Module 5 problem set

INSERT YOUR NAME HERE

INSERT DATE HERE

Contents

Purpose

Data you will be working with

In this problem set, we are working with data from the the list of prospective students that Western Washington University purchased from College Board. We have also merged in U.S. Census Bureau data on socioeconomic/racial characteristics and National Center for Education Statistics data on school characteristics to the prospect-level data from College Board. Hence, the dataset you will be working with has one observation per prospective student. Some variables are prospet-level variables (e.g., `ethn_code` is a measure of race/ethnicity that varies by prospect). Other variables are measured at the zip-code level or state-level. These are measures of the racial composition for the zip code the prospect lives in and measures of the racial composition for the state in which the prospect lives; they do not vary across prospects within the same zip-code or state.

Task

For this problem set, you are a researcher and your goal is to identify systematic racial and socioeconomic bias in student list purchases by Western Washington University. That is, do the prospects purchased by Western Washington tend to have different racial and socioeconomic characteristics than other people in their state or zip-code?

Note that there is a lot of data cleaning required before conducting `group_by` and `summarise()` analyses. Much of this data cleaning involves creating prospect-level and zipcode/state-level measures of race/ethnicity that are consistent to one another. Therefore, I have answered some of the data cleaning questions for you to avoid making the problem set too long. I intentionally left data cleaning code for you all to get a sense of the process of investigating and cleaning your data.

Caveat

Merging data from other sources (e.g. College Board & Census) gives us breadth in investigating racial and socioeconomic bias beyond the prospect (student) level, yet at the same time, we are limited in the choices we make for disaggregating by race and ethnicity (in addition to other variables). Further, there are some fundamental differences between how College Board and Census define race/ethnicity that cannot be overcome with data cleaning. Therefore, we have to make some assumptions about comparisons between race/ethncity variables from College Board and race/ethnicity variables from Census.

Definitions for race and ethnicity used by Census and College Board

Here is some background information on how U.S. Census and College Board define race and etncity:

- U.S. Census
  - Census definitions of race and ethnicity LINK HERE
  - Census categories of race and ethnicity LINK HERE
- College Board
  - College Board Categories of race and ethnicity LINK HERE; SEE TABLE 1 on Page 4
  - College Board race and ethnicity questions from SAT Questionnaire LINK HERE; SEE FIGURE A on PAGE 3

Idiosyncrasies about the way race/ethnicity is defined by College Board vs. U.S. Census in the dataset you will be working with

- The College Board survey asks a question about "ethnicity" and then a separate question about "race"; However, the data sent to us by Western Washington combined race and ethnicity into one variable called `ethn_code`
- The College Board survey questions for ethnicity and race uses the following rules:
  - "Students may select all options that apply. In prior years, they were asked to select one option."
- By contrast, US Census data asks respondents to select one option; there is a separate option for "Two or More Races"

- As a result of these differences, the College Board race/ethnicity variable has a much higher percentage of people who identify as "2 or more races" than data from U.S. Census

Load library and data

```
library(tidyverse)
#> -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
#> v ggplot2 3.3.6     v purrr   0.3.4
#> v tibble  3.1.8     v dplyr   1.0.9
#> v tidyr   1.2.0     v stringr 1.4.0
#> v readr   2.1.2     v forcats 0.5.1
#> -- Conflicts ------------------------------------------ tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

```
rm(list = ls()) # remove all objects

load(url("https://github.com/ksalazar3/HED696C_RClass/raw/master/data/prospect_list/wwlist_merged.RData
#getwd()
#load("../../../documents/rclass/data/prospect_list/wwlist_merged.RData")
```

Cleaning the data before creating summary measures using `group_by()` and `summarise()`

In general, for all questions that ask you to drop certain observations or create new variables, assign these changes to the existing object `wwlist`

Part I: Questions related to keeping/dropping specfic observations

Question 1

- Do the following:
  - Count the number of observations that have `NA` for the variable `state`
  - Using `filter()` drop all observations that have `NA` for the variable `state` (i.e., keep observations that are not `NA` for variable `state`)
  - Using `mutate()` and `if_else()`, create a [and retain] 0/1 variable `in_state` that equals 1 if `state` equals Washington and equals `0` otherwise
  - Investigate the values of the new variable `in_state`, including confirming that this variable has no missing values

```
#check variable names
names(wwlist)
#>  [1] "receive_date"           "psat_range"             "state"
#>  [4] "zip9"                   "for_country"            "sex"
#>  [7] "hs_ceeb_code"           "hs_name"                "hs_city"
#> [10] "hs_state"               "hs_grad_date"           "ethn_code"
#> [13] "homeschool"             "firstgen"               "zip5"
#> [16] "pop_total_zip"          "pop_white_zip"          "pop_black_zip"
#> [19] "pop_asian_zip"          "pop_latinx_zip"         "pop_nativeam_zip"
#> [22] "pop_nativehawaii_zip"   "pop_multirace_zip"      "pop_otherrace_zip"
#> [25] "med_inc_zip"            "school_type"            "merged_hs"
#> [28] "school_category"        "total_12"               "total_students"
```

```
#> [31] "fr_lunch"               "pop_total_state"        "pop_white_state"
#> [34] "pop_black_state"        "pop_nativeam_state"     "pop_asian_state"
#> [37] "pop_nativehawaii_state" "pop_otherrace_state"    "pop_multirace_state"
#> [40] "pop_latinx_state"       "med_inc_state"

#count number of obs w/ missing values for state
wwlist %>% filter(is.na(state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1    85

#drop observations for missing values for state
wwlist <- wwlist %>% filter(!is.na(state))

#Create [and retain] new variable in_state

wwlist <- wwlist %>% mutate(in_state = if_else(state=="WA",1,0))

