Module 4 problem set

INSERT YOUR NAME HERE

INSERT DATE

Step 1: Make changes to YAML header

Read XAG section 3.3 before answering these questions

1. Add a table of contents to YAML header
2. table of contents should have "depth" of 2
3. Change "data frame printing" option to "tibble"

Step 2: Load packages, load data, and rename variables

1. Load the tidyverse package

```
#install.packages("tidyverse") #install if you do not have tidyverse installed
library(tidyverse)
#> -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
#> v dplyr     1.1.4     v readr     2.1.5
#> v forcats   1.0.0     v stringr   1.5.1
#> v ggplot2   3.5.1     v tibble    3.2.1
#> v lubridate 1.9.4     v tidyr     1.3.1
#> v purrr     1.0.2
#> -- Conflicts ------------------------------------------ tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
#> i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

2. Load the data frame data frame `df_school_all`
   - The URL for this data frame is: (https://github.com/ksalazar3/HED696C_RClass/raw/master/data/recruiting/recruit_school_allvars.RData)
   - The data frame `df_school_all` has one observation for each high school (public and private).
   - The variables that begin with `visits_by_...` identify how many off-campus recruiting visits the high school received from a particular public university. For example, UC Berkeley has the ID `110635` so the variable `visits_by_110635` identifies how many visits the high school received from UC Berkeley.
   - The variable `total_visits` identifies the number of visits the high school received from all (16) public research universities in this data collection sample.

```
load(url("https://github.com/ksalazar3/HED696C_RClass/raw/master/data/recruiting/recruit_school_allvars
```

3. Run the following code which drops some variables, renames other variables, and assigns these changes to the existing object `df_school_all` and then print the names of all the variables using the `names()` function.

```
df_school_all <- df_school_all %>%
  select(-contains("inst_")) %>% # remove vars that start with "inst_"
```

```
  rename(
    visits_by_berkeley = visits_by_110635,
    visits_by_boulder = visits_by_126614,
    visits_by_bama = visits_by_100751,
    visits_by_stonybrook = visits_by_196097,
    visits_by_rutgers = visits_by_186380,
    visits_by_pitt = visits_by_215293,
    visits_by_cinci = visits_by_201885,
    visits_by_nebraska = visits_by_181464,
    visits_by_georgia = visits_by_139959,
    visits_by_scarolina = visits_by_218663,
    visits_by_ncstate = visits_by_199193,
    visits_by_irvine = visits_by_110653,
    visits_by_kansas = visits_by_155317,
    visits_by_arkansas = visits_by_106397,
    visits_by_sillinois = visits_by_149222,
    visits_by_umass = visits_by_166629,
    num_took_read = num_took_rla,
    num_prof_read = num_prof_rla,
    med_inc = avgmedian_inc_2564
  )

names(df_school_all)
#>  [1] "state_code"           "school_type"          "ncessch"
#>  [4] "name"                 "address"              "city"
#>  [7] "zip_code"             "pct_white"            "pct_black"
#> [10] "pct_hispanic"         "pct_asian"            "pct_amerindian"
#> [13] "pct_other"            "num_fr_lunch"         "total_students"
#> [16] "num_took_math"        "num_prof_math"        "num_took_read"
#> [19] "num_prof_read"        "med_inc"              "latitude"
#> [22] "longitude"            "visits_by_stonybrook" "visits_by_rutgers"
#> [25] "visits_by_pitt"       "visits_by_cinci"      "visits_by_nebraska"
#> [28] "visits_by_georgia"    "visits_by_scarolina"  "visits_by_bama"
#> [31] "visits_by_ncstate"    "visits_by_berkeley"   "visits_by_irvine"
#> [34] "visits_by_boulder"    "visits_by_kansas"     "visits_by_arkansas"
#> [37] "visits_by_sillinois"  "visits_by_umass"      "total_visits"
```

Step 3: creating variables using mutate() and `if_else()`

The focus of this set of questions will be practicing creating some variables from the data frame `df_school_all`. You will be using the `mutate()` function, **sometimes** combined with the `if_else()` function. Additionally, be sure to investigate the values of "input" variables before creating new "analysis" variables.

Before presenting questions, here are some examples of code that may be useful in checking variable values. The below lines of code count:

- the number of observations in the data frame `df_school_all`
- the number of observations that have missing values for the variable `state_code`
- the number of observations that have missing values for the variable `school_type`
- a frequency count of the variable `school_type`

```
df_school_all %>% count()
#> # A tibble: 1 x 1
#>       n
```

```
#>   <int>
#> 1 21301
count(df_school_all) # same as above
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1 21301
df_school_all %>% filter(is.na(state_code)) %>% count() # number with NA for state_code
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
df_school_all %>% filter(is.na(school_type)) %>% count() # number with NA for school_type
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
df_school_all %>% count(school_type) # frequency count of school_type
#> # A tibble: 2 x 2
#>   school_type     n
#>   <chr>       <int>
#> 1 private      3822
#> 2 public      17479
```

1. Using `mutate()` with `ifelse()` create a 0/1 indicator called `ca_school` that indicates whether the high school is in California and then use `count()` to create a frequency table for the values of `ca_school` (you don't need to assign/retain the new variable)
2. Using `mutate()` with `ifelse()` create a 0/1 indicator called `ca_pub_school` that indicates whether the school is a public high school in California and then use `count()` to create a frequency table for the values of `ca_pub_school` (you don't need to assign/retain the new variable)
3. By combining the `is.na()` function with the `filter()` function, identify the number of observations that have missing values for the following variables:
    - `pct_black`, `pct_hispanic`, `pct_amerindian`
4. Create a new variable `pct_bl_hisp_ai` that represents the percent of students at the school that identify as black, hispanic, or american indian (hint: just sum all the pct vars for each race/ethnicity). Retain this variable by assigning it to the object `df_school_all`
5. Using `mutate()` with `ifelse()`, create a new 0/1 indicator variable `gt50pct_bl_hisp_ai` that identifies whether more than 50% of students identify as black, hispanic, or american indian and create a frequency count of this variable (no need to retain this variable)
6. Using `mutate()` with `ifelse()`, create the following 0/1 indicator variables, retain them (assign to object `df_school_all`), and then create frequency counts of these variables:
    - Variable `miss_took_math` for whether the school has missing values for the variable `num_took_math`
    - Variable `miss_prof_math` for whether the school has missing values for the variable `num_prof_math`
    - Variable `miss_took_or_prof_math` for whether the school has missing values for the variable `num_took_math` OR `num_prof_math`

Step 4: creating variables using `mutate()` + `case_when()`

For this set of questions, you will work with the data frame `wwlist` which has one observation for each prospective student purchased by Western Washington University from the College Board.

The objective of this set of questions is to create a three-category variable that identifies whether the prospect lives: - (1) in-state (i.e., in Washington), (2) out-of-state but in a US state/territory; (3) not in the US

1. Load the data frame `wwlist` which has information on prospects purchased by Western Washington University

```
load(url("https://github.com/ksalazar3/HED696C_RClass/raw/master/data/prospect_list/wwlist_merged.RData
```

2. Apply the `str()` function to the variables `state` and `for_country`; and using the `count()` function to create frequency tables for the variables `state`
    - `state`
    - `for_country`
3. Using the `filter()` function and `is.na()` function do the following:
    - count how many missing observations (`NAs`) the variable `state` has
    - count how many missing observations the variable `for_country` has
4. Create a frequency count for the variable `for_country` for the observations where `state` equals `NA` (hint: use the `is.na()`) function
5. Create a frequency count for the variable `for_country` for the observations where `state` does not equal `NA` (hint: use `!is.na()`) function
6. Count the number of observations that have the value "No Response" for the variable `for_country`
7. Using the `case_when` function within `mutate()` create a character variable called `residency` that has the following values: "in_state"; "out_state_us"; "not_in_us"

- This variable should have the value `NA` for observations where `for_country=="No Response"`
- Retain this variable (assign to object `wwlist`) and create a frequency count of this variable

THIS IS A BONUS QUESTION

BONUS QUESTION: Complete questions 4 and 5 in Step 3 using base R syntax (without the tidyverse `mutate()` function). To create frequency counts, you can still use the `count()` function **after** you've created the new variables via base R (see slides 62-67 on Module 4 presentation)

Once finished, knit to (pdf) and upload both .Rmd and PDF files