cm010 Exercises: Tibble Joins

Requirements

You will need Joey's `singer` R package for this exercise. And to install that, you'll need to install `devtools`. Running this code in your console should do the trick:

```
install.packages("devtools")
devtools::install_github("JoeyBernhardt/singer")
```

If you can't install the `singer` package, we've put the singer data on the `STAT545-UBC/Classroom` repo, and you can load those in instead by running the following lines of code:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------------- tidyverse 1.2.1
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'tidyr' was built under R version 3.5.2
```

```
## Warning: package 'purrr' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
## Warning: package 'forcats' was built under R version 3.5.2
```

```
## -- Conflicts ------------------------------------------------------------------ tidyverse_conflicts()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
songs <- read_csv("https://raw.githubusercontent.com/STAT545-UBC/Classroom/master/data/singer/songs.csv
```

```
## Parsed with column specification:
## cols(
##   title = col_character(),
##   artist_name = col_character(),
##   year = col_double()
## )
```

```
locations <- read_csv("https://raw.githubusercontent.com/STAT545-UBC/Classroom/master/data/singer/loc.c
```

```
## Parsed with column specification:
## cols(
##   artist_name = col_character(),
##   city = col_character(),
##   release = col_character(),
##   title = col_character()
## )
```

Load required packages:

Exercise 1: singer

The package singer comes with two smallish data frames about songs. Let's take a look at them (after minor modifications by renaming and shuffling):

```
(time <- as_tibble(songs) %>%
   rename(song = title))
```

```
## # A tibble: 22 x 3
##    song                             artist_name      year
##    <chr>                            <chr>           <dbl>
##  1 Corduroy                         Pearl Jam        1994
##  2 Grievance                        Pearl Jam        2000
##  3 Stupidmop                        Pearl Jam        1994
##  4 Present Tense                    Pearl Jam        1996
##  5 MFC                              Pearl Jam        1998
##  6 Lukin                            Pearl Jam        1996
##  7 It's Lulu                        The Boo Radleys  1995
##  8 Sparrow                          The Boo Radleys  1992
##  9 Martin_ Doom! It's Seven O'Clock The Boo Radleys  1995
## 10 Leaves And Sand                  The Boo Radleys  1993
## # ... with 12 more rows
```

```
(album <- as_tibble(locations) %>%
   select(title, everything()) %>%
   rename(album = release,
          song  = title))
```

```
## # A tibble: 14 x 4
##    song                        artist_name   city        album
##    <chr>                       <chr>         <chr>       <chr>
##  1 Grievance                   Pearl Jam     Seattle, WA Binaural
##  2 Stupidmop                   Pearl Jam     Seattle, WA Vitalogy
##  3 Present Tense               Pearl Jam     Seattle, WA No Code
##  4 MFC                         Pearl Jam     Seattle, WA Live On Two Legs
##  5 Lukin                       Pearl Jam     Seattle, WA Seattle Washingto~
##  6 Stuck On Amber              The Boo Radl~ Liverpool,~ Wake Up!
##  7 It's Lulu                   The Boo Radl~ Liverpool,~ Best Of
##  8 Sparrow                     The Boo Radl~ Liverpool,~ Everything's Alri~
##  9 High as Monkeys             The Boo Radl~ Liverpool,~ Kingsize
## 10 Butterfly McQueen           The Boo Radl~ Liverpool,~ Giant Steps
## 11 My One and Only Love        Carly Simon   New York, ~ Moonlight Serenade
## 12 It Was So Easy  (LP Versio~ Carly Simon   New York, ~ No Secrets
## 13 I've Got A Crush On You     Carly Simon   New York, ~ Clouds In My Coff~
## 14 "Manha De Carnaval (Theme ~ Carly Simon   New York, ~ Into White
```

1. We really care about the songs in time. But, which of those songs do we know its corresponding album?