#Investigate values of in_state
str(wwlist$in_state)
#>  num [1:268311] 1 1 1 1 1 1 1 1 1 0 ...
wwlist %>% count(in_state)
#> # A tibble: 2 x 2
#>   in_state      n
#>      <dbl>  <int>
#> 1        0 172289
#> 2        1  96022
wwlist %>% filter(is.na(in_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
```

Question 2

- Do the following:
  - Count the number of observations where the value of `pop_total_zip` equals `0`
  - Count the number of observations where the value of `pop_total_zip` equals `NA`
  - Drop observations where the value of `pop_total_zip` is equal to `0`
    - NOTE: we won't drop observations where value of `pop_total_zip` equals `NA`

NOTE: IN THIS QUESTION, I GIVE YOU THE ANSWERS; ALL YOU HAVE TO DO IS RUN THE BELOW CODE CHUNK

```
wwlist %>% filter(pop_total_zip ==0) %>% count() # number of obs that equal 0
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1    23
wwlist %>% filter(is.na(pop_total_zip)) %>% count() # number of obs that equal NA
#> # A tibble: 1 x 1
#>       n
#>   <int>
```

```
#> 1  1576

wwlist %>% filter(pop_total_zip != 0 | is.na(pop_total_zip)) %>%
  count() # number of obs where pop_total zip is either not equal to 0 or is equal to NA
#> # A tibble: 1 x 1
#>        n
#>    <int>
#> 1 268288

wwlist <- wwlist %>%
  filter(pop_total_zip != 0 | is.na(pop_total_zip)) # keep obs where pop_total_zip is not equal to 0 or
```

Question 3

- Remove observations the have the following values for the variable **state**: "AP", "MP"
  - these values either refer to territories or are errors

NOTE: IN THIS QUESTION, I GIVE YOU THE ANSWERS; ALL YOU HAVE TO DO IS RUN THE BELOW CODE CHUNK

```
wwlist %>% filter(state %in% c("AP","MP")) %>% count() # equal to AP or MP
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     2
wwlist %>% filter(!state %in% c("AP","MP")) %>% count() # not equal to AP or MP
#> # A tibble: 1 x 1
#>        n
#>    <int>
#> 1 268286

wwlist <- wwlist %>% filter(!state %in% c("AP","MP")) # not equal to AP or MP
wwlist %>% count(state)
#> # A tibble: 51 x 2
#>    state     n
#>    <chr> <int>
#>  1 AK     3671
#>  2 AL      136
#>  3 AR       78
#>  4 AZ    10358
#>  5 CA    62382
#>  6 CO    24822
#>  7 CT      173
#>  8 DC       35
#>  9 DE       37
#> 10 FL     1287
#> # ... with 41 more rows
#> # i Use `print(n = ...)` to see more rows
```

Part II: Questions related to creating new variables prior to creating summary measures using group_by() and summarise()

This set of questions primarily relates to creating prospect-level measures of race/ethnicity (data from College Board) that are consistent with zip-code-level and state-level measures of race/ethnicity (data from US Census)

Question 1

- Investigate the prospect-level race/ethnicity variable `ethn_code` as follows:
  - what "type" of variable is it
  - create a frequency table
  - count the number of `NA` values

NOTE: IN THIS QUESTION, I GIVE YOU THE ANSWERS; ALL YOU HAVE TO DO IS RUN THE BELOW CODE CHUNK

```
str(wwlist$ethn_code)
#>  chr [1:268286] "other-2 or more" "white" "white" "other-2 or more" "white" ...
wwlist %>% count(ethn_code)
#> # A tibble: 10 x 2
#>    ethn_code                                          n
#>    <chr>                                          <int>
#>  1 american indian or alaska native                 202
#>  2 asian or native hawaiian or other pacific islander  2385
#>  3 black or african american                        563
#>  4 cuban                                             70
#>  5 mexican/mexican american                       6548
#>  6 not reported                                   5736
#>  7 other spanish/hispanic                         2429
#>  8 other-2 or more                               90543
#>  9 puerto rican                                    195
#> 10 white                                        159615
wwlist %>% filter(is.na(ethn_code)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
```

Question 2

- The prospect-level variable `ethn_code` combines Asian, Native Hawaiian and Pacific Islander into one category. For zip code population level race/ethnicity variables to be consistent with the prospect-level variable `ethn_code`, create a variable `pop_api_zip` equal to the sum of `pop_asian_zip` and `pop_nativehawaii_zip`. Follow these steps:
  - check how many missing values the "input variables" `pop_asian_zip` and `pop_nativehawaii_zip` have
  - create the new variable
  - check the value of the new variable for observations that had missing values in the input variables
  - delete the input variables

NOTE: IN THIS QUESTION, I GIVE YOU THE ANSWERS; ALL YOU HAVE TO DO IS RUN THE BELOW CODE CHUNK

```
#investigate input variables [zip-code level race/ethnicity vars]
```

```
wwlist %>% filter(is.na(pop_asian_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_nativehawaii_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574

#create variable
wwlist <- wwlist %>% mutate(
    pop_api_zip = pop_asian_zip + pop_nativehawaii_zip
  )

#check value of new variable; and check the value of the new variable against value of input variables
wwlist %>% filter(is.na(pop_api_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_asian_zip)) %>% count(pop_api_zip)
#> # A tibble: 1 x 2
#>   pop_api_zip     n
#>         <int> <int>
#> 1          NA  1574
wwlist %>% filter(is.na(pop_nativehawaii_zip)) %>% count(pop_api_zip)
#> # A tibble: 1 x 2
#>   pop_api_zip     n
#>         <int> <int>
#> 1          NA  1574

#remove input variables
wwlist <- wwlist %>% select(-pop_asian_zip,-pop_nativehawaii_zip)

