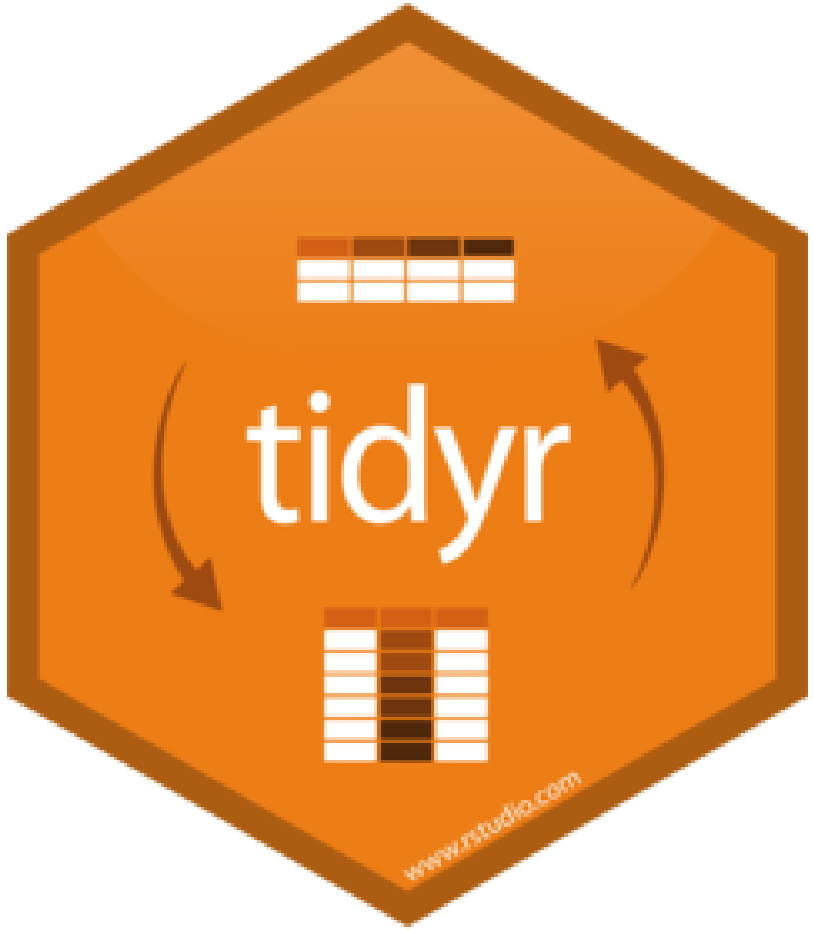
Die Datenaufbereitung

Daten aufarbeiten mit `tidyr`



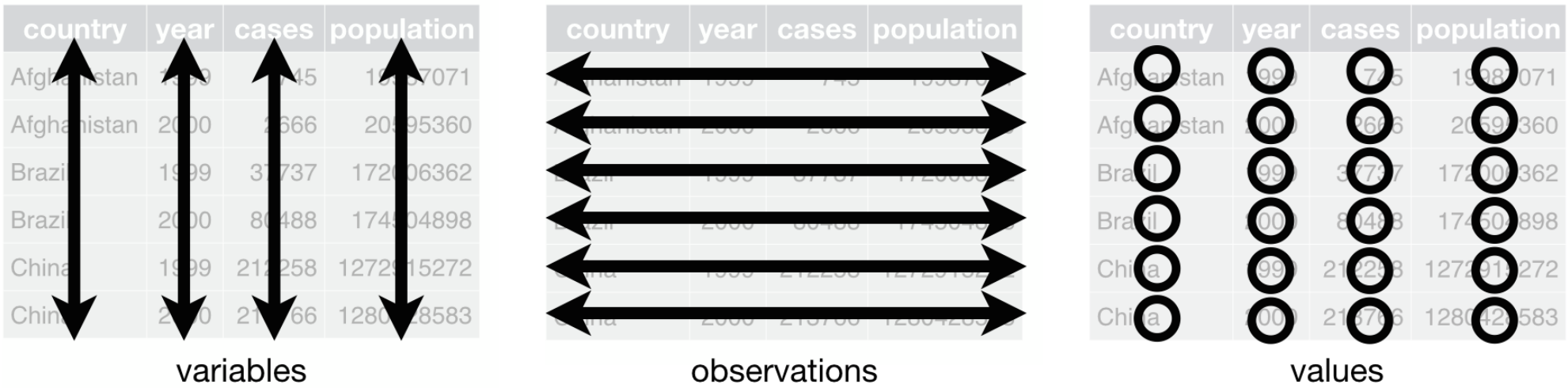
Daten aufarbeiten ("tidy")

```
"Tidy datasets are all alike but every messy dataset is messy in its own way."
- Hadley Wickham
```

