Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Data Wrangling (3)
Reshape and Combine Tables (con'd)

Haohan Chen

POLI3148 Data Science in PPA (The University of Hong Kong)

Last update: October 12, 2023

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Setup

Recap

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

- ▶ Reshape (long <-> wide) with `pivot_longer` and `pivot_wider`
- ▶ Stack tables by row or by column with `bind_rows` and `bind_cols` (or, alternatively, `cbind` and `rbind`)
- ▶ Merge two tables with `inner_join`, `full_join`, `left_join`, `right_join`, `semi_join`, and `anti_join`

Example: The V-Dem Data

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
library(tidyverse)
d <- read_csv("_DataPublic_/vdem/1984_2022/vdem_1984_2022_external.csv")
d |> print(n = 3)
```

```
## # A tibble: 6,789 x 211
##   country_name country_text_id country_id  year historical_date project
##   <chr>        <chr>                <dbl> <dbl> <date>            <dbl>
## 1 Mexico       MEX                      3  1984 1984-12-31            0
## 2 Mexico       MEX                      3  1985 1985-12-31            0
## 3 Mexico       MEX                      3  1986 1986-12-31            0
## # i 6,786 more rows
## # i 205 more variables: historical <dbl>, histname <chr>, codingstart <dbl>,
## #   codingend <dbl>, codingstart_contemp <dbl>, codingend_contemp <dbl>,
## #   codingstart_hist <dbl>, codingend_hist <dbl>, gapstart1 <dbl>,
## #   gapstart2 <dbl>, gapstart3 <dbl>, gapend1 <dbl>, gapend2 <dbl>,
## #   gapend3 <dbl>, gap_index <dbl>, COWcode <dbl>, e_v2x_api_3C <dbl>,
## #   e_v2x_api_4C <dbl>, e_v2x_api_5C <dbl>, e_v2x_civlib_3C <dbl>, ...
```

Example: The V-Dem Data

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Focus on the economic indicators: GDP and GDP per capita.

```
d_gdp <- d |>
  select(country_text_id, year, e_gdp, e_gdppc) |>
  rename("gdp" = "e_gdp", "gdppc" = "e_gdppc")

d_gdp |> print(n = 3)
```

```
## # A tibble: 6,789 x 4
##   country_text_id  year    gdp gdppc
##   <chr>           <dbl>  <dbl> <dbl>
## 1 MEX              1984 93563.  11.7
## 2 MEX              1985 94259.  11.5
## 3 MEX              1986 92750.  11.1
## # i 6,786 more rows
```

Reshape a Table

Wide to Long: `pivot_longer`

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
d_gdp_long <- d_gdp |>
  pivot_longer(cols = c("gdp", "gdppc"),
               names_to = "variable", values_to = "value")

d_gdp_long |> print(n = 4)
```

```
## # A tibble: 13,578 x 4
##    country_text_id  year variable    value
##    <chr>           <dbl> <chr>       <dbl>
## 1 MEX               1984 gdp        93563.
## 2 MEX               1984 gdppc        11.7
## 3 MEX               1985 gdp        94259.
## 4 MEX               1985 gdppc        11.5
## # i 13,574 more rows
```

Long to Wide: `pivot_wider`

Data Wrangling (3)

Haohan Chen

Setup

Reshape a Table

Stack Tables

Join Tables

Task: Reverse the above `pivot_long` operation.

```
d_gdp_wide_1 <- d_gdp_long |>
  pivot_wider(names_from = "variable", values_from = "value")


d_gdp_wide_1 |> print(n = 4)
```

```
## # A tibble: 6,789 x 4
##    country_text_id  year    gdp gdppc
##    <chr>           <dbl>  <dbl> <dbl>
## 1 MEX              1984  93563.  11.7
## 2 MEX              1985  94259.  11.5
## 3 MEX              1986  92750.  11.1
## 4 MEX              1987  93220.  10.9
## # i 6,785 more rows
```