#names(wwlist)
```

Question 3

- Follow the same steps as above to create a state-level Asian-Pacific Islander variable `pop_api_state` from the input variables `pop_asian_state` and `pop_nativehawaii_state`

```
#investigate input variables
wwlist %>% filter(is.na(pop_asian_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_nativehawaii_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
```

```
#create variable
wwlist <- wwlist %>% mutate(
    pop_api_state= pop_asian_state + pop_nativehawaii_state
  )

#check value of new variable against value of input variable
wwlist %>% filter(is.na(pop_api_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_asian_state)) %>% count(pop_api_state)
#> # A tibble: 0 x 2
#> # ... with 2 variables: pop_api_state <int>, n <int>
#> # i Use `colnames()` to see all variable names
wwlist %>% filter(is.na(pop_nativehawaii_state)) %>% count(pop_api_state)
#> # A tibble: 0 x 2
#> # ... with 2 variables: pop_api_state <int>, n <int>
#> # i Use `colnames()` to see all variable names

#remove input variables
wwlist <- wwlist %>% select(-pop_asian_state,-pop_nativehawaii_state)
```

Question 4

- Next, we'll use the zip-code level measures of number of people by race/ethnicity to create zip-code level measures of **percent** of people by race/ethnicity
  - Before creating the new variables, investigate presence of missing observations in input variables
  - after you create the variables, investigate the value of the new variables and their value against missing values of the input variables. Do this for two of the new race variables you created

NOTE: IN THIS QUESTION, I GIVE YOU THE ANSWERS; ALL YOU HAVE TO DO IS RUN THE BELOW CODE CHUNK

```
#show names of zip code level race vars
wwlist %>% select(ends_with("_zip"),-med_inc_zip) %>% names()
#> [1] "pop_total_zip"     "pop_white_zip"     "pop_black_zip"
#> [4] "pop_latinx_zip"    "pop_nativeam_zip"  "pop_multirace_zip"
#> [7] "pop_otherrace_zip" "pop_api_zip"

#Investigate presence of missing values in input variables
wwlist %>% filter(is.na(pop_total_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_white_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_black_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
```

```
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_latinx_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_nativeam_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_multirace_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_otherrace_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_api_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574

#create new variables
  #note: we multiply by 100 so that we have percentages rather than proportions, which are easier to re
wwlist <- wwlist %>%
  mutate(
    pct_white_zip= pop_white_zip/pop_total_zip*100,
    pct_black_zip= pop_black_zip/pop_total_zip*100,
    pct_latinx_zip= pop_latinx_zip/pop_total_zip*100,
    pct_nativeam_zip= pop_nativeam_zip/pop_total_zip*100,
    pct_multirace_zip= pop_multirace_zip/pop_total_zip*100,
    pct_otherrace_zip= pop_otherrace_zip/pop_total_zip*100,
    pct_api_zip= pop_api_zip/pop_total_zip*100,
  )

#Investigate values of new variables against values of input vars for two of the race categories

wwlist %>% summarise(pct_white_zip= mean(pct_white_zip, na.rm = TRUE)) # average percent white across a
#> # A tibble: 1 x 1
#>   pct_white_zip
#>           <dbl>
#> 1          68.0

wwlist %>% filter(is.na(pct_white_zip)) %>% count() # number missing
#> # A tibble: 1 x 1
#>       n
#>   <int>
```

```
#> 1  1574
wwlist %>% filter(is.na(pop_white_zip) | is.na(pop_total_zip)) %>%
  count(pct_white_zip) # count values of pct_white_zip if either of the input vars is missing
#> # A tibble: 1 x 2
#>   pct_white_zip     n
#>           <dbl> <int>
#> 1            NA  1574

wwlist %>% filter(is.na(pct_black_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1  1574
wwlist %>% filter(is.na(pop_black_zip) | is.na(pop_total_zip)) %>%
  count(pct_white_zip)
#> # A tibble: 1 x 2
#>   pct_white_zip     n
#>           <dbl> <int>
#> 1            NA  1574
```

Question 5

- Follow the same steps as above to create state-level measures of percent of people by race/ethnicity
  - after you create the variables, investigate the value of the new variables and their value against missing values of the input variables for two of the new race variables

```
#Investigate presence of missing values in input variables
wwlist %>% filter(is.na(pop_total_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_white_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_black_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_latinx_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_nativeam_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_multirace_state)) %>% count()
#> # A tibble: 1 x 1
```

```
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_otherrace_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_api_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0

#create new variables
wwlist <- wwlist %>%
  mutate(
    pct_white_state= pop_white_state/pop_total_state*100,
    pct_black_state= pop_black_state/pop_total_state*100,
    pct_latinx_state= pop_latinx_state/pop_total_state*100,
    pct_nativeam_state= pop_nativeam_state/pop_total_state*100,
    pct_multirace_state= pop_multirace_state/pop_total_state*100,
    pct_otherrace_state= pop_otherrace_state/pop_total_state*100,
    pct_api_state= pop_api_state/pop_total_state*100,
  )

