The verb **`filter`** lets us choose rows based on a column's values.

Use the tuberculosis (TB) long form data to explore the use of **`filter`**. If you're continuing from where you left off, the TB long form data should be available in your R session under the name `tb_long`. If not, download `tb_long.rds` (https://github.com/datascienceprogram/ids_course_data/blob/master/tb_long.rds) and store it in your project data folder e.g. **first_project/data/tb_long.rds**.

Once you've done this, open up RStudio and create a new R Markdown file. You can then load the collection of **`tidyverse`** packages and read the TB long form data by running the following code chunk:

```
library(tidyverse)
tb_long <- read_rds("data/tb_long.rds")
tb_long
```

```
## # A tibble: 157,820 x 7
##    country     iso3   year type   sex   age_group count
##    <chr>       <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Afghanistan AFG    1980 new_sp m     04           NA
##  2 Afghanistan AFG    1981 new_sp m     04           NA
##  3 Afghanistan AFG    1982 new_sp m     04           NA
##  4 Afghanistan AFG    1983 new_sp m     04           NA
##  5 Afghanistan AFG    1984 new_sp m     04           NA
##  6 Afghanistan AFG    1985 new_sp m     04           NA
##  7 Afghanistan AFG    1986 new_sp m     04           NA
##  8 Afghanistan AFG    1987 new_sp m     04           NA
##  9 Afghanistan AFG    1988 new_sp m     04           NA
## 10 Afghanistan AFG    1989 new_sp m     04           NA
## # ... with 157,810 more rows
```

You can use **`filter`** to select rows based on the country variable being **equal** to "Australia". Try it with the following code chunk:

```
tb_au <- filter(tb_long, country == "Australia")
```

The `==` does a logical equals check. On your own, consider experimenting with just a single equal sign, `=`. What happens?

Using **`filter`** requires some comparison to find the subset of observations you are interested in and requires the comparison to return a **TRUE/FALSE** answer.

When you're ready, copy and run the following code chunks:

```
filter(tb_long, country != "Australia")
```

```
## # A tibble: 157,080 x 7
##    country     iso3   year type   sex   age_group count
##    <chr>       <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Afghanistan AFG    1980 new_sp m     04           NA
##  2 Afghanistan AFG    1981 new_sp m     04           NA
##  3 Afghanistan AFG    1982 new_sp m     04           NA
##  4 Afghanistan AFG    1983 new_sp m     04           NA
##  5 Afghanistan AFG    1984 new_sp m     04           NA
##  6 Afghanistan AFG    1985 new_sp m     04           NA
##  7 Afghanistan AFG    1986 new_sp m     04           NA
##  8 Afghanistan AFG    1987 new_sp m     04           NA
##  9 Afghanistan AFG    1988 new_sp m     04           NA
## 10 Afghanistan AFG    1989 new_sp m     04           NA
## # ... with 157,070 more rows
```

The **`country != "Australia"`** code finds rows corresponding to all countries **except** “Australia”.

```
filter(tb_long, count > 10)
```

```
## # A tibble: 30,927 x 7
##    country   iso3   year type   sex   age_group count
##    <chr>     <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Angola    AGO    2011 new_sp m     04          108
##  2 Angola    AGO    2012 new_sp m     04           58
##  3 Argentina ARG    2006 new_sp m     04           19
##  4 Argentina ARG    2007 new_sp m     04           14
##  5 Argentina ARG    2008 new_sp m     04           11
##  6 Argentina ARG    2010 new_sp m     04           13
##  7 Argentina ARG    2011 new_sp m     04           50
##  8 Botswana  BWA    2009 new_sp m     04           12
##  9 Botswana  BWA    2010 new_sp m     04           11
## 10 Botswana  BWA    2011 new_sp m     04           14
## # ... with 30,917 more rows
```

