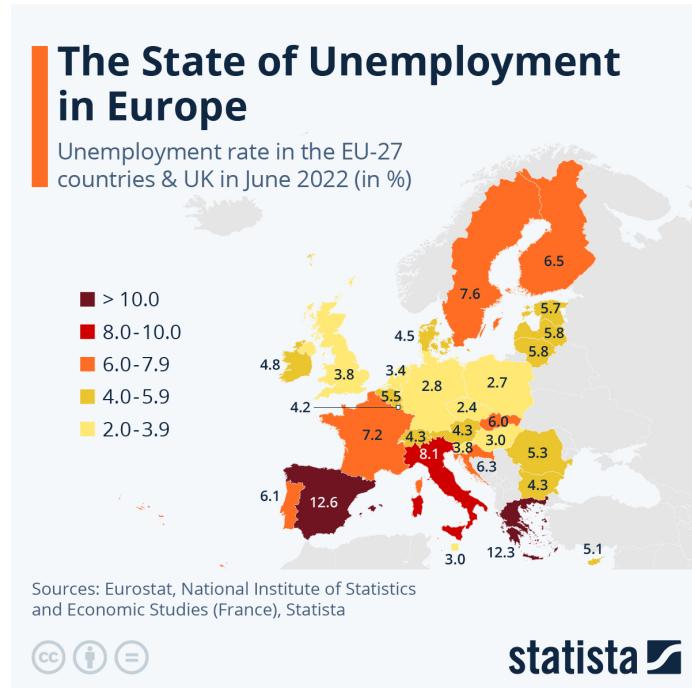


# Case-Study zur Arbeitslosigkeit in Deutschland



# Ziel der Case-Study

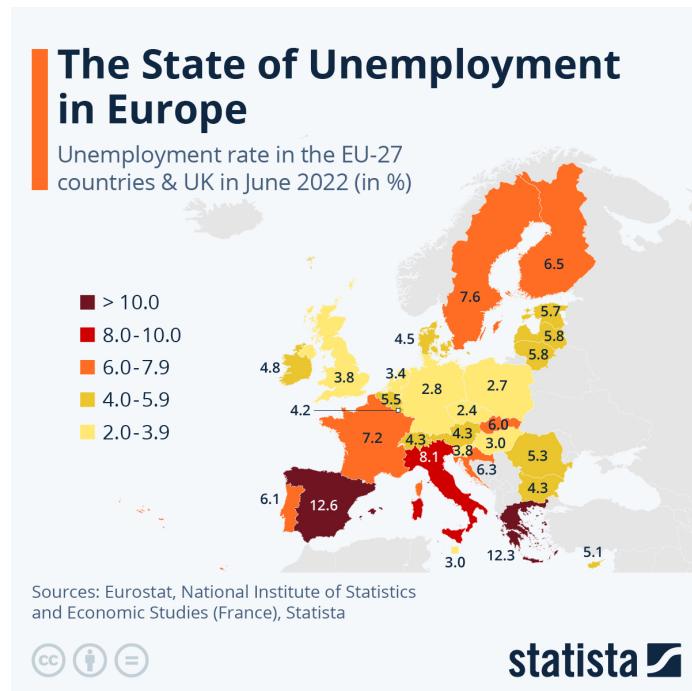
Deutschland hat europaweit eine der niedrigsten Arbeitslosenquoten:



Quelle: [Fleck, A. \(August 11, 2022\). The State of Unemployment in Europe \(Digital image\).](#)

# 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 werden wir in dieser Case-Study auf den Grund gehen.

Quelle: [Fleck, A. \(August 11, 2022\). The State of Unemployment in Europe \(Digital image\).](#)

# 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

In diesem Foliensatz nutze ich die magitr Pipe %>%<sup>1</sup>, da dies mit dem Paket flipbookr nicht anders möglich ist. Es ist jedoch empfehlenswert grundsätzlich die Base R Pipe zu nutzen | >

# 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



# 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!

# Klonen Sie unsere Github Seite

- + Gehen Sie auf die [Github Seite des Projektkurses](#)
- + Klicken Sie auf des grünen "Code" Button
- + Kopieren Sie sich die [angezeigte HTTPS](#)
- + Gehen Sie in Github Desktop und fügen dort die kopierte HTTPS in "Clone a repository" -> "URL"

[Hier eine Step-by-Step Anleitung](#)

Wenn Sie zu Beginn der Woche in Github Desktop auf "Pull" klicken werden alle Vorlesungsinhalte automatisch aktualisiert, d.h. alle Vorlesungsfolien, die Case-Study, Tutorials etc.! 

05 : 00

# Nötige Pakete laden

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

```
## — Attaching core tidyverse packages ————— tidyverse 2.0.0 —
## ✓ dplyr     1.1.4    ✓ readr     2.1.5
## ✓forcats   1.0.0    ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1    ✓ tibble    3.2.1
## ✓ lubridate 1.9.3    ✓ tidyverse  1.3.1
## ✓ purrr    1.0.2
## — Conflicts ————— tidyverse_conflicts() —
## ✘ dplyr::filter() masks stats::filter()
## ✘ dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

# Nötige Pakete laden

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

```
## — Attaching core tidyverse packages ————— tidyverse 2.0.0 —
## ✓ dplyr     1.1.4    ✓ readr     2.1.5
## ✓forcats   1.0.0    ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1    ✓ tibble    3.2.1
## ✓ lubridate 1.9.3    ✓ tidyverse  1.3.1
## ✓ purrr    1.0.2
## — Conflicts ————— tidyverse_conflicts() —
## ✘ dplyr::filter() masks stats::filter()
## ✘ dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Was bedeuten die "Messages" nach dem Laden von library(tidyverse)?

# Daten einlesen

Unterschiedliche Dateien und unterschiedliche Tabellenblätter, was sollten wir verwenden?

```
# Öffnen des ZIP-Archivs
# Es sind zwei Tabellen in dem ZIP Archiv, wir interessieren uns für die Anzahl der Arbeitslosen und wählen c
alo_name <- as.character(unzip("../case-study/data/Arbeitslose_2022.xlsx.zip", list = TRUE)$Name)
alo_name <- alo_name[1]
unzip("../case-study/data/Arbeitslose_2022.xlsx.zip", alo_name)
```

# Daten einlesen

Unterschiedliche Dateien und unterschiedliche Tabellenblätter, was sollten wir verwenden?

```
# Öffnen des ZIP-Archivs
# Es sind zwei Tabellen in dem ZIP Archiv, wir interessieren uns für die Anzahl der Arbeitslosen und wählen c
alo_name <- as.character(unzip("../case-study/data/Arbeitslose_2022.xlsx.zip", list = TRUE)$Name)
alo_name <- alo_name[1]
unzip("../case-study/data/Arbeitslose_2022.xlsx.zip", alo_name)
```

**Vermutung:** Durch Tabellenblatt "Inhaltsverzeichnis" könnten wir schlauer werden

```
alo_inhalt <- read_xlsx(alo_name, sheet = "Inhaltsverzeichnis")
head(alo_inhalt, 15)
```

```
## # A tibble: 15 × 1
##   Inhaltsverzeichnis
##   <chr>
## 1 <NA>
## 2 <NA>
## 3 Arbeitslose - Zeitreihe
## 4 <NA>
## 5 <NA>
## 6 Tabelle
## 7 Bestand an Arbeitslosen
## 8 Kreiszusammenfassung
## 9 Übersicht nach Kreisen
## 10 <NA>
## 11 Insgesamt
## 12 Rechtskreis
## 13 SGB III
## 14 SGB II
## 15 Geschlecht
```

```
alo_inhalt <- read_xlsx(alo_name, sheet = "Inhaltsverzeichnis")
head(alo_inhalt, 15)
```

```
## # A tibble: 15 × 1
##   Inhaltsverzeichnis
##   <chr>
## 1 NA
## 2 NA
## 3 Arbeitslose - Zeitreihe
## 4 NA
## 5 NA
## 6 Tabelle
## 7 Bestand an Arbeitslosen
## 8 Kreiszusammenfassung
## 9 Übersicht nach Kreisen
## 10 NA
## 11 Insgesamt
## 12 Rechtskreis
## 13 SGB III
## 14 SGB II
## 15 Geschlecht
```

```
alo_inhalt <- read_xlsx(alo_name, sheet = "Inhaltsverzeichnis")
head(alo_inhalt, 15)
```

```
## # A tibble: 15 × 1
##   Inhaltsverzeichnis
##   <chr>
## 1 NA
## 2 NA
## 3 Arbeitslose - Zeitreihe
## 4 NA
## 5 NA
## 6 Tabelle
## 7 Bestand an Arbeitslosen
## 8 Kreiszusammenfassung
## 9 Übersicht nach Kreisen
## 10 NA
## 11 Insgesamt
## 12 Rechtskreis
## 13 SGB III
## 14 SGB II
## 15 Geschlecht
```



**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) **aus dem Jahr 2022**
- ✚ 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) **aus dem Jahr 2022**
- + 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) **aus dem Jahr 2022**
- + 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?



**Bestand an Arbeitslosen - Gesamt**

Länder, Regierungsbezirke, Kreise und Gemeinden (Gebietsstand = Datenstand)

Zeitreihe, Datenstand: Februar 2022

**Rechtskreis Insgesamt**

Aus Datenschutzgründen und Gründen der statistischen Geheimhaltung werden Zahlenwerte von 1 oder 2 und Daten, aus denen rechnerisch auf einen solchen Zahlenwert geschlossen werden kann, anonymisiert.

[zurück zum Inhalt](#)