#Investigate values of new variables against values of input vars for two of the race categories
wwlist %>% filter(is.na(pct_white_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_white_state) | is.na(pop_total_state)) %>%
  count(pct_white_state)
#> # A tibble: 0 x 2
#> # ... with 2 variables: pct_white_state <dbl>, n <int>
#> # i Use `colnames()` to see all variable names

wwlist %>% filter(is.na(pct_black_state)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(is.na(pop_black_state) | is.na(pop_total_state)) %>%
  count(pct_white_state)
#> # A tibble: 0 x 2
#> # ... with 2 variables: pct_white_state <dbl>, n <int>
#> # i Use `colnames()` to see all variable names
```

Question 6

- Next, we'll make a new version of the prospect level race/ethnicity variable that is consistent with the Census zip code level and state level race/ethnicity variables
  - First, investigate the input variable `ethn_code` including:

    - identifying variable "type"
    - creating a frequency table
    - counting the number of missing values
  - Second, Using the `recode()` function within `mutate()`, create a variable called `ethn_race` that recodes the input variable `ethn_code` as follows:
    - "american indian or alaska native" = "nativeam",
    - "asian or native hawaiian or other pacific islander" = "api",
    - "black or african american" = "black",
    - "cuban" = "latinx",
    - "mexican/mexican american" = "latinx",
    - "not reported" = "not_reported",
    - "other-2 or more" = "multirace",
    - "other spanish/hispanic" = "latinx",
    - "puerto rican" = "latinx",
    - "white" = "white",

  - Third, investigate the values of the new variable `ethn_race` including:
    - variable type
    - creating a frequency table
    - counting the number of missing values
    - Then run this code to check the values of the new variable against the values of the input variable:
    - `wwlist %>% group_by(ethn_race) %>% count(ethn_code)`

```
#investigate input var ethn_code
str(wwlist$ethn_code)
#>  chr [1:268286] "other-2 or more" "white" "white" "other-2 or more" "white" ...
wwlist %>% count(ethn_code)
#> # A tibble: 10 x 2
#>    ethn_code                                          n
#>    <chr>                                          <int>
#>  1 american indian or alaska native                 202
#>  2 asian or native hawaiian or other pacific islander  2385
#>  3 black or african american                        563
#>  4 cuban                                             70
#>  5 mexican/mexican american                       6548
#>  6 not reported                                   5736
#>  7 other spanish/hispanic                         2429
#>  8 other-2 or more                               90543
#>  9 puerto rican                                     195
#> 10 white                                         159615
wwlist %>% filter(is.na(ethn_code)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0

#create new variable ethn_race
wwlist <- wwlist %>%
  mutate(ethn_race =
    recode(ethn_code,
      "american indian or alaska native" = "nativeam",
      "asian or native hawaiian or other pacific islander" = "api",
      "black or african american" = "black",
```

```
      "cuban" = "latinx",
      "mexican/mexican american" = "latinx",
      "not reported" = "not_reported",
      "other-2 or more" = "multirace",
      "other spanish/hispanic" = "latinx",
      "puerto rican" = "latinx",
      "white" = "white",
    )
  )

#investigate values of new variable
str(wwlist$ethn_race)
#>  chr [1:268286] "multirace" "white" "white" "multirace" "white" "multirace" ...
wwlist %>% count(ethn_race)
#> # A tibble: 7 x 2
#>   ethn_race         n
#>   <chr>         <int>
#> 1 api            2385
#> 2 black           563
#> 3 latinx         9242
#> 4 multirace     90543
#> 5 nativeam        202
#> 6 not_reported   5736
#> 7 white        159615
wwlist %>% filter(is.na(ethn_race)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% group_by(ethn_race) %>% count(ethn_code)
#> # A tibble: 10 x 3
#>    ethn_race    ethn_code                                          n
#>    <chr>        <chr>                                          <int>
#>  1 api          asian or native hawaiian or other pacific islander  2385
#>  2 black        black or african american                         563
#>  3 latinx       cuban                                              70
#>  4 latinx       mexican/mexican american                         6548
#>  5 latinx       other spanish/hispanic                           2429
#>  6 latinx       puerto rican                                      195
#>  7 multirace    other-2 or more                                 90543
#>  8 nativeam     american indian or alaska native                  202
#>  9 not_reported not reported                                     5736
#> 10 white        white                                          159615
```

Question 7

- Based on the variable `ethn_race` you just created, create a set of 0/1 prospect-level race indicator indicators
- `nativeam_stu`; `api_stu`; `black_stu`; `latinx_stu`; `multirace_stu`; `white_stu`, `notreported_stu`
- after creating the 0/1 indicators check their values against the value of the input variable

NOTE: IN THE BELOW CODE CHUNK, I'LL CREATE THE INDICATOR FOR `nativeam_stu`; YOU CREATE THE REMAINING
Uncomment this code chunk after creating the `ethn_code` variable from the code chunk above

```r
#investigate input var
wwlist %>% count(ethn_code)
#> # A tibble: 10 x 2
#>    ethn_code                                              n
#>    <chr>                                              <int>
#>  1 american indian or alaska native                     202
#>  2 asian or native hawaiian or other pacific islander  2385
#>  3 black or african american                            563
#>  4 cuban                                                 70
#>  5 mexican/mexican american                            6548
#>  6 not reported                                        5736
#>  7 other spanish/hispanic                              2429
#>  8 other-2 or more                                    90543
#>  9 puerto rican                                         195
#> 10 white                                             159615

#Create var
wwlist <- wwlist %>%
  mutate(nativeam_stu = if_else(ethn_race == "nativeam",1,0))

#Investigate var
wwlist %>% count(nativeam_stu)
#> # A tibble: 2 x 2
#>   nativeam_stu      n
#>          <dbl>  <int>
#> 1            0 268084
#> 2            1    202
wwlist %>% group_by(nativeam_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   nativeam_stu ethn_race         n
#>          <dbl> <chr>         <int>
#> 1            0 api            2385
#> 2            0 black           563
#> 3            0 latinx         9242
#> 4            0 multirace     90543
#> 5            0 not_reported   5736
#> 6            0 white        159615
#> 7            1 nativeam        202

