R Data Wrangling: tidyverse package dplyr

Presented as part of Princeton’s University’s 2022 Wintersession

Dawn Koffman
Office of Population Research (OPR)
Princeton University

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Slides and R script
Basic dplyr Principles

consistent with tidyr philosophy

input: data frame
output: data frame

first argument to dplyr commands is a data frame

input data frame is never modified in place ...
may want to save results in a new data frame

commands are optimized for
  - clairty (clean, clear syntax)
  - computation time (written in C++)

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
dplyr Commands: Verbs

**filter()**  subset observations (rows)

**arrange()**  order observations (rows)

**select()**  subset variables (columns)

**rename()**  change name of variables (column headers)

**mutate()**  add new variables (columns)

**group_by()**  partition observations into groups based on variable values

**summarise()**  collapse each group into a single row of values

# check structure of gapminder data
str(gapminder)

# tibble: improved data.frame for which dplyr provides nice methods for high-level inspection
# these methods do something sensible for datasets with many observations and/or variables

gdf <- as.data.frame(gapminder)
str(gdf)

gtdf <- as_tibble(gdf)
str(gtdf)

# high-level inspection of tibble
glimpse(gapminder)
Rows: 1,704
Columns: 6
$ country <fct> Afghanistan, Afghanistan, Afghanistan, Afghanistan, Afghanistan,...
$ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia...
$ pop <int> 8425333, 9240934, 10267083, 11537966, 13079460, 14880372, 1288181...
$ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 786.1134, 978.0...

View(gapminder) # note capital V

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
<table>
<thead>
<tr>
<th>country</th>
<th>continent</th>
<th>year</th>
<th>lifeExp</th>
<th>pop</th>
<th>gdpPercap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1952</td>
<td>28.801</td>
<td>8425333</td>
<td>779.4453</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1957</td>
<td>30.332</td>
<td>9240934</td>
<td>820.8530</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1962</td>
<td>31.997</td>
<td>10267083</td>
<td>853.1007</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1967</td>
<td>34.020</td>
<td>11537966</td>
<td>836.1971</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1972</td>
<td>36.088</td>
<td>13079460</td>
<td>739.9811</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1977</td>
<td>38.438</td>
<td>14880372</td>
<td>786.1134</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1982</td>
<td>39.854</td>
<td>12881816</td>
<td>978.0114</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1987</td>
<td>40.822</td>
<td>13867957</td>
<td>852.3959</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1992</td>
<td>41.674</td>
<td>16317921</td>
<td>649.3414</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1997</td>
<td>41.763</td>
<td>22227415</td>
<td>635.3414</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>2002</td>
<td>42.129</td>
<td>25268405</td>
<td>726.7341</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>2007</td>
<td>43.828</td>
<td>31889923</td>
<td>974.5803</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1952</td>
<td>48.451</td>
<td>3080907</td>
<td>406.8841</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1957</td>
<td>50.469</td>
<td>3646340</td>
<td>518.7643</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1962</td>
<td>52.358</td>
<td>4277736</td>
<td>527.2722</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1967</td>
<td>53.995</td>
<td>4995432</td>
<td>569.7951</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1972</td>
<td>55.635</td>
<td>5861135</td>
<td>799.3622</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1977</td>
<td>57.674</td>
<td>6642107</td>
<td>685.5877</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1982</td>
<td>60.363</td>
<td>7636524</td>
<td>788.8550</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1987</td>
<td>62.351</td>
<td>9216418</td>
<td>706.1573</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1992</td>
<td>60.377</td>
<td>10704340</td>
<td>693.4208</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>1997</td>
<td>46.809</td>
<td>11404948</td>
<td>792.4500</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>2002</td>
<td>39.989</td>
<td>11926563</td>
<td>672.0386</td>
</tr>
<tr>
<td>Zimbabwe</td>
<td>Africa</td>
<td>2007</td>
<td>43.487</td>
<td>12311143</td>
<td>469.7093</td>
</tr>
</tbody>
</table>
Subset Observations

filter(gapminder, country == "United States")

filter(gapminder, lifeExp < 30)

filter(gapminder, pop < 1000000)

filter(gapminder, pop < 1000000, year == 2007)

filter(gapminder, pop < 1000000 & year == 2007)

filter(gapminder, country == "United States" | country == "Canada", year > 2000)

filter(gapminder, country %in% c("United States", "Canada"), year > 2000)

distinct(gapminder, country)
View(distinct(gapminder, country))
distinct(gapminder, country) %>% View()

distinct(as.data.frame(gapminder), country)