**Jahresdurchschnitte**

**Rechtskreis Insgesamt - Bestand an Arbeitslosen - Gesamt**

Jahresdurchschnitt

Jahresdurchschnitt

Januar 2021

Februar 2021

März 2021

April 2021

Mai 2021

Juni 2021

Juli 2021

August 2021

September 2021

Oktober 2021

November 2021

Region	2021	2022	1	2	3	4	5	6	7	8	9	10	11
Deutschland	2.613.489	2.418.133	2.900.663	2.904.413	2.827.449	2.771.232	2.687.191	2.613.825	2.590.310	2.578.471	2.464.793	2.376.925	2.317.0
01 Schleswig-Holstein	88.865	81.564	98.931	99.931	97.683	95.571	92.043	89.203	89.315	85.183	81.897	79.562	77.8
01001 Flensburg, Stadt	4.369	3.970	4.852	4.790	4.709	4.740	4.602	4.440	4.458	4.226	4.061	3.964	3.7
01002 Kiel, Landeshauptstadt	11.097	10.315	12.070	12.051	11.966	12.072	11.531	11.284	11.356	10.946	10.418	10.084	9.6
01003 Lübeck, Hansestadt	9.347	8.776	10.073	10.113	10.275	10.237	9.903	9.377	9.442	8.932	8.720	8.489	8.2
01004 Neumünster, Stadt	3.771	3.359	4.080	4.214	4.154	4.074	3.927	3.911	3.884	3.679	3.500	3.368	3.2
01051 Dithmarschen	4.143	3.858	4.741	4.828	4.646	4.482	4.277	4.080	4.091	3.940	3.776	3.579	3.5
01053 Herzogtum Lauenburg	5.603	5.351	6.072	6.121	6.043	5.910	5.737	5.647	5.625	5.464	5.285	5.186	5.0
01054 Nordfriesland	4.699	4.155	5.688	5.877	5.633	5.273	4.720	4.394	4.270	4.067	3.881	3.842	4.1
01055 Ostholstein	5.371	4.824	6.598	6.658	6.388	5.887	5.539	4.899	4.943	4.696	4.525	4.498	4.7
01056 Pinneberg	9.371	8.547	10.312	10.477	10.115	9.990	9.786	9.662	9.542	9.030	8.675	8.562	8.1
01057 Plön	2.854	2.572	3.312	3.314	3.109	2.974	2.869	2.819	2.835	2.734	2.624	2.560	2.4
01058 Rendsburg-Eckernförde	6.170	5.705	6.851	6.861	6.626	6.509	6.406	6.235	6.348	6.029	5.773	5.544	5.3
01059 Schleswig-Flensburg	5.567	5.026	6.160	6.227	6.007	5.868	5.648	5.523	5.617	5.454	5.249	5.102	4.9
01060 Segeberg	7.456	6.804	8.124	8.241	8.123	7.976	7.802	7.719	7.594	7.186	6.933	6.693	6.4
01061 Steinburg	4.250	3.851	4.550	4.671	4.572	4.501	4.362	4.381	4.389	4.185	4.075	3.862	3.6
01062 Stormarn	4.798	4.450	5.448	5.488	5.317	5.078	4.934	4.832	4.921	4.615	4.402	4.229	4.1
02 Hamburg	80.395	73.800	86.933	86.962	86.117	85.780	83.895	82.248	82.023	78.572	75.986	73.897	71.3
02000 Hamburg, Freie und Hansestadt	80.395	73.800	86.933	86.962	86.117	85.780	83.895	82.248	82.023	78.572	75.986	73.897	71.3
03 Niedersachsen	243.021	230.553	267.035	269.037	261.259	257.179	249.606	241.996	239.482	242.119	229.605	222.009	217.5
031 Statistische Region Braunschwe	48.055	46.871	52.577	52.814	51.329	50.739	49.127	47.989	47.606	48.148	45.447	44.138	43.2
03101 Braunschweig, Stadt	7.340	7.014	8.081	8.036	7.784	7.711	7.510	7.370	7.257	7.385	6.929	6.827	6.6
03102 Salzgitter, Stadt	5.002	4.944	5.461	5.521	5.329	5.291	5.111	4.972	4.881	4.835	4.720	4.662	4.6
03103 Wolfsburg, Stadt	3.599	3.885	3.749	3.779	3.717	3.752	3.704	3.599	3.598	3.644	3.432	3.409	3.4
03151 Gifhorn	4.150	4.205	4.519	4.646	4.501	4.377	4.203	4.084	4.071	4.131	3.919	3.817	3.7
03153 Goslar	4.527	4.246	5.087	5.171	5.004	4.911	4.768	4.610	4.426	4.397	4.117	3.938	3.8
03154 Helmstedt	3.018	2.796	3.316	3.315	3.246	3.222	3.071	3.010	3.006	2.943	2.882	2.781	2.7
03155 Northeim	3.802	3.733	4.204	4.230	4.112	4.005	3.837	3.762	3.788	3.887	3.589	3.437	3.3
03157 Peine	3.740	3.720	4.063	3.972	3.928	3.940	3.813	3.713	3.748	3.782	3.564	3.513	3.3
03158 Wolfenbüttel	3.038	3.130	3.313	3.316	3.211	3.175	3.074	2.976	3.008	3.081	2.903	2.809	2.7
03159 Göttingen	9.840	9.200	10.784	10.828	10.497	10.355	10.036	9.893	9.823	10.063	9.392	8.945	8.7
032 Statistische Region Hannover	78.518	75.039	83.909	84.980	83.013	82.607	80.810	78.814	77.475	78.366	75.321	73.303	71.7
03241 Region Hannover	48.229	44.948	51.275	51.866	50.915	50.779	49.682	48.460	47.587	48.128	46.484	45.286	44.2
03251 Diepholz	5.362	5.033	5.969	6.023	5.791	5.621	5.440	5.315	5.269	5.317	5.033	4.948	4.7
03252 Hameln-Pyrmont	5.169	5.113	5.615	5.638	5.480	5.536	5.409	5.292	5.044	4.992	4.888	4.734	4.6

# Spezifizieren welche Spalten eingelesen werden sollen

Neben der Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam) **aus dem Jahr 2022** benötigen wir noch die "Gemeinde-ID" und den Gemeindenamen.

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

# Spezifizieren welche Spalten eingelesen werden sollen

Neben der Anzahl aller Arbeitslosen pro Gemeinde (d.h. SGB II und III gemeinsam) **aus dem Jahr 2022** benötigen wir noch die "Gemeinde-ID" und den Gemeindenamen.

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

# Spezifizieren welche Spalten eingelesen werden sollen

count: false

```
alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
```

```

alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
alo_skip
## # A tibble: 11,230 × 27
##   ...1     Jahresdurchschnitte ...3  Rechtskreis Insgesam...¹ ...5 ...6 ...
##   <chr>    <chr>                  <chr> <chr>           <chr> <chr> <chr>
## 1 <NA>      Jahresdurch-schnitt Jahr... 44197          44228 44256 442
## 2 Region     2021                  2022  1              2       3       4
## 3 Deutschla... 2613489            2418... 2900663        2904... 2827... 27...
## 4 01 Schles... 88864.75          8156... 98931        99931 97683 955
## 5 01001 Fle... 4369.166666666667 3970... 4852        4790 4709 474
## 6 01001000 ... 4369.166666666667 3970... 4852        4790 4709 474
## 7 01002 Kie... 11096.833333333334 1031... 12070       12051 11966 120
## 8 01002000 ... 11096.833333333334 1031... 12070       12051 11966 120
## 9 01003 Lüb... 9346.916666666661 8776... 10073       10113 10275 102
## 10 01003000 ... 9346.916666666661 8776... 10073      10113 10275 102
## # i 11,220 more rows
## # i abbreviated name:
## #   `Rechtskreis Insgesamt - Bestand an Arbeitslosen - Gesamt...`¹
## # i 20 more variables: ...8 <chr>, ...9 <chr>, ...10 <chr>, ...11 <chr>,
## #   ...12 <chr>, ...13 <chr>, ...14 <chr>, ...15 <chr>,
## #   `Rechtskreis Insgesamt - Bestand an Arbeitslosen - Gesamt...`¹<chr>,
## #   ...17 <chr>, ...18 <chr>, ...19 <chr>, ...20 <chr>, ...21 <chr>, ...

```

```
alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`))

## # A tibble: 11,230 × 3
##   ...1                Jahresdurchschnitte ...3
##   <chr>              <chr>                  <chr>
## 1 <NA>               Jahresdurch-schnitt  Jahresdurch-schnitt
## 2 Region             2021                  2022
## 3 Deutschland         2613489               2418132.75
## 4 01 Schleswig-Holstein 88864.75            81564.08333333328
## 5 01001 Flensburg, Stadt 4369.16666666667 3970.333333333335
## 6 01001000 Flensburg, Stadt 4369.16666666667 3970.333333333335
## 7 01002 Kiel, Landeshauptstadt 11096.83333333334 10315.166666666666
## 8 01002000 Kiel, Landeshauptstadt 11096.83333333334 10315.166666666666
## 9 01003 Lübeck, Hansestadt 9346.916666666661 8776.416666666661
## 10 01003000 Lübeck, Hansestadt 9346.916666666661 8776.416666666661
## # i 11,220 more rows
```

```
alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  mutate(Regionalschluessel = str_extract(`...1`, "[A-Z].*"))
  Gemeinde = str_extract(`...1`, "[A-Z].*"))
```

```
## # A tibble: 11,230 × 5
##   ...1                Jahresdurchschnitte ...3 Regionalschluessel Gemeinde
##   <chr>              <chr>          <chr>          <chr>
## 1 <NA>               Jahresdurch-schnitt Jahr... <NA>
## 2 Region             2021            2022 <NA>
## 3 Deutschland         2613489         2418... <NA>
## 4 01 Schleswig-Holstein 88864.75      8156... 01
## 5 01001 Flensburg, Stadt 4369.166666666667 3970... 01001
## 6 01001000 Flensburg, St... 4369.166666666667 3970... 01001000
## 7 01002 Kiel, Landeshaup... 11096.833333333334 1031... 01002
## 8 01002000 Kiel, Landesh... 11096.833333333334 1031... 01002000
## 9 01003 Lübeck, Hansesta... 9346.916666666661 8776... 01003
## 10 01003000 Lübeck, Hanse... 9346.916666666661 8776... 01003000
## # i 11,220 more rows
```

```
alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  mutate(Regionalschluessel = str_extract(`...1`, "["
    Gemeinde = str_extract(`...1`, "[A-Z].*"))
  mutate(alo = as.numeric(`...3`))
```

```
## # A tibble: 11,230 × 6
##   ...1           Jahresdurchschnitte ...3  Regionalschluessel Gemeinde     a
##   <chr>          <chr>                  <chr> <chr>           <chr>      <dk>
## 1 <NA>            Jahresdurch-schnitt Jahr... <NA>           <NA>       NA
## 2 Region          2021                  2022  <NA>           Region     2.02
## 3 Deutschland     2613489               2418... <NA>           Deutsch... 2.42
## 4 01 Schleswig-H. 88864.75             8156... 01           Schlesw... 8.16
## 5 01001 Flensbur... 4369.166666666667 3970... 01001       Flensbu... 3.97
## 6 01001000 Flens... 4369.166666666667 3970... 01001000  Flensbu... 3.97
## 7 01002 Kiel, La... 11096.833333333334 1031... 01002       Kiel, L... 1.03
## 8 01002000 Kiel,... 11096.833333333334 1031... 01002000  Kiel, L... 1.03
## 9 01003 Lübeck, ... 9346.916666666661 8776... 01003       Lübeck,... 8.78
## 10 01003000 Lüb... 9346.916666666661 8776... 01003000  Lübeck,... 8.78
## # i 11,220 more rows
```

```
alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  mutate(Regionalschluessel = str_extract(`...1`, "[Gemeinde = str_extract(`...1`, "[A-Z].*"))
mutate(alo = as.numeric(`...3`)) %>%
  select(-c(`...1`, Jahresdurchschnitte, `...3`))

## # A tibble: 11,230 × 3
##   Regionalschluessel Gemeinde      alo
##   <chr>           <chr>        <dbl>
## 1 <NA>            <NA>         NA
## 2 <NA>            Region       2022
## 3 <NA>            Deutschland 2418133.
## 4 01              Schleswig-Holstein 81564.
## 5 01001           Flensburg, Stadt 3970.
## 6 01001000        Flensburg, Stadt 3970.
## 7 01002           Kiel, Landeshauptstadt 10315.
## 8 01002000        Kiel, Landeshauptstadt 10315.
## 9 01003           Lübeck, Hansestadt 8776.
## 10 01003000       Lübeck, Hansestadt 8776.
## # i 11,220 more rows
```

```
alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk
alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  mutate(Regionalschluessel = str_extract(`...1`, "[[  
  Gemeinde = str_extract(`...1`, "[A-Z].*")  
mutate(alo = as.numeric(`...3`)) %>%
  select(-c(`...1`, Jahresdurchschnitte, `...3`)) %>
  filter(!is.na(alo))
```

```
## # A tibble: 11,186 × 3
##   Regionalschluessel Gemeinde      alo
##   <chr>           <chr>        <dbl>
## 1 <NA>            Region       2022
## 2 <NA>            Deutschland  2418133.
## 3 01              Schleswig-Holstein 81564.
## 4 01001          Flensburg, Stadt 3970.
## 5 01001000       Flensburg, Stadt 3970.
## 6 01002          Kiel, Landeshauptstadt 10315.
## 7 01002000       Kiel, Landeshauptstadt 10315.
## 8 01003          Lübeck, Hansestadt 8776.
## 9 01003000       Lübeck, Hansestadt 8776.
## 10 01004         Neumünster, Stadt 3359.
## # i 11,176 more rows
```

```

alo_skip <- read_xlsx(alo_name, sheet = "Gesamt", sk

alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  mutate(Regionalschluessel = str_extract(`...1`, "[[A-Z].*"))
  Gemeinde = str_extract(`...1`, "[A-Z].*"))
  mutate(alo = as.numeric(`...3`)) %>%
  select(-c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  filter(!is.na(alo))

## # A tibble: 11,186 × 3
##   Regionalschluessel Gemeinde      alo
##   <chr>           <chr>        <dbl>
## 1 <NA>            Region       2022
## 2 <NA>            Deutschland 2418133.
## 3 01              Schleswig-Holstein 81564.
## 4 01001           Flensburg, Stadt 3970.
## 5 01001000        Flensburg, Stadt 3970.
## 6 01002           Kiel, Landeshauptstadt 10315.
## 7 01002000        Kiel, Landeshauptstadt 10315.
## 8 01003           Lübeck, Hansestadt 8776.
## 9 01003000        Lübeck, Hansestadt 8776.
## 10 01004          Neumünster, Stadt 3359.
## # i 11,176 more rows