#Create remaining vars
wwlist <- wwlist %>%
  mutate(
    api_stu = if_else(ethn_race == "api",1,0),
    black_stu = if_else(ethn_race == "black",1,0),
    latinx_stu = if_else(ethn_race == "latinx",1,0),
    multirace_stu = if_else(ethn_race == "multirace",1,0),
    white_stu = if_else(ethn_race == "white",1,0),
    notreported_stu = if_else(ethn_race == "not_reported",1,0),
  )

#Investigate remaining vars
wwlist %>% count(api_stu)
#> # A tibble: 2 x 2
```

```
#>   api_stu      n
#>     <dbl>  <int>
#> 1       0 265901
#> 2       1   2385
wwlist %>% group_by(api_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   api_stu ethn_race         n
#>     <dbl> <chr>         <int>
#> 1       0 black           563
#> 2       0 latinx         9242
#> 3       0 multirace     90543
#> 4       0 nativeam        202
#> 5       0 not_reported   5736
#> 6       0 white        159615
#> 7       1 api            2385

wwlist %>% count(black_stu)
#> # A tibble: 2 x 2
#>   black_stu      n
#>       <dbl>  <int>
#> 1         0 267723
#> 2         1    563
wwlist %>% group_by(black_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   black_stu ethn_race         n
#>       <dbl> <chr>         <int>
#> 1         0 api            2385
#> 2         0 latinx         9242
#> 3         0 multirace     90543
#> 4         0 nativeam        202
#> 5         0 not_reported   5736
#> 6         0 white        159615
#> 7         1 black           563

wwlist %>% count(latinx_stu)
#> # A tibble: 2 x 2
#>   latinx_stu      n
#>        <dbl>  <int>
#> 1          0 259044
#> 2          1   9242
wwlist %>% group_by(latinx_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   latinx_stu ethn_race         n
#>        <dbl> <chr>         <int>
#> 1          0 api            2385
#> 2          0 black           563
#> 3          0 multirace     90543
#> 4          0 nativeam        202
#> 5          0 not_reported   5736
#> 6          0 white        159615
#> 7          1 latinx         9242

wwlist %>% count(multirace_stu)
```

```
#> # A tibble: 2 x 2
#>   multirace_stu      n
#>           <dbl>  <int>
#> 1             0 177743
#> 2             1  90543
wwlist %>% group_by(multirace_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   multirace_stu ethn_race         n
#>           <dbl> <chr>         <int>
#> 1             0 api            2385
#> 2             0 black           563
#> 3             0 latinx         9242
#> 4             0 nativeam        202
#> 5             0 not_reported   5736
#> 6             0 white        159615
#> 7             1 multirace     90543

wwlist %>% count(white_stu)
#> # A tibble: 2 x 2
#>   white_stu      n
#>       <dbl>  <int>
#> 1         0 108671
#> 2         1 159615
wwlist %>% group_by(white_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   white_stu ethn_race         n
#>       <dbl> <chr>         <int>
#> 1         0 api            2385
#> 2         0 black           563
#> 3         0 latinx         9242
#> 4         0 multirace     90543
#> 5         0 nativeam        202
#> 6         0 not_reported   5736
#> 7         1 white        159615

wwlist %>% count(notreported_stu)
#> # A tibble: 2 x 2
#>   notreported_stu      n
#>             <dbl>  <int>
#> 1               0 262550
#> 2               1   5736
wwlist %>% group_by(notreported_stu) %>% count(ethn_race)
#> # A tibble: 7 x 3
#>   notreported_stu ethn_race         n
#>             <dbl> <chr>         <int>
#> 1               0 api            2385
#> 2               0 black           563
#> 3               0 latinx         9242
#> 4               0 multirace     90543
#> 5               0 nativeam        202
#> 6               0 white        159615
#> 7               1 not_reported   5736
```

Part III: group_by() and summarise() questions

Now that we have cleaned data and created variables in prospect-level dataset, we can use `group_by()` and `summarise()` to perform calculations across rows about the characteristics of prospects purchased and how they compare to the general population. Generally, for the below questions you don't need to retain/assign the object created by `group_by()` and `summarise()`

Question 1

- Grouping by the variable `in_state`, use `summarise()` to create the following measures:
  - `tot_prosp`: a count of the number of prospects purchased

```
names(wwlist)
#>  [1] "receive_date"        "psat_range"          "state"
#>  [4] "zip9"                "for_country"         "sex"
#>  [7] "hs_ceeb_code"        "hs_name"             "hs_city"
#> [10] "hs_state"            "hs_grad_date"        "ethn_code"
#> [13] "homeschool"          "firstgen"            "zip5"
#> [16] "pop_total_zip"       "pop_white_zip"       "pop_black_zip"
#> [19] "pop_latinx_zip"      "pop_nativeam_zip"    "pop_multirace_zip"
#> [22] "pop_otherrace_zip"   "med_inc_zip"         "school_type"
#> [25] "merged_hs"           "school_category"     "total_12"
#> [28] "total_students"      "fr_lunch"            "pop_total_state"
#> [31] "pop_white_state"     "pop_black_state"     "pop_nativeam_state"
#> [34] "pop_otherrace_state" "pop_multirace_state" "pop_latinx_state"
#> [37] "med_inc_state"       "in_state"            "pop_api_zip"
#> [40] "pop_api_state"       "pct_white_zip"       "pct_black_zip"
#> [43] "pct_latinx_zip"      "pct_nativeam_zip"    "pct_multirace_zip"
#> [46] "pct_otherrace_zip"   "pct_api_zip"         "pct_white_state"
#> [49] "pct_black_state"     "pct_latinx_state"    "pct_nativeam_state"
#> [52] "pct_multirace_state" "pct_otherrace_state" "pct_api_state"
#> [55] "ethn_race"           "nativeam_stu"        "api_stu"
#> [58] "black_stu"           "latinx_stu"          "multirace_stu"
#> [61] "white_stu"           "notreported_stu"
wwlist %>% group_by(in_state) %>% summarise(total_prosp=n())
#> # A tibble: 2 x 2
#>   in_state total_prosp
#>      <dbl>       <int>
#> 1        0      172268
#> 2        1       96018
```

Question 2

- Grouping by the variable `in_state`, use `summarise()` to create the following measures:
  - `tot_prosp`: a count of the number of prospects purchased
  - `white`: a count of number of white prospects purchased, based on the input var `white_stu`
    - **hint:** `newvar = sum(input_var, na.rm=TRUE)`