Long to Wide: `pivot_wider`

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Task: Make `year` the column variable.

```
d_gdp_wide_2 <- d_gdp_long |>
  pivot_wider(names_from = "year", values_from = "value")

d_gdp_wide_2 |> print(n = 2)
```

```
## # A tibble: 362 x 41
##   country_text_id variable  `1984`  `1985`  `1986`  `1987`  `1988`  `1989` `1990`
##   <chr>           <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>
## 1 MEX             gdp       93563.  94259.  92750.  93220.  94687. 9.81e4 1.03e5
## 2 MEX             gdppc      11.7    11.5    11.1    10.9    10.8  1.10e1 1.14e1
## # i 360 more rows
## # i 32 more variables: `1991` <dbl>, `1992` <dbl>, `1993` <dbl>, `1994` <dbl>,
## #   `1995` <dbl>, `1996` <dbl>, `1997` <dbl>, `1998` <dbl>, `1999` <dbl>,
## #   `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>, ...
```

Long to Wide: `pivot_wider`

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Task: Make `country_text_id` the column variable.

```
d_gdp_wide_3 <- d_gdp_long |>
  pivot_wider(names_from = "country_text_id", values_from = "value")

d_gdp_wide_3 |> print(n = 2)
```

```
## # A tibble: 78 x 183
##    year variable      MEX    SUR    SWE    CHE    GHA    ZAF    JPN    MMR    RUS
##   <dbl> <chr>       <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1  1984 gdp       93563.  286.   2.35e4 2.31e4 3.02e3 3.15e4 2.87e5 4.18e3 3.49e5
## 2  1984 gdppc        11.7   7.43 2.66e1 3.32e1 2.20e0 9.03e0 2.26e1 1.10e0 1.65e1
## # i 76 more rows
## # i 172 more variables: ALB <dbl>, EGY <dbl>, YEM <dbl>, COL <dbl>, POL <dbl>,
## #   BRA <dbl>, USA <dbl>, PRT <dbl>, SLV <dbl>, YMD <dbl>, BGD <dbl>,
## #   BOL <dbl>, HTI <dbl>, HND <dbl>, MLI <dbl>, PAK <dbl>, PER <dbl>,
## #   SEN <dbl>, SSD <dbl>, SDN <dbl>, VNM <dbl>, AFG <dbl>, ARG <dbl>,
## #   ETH <dbl>, IND <dbl>, KEN <dbl>, PRK <dbl>, KOR <dbl>, XKX <dbl>,
## #   LBN <dbl>, NGA <dbl>, PHL <dbl>, TZA <dbl>, TWN <dbl>, THA <dbl>, ...
```

Why Do We Reshape Data?

Data Wrangling (3)

Haohan Chen

Setup

Reshape a Table

Stack Tables

Join Tables

- ▶ **For data cleaning:** Sometime it is much easier to clean the data after reshaping

- ▶ **For data visualization:** Some data visualization functions only take tables shaped in a specific way

- ▶ **For data sharing:** Sometimes you want to export the data for human readers (e.g., data coding/ labeling)

"But I am sure Excel can do the same thing!" It can do it for HUGE data reliably and fast. And the process is replicable.

Stack Tables

Task

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

▶ Let's say we want to merge your GDP data `d_gdp` with some additional datasets that *you know* you can just safely stack together.