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Subset Columns

```r
select(gapminder, country, continent)

country_continent <- select(gapminder, country, continent) %>% distinct()
country_continent

select(gapminder, -continent) # "-" means not ... gives TIDIER data set
tgap <- select(gapminder, -continent)

# But how to combine tgap and country_continent when want
# to summarize values by continent???
# Will later use a "join" function to combine

select(gapminder, year, country, continent, lifeExp) # select and re-order columns

select(gapminder, starts_with("co")) # select column headers that start with "co"

select(gapminder, country:lifeExp) # range
```

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Using filter() and select()

# list all countries showing only life expectancy for 2007

```
filter(gapminder, year == 2007) %>% select(country, year, lifeExp)
```

# list all countries showing only life expectancy for 2007
# with life expectancy variable named le (rather than lifeExp)

```
filter(gapminder, year == 2007) %>%
select(country, year, lifeExp) %>%
rename(le = lifeExp)
```
Order Rows

arrange(gapminder, year)

rename(gapminder, le = lifeExp) %>% filter(year == 2007) %>%
  select(country, year, le) %>%
  arrange(le)

rename(gapminder, le = lifeExp) %>% filter(year == 2007) %>%
  select(country, year, le) %>%
  arrange(desc(le))  # order by descending le

# to list all rows, can use gapminder as a data frame
rename(as.data.frame(gapminder), le = lifeExp) %>%
  filter(year == 2007) %>%
  select(country, year, le) %>%
  arrange(desc(le))

# list 5 countries with highest life expectancy in 2007
# show country, year, and le
rename(gapminder, le = lifeExp) %>%
  filter(year == 2007) %>%
  select(country, year, le) %>%
  arrange(desc(le)) %>%
  slice(1:5) # filter rows a second time, by position

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
# list 10 countries with lowest life expectancy values, 
# with lowest value at the top; 
# show country, year, and le

rename(gapminder, le=lifeExp) %>%
  filter(year == 2007) %>%
  select(country, year, le) %>%
  arrange(le) %>%  # low to high
  slice(1:10)  # filter rows a second time, by position
Construct New Columns

mutate(gapminder, popMil = round(pop / 1000000, 1), le = round(lifeExp, 0))

# A tibble: 1,704 x 8

<table>
<thead>
<tr>
<th>country</th>
<th>continent</th>
<th>year</th>
<th>lifeExp</th>
<th>pop</th>
<th>gdpPercap</th>
<th>popMil</th>
<th>le</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1952</td>
<td>28.801</td>
<td>8425333</td>
<td>779.</td>
<td>8.4</td>
<td>29</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1957</td>
<td>30.332</td>
<td>9240934</td>
<td>820.</td>
<td>9.2</td>
<td>30</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1962</td>
<td>31.997</td>
<td>10267083</td>
<td>853.</td>
<td>10.3</td>
<td>32</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Asia</td>
<td>1967</td>
<td>34.020</td>
<td>11537966</td>
<td>836.</td>
<td>11.5</td>
<td>34</td>
</tr>
</tbody>
</table>

transmute(gapminder, country = country, y = year, popMil = round(pop / 1000000, 1), le = round(lifeExp, 0)) %>%

arrange(y, country)

# A tibble: 1,704 x 4

<table>
<thead>
<tr>
<th>country</th>
<th>y</th>
<th>popMil</th>
<th>le</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>1952</td>
<td>8.4</td>
<td>29</td>
</tr>
<tr>
<td>Albania</td>
<td>1952</td>
<td>1.3</td>
<td>55</td>
</tr>
<tr>
<td>Algeria</td>
<td>1952</td>
<td>9.3</td>
<td>43</td>
</tr>
<tr>
<td>Angola</td>
<td>1952</td>
<td>4.2</td>
<td>30</td>
</tr>
<tr>
<td>Argentina</td>
<td>1952</td>
<td>17.9</td>
<td>62</td>
</tr>
</tbody>
</table>

...    ...    ...    ...