```
wwlist %>% group_by(in_state) %>%
  summarise(
    tot_prosp=n(),
    white=sum(white_stu, na.rm=TRUE)
    )
#> # A tibble: 2 x 3
#>   in_state tot_prosp  white
```

```
#>      <dbl>     <int>  <dbl>
#> 1        0    172268 103981
#> 2        1     96018  55634
```

Question 3

- Grouping by the variable `in_state`, use `summarise()` to create the following measures:
  - `tot_prosp`: a count of the number of prospects purchased
  - `report_race`: the total number of prospects purchased that reported race (**hint:** `sum(ethn_race !="not_reported", na.rm=TRUE)`)
  - `white`: a count of number of white prospects purchased, based on the input var `white_stu`

```
wwlist %>% count(ethn_race)
#> # A tibble: 7 x 2
#>   ethn_race         n
#>   <chr>         <int>
#> 1 api            2385
#> 2 black           563
#> 3 latinx         9242
#> 4 multirace     90543
#> 5 nativeam        202
#> 6 not_reported   5736
#> 7 white        159615
#wwlist %>% group_by(in_state) %>% count(ethn_race)

wwlist %>% group_by(in_state) %>%
  summarise(
    tot_prosp=n(),
    report_race = sum(ethn_race != "not_reported", na.rm=TRUE),
    white=sum(white_stu, na.rm=TRUE)
    )
#> # A tibble: 2 x 4
#>   in_state tot_prosp report_race  white
#>      <dbl>     <int>       <int>  <dbl>
#> 1        0    172268      168877 103981
#> 2        1     96018       93673  55634
```

Question 4

- Grouping by the variable `in_state`, use `summarise()` to create the following measures:
  - `tot_prosp`: a count of the number of prospects purchased
  - 'report_race: the total number of prospects purchased that reported race

  - a count of number of prospects purchased by race based on each of the following input variables (that is, you will create 7 variables)
    - **nativeam_stu , api_stu , black_stu , latinx_stu , multirace_stu , white_stu , notreported_stu**

```
wwlist %>% group_by(in_state) %>%
  summarise(
    tot_prosp=n(),
    report_race = sum(ethn_race != "not_reported", na.rm=TRUE),
    nativeam=sum(nativeam_stu, na.rm=TRUE),
    api=sum(api_stu, na.rm=TRUE),
```

```
    black=sum(black_stu, na.rm=TRUE),
    latinx=sum(latinx_stu, na.rm=TRUE),
    multirace=sum(multirace_stu, na.rm=TRUE),
    white=sum(white_stu, na.rm=TRUE),
    notreported=sum(notreported_stu, na.rm=TRUE)
  )
#> # A tibble: 2 x 10
#>   in_state tot_prosp report_~1 nativ~2   api black latinx multi~3  white notre~4
#>      <dbl>     <int>     <int>   <dbl> <dbl> <dbl>  <dbl>   <dbl>  <dbl>   <dbl>
#> 1        0    172268    168877     102  1323   229   3974   59268 103981    3391
#> 2        1     96018     93673     100  1062   334   5268   31275  55634    2345
#> # ... with abbreviated variable names 1: report_race, 2: nativeam,
#> #   3: multirace, 4: notreported
```

Question 5

- Grouping by the variable `in_state`, use `summarise()` to create the following measures:
- `tot_prosp`: a count of the number of prospects purchased
- `white`: a count of number of white prospects purchased, based on the input var `white_stu`

- `p_white`: the proportion of prospects purchased that were white for each by group, based on the 0/1 input var `white_stu`

- **hint:** `newvar = mean(input_var, na.rm=TRUE)`

```
wwlist %>% group_by(in_state) %>%
  summarise(
    tot_prosp=n(),
    white=sum(white_stu, na.rm=TRUE),
    p_white=mean(white_stu, na.rm=TRUE)
  )
#> # A tibble: 2 x 4
#>   in_state tot_prosp  white p_white
#>      <dbl>     <int>  <dbl>   <dbl>
#> 1        0    172268 103981   0.604
#> 2        1     96018  55634   0.579
```

Question 6

- Grouping by the variable `in_state`, use `summarise()` to create the following measures:
- `tot_prosp`: a count of the number of prospects purchased
- the **percent** of prospects purchased from each race group based on the following 0/1 indicator variables (that is, you will create 7 variables)
  - **nativeam_stu , api_stu , black_stu , latinx_stu , multirace_stu , white_stu , notreported_stu**
  - **hint:** since you are creating **percent** measures rather than **proportion**: `newvar = mean(input_var)*100`

```
wwlist %>% group_by(in_state) %>%
  summarise(
    tot_prosp=n(),
    p_nativeam=mean(nativeam_stu, na.rm=TRUE)*100,
    p_api=mean(api_stu, na.rm=TRUE)*100,
    p_black=mean(black_stu, na.rm=TRUE)*100,
```

