
```

```
## # A tibble: 399 x 2
##   Regionalschlüssel   erw
##   <chr>              <dbl>
## 1 08111              524127.
## 2 08115              236642
## 3 08116              282159
## 4 08117              124248
## 5 08118              265086
## 6 08119              204669
## 7 08121               96895
## 8 08125              176665
## 9 08126               71992
## 10 08127             111521
## # ... with 389 more rows
```



```
# Anzahl an Erwerbstätigen für das Jahr 2017
erwerb_long %>%
  filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
  select(-Jahr) ->
erwerb_kombi
```

```
# Anzahl an Einwohner für das Jahr 2017
```

```
einwohner_long
```

```
## # A tibble: 11,125 x 3
```

```
##   Regionalschlüssel Jahr einwohner
```

```
##   <chr>           <dbl>     <dbl>
```

```
## 1 08                1992  10050431
```

```
## 2 081               1992   3771006
```

```
## 3 08111            1992   593628
```

```
## 4 08115            1992   343190
```

```
## 5 08116            1992   487370
```

```
## 6 08117            1992   248688
```

```
## 7 08118            1992   475248
```

```
## 8 08119            1992   389670
```

```
## 9 08121            1992   118566
```

```
## 10 08125           1992   283163
```

```
## # ... with 11,115 more rows
```

```
# Anzahl an Einwohner für das Jahr 2017
einwohner_long %>%
  filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
```

```
## # A tibble: 399 x 3
##   Regionalschlüssel Jahr einwohner
##   <chr>             <dbl>     <dbl>
## 1 08111             2017     630388
## 2 08115             2017     387718
## 3 08116             2017     530620
## 4 08117             2017     255482
## 5 08118             2017     540266
## 6 08119             2017     423788
## 7 08121             2017     124442
## 8 08125             2017     339172
## 9 08126             2017     111041
## 10 08127            2017     193581
## # ... with 389 more rows
```

```
# Anzahl an Einwohner für das Jahr 2017
einwohner_long %>%
  filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
select(-Jahr)
```

```
## # A tibble: 399 x 2
##   Regionalschlüssel einwohner
##   <chr>             <dbl>
## 1 08111             630388
## 2 08115             387718
## 3 08116             530620
## 4 08117             255482
## 5 08118             540266
## 6 08119             423788
## 7 08121             124442
## 8 08125             339172
## 9 08126             111041
## 10 08127            193581
## # ... with 389 more rows
```

```
# Anzahl an Einwohner für das Jahr 2017
einwohner_long %>%
  filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
  select(-Jahr) ->
einwohner_kombi
```

```
# Anzahl der Einwohner mit dem BIP verbinden um das  
left_join(bip_long, einwohner_long, by=c("Regionalschlüssel", "Jahr"))
```

```
## # A tibble: 11,125 x 4  
##   Regionalschlüssel Jahr      bip einwohner  
##   <chr>           <dbl>    <dbl>    <dbl>  
## 1 08                1992 254933036000 10050431  
## 2 081               1992 109854756000 3771006  
## 3 08111            1992 32014381000 593628  
## 4 08115            1992 11977210000 343190  
## 5 08116            1992 12212272000 487370  
## 6 08117            1992 5119553000 248688  
## 7 08118            1992 11486777000 475248  
## 8 08119            1992 8487681000 389670  
## 9 08121            1992 4214178000 118566  
## 10 08125           1992 6112032000 283163  
## # ... with 11,115 more rows
```

```
# Anzahl der Einwohner mit dem BIP verbinden um das  
left_join(bip_long, einwohner_long, by=c("Regionalschl  
mutate(bip_pro_kopf = bip / einwohner)
```

```
## # A tibble: 11,125 x 5  
##   Regionalschlüssel Jahr      bip einwohner bip_pro_kopf  
##   <chr>           <dbl>    <dbl>    <dbl>    <dbl>  
## 1 08              1992 254933036000 10050431 25365.  
## 2 081             1992 109854756000 3771006 29131.  
## 3 08111          1992 32014381000 593628 53930.  
## 4 08115          1992 11977210000 343190 34900.  
## 5 08116          1992 12212272000 487370 25057.  
## 6 08117          1992 5119553000 248688 20586.  
## 7 08118          1992 11486777000 475248 24170.  
## 8 08119          1992 8487681000. 389670 21782.  
## 9 08121          1992 4214178000 118566 35543.  
## 10 08125         1992 6112032000 283163 21585.  
## # ... with 11,115 more rows
```