▶ Example

  ▶ Merge with GDP data from 1906 to 1983

  ▶ Merge with education and Freedom House data from 1984 to 2022

Housekeeping: Load New Data

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

To demonstrate how to stack data vertically, I make a table with GDP data
from two previous time periods (1945 to 1983 and 1906-1944).

```r
d_gdp_1945 <-
  read_csv("_DataPublic_/vdem/1945_1983/vdem_1945_1983_external.csv") |>
  select(country_text_id, year, e_gdp, e_gdppc) |>
  rename("gdp" = "e_gdp", "gdppc" = "e_gdppc")

d_gdp_1906 <-
  read_csv("_DataPublic_/vdem/1906_1944/vdem_1906_1944_external.csv") |>
  select(country_text_id, year, e_gdp, e_gdppc) |>
  rename("gdp" = "e_gdp", "gdppc" = "e_gdppc")

d_gdp_1945 |> print(n = 2)
```

```
## # A tibble: 6,082 x 4
##   country_text_id  year   gdp gdppc
##   <chr>           <dbl> <dbl> <dbl>
## 1 MEX              1945 7827.  3.08
## 2 MEX              1946 8331.  3.17
## # i 6,080 more rows
```

Housekeeping: Load New Data

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

To demonstrate how to stack data horizontally, I make two subsets of `d` — one
with education indicators, another with Freedom House indicators.

```
d_edu <- d |>
  select(e_peaveduc, e_peedgini) |>
  rename("edu_15" = "e_peaveduc", "edu_gini" = "e_peedgini")

d_fh <- d |>
  select(starts_with("e_fh")) |>
  rename("fh_CivilLiberty" = "e_fh_cl", "fh_PoliticalRight" = "e_fh_pr",
         "fh_RuleOfLaw" = "e_fh_rol", "fh_Status" = "e_fh_status")

d_fh |> print(n = 2)
```

```
## # A tibble: 6,789 x 4
##    fh_CivilLiberty fh_PoliticalRight fh_RuleOfLaw fh_Status
##              <dbl>             <dbl>        <dbl>     <dbl>
## 1                4                 3           NA         2
## 2                4                 4           NA         2
## # i 6,787 more rows
```

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

bind_rows

```
d_gdp_1945_2022 <- bind_rows(d_gdp, d_gdp_1945)
d_gdp_1945_2022 |> print(n = 3)
```

```
## # A tibble: 12,871 x 4
##   country_text_id year     gdp gdppc
##   <chr>          <dbl>   <dbl> <dbl>
## 1 MEX             1984 93563.   11.7
## 2 MEX             1985 94259.   11.5
## 3 MEX             1986 92750.   11.1
## # i 12,868 more rows
```

```
unique(d_gdp_1945_2022$year) |> sort()
```

```
##  [1] 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959
## [16] 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974
## [31] 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989
## [46] 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004
## [61] 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019
## [76] 2020 2021 2022
```

bind_rows

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
d_gdp_1906_2022 <- bind_rows(d_gdp, d_gdp_1945, d_gdp_1906) # can take multiple data frames
d_gdp_1906_2022 |> print(n = 3)
```

```
## # A tibble: 18,559 x 4
##   country_text_id  year    gdp gdppc
##   <chr>           <dbl>  <dbl> <dbl>
## 1 MEX              1984 93563.  11.7
## 2 MEX              1985 94259.  11.5
## 3 MEX              1986 92750.  11.1
## # i 18,556 more rows
```

```
unique(d_gdp_1906_2022$year) |> sort()
```

```
##   [1] 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920
##  [16] 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935
##  [31] 1936 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950
##  [46] 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965
##  [61] 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980
##  [76] 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995
##  [91] 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
## [106] 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022
```

bind_cols

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```r
d_gdp_edu_fh <- bind_cols(d_gdp, d_edu, d_fh) # can take multiple data frames
d_gdp_edu_fh |> print(n = 3)
```

```
## # A tibble: 6,789 x 10
##   country_text_id  year    gdp gdppc edu_15 edu_gini fh_CivilLiberty
##   <chr>           <dbl>  <dbl> <dbl>  <dbl>    <dbl>           <dbl>
## 1 MEX              1984 93563.  11.7   6.08     32.7               4
## 2 MEX              1985 94259.  11.5   6.22     32.4               4
## 3 MEX              1986 92750.  11.1   6.36     31.9               4
## # i 6,786 more rows
## # i 3 more variables: fh_PoliticalRight <dbl>, fh_RuleOfLaw <dbl>,
## #   fh_Status <dbl>
```

```r
names(d_gdp_edu_fh)
```

```
##  [1] "country_text_id"   "year"              "gdp"
##  [4] "gdppc"             "edu_15"            "edu_gini"
##  [7] "fh_CivilLiberty"   "fh_PoliticalRight" "fh_RuleOfLaw"
## [10] "fh_Status"
```

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

!! WARNING !!