```
    p_latinx=mean(latinx_stu, na.rm=TRUE)*100,
    p_multirace=mean(multirace_stu, na.rm=TRUE)*100,
    p_white=mean(white_stu, na.rm=TRUE)*100,
    p_notreported=mean(notreported_stu, na.rm=TRUE)*100
  )
#> # A tibble: 2 x 9
#>   in_state tot_prosp p_nativeam p_api p_black p_latinx p_multi~1 p_white p_not~2
#>      <dbl>     <int>      <dbl> <dbl>   <dbl>    <dbl>     <dbl>   <dbl>   <dbl>
#> 1        0    172268     0.0592 0.768   0.133     2.31      34.4    60.4    1.97
#> 2        1     96018     0.104  1.11    0.348     5.49      32.6    57.9    2.44
#> # ... with abbreviated variable names 1: p_multirace, 2: p_notreported
```

Question 7

- Now we will group_by the variable **state** (rather than **in_state**), use **summarise()** to create the following measures:
  - **tot_prosp**: a count of the number of prospects purchased
  - **white**: a count of number of white prospects purchased, based on the input var **white_stu**
  - **p_white**: the **percent** of prospects purchased that were white for each by group, based on the 0/1 input var **white_stu**

```
wwlist %>% group_by(state) %>%
  summarise(
    tot_prospects=n(),
    white=sum(white_stu, na.rm=TRUE),
    p_white=mean(white_stu, na.rm=TRUE)*100
  )
#> # A tibble: 51 x 4
#>    state tot_prospects white p_white
#>    <chr>         <int> <dbl>   <dbl>
#>  1 AK             3671  2457    66.9
#>  2 AL              136   110    80.9
#>  3 AR               78    68    87.2
#>  4 AZ            10358  6659    64.3
#>  5 CA            62382 29981    48.1
#>  6 CO            24822 18740    75.5
#>  7 CT              173   147    85.0
#>  8 DC               35    23    65.7
#>  9 DE               37    29    78.4
#> 10 FL             1287   882    68.5
#> # ... with 41 more rows
#> # i Use `print(n = ...)` to see more rows
```

Part IV: Comparing prospects purchased to regional income and racial demographics

Question 1

In this question, we will compare median zip code income of prospects purchased to the median income in the states they live in. The goal is to assess whether Western Washington is disproportionately purchasing more affluent prospects. The variable **med_inc_state** identifies the median income of all people in the state aged 25-64. This variable has the same value for all prospects in the same state. Therefore, when using

`group_by()` and `summarise()`, we can just grab the first observation for each state (hint: `first(input_var)` or `nth(input_var,1)`).

To answer this question, group_by **`state`** and use `summarise()` to create the following measures:

- `tot_prosp`: a count of the number of prospects purchased
- `med_inc_zip_stu`: the mean value of the variable `med_inc_zip` for each by group
- `med_inc_state`: the first value of the variable `med_inc_state` for each by group

```
wwlist %>% group_by(state) %>%
  summarise(
    tot_prosp=n(),
    med_inc_zip_stu=mean(med_inc_zip, na.rm=TRUE),
    med_inc_state=first(med_inc_state),
  )
#> # A tibble: 51 x 4
#>    state tot_prosp med_inc_zip_stu med_inc_state
#>    <chr>     <int>           <dbl>         <dbl>
#>  1 AK         3671          93424.        81289
#>  2 AL          136          80987.        51192.
#>  3 AR           78          64461.        48587
#>  4 AZ        10358          77840.        58138.
#>  5 CA        62382         132135.        71674.
#>  6 CO        24822          94807.        71388.
#>  7 CT          173         181426.        82469
#>  8 DC           35         140784.        80166
#>  9 DE           37         102944.        69466.
#> 10 FL         1287          75452.        54650.
#> # ... with 41 more rows
#> # i Use `print(n = ...)` to see more rows

#Playing with formatting [optional]
wwlist %>% group_by(state) %>%
  summarise(
    tot_prosp=n(),
    med_inc_zip_stu=round(mean(med_inc_zip, na.rm=TRUE)),
    med_inc_state=round(first(med_inc_state)),
  )
#> # A tibble: 51 x 4
#>    state tot_prosp med_inc_zip_stu med_inc_state
#>    <chr>     <int>           <dbl>         <dbl>
#>  1 AK         3671           93424         81289
#>  2 AL          136           80987         51192
#>  3 AR           78           64461         48587
#>  4 AZ        10358           77840         58138
#>  5 CA        62382          132135         71674
#>  6 CO        24822           94807         71388
#>  7 CT          173          181426         82469
#>  8 DC           35          140784         80166
#>  9 DE           37          102944         69466
#> 10 FL         1287           75452         54650
#> # ... with 41 more rows
#> # i Use `print(n = ...)` to see more rows
```

```
#format(round(as.numeric(1000.64), 1), nsmall=1, big.mark=",")
wwlist %>% group_by(state) %>%
  summarise(
    tot_prosp=n(),
    med_inc_zip_stu=format(round(mean(med_inc_zip, na.rm=TRUE)),nsmall=0, big.mark=",") ,
    med_inc_state=format(round(first(med_inc_state)),nsmall=0, big.mark=",") ,
  )
#> # A tibble: 51 x 4
#>    state tot_prosp med_inc_zip_stu med_inc_state
#>    <chr>     <int> <chr>           <chr>
#>  1 AK         3671 93,424          81,289
#>  2 AL          136 80,987          51,192
#>  3 AR           78 64,461          48,587
#>  4 AZ        10358 77,840          58,138
#>  5 CA        62382 132,135         71,674
#>  6 CO        24822 94,807          71,388
#>  7 CT          173 181,426         82,469
#>  8 DC           35 140,784         80,166
#>  9 DE           37 102,944         69,466
#> 10 FL         1287 75,452          54,650
#> # ... with 41 more rows
#> # i Use `print(n = ...)` to see more rows
```

Question 2

For each state, we want to compare the percent of prospects purchased who are white to the percent of people in the state who are white. The variable `pct_white_state` identifies the percent of people in the state who are white. This variable has the same value for all prospects in the same state. Therefore, when using `group_by()` and `summarise()`, we can grab the first observation for each state (hint: `first(input_var)` or `nth(input_var,1)`).

- group_by **`state`** and use `summarise()` to create the following measures:
  - `tot_prosp`: a count of the number of prospects purchased
  - `white`: a count of number of white prospects purchased, based on the input var `white_stu`
  - `p_white`: the **percent** of prospects purchased that were white for each by group, based on the 0/1 input var `white_stu`
  - `p_white_st`: the percent of people in the state who are White, based on the input variable `pct_white_state`