```
# Anzahl der Einwohner mit dem BIP verbinden um das  
left_join(bip_long, einwohner_long, by=c("Regionalschl  
mutate(bip_pro_kopf = bip / einwohner) %>%  
# BIP auf Landkreisebene im Jahr 2017  
filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
```

```
## # A tibble: 399 x 5  
##   Regionalschlüssel  Jahr      bip einwohner bip_pro_kopf  
##   <chr>             <dbl>    <dbl>    <dbl>    <dbl>  
## 1 08111             2017 53065815000    630388    84180.  
## 2 08115             2017 25861060000    387718    66701.  
## 3 08116             2017 22132764000    530620    41711.  
## 4 08117             2017  8732092000    255482    34179.  
## 5 08118             2017 24948172000    540266    46178.  
## 6 08119             2017 14726348000    423788    34749.  
## 7 08121             2017  6570209000    124442    52797.  
## 8 08125             2017 16903400000.    339172    49837.  
## 9 08126             2017  5519033000    111041    49703.  
## 10 08127            2017  7876380000    193581    40688.  
## # ... with 389 more rows
```



```
# Anzahl der Einwohner mit dem BIP verbinden um das
left_join(bip_long, einwohner_long, by=c("Regionalschlüssel"))
mutate(bip_pro_kopf = bip / einwohner) %>%
# BIP auf Landkreisebene im Jahr 2017
filter(nchar(Regionalschlüssel) == 5 & Jahr == 2017)
select(-c(Jahr, einwohner))
```

```
## # A tibble: 399 x 3
##   Regionalschlüssel      bip bip_pro_kopf
##   <chr>              <dbl>     <dbl>
## 1 08111              53065815000 84180.
## 2 08115              25861060000 66701.
## 3 08116              22132764000 41711.
## 4 08117               8732092000 34179.
## 5 08118              24948172000 46178.
## 6 08119              14726348000 34749.
## 7 08121               6570209000 52797.
## 8 08125              16903400000. 49837.
## 9 08126              5519033000 49703.
## 10 08127             7876380000 40688.
## # ... with 389 more rows
```

```
# Anzahl der Einwohner mit dem BIP verbinden um das
left_join(bip_long, einwohner_long, by=c("Regionalsc
mutate(bip_pro_kopf = bip / einwohner) %>%
# BIP auf Landkreisebene im Jahr 2017
filter(nchar(Regionalschlüssel) == 5 & Jahr == 20
select(-c(Jahr, einwohner)) ->
bip_kombi
```

```
# Datensätze zusammenführen  
  
# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis  
# Namen der Bundesländer zumergen  
alo_landkreis
```

```
## # A tibble: 401 x 2  
##   Regionalschlüssel total_alo  
##   <chr>                <dbl>  
## 1 01001                  4512  
## 2 01002                 12345  
## 3 01003                  9692  
## 4 01004                  3836  
## 5 01051                  4632  
## 6 01053                  5592  
## 7 01054                  5657  
## 8 01055                  5748  
## 9 01056                  8599  
## 10 01057                 3264  
## # ... with 391 more rows
```

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlüssel
```

```
## # A tibble: 401 x 3
##   Regionalschlüssel total_alo bundesland
##   <chr>                <dbl> <chr>
## 1 01001                  4512 01
## 2 01002                 12345 01
## 3 01003                  9692 01
## 4 01004                  3836 01
## 5 01051                  4632 01
## 6 01053                  5592 01
## 7 01054                  5657 01
## 8 01055                  5748 01
## 9 01056                  8599 01
## 10 01057                 3264 01
## # ... with 391 more rows
```

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlüssel
left_join(., schulden_kombi, by = "Regionalschlues
```

