Class Activity 7

Your name here

March 19 2024

Problem 1: Boolean Operators

Use Boolean operators to alter the code below to return only the rows that contain:

a. Girls named Rhea

```
filter(babynames, name == "Rhea", sex == "F")
# A tibble: 136 x 5
    year sex   name      n      prop
   <dbl> <chr> <chr> <int>     <dbl>
 1  1882 F     Rhea      7 0.0000605
 2  1883 F     Rhea      8 0.0000666
 3  1884 F     Rhea     13 0.0000945
 4  1885 F     Rhea     11 0.0000775
 5  1886 F     Rhea     13 0.0000846
 6  1887 F     Rhea     14 0.0000901
 7  1888 F     Rhea     20 0.000106
 8  1889 F     Rhea     31 0.000164
 9  1890 F     Rhea     39 0.000193
10  1891 F     Rhea     24 0.000122
# i 126 more rows
babynames %>%  filter(name == "Rhea", sex == "F")
# A tibble: 136 x 5
    year sex   name      n      prop
   <dbl> <chr> <chr> <int>     <dbl>
 1  1882 F     Rhea      7 0.0000605
 2  1883 F     Rhea      8 0.0000666
 3  1884 F     Rhea     13 0.0000945
 4  1885 F     Rhea     11 0.0000775
 5  1886 F     Rhea     13 0.0000846
 6  1887 F     Rhea     14 0.0000901
 7  1888 F     Rhea     20 0.000106
 8  1889 F     Rhea     31 0.000164
 9  1890 F     Rhea     39 0.000193
10  1891 F     Rhea     24 0.000122
# i 126 more rows
```

b. Names that were used by exactly 5 or 6 children in 1990

```
filter(babynames, year == 1990, n == 5 | n == 6)
# A tibble: 6,144 x 5
    year sex   name         n       prop
```

```
   <dbl> <chr> <chr>    <int>      <dbl>
 1  1990 F     Aariel       6 0.00000292
 2  1990 F     Aarion       6 0.00000292
 3  1990 F     Abagael      6 0.00000292
 4  1990 F     Abbye        6 0.00000292
 5  1990 F     Abiola       6 0.00000292
 6  1990 F     Abreanna     6 0.00000292
 7  1990 F     Abygail      6 0.00000292
 8  1990 F     Acadia       6 0.00000292
 9  1990 F     Adilenne     6 0.00000292
10  1990 F     Adriena      6 0.00000292
# i 6,134 more rows
babynames %>% filter(year == "1990", n == 5 | n == 6)
# A tibble: 6,144 x 5
    year sex   name         n       prop
   <dbl> <chr> <chr>    <int>      <dbl>
 1  1990 F     Aariel       6 0.00000292
 2  1990 F     Aarion       6 0.00000292
 3  1990 F     Abagael      6 0.00000292
 4  1990 F     Abbye        6 0.00000292
 5  1990 F     Abiola       6 0.00000292
 6  1990 F     Abreanna     6 0.00000292
 7  1990 F     Abygail      6 0.00000292
 8  1990 F     Acadia       6 0.00000292
 9  1990 F     Adilenne     6 0.00000292
10  1990 F     Adriena      6 0.00000292
# i 6,134 more rows
```

c. Names that are one of Apple, Yoroi, Ada

```
filter(babynames, name == "Apple" | name == "Yoroi" | name == "Ada")
# A tibble: 200 x 5
    year sex   name      n      prop
   <dbl> <chr> <chr> <int>     <dbl>
 1  1880 F     Ada     652 0.00668
 2  1881 F     Ada     628 0.00635
 3  1882 F     Ada     689 0.00596
 4  1883 F     Ada     778 0.00648
 5  1884 F     Ada     854 0.00621
 6  1885 F     Ada     876 0.00617
 7  1885 M     Ada       5 0.0000431
 8  1886 F     Ada     915 0.00595
 9  1886 M     Ada       6 0.0000504
10  1887 F     Ada     910 0.00586
# i 190 more rows
```

d. Store the data tibble in part c into a new tibble and change all the character columns to upper case. Also, rename the `n` variable to `count`.

```
aya <- babynames %>% filter(name == "Apple" | name == "Yoroi" | name == "Ada")
aya %>% mutate_if(is.character, toupper)
# A tibble: 200 x 5
    year sex   name      n      prop
```

```
   <dbl> <chr> <chr> <int>     <dbl>
 1  1880 F     ADA     652 0.00668
 2  1881 F     ADA     628 0.00635
 3  1882 F     ADA     689 0.00596
 4  1883 F     ADA     778 0.00648
 5  1884 F     ADA     854 0.00621
 6  1885 F     ADA     876 0.00617
 7  1885 M     ADA       5 0.0000431
 8  1886 F     ADA     915 0.00595
 9  1886 M     ADA       6 0.0000504
10  1887 F     ADA     910 0.00586
# i 190 more rows
aya %>% mutate_at(vars(name), toupper)
# A tibble: 200 x 5
    year sex   name      n      prop
   <dbl> <chr> <chr> <int>     <dbl>
 1  1880 F     ADA     652 0.00668
 2  1881 F     ADA     628 0.00635
 3  1882 F     ADA     689 0.00596
 4  1883 F     ADA     778 0.00648
 5  1884 F     ADA     854 0.00621
 6  1885 F     ADA     876 0.00617
 7  1885 M     ADA       5 0.0000431
 8  1886 F     ADA     915 0.00595
 9  1886 M     ADA       6 0.0000504
10  1887 F     ADA     910 0.00586
# i 190 more rows
aya %>% rename(count = n)
# A tibble: 200 x 5
    year sex   name  count      prop
   <dbl> <chr> <chr> <int>     <dbl>
 1  1880 F     Ada     652 0.00668
 2  1881 F     Ada     628 0.00635
 3  1882 F     Ada     689 0.00596
 4  1883 F     Ada     778 0.00648
 5  1884 F     Ada     854 0.00621
 6  1885 F     Ada     876 0.00617
 7  1885 M     Ada       5 0.0000431
 8  1886 F     Ada     915 0.00595
 9  1886 M     Ada       6 0.0000504
10  1887 F     Ada     910 0.00586
# i 190 more rows
```

e. Change all the column names to upper case in the previous problem.

```
aya %>% rename_at(vars(year:prop), toupper)
# A tibble: 200 x 5
    YEAR SEX   NAME      N      PROP
   <dbl> <chr> <chr> <int>     <dbl>
 1  1880 F     Ada     652 0.00668
 2  1881 F     Ada     628 0.00635
 3  1882 F     Ada     689 0.00596
 4  1883 F     Ada     778 0.00648
```

```
 5  1884 F     Ada     854 0.00621
 6  1885 F     Ada     876 0.00617
 7  1885 M     Ada       5 0.0000431
 8  1886 F     Ada     915 0.00595
 9  1886 M     Ada       6 0.0000504
10  1887 F     Ada     910 0.00586
# i 190 more rows
```

f. What do these commands do?

```
polluted_cities %>% select_if(is.numeric) #1
polluted_cities %>% rename_all(toupper) #2
polluted_cities %>% rename_if(is.character, toupper) #3
polluted_cities %>% rename_at(vars(contains("it")), toupper) #4
```

answer:

1. Selects all numeric columns from the polluted_cities dataset.
2. Renames all column names in the polluted_cities dataset to uppercase.
3. Renames column names with character data type in the polluted_cities dataset to uppercase.
4. Renames column names containing "it" in the polluted_cities dataset to uppercase.

```
polluted_cities %>% select_if(is.numeric) #1
# A tibble: 8 x 1
  amount
   <dbl>
1     23
2     14
3     22
4     16
5    121
6     56
7     32
8     16
polluted_cities %>% rename_all(toupper) #2
# A tibble: 8 x 3
  CITY     SIZE  AMOUNT
  <chr>    <chr>  <dbl>
1 New York large     23
2 New York small     14
3 London   large     22
4 London   small     16
5 Beijing  large    121
6 Beijing  small     56
7 Paris    large     32
8 Paris    small     16
polluted_cities %>% rename_if(is.character, toupper) #3
# A tibble: 8 x 3
  CITY     SIZE  amount
  <chr>    <chr>  <dbl>
1 New York large     23
2 New York small     14
3 London   large     22
4 London   small     16
```

```
5 Beijing  large    121
6 Beijing  small     56
7 Paris    large     32
8 Paris    small     16
polluted_cities %>% rename_at(vars(contains("it")), toupper) #4
# A tibble: 8 x 3
  CITY     size  amount
  <chr>    <chr>  <dbl>
1 New York large     23
2 New York small     14
3 London   large     22
4 London   small     16
5 Beijing  large    121
6 Beijing  small     56
7 Paris    large     32
8 Paris    small     16
```

Let's look at an interesting example on how to join related information on various artists, bands, songs, and their labels.

```
artists <- tibble(first = c("Jimmy", "George", "Mick", "Tom", "Davy", "John",
                            "Paul", "Jimmy", "Joe", "Elvis", "Keith", "Paul",
                            "Ringo", "Joe", "Brian", "Nancy"),
                  last = c("Buffett", "Harrison", "Jagger", "Jones", "Jones",
                           "Lennon", "McCartney", "Page", "Perry", "Presley",
                           "Richards", "Simon", "Starr", "Walsh", "Wilson", "Wilson"),
                  instrument = c("Guitar", "Guitar", "Vocals", "Vocals", "Vocals",
                                 "Guitar", "Bass", "Guitar", "Guitar", "Vocals", "Guitar",
                                 "Guitar", "Drums", "Guitar", "Vocals", "Vocals"))


bands <- tibble(first = c("John", "John Paul", "Jimmy", "Robert", "George", "John",
                          "Paul", "Ringo", "Jimmy", "Mick", "Keith", "Charlie", "Ronnie"),
                last = c("Bonham", "Jones", "Page", "Plant", "Harrison", "Lennon",
                         "McCartney", "Starr", "Buffett", "Jagger", "Richards", "Watts", "Wood"),
                band = c("Led Zeppelin", "Led Zeppelin", "Led Zeppelin", "Led Zeppelin",
                         "The Beatles", "The Beatles", "The Beatles", "The Beatles",
                         "The Coral Reefers", "The Rolling Stones", "The Rolling Stones",
                         "The Rolling Stones", "The Rolling Stones"))

albums <- tibble(album = c("A Hard Day's Night", "Magical Mystery Tour", "Beggar's Banquet",
                           "Abbey Road", "Led Zeppelin IV", "The Dark Side of the Moon", "Aerosmith",
                           "Rumours", "Hotel California"),
                 band = c("The Beatles", "The Beatles", "The Rolling Stones", "The Beatles",
                          "Led Zeppelin", "Pink Floyd", "Aerosmith", "Fleetwood Mac", "Eagles"),
                 year = c(1964,1967,1968,1969,1971,1973,1973,1977,1982))


songs <- tibble(song = c("Come Together", "Dream On", "Hello, Goodbye", "It's Not Unusual"),
                album  = c("Abbey Road", "Aerosmith", "Magical Mystery Tour", "Along Came Jones"),
                first = c("John", "Steven", "Paul", "Tom"),
                last = c("Lennon", "Tyler", "McCartney", "Jones"))
```

```
labels <- tibble(album = c("Abbey Road", "A Hard Days Night", "Magical Mystery Tour",
                           "Led Zeppelin IV", "The Dark Side of the Moon", "Hotel California",
                           "Rumours", "Aerosmith", "Beggar's Banquet"),
                 label = c("Apple", "Parlophone", "Parlophone", "Atlantic", "Harvest",
                           "Asylum", "Warner Brothers", "Columbia", "Decca"))
```

Let's take a glimpse of the tibbles `artists` and `bands`. Notice that there are different number of rows in the dataset.

```
glimpse(artists)
Rows: 16
Columns: 3
$ first      <chr> "Jimmy", "George", "Mick", "Tom", "Davy", "John", "Paul", "~
$ last       <chr> "Buffett", "Harrison", "Jagger", "Jones", "Jones", "Lennon"~
$ instrument <chr> "Guitar", "Guitar", "Vocals", "Vocals", "Vocals", "Guitar",~
glimpse(bands)
Rows: 13
Columns: 3
$ first <chr> "John", "John Paul", "Jimmy", "Robert", "George", "John", "Paul"~
$ last  <chr> "Bonham", "Jones", "Page", "Plant", "Harrison", "Lennon", "McCar~
$ band  <chr> "Led Zeppelin", "Led Zeppelin", "Led Zeppelin", "Led Zeppelin", ~
glimpse(albums)
Rows: 9
Columns: 3
$ album <chr> "A Hard Day's Night", "Magical Mystery Tour", "Beggar's Banquet"~
$ band  <chr> "The Beatles", "The Beatles", "The Rolling Stones", "The Beatles~
$ year  <dbl> 1964, 1967, 1968, 1969, 1971, 1973, 1973, 1977, 1982
glimpse(songs)
Rows: 4
Columns: 4
$ song  <chr> "Come Together", "Dream On", "Hello, Goodbye", "It's Not Unusual"
$ album <chr> "Abbey Road", "Aerosmith", "Magical Mystery Tour", "Along Came J~
$ first <chr> "John", "Steven", "Paul", "Tom"
$ last  <chr> "Lennon", "Tyler", "McCartney", "Jones"
glimpse(labels)
Rows: 9
Columns: 2
$ album <chr> "Abbey Road", "A Hard Days Night", "Magical Mystery Tour", "Led ~
$ label <chr> "Apple", "Parlophone", "Parlophone", "Atlantic", "Harvest", "Asy~
```

Problem 2: Joining artists and bands data

a. Join the artists and bands tibbles using `left_join()`, `right_join()`, and `full_join()`. Verify that the datasets obtained from `left_join()` and `right_join()` are the same using `setequal()`.

```
bands2 <- left_join(bands, artists, by = c("first", "last"))
bands3 <- right_join(artists, bands, by = c("first", "last"))
full_bands <- full_join(artists, bands, by = c("first", "last"))

# Check if the datasets are the same
setequal(bands2, bands3)
[1] TRUE
```

b. Use the pipe operator, %>%, to create one table that combines all information from `artists`, `bands`, `albums`, `songs`, and `labels`.

```
all_info <- artists %>%
  full_join(bands, by = c("first", "last")) %>%
  full_join(songs, by = c("first", "last")) %>%
  full_join(albums, by = c("album", "band")) %>%
  full_join(labels, by = c("album"))
all_info
# A tibble: 30 x 8
   first  last      instrument band               song         album  year label
   <chr>  <chr>     <chr>      <chr>              <chr>        <chr> <dbl> <chr>
 1 Jimmy  Buffett   Guitar     The Coral Reefers  <NA>         <NA>     NA <NA>
 2 George Harrison  Guitar     The Beatles        <NA>         <NA>     NA <NA>
 3 Mick   Jagger    Vocals     The Rolling Stones <NA>         <NA>     NA <NA>
 4 Tom    Jones     Vocals     <NA>               It's Not Un~ Alon~    NA <NA>
 5 Davy   Jones     Vocals     <NA>               <NA>         <NA>     NA <NA>
 6 John   Lennon    Guitar     The Beatles        Come Togeth~ Abbe~  1969 Apple
 7 Paul   McCartney Bass       The Beatles        Hello, Good~ Magi~  1967 Parl~
 8 Jimmy  Page      Guitar     Led Zeppelin       <NA>         <NA>     NA <NA>
 9 Joe    Perry     Guitar     <NA>               <NA>         <NA>     NA <NA>
10 Elvis  Presley   Vocals     <NA>               <NA>         <NA>     NA <NA>
# i 20 more rows
```

Problem 3: Filtering and counting rows in the data

a. Collect artists that have songs provided, and return rows of artists that don't have bands info.

```
# Artists with songs
artists_with_songs <- artists %>%
  semi_join(songs, by = c("first", "last"))

# Artists without bands info
artists_without_bands <- artists %>%
  anti_join(bands, by = c("first","last"))

artists_with_songs
# A tibble: 3 x 3
  first last      instrument
  <chr> <chr>     <chr>
1 Tom   Jones     Vocals
2 John  Lennon    Guitar
3 Paul  McCartney Bass
artists_without_bands
# A tibble: 8 x 3
  first last    instrument
  <chr> <chr>   <chr>
1 Tom   Jones   Vocals
2 Davy  Jones   Vocals
3 Joe   Perry   Guitar
4 Elvis Presley Vocals
```

```
5 Paul  Simon   Guitar
6 Joe   Walsh   Guitar
7 Brian Wilson  Vocals
8 Nancy Wilson  Vocals
```

b. Collect the albums made by a band, count the number of rows, find the rows of songs that match a row in labels, and count the number of rows.

```
# Albums made by a band
albums_by_band <- bands %>% semi_join(albums, by = "band")
n_albums_by_band <- nrow(albums_by_band)

# Rows of songs that match a row in labels
songs_with_labels <- songs %>% semi_join(labels, by = "album")
n_songs_with_labels <- nrow(songs_with_labels)

n_albums_by_band
[1] 12
n_songs_with_labels
[1] 3
```