# A summary of the tidyverse

Jaime Davila

2/2/2022

## Transforming and visualizing with tidyverse

`tidyverse` is a powerful collection of functions and libraries that allows us interact and wrangle datasets in an effective manner. Before we use any of those commands we need to instruct R to the `tidyverse` library which can be done with the following command:

```
library(tidyverse)
```

During this session we will be analyzing datasets from Defining the ’90s Music Cannon so we will start by using `read_csv` to load our dataset into the tibble `song.year.tbl`.

```
file.path="../data/song.year.csv"
#file.path="~/Mscs 341 S22/Class/Data/song.year.csv"
song.year.tbl <- read_csv(file.path)
song.year.tbl
```

```
## # A tibble: 344 x 3
##    artist            song                        year
##    <chr>             <chr>                      <dbl>
##  1 2 Pac             California Love             1996
##  2 2Pac              How Do U Want It            1996
##  3 702               Where My Girls At?          1999
##  4 Ace Of Base       All That She Wants          1993
##  5 Ace Of Base       Don’t Turn Around           1994
##  6 Ace Of Base       The Sign                    1994
##  7 Adina Howard      Freak Like Me               1995
##  8 Aerosmith         I Don’t Want To Miss A Thing 1998
##  9 Aerosmith         Janie’s Got A Gun           1990
## 10 Alanis Morissette Ironic                      1996
## # ... with 334 more rows
```

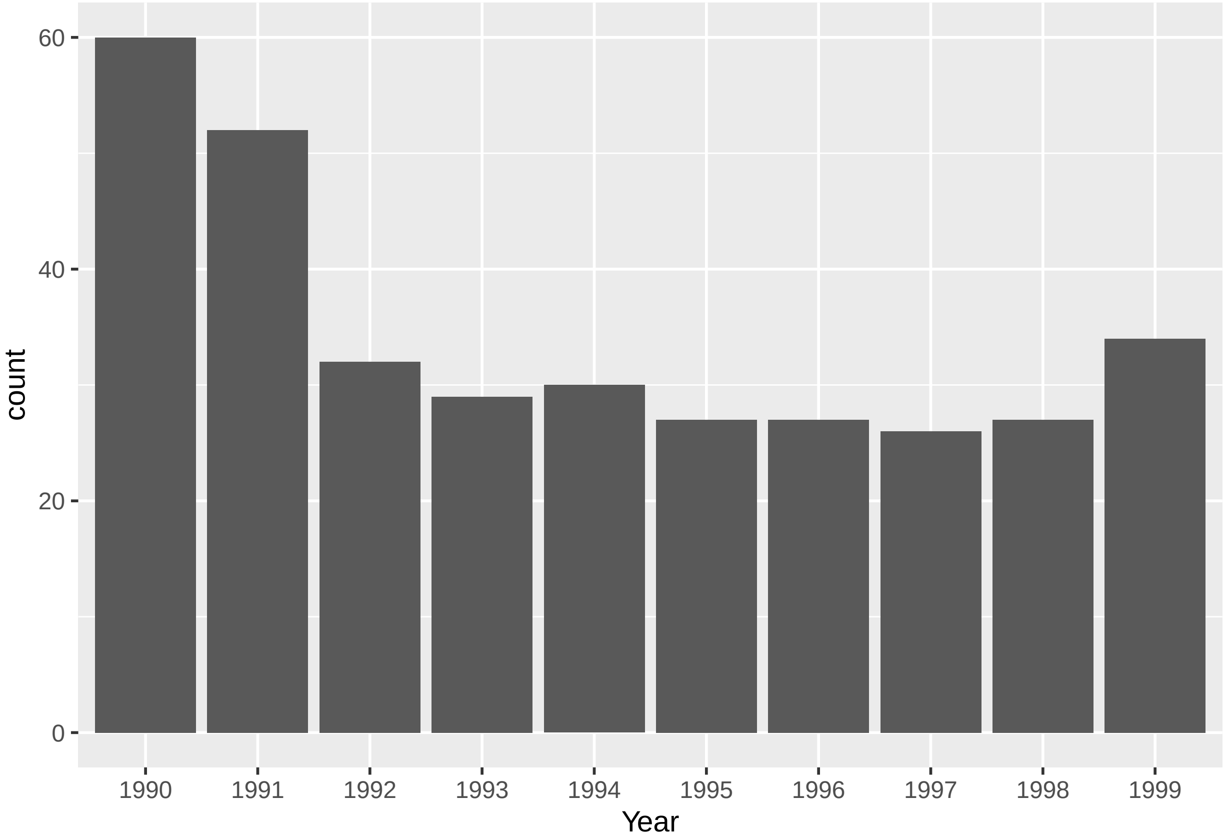
### A first plot

We will start by doing a simple plot summarizing the number of songs per year of our dataset by using `ggplot`. A good presentation of `ggplot` can be found in Chapter 3:Data Visualization. A couple of quick things to note about this code:

- We will be using `geom_histogram` as our geometric object

- Notice that sinceyear is a number we need to convert it to a categorical variable using the command as.factor

```
ggplot(song.year.tbl, aes(x=as.factor(year)))+
  geom_histogram(stat="count")+
  labs(x="Year")
```



## A first summary

We can obtain the number of songs by combining the commands group_by and summarize.Again notice that:

- We are using the pipe (%>%) to combine the two commands together.
- group_by can take as an argument any combination of the columns from the tibble.
- summarize is a very flexible command and allows the calculation of any number of statistics like average, minimum or maximum. In this particular case we are just counting the number of elements by using the function n().

```
song.year.tbl %>%
  group_by(year) %>%
  summarize(n=n())
```

```
## # A tibble: 10 x 2
##     year     n
##    <dbl> <int>
##  1  1990    60
##  2  1991    52
##  3  1992    32
##  4  1993    29
##  5  1994    30
##  6  1995    27
##  7  1996    27
##  8  1997    26
##  9  1998    27
## 10  1999    34
```

## Transforming datasets

Before attempting the following exercises we recommend that you read [Chapter 5: Data Transformation] (https://r4ds.had.co.nz/transform.html).

