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++)
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

https://pop.princeton.edu/events/2022/r-data-wrangling-tidyverse-packages-tidyr-dplyr
Load gapminder tibble

# 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>
</tbody>
</table>

... (omitted for brevity) ...
Subset Observations

\begin{verbatim}
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)
\end{verbatim}

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

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
# 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)

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

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

\[
\text{mutate(gapminder, popMil = round(pop / 1000000, 1), le = round(lifeExp, 0))}
\]

# 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>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[
\text{transmute(gapminder, country = country, y = year,}

\text{popMil = round(pop / 1000000, 1), le = round(lifeExp, 0)) %>%}
\]

\[
\text{arrange(y, country)}
\]

# 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>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
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()

```
country continent year lifeExp pop gdpPercap popMil cumpopMil
1 Algeria Africa 1952 43.077 9279525 2449.0082 9.3 9.3
2 Angola Africa 1952 30.015 4232095 3520.6103 4.2 13.5
3 Benin Africa 1952 38.223 1738315 1062.7522 1.7 15.2
4 Botswana Africa 1952 47.622 442308 851.2411 0.4 15.6
... ... ... ... ... ... ... ...
142 New Zealand Oceania 1952 69.390 1994794 10556.5757 2.0 2406.6
```

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

```
country continent year lifeExp pop gdpPercap popMil cumpopMil
1 Algeria Africa 2007 72.301 33333216 6223.3675 33.3 33.3
2 Angola Africa 2007 42.731 12420476 4797.2313 12.4 45.7
... ... ... ... ... ... ... ...
142 New Zealand Oceania 2007 80.204 4115771 25185.0091 4.1 6251.1
```

Group Data:
Construct New Column Values By Group

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

<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>cumpopMil</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sao Tome and Principe</td>
<td>Africa</td>
<td>2007</td>
<td>65.528</td>
<td>199579</td>
<td>1598.4351</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Djibouti</td>
<td>Africa</td>
<td>2007</td>
<td>54.791</td>
<td>496374</td>
<td>2082.4816</td>
<td>0.5</td>
<td>0.7</td>
</tr>
<tr>
<td>Equatorial Guinea</td>
<td>Africa</td>
<td>2007</td>
<td>51.579</td>
<td>551201</td>
<td>12154.0897</td>
<td>0.6</td>
<td>1.3</td>
</tr>
<tr>
<td>Comoros</td>
<td>Africa</td>
<td>2007</td>
<td>65.152</td>
<td>710960</td>
<td>986.1479</td>
<td>0.7</td>
<td>2.0</td>
</tr>
<tr>
<td>Reunion</td>
<td>Africa</td>
<td>2007</td>
<td>76.442</td>
<td>798094</td>
<td>7670.1226</td>
<td>0.8</td>
<td>2.8</td>
</tr>
<tr>
<td>Swaziland</td>
<td>Africa</td>
<td>2007</td>
<td>39.613</td>
<td>1133066</td>
<td>4513.4806</td>
<td>1.1</td>
<td>3.9</td>
</tr>
<tr>
<td>Nigeria</td>
<td>Africa</td>
<td>2007</td>
<td>46.869</td>
<td>135031164</td>
<td>2013.9773</td>
<td>135.0</td>
<td>929.6</td>
</tr>
<tr>
<td>Trinidad and Tobago</td>
<td>Americas</td>
<td>2007</td>
<td>69.819</td>
<td>1056608</td>
<td>18008.5092</td>
<td>1.1</td>
<td>1.1</td>
</tr>
</tbody>
</table>

```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)) %>% View()
```
Group Data

# list only rows that experienced a population decline during the previous 5 years
# show country, year, pop, pop_chg, pop_pctchg

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), n_countries = 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), n_countries = n(),
             avg_country_le = mean(lifeExp), sd_country_le = sd(lifeExp))

filter(gapminder, year == 1952) %>%
  group_by(continent) %>%
  summarise(year = mean(year), n_countries = 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
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))
summarise() “Peels Off” group_by()

group_by(gapminder, continent, country) %>%
summarise(avg_le_cc = mean(lifeExp))

<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>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<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>

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

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

<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>

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

# for dplyr 1.0 and later use across()
group_by(gapminder, continent, year) %>%
summarise(across(c(lifeExp, pop), list(mean=mean, median=median, max=max))) %>% View()

continent  year  lifeExp_min  lifeExp_median  lifeExp_max  pop_min  pop_median  pop_max
1 Africa  1952    30.000       38.8330       52.724     60011  2668125  33119096
2 Africa  1957    31.570       40.5925       58.089     61325  2885791  37173340
3 Africa  1962    32.767       42.6305       60.246     65345  3145210  41871351
4 Africa  1967    34.113       44.6985       61.557     70787  3473693  47287752
.    ...    ...       ...          ...       ...     ...      ...     ...
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() +
geom_line(aes(y = lifeExp_min), linetype = "dashed") +
geom_line(aes(y = lifeExp_max), linetype = "dashed") +
facet_grid(continent ~ .) +
labs(y="Life Expectancy for Countries: min, median, max",
x="")
count() function

# 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)

<table>
<thead>
<tr>
<th>continent</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>624</td>
</tr>
<tr>
<td>Americas</td>
<td>300</td>
</tr>
<tr>
<td>Asia</td>
<td>396</td>
</tr>
<tr>
<td>Europe</td>
<td>360</td>
</tr>
<tr>
<td>Oceania</td>
<td>24</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>continent</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>104</td>
</tr>
<tr>
<td>Americas</td>
<td>50</td>
</tr>
<tr>
<td>Asia</td>
<td>66</td>
</tr>
<tr>
<td>Europe</td>
<td>60</td>
</tr>
<tr>
<td>Oceania</td>
<td>4</td>
</tr>
</tbody>
</table>

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

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