# Importing necessary libraries  
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.4.1

## Warning: package 'ggplot2' was built under R version 4.4.1

## Warning: package 'tidyr' was built under R version 4.4.1

## Warning: package 'readr' was built under R version 4.4.1

## Warning: package 'purrr' was built under R version 4.4.1

## Warning: package 'stringr' was built under R version 4.4.1

## Warning: package 'forcats' was built under R version 4.4.1

## Warning: package 'lubridate' was built under R version 4.4.1

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(ggplot2)  
library(lubridate)  
library(readr)

# Filepath of the dataset  
filepath <- "D:/Mini Project/DAT/spotify-2023.csv"  
  
# Load the spotify dataset  
spotify\_df <- read\_csv(filepath, locale = locale(encoding = "latin1"))

## Rows: 953 Columns: 24  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (5): track\_name, artist(s)\_name, streams, key, mode  
## dbl (17): artist\_count, released\_year, released\_month, released\_day, in\_spot...  
## num (2): in\_deezer\_playlists, in\_shazam\_charts  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# View the first 5 rows of the data  
head(spotify\_df)

## # A tibble: 6 × 24  
## track\_name `artist(s)\_name` artist\_count released\_year released\_month  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 Seven (feat. Latto… Latto, Jung Kook 2 2023 7  
## 2 LALA Myke Towers 1 2023 3  
## 3 vampire Olivia Rodrigo 1 2023 6  
## 4 Cruel Summer Taylor Swift 1 2019 8  
## 5 WHERE SHE GOES Bad Bunny 1 2023 5  
## 6 Sprinter Dave, Central C… 2 2023 6  
## # ℹ 19 more variables: released\_day <dbl>, in\_spotify\_playlists <dbl>,  
## # in\_spotify\_charts <dbl>, streams <chr>, in\_apple\_playlists <dbl>,  
## # in\_apple\_charts <dbl>, in\_deezer\_playlists <dbl>, in\_deezer\_charts <dbl>,  
## # in\_shazam\_charts <dbl>, bpm <dbl>, key <chr>, mode <chr>,  
## # `danceability\_%` <dbl>, `valence\_%` <dbl>, `energy\_%` <dbl>,  
## # `acousticness\_%` <dbl>, `instrumentalness\_%` <dbl>, `liveness\_%` <dbl>,  
## # `speechiness\_%` <dbl>

# Combine the columns into a single 'release\_date' column  
spotify\_df <- spotify\_df %>%  
 mutate(release\_date = make\_date(released\_year, released\_month, released\_day))

# Drop the original year, month, and day columns  
spotify\_df <- spotify\_df %>%  
 select(-released\_year, -released\_month, -released\_day)

# Drop unnecessary columns  
spotify\_df <- spotify\_df %>%  
 select(-bpm, -key, -mode)

# Check for duplicates in the 'track\_name' column and drop the row causing the issue  
spotify\_df <- spotify\_df %>%  
 filter(!duplicated(track\_name))

# Convert 'streams' column to numeric format  
spotify\_df <- spotify\_df %>%  
 mutate(streams = as.numeric(streams))

## Warning: There was 1 warning in `mutate()`.  
## ℹ In argument: `streams = as.numeric(streams)`.  
## Caused by warning:  
## ! NAs introduced by coercion

# Sort the dataset by 'streams' in descending order  
spotify\_df <- spotify\_df %>%  
 arrange(desc(streams))

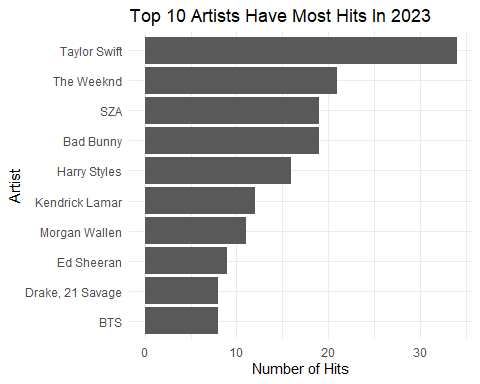
# Drop duplicates in 'track\_name', keeping the first occurrence (highest streams)  
spotify\_df <- spotify\_df %>%  
 distinct(track\_name, .keep\_all = TRUE)  
  
# Check for null values and drop unnecessary columns  
spotify\_df <- spotify\_df %>%  
 select(-in\_shazam\_charts)

# Remove commas and convert 'in\_deezer\_playlists' column to numeric  
spotify\_df <- spotify\_df %>%  
 mutate(in\_deezer\_playlists = as.numeric(gsub(",", "", in\_deezer\_playlists)))  
  
# Display the structure of the dataframe  
str(spotify\_df)

## tibble [943 × 18] (S3: tbl\_df/tbl/data.frame)  
## $ track\_name : chr [1:943] "Blinding Lights" "Shape of You" "Someone You Loved" "Dance Monkey" ...  
## $ artist(s)\_name : chr [1:943] "The Weeknd" "Ed Sheeran" "Lewis Capaldi" "Tones and I" ...  
## $ artist\_count : num [1:943] 1 1 1 1 2 3 2 1 2 2 ...  
## $ in\_spotify\_playlists: num [1:943] 43899 32181 17836 24529 24094 ...  
## $ in\_spotify\_charts : num [1:943] 69 10 53 0 78 24 36 23 0 79 ...  
## $ streams : num [1:943] 3.70e+09 3.56e+09 2.89e+09 2.86e+09 2.81e+09 ...  
## $ in\_apple\_playlists : num [1:943] 672 33 440 533 372 433 492 250 315 281 ...  
## $ in\_apple\_charts : num [1:943] 199 0 125 167 117 107 99 121 159 137 ...  
## $ in\_deezer\_playlists : num [1:943] 3421 6808 1800 3595 843 ...  
## $ in\_deezer\_charts : num [1:943] 20 7 0 6 4 0 31 10 0 1 ...  
## $ danceability\_% : num [1:943] 50 83 50 82 76 77 59 77 75 68 ...  
## $ valence\_% : num [1:943] 38 93 45 54 91 36 48 74 64 49 ...  
## $ energy\_% : num [1:943] 80 65 41 59 50 63 76 78 52 59 ...  
## $ acousticness\_% : num [1:943] 0 58 75 69 54 1 4 4 41 16 ...  
## $ instrumentalness\_% : num [1:943] 0 0 0 0 0 0 0 0 0 0 ...  
## $ liveness\_% : num [1:943] 9 9 11 18 7 36 10 23 11 13 ...  
## $ speechiness\_% : num [1:943] 7 8 3 10 5 5 5 11 3 28 ...  
## $ release\_date : Date[1:943], format: "2019-11-29" "2017-01-06" ...

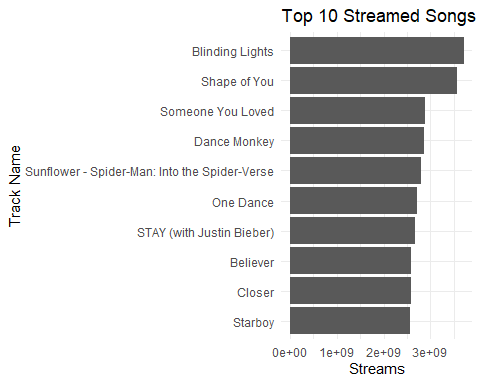
# Top 10 Artists Have Most Streamed Songs in 2023  
top\_artists <- spotify\_df %>%  
 count(`artist(s)\_name`, sort = TRUE) %>%  
 head(10)

# Plot the top 10 artists  
ggplot(top\_artists, aes(x = reorder(`artist(s)\_name`, n), y = n)) +  
 geom\_bar(stat = "identity") +  
 coord\_flip() +  
 labs(title = "Top 10 Artists Have Most Hits In 2023", x = "Artist", y = "Number of Hits") +  
 theme\_minimal()



# Top 10 Streamed Songs in 2023  
top\_10\_songs <- spotify\_df %>%  
 top\_n(10, streams)

# Plot the top 10 streamed songs  
ggplot(top\_10\_songs, aes(x = reorder(track\_name, streams), y = streams)) +  
 geom\_bar(stat = "identity") +  
 coord\_flip() +  
 labs(title = "Top 10 Streamed Songs in 2023", x = "Track Name", y = "Streams") +  
 theme\_minimal()

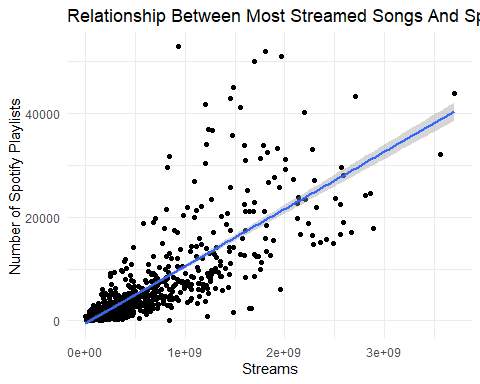


# Relationship Between Most Streamed Songs And Platforms In 2023  
# Spotify  
ggplot(spotify\_df, aes(x = streams, y = in\_spotify\_playlists)) +  
 geom\_point() +  
 geom\_smooth(method = "lm") +  
 labs(title = "Relationship Between Most Streamed Songs And Spotify Playlists In 2023",  
 x = "Streams", y = "Number of Spotify Playlists") +  
 theme\_minimal()

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: Removed 1 row containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 1 row containing missing values or values outside the scale range  
## (`geom\_point()`).

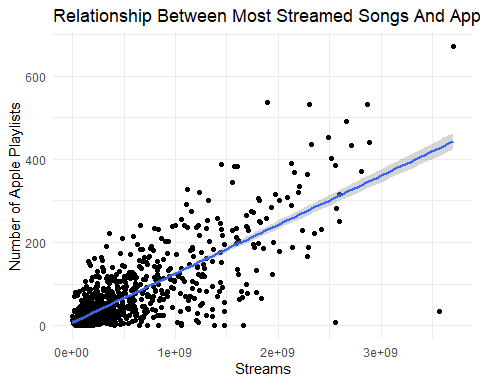


# Apple  
ggplot(spotify\_df, aes(x = streams, y = in\_apple\_playlists)) +  
 geom\_point() +  
 geom\_smooth(method = "lm") +  
 labs(title = "Relationship Between Most Streamed Songs And Apple Playlists In 2023",  
 x = "Streams", y = "Number of Apple Playlists") +  
 theme\_minimal()

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: Removed 1 row containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 1 row containing missing values or values outside the scale range  
## (`geom\_point()`).

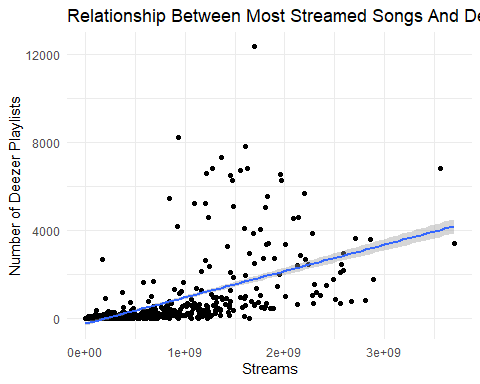


# Deezer  
ggplot(spotify\_df, aes(x = streams, y = in\_deezer\_playlists)) +  
 geom\_point() +  
 geom\_smooth(method = "lm") +  
 labs(title = "Relationship Between Most Streamed Songs And Deezer Playlists In 2023",  
 x = "Streams", y = "Number of Deezer Playlists") +  
 theme\_minimal()

## `geom\_smooth()` using formula = 'y ~ x'

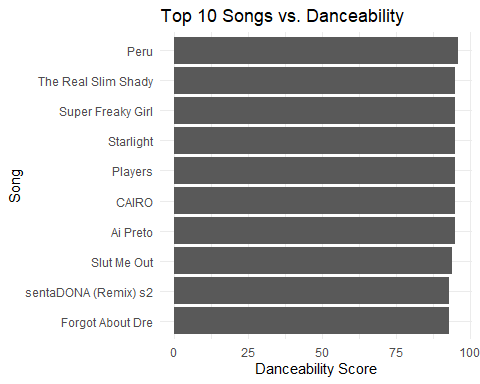
## Warning: Removed 1 row containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 1 row containing missing values or values outside the scale range  
## (`geom\_point()`).

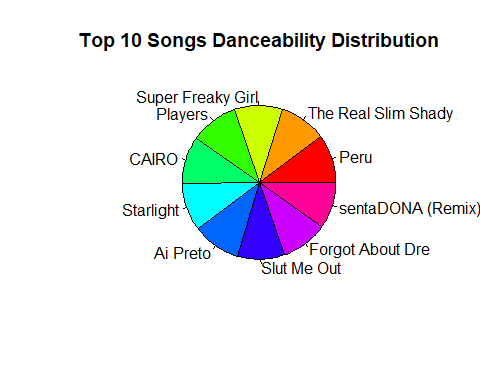


# Top 10 Songs vs. Danceability  
top\_10\_danceable\_songs <- spotify\_df %>%  
 arrange(desc(`danceability\_%`)) %>%  
 head(10)

# Horizontal bar chart for danceability  
ggplot(top\_10\_danceable\_songs, aes(x = `danceability\_%`, y = reorder(track\_name, `danceability\_%`))) +  
 geom\_bar(stat = "identity") +  
 labs(title = "Top 10 Songs vs. Danceability", x = "Danceability Score", y = "Song") +  
 theme\_minimal()



# Pie chart for danceability  
pie(top\_10\_danceable\_songs$`danceability\_%`, labels = top\_10\_danceable\_songs$track\_name, main = "Top 10 Songs Danceability Distribution", col = rainbow(10))



# Distribution of Songs in Playlists by Platform in 2023  
platform\_totals <- spotify\_df %>%  
 summarise(Spotify = sum(in\_spotify\_playlists, na.rm = TRUE),  
 Apple = sum(in\_apple\_playlists, na.rm = TRUE),  
 Deezer = sum(in\_deezer\_playlists, na.rm = TRUE))

# Pie chart for platform distribution  
pie(unlist(platform\_totals), labels = names(platform\_totals), main = "Distribution of Songs in Playlists by Platform in 2023", col = rainbow(3))