```

#Abspeichern als Datensatz data\_alo

```

data_alo <- alo_skip %>%
  select(c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  mutate(Regionalschluessel = str_extract(`...1`, "[[:digit:]]+"),
  Gemeinde = str_extract(`...1`, "[A-Z].*")) %>%
  mutate(alo = as.numeric(`...3`)) %>%
  select(-c(`...1`, Jahresdurchschnitte, `...3`)) %>%
  filter(!is.na(alo))

data_alo <- data_alo[-c(1,2),]

```

# 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 2022.
  - + zweistelligen Regionalschlüssel
  - + "Buchstaben" für jeden Regionalschlüssel zählen (`nchar()` (number of characters))
- + Alternative Datenquelle: Die Anzahl der Arbeitslosen für das Jahr 2022 unterteilt nach Ländern der Arbeitsagentur
  - + Wichtig: Tabellenblatt 8

```
data_alo
```

```
## # A tibble: 11,184 × 3
##   Regionalschluessel Gemeinde      alo
##   <chr>              <chr>       <dbl>
## 1 01                Schleswig-Holstein 81564.
## 2 01001              Flensburg, Stadt 3970.
## 3 01001000           Flensburg, Stadt 3970.
## 4 01002              Kiel, Landeshauptstadt 10315.
## 5 01002000           Kiel, Landeshauptstadt 10315.
## 6 01003              Lübeck, Hansestadt 8776.
## 7 01003000           Lübeck, Hansestadt 8776.
## 8 01004              Neumünster, Stadt 3359.
## 9 01004000           Neumünster, Stadt 3359.
## 10 01051             Dithmarschen    3858.
## # i 11,174 more rows
```

```
data_alo %>%
```

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

```
## # A tibble: 16 × 3
##   Regionalschluessel Gemeinde      alo
##   <chr>              <chr>       <dbl>
## 1 01                Schleswig-Holstein 81564.
## 2 02                Hamburg        73800.
## 3 03                Niedersachsen 230553.
## 4 04                Bremen         37214.
## 5 05                Nordrhein-Westfalen 668502.
## 6 06                Hessen         164492.
## 7 07                Rheinland-Pfalz 102515.
## 8 08                Baden-Württemberg 223119.
## 9 09                Bayern         235850.
## 10 10               Saarland        33017.
## 11 11               Berlin          179327.
## 12 12               Brandenburg    74242
## 13 13               Mecklenburg-Vorpommern 59571.
## 14 14               Sachsen         118216.
## 15 15               Sachsen-Anhalt 77978.
## 16 16               Thüringen     58172.
```

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

```
## # A tibble: 16 × 3
##   bundesland Gemeinde      alo
##   <chr>     <chr>       <dbl>
## 1 01        Schleswig-Holstein 81564.
## 2 02        Hamburg          73800.
## 3 03        Niedersachsen  230553.
## 4 04        Bremen            37214.
## 5 05        Nordrhein-Westfalen 668502.
## 6 06        Hessen            164492.
## 7 07        Rheinland-Pfalz  102515.
## 8 08        Baden-Württemberg 223119.
## 9 09        Bayern            235850.
## 10 10       Saarland          33017.
## 11 11       Berlin            179327.
## 12 12       Brandenburg      74242
## 13 13       Mecklenburg-Vorpommern 59571.
## 14 14       Sachsen           118216.
## 15 15       Sachsen-Anhalt    77978.
## 16 16       Thüringen         58172.
```

```
# Abspeichern als check_alo_bundesland
```

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

## check\_alo\_bundesland

```
## # A tibble: 16 × 3
##   bundesland Gemeinde      alo
##   <chr>     <chr>       <dbl>
## 1 01        Schleswig-Holstein 81564.
## 2 02        Hamburg          73800.
## 3 03        Niedersachsen  230553.
## 4 04        Bremen            37214.
## 5 05        Nordrhein-Westfalen 668502.
## 6 06        Hessen            164492.
## 7 07        Rheinland-Pfalz   102515.
## 8 08        Baden-Württemberg 223119.
## 9 09        Bayern            235850.
## 10 10       Saarland          33017.
## 11 11       Berlin            179327.
## 12 12       Brandenburg      74242
## 13 13       Mecklenburg-Vorpommern 59571.
## 14 14       Sachsen           118216.
## 15 15       Sachsen-Anhalt   77978.
## 16 16       Thüringen        58172.
```

## Deutschland und Länder

Berichtsjahr: 2022

Region	Insgesamt		B
	absolut	Anteil in %	
	1	2	
Deutschland	2.418.133	100	
Westdeutschland	1.850.626	76,5	
Ostdeutschland	567.507	23,5	
01 Schleswig-Holstein	81.564	3,4	
02 Hamburg	73.800	3,1	
03 Niedersachsen	230.553	9,5	
04 Bremen	37.214	1,5	
05 Nordrhein-Westfalen	668.502	27,6	
06 Hessen	164.492	6,8	
07 Rheinland-Pfalz	102.515	4,2	
08 Baden-Württemberg	223.119	9,2	
09 Bayern	235.851	9,8	
10 Saarland	33.017	1,4	
11 Berlin	179.327	7,4	
12 Brandenburg	74.242	3,1	
13 Mecklenburg-Vorpommern	59.571	2,5	
14 Sachsen	118.216	4,9	
15 Sachsen-Anhalt	77.978	3,2	
16 Thüringen	58.172	2,4	

## check\_alo\_bundesland

```
## # A tibble: 16 × 3
##   bundesland Gemeinde      alo
##   <chr>     <chr>       <dbl>
## 1 01        Schleswig-Holstein 81564.
## 2 02        Hamburg          73800.
## 3 03        Niedersachsen  230553.
## 4 04        Bremen            37214.
## 5 05        Nordrhein-Westfalen 668502.
## 6 06        Hessen            164492.
## 7 07        Rheinland-Pfalz   102515.
## 8 08        Baden-Württemberg 223119.
## 9 09        Bayern            235850.
## 10 10       Saarland          33017.
## 11 11       Berlin            179327.
## 12 12       Brandenburg      74242
## 13 13       Mecklenburg-Vorpommern 59571.
## 14 14       Sachsen           118216.
## 15 15       Sachsen-Anhalt   77978.
## 16 16       Thüringen        58172.
```

## Deutschland und Länder

Berichtsjahr: 2022

Region	B	
	Insgesamt	
	absolut 1	Anteil in % 2
Deutschland	2.418.133	100
Westdeutschland	1.850.626	76,5
Ostdeutschland	567.507	23,5
01 Schleswig-Holstein	81.564	3,4
02 Hamburg	73.800	3,1
03 Niedersachsen	230.553	9,5
04 Bremen	37.214	1,5
05 Nordrhein-Westfalen	668.502	27,6
06 Hessen	164.492	6,8
07 Rheinland-Pfalz	102.515	4,2
08 Baden-Württemberg	223.119	9,2
09 Bayern	235.851	9,8
10 Saarland	33.017	1,4
11 Berlin	179.327	7,4
12 Brandenburg	74.242	3,1
13 Mecklenburg-Vorpommern	59.571	2,5
14 Sachsen	118.216	4,9
15 Sachsen-Anhalt	77.978	3,2
16 Thüringen	58.172	2,4

Beide Datenreihen sind identisch

# 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(Regionalschluessel) == 8) %>%
  mutate(landkreis = str_extract(Regionalschluessel, "^.{5}"),
        bundesland = str_extract(Regionalschluessel, "^.{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(Regionalschluessel = landkreis)
```

```
data_alo
```

```
## # A tibble: 11,184 × 3
##   Regionalschlüssel Gemeinde      alo
##   <chr>           <chr>       <dbl>
## 1 01              Schleswig-Holstein 81564.
## 2 01001           Flensburg, Stadt 3970.
## 3 01001000        Flensburg, Stadt 3970.
## 4 01002           Kiel, Landeshauptstadt 10315.
## 5 01002000        Kiel, Landeshauptstadt 10315.
## 6 01003           Lübeck, Hansestadt 8776.
## 7 01003000        Lübeck, Hansestadt 8776.
## 8 01004           Neumünster, Stadt 3359.
## 9 01004000        Neumünster, Stadt 3359.
## 10 01051          Dithmarschen    3858.
## # i 11,174 more rows
```

```
data_alo %>%
```

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

```
## # A tibble: 10,745 × 3
##   Regionalschlüssel Gemeinde      alo
##   <chr>           <chr>       <dbl>
## 1 01001000     Flensburg, Stadt 3970.
## 2 01002000     Kiel, Landeshauptstadt 10315.
## 3 01003000     Lübeck, Hansestadt  8776.
## 4 01004000     Neumünster, Stadt  3359.
## 5 01051001     Albersdorf        95.1 
## 6 01051002     Arkebek            3.92 
## 7 01051003     Averlak             10    
## 8 01051004     Bargenstedt        7.42 
## 9 01051005     Barkenholm         3      
## 10 01051006    Barlt               14.7  
## # i 10,735 more rows
```

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

```
## # A tibble: 10,745 × 4
##   Regionalschluessel Gemeinde      alo landkreis
##   <chr>           <chr>        <dbl> <chr>
## 1 01001000    Flensburg, Stadt  3970. 01001
## 2 01002000    Kiel, Landeshauptstadt 10315. 01002
## 3 01003000    Lübeck, Hansestadt  8776. 01003
## 4 01004000    Neumünster, Stadt  3359. 01004
## 5 01051001    Albersdorf          95.1  01051
## 6 01051002    Arkebek              3.92  01051
## 7 01051003    Averlak                10    01051
## 8 01051004    Bargenstedt         7.42  01051
## 9 01051005    Barkenholm            3    01051
## 10 01051006   Barlt                 14.7  01051
## # i 10,735 more rows
```

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

```
## # A tibble: 10,745 × 5
##   Regionalschluessel Gemeinde          alo landkreis bundesland
##   <chr>              <chr>           <dbl> <chr>    <chr>
## 1 01001000 Flensburg, Stadt     3970. 01001    01
## 2 01002000 Kiel, Landeshauptstadt 10315. 01002    01
## 3 01003000 Lübeck, Hansestadt   8776. 01003    01
## 4 01004000 Neumünster, Stadt   3359. 01004    01
## 5 01051001 Albersdorf        95.1  01051    01
## 6 01051002 Arkebek            3.92  01051    01
## 7 01051003 Averlak             10    01051    01
## 8 01051004 Bargenstedt       7.42  01051    01
## 9 01051005 Barkenholm        3    01051    01
## 10 01051006 Barlt              14.7  01051   01
## # i 10,735 more rows
```

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

```
alo_meta
```

```
## # A tibble: 10,745 × 5
##   Regionalschlüssel Gemeinde      alo landkreis bundesland
##   <chr>           <chr>          <dbl> <chr>    <chr>
## 1 01001000       Flensburg, Stadt 3970. 01001    01
## 2 01002000       Kiel, Landeshauptstadt 10315. 01002    01
## 3 01003000       Lübeck, Hansestadt 8776. 01003    01
## 4 01004000       Neumünster, Stadt 3359. 01004    01
## 5 01051001       Albersdorf        95.1  01051    01
## 6 01051002       Arkebek           3.92   01051    01
## 7 01051003       Averlak            10    01051    01
## 8 01051004       Bargenstedt        7.42   01051    01
## 9 01051005       Barkenholm        3     01051    01
## 10 01051006      Barlt              14.7   01051   01
## # i 10,735 more rows
```