# A tibble: 1,704 x 4

<table>
<thead>
<tr>
<th>country</th>
<th>y</th>
<th>popMil</th>
<th>le</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>1952</td>
<td>8.4</td>
<td>29</td>
</tr>
<tr>
<td>Albania</td>
<td>1952</td>
<td>1.3</td>
<td>55</td>
</tr>
<tr>
<td>Algeria</td>
<td>1952</td>
<td>9.3</td>
<td>43</td>
</tr>
<tr>
<td>Angola</td>
<td>1952</td>
<td>4.2</td>
<td>30</td>
</tr>
<tr>
<td>Argentina</td>
<td>1952</td>
<td>17.9</td>
<td>62</td>
</tr>
</tbody>
</table>

...    ...    ...    ...

11
Vector Functions

# window functions take a vector of n values and return n values
# types of vector functions:
#   - ranking and ordering functions
#   - cumulative aggregates
#   - access to previous and next values

# assign lowest rank to lowest life expectancy
filter(gapminder, year == 2007) %>%
mutate(le_rank = dense_rank(lifeExp)) %>%
select(country, continent, year, lifeExp, le_rank) %>%
arrange(le_rank)

# assign lowest rank to highest life expectancy
filter(gapminder, year == 2007) %>%
mutate(le_rank = dense_rank(-lifeExp)) %>%
select(country, continent, year, lifeExp, le_rank) %>%
arrange(le_rank)

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Vector Functions: Cumulative Sum

filter(gapminder, year == 1952) %>%
  arrange(continent, country) %>%
  mutate(popMil = round(pop / 1000000, 1)) %>%
  mutate(cumpopMil = cumsum(popMil)) %>% View()

filter(gapminder, year == 2007) %>%
  arrange(continent, country) %>%
  mutate(popMil = round(pop / 1000000, 1)) %>%
  mutate(cumpopMil = cumsum(popMil)) %>% View()

[Table with data]

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Group Data:
Construct New Column Values By Group

filter(gapminder, year == 2007) %>%
  mutate(popMil = round(pop / 1000000, 1)) %>%
  arrange(continent, popMil) %>%
  group_by(continent) %>%
  mutate(cumpopMil = cumsum(popMil)) %>% View()

country continent year lifeExp pop  gdpPercap popMil cumpopMil
1 Sao Tome and Principe Africa 2007 65.528    199579  1598.4351    0.2       0.2
2              Djibouti  Africa 2007 54.791    496374  2082.4816    0.5       0.7
3     Equatorial Guinea  Africa 2007 51.579    551201 12154.0897    0.6       1.3
4               Comoros  Africa 2007 65.152    710960   986.1479    0.7       2.0
5               Reunion  Africa 2007 76.442    798094  7670.1226    0.8       2.8
6             Swaziland  Africa 2007 39.613   1133066  4513.4806    1.1       3.9
52              Nigeria  Africa 2007 46.869 135031164  2013.9773  135.0     929.6
53 Trinidad and Tobago Americas 2007 69.819   1056608 18008.5092    1.1       1.1

select(gapminder, country, year, pop) %>%
  group_by(country) %>%
  mutate(pop_lag = lag(pop), pop_chg = pop - pop_lag,
        pop_pctchg = round(pop_chg/pop_lag * 100, 1)) %>% View()

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
# list only rows that experienced a population decline during the previous 5 years
# show country, year, pop, pop_chg, pop_pctchg

```r
select(gapminder, country, year, pop) %>%
group_by(country) %>%
mutate(pop_lag = lag(pop), pop_chg = pop - pop_lag,
       pop_pctchg = round(pop_chg/pop_lag * 100, 1)) %>%
filter(pop_chg < 0) %>% View()
```
Summarise Data