The **`count > 10`** code finds all rows where the count has values **higher** than 10.

```
filter(tb_long, count >= 10)
```

```
## # A tibble: 31,507 x 7
##    country   iso3   year type   sex   age_group count
##    <chr>     <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Angola    AGO    2011 new_sp m     04          108
##  2 Angola    AGO    2012 new_sp m     04           58
##  3 Argentina ARG    2006 new_sp m     04           19
##  4 Argentina ARG    2007 new_sp m     04           14
##  5 Argentina ARG    2008 new_sp m     04           11
##  6 Argentina ARG    2010 new_sp m     04           13
##  7 Argentina ARG    2011 new_sp m     04           50
##  8 Botswana  BWA    2009 new_sp m     04           12
##  9 Botswana  BWA    2010 new_sp m     04           11
## 10 Botswana  BWA    2011 new_sp m     04           14
## # ... with 31,497 more rows
```

The **`count >= 10`** code find all rows where the count has values bigger than *or equal to* 10.

```
filter(tb_long, iso3 %in% c("AUS", "NZL", "IDN"))
```

```
## # A tibble: 2,220 x 7
##    country   iso3   year type   sex   age_group count
##    <chr>     <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Australia AUS    1980 new_sp m     04           NA
##  2 Australia AUS    1981 new_sp m     04           NA
##  3 Australia AUS    1982 new_sp m     04           NA
##  4 Australia AUS    1983 new_sp m     04           NA
##  5 Australia AUS    1984 new_sp m     04           NA
##  6 Australia AUS    1985 new_sp m     04           NA
##  7 Australia AUS    1986 new_sp m     04           NA
##  8 Australia AUS    1987 new_sp m     04           NA
##  9 Australia AUS    1988 new_sp m     04           NA
## 10 Australia AUS    1989 new_sp m     04           NA
## # ... with 2,210 more rows
```

The **`iso3 %in% c("AUS", "NZL", "IDN")`** code selects **all** rows where the iso3 country code is either AUS, NZL or IDN. If you are making multiple comparisons use **`%in%`** rather than **`==`**.

```
filter(tb_long, !(iso3 %in% c("AUS", "NZL", "IDN")))
```

```
## # A tibble: 155,600 x 7
##    country     iso3   year type   sex   age_group count
##    <chr>       <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Afghanistan AFG    1980 new_sp m     04           NA
##  2 Afghanistan AFG    1981 new_sp m     04           NA
##  3 Afghanistan AFG    1982 new_sp m     04           NA
##  4 Afghanistan AFG    1983 new_sp m     04           NA
##  5 Afghanistan AFG    1984 new_sp m     04           NA
##  6 Afghanistan AFG    1985 new_sp m     04           NA
##  7 Afghanistan AFG    1986 new_sp m     04           NA
##  8 Afghanistan AFG    1987 new_sp m     04           NA
##  9 Afghanistan AFG    1988 new_sp m     04           NA
## 10 Afghanistan AFG    1989 new_sp m     04           NA
## # ... with 155,590 more rows
```

The exclamation mark **`!`** performs a logical negation, so the **`!(iso3 %in% c("AUS", "NZL", "IDN"))`** expression selects **all** rows where the is03 country code is **not** AUS, NZL or IDN.

```
filter(tb_long, is.na(count))
```

```
## # A tibble: 106,846 x 7
##    country     iso3   year type   sex   age_group count
##    <chr>       <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Afghanistan AFG    1980 new_sp m     04           NA
##  2 Afghanistan AFG    1981 new_sp m     04           NA
##  3 Afghanistan AFG    1982 new_sp m     04           NA
##  4 Afghanistan AFG    1983 new_sp m     04           NA
##  5 Afghanistan AFG    1984 new_sp m     04           NA
##  6 Afghanistan AFG    1985 new_sp m     04           NA
##  7 Afghanistan AFG    1986 new_sp m     04           NA
##  8 Afghanistan AFG    1987 new_sp m     04           NA
##  9 Afghanistan AFG    1988 new_sp m     04           NA
## 10 Afghanistan AFG    1989 new_sp m     04           NA
## # ... with 106,836 more rows
```