```
(time %>%
  inner_join(album, by = "song"))
```

```
## # A tibble: 13 x 6
##    song             artist_name.x   year artist_name.y city     album
##    <chr>            <chr>          <dbl> <chr>         <chr>    <chr>
##  1 Grievance        Pearl Jam       2000 Pearl Jam     Seattle~ Binaural
##  2 Stupidmop        Pearl Jam       1994 Pearl Jam     Seattle~ Vitalogy
##  3 Present Tense    Pearl Jam       1996 Pearl Jam     Seattle~ No Code
##  4 MFC              Pearl Jam       1998 Pearl Jam     Seattle~ Live On Tw~
##  5 Lukin            Pearl Jam       1996 Pearl Jam     Seattle~ Seattle Wa~
##  6 It's Lulu        The Boo Radle~  1995 The Boo Radl~ Liverpo~ Best Of
##  7 Sparrow          The Boo Radle~  1992 The Boo Radl~ Liverpo~ Everything~
##  8 High as Monkeys  The Boo Radle~  1998 The Boo Radl~ Liverpo~ Kingsize
##  9 Butterfly McQue~ The Boo Radle~  1993 The Boo Radl~ Liverpo~ Giant Steps
## 10 My One and Only~ Carly Simon     2005 Carly Simon   New Yor~ Moonlight ~
## 11 It Was So Easy ~ Carly Simon     1972 Carly Simon   New Yor~ No Secrets
## 12 I've Got A Crus~ Carly Simon     1994 Carly Simon   New Yor~ Clouds In ~
## 13 "Manha De Carna~ Carly Simon     2007 Carly Simon   New Yor~ Into White
```

2. Go ahead and add the corresponding albums to the `time` tibble, being sure to preserve rows even if album info is not readily available.

```
(x <- time %>%
  left_join(album, by = "song"))
```

```
## # A tibble: 22 x 6
##    song          artist_name.x   year artist_name.y  city     album
##    <chr>         <chr>          <dbl> <chr>          <chr>    <chr>
##  1 Corduroy      Pearl Jam       1994 <NA>           <NA>     <NA>
##  2 Grievance     Pearl Jam       2000 Pearl Jam      Seattle~ Binaural
##  3 Stupidmop     Pearl Jam       1994 Pearl Jam      Seattle~ Vitalogy
##  4 Present Tense Pearl Jam       1996 Pearl Jam      Seattle~ No Code
##  5 MFC           Pearl Jam       1998 Pearl Jam      Seattle~ Live On Two ~
##  6 Lukin         Pearl Jam       1996 Pearl Jam      Seattle~ Seattle Wash~
##  7 It's Lulu     The Boo Radle~  1995 The Boo Radle~ Liverpo~ Best Of
##  8 Sparrow       The Boo Radle~  1992 The Boo Radle~ Liverpo~ Everything's~
##  9 Martin_ Doom~ The Boo Radle~  1995 <NA>           <NA>     <NA>
## 10 Leaves And S~ The Boo Radle~  1993 <NA>           <NA>     <NA>
## # ... with 12 more rows
```

3. Which songs do we have "year", but not album info?

```
(x %>%
   filter(album == "NA"))
```

```
## # A tibble: 0 x 6
## # ... with 6 variables: song <chr>, artist_name.x <chr>, year <dbl>,
## #   artist_name.y <chr>, city <chr>, album <chr>
```

4. Which artists are in `time`, but not in `album`?

```
time %>%
  anti_join(album, by = "artist_name")
```

```
## # A tibble: 5 x 3
##   song                   artist_name   year
##   <chr>                  <chr>        <dbl>
## 1 Mine Again             Mariah Carey  2005
## 2 Don't Forget About Us Mariah Carey  2005
## 3 Babydoll               Mariah Carey  1997
## 4 Don't Forget About Us Mariah Carey  2005
## 5 Vision Of Love         Mariah Carey  1990
```

5. You've come across these two tibbles, and just wish all the info was available in one tibble. What would you do?

```
time %>%
  full_join(album, by = "song")
```

```
## # A tibble: 23 x 6
##    song          artist_name.x   year artist_name.y  city     album
##    <chr>         <chr>          <dbl> <chr>          <chr>    <chr>
##  1 Corduroy      Pearl Jam       1994 <NA>           <NA>     <NA>
##  2 Grievance     Pearl Jam       2000 Pearl Jam      Seattle~ Binaural
##  3 Stupidmop     Pearl Jam       1994 Pearl Jam      Seattle~ Vitalogy
##  4 Present Tense Pearl Jam       1996 Pearl Jam      Seattle~ No Code
##  5 MFC           Pearl Jam       1998 Pearl Jam      Seattle~ Live On Two ~
##  6 Lukin         Pearl Jam       1996 Pearl Jam      Seattle~ Seattle Wash~
##  7 It's Lulu     The Boo Radle~  1995 The Boo Radle~ Liverpo~ Best Of
##  8 Sparrow       The Boo Radle~  1992 The Boo Radle~ Liverpo~ Everything's~
##  9 Martin_ Doom~ The Boo Radle~  1995 <NA>           <NA>     <NA>
## 10 Leaves And S~ The Boo Radle~  1993 <NA>           <NA>     <NA>
## # ... with 13 more rows
```

Exercise 2: LOTR

Load in the three Lord of the Rings tibbles that we saw last time:

```
fell <- read_csv("https://raw.githubusercontent.com/jennybc/lotr-tidy/master/data/The_Fellowship_Of_The_
```

```
## Parsed with column specification:
## cols(
##   Film = col_character(),
##   Race = col_character(),
##   Female = col_double(),
##   Male = col_double()
## )
```

```
ttow <- read_csv("https://raw.githubusercontent.com/jennybc/lotr-tidy/master/data/The_Two_Towers.csv")
```

```
## Parsed with column specification:
## cols(
##   Film = col_character(),
##   Race = col_character(),
##   Female = col_double(),
##   Male = col_double()
## )
```

```
retk <- read_csv("https://raw.githubusercontent.com/jennybc/lotr-tidy/master/data/The_Return_Of_The_King
```

```
## Parsed with column specification:
## cols(
##   Film = col_character(),
##   Race = col_character(),
##   Female = col_double(),
##   Male = col_double()
## )
```

1. Combine these into a single tibble.

```
full_join(fell, ttow) %>%
  full_join(retk)
```

```
## Joining, by = c("Film", "Race", "Female", "Male")
## Joining, by = c("Film", "Race", "Female", "Male")
```

```
## # A tibble: 9 x 4
##   Film                        Race   Female  Male
##   <chr>                       <chr>   <dbl> <dbl>
## 1 The Fellowship Of The Ring Elf      1229   971
## 2 The Fellowship Of The Ring Hobbit     14  3644
## 3 The Fellowship Of The Ring Man         0  1995
## 4 The Two Towers              Elf       331   513
## 5 The Two Towers              Hobbit      0  2463
## 6 The Two Towers              Man       401  3589
## 7 The Return Of The King      Elf       183   510
## 8 The Return Of The King      Hobbit      2  2673
## 9 The Return Of The King      Man       268  2459
```

2. Which races are present in "The Fellowship of the Ring" (`fell`), but not in any of the other ones?

```
fell %>%
  anti_join(ttow, by = "Race") %>%
  anti_join(retk, by = "Race")
```

```
## # A tibble: 0 x 4
## # ... with 4 variables: Film <chr>, Race <chr>, Female <dbl>, Male <dbl>
```

Exercise 3: Set Operations

Let's use three set functions: `intersect`, `union` and `setdiff`. We'll work with two toy tibbles named `y` and `z`, similar to Data Wrangling Cheatsheet

```
(y <-  tibble(x1 = LETTERS[1:3], x2 = 1:3))
```

```
## # A tibble: 3 x 2
##   x1       x2
##   <chr> <int>
## 1 A         1
## 2 B         2
## 3 C         3
```

```
(z <- tibble(x1 = c("B", "C", "D"), x2 = 2:4))
```

```
## # A tibble: 3 x 2
##   x1       x2
##   <chr> <int>
## 1 B         2
## 2 C         3
## 3 D         4
```

1. Rows that appear in both `y` and `z`

```
union(y, z)
```

```
## # A tibble: 4 x 2
##   x1       x2
##   <chr> <int>
## 1 A         1
## 2 B         2
## 3 C         3
## 4 D          4
```

2. You collected the data in `y` on Day 1, and `z` in Day 2. Make a data set to reflect that.

```
tibble(
  mutate(y, day = "Day 1"),
  mutate(z, day = "Day 2")
)
```

```
## # A tibble: 3 x 2
##   `mutate(y, day = "Day 1~   $x2 $day  `mutate(z, day = "Day 2~   $x2 $day
##   <chr>                     <int> <chr> <chr>                     <int> <chr>
## 1 A                             1 Day 1 B                             2 Day 2
## 2 B                             2 Day 1 C                             3 Day 2
## 3 C                             3 Day 1 D                             4 Day 2
```

3. The rows contained in `z` are bad! Remove those rows from `y`.

```
anti_join(y, z)
```

```
## Joining, by = c("x1", "x2")
```

```
## # A tibble: 1 x 2
##   x1       x2
##   <chr> <int>
## 1 A         1
```