# use summarise() with a summary function to change the unit of observation
# summary functions take a vector of values and return a single value
# very often used with group_by()

filter(gapminder, year == 2007) %>%
  summarise(year = mean(year), ncountries = n(),
            avg_country_le = mean(lifeExp), sd_country_le = sd(lifeExp))  # not used with group_by()

filter(gapminder, year == 2007) %>%
  group_by(continent) %>
  summarise(avg_country_le = mean(lifeExp))

filter(gapminder, year == 2007) %>%
  group_by(continent) %>
  summarise(year = mean(year), ncountries = n(),
            avg_country_le = mean(lifeExp), sd_country_le = sd(lifeExp))

filter(gapminder, year == 1952) %>%
  group_by(continent) %>
  summarise(year = mean(year), ncountries = n(),
            avg_country_le = mean(lifeExp), sd_country_le = sd(lifeExp))
Summarise Data

# Show data by continent and years 1952 AND 2007
# list number of countries, avg_country_le, and sd_country_le

```r
filter(gapminder, year == 1952 | year == 2007) %>%
group_by(continent, year) %>%
summarise(ncountries = n(), avg_country_le = mean(lifeExp), sd_country_le = sd(lifeExp))
```

`summarise()` has grouped output by 'continent' (override with `.groups` argument)

# By continent and year, show ncountries, avg_country_le, sd_country_le for ALL years of data

group_by(gapminder, continent, year) %>%
summarise(ncountries = n(), avg_country_le = mean(lifeExp), sd_country_le = sd(lifeExp)) %>%
View()

# Make a simple graph that shows avg_country_le over time, # for each continent

group_by(gapminder, continent, year) %>%
summarise(avg_country_le = mean(lifeExp)) %>%
ggplot(aes(x = year, y = avg_country_le, color = continent)) + geom_line()
summarise() “Peels Off” group_by()

# how many continents each country
# has belonged to over time

group_by(gapminder, country) %>%
  summarise(n_continents = n_distinct(continent))

# each summarise() "peels off" one level of group_by()

group_by(gapminder, country) %>%
  summarise(n_continents = n_distinct(continent)) %>%
  summarise(avg_n_continents = mean(n_continents))

<table>
<thead>
<tr>
<th>country</th>
<th>n_continents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>1</td>
</tr>
<tr>
<td>Albania</td>
<td>1</td>
</tr>
<tr>
<td>Algeria</td>
<td>1</td>
</tr>
<tr>
<td>Angola</td>
<td>1</td>
</tr>
<tr>
<td>Argentina</td>
<td>1</td>
</tr>
<tr>
<td>Australia</td>
<td>1</td>
</tr>
<tr>
<td>Austria</td>
<td>1</td>
</tr>
<tr>
<td>Bahrain</td>
<td>1</td>
</tr>
<tr>
<td>Bangladesh</td>
<td>1</td>
</tr>
<tr>
<td>Belgium</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

avg_n_continents       
<dbl>      
1.0

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
summarise(“Peels Off” group_by() 

```r
summarise(avg_le_cc = mean(lifeExp))
```  

### Continent vs. Country

<table>
<thead>
<tr>
<th>continent</th>
<th>country</th>
<th>avg_le_cc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>Algeria</td>
<td>59.0</td>
</tr>
<tr>
<td>Africa</td>
<td>Angola</td>
<td>37.9</td>
</tr>
<tr>
<td>Africa</td>
<td>Benin</td>
<td>48.8</td>
</tr>
<tr>
<td>Africa</td>
<td>Botswana</td>
<td>54.6</td>
</tr>
<tr>
<td>Africa</td>
<td>Burkina Faso</td>
<td>44.7</td>
</tr>
<tr>
<td>Africa</td>
<td>Burundi</td>
<td>44.8</td>
</tr>
<tr>
<td>Africa</td>
<td>Cameroon</td>
<td>48.1</td>
</tr>
</tbody>
</table>