Damit alle Bearbeitungsschritte innerhalb von R und `tidyverse` funktionieren müssen die Daten in einem bestimmten Format vorliegen:

→ Die Daten müssen *tidy* sein

Daten aufarbeiten ("tidy")



Quelle: Wickham, H., & Grolemund, G. (2016). R for data science: import, tidy, transform, visualize, and model data." O'Reilly Media, Inc.

- Jede Variable ist in einer eigenen Spalte repräsentiert
- Jede Beobachtung ist in einer extra Reihe (repräsentiert eine eigene Beobachtung)
- Wird auch als "long"-Format bezeichnet

Ist dieser Datensatz `tidy`?

```
head(geburtenrate)
```

```
# A tibble: 2 × 67
  country  `1950` `1951` `1952` `1953` `1954` `1955` `1956` `1957` `1958` `1959`
  <chr>     <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 Germany    2.07   2.08   2.11   2.14   2.17   2.21   2.25   2.29   2.33   2.37
2 South K…   4.02   4.33   4.89   5.35   5.73   6.01   6.2    6.31   6.33   6.27
# … with 56 more variables: 1960 <dbl>, 1961 <dbl>, 1962 <dbl>, 1963 <dbl>,
#   1964 <dbl>, 1965 <dbl>, 1966 <dbl>, 1967 <dbl>, 1968 <dbl>, 1969 <dbl>,
#   1970 <dbl>, 1971 <dbl>, 1972 <dbl>, 1973 <dbl>, 1974 <dbl>, 1975 <dbl>,
#   1976 <dbl>, 1977 <dbl>, 1978 <dbl>, 1979 <dbl>, 1980 <dbl>, 1981 <dbl>,
#   1982 <dbl>, 1983 <dbl>, 1984 <dbl>, 1985 <dbl>, 1986 <dbl>, 1987 <dbl>,
#   1988 <dbl>, 1989 <dbl>, 1990 <dbl>, 1991 <dbl>, 1992 <dbl>, 1993 <dbl>,
#   1994 <dbl>, 1995 <dbl>, 1996 <dbl>, 1997 <dbl>, 1998 <dbl>, 1999 <dbl>, …
```

Quelle: GapMinder

Ist dieser Datensatz `tidy`?

- Hier bekommen wir alle von uns gewünschten Werte über die Geburtenrate

Ein Blick auf die ersten 5 Spalten:

```
select(geburtenrate, country, "1950":"1954")
```

```
# A tibble: 2 × 6
  country     `1950` `1951` `1952` `1953` `1954`
  <chr>        <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 Germany       2.07   2.08   2.11   2.14   2.17
2 South Korea   4.02   4.33   4.89   5.35   5.73
```

Ist dieser Datensatz `tidy`?

- Hier bekommen wir alle von uns gewünschten Werte über die Geburtenrate

Ein Blick auf die ersten 5 Spalten:

```
select(geburtenrate, country, "1950":"1954")
```

```
# A tibble: 2 × 6
  country     `1950` `1951` `1952` `1953` `1954`
  <chr>        <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 Germany       2.07   2.08   2.11   2.14   2.17
2 South Korea   4.02   4.33   4.89   5.35   5.73
```

- *Allerdings:* Der Datensatz ist im `wide`-Format
  - Jede Reihe beinhaltet mehrere Beobachtungen
  - Die Variable ist in der Kopfzeile definiert

→ Daten sind **nicht** tidy

Reshaping

Datenformat anpassen

- ✚ Daten eingelesen ✓

To do:

- ✚ Daten in ein geeignetes Format zur Analyse überführen

→ Hier helfen uns die Funktionen des `tidyr` Pakets

- ✚ `pivot_longer` und `separate`

pivot_longer()

- Mit `pivot_longer` können Daten aus dem `wide`-Format in das `long`-Format überführt werden