These are error-prone operations

- ▶ Do `bind_rows` and `bind_cols` ONLY WHEN you know for sure that there will not be a mismatch!
- ▶ If you have any slightest doubt, don't use them.

Join Tables

Tasks

Understand the behavior of different `join_` functions

▶ `left_join`: Merge and only keep observations that appear in the left-hand-side table.

▶ `right_join`: Merge and only keep observations that appear in the right-hand-side table.

▶ `inner_join`: Merge and only keep observations that appear in both tables.

▶ `full_join`: Merge and keep observations that appear either table.

▶ `anti_join`: Filter out observations that does appears in the right-hand-side table

▶ `semi_join`: Filter out observations that does not appear in the right-hand-side table

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Task 1: The Case

Join two datasets from the V-Dem data using the above different join_
functions

- ▶ *GDP* data from 2000-2022

- ▶ *GDP per capita* data from 1984 to 2010

Task 1: Setup

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```r
d_gdp_2000_2022 <- d |> filter(year %in% 2000:2022) |>
  select(country_text_id, year, e_gdp) |> rename("gdp" = "e_gdp")

d_gdppc_1984_2010 <- d |> filter(year %in% 1984:2010) |>
  select(country_text_id, year, e_gdppc) |> rename("gdppc" = "e_gdppc")

d_gdp_2000_2022 |> print(n = 2)
```

```
## # A tibble: 4,099 x 3
##   country_text_id  year      gdp
##   <chr>           <dbl>    <dbl>
## 1 MEX              2000 145206.
## 2 MEX              2001 146993.
## # i 4,097 more rows
```

```r
d_gdppc_1984_2010 |> print(n = 2)
```

```
## # A tibble: 4,641 x 3
##   country_text_id  year gdppc
##   <chr>           <dbl> <dbl>
## 1 MEX              1984  11.7
## 2 MEX              1985  11.5
## # i 4,639 more rows
```

left_join

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
d_lj <- d_gdp_2000_2022 |>
  left_join(d_gdppc_1984_2010, by = c("country_text_id", "year"))

d_lj |> print(n = 2)
```

```
## # A tibble: 4,099 x 4
##   country_text_id  year    gdp gdppc
##   <chr>           <dbl>  <dbl> <dbl>
## 1 MEX              2000 145206.  13.7
## 2 MEX              2001 146993.  13.6
## # i 4,097 more rows
```

```
unique(d_lj$year) |> sort()
```

```
##  [1] 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014
## [16] 2015 2016 2017 2018 2019 2020 2021 2022
```

right_join

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
d_rj <- d_gdp_2000_2022 |>
  right_join(d_gdppc_1984_2010, by = c("country_text_id", "year"))

d_rj |> print(n = 2)
```

```
## # A tibble: 4,641 x 4
##   country_text_id year    gdp gdppc
##   <chr>          <dbl>  <dbl> <dbl>
## 1 MEX             2000 145206.  13.7
## 2 MEX             2001 146993.  13.6
## # i 4,639 more rows
```