### Continent vs. avg_le_c

<table>
<thead>
<tr>
<th>continent</th>
<th>avg_le_c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>48.9</td>
</tr>
<tr>
<td>Americas</td>
<td>64.7</td>
</tr>
<tr>
<td>Asia</td>
<td>60.1</td>
</tr>
<tr>
<td>Europe</td>
<td>71.9</td>
</tr>
<tr>
<td>Oceania</td>
<td>74.3</td>
</tr>
</tbody>
</table>

### Continent vs. avg_le

<table>
<thead>
<tr>
<th>continent</th>
<th>avg_le</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>64.0</td>
</tr>
</tbody>
</table>

<https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr>
More Summary Functions

```r
group_by(gapminder, country) %>%
  summarise(year = first(year), le = first(lifeExp))
```

<table>
<thead>
<tr>
<th>country</th>
<th>year</th>
<th>le</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>1952</td>
<td>28.8</td>
</tr>
<tr>
<td>Albania</td>
<td>1952</td>
<td>55.2</td>
</tr>
<tr>
<td>Algeria</td>
<td>1952</td>
<td>43.1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

```r
group_by(gapminder, country) %>%
  summarise(year = last(year), le = last(lifeExp))
```

```r
group_by(gapminder, country) %>%
  summarise(year = nth(year, 3), le = nth(lifeExp, 3))
```

<table>
<thead>
<tr>
<th>country</th>
<th>year</th>
<th>le</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>1962</td>
<td>32.0</td>
</tr>
<tr>
<td>Albania</td>
<td>1962</td>
<td>64.8</td>
</tr>
<tr>
<td>Algeria</td>
<td>1962</td>
<td>48.3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Using summarise() across columns

# Apply one or more functions to one or more columns.
# Grouping variables are always excluded from modification.

# Note that summarise_at() has been deprecated (on its way out, as of dplyr 1.0)

```r
group_by(gapminder, continent, year) %>%
summarise_at(c("lifeExp", "pop"), funs(min, median, max)) %>% View()
```

# for dplyr 1.0 and later use across()

```r
group_by(gapminder, continent, year) %>%
summarise(across(c(lifeExp, pop), list(mean=mean, median=median, max=max))) %>% View()
```

<table>
<thead>
<tr>
<th>continent</th>
<th>year</th>
<th>lifeExp_min</th>
<th>lifeExp_median</th>
<th>lifeExp_max</th>
<th>pop_min</th>
<th>pop_median</th>
<th>pop_max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>1952</td>
<td>30.000</td>
<td>38.8330</td>
<td>52.724</td>
<td>60011</td>
<td>2668125</td>
<td>33119096</td>
</tr>
<tr>
<td>Africa</td>
<td>1957</td>
<td>31.570</td>
<td>40.5925</td>
<td>58.089</td>
<td>61325</td>
<td>2885791</td>
<td>37173340</td>
</tr>
<tr>
<td>Africa</td>
<td>1962</td>
<td>32.767</td>
<td>42.6305</td>
<td>60.246</td>
<td>65345</td>
<td>3145210</td>
<td>41871351</td>
</tr>
<tr>
<td>Africa</td>
<td>1967</td>
<td>34.113</td>
<td>44.6985</td>
<td>61.557</td>
<td>70787</td>
<td>3473693</td>
<td>47287752</td>
</tr>
</tbody>
</table>

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Graphing Results of Multiple Functions

# graph min, median, max country life expectancy, by continent

group_by(gapminder, continent, year) %>%
summarise(across(c(lifeExp, pop),
list(min=min, median=median, max=max))) %>%

ggplot(aes(x=year, y = lifeExp_median)) + geom_line() +
ggplot(aes(y = lifeExp_min), linetype = "dashed") +
ggplot(aes(y = lifeExp_max), linetype = "dashed") +
facet_grid(continent ~ .) +
labs(y="Life Expectancy for Countries: min, median, max",
x="")
# count() function wraps up the
# common combination of group_by() and summarise()