In particular we will be using the following 7 commands (also called verbs) from `tidyverse`

- `group_by`
- `summarize`
- `slice`
- `arrange`
- `select`
- `filter`
- `mutate`

1. In the following exercises we will modify our original dataset to answer specific questions:

    a. Generate a table with top-5 artists from the 90s according to their number of songs and call it `top5.tbl`.

    b. Let's explore Mariah Carey's career in depth. Start by summarizing the number of hits of Mariah Carey by year and find out what was her best year.

```
## # A tibble: 10 x 2
##     year     n
##    <dbl> <int>
##  1  1991     3
##  2  1990     2
##  3  1992     2
##  4  1993     2
##  5  1994     2
##  6  1995     2
##  7  1999     2
##  8  1996     1
##  9  1997     1
## 10  1998     1
```

```
c. What are the songs from her best year? Do you recognize any of them?
```

```
song.year.tbl %>%
  filter(artist == "Mariah Carey" & year==1991) %>%
  select(song)
```

```
## # A tibble: 3 x 1
##   song
##   <chr>
## 1 Emotions
## 2 I Don’t Wanna Cry
## 3 Someday
```

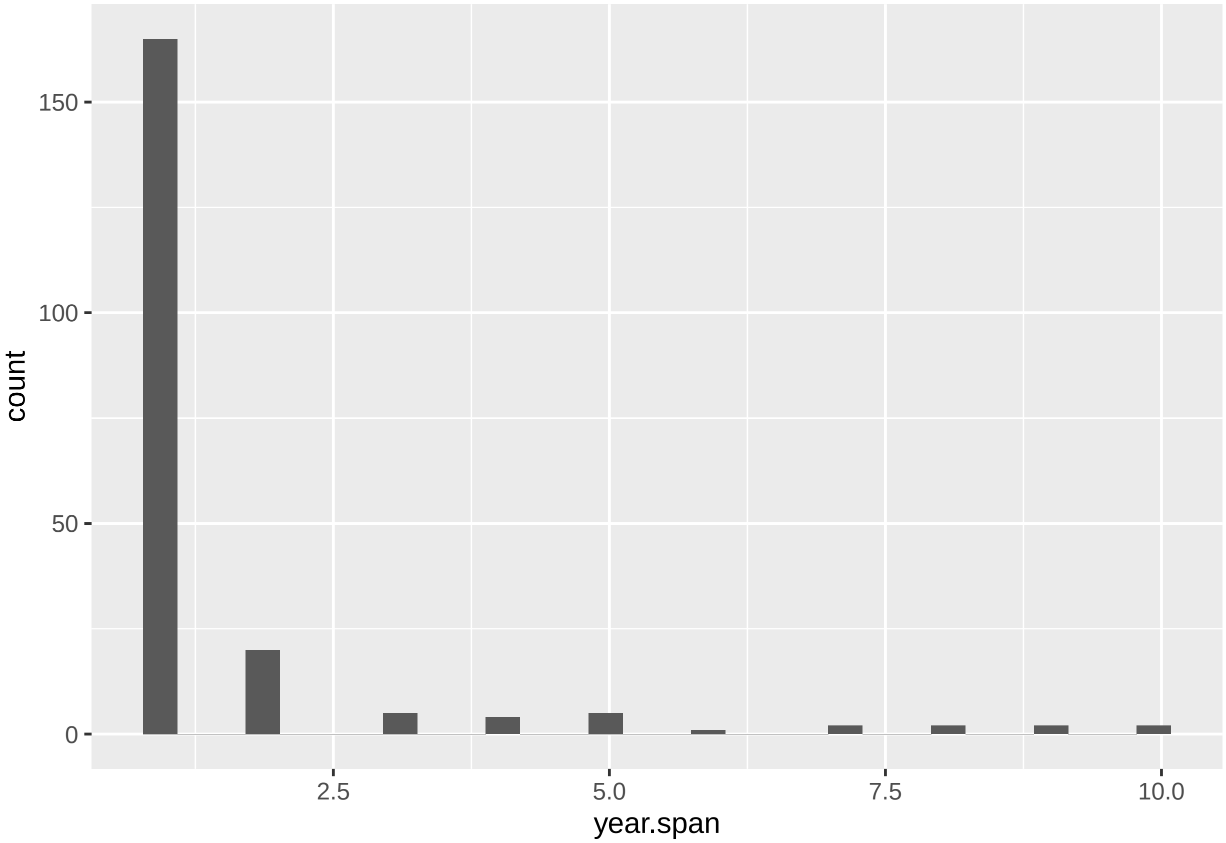
2. It seems like some artists like Mariah Carey had a song in the billboard in every year of the 90s decade, while others only had one hit in the entire decade.

   a. We are interested in calculating the `year.span` of an artist, which is basically defined as the difference in years between their latest and earliest song in the 90s. Create a table `artist.span.tbl` with such information and include `earliest.year` and `latest.year` as columns:

   b. What are the top-5 artists with biggest span?