```
alo_meta %>%
  group_by(bundesland)

## # A tibble: 10,745 × 5
## # Groups:   bundesland [16]
## # ... with 16 more variables:
## #   Regionalschlüssel     <chr>,
## #   Gemeinde               <chr>,
## #   alo                    <dbl>,
## #   landkreis              <chr>,
## #   bundesland              <chr>
## #   ...
## #   1 01001000  Flensburg, Stadt      3970. 01001 01
## #   2 01002000  Kiel, Landeshauptstadt 10315. 01002 01
## #   3 01003000  Lübeck, Hansestadt    8776. 01003 01
## #   4 01004000  Neumünster, Stadt     3359. 01004 01
## #   5 01051001  Albersdorf           95.1   01051 01
## #   6 01051002  Arkebek              3.92   01051 01
## #   7 01051003  Averlak                10    01051 01
## #   8 01051004  Bargenstedt          7.42   01051 01
## #   9 01051005  Barkenholm           3    01051 01
## #  10 01051006 Barlt                 14.7   01051 01
## # ... with 10,735 more rows
```

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

```
## # A tibble: 16 × 2
##   bundesland total_alo
##   <chr>        <dbl>
## 1 01          81564.
## 2 02          73800.
## 3 03         230553.
## 4 04         37214.
## 5 05         668502.
## 6 06         164492.
## 7 07         102515.
## 8 08         223119.
## 9 09         235850.
## 10 10        33017.
## 11 11        179327.
## 12 12        74242
## 13 13        59571.
## 14 14       118216.
## 15 15        77978.
## 16 16        58172.
```

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

```
alo_meta
```

```
## # A tibble: 10,745 × 5
##   Regionalschlüssel Gemeinde      alo landkreis bundesland
##   <chr>           <chr>          <dbl> <chr>    <chr>
## 1 01001000       Flensburg, Stadt 3970. 01001    01
## 2 01002000       Kiel, Landeshauptstadt 10315. 01002    01
## 3 01003000       Lübeck, Hansestadt 8776. 01003    01
## 4 01004000       Neumünster, Stadt 3359. 01004    01
## 5 01051001       Albersdorf        95.1  01051    01
## 6 01051002       Arkebek           3.92   01051    01
## 7 01051003       Averlak            10    01051    01
## 8 01051004       Bargenstedt        7.42   01051    01
## 9 01051005       Barkenholm         3    01051    01
## 10 01051006      Barlt              14.7  01051    01
## # i 10,735 more rows
```

```
alo_meta %>%
  group_by(landkreis)

## # A tibble: 10,745 × 5
## # Groups:   landkreis [400]
##   Regionalschlüssel Gemeinde
##   <chr>           <chr>
## 1 01001000       Flensburg, Stadt
## 2 01002000       Kiel, Landeshauptstadt
## 3 01003000       Lübeck, Hansestadt
## 4 01004000       Neumünster, Stadt
## 5 01051001       Albersdorf
## 6 01051002       Arkebek
## 7 01051003       Averlak
## 8 01051004       Bargenstedt
## 9 01051005       Barkenholm
## 10 01051006      Barlt
## # i 10,735 more rows
```

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

```
## # A tibble: 400 × 2
##   landkreis total_alo
##   <chr>        <dbl>
## 1 01001       3970.
## 2 01002      10315.
## 3 01003       8776.
## 4 01004       3359.
## 5 01051       3858.
## 6 01053       5351.
## 7 01054       4155.
## 8 01055       4824.
## 9 01056       8547.
## 10 01057      2572.
## # i 390 more rows
```

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

## # A tibble: 400 × 2
##       Regionalschluessel total_alo
##   <chr>                <dbl>
## 1 01001                 3970.
## 2 01002                 10315.
## 3 01003                 8776.
## 4 01004                 3359.
## 5 01051                 3858.
## 6 01053                 5351.
## 7 01054                 4155.
## 8 01055                 4824.
## 9 01056                 8547.
## 10 01057                2572.
## # i 390 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_bundeland`: Auf Bundesland aggregierte Zahlen der Arbeitslosigkeit aus den Gemeinden
- ✚ Datessatz `alo_bundesland`: Die schon von der Arbeitsagentur aggregierte Zahlen in unserem Datensatz

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

```
## # A tibble: 16 × 4
##   bundesland Gemeinde      alo total_alo
##   <chr>       <chr>     <dbl>     <dbl>
## 1 01         Schleswig-Holstein 81564.    81564.
## 2 02         Hamburg        73800.    73800.
## 3 03         Niedersachsen 230553.   230553.
## 4 04         Bremen          37214.    37214.
## 5 05         Nordrhein-Westfalen 668502.   668502.
## 6 06         Hessen          164492.   164492.
## 7 07         Rheinland-Pfalz 102515.   102515.
## 8 08         Baden-Württemberg 223119.   223119.
## 9 09         Bayern          235850.   235850.
## 10 10        Saarland        33017.    33017.
## 11 11        Berlin          179327.   179327.
## 12 12        Brandenburg    74242     74242
## 13 13        Mecklenburg-Vorpommern 59571.   59571.
## 14 14        Sachsen          118216.   118216.
## 15 15        Sachsen-Anhalt  77978.    77978.
## 16 16        Thüringen       58172.    58172.
```

```
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 × 4  
## bundesland Gemeinde alo total\_alo  
## <chr> <chr> <dbl> <dbl>  
## 1 01 Schleswig-Holstein 81564. 81564.  
## 2 02 Hamburg 73800. 73800.  
## 3 03 Niedersachsen 230553. 230553.  
## 4 04 Bremen 37214. 37214.  
## 5 05 Nordrhein-Westfalen 668502. 668502.  
## 6 06 Hessen 164492. 164492.  
## 7 07 Rheinland-Pfalz 102515. 102515.  
## 8 08 Baden-Württemberg 223119. 223119.  
## 9 09 Bayern 235850. 235850.  
## 10 10 Saarland 33017. 33017.  
## 11 11 Berlin 179327. 179327.  
## 12 12 Brandenburg 74242 74242  
## 13 13 Mecklenburg-Vorpommern 59571. 59571.  
## 14 14 Sachsen 118216. 118216.  
## 15 15 Sachsen-Anhalt 77978. 77978.  
## 16 16 Thüringen 58172. 58172.

```
left_join(check_alo_bundesland, alo_bundesland, by =  
  check_consistency  
check_consistency %>%  
  mutate(diff = alo - total_alo)  
  
## # A tibble: 16 × 5  
##   bundesland Gemeinde      alo total_alo    diff  
##   <chr>     <chr>     <dbl>     <dbl>    <dbl>  
## 1 01       Schleswig-Holstein 81564.    81564.    0  
## 2 02       Hamburg        73800.    73800.    0  
## 3 03       Niedersachsen 230553.   230553.    0  
## 4 04       Bremen         37214.    37214. -7.28e-12  
## 5 05       Nordrhein-Westfalen 668502.   668502.    0  
## 6 06       Hessen         164492.   164492.    0  
## 7 07       Rheinland-Pfalz 102515.   102515.    0  
## 8 08       Baden-Württemberg 223119.   223119.    0  
## 9 09       Bayern          235850.   235850.    0  
## 10 10      Saarland        33017.    33017.    0  
## 11 11      Berlin          179327.   179327.    0  
## 12 12      Brandenburg    74242     74242     0  
## 13 13      Mecklenburg-Vorpommern 59571.   59571.    0  
## 14 14      Sachsen         118216.   118216.    0  
## 15 15      Sachsen-Anhalt  77978.    77978.    0  
## 16 16      Thüringen      58172.    58172.    0
```

```

left_join(check_alo_bundesland, alo_bundesland, by =
  check_consistency
check_consistency %>%
  mutate(diff = alo - total_alo)
## # A tibble: 16 × 5
##   bundesland Gemeinde      alo total_alo    diff
##   <chr>     <chr>      <dbl>    <dbl>    <dbl>
## 1 01        Schleswig-Holstein 81564.  81564.    0
## 2 02        Hamburg       73800.  73800.    0
## 3 03        Niedersachsen 230553. 230553.    0
## 4 04        Bremen        37214.  37214. -7.28e-12
## 5 05        Nordrhein-Westfalen 668502. 668502.    0
## 6 06        Hessen        164492. 164492.    0
## 7 07        Rheinland-Pfalz 102515. 102515.    0
## 8 08        Baden-Württemberg 223119. 223119.    0
## 9 09        Bayern        235850. 235850.    0
## 10 10       Saarland      33017.  33017.    0
## 11 11       Berlin         179327. 179327.    0
## 12 12       Brandenburg   74242.  74242.    0
## 13 13       Mecklenburg-Vorpommern 59571. 59571.    0
## 14 14       Sachsen        118216. 118216.    0
## 15 15       Sachsen-Anhalt 77978.  77978.    0
## 16 16       Thüringen     58172.  58172.    0

```

Es bestehen keine Unstimmigkeiten.

# Pro-Kopf Verschuldung

# Pro-Kopf Verschuldung auf Gemeindeebene

- ✚ Auf Gemeindeebene aus dem Jahr 2022
- ✚ Querschnittsdaten
- ✚ Vom Statistischen Bundesamt direkt als Excel-Tabelle heruntergeladen (✓)

## Welche Tabellenblätter sollten wir nutzen?

```
excel_sheets("../case-study/data/Schulden_2022.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 müssen wir potentiell eine Operation über mehrere Tabellenblättern anwenden, z.B. durch eine Schleife

# Mehrere Tabellenblätter einlesen

- + Nicht alle Informationen in **einem Tabellenblatt** enthalten
  - + Viele separate Tabellenblätter
  - + Hier müssen wir potentiell eine Operation über mehrere Tabellenblättern anwenden, z.B. durch eine Schleife

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

# Mehrere Tabellenblätter einlesen

```
sh <- read_xlsx("../case-study/data/Schulden_2022.xlsx", sheet = "SH")
head(sh, 20)
```

```
## # A tibble: 20 × 21
##   `Zurück zum Inhalt...` ...1` ...2` ...3` ...4` ...5` ...6` ...7` ...8` ...9` ...10` 
##   <chr>                  <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 <NA>                  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 2 <NA>                  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 3 "Tabelle 1: Schulde... <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 4 "nach Höhe der Beteili... <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 5 "Regional-\r\nschlüsse... Geme... Verw... "Ein... Schu... Verä... "Sch... Schu... <NA>  <NA>
## 6 <NA>                  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  zusa... Verä... Schu...
## 7 <NA>                  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 8 <NA>                  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 9 <NA>                  <NA>  <NA>  <NA>  EUR  %    "EUR" <NA>  %    EUR
## 10 <NA>                 <NA>  <NA>  <NA>  1    2    "3"   4    5    6
## 11 "010010000000"       Flen... krei... "919... 4617... 1.6   "501... 2230... -3.3  3571...
## 12 "010020000000"       Kiel... krei... "246... 1101... 5.2   "446... 5924... 7.1   5921...
## 13 "010030000000"       Lübe... krei... "217... 9835... -7.5  "451... 3582... -16.8 3549...
## 14 "010040000000"       Neum... krei... "798... 5036... 11   "630... 1245... 9.1   1142...
## 15 "01051"               Krei... Krei... "{13... 5767... 31   "427... 3696... 79.4  3693...
```

# 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 1-7
schulden_individuell <- read_xlsx("../case-study/data/Schulden_2022.xlsx", sheet = "SH", skip = 5) [1:7]
# Umbenennen der ersten 7 Spalten
colnames(schulden_individuell) <- c("Regionalschlüssel", "Gemeinde",
                                      "Verwaltungsform", "Einwohner", "Schulden_gesamt", "Veraenderung_Vorjahr")

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

```
# Einlesen des Tabellenblattes "SH" ohne die ersten
read_xlsx("../case-study/data/Schulden_2022.xlsx", s
## # A tibble: 1,310 × 7
##   `Regional-\\r\\nschlüssel` `Gemeinde/Gemeindeverband` Verwaltungsform
##   <chr>                  <chr>                      <chr>
## 1 <NA>                   <NA>                      <NA>
## 2 <NA>                   <NA>                      <NA>
## 3 <NA>                   <NA>                      <NA>
## 4 <NA>                   <NA>                      <NA>
## 5 <NA>                   <NA>                      <NA>
## 6 010010000000          Flensburg, Stadt      kreisfreie Stadt
## 7 010020000000          Kiel, Landeshauptstadt kreisfreie Stadt
## 8 010030000000          Lübeck, Hansestadt    kreisfreie Stadt
## 9 010040000000          Neumünster, Stadt    kreisfreie Stadt
## 10 01051                 Kreisverwaltung Dithmarschen Kreisverwaltung
## # i 1,300 more rows
## # i 4 more variables: `Einwohner/-innen\\r\\nam\\r\\n30.06.2022` <chr>,
## #   `Schulden des öffentlichen Bereichs insgesamt` <chr>,
## #   `Veränderung zum Vorjahr` <chr>, `Schulden je \\r\\nEinwohner/-in` <chr>
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten  
read_xlsx("../case-study/data/Schulden_2022.xlsx", s  
schulden_individuell
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten  
read_xlsx("../case-study/data/Schulden_2022.xlsx", s  
schulden_individuell  
  
# Umbenennen der ersten 7 Spalten  
colnames(schulden_individuell) <- c("Regionalschlues  
"Verwaltungsform
```

```
# Einlesen des Tabellenblattes "SH" ohne die ersten
read_xlsx("../case-study/data/Schulden_2022.xlsx", s
schulden_individuell

# Umbenennen der ersten 7 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("../case-study/data/Schulden_2022.xlsx", s
schulden_individuell

# Umbenennen der ersten 7 Spalten
colnames(schulden_individuell) <- c("Regionalschlues
"Verwaltungsform

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

schulden_individuell

```

```

## # A tibble: 1,310 × 8
##   Regionalschlüssel Gemeinde Verwaltungsform Einwohner Schulden_gesam
##   <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 Flensburg, Stadt kreisfreie Sta... 91992  461784578.970
## 7 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712  1101148771.960
## 8 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799  983573069.740
## 9 010040000000 Neumünster, Sta... kreisfreie Sta... 79889   503685790.819
## 10 01051          Kreisverwaltung... Kreisverwaltung {135 009} 57672347.289
## # i 1,300 more rows
## # i 3 more variables: Veraenderung_Vorjahr <chr>, Schulden_pro_kopf <chr>,
## #   Bundesland <chr>

```

# Mehrere Tabellenblätter einlesen

Formen Sie Teams von zwei Personen und erarbeiten Sie auf Basis des Beispiels für ein Bundesland einen Code um alle Bundesländer in R einzulesen.

Nutzen Sie für diese Aufgabe **bwGPT**.

Posten Sie ihren Code hier:



10 : 00

# Variablen umformen

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

```
## # A tibble: 15 × 8
##   Regionalschlüssel 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  Flensburg, Stadt kreisfreie Sta... 91992    461784578.9700...
## 7 010020000000  Kiel, Landeshau... kreisfreie Sta... 246712   1101148771.960...
## 8 010030000000  Lübeck, Hansest... kreisfreie Sta... 217799   983573069.7400...
## 9 010040000000  Neumünster, Sta... kreisfreie Sta... 79889    503685790.8199...
## 10 01051          Kreisverwaltung... Kreisverwaltung {135 009} 57672347.28999...
## 11 010510011011  Brunsbüttel, St... amtsfreie Geme... 12518    57881948.27000...
## 12 010510044044  Heide, Stadt     amtsfreie Geme... 21919    49208227.03999...
## 13 010515163     Amtsverwaltung ... Amtsverwaltung {15 770} 825202.43
## 14 010515163003  Averlak          amtsangehörige... 572     2247182.660000...
## 15 010515163010  Brickeln        amtsangehörige... 202     1435532.829999...
## # i 3 more variables: Veraenderung_Vorjahr <chr>, Schulden_pro_kopf <chr>,
```

# Variablen umformen

Wir sehen, es gibt immer noch einige Probleme:

- + Die Werte unserer Variablen stehen nicht direkt unter dem Variablenamen
  - + Dies können wir am einfachsten bereinigen indem wir alle \NAs im Regionalschlüssel entfernen

# Variablen umformen

Wir sehen, es gibt immer noch einige Probleme:

- + Die Werte unserer Variablen stehen nicht direkt unter dem Variablenamen
  - + Dies können wir am einfachsten bereinigen indem wir alle NAs im Regionalschlüssel entfernen
- + Die Variablen "Einwohner", "Schulden\_gesamt" und "Schulden\_pro\_Kopf" sind alle als character hinterlegt (<chr> unter dem Variablenamen in der vorherigen Tabelle)
  - + Beispiel warum Klasse character (Zeile 28): Es sind geschweifte Klammern enthalten

```
schulden_individuell[28, ]
```

```
## # A tibble: 1 × 8
##   Regionalschluessel    Gemeinde      Verwaltungsform Einwohner Schulden_gesamt
##   <chr>                  <chr>          <chr>           <chr>       <chr>
## 1 010515163_Summe(Amt) Amt Burg-St. M... Amtsgebiet     {15 770}  {50 131 118}
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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 Variablenamen
  - + Dies können wir am einfachsten bereinigen indem wir alle NAs im Regionalschlüssel entfernen
- + Die Variablen "Einwohner", "Schulden\_gesamt" und "Schulden\_pro\_Kopf" sind alle als character hinterlegt (<chr> unter dem Variablenamen in der vorherigen Tabelle)
  - + Beispiel warum Klasse character (Zeile 28): Es sind geschweifte Klammern enthalten

```
schulden_individuell[28, ]
```

```
## # A tibble: 1 × 8
##   Regionalschluessel    Gemeinde      Verwaltungsform Einwohner Schulden_gesamt
##   <chr>                 <chr>          <chr>           <chr>       <chr>
## 1 010515163_Summe(Amt) Amt Burg-St. M... Amtsgebiet     {15 770}  {50 131 118}
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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 gezog  
schulden_individuell
```

```
## # A tibble: 13,266 × 8  
##   Regionalschlüssel 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>      <NA>  
## 6 010010000000 Flensburg, Stadt kreisfreie Sta... 91992 461784578.970  
## 7 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712 1101148771.960  
## 8 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799 983573069.740  
## 9 010040000000 Neumünster, Sta... kreisfreie Sta... 79889 503685790.819  
## 10 01051          Kreisverwaltung... Kreisverwaltung {135 009} 57672347.289  
## # i 13,256 more rows  
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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(Regionalschluessel))
```

```
## # A tibble: 13,149 × 8
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>           <chr>     <chr>           <chr>      <chr>
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992    461784578.970
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712   1101148771.960
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799   983573069.740
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889    503685790.810
## 5 01051
Kreisverwaltung... Kreisverwaltung {135 009} 57672347.2890
## 6 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518    57881948.2700
## 7 010510044044 Heide, Stadt      amtsfreie Geme... 21919    49208227.0390
## 8 010515163
Amtsverwaltung ... Amtsverwaltung {15 770} 825202.430
## 9 010515163003 Averlak          amtsangehörige... 572     2247182.6600
## 10 010515163010 Brickeln        amtsangehörige... 202    1435532.8299
## # i 13,139 more rows
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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(Regionalschluessel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
## # A tibble: 13,149 × 8
##   Regionalschluessel Gemeinde      Verwaltungsform Einwohner Schulden_gesamt
##   <chr>              <chr>          <chr>          <chr>          <dbl>
## 1 010010000000    Flensburg, Stadt kreisfreie Sta... 91992        4617845
## 2 010020000000    Kiel, Landeshau... kreisfreie Sta... 246712       11011487
## 3 010030000000    Lübeck, Hansest... kreisfreie Sta... 217799       9835730
## 4 010040000000    Neumünster, Sta... kreisfreie Sta... 79889        5036857
## 5 01051                 Kreisverwaltung... Kreisverwaltung {135 009} 5767234
## 6 010510011011    Brunsbüttel, St... amtsfreie Geme... 12518        5788194
## 7 010510044044    Heide, Stadt      amtsfreie Geme... 21919       4920822
## 8 010515163          Amtsverwaltung ... Amtsverwaltung {15 770} 82520
## 9 010515163003    Averlak           amtsangehörige... 572        224718
## 10 010515163010    Brickeln         amtsangehörige... 202       143551
## # i 13,139 more rows
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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(Regionalschluessel)) %>%  
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam  
  mutate(Einwohner = as.numeric(Einwohner))
```

```
## # A tibble: 13,149 × 8  
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesamt  
##   <chr>           <chr>    <chr>          <dbl>        <dbl>  
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992 4617845  
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712 11011487  
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799 9835730  
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889 5036857  
## 5 01051 Kreisverwaltung... Kreisverwaltung NA 5767234  
## 6 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518 5788194  
## 7 010510044044 Heide, Stadt amtsfreie Geme... 21919 4920822  
## 8 010515163 Amtsverwaltung ... Amtsverwaltung NA 82520  
## 9 010515163003 Averlak amtsangehörige... 572 224718  
## 10 010515163010 Brickeln amtsangehörige... 202 143551  
## # i 13,139 more rows  
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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(Regionalschluessel)) %>%
  mutate(Schulden_gesamt = as.numeric(Schulden_gesam
  mutate(Einwohner = as.numeric(Einwohner)) %>%
  mutate(Schulden_pro_kopf = as.numeric(Schulden_pro

## # A tibble: 13,149 × 8
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>           <chr>     <chr>           <dbl>          <dbl>
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992        4617845
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712       11011487
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799       9835730
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889        5036857
## 5 01051 Kreisverwaltung... Kreisverwaltung NA        5767234
## 6 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518        5788194
## 7 010510044044 Heide, Stadt amtsfreie Geme... 21919        4920822
## 8 010515163 Amtsverwaltung ... Amtsverwaltung NA        82520
## 9 010515163003 Averlak amtsangehörige... 572        224718
## 10 010515163010 Brickeln amtsangehörige... 202        143551
## # i 13,139 more rows
## # i 3 more variables: Veraenderung_Vorjahr <chr>, 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(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,
## # A tibble: 13,149 × 9
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>           <chr>    <chr>          <dbl>        <dbl>
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992        4617845
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712       11011487
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799       9835730
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889        5036857
## 5 01051                 Kreisverwaltung... Kreisverwaltung     NA        5767234
## 6 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518        5788194
## 7 010510044044 Heide, Stadt      amtsfreie Geme... 21919       4920822
## 8 010515163            Amtsverwaltung ... Amtsverwaltung     NA        82520
## 9 010515163003 Averlak             amtsangehörige... 572        224718
## 10 010515163010 Brickeln           amtsangehörige... 202        143551
## # i 13,139 more rows
## # i 4 more variables: Veraenderung_Vorjahr <chr>, 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,  
  select(-Veraenderung_Vorjahr)
```

```
## # A tibble: 13,149 × 8  
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesa  
##   <chr>           <chr>    <chr>        <dbl>       <dbl>  
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992 4617845  
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712 11011487  
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799 9835730  
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889 5036857  
## 5 01051 Kreisverwaltung... Kreisverwaltung NA 5767234  
## 6 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518 5788194  
## 7 010510044044 Heide, Stadt amtsfreie Geme... 21919 4920822  
## 8 010515163 Amtsverwaltung ... Amtsverwaltung NA 82520  
## 9 010515163003 Averlak amtsangehörige... 572 224718  
## 10 010515163010 Brickeln amtsangehörige... 202 143551  
## # i 13,139 more rows  
## # i 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,  
  select(-Veraenderung_Vorjahr) %>%  
#manche Landkreise haben keine Infos zu den Einwohne  
  filter( !is.na( Einwohner ))
```

```
## # A tibble: 10,782 × 8  
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesa  
##   <chr>           <chr>    <chr>        <dbl>       <dbl>  
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992 4617845  
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712 11011487  
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799 9835730  
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889 5036857  
## 5 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518 5788194  
## 6 010510044044 Heide, Stadt amtsfreie Geme... 21919 4920822  
## 7 010515163003 Averlak      amtsangehörige... 572 224718  
## 8 010515163010 Brickeln     amtsangehörige... 202 143551  
## 9 010515163012 Buchholz     amtsangehörige... 998 367565  
## 10 010515163016 Burg (Dithmarsc... amtsangehörige... 4212 143772  
## # i 10,772 more rows  
## # i 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,  
    select(-Veraenderung_Vorjahr) %>%  
#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,
  select(-Veraenderung_Vorjahr) %>%
#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 `gsub( "[{}]" )`), 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

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

# Interne Validität Schulden pro Kopf

✚ `Schulden_pro_Kopf_new` von Hand berechnen

✚ Beachte:

- ✚ Geschweiften Klammern entfernen bei `Schulden_gesamt` (mit `gsub( "[{}]" )`), 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

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

Was macht `gsub()` und `str_extract()` hier genau? Und was sind reguläre Ausdrücke?

```
# Erstellen der Vergleichstabelle
schulden_individuell
```

```
## # A tibble: 13,266 × 8
##   Regionalschlüssel 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 Flensburg, Stadt kreisfreie Sta... 91992  461784578.970
## 7 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712  1101148771.960
## 8 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799  983573069.740
## 9 010040000000 Neumünster, Sta... kreisfreie Sta... 79889   503685790.819
## 10 01051          Kreisverwaltung... Kreisverwaltung {135 009} 57672347.289
## # i 13,256 more rows
## # i 3 more variables: Veraenderung_Vorjahr <chr>, Schulden_pro_kopf <chr>,
## #   Bundesland <chr>
```

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

```
## # A tibble: 13,108 × 8
##   Regionalschluessel Gemeinde Verwaltungsform Einwohner Schulden_gesamt
##   <chr>           <chr>    <chr>      <chr>        <chr>
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992 461784578.970
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712 1101148771.960
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799 983573069.740
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889 503685790.819
## 5 01051
Kreisverwaltung... Kreisverwaltung {135 009} 57672347.289
## 6 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518 57881948.270
## 7 010510044044 Heide, Stadt      amtsfreie Geme... 21919 49208227.039
## 8 010515163
Amtsverwaltung ... Amtsverwaltung {15 770} 825202.43
## 9 010515163003 Averlak          amtsangehörige... 572 2247182.66000
## 10 010515163010 Brickeln         amtsangehörige... 202 1435532.82995
## # i 13,098 more rows
## # i 3 more variables: Veraenderung_Vorjahr <chr>, Schulden_pro_kopf <chr>,
## #     Bundesland <chr>
```

```

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

```

## # A tibble: 13,108 × 12

	Regionalschluessel	Gemeinde	Verwaltungsform	Einwohner	Schulden_gesa
	<chr>	<chr>	<chr>	<chr>	<dbl>
## 1	010010000000	Flensburg, Stadt	kreisfreie Sta...	91992	4617845
## 2	010020000000	Kiel, Landeshau...	kreisfreie Sta...	246712	11011487
## 3	010030000000	Lübeck, Hansest...	kreisfreie Sta...	217799	9835730
## 4	010040000000	Neumünster, Sta...	kreisfreie Sta...	79889	5036857
## 5	01051	Kreisverwaltung...	Kreisverwaltung {135 009}	576723	4
## 6	010510011011	Brunsbüttel, St...	amtsfreie Geme...	12518	578819
## 7	010510044044	Heide, Stadt	amtsfreie Geme...	21919	492082
## 8	010515163	Amtsverwaltung ...	Amtsverwaltung {15 770}	82520	8
## 9	010515163003	Averlak	amtsangehörige...	572	22471
## 10	010515163010	Brickeln	amtsangehörige...	202	14355
## # i 13,098 more rows					
## # i 7 more variables: Veraenderung_Vorjahr <chr>, 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(Regionalschluess)
  mutate(
    Einwohner_num = as.numeric(gsub("[[:space:]]{}]", "", Regionalschluess)),
    Schulden_gesamt = as.numeric(gsub("[[:space:]]{}]", "", Schulden_gesamt)),
    Schulden_pro_kopf = as.numeric(gsub("[[:space:]]{}]", "", Schulden_pro_kopf)),
    Schulden_pro_kopf_new = round(Schulden_gesamt / Einwohner_num),
    landkreis = str_extract(Regionalschluessel, "^.{10}"),
    differenz = Schulden_pro_kopf - Schulden_pro_kopf_new
  ) %>%
  relocate(Regionalschluessel, Einwohner, Einwohner_num, Schulden_pro_kopf,
  Schulden_gesamt, differenz, landkreis, .after = Schulden_pro_kopf_new)

```

## # A tibble: 13,108 × 12

	Regionalschluessel	Einwohner	Einwohner_num	Schulden_pro_kopf
## 1	010010000000	91992	91992	5020.
## 2	010020000000	246712	246712	4463.
## 3	010030000000	217799	217799	4516.
## 4	010040000000	79889	79889	6305.
## 5	01051	{135 009}	135009	427.
## 6	010510011011	12518	12518	4624.
## 7	010510044044	21919	21919	2245
## 8	010515163	{15 770}	15770	52.3
## 9	010515163003	572	572	3929.
## 10	010515163010	202	202	7107.
## # i 13,098 more rows				
## # i 8 more variables: Schulden_pro_kopf_new <dbl>, Gemeinde <chr>, Verwaltungsform <chr>, Schulden_gesamt <dbl>, Veraenderung_Vorjahr <chr>, Bundesland <chr>, landkreis <chr>, differenz <dbl>				

```
# Erstellen der Vergleichstabelle
schulden_individuell %>%
  filter(!is.na(Einwohner) & !is.na(Regionalschluess
    mutate(
      Einwohner_num = as.numeric(gsub("[:space:]{}]",",
      Schulden_gesamt = as.numeric(gsub("[:space:]{}]")
      Schulden_pro_kopf = as.numeric(gsub("[:space:]{
      Schulden_pro_kopf_new = round(Schulden_gesamt /
      landkreis = str_extract(Regionalschluessel, "^.{
      differenz = Schulden_pro_kopf - Schulden_pro_kop
    ) %>%
    relocate(Regionalschluessel, Einwohner, Einwohner_
  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 keine nicht verfügbaren Werte, was gut ist bzgl. der internen Validität.

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

```
## # A tibble: 0 × 12
## # i 12 variables: Regionalschlüssel <chr>, Einwohner <chr>,
## #   Einwohner_num <dbl>, Schulden_pro_kopf <dbl>, Schulden_pro_kopf_new <dbl>,
## #   Gemeinde <chr>, Verwaltungsform <chr>, Schulden_gesamt <dbl>,
## #   Veraenderung_Vorjahr <chr>, Bundesland <chr>, landkreis <chr>,
## #   differenz <dbl>
```

# 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("../case-study/data/BIP_2023.xlsx", sheet="1.1", skip = 4)  
erwerb <- read_xlsx("../case-study/data/BIP_2023.xlsx", sheet="3.1", skip = 4)  
einwohner <- read_xlsx("../case-study/data/BIP_2023.xlsx", sheet = "5", skip = 4)
```

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

```
## # A tibble: 445 × 38
##   `Lfd. Nr.` `EU-Code` `Regional-schlüssel` Land `NUTS 1` `NUTS 2` `NUTS 3` ...
##   <dbl>     <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
## # i 435 more rows
## # i 31 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>, `2018` <dbl>, `2019` <dbl>, ...
```

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

```
## # A tibble: 444 × 38
##   `Lfd. Nr.` `EU-Code` `Regional-schlüssel` Land `NUTS 1` `NUTS 2` `NUTS 3`
##   <dbl>     <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
## # i 434 more rows
## # i 31 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>, `2018` <dbl>, `2019` <dbl>, ...
```

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

## # A tibble: 444 × 31
##   `Regional-schlüssel` `1992`  `1994`  `1995`  `1996`  `1997`  `1998`  `1999`  `2000` 
##   <chr>          <chr>    <chr>    <chr>    <chr>    <chr>    <chr>    <chr>    <chr> 
## 1 08             255866... 26264... 27174... 27677... 28219... 29109... 30072... 3.09...
## 2 081            110977... 11160... 11528... 11678... 12086... 12384... 12779... 1.30...
## 3 08111          32946.... 31736... 32281... 32802... 34339... 33553... 35048... 3.53...
## 4 08115          12090.... 11833... 11937... 12097... 13919... 13679... 14424... 1.39...
## 5 08116          12275.... 12482... 12748... 13169... 13284... 13952... 14192... 1.44...
## 6 08117          5062.0... 5180.... 5447.... 5643.... 5667.... 5838.... 5920.... 6.00...
## 7 08118          11714.... 12163... 12756... 12895... 13143... 13516... 13866... 1.47...
## 8 08119          8500.4... 8723.... 9320.... 8780.... 8928.... 9175.... 9707.... 1.04...
## 9 08121          4219.2... 4387.... 4522.... 4510.... 4581.... 5645.... 5282.... 5.27...
## 10 08125         6073.5... 6126.... 6577.... 6811.... 7019.... 7645.... 7928.... 8.45...
## # i 434 more rows
## # i 22 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>, `2018` <dbl>, `2019` <dbl>,
## #   `2020` <dbl>, `2021` <dbl>, `2022` <dbl>
```

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

```
## # A tibble: 444 × 31
##   Regionalschluessel `1992`  `1994`  `1995`  `1996`  `1997`  `1998`  `1999`  `2000` ...
##   <chr>           <chr>    <chr>    <chr>    <chr>    <chr>    <chr>    <chr>    <dbl>
## 1 08              255866.4... 26264... 27174... 27677... 28219... 29109... 30072... 3.09
## 2 081             110977.0... 11160... 11528... 11678... 12086... 12384... 12779... 1.30
## 3 08111            32946.88... 31736... 32281... 32802... 34339... 33553... 35048... 3.53
## 4 08115            12090.93   11833... 11937... 12097... 13919... 13679... 14424... 1.39
## 5 08116            12275.605  12482... 12748... 13169... 13284... 13952... 14192... 1.44
## 6 08117            5062.037... 5180.... 5447.... 5643.... 5667.... 5838.... 5920.... 6.00
## 7 08118            11714.16   12163... 12756... 12895... 13143... 13516... 13866... 1.47
## 8 08119            8500.405... 8723.... 9320.... 8780.... 8928.... 9175.... 9707.... 1.04
## 9 08121            4219.259  4387.... 4522.... 4510.... 4581.... 5645.... 5282.... 5.27
## 10 08125            6073.524... 6126.... 6577.... 6811.... 7019.... 7645.... 7928.... 8.45
## # i 434 more rows
## # i 22 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>, `2018` <dbl>, `2019` <dbl>,
## #   `2020` <dbl>, `2021` <dbl>, `2022` <dbl>
```

```
# Zeile löschen in der die `Lfd. Nr.` nicht nummeriert ist
# 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 nummerisch 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(Regionalschluessel = `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(Regionalschluessel = `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 × 31
##   Regionalschlüssel `1992`    `1994` `1995` `1996` `1997` `1998` `1999` `2000` 
##   <chr>           <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 08             255866.41... 26264... 27174... 27677... 28219... 29109... 30072... 3.09e5
## 2 081            110977.071 11160... 11528... 11678... 12086... 12384... 12779... 1.30e5
## 3 08111          32946.883... 31736... 32281... 32802... 34339... 33553... 35048... 3.53e4
## # i 22 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>, `2018` <dbl>, `2019` <dbl>,
## #   `2020` <dbl>, `2021` <dbl>, `2022` <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 × 31
##   Regionalschlüssel `1992`    `1994` `1995` `1996` `1997` `1998` `1999` `2000` 
##   <chr>           <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 08             255866.41... 26264... 27174... 27677... 28219... 29109... 30072... 3.09e5
## 2 081            110977.071 11160... 11528... 11678... 12086... 12384... 12779... 1.30e5
## 3 08111          32946.883... 31736... 32281... 32802... 34339... 33553... 35048... 3.53e4
## # i 22 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>, `2018` <dbl>, `2019` <dbl>,
## #   `2020` <dbl>, `2021` <dbl>, `2022` <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":"2022") , names_to = "Jahr", values_to = "BIP")
```

```
Fehler: Can't combine `1992` <character> and `2000` <double>.
```

# Daten in das long-Format überführen

BIP sollte normalerweise nummerisch 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: 444 × 31  
##   Regionalschluessel `1992`   `1994`   `1995`   `1996`   `1997`   `1998`   `1999`   `2000`  
##   <chr>          <chr>     <chr>     <chr>     <chr>     <chr>     <chr>     <chr>     <dbl>  
## 1 08             255866.4... 26264... 27174... 27677... 28219... 29109... 30072... 3.09  
## 2 081           110977.0... 11160... 11528... 11678... 12086... 12384... 12779... 1.30  
## 3 08111         32946.88... 31736... 32281... 32802... 34339... 33553... 35048... 3.53  
## 4 08115         12090.93   11833... 11937... 12097... 13919... 13679... 14424... 1.39  
## 5 08116         12275.605  12482... 12748... 13169... 13284... 13952... 14192... 1.44  
## 6 08117         5062.037... 5180.... 5447.... 5643.... 5667.... 5838.... 5920.... 6.00  
## 7 08118         11714.16   12163... 12756... 12895... 13143... 13516... 13866... 1.47  
## 8 08119         8500.405... 8723.... 9320.... 8780.... 8928.... 9175.... 9707.... 1.04  
## 9 08121         4219.259  4387.... 4522.... 4510.... 4581.... 5645.... 5282.... 5.27  
## 10 08125        6073.524... 6126.... 6577.... 6811.... 7019.... 7645.... 7928.... 8.45  
## # i 434 more rows  
## # i 22 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>, `2018` <dbl>, `2019` <dbl>,  
## #   `2020` <dbl>, `2021` <dbl>, `2022` <dbl>
```

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

```
## # A tibble: 444 × 7  
##   `1992`     `1994`     `1995`     `1996`     `1997`     `1998`     `1999`  
##   <chr>      <chr>      <chr>      <chr>      <chr>      <chr>      <chr>  
## 1 255866.41899999999 262645.41600000003 271746.699... 27677... 28219... 29109... 3007  
## 2 110977.071          111602.66499999999 115280.807    11678... 12086... 12384... 1277  
## 3 32946.883999999998 31736.567999999999 32281.0040... 32802... 34339... 33553... 3504  
## 4 12090.93           11833.816000000001 11937.788    12097... 13919... 13679... 1442  
## 5 12275.605          12482.948           12748.703    13169... 13284... 13952... 1419  
## 6 5062.0370000000003 5180.073999999996 5447.49399... 5643... 5667.... 5838.... 5920  
## 7 11714.16            12163.822           12756.3989... 12895... 13143... 13516... 1386  
## 8 8500.4050000000007 8723.0990000000002 9320.15600... 8780.... 8928.... 9175.... 9707  
## 9 4219.259           4387.480999999998 4522.82399... 4510.... 4581.... 5645.... 5282  
## 10 6073.524999999996 6126.3310000000001 6577.05599... 6811.... 7019.... 7645.... 7928  
## # i 434 more rows
```

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

```
## # A tibble: 444 × 7  
##   `1992`   `1994`   `1995`   `1996`   `1997`   `1998`   `1999`  
##   <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>  
## 1 255866. 262645. 271747. 276777. 282190. 291100. 300727.  
## 2 110977. 111603. 115281. 116787. 120867. 123842. 127799.  
## 3 32947. 31737. 32281. 32803. 34340. 33553. 35048.  
## 4 12091. 11834. 11938. 12097. 13919. 13679. 14424.  
## 5 12276. 12483. 12749. 13169. 13285. 13952. 14192.  
## 6 5062. 5180. 5447. 5643. 5668. 5839. 5920.  
## 7 11714. 12164. 12756. 12895. 13144. 13516. 13867.  
## 8 8500. 8723. 9320. 8781. 8928. 9176. 9708.  
## 9 4219. 4387. 4523. 4511. 4581. 5646. 5282.  
## 10 6074. 6126. 6577. 6812. 7020. 7646. 7929.  
## # i 434 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: There were 7 warnings in `mutate()` .
## The first warning was:
##  i In argument: `across(is.character, as.double)` .
## Caused by warning:
## ! NAs durch Umwandlung erzeugt
##  i Run `dplyr::last_dplyr_warnings()` to see the 6 remaining warnings.
```

```
# Einwohner von 1992 – 1999 umformen (als numerische Variable)
einwohner_double <- einwohner_wide %>%
  select(`1992`:`1999`) %>%
  mutate(across(is.character, as.double))
```

```
## Warning: There were 7 warnings in `mutate()` .
## The first warning was:
##  i In argument: `across(is.character, as.double)` .
## Caused by warning:
```

# Daten in das long-Format überführen

Wir überprüfen, welche Spalten die Warnung hervorgerufen haben und wo NAs erzeugt wurden

```
bip_wide_test <- bip_wide %>%
  bind_cols(bip_double)

head(filter(bip_wide_test, is.na(`1992...32`)))
```

```
## # A tibble: 6 × 38
##   Regionalschlüssel `1992...2` `1994...3` `1995...4` `1996...5` `1997...6`
##   <chr>          <chr>      <chr>      <chr>      <chr>      <chr>
## 1 13003          .          .          .          .          .
## 2 13004          .          .          .          .          .
## 3 13071          .          .          .          .          .
## 4 13072          .          .          .          .          .
## 5 13073          .          .          .          .          .
## 6 13074          .          .          .          .          .
## # i 32 more variables: `1998...7` <chr>, `1999...8` <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>, `2018` <dbl>, `2019` <dbl>, `2020` <dbl>,
```

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 Erwerbstätige sind in 1000 Personen angegeben, daher Erwerbstätige 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: 13,320 × 3
##   Regionalschlüssel Jahr      bip
##   <chr>              <chr>     <dbl>
## 1 08                2000    308823.
## 2 08                2001    323078.
## 3 08                2002    325510.
## 4 08                2003    329164.
## 5 08                2004    333276.
## 6 08                2005    335789.
## 7 08                2006    357283.
## 8 08                2007    377021.
## 9 08                2008    381903.
## 10 08               2009    353463.
## # i 13,310 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: 13,320 × 3  
##   Regionalschlüssel Jahr      bip  
##   <chr>          <dbl>      <dbl>  
## 1 08            2000 308822815000  
## 2 08            2001 323077717000  
## 3 08            2002 325510403000  
## 4 08            2003 329164078000  
## 5 08            2004 333275845000  
## 6 08            2005 335788716000  
## 7 08            2006 357283378000  
## 8 08            2007 377021382000  
## 9 08            2008 381902739000  
## 10 08           2009 353462984000  
## # i 13,310 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: 13,320 × 3
##   Regionalschlüssel Jahr      bip
##   <chr>           <dbl>    <dbl>
## 1 08              1992 255866419000
## 2 081             1992 110977071000
## 3 08111           1992 32946884000
## 4 08115           1992 12090930000
## 5 08116           1992 12275605000
## 6 08117           1992 5062037000
## 7 08118           1992 11714160000
## 8 08119           1992 8500405000
## 9 08121           1992 4219259000
## 10 08125          1992 6073525000
## # i 13,310 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 = "einwohner")
  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 (ARS) 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 2022 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: 10,782 × 8
##   Regionalschlüssel Gemeinde Verwaltungsform Einwohner Schuldens_gesamt
##   <chr>           <chr>    <chr>           <dbl>          <dbl>
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992        46178457
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712       110114871
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799       98357307
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889        50368579
## 5 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518        57881942
## 6 010510044044 Heide, Stadt amtsfreie Geme... 21919        49208222
## 7 010515163003 Averlak      amtsangehörige... 572         224718
## 8 010515163010 Brickeln     amtsangehörige... 202         143550
## 9 010515163012 Buchholz     amtsangehörige... 998         367565
## 10 010515163016 Burg (Dithmarsc... amtsangehörige... 4212       1437729
## # i 10,772 more rows
## # i 3 more variables: Schulden_pro_kopf <dbl>, Bundesland <chr>,
## #   landkreis <chr>
```

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

```
## # A tibble: 10,782 × 8
## # Groups:   landkreis [396]
##   Regionalschlüssel Gemeinde      Verwaltungsform Einwohner Schulden_gesamt
##   <chr>           <chr>           <chr>           <dbl>           <dbl>
## 1 010010000000 Flensburg, Stadt kreisfreie Sta... 91992  4617845
## 2 010020000000 Kiel, Landeshau... kreisfreie Sta... 246712 11011487
## 3 010030000000 Lübeck, Hansest... kreisfreie Sta... 217799 9835730
## 4 010040000000 Neumünster, Sta... kreisfreie Sta... 79889  50368579
## 5 010510011011 Brunsbüttel, St... amtsfreie Geme... 12518   5788194
## 6 010510044044 Heide, Stadt    amtsfreie Geme... 21919   4920822
## 7 010515163003 Averlak          amtsangehörige... 572    224718
## 8 010515163010 Brickeln         amtsangehörige... 202    143553
## 9 010515163012 Buchholz        amtsangehörige... 998    367565
## 10 010515163016 Burg (Dithmarsc... amtsangehörige... 4212   1437729
## # i 10,772 more rows
## # i 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: 396 × 4
##   landkreis Schulden_pro_kopf_lk Einwohner Schulden_gesamt
##   <chr>          <dbl>      <dbl>          <dbl>
## 1 01001           5020.    91992        461784579.
## 2 01002           4463.    246712       1101148772.
## 3 01003           4516.    217799       983573070.
## 4 01004           6305.    79889        503685791.
## 5 01051           3344.    135009       451514241.
## 6 01053           1751.    203365       356087687.
## 7 01054           3360.    169183       568447168.
## 8 01055           2835.    204097       578703178.
## 9 01056           3049.    320985       978605223.
## 10 01057          2495.    130724       326157495.
## # i 386 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))
ungroup()
```

```
## # A tibble: 396 × 4
##   landkreis Schulden_pro_kopf_lk Einwohner Schulden_gesamt
##   <chr>          <dbl>      <dbl>          <dbl>
## 1 01001           5020.     91992        461784579.
## 2 01002           4463.     246712       1101148772.
## 3 01003           4516.     217799       983573070.
## 4 01004           6305.     79889        503685791.
## 5 01051           3344.     135009       451514241.
## 6 01053           1751.     203365       356087687.
## 7 01054           3360.     169183       568447168.
## 8 01055           2835.     204097       578703178.
## 9 01056           3049.     320985       978605223.
## 10 01057          2495.     130724       326157495.
## # i 386 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))
ungroup() %>%
  rename(Regionalschlüssel = landkreis)
```

```
## # A tibble: 396 × 4
##   Regionalschlüssel Schulden_pro_kopf_lk Einwohner Schulden_gesamt
##   <chr>                  <dbl>        <dbl>          <dbl>
## 1 01001                  5020.       91992        461784579.
## 2 01002                  4463.       246712       1101148772.
## 3 01003                  4516.       217799       983573070.
## 4 01004                  6305.       79889        503685791.
## 5 01051                  3344.       135009       451514241.
## 6 01053                  1751.       203365       356087687.
## 7 01054                  3360.       169183       568447168.
## 8 01055                  2835.       204097       578703178.
## 9 01056                  3049.       320985       978605223.
## 10 01057                 2495.       130724       326157495.
## # i 386 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))
ungroup() %>%
  rename(Regionalschlüssel = landkreis) ->
schulden_kombi
```

```
# Anzahl an Erwerbst tigen f r das Jahr 2022
```

```
erwerb_long
```

```
## # A tibble: 13,320 × 3
##   Regionalschlüssel Jahr     erw
##   <chr>           <dbl>   <dbl>
## 1 08                 1992 5230587
## 2 081                1992 2046858
## 3 08111               1992 486895
## 4 08115               1992 188312
## 5 08116               1992 237498
## 6 08117               1992 118140
## 7 08118               1992 223059
## 8 08119               1992 176939
## 9 08121               1992 94510
## 10 08125              1992 115906
## # i 13,310 more rows
```

```
# Anzahl an Erwerbst tigen f r das Jahr 2022  
erwerb_long %>%  
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20)
```

```
## # A tibble: 398 x 3  
##   Regionalschluessel Jahr   erw  
##   <chr>           <dbl> <dbl>  
## 1 08111            2022 537749  
## 2 08115            2022 230680  
## 3 08116            2022 286414  
## 4 08117            2022 120195  
## 5 08118            2022 270068  
## 6 08119            2022 206802  
## 7 08121            2022 100428  
## 8 08125            2022 182581  
## 9 08126            2022  75160  
## 10 08127           2022 116646  
## # i 388 more rows
```

```
# Anzahl an Erwerbst tigen f r das Jahr 2022  
erwerb_long %>%  
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20  
  select(-Jahr)
```

```
## # A tibble: 398 x 2  
##   Regionalschluessel    erw  
##   <chr>                 <dbl>  
## 1 08111                537749  
## 2 08115                230680  
## 3 08116                286414  
## 4 08117                120195  
## 5 08118                270068  
## 6 08119                206802  
## 7 08121                100428  
## 8 08125                182581  
## 9 08126                75160  
## 10 08127               116646  
## # i 388 more rows
```

```
# Anzahl an Erwerbst tigen f r das Jahr 2022  
erwerb_long %>%  
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20  
  select(-Jahr) ->  
erwerb_kombi
```

```
# Anzahl an Einwohner für das Jahr 2022  
einwohner_long
```

```
## # A tibble: 13,320 × 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  
## # i 13,310 more rows
```

```
# Anzahl an Einwohner für das Jahr 2022  
einwohner_long %>%  
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20)
```

```
## # A tibble: 398 × 3  
##   Regionalschluessel Jahr einwohner  
##   <chr>           <dbl>    <dbl>  
## 1 08111            2022    629570  
## 2 08115            2022    395862  
## 3 08116            2022    536807  
## 4 08117            2022    260452  
## 5 08118            2022    547865  
## 6 08119            2022    429857  
## 7 08121            2022    126974  
## 8 08125            2022    350541  
## 9 08126            2022    114191  
## 10 08127           2022    201116  
## # i 388 more rows
```

```
# Anzahl an Einwohner für das Jahr 2022
einwohner_long %>%
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20
  select(-Jahr)
```

```
## # A tibble: 398 × 2
##   Regionalschluessel einwohner
##   <chr>                <dbl>
## 1 08111               629570
## 2 08115               395862
## 3 08116               536807
## 4 08117               260452
## 5 08118               547865
## 6 08119               429857
## 7 08121               126974
## 8 08125               350541
## 9 08126               114191
## 10 08127              201116
## # i 388 more rows
```

```
# Anzahl an Einwohner für das Jahr 2022
einwohner_long %>%
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20
  select(-Jahr) ->
einwohner_kombi
```

```
# Anzahl der Einwohner mit dem BIP verbinden um das  
left_join(bip_long, einwohner_long, by=c("Regionalsc  
  
## # A tibble: 13,320 × 4  
##   Regionalschluessel Jahr      bip einwohner  
##   <chr>          <dbl>    <dbl>    <dbl>  
## 1 08            1992 255866419000 10050431  
## 2 081           1992 110977071000 3771006  
## 3 08111         1992 32946884000 593628  
## 4 08115         1992 12090930000 343190  
## 5 08116         1992 12275605000 487370  
## 6 08117         1992 5062037000 248688  
## 7 08118         1992 11714160000 475248  
## 8 08119         1992 8500405000 389670  
## 9 08121         1992 4219259000 118566  
## 10 08125        1992 6073525000 283163  
## # i 13,310 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)
```

```
## # A tibble: 13,320 × 5  
##   Regionalschluessel Jahr      bip einwohner bip_pro_kopf  
##   <chr>          <dbl>    <dbl>     <dbl>       <dbl>  
## 1 08            1992 255866419000 10050431 25458.  
## 2 081           1992 110977071000 3771006 29429.  
## 3 08111         1992 32946884000 593628 55501.  
## 4 08115         1992 12090930000 343190 35231.  
## 5 08116         1992 12275605000 487370 25187.  
## 6 08117         1992 5062037000 248688 20355.  
## 7 08118         1992 11714160000 475248 24649.  
## 8 08119         1992 8500405000 389670 21814.  
## 9 08121         1992 4219259000 118566 35586.  
## 10 08125        1992 6073525000 283163 21449.  
## # i 13,310 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  
filter(nchar(Regionalschluessel) == 5 & Jahr == 20
```

```
## # A tibble: 398 × 5  
##   Regionalschluessel Jahr      bip einwohner bip_pro_kopf  
##   <chr>          <dbl>    <dbl>    <dbl>        <dbl>  
## 1 08111           2022 58703587000  629570       93244.  
## 2 08115           2022 29582502000  395862       74729.  
## 3 08116           2022 27010925000  536807       50318.  
## 4 08117           2022 9005138000  260452       34575.  
## 5 08118           2022 26245037000  547865       47904.  
## 6 08119           2022 16541107000  429857       38480.  
## 7 08121           2022 8132814000  126974       64051.  
## 8 08125           2022 23554259000  350541       67194.  
## 9 08126           2022 6489785000  114191       56833.  
## 10 08127          2022 10200716000  201116       50721.  
## # i 388 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  
    filter(nchar(Regionalschluessel) == 5 & Jahr == 20  
    select(-c(Jahr, einwohner))
```

```
## # A tibble: 398 × 3  
##   Regionalschluessel      bip bip_pro_kopf  
##   <chr>          <dbl>        <dbl>  
## 1 08111     58703587000  93244.  
## 2 08115     29582502000  74729.  
## 3 08116     27010925000  50318.  
## 4 08117     9005138000  34575.  
## 5 08118     26245037000  47904.  
## 6 08119     16541107000  38480.  
## 7 08121     8132814000  64051.  
## 8 08125     23554259000  67194.  
## 9 08126     6489785000  56833.  
## 10 08127    10200716000  50721.  
## # i 388 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  
  filter(nchar(Regionalschluessel) == 5 & Jahr == 20  
  select(-c(Jahr, einwohner)) ->  
bip_kombi
```

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

```
## # A tibble: 400 × 2  
##   Regionalschlüssel total_alo  
##   <chr>              <dbl>  
## 1 01001               3970.  
## 2 01002              10315.  
## 3 01003              8776.  
## 4 01004              3359.  
## 5 01051              3858.  
## 6 01053              5351.  
## 7 01054              4155.  
## 8 01055              4824.  
## 9 01056              8547.  
## 10 01057             2572.  
## # i 390 more rows
```

```
# Datensätze zusammenführen  
  
# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis  
# Namen der Bundesländer zumergen  
alo_landkreis %>%  
  mutate(bundesland = str_extract(Regionalschluessel  
  
## # A tibble: 400 × 3  
##   Regionalschluessel total_alo bundesland  
##   <chr>              <dbl> <chr>  
## 1 01001             3970. 01  
## 2 01002            10315. 01  
## 3 01003            8776. 01  
## 4 01004            3359. 01  
## 5 01051            3858. 01  
## 6 01053            5351. 01  
## 7 01054            4155. 01  
## 8 01055            4824. 01  
## 9 01056            8547. 01  
## 10 01057           2572. 01  
## # i 390 more rows
```

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

```
## # A tibble: 400 × 6  
##   Regionalschluessel total_alo bundesland Schulden_pro_kopf_lk Einwohner  
##   <chr>          <dbl> <chr>           <dbl>      <dbl>  
## 1 01001            3970. 01             5020.     91992  
## 2 01002            10315. 01            4463.    246712  
## 3 01003            8776. 01            4516.    217799  
## 4 01004            3359. 01            6305.    79889  
## 5 01051            3858. 01            3344.    135009  
## 6 01053            5351. 01            1751.    203365  
## 7 01054            4155. 01            3360.    169183  
## 8 01055            4824. 01            2835.    204097  
## 9 01056            8547. 01            3049.    320985  
## 10 01057           2572. 01            2495.   130724  
## # i 390 more rows  
## # i 1 more variable: Schulden_gesamt <dbl>
```

```
# Datensätze zusammenführen  
  
# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis  
# Namen der Bundesländer zumergen  
alo_landkreis %>%  
  mutate(bundesland = str_extract(Regionalschluessel  
    left_join(., schulden_kombi, by = "Regionalschlues  
    left_join(., bip_kombi, by = "Regionalschluessel")  
  
## # A tibble: 400 × 8  
##   Regionalschluessel total_alo bundesland Schulden_pro_kopf_lk Einwohner  
##   <chr>          <dbl> <chr>           <dbl>      <dbl>  
## 1 01001            3970. 01             5020.     91992  
## 2 01002            10315. 01            4463.    246712  
## 3 01003            8776. 01            4516.    217799  
## 4 01004            3359. 01            6305.    79889  
## 5 01051            3858. 01            3344.    135009  
## 6 01053            5351. 01            1751.    203365  
## 7 01054            4155. 01            3360.    169183  
## 8 01055            4824. 01            2835.    204097  
## 9 01056            8547. 01            3049.    320985  
## 10 01057           2572. 01            2495.   130724  
## # i 390 more rows  
## # i 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(Regionalschluessel
    left_join(., schulden_kombi, by = "Regionalschlues
    left_join(., bip_kombi, by = "Regionalschluessel"))
# Zahl der Erwerbstätigen zumergen
  left_join(., erwerb_kombi, by = "Regionalschluesse
```

```
## # A tibble: 400 × 9
##   Regionalschluessel total_alo bundesland Schulden_pro_kopf_lk Einwohner
##   <chr>           <dbl> <chr>             <dbl>      <dbl>
## 1 01001            3970. 01              5020.     91992
## 2 01002            10315. 01             4463.     246712
## 3 01003            8776. 01              4516.     217799
## 4 01004            3359. 01              6305.     79889
## 5 01051            3858. 01              3344.     135009
## 6 01053            5351. 01              1751.     203365
## 7 01054            4155. 01              3360.     169183
## 8 01055            4824. 01              2835.     204097
## 9 01056            8547. 01              3049.     320985
## 10 01057           2572. 01              2495.     130724
## # i 390 more rows
## # i 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(Regionalschluessel  
    left_join(., schulden_kombi, by = "Regionalschlues  
    left_join(., bip_kombi, by = "Regionalschluessel")  
# Zahl der Erwerbstätigen zumergen  
  left_join(., erwerb_kombi, by = "Regionalschluesse  
gesamtdaten
```

```
# Datensätze zusammenführen

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

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

```
# Datensätze zusammenführen

# Basisdatensatz -> Arbeitslosenzahlen pro Landkreis
# Namen der Bundesländer zumergen
alo_landkreis %>%
  mutate(bundesland = str_extract(Regionalschluessel
  left_join(., schulden_kombi, by = "Regionalschlues
  left_join(., bip_kombi, by = "Regionalschluessel")
# 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(Regionalschluessel
  left_join(., schulden_kombi, by = "Regionalschlues
  left_join(., bip_kombi, by = "Regionalschluessel")
# 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.
- ✚ Vergleichen Sie ihren Datensatz mit dem auf Github bereitgestellten Datensatz ("einkommen.rds"). Stimmen diese überein?
- ✚ Verbinden Sie die Informationen zu den durchschnittlichen Einkommen mit dem *gesamtdatensatz* aus dem vorherigen Abschnitt.