Allgemeiner Befehl:

```
pivot_longer( names_to = Daten, values_to = Spaltenvariable, cols =
Beobachtungsvariable)
```

```
tidy_data <- geburtenrate %>%
  pivot_longer(names_to = "jahr", values_to = "geburtenrate",
               cols = c("1950":"2015"))
```

pivot_longer()

- Mit `pivot_longer` können Daten aus dem `wide`-Format in das `long`-Format überführt werden

Allgemeiner Befehl:

```
pivot_longer( names_to = Daten, values_to = Spaltenvariable, cols =
Beobachtungsvariable)
```

```
tidy_data <- geburtenrate %>%
  pivot_longer(names_to = "jahr", values_to = "geburtenrate",
               cols = c("1950":"2015"))
```

- cols: Welche Spalten sollen zusammengefasst werden?
- names_to: Wie soll die neue, zusammengefasste Spalte heißen?
- values_to: Wo sollen die Werte, welche aktuell in den Spalten stehen abgespeichert werden?

pivot_longer()

Alternativ können Sie auch die Spalte spezifizieren, welche **nicht** zusammengefasst werden sollen:

```
tidy_data <- geburtenrate %>%
  pivot_longer(names_to = "jahr",
               values_to = "geburtenrate", -country)
```

pivot_longer()

Alternativ können Sie auch die Spalte spezifizieren, welche **nicht** zusammengefasst werden sollen:

```
tidy_data <- geburtenrate %>%
  pivot_longer(names_to = "jahr",
               values_to = "geburtenrate", -country)
```

Der neu erzeugte Data Frame ist *tidy*:

```
head(tidy_data,4)
```

```
# A tibble: 4 × 3
  country jahr  geburtenrate
  <chr>   <chr>        <dbl>
1 Germany 1950          2.07
2 Germany 1951          2.08
3 Germany 1952          2.11
4 Germany 1953          2.14
```

`pivot_longer()`

Problem:

- Die Daten in der Spalte `jahr` sind nicht nummerisch
  - `pivot_longer` geht davon aus, dass Spaltennamen immer aus Buchstaben bestehen

```
class(tidy_data$jahr)
```

```
[1] "character"
```

pivot_longer()

Problem:

- Die Daten in der Spalte `jahr` sind nicht nummerisch
  - `pivot_longer` geht davon aus, dass Spaltennamen immer aus Buchstaben bestehen

```
class(tidy_data$jahr)
```

```
[1] "character"
```

- Können Sie durch ein zusätzliches `mutate` lösen:

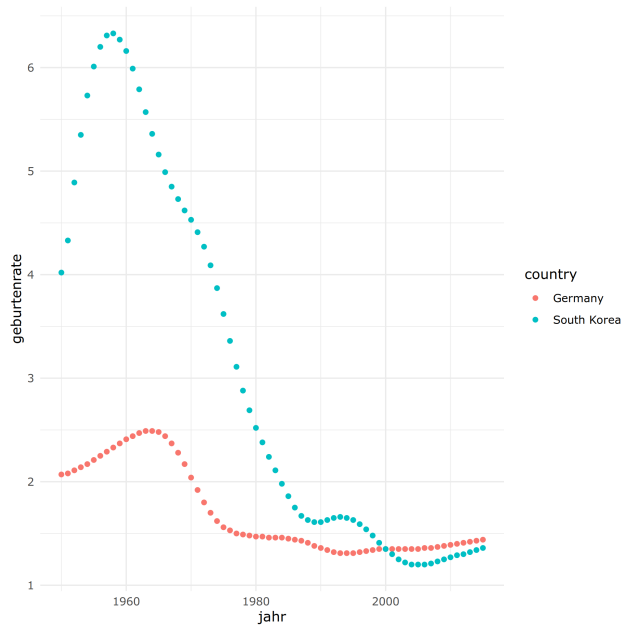
```
tidy_data <- geburtenrate %>%
  pivot_longer(names_to = "jahr",
               values_to = "geburtenrate", -country) %>%
  mutate( jahr = as.numeric(jahr))

class(tidy_data$jahr)
```

```
[1] "numeric"
```