```
## # A tibble: 401 x 6
##   Regionalschlues... total_alo bundesland Schulden_pro_ko... Einwohner
##   <chr>                <dbl> <chr>                <dbl>     <dbl>
## 1 01001                 4512 01                 5791.     87770
## 2 01002                12345 01                 3839.    247135
## 3 01003                 9692 01                 5567.    216739
## 4 01004                 3836 01                 5409.     78759
## 5 01051                 4632 01                 1670.    133684
## 6 01053                 5592 01                 1293.    195677
## 7 01054                 5657 01                 2624.    165642
## 8 01055                 5748 01                 1890.    200931
## 9 01056                 8599 01                 2225.    311713
## 10 01057                 3264 01                 1532.    128763
## # ... with 391 more rows, and 1 more variable: Schulden_gesamt <dbl>
```

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlues
left_join(., schulden_kombi, by = "Regionalschlues
left_join(., bip_kombi, by = "Regionalschlues"))
```

```
## # A tibble: 401 x 8
##   Regionalschlues... total_alo bundesland Schulden_pro_ko... Einwohner
##   <chr>                <dbl> <chr>                <dbl>     <dbl>
## 1 01001                 4512 01                 5791.     87770
## 2 01002                12345 01                 3839.    247135
## 3 01003                 9692 01                 5567.    216739
## 4 01004                 3836 01                 5409.     78759
## 5 01051                 4632 01                 1670.    133684
## 6 01053                 5592 01                 1293.    195677
## 7 01054                 5657 01                 2624.    165642
## 8 01055                 5748 01                 1890.    200931
## 9 01056                 8599 01                 2225.    311713
## 10 01057                3264 01                 1532.    128763
## # ... with 391 more rows, and 3 more variables: Schulden_gesamt <dbl>, bip <dbl>
## #   bip_pro_kopf <dbl>
```

```

# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlues
left_join(., schulden_kombi, by = "Regionalschlues
left_join(., bip_kombi, by = "Regionalschluesse
# Zahl der Erwerbstätigen zumergen
left_join(., erwerb_kombi, by = "Regionalschluesse

```

```

## # A tibble: 401 x 9
##   Regionalschlues... total_alo bundesland Schulden_pro_ko... Einwohner
##   <chr>                <dbl> <chr>                <dbl>     <dbl>
## 1 01001                 4512 01                 5791.     87770
## 2 01002                12345 01                 3839.    247135
## 3 01003                 9692 01                 5567.    216739
## 4 01004                 3836 01                 5409.     78759
## 5 01051                 4632 01                 1670.    133684
## 6 01053                 5592 01                 1293.    195677
## 7 01054                 5657 01                 2624.    165642
## 8 01055                 5748 01                 1890.    200931
## 9 01056                 8599 01                 2225.    311713
## 10 01057                 3264 01                 1532.    128763
## # ... with 391 more rows, and 4 more variables: Schulden_gesamt <dbl>, bip <dbl>
## #   bip_pro_kopf <dbl>, erw <dbl>

```

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlüssel
  left_join(., schulden_kombi, by = "Regionalschlüssel")
  left_join(., bip_kombi, by = "Regionalschlüssel")
# Zahl der Erwerbstätigen zumergen
  left_join(., erwerb_kombi, by = "Regionalschlüssel")
gesamtdaten
```



```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlüssel
  left_join(., schulden_kombi, by = "Regionalschlüssel")
  left_join(., bip_kombi, by = "Regionalschlüssel")
# Zahl der Erwerbstätigen zumergen
  left_join(., erwerb_kombi, by = "Regionalschlüssel")
gesamtdaten

#saveRDS(gesamtdaten, "data/gesamtdaten.rds") #save
```

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlüssel
  left_join(., schulden_kombi, by = "Regionalschlues
  left_join(., bip_kombi, by = "Regionalschlüssel")
# Zahl der Erwerbstätigen zumergen
  left_join(., erwerb_kombi, by = "Regionalschluesse
gesamtdaten

#saveRDS(gesamtdaten, "data/gesamtdaten.rds") #save
#saveRDS(schulden_bereinigt, "data/schulden_bereinig
```

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschlues
  left_join(., schulden_kombi, by = "Regionalschlues
  left_join(., bip_kombi, by = "Regionalschluesel")
# Zahl der Erwerbstätigen zumergen
  left_join(., erwerb_kombi, by = "Regionalschluesse
gesamtdaten

#saveRDS(gesamtdaten, "data/gesamtdaten.rds") #save
#saveRDS(schulden_bereinigt, "data/schulden_bereinig
#saveRDS(bip_zeitreihe, "data/bip_zeitreihe.rds") #
```

Übungsaufgaben

Laden Sie sich das durchschnittliche [Arbeitnehmerentgelt pro Arbeitnehmer und Landkreis](#) auf der Seite der Statistischen Ämter des Bundes und der Länder herunter und lesen Sie diesen in R ein.

- + Finden Sie in dem heruntergeladenen Datensatz heraus, was der Unterschied zwischen *Arbeitnehmerentgelt* und *Bruttolöhne- und Gehälter* ist.
- + Lesen Sie die für Sie relevante Tabelle *Bruttolöhne- und Gehälter* in R ein.
- + Bereinigen Sie die Tabelle, d.h. der Datensatz sollte danach `tidy` sein.
- + Berechnen Sie die Bruttolöhne pro Bundesland mit den Bruttolöhnen der einzelnen Landkreise als Konsistenzcheck.