```
wwlist %>% group_by(state) %>%
  summarise(
    tot_prosp=n(),
    white=sum(white_stu, na.rm=TRUE),
    p_white=mean(white_stu, na.rm=TRUE)*100,
    p_white_st = first(pct_white_state)
  )
#> # A tibble: 51 x 5
#>    state tot_prosp white p_white p_white_st
#>    <chr>     <int> <dbl>   <dbl>      <dbl>
#>  1 AK         3671  2457    66.9       62.0
#>  2 AL          136   110    80.9       66.2
#>  3 AR           78    68    87.2       73.4
#>  4 AZ        10358  6659    64.3       56.1
```

```
#>  5 CA        62382 29981    48.1       38.4
#>  6 CO        24822 18740    75.5       69.0
#>  7 CT          173   147    85.0       68.7
#>  8 DC           35    23    65.7       35.8
#>  9 DE           37    29    78.4       63.5
#> 10 FL         1287   882    68.5       55.6
#> # ... with 41 more rows
#> # i Use `print(n = ...)` to see more rows
```

Question 3

- group_by **state** and use `summarise()` to create the following measures:
  - `tot_prosp`: a count of the number of prospects purchased
  - Create (A) a measure of the percent of prospects who identify as a particular race/ethnicity group and (B) the percent of people in the state who identify as that particular race/ethnicity group for the following race/ethnicity groups: **multirace, white, api, black, latinx**

```
wwlist %>% group_by(state) %>%
  summarise(
    tot_prosp=n(),
    p_multirace=mean(multirace_stu, na.rm=TRUE)*100,
    p_multirace_st=first(pct_multirace_state),
    p_white=mean(white_stu, na.rm=TRUE)*100,
    p_white_st = first(pct_white_state),
    p_api=mean(api_stu, na.rm=TRUE)*100,
    p_api_st = first(pct_api_state),
    p_black=mean(black_stu, na.rm=TRUE)*100,
    p_black_st = first(pct_black_state),
    p_latinx=mean(latinx_stu, na.rm=TRUE)*100,
    p_latinx_st = first(pct_latinx_state),
  )
#> # A tibble: 51 x 12
#>    state tot_prosp p_mul~1 p_mul~2 p_white p_whi~3 p_api p_api~4 p_black p_bla~5
#>    <chr>     <int>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>   <dbl>   <dbl>
#>  1 AK         3671    29.0    7.39    66.9    62.0 0.463    7.05  0.191     3.12
#>  2 AL          136    17.6    1.61    80.9    66.2 0        1.27  0.735    26.4
#>  3 AR           78    10.3    1.96    87.2    73.4 0        1.61  0        15.4
#>  4 AZ        10358    27.8    2.08    64.3    56.1 0.463    3.14  0.251     4.01
#>  5 CA        62382    45.7    2.87    48.1    38.4 1.03    14.0   0.0978    5.58
#>  6 CO        24822    21.8    2.30    75.5    69.0 0.616    3.00  0.262     3.87
#>  7 CT          173    12.1    1.97    85.0    68.7 0        4.24  0.578     9.69
#>  8 DC           35    25.7    2.21    65.7    35.8 0        3.63  0        47.4
#>  9 DE           37    21.6    2.29    78.4    63.5 0        3.68  0        21.1
#> 10 FL         1287    27.0    1.75    68.5    55.6 0.389    2.61  0        15.4
#> # ... with 41 more rows, 2 more variables: p_latinx <dbl>, p_latinx_st <dbl>,
#> #   and abbreviated variable names 1: p_multirace, 2: p_multirace_st,
#> #   3: p_white_st, 4: p_api_st, 5: p_black_st
#> # i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

Question 4

- The goal of this question is to compare the race of prospects purchased from Washington to the racial composition of zip-codes in Washington. For this question, you will filter to **only include prospects who are from Washington AND do not have the value `NA` for the variable `pop_total_zip`**,

then group by the variable `zip5` and use `summarise()` to create the following variables:
  - `tot_prosp`: a count of the number of prospects purchased
  - Create (A) a measure of the percent of prospects in the zip-code who identify as a particular race/ethnicity group and (B) the percent of people in the zip-code who identify as that particular race/ethnicity group for the following race/ethnicity groups: **multirace, white, api, black, latinx**

```
wwlist %>% filter(is.na(zip5)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0
wwlist %>% filter(state == "WA", is.na(pop_total_zip)) %>% count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1   429

wwlist %>% filter(state == "WA",!is.na(pop_total_zip)) %>% group_by(zip5) %>%
  summarise(
    tot_prosp=n(),
    p_multirace=mean(multirace_stu, na.rm=TRUE)*100,
    p_multirace_zip=first(pct_multirace_zip),
    p_white=mean(white_stu, na.rm=TRUE)*100,
    p_white_zip = first(pct_white_zip),
    p_api=mean(api_stu, na.rm=TRUE)*100,
    p_api_zip = first(pct_api_zip),
    p_black=mean(black_stu, na.rm=TRUE)*100,
    p_black_zip = first(pct_black_zip),
    p_latinx=mean(latinx_stