Knowledge Quiz 2

Jessica Schmidt

Please answer the following questions, render a pdf, and submit via moodle by 11 PM on Fri Oct 24.

Guidelines:

- No consulting with anyone else (other than clarification questions from Professor Roback)
- You may use only materials from this class (our class webpage, links on moodle, our 3 online textbooks, files posted to the RStudio server)
- No online searches or use of large language models like ChatGPT

Pledge:

I pledge my honor that on this quiz I have neither given nor received assistance not explicitly approved by the professor and that I an aware of no dishonest work.

- type your name here to acknowledge the pledge: Jessica Schmidt
- OR
- place an X here if you intentionally are not signing the pledge: _____

```
library(tidyverse)
```

1. Here's is a crazy list that tells you some stuff about data science. Give code that will produce **exactly** the following outputs.

```
data_sci <- list(
  first = c("first it must work", "then it can be" , "pretty"),
  DRY = c("Do not", "Repeat", "Yourself"),
  dont_forget = c("garbage", "in", "out"),
  our_first_tibble = mpg,
  integers = 1:25,
  doubles = sqrt(1:25),
  tidyverse = c(pack1 = "ggplot2",
```

```
                pack2 = "dplyr",
                pack3 = "lubridate",
                etc = "and more!"),
  opinion = list("SDS 264 is",
                 c("awesome!", "amazing!", "rainbows!"))
)

str(data_sci)
```

```
List of 8
 $ first           : chr [1:3] "first it must work" "then it can be" "pretty"
 $ DRY             : chr [1:3] "Do not" "Repeat" "Yourself"
 $ dont_forget     : chr [1:3] "garbage" "in" "out"
 $ our_first_tibble: tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
  ..$ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
  ..$ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
  ..$ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
  ..$ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
  ..$ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
  ..$ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
  ..$ drv         : chr [1:234] "f" "f" "f" "f" ...
  ..$ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
  ..$ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
  ..$ fl          : chr [1:234] "p" "p" "p" "p" ...
  ..$ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
 $ integers        : int [1:25] 1 2 3 4 5 6 7 8 9 10 ...
 $ doubles         : num [1:25] 1 1.41 1.73 2 2.24 ...
 $ tidyverse       : Named chr [1:4] "ggplot2" "dplyr" "lubridate" "and more!"
  ..- attr(*, "names")= chr [1:4] "pack1" "pack2" "pack3" "etc"
 $ opinion         :List of 2
  ..$ : chr "SDS 264 is"
  ..$ : chr [1:3] "awesome!" "amazing!" "rainbows!"
```

[1] “Do not” “Yourself” “Repeat”

```
data_sci$DRY[c(1,3,2)]
```

```
[1] "Do not"   "Yourself" "Repeat"
```

A tibble: 10 × 2

trans mean_cty 1 auto(av) 20
2 auto(l3) 21
3 auto(l4) 15.9 4 auto(l5) 14.7 5 auto(l6) 13.7 6 auto(s4) 18.7 7 auto(s5) 17.3 8 auto(s6) 17.4
9 manual(m5) 19.3 10 manual(m6) 16.9

```
data_sci$our_first_tibble |>
  group_by(trans) |>
  summarize(mean_cty = mean(cty))
```

```
# A tibble: 10 x 2
   trans      mean_cty
   <chr>         <dbl>
 1 auto(av)       20
 2 auto(l3)       21
 3 auto(l4)       15.9
 4 auto(l5)       14.7
 5 auto(l6)       13.7
 6 auto(s4)       18.7
 7 auto(s5)       17.3
 8 auto(s6)       17.4
 9 manual(m5)     19.3
10 manual(m6)     16.9
```

[1] 3.162278 3.316625 3.464102 3.605551 3.741657 3.872983 4.000000 [8] 4.123106 4.242641 4.358899 4.472136 4.582576 4.690416 4.795832 [15] 4.898979 5.000000

```
data_sci$doubles[10:25]
```

```
 [1] 3.162278 3.316625 3.464102 3.605551 3.741657 3.872983 4.000000 4.123106
 [9] 4.242641 4.358899 4.472136 4.582576 4.690416 4.795832 4.898979 5.000000
```

[1] “SDS 264 is awesome!”

```
str_c(data_sci$opinion[1], " ", data_sci$opinion[[2]][1])
```

```
[1] "SDS 264 is awesome!"
```

2. Write a function called `popularity_factors()` that allows a user to input a tibble, an artist name, and a numeric variable in that tibble that could serve as a predictor of `track_popularity`. Your function should produce a scatterplot of the user's predictor on the x-axis, `track_popularity` on the y-axis, and a smoother through the points (where `se = FALSE` is the default unless changed by the user). Your plot title should reflect the artist chosen by the user, and if the chosen artist is not found in the data set, your function should return an error.

```
popularity_factors <- function(df, artist, x, se = FALSE){
  if (artist %in% df$track_artist) {
    label <- rlang::englue("Songs by {{artist}}")
    df |>
      filter(track_artist == artist) |>
      ggplot(aes(x = {{x}}, y = track_popularity)) +
        geom_point() +
        geom_smooth(se = se) +
        labs(title = label)
  } else {
    stop("Artist not listed in the data set", call. = FALSE)
  }
}
```

```
bigspotify <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/
```

```
Rows: 32833 Columns: 23
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (10): track_id, track_name, track_artist, track_album_id, track_album_na...
dbl (13): track_popularity, danceability, energy, key, loudness, mode, speec...

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.
```

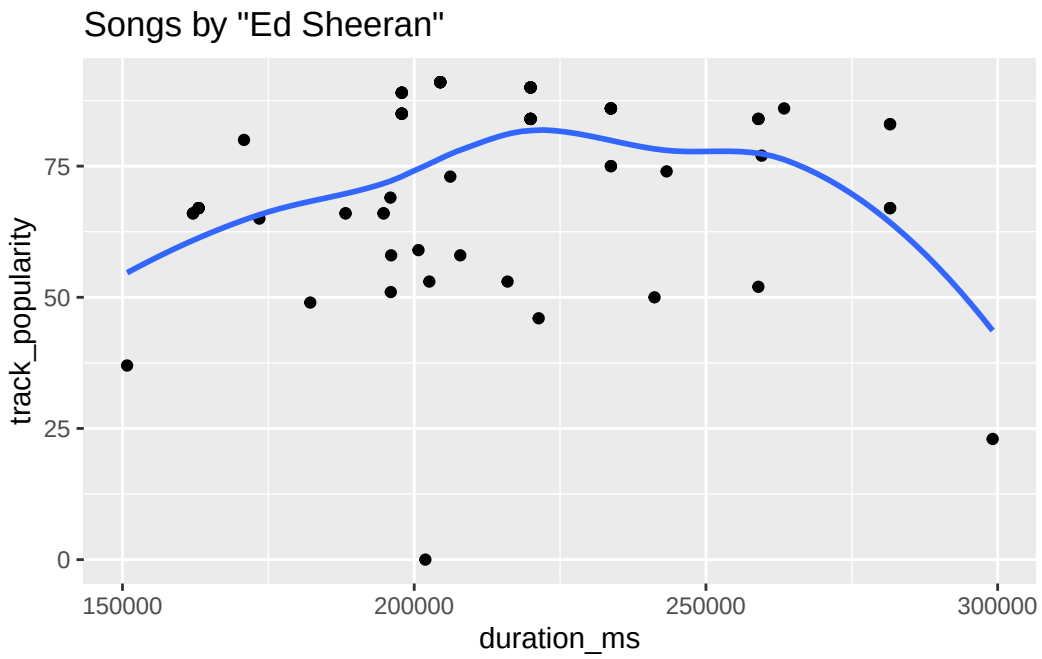
```
bigspotify
```

```
# A tibble: 32,833 x 23
   track_id              track_name track_artist track_popularity track_album_id
   <chr>                 <chr>      <chr>                  <dbl> <chr>
 1 6f807x0ima9a1j3VPbc7~ I Don't C~ Ed Sheeran                66 2oCs0DGTsRO98~
 2 0r7CVbZTWZgbTCYdfa2P~ Memories ~ Maroon 5                  67 63rPSO264uRjW~
 3 1z1Hg7Vb0AhHDiEmnDE7~ All the T~ Zara Larsson              70 1HoSmj2eLcsrR~
```

```
 4 75FpbthrwQmzHlBJLuGd~ Call You ~ The Chainsm~              60 1nqYsOef1yKKu~
 5 1e8PAfcKUYoKkxPhrHqw~ Someone Y~ Lewis Capal~              69 7m7vv9wlQ4i0L~
 6 7fvUMiyapMsRRxr07cU8~ Beautiful~ Ed Sheeran                67 2yiy9cd2QktrN~
 7 2OAylPUDDfwRGfe0lYql~ Never Rea~ Katy Perry                62 7INHYSeusaFly~
 8 6b1RNvAcJjQH73eZO4BL~ Post Malo~ Sam Feldt                 69 6703SRPsLkS4b~
 9 7bF6tCO3gFb8INrEDcjN~ Tough Lov~ Avicii                    68 7CvAfGvq4RlIw~
10 1IXGILkPm0tOCNeq00kC~ If I Can'~ Shawn Mendes             67 4QxzbfSsVryEQ~
# i 32,823 more rows
# i 18 more variables: track_album_name <chr>, track_album_release_date <chr>,
#   playlist_name <chr>, playlist_id <chr>, playlist_genre <chr>,
#   playlist_subgenre <chr>, danceability <dbl>, energy <dbl>, key <dbl>,
#   loudness <dbl>, mode <dbl>, speechiness <dbl>, acousticness <dbl>,
#   instrumentalness <dbl>, liveness <dbl>, valence <dbl>, tempo <dbl>,
#   duration_ms <dbl>
```

```
popularity_factors(df = bigspotify,
                   artist = "Ed Sheeran",
                   x = duration_ms)
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



```
# Works!  See KQ2_plot.png

popularity_factors(df = bigspotify,
                   artist = "Edward Sheeran",
                   x = duration_ms)
```

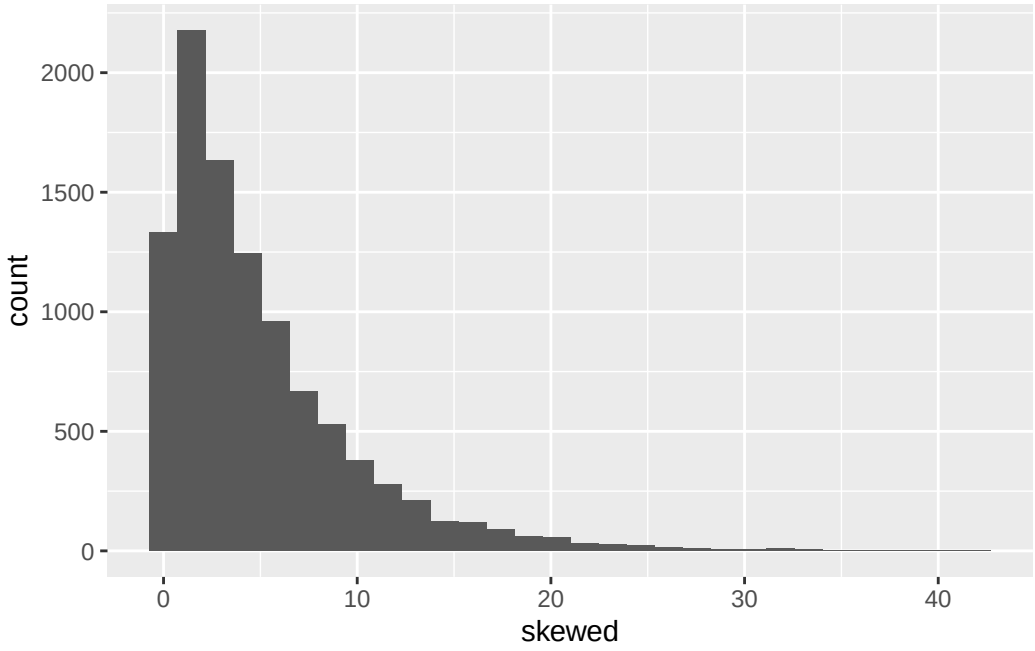
```
Error: Artist not listed in the data set
```

```
# Throws an error that says "Artist not listed in the data set"
```

3. The Central Limit Theorem is one of the most amazing results in all of mathematics. It says that if you take random samples from any population, if the sample size is large enough, the sample means will follow a normal distribution. This is true no matter how not-normal the original population is - crazy but true!! Let's explore the CLT in two steps.

```
# Histogram of skewed population we're sampling from
skewed <- rexp(10000, rate = 0.2)
tibble(skewed = skewed) |>
  ggplot(aes(x = skewed)) +
  geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```

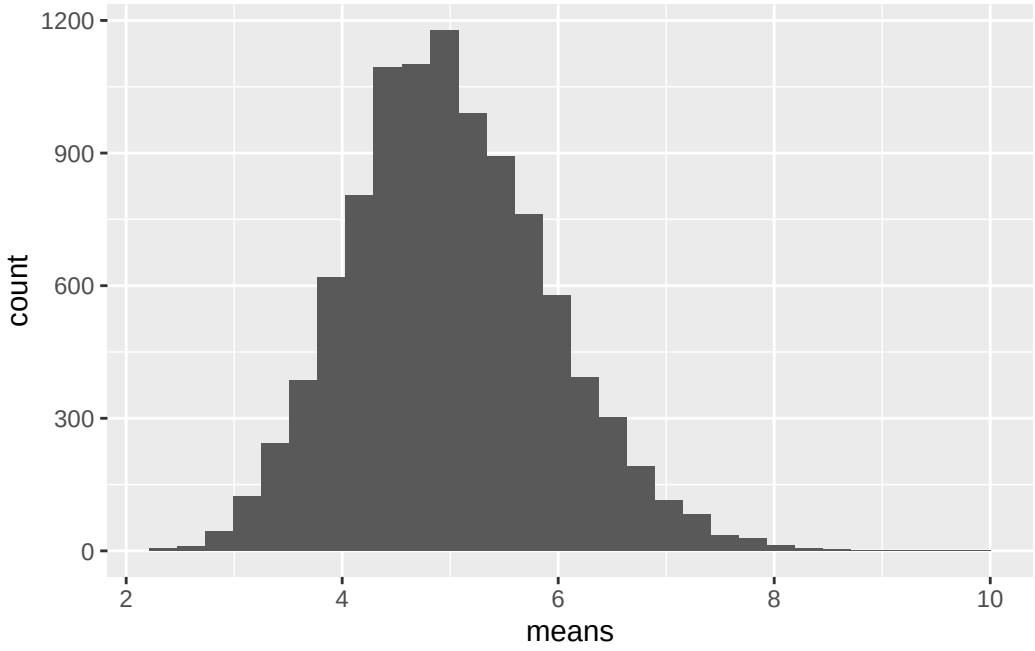


a) Write a for loop that takes 10,000 samples of size 30 from a skewed distribution and then plots the 10,000 means in a histogram to let us see if the histogram follows a normal distribution. Here are a couple of hints:

- `rexp(30, rate = 0.2)` will produce a random sample of size 30 from a skewed distribution
- `tibble(x = x)` will take a vector `x` and turn it into a column of a tibble that can be used in `ggplot`

```
means = vector("double", 10000)
for (i in 1:10000) {
  sample = rexp(30, rate = 0.2)
  means[i] = mean(sample)
}
ggplot(tibble(means = means), aes(x = means)) +
  geom_histogram()
```

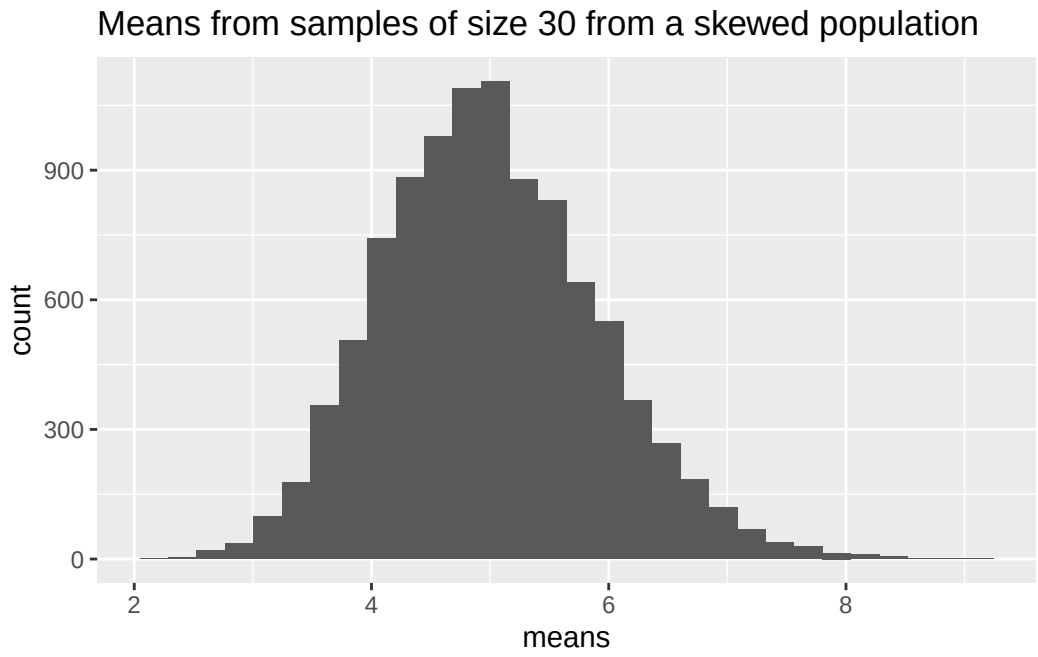
```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



b) Turn your for loop from (a) into a function whose attributes are `samp_size` with default of 30, and `n_means` with default of 10000. In addition, your histogram should now have a title that says "Means from samples of size 30 from a skewed population", where 30 is replaced with the user's input.

```
clt <- function(samp_size = 30, n_means = 10000){
  means = vector("double", n_means)
  for (i in 1:n_means) {
    sample = rexp(samp_size, rate = 0.2)
    means[i] = mean(sample)
  }
  label <- rlang::englue("Means from samples of size {{samp_size}} from a skewed population")
  ggplot(tibble(means), aes(x = means)) +
      geom_histogram() +
      labs(title = label)
}
clt()
```

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```




```
clt(samp_size = 40)
```

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```