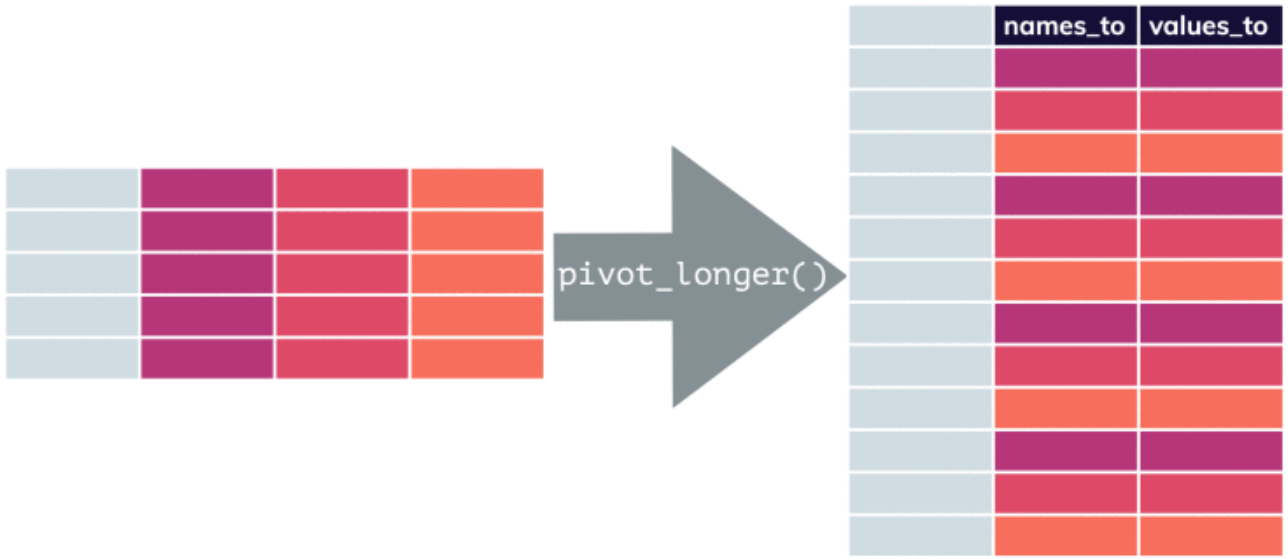
pivot_longer()

Wollen Sie nun die Datenreihen grafisch darstellen ist dies einfach möglich:

```
tidy_data %>%
  ggplot(aes(jahr, geburtenrate, color = country)) +
  geom_point()
```



pivot_longer()



Quelle: Erstellt von Apres Hill

Die `pivot_wider` Funktion

- `pivot_wider` ist das Pendant zu `pivot_longer`
- Manchmal ist es wichtig Datensätze in das `wide` Format zu konvertieren
  - Wird oft als Zwischenschritt gemacht

```
wide_data_neu <- tidy_data %>%
  pivot_wider(names_from = jahr, values_from = geburtenrate)

wide_data_neu %>%
  select(country, "1950":"1954")
```

```
# A tibble: 2 × 6
  country     `1950` `1951` `1952` `1953` `1954`
  <chr>        <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 Germany       2.07   2.08   2.11   2.14   2.17
2 South Korea   4.02   4.33   4.89   5.35   5.73
```

Die `pivot_wider` Funktion

- `pivot_wider` ist das Pendant zu `pivot_longer`
- Manchmal ist es wichtig Datensätze in das `wide` Format zu konvertieren
  - Wird oft als Zwischenschritt gemacht

```
wide_data_neu <- tidy_data %>%
  pivot_wider(names_from = jahr, values_from = geburtenrate)

wide_data_neu %>%
  select(country, "1950":"1954")
```

```
# A tibble: 2 × 6
  country     `1950` `1951` `1952` `1953` `1954`
  <chr>        <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 Germany       2.07   2.08   2.11   2.14   2.17
2 South Korea   4.02   4.33   4.89   5.35   5.73
```

- names_from: Welche Variable soll als Spaltenname fungieren soll?
- values_from: Welche Variable soll die Beobachtungen liefern?

pivot_longer()