```
unique(d_rj$year) |> sort()
```

```
##  [1] 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998
## [16] 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
```

inner_join

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
d_ij <- d_gdp_2000_2022 |>
  inner_join(d_gdppc_1984_2010, by = c("country_text_id", "year"))

d_ij |> print(n = 2)
```

```
## # A tibble: 1,951 x 4
##   country_text_id  year     gdp gdppc
##   <chr>           <dbl>   <dbl> <dbl>
## 1 MEX              2000 145206.  13.7
## 2 MEX              2001 146993.  13.6
## # i 1,949 more rows
```

```
unique(d_ij$year) |> sort()
```

```
## [1] 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
```

full_join

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```r
d_fj <- d_gdp_2000_2022 |>
  full_join(d_gdppc_1984_2010, by = c("country_text_id", "year"))

d_fj |> print(n = 2)
```

```
## # A tibble: 6,789 x 4
##   country_text_id  year     gdp gdppc
##   <chr>           <dbl>   <dbl> <dbl>
## 1 MEX              2000 145206.  13.7
## 2 MEX              2001 146993.  13.6
## # i 6,787 more rows
```

```r
unique(d_fj$year) |> sort()
```

```
##  [1] 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998
## [16] 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013
## [31] 2014 2015 2016 2017 2018 2019 2020 2021 2022
```

semi_join

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```r
d_sj <- d_gdp_2000_2022 |>
  semi_join(d_gdppc_1984_2010, by = c("country_text_id", "year"))

d_sj |> print(n = 2)
```

```
## # A tibble: 1,951 x 3
##   country_text_id  year     gdp
##   <chr>           <dbl>   <dbl>
## 1 MEX              2000 145206.
## 2 MEX              2001 146993.
## # i 1,949 more rows
```

```r
unique(d_sj$year) |> sort()
```

```
##  [1] 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
```

anti_join

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```
d_aj <- d_gdp_2000_2022 |>
  anti_join(d_gdppc_1984_2010, by = c("country_text_id", "year"))

d_aj |> print(n = 2)
```

```
## # A tibble: 2,148 x 3
##   country_text_id  year     gdp
##   <chr>           <dbl>   <dbl>
## 1 MEX              2011 185824.
## 2 MEX              2012 192272.
## # i 2,146 more rows
```

```
unique(d_aj$year) |> sort()
```

```
##  [1] 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022
```

Join by Identifiers with Different Variable Names

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

If the identifiers have different names, you have two options

▶ Rename it beforehand

▶ Specify the `by =` part differently.

```r
# This is an artificial example whose variable name of a matching identifier
# is different from d_gdp_2020_2022
d_gdppc_1984_2010_t <- d_gdppc_1984_2010 |>
  rename("country_id" = "country_text_id")
```

```r
# Option 1: Rename the variables beforehand
d_aj_t <- d_gdp_2000_2022 |>
  rename("country_id" = "country_text_id") |>
  anti_join(d_gdppc_1984_2010_t, by = c("country_id", "year"))
```

```r
# Option 2: Specify the "by =" argument with a *named vector*
d_aj_t_2 <- d_gdp_2000_2022 |>
  anti_join(d_gdppc_1984_2010_t,
            by = c("country_text_id" = "country_id",
                   "year" = "year"))
```

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Many-to-One Join: Repeat!

Calculate each country's average 1984-2010 *GDP per capita* and merge it with our annual GDP data from 2000 to 2022.

```
d_gdppc_1984_2010_avg <- d_gdppc_1984_2010 |> group_by(country_text_id) |>
  summarise(gdppc_1984to2010 = mean(gdppc, na.rm = TRUE))
d_gdppc_1984_2010_avg |> print(n = 2)
```

```
## # A tibble: 180 x 2
##   country_text_id gdppc_1984to2010
##   <chr>                      <dbl>
## 1 AFG                         1.22
## 2 AGO                         3.35
## # i 178 more rows
```

```
d_lj_ManyToOne <- d_gdp_2000_2022 |>
  left_join(d_gdppc_1984_2010_avg, by = "country_text_id")
d_lj_ManyToOne |> print(n = 2)
```

```
## # A tibble: 4,099 x 4
##   country_text_id year     gdp gdppc_1984to2010
##   <chr>          <dbl>   <dbl>            <dbl>
## 1 MEX             2000 145206.             12.8
## 2 MEX             2001 146993.             12.8
## # i 4,097 more rows
```

Good Habit

Data Wrangling (3)

Haohan Chen

Setup

Reshape a Table

Stack Tables

Join Tables

Joining tables is also error-prone.

- ▶ You want to have a clear mind about which variables from which datasets are kept in your final merged data.

- ▶ Failing to do so can cause difficulty with replication.

Some advice based on personal experience

- ▶ Add suffixes or prefixes indicating data sources

- ▶ Add binary indicators (1/0) indicating from in which dataset is each observation available

Good Habit: Add Availability Indicators

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Add binary indicators about data availability in each sources.

```r
# The d_gdp_2000_2022 data are from V-Dem
d_gdp_2000_2022_t <- d_gdp_2000_2022 |> mutate(source_vdem = 1)

# *Pretend* that the d_gdppc_1984_2010 data are from the World Bank
d_gdppc_1984_2010_t <- d_gdppc_1984_2010 |> mutate(source_wb = 1)

d_fj_habit <- d_gdp_2000_2022_t |>
  full_join(d_gdppc_1984_2010_t, by = c("country_text_id", "year"))

d_fj_habit |> print(n = 3)
```

```
## # A tibble: 6,789 x 6
##   country_text_id  year      gdp source_vdem gdppc source_wb
##   <chr>           <dbl>    <dbl>       <dbl> <dbl>     <dbl>
## 1 MEX              2000 145206.           1  13.7         1
## 2 MEX              2001 146993.           1  13.6         1
## 3 MEX              2002 148549.           1  13.6         1
## # i 6,786 more rows
```

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

Good Habit: Add Availability Indicators

What can you do with these binary indicators? We can know the overlaps of multiple sources.

```
d_fj_habit |>
  group_by(source_vdem, source_wb) |>
  count()
```

```
## # A tibble: 3 x 3
## # Groups:   source_vdem, source_wb [3]
##   source_vdem source_wb     n
##         <dbl>     <dbl> <int>
## ## 1           1         1  1951
## ## 2           1        NA  2148
## ## 3          NA         1  2690
```

If the overlap looks weird to you, you will know that you need to re-examine the data merging process.

Good Habit: Add Availability Indicators

Data Wrangling (3)

Haohan Chen

Setup

Reshape a Table

Stack Tables

Join Tables

Question: Why not just check `NA` in each variables?

Answer: An observation can be missing for two reasons

- ▶ It is in the one of the tables but it does not contain a value.

- ▶ It is not in any of the tables at all.

`join_` make it hard to distinguish between the two scenarios.

Good Habit: Add prefix or suffix to variable names

Data Wrangling (3)

Haohan Chen

Setup

Reshape a Table

Stack Tables

Join Tables

- My previous advice: Give informative names to variable

- New advice: Add the source of the variables as part of their names if your final dataset is a combination of many different datasets

Good Habit: Add prefix or suffix to variable names

Data
Wrangling (3)

Haohan Chen

Setup

Reshape a
Table

Stack Tables

Join Tables

```r
d_gdp_2000_2022_rn <- d_gdp_2000_2022 |>
  rename("vdem_gdp" = "gdp")
  # rename_at(vars(-c("country_text_id", "year")), ~str_c("vdem_", .))

d_gdppc_1984_2010_rn <- d_gdppc_1984_2010 |>
  rename("wb_gdppc" = "gdppc")
  # rename_at(vars(-c("country_text_id", "year")), ~str_c("wb_", .))

d_fj_habit_2 <- d_gdp_2000_2022_rn |>
  full_join(d_gdppc_1984_2010_rn, by = c("country_text_id", "year"))

d_fj_habit_2 |> print(n = 3)
```

```
## # A tibble: 6,789 x 4
##   country_text_id  year vdem_gdp wb_gdppc
##   <chr>           <dbl>    <dbl>    <dbl>
## 1 MEX              2000  145206.     13.7
## 2 MEX              2001  146993.     13.6
## 3 MEX              2002  148549.     13.6
## # i 6,786 more rows
```