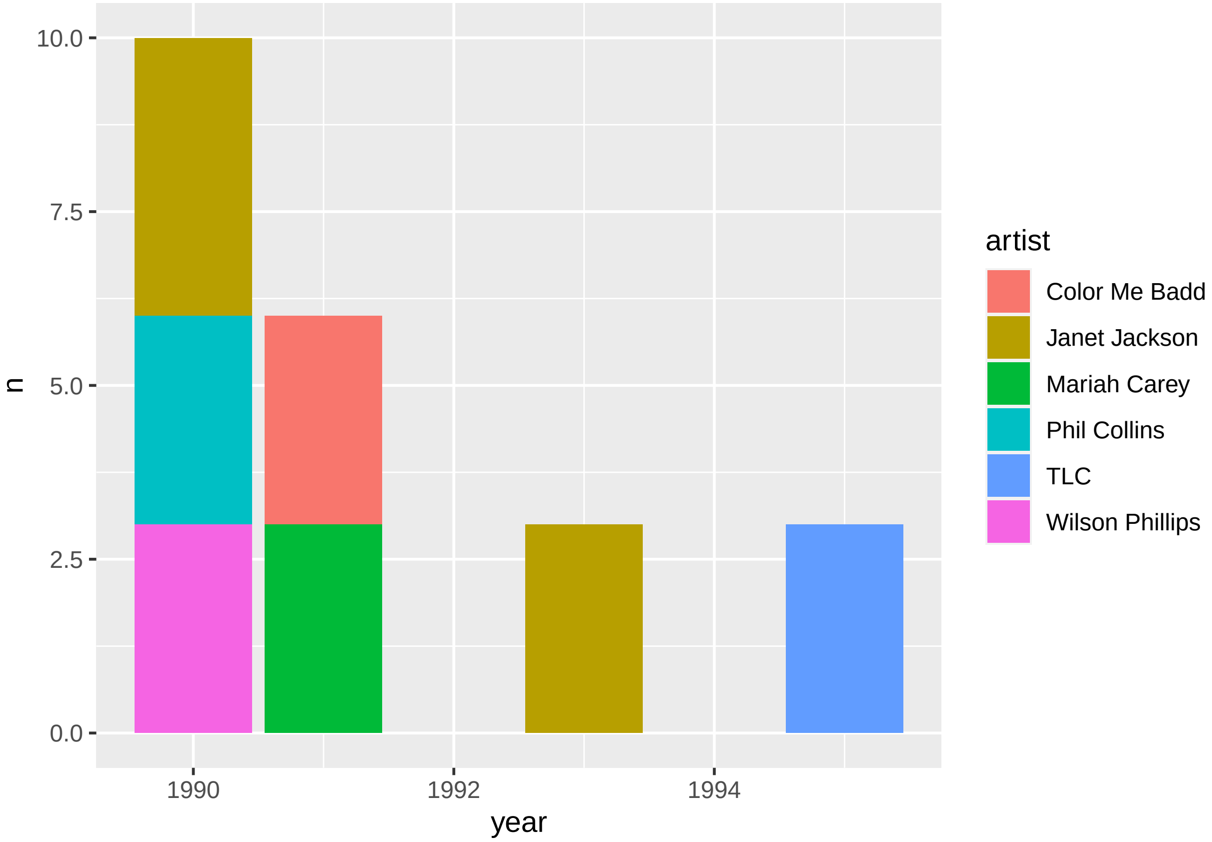
```
## # A tibble: 5 x 3
##   artist              n year.span
##   <chr>           <int>     <dbl>
## 1 Mariah Carey       18        10
## 2 Whitney Houston     9        10
## 3 Aerosmith           2         9
## 4 Madonna             8         9
## 5 Celine Dion         7         8
```

```
c. Generate a histogram describing the span across all the artist in our table:
```



3. Create a table with year and number of songs by artist and year. Only select artists that had at least 3 hits in a year. Generate a graph depicting this info.

```
## # A tibble: 7 x 3
## # Groups:   artist [6]
##   artist           year     n
##   <chr>           <dbl> <int>
## 1 Janet Jackson    1990     4
## 2 Color Me Badd    1991     3
## 3 Janet Jackson    1993     3
## 4 Mariah Carey     1991     3
## 5 Phil Collins     1990     3
## 6 TLC              1995     3
## 7 Wilson Phillips  1990     3
```



## Joinining datasets

A common situation in analysis is that all of the information is not contained in just a single dataset. Let's start by loading a different dataset which has recognition metrics for all of our previous songs

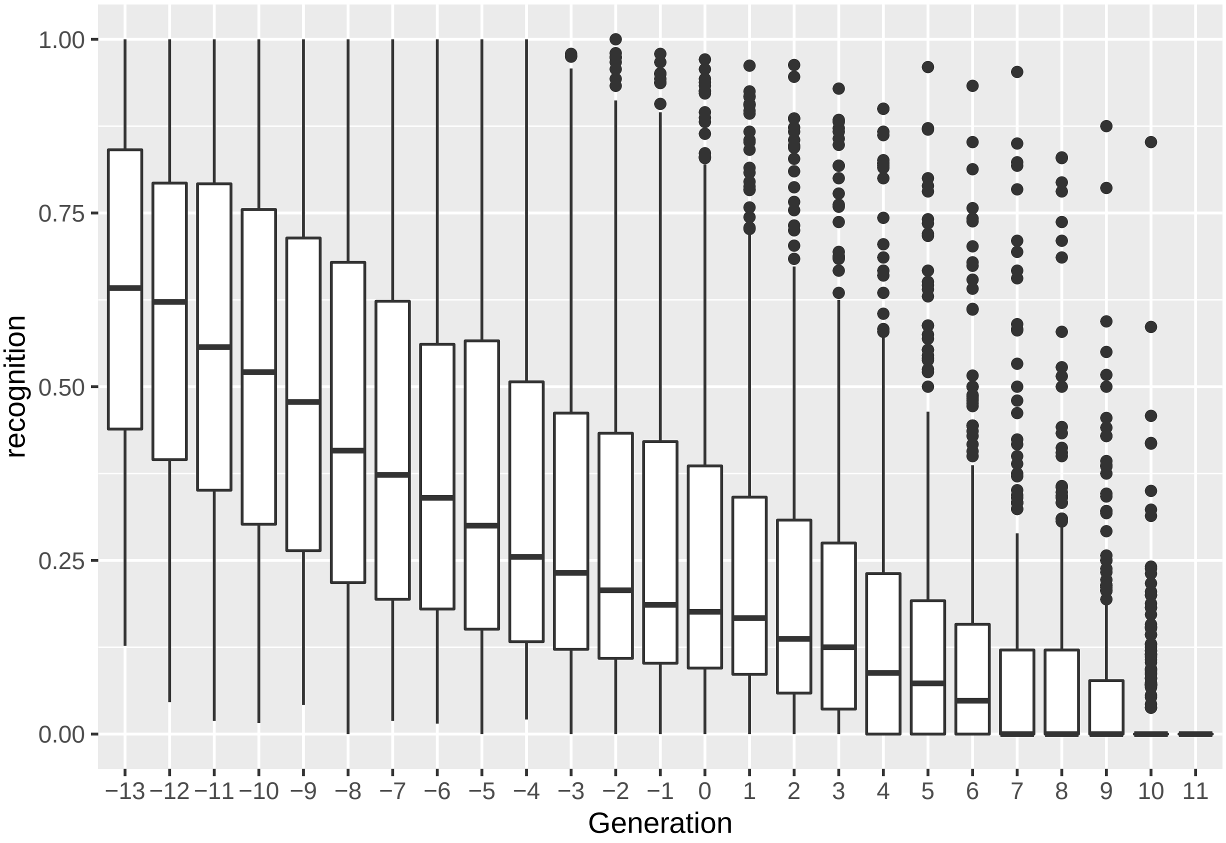
```
file.path="../data/song.recognition.csv"
#file.path="~/Mscs 341 S22/Class/Data/song.year.csv"

song.recognition.tbl <- read_csv(file.path)
```

Notice how `song.recognition.tbl` has `artist`, `song`, `generation` and `recognition` as variables. `generation` is measured as the age of the respondents when the song was released. For example, Macarena was released in 1996, so -10 represents the people who *were 10 years old* when it was released (and were born in 1986), while 5 represents the people who *were born 5 years after the song* (and were born in 2001).

Let's take a look at the entire dataset using a boxplot

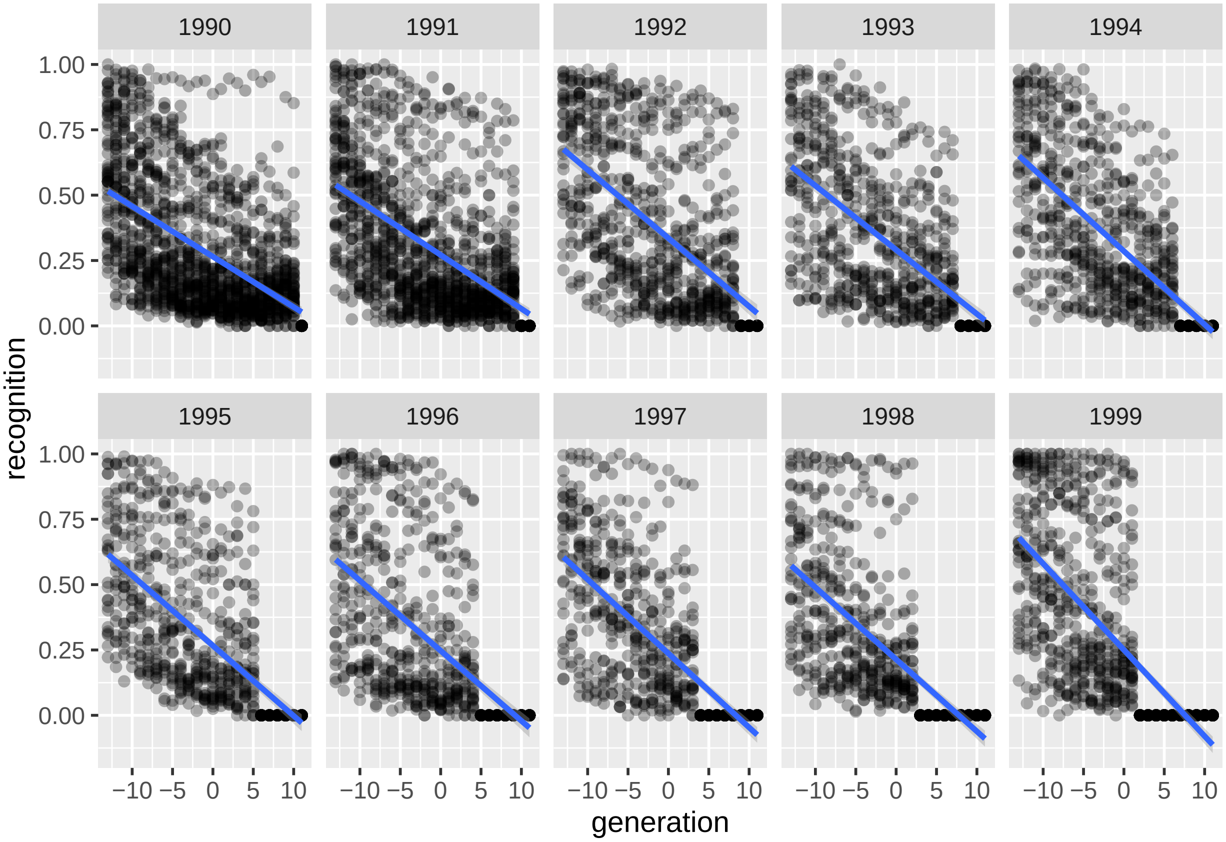
```
ggplot(song.recognition.tbl, aes(as.factor(generation), recognition)) +
  geom_boxplot()+
  labs(x="Generation")
```



In the following exercises we will explore this dataset in more detail. Make sure to consult Chapter 13:Relational data and pay close attention to the function `inner_join()`

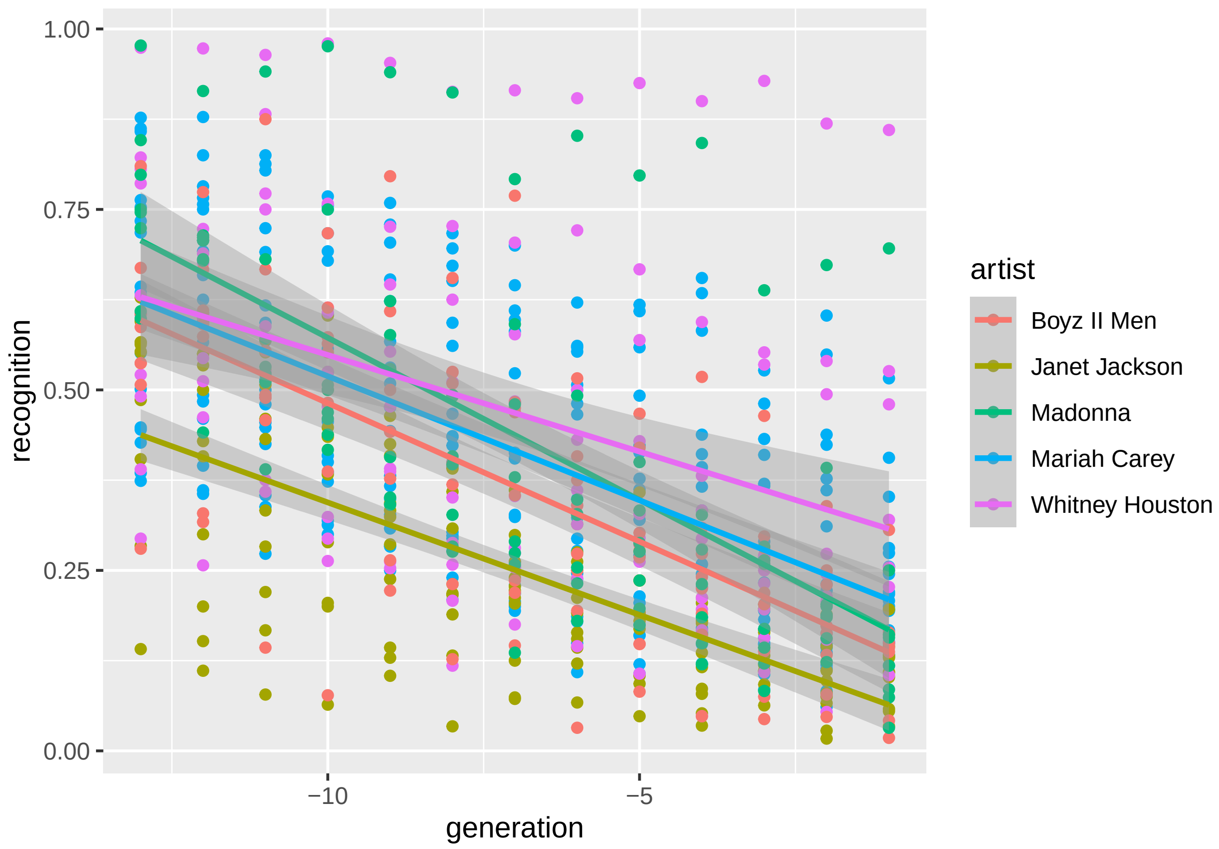
4. Let's explore the decaying trend in recognition is the same according to the year that the song came out. Let's do that using the following steps:

    1. Combine `song.recognition.tbl` and `song.year.tbl` in a single table called `song.decay.tbl`
    2. Create the following which shows the trends for each different year. As the years get closer to the end of the decade we get a lot of observations where the y value is 0. Can you explain why?

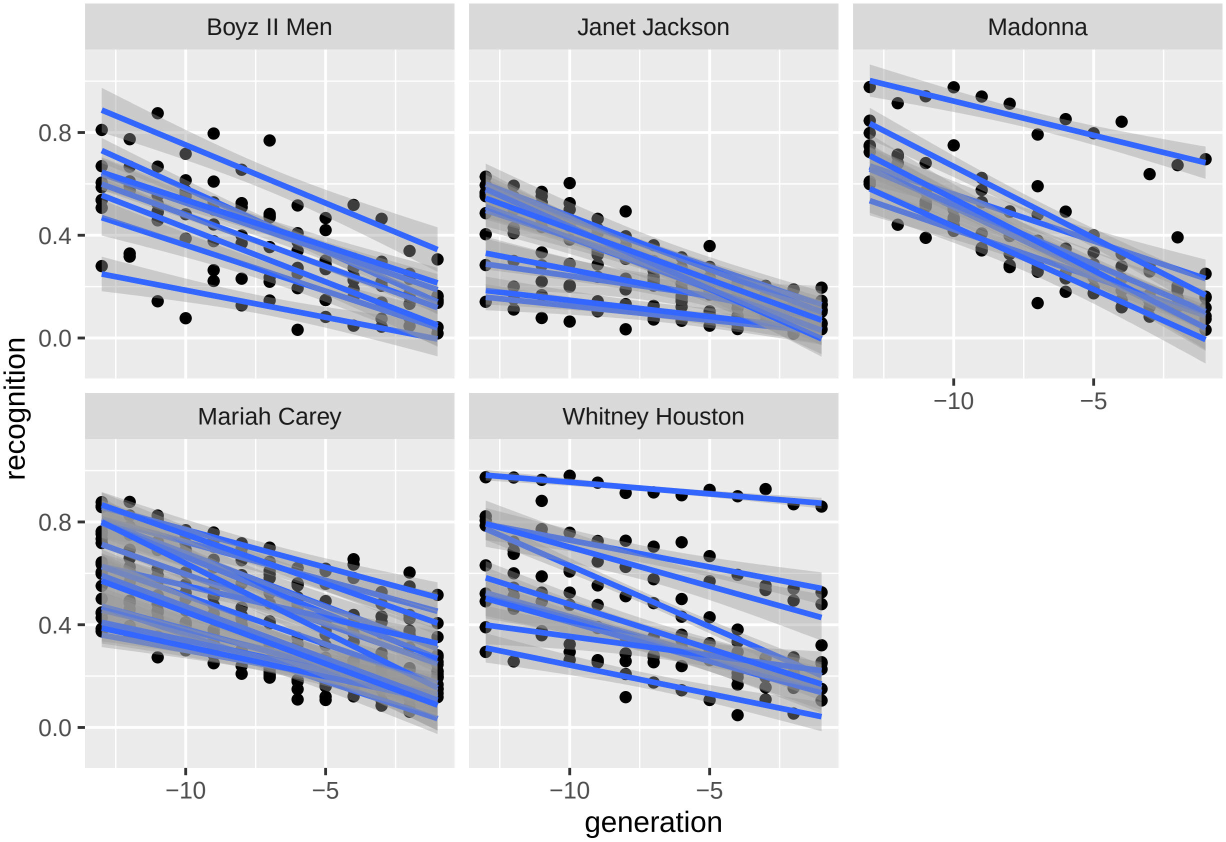


5. Let's look at the trends for the top 5 most popular artist by doing a join with `top5.tbl` and subset to only negative generations.

    1. Plot the trends for every artist. Who is the most recognized artist (dare I say *diva*) from the 90s?

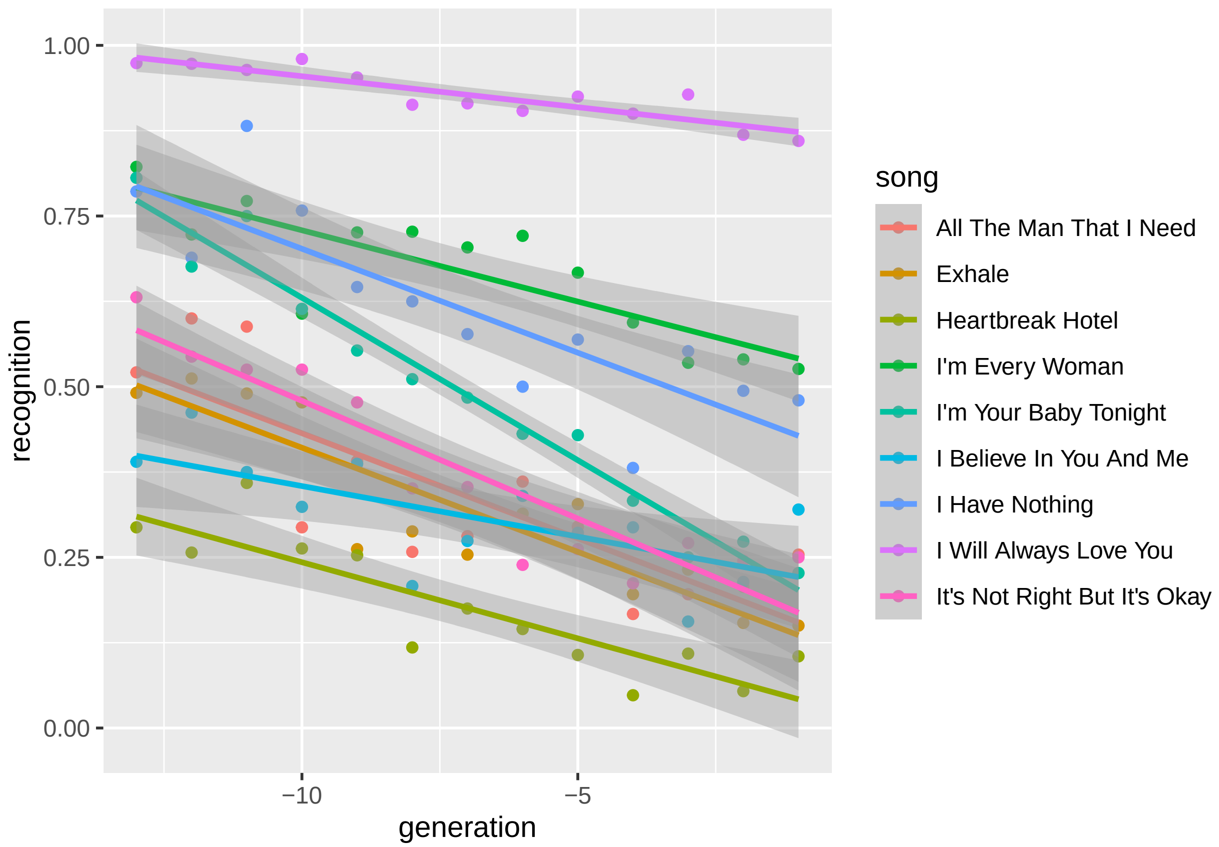


2. Look at the individual trends for each artist by generating the following plot.

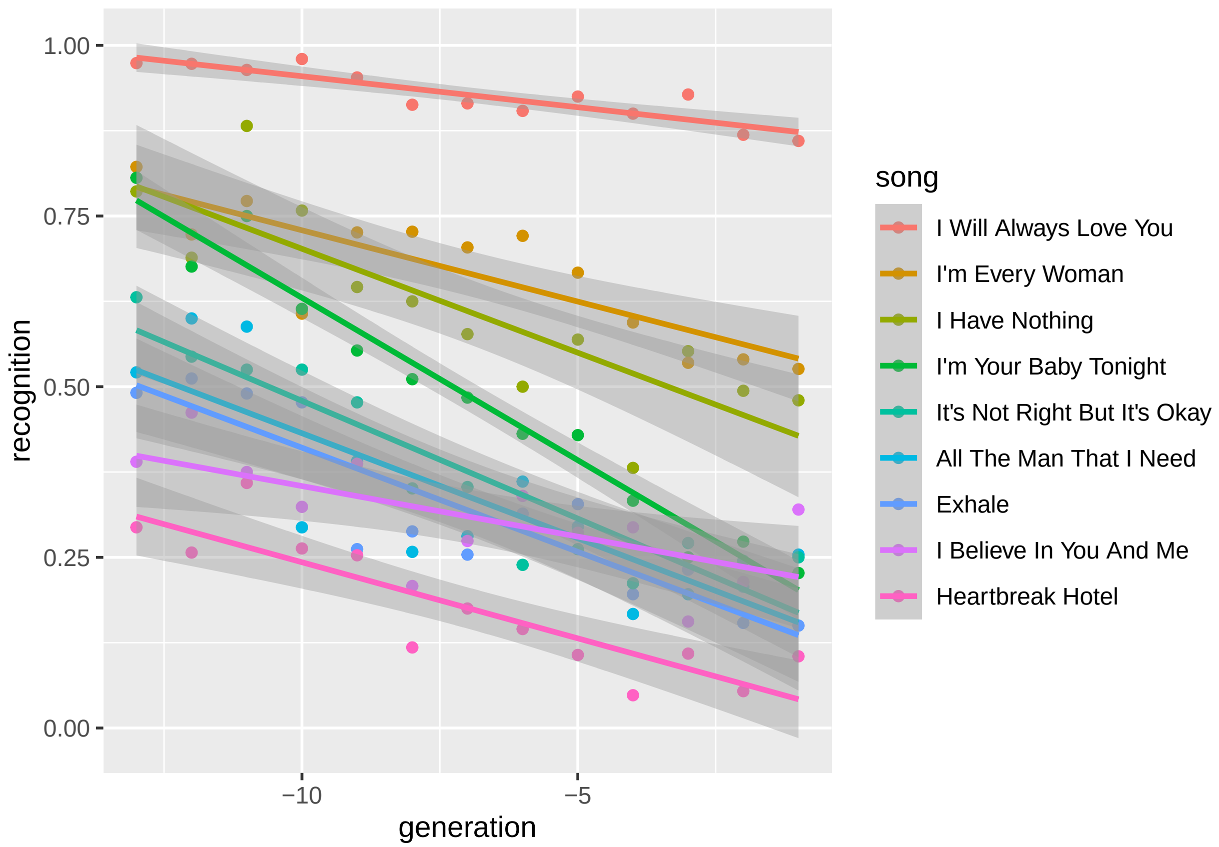


6. In the following exercise let’s focus on songs by Whitney Houston.

    1. Identify the most and least recognized songs by Whitney Houston by generating the following plot



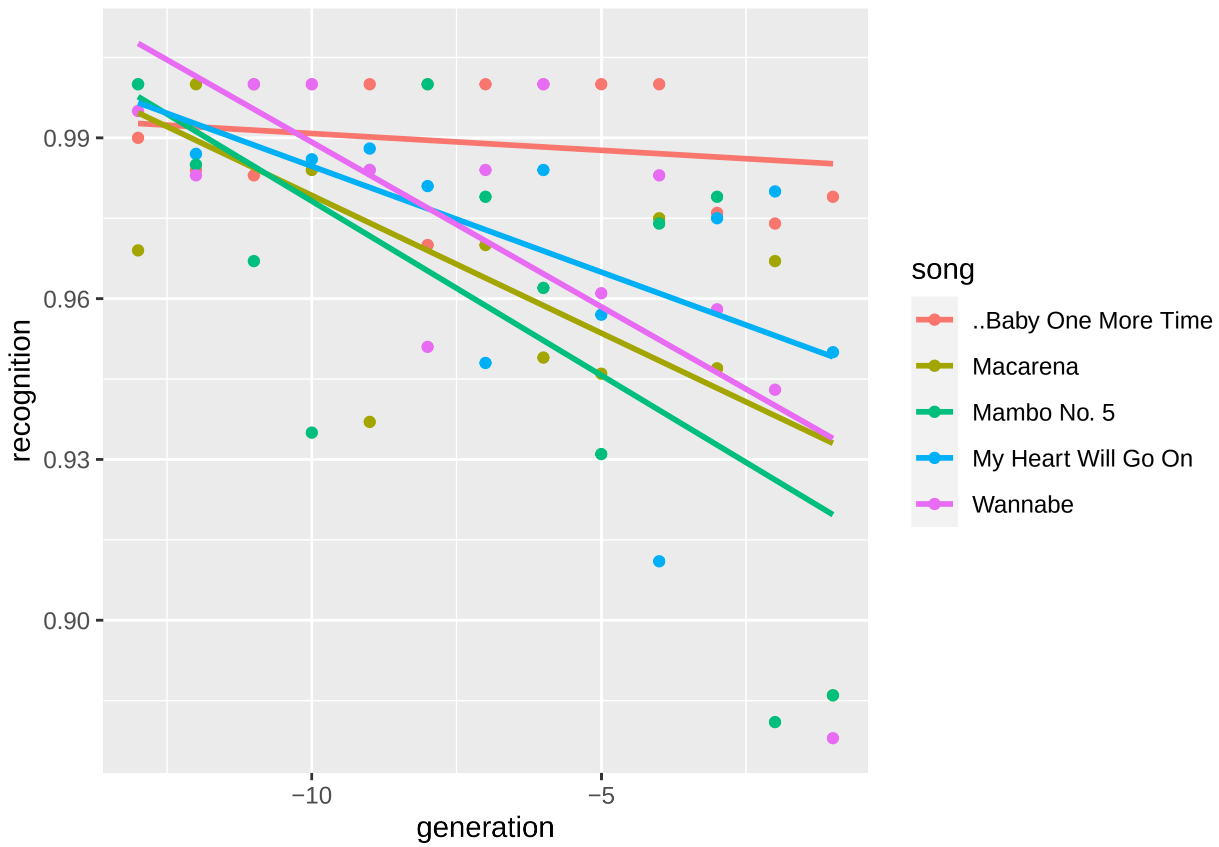
```
2. Modify the previous plot so that the names are in order of recognition (*Hint*: Create a new categori
```



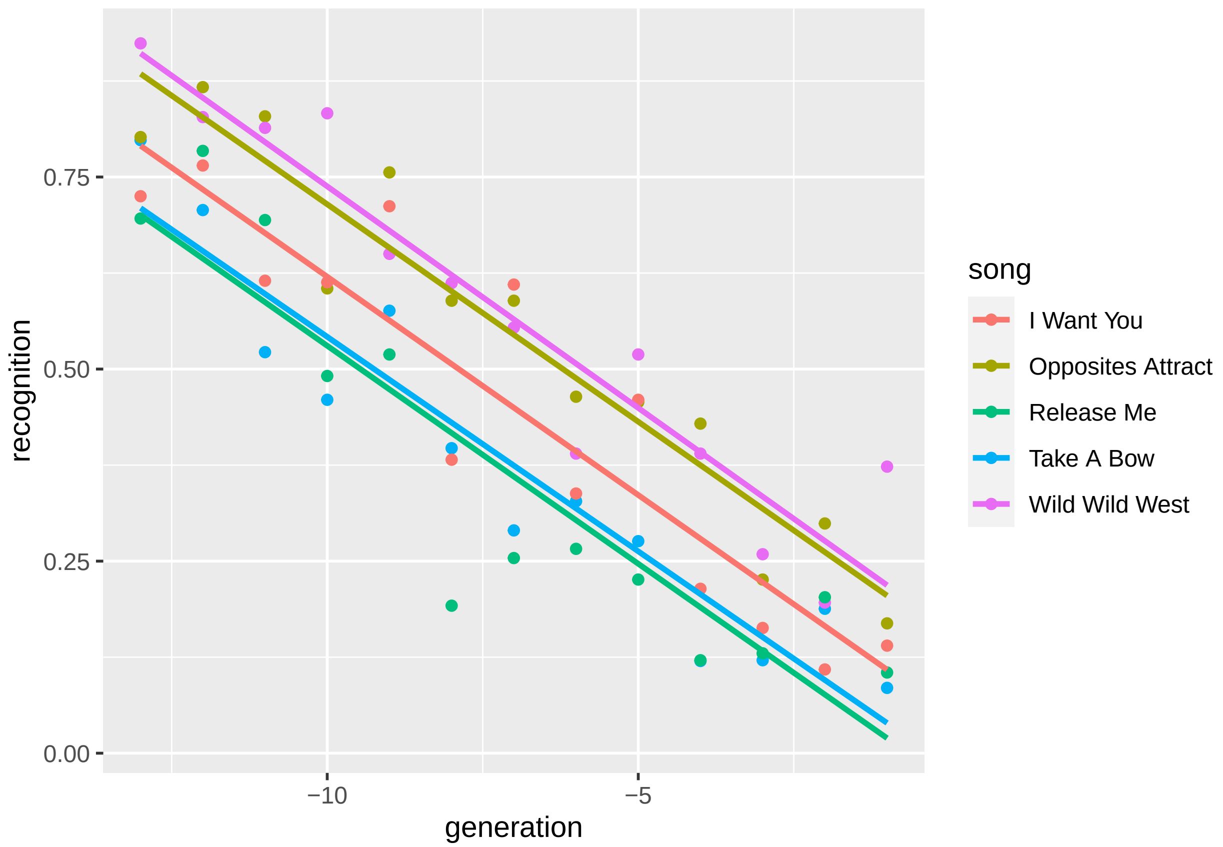
7. In this exercise we will plot the trends of particular sets of songs. For the purposes of the exercise we will subset `song.decay.tbl` to songs before generation 0

```
song.decay.filter.tbl <- song.decay.tbl %>%
  filter(generation<0)
```

```
1. Identify the top-5 songs with the highest recognition and plot their trends. Make sure to use ‘inner
```



1. Identify the top-5 songs with the highest variability and plot their trends. Make sure to use ‘inner



## Using pivots

Make sure to read Chapter 12:Tidy data and keep in mind the following commands:

- `pivot_longer`
- `pivot_wider`
- `separate`

8. We are interested in calculating how people from different generation remember songs from the 90s. In particular we are interested in calculating the average song recognition across milennials (people born between 1980 to 1994) and generation Z (people born after 1994)

To do that create a new table using the following steps:

- Calculate the year a person was born based on the `year` and `generation` fields.
- Keep only entries of people who were born starting in 1980.
- People who were born between 1980 and 1994 will be milennials and the rest will be generation Z.
- Finally calculate the average recognition grouping by the generational group
- Name the resulting table `song.gen.tbl`

```
## # A tibble: 684 x 3
## # Groups:   song [342]
##    song                   generation.group avg.recognition
##    <chr>                  <chr>                      <dbl>
##  1 ..Baby One More Time   Milennial                  0.992
##  2 ..Baby One More Time   Z                          0.963
##  3 4 Seasons Of Loneliness Milennial                 0.139
##  4 4 Seasons Of Loneliness Z                         0.066
##  5 A Whole New World      Milennial                  0.868
##  6 A Whole New World      Z                          0.706
##  7 Achy Breaky Heart      Milennial                  0.838
##  8 Achy Breaky Heart      Z                          0.562
##  9 Adia                   Milennial                  0.387
## 10 Adia                   Z                          0.119
## # ... with 674 more rows
```

9. As we can see `song.gen.tbl` has one different row for each `generation.group`, however we would like to have just one row per song and have columns with the average recognition for each generation. Use a pivot function to obtain the following table and name it `song.pivot.tbl`

```
## # A tibble: 342 x 3
## # Groups:   song [342]
##    song                 Milennial     Z
##    <chr>                    <dbl> <dbl>
##  1 ..Baby One More Time     0.992 0.963
##  2 Wannabe                  0.982 0.904
##  3 Believe                  0.978 0.899
##  4 My Heart Will Go On      0.974 0.962
##  5 Mambo No. 5              0.971 0.928
##  6 Macarena                 0.970 0.862
##  7 Everybody                0.965 0.864
##  8 I Believe I Can Fly      0.958 0.843
##  9 All Star                 0.946 0.932
## 10 The Power                0.932 0.876
## # ... with 332 more rows
```

10. Consider one of the tables for the original publication `time.series.tbl`

```
url <- "https://raw.githubusercontent.com/the-pudding/song-decay-clean/master/src/assets/data/time_seri
time.series.tbl <- read_csv(url)
time.series.tbl
```

```
## # A tibble: 344 x 49
##    artist_song ‘-13‘ ‘-12‘ ‘-11‘ ‘-10‘  ‘-9‘  ‘-8‘  ‘-7‘  ‘-6‘  ‘-5‘  ‘-4‘  ‘-3‘
##    <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 2 Pac|||Ca~ 0.648 0.54  0.712 0.435 0.644 0.594 0.6   0.421 0.446 0.391 0.309
##  2 2Pac|||How~ 0.265 0.226 0.382 0.296 0.230 0.284 0.139 0.169 0.215 0.1   0.109
##  3 702|||Wher~ 0.631 0.614 0.660 0.415 0.532 0.375 0.415 0.276 0.304 0.408 0.213
##  4 Ace Of Bas~ 0.963 0.94  0.976 0.877 0.955 0.954 1     0.853 0.9   0.875 0.779
##  5 Ace Of Bas~ 0.899 0.909 0.827 0.8   0.757 0.773 0.667 0.516 0.557 0.629 0.625
##  6 Ace Of Bas~ 0.979 0.955 0.984 0.933 0.930 0.908 0.943 0.893 0.905 0.841 0.806
##  7 Adina Howa~ 0.415 0.28  0.302 0.297 0.2   0.169 0.179 0.190 0.238 0.172 0.25
##  8 Aerosmith|~ 0.959 0.953 0.966 0.987 0.870 0.931 0.862 0.8   0.848 0.875 0.818
##  9 Aerosmith|~ 0.820 0.846 0.757 0.833 0.683 0.590 0.571 0.549 0.735 0.585 0.478
## 10 Alanis Mor~ 0.970 0.979 0.986 0.95  0.926 0.937 0.973 0.84  0.919 0.778 0.766
## # ... with 334 more rows, and 37 more variables: -2 <dbl>, -1 <dbl>, 0 <dbl>,
## #   1 <dbl>, 2 <dbl>, 3 <dbl>, 4 <dbl>, 5 <dbl>, 6 <dbl>, 7 <dbl>, 8 <dbl>,
## #   9 <dbl>, 10 <dbl>, 11 <dbl>, 12 <dbl>, 13 <dbl>, 14 <dbl>, 15 <dbl>,
## #   16 <dbl>, 17 <dbl>, 18 <dbl>, 19 <dbl>, 20 <dbl>, 21 <dbl>, 22 <dbl>,
## #   23 <dbl>, 24 <dbl>, 25 <dbl>, 26 <dbl>, 27 <dbl>, 28 <dbl>, 29 <dbl>,
## #   30 <dbl>, 31 <dbl>, 32 <dbl>, 33 <dbl>, 34 <dbl>
```

Convert this table into a table with the following format (Note that generation is an integer and recognition is a double) and make sure to subset generation from `-13` to `10`

```
## # A tibble: 8,256 x 4
##    artist song            generation recognition
##    <chr>  <chr>                <int>       <dbl>
##  1 2 Pac  California Love        -13       0.648
##  2 2 Pac  California Love        -12       0.54
##  3 2 Pac  California Love        -11       0.712
##  4 2 Pac  California Love        -10       0.435
##  5 2 Pac  California Love         -9       0.644
##  6 2 Pac  California Love         -8       0.594
##  7 2 Pac  California Love         -7       0.6
##  8 2 Pac  California Love         -6       0.421
##  9 2 Pac  California Love         -5       0.446
## 10 2 Pac  California Love         -4       0.391
## # ... with 8,246 more rows
```