Wenn Sie mehre Variablen im Datensatz haben, funktioniert `pivot_longer` nicht mehr:

```
#Erste fünf Spalten zeigen
select(leben_und_geburt, 1:5)
```

```
# A tibble: 8 × 5
  country      `1950_life_expe… `1951_life_expe… `1952_life_expe… `1953_life_expe…
  <chr>        <chr>            <chr>            <chr>            <chr>
1 Brazil       50.33            50.59            51.1             51.62
2 Canada       68.26            68.53            68.72            69.1
3 China        41.04            41.98            42.91            43.85
4 Germany      66.91            67.08            67.4             67.7
5 India        34.77            35.1             35.76            36.44
6 South Korea  43.02            40.52            40.02            45.02
7 Russia       57.27            57.76            58.16            58.96
8 South Africa 43.53            43.92            44.67            45.37
```

`pivot_longer()`

- Hier ist es nicht mehr möglich den Datensatz nur nach einer Variablen umzustellen
- Jedoch können Sie eine Platzhaltervariable für den Namen der Variablen einführen (`name`)
- Weiterhin können Sie sich eine Platzhaltervariable für den `Wert` der Variablen definieren

pivot_longer()

- Hier ist es nicht mehr möglich den Datensatz nur nach einer Variablen umzustellen
- Jedoch können Sie eine Platzhaltervariable für den Namen der Variablen einführen (`name`)
- Weiterhin können Sie sich eine Platzhaltervariable für den `Wert` der Variablen definieren

```
daten <- leben_und_geburt %>%
  # man beachte die "" um "name" und "Wert"
  pivot_longer(names_to = "name", values_to = "Wert", -country)

head(daten,4) # für die ersten 4 Zeilen
```

```
# A tibble: 4 × 3
  country name                 Wert
  <chr>   <chr>                <chr>
1 Brazil  1950_life_expectancy 50.33
2 Brazil  1951_life_expectancy 50.59
3 Brazil  1952_life_expectancy 51.1
4 Brazil  1953_life_expectancy 51.62
```

`separate()`

- Beide Variablen sind nun in `name` gespeichert
- Das jeweilige Jahr sollte in einer separaten Variable gespeichert sein
  - Kann durch den Unterstrich "_" getrennt werden
- `separate` schafft Abhilfe:
  - Spaltenname welche getrennt werden soll,
  - Spaltenname der neuen Spalte und
  - das Zeichen, nach dem getrennt werden soll

```
daten %>% separate(name, c("jahr", "variablen_name"), "_") %>%
  head(2)
```

```
# A tibble: 2 × 4
  country jahr  variablen_name Wert
  <chr>   <chr> <chr>          <chr>
1 Brazil  1950  life           50.33
2 Brazil  1951  life           50.59
```

`separate()`

- **Problem:** `life_expectancy` wird auch durch "_" getrennt
- *Lösung:* Wenn eine zusätzliche Trennung beim Variablennamen vorliegt können Sie diese als letztes getrennten Worte durch `merge` wieder zusammenführen

```
daten %>%
  separate(name, c("jahr", "variablen_name"), sep = "_", extra = "merge") %>%
  head()
```

```
# A tibble: 6 × 4
  country jahr  variablen_name  Wert
  <chr>   <chr> <chr>           <chr>
1 Brazil  1950  life_expectancy 50.33
2 Brazil  1951  life_expectancy 50.59
3 Brazil  1952  life_expectancy 51.1
4 Brazil  1953  life_expectancy 51.62
5 Brazil  1954  life_expectancy 52.14
6 Brazil  1955  life_expectancy 52.66
```

separate()

- Wir wollen jedoch zwei Spalten mit den Variablennamen anstatt die Variablen in Reihen
  - Nutzen der zuvor gelernten `pivot_wider` Funktion
- Weiterhin sollten die Variablen `life_expectancy` und `fertility` nummerisch sein und keine Zeichenketten

```
tidy_data_extended <- daten %>%
  separate(name, c("jahr", "variablen_name"),
           sep = "_", extra = "merge", convert=TRUE) %>%
  pivot_wider(names_from = variablen_name, values_from =  Wert) %>%
  mutate( life_expectancy = as.numeric(life_expectancy),
          fertility = as.numeric(fertility))

head(tidy_data_extended,3)
```

```
# A tibble: 3 × 4
  country  jahr life_expectancy fertility
  <chr>   <int>           <dbl>     <dbl>
1 Brazil   1950            50.3      6.18
2 Brazil   1951            50.6      6.17
3 Brazil   1952            51.1      6.15
```

```
saveRDS(tidy_data_extended, file= "data/gapminder_life.rds")
```