# How many rows for each value of continent?
count(gapminder, continent)

# How many rows for each value of continent and year?
count(gapminder, continent, year) %>% View()

# How many rows for each continent, for years 2002 and 2007?
filter(gapminder, year == 2002 | year == 2007) %>%
count(continent)
**dpolyr Commands to Combine/Compare Data Sets**

- `left_join()`
- `right_join()` - potentially add variables (columns) to data sets, making them wider
- `inner_join()`
- `full_join()`
- `semi_join()` - potentially remove observations (rows) from data sets, making them shorter
- `anti_join()`
- `bind_rows()` - add observations (rows) to data sets, making them longer

Combining Data Sets: `left_join()`

# saw above that every country has been associated with just one continent during time period
# so ...

# continent belongs in a table where unit of observation is country
# other variables belong in a table where unit of observation is country-year:

country_continent <- select(gapminder, country, continent) %>% distinct()
country_continent
tgap <- select(gapminder, -continent)  # tidy version of gapminder data
tgap

# BUT descriptive exploration has required
# continent be included in data set for grouping
# HOW TO COMBINE ("join" or "merge") tgap and country_continent?

# join matching rows from second data set to first
left_join(tgap, country_continent, by = "country") %>% View()
Combining Data Sets: `left_join()` and `right_join()`

country_continent_inc <- slice(country_continent, 6:142) # cut out rows 1-5
View(country_continent_inc)

tgap_inc <- slice(tgap, 49:144) # cut out rows 1-48 and rows 145-1704
View(tgap_inc)

# join matching rows from 2nd data set to first
left_join(tgap, country_continent_inc, by = "country") %>% View()

# join matching rows from first data set to 2nd
right_join(tgap, country_continent, by = "country") %>% View()

right_join(tgap, country_continent_inc, by = "country") %>% View()
# country_continent_inc is the driver ... result does not contain first 5 countries

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Combining Data Sets

inner_join(), full_join(), semi_join(), anti_join()

# join and retain only rows in both data sets
inner_join(tgap_inc, country_continent_inc, by = "country") %>% View()

# join and retain all values, all rows
full_join(tgap_inc, country_continent_inc, by = "country") %>% View()

# retain all rows in first data set that have a match in second data set
# (but don't add columns)
semi_join(tgap_inc, country_continent_inc, by = "country") %>% View()

# retain all rows in first data set that do not have a match in second data set
# (but don't add columns)
anti_join(tgap_inc, country_continent_inc, by = "country") %>% View()

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Appending Data Sets

# TO APPEND ROWS use bind_rows() ... more efficient than rbind()

tgap1992 <- filter(tgap, year == 1992) %>% select(-year)
tgap1997 <- filter(tgap, year == 1997) %>% select(-year)
tgap2002 <- filter(tgap, year == 2002) %>% select(-year)
tgap2007 <- filter(tgap, year == 2007) %>% select(-year)

tgap1992
tgap2007

bind_rows(tgap1992, tgap1997, tgap2002, tgap2007) %>% View() # ... OOPS .. not quite right!

bind_rows(tgap1992, tgap1997, tgap2002, tgap2007, .id="id") %>% View() # ... a bit better!

Where to Learn More

“official documentation”

https://dplyr.tidyverse.org/

https://www.tidyverse.org/blog/2020/06/dplyr-1-0-0/

https://cran.r-project.org/web/packages/dplyr/dplyr.pdf

https://cran.r-project.org/web/packages/dplyr/vignettes/dplyr.html

https://cran.r-project.org/web/packages/dplyr/vignettes/base.html

https://cran.r-project.org/web/packages/dplyr/vignettes/grouping.html

https://cran.r-project.org/web/packages/dplyr/vignettes/two-table.html

Data transformation with dplyr cheat sheet:

https://www.rstudio.com/resources/cheatsheets/

Chapter 5 of 2017 book R for Data Science by Hadley Wickham:

https://r4ds.had.co.nz/transform.html