The **`is.na(count)`** code finds all rows that are coded as missing values for the variable **`count`** . The absence of such values is normally indicated as NA. Since NA is not a value, you cannot use `==` to filter it.

```
filter(tb_long, !is.na(count) & country == "India")
```

```
## # A tibble: 252 x 7
##    country iso3   year type   sex   age_group count
##    <chr>   <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 India   IND    1995 new_sp m     014          16
##  2 India   IND    1996 new_sp m     014          47
##  3 India   IND    1997 new_sp m     014          50
##  4 India   IND    1998 new_sp m     014          84
##  5 India   IND    1999 new_sp m     014         327
##  6 India   IND    2000 new_sp m     014        1588
##  7 India   IND    2001 new_sp m     014        1063
##  8 India   IND    2002 new_sp m     014        2551
##  9 India   IND    2003 new_sp m     014        2411
## 10 India   IND    2004 new_sp m     014        3018
## # ... with 242 more rows
```

```
# OR
filter(tb_long, !is.na(count), country == "India")
```

```
## # A tibble: 252 x 7
##    country iso3   year type   sex   age_group count
##    <chr>   <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 India   IND    1995 new_sp m     014          16
##  2 India   IND    1996 new_sp m     014          47
##  3 India   IND    1997 new_sp m     014          50
##  4 India   IND    1998 new_sp m     014          84
##  5 India   IND    1999 new_sp m     014         327
##  6 India   IND    2000 new_sp m     014        1588
##  7 India   IND    2001 new_sp m     014        1063
##  8 India   IND    2002 new_sp m     014        2551
##  9 India   IND    2003 new_sp m     014        2411
## 10 India   IND    2004 new_sp m     014        3018
## # ... with 242 more rows
```

The `!is.na(count) & country == "India"` code selects **all** rows where there is **not** a missing count *and* the country is India. This can also be specified using a `,` inside `filter()` instead of the `&`.

```
filter(tb_long, !is.na(count) | country == "India")
```

```
## # A tibble: 51,462 x 7
##    country     iso3   year type   sex   age_group count
##    <chr>       <chr> <dbl> <chr>  <chr> <chr>     <dbl>
##  1 Afghanistan AFG    2010 new_sp m     04            4
##  2 Afghanistan AFG    2011 new_sp m     04            2
##  3 Afghanistan AFG    2012 new_sp m     04            0
##  4 Albania     ALB    2005 new_sp m     04            0
##  5 Albania     ALB    2006 new_sp m     04            1
##  6 Albania     ALB    2007 new_sp m     04            0
##  7 Albania     ALB    2008 new_sp m     04            1
##  8 Albania     ALB    2009 new_sp m     04            0
##  9 Albania     ALB    2010 new_sp m     04            0
## 10 Albania     ALB    2011 new_sp m     04            0
## # ... with 51,452 more rows
```

The `!is.na(count) | country == "India"` selects all rows where there is not a missing count *or* the country is India.

Piping %>%

Piping is particularly useful when using wrangling verbs like `filter`, `select`, `mutate` and others. The pipe operator is `%>%` and *takes everything before it and feeds it into the next step*, which greatly simplify your code when many wrangling verbs are required. Below is are examples of how the pipe operator is used to filter the `tb_long` data:

```
# Take tb_long and filter for all countries except Australia
tb_long %>% filter(country != "Australia")

# Take tb_long and filter for count greater than or equal to 10
tb_long %>% filter(count >= 10)

# Take tb_long and filter for iso3
filter(tb_long, iso3 %in% c("AUS", "NZL", "IDN"))
```

You'll learn more about pipes over the next few sections on wrangling data.

Give it a go!

Continue to develop your skills with `filter` by making your way through this exercise - working with tuberculosis (TB) data.

When you're ready, create a **new** R code chunk and use the verb `filter` to select TB cases **only** from:

- the year 2012
- the years 2013 - 2018
- countries in North America
- countries in North America but not ages 0-4, 0-14 or u.

To keep your work organised, use one R code chunk for each of the cases you need to `filter`.

Tell us how you went

Within the **Comments**, share with other learners the results from filtering. What did you find out about each case and how did the verb `filter` help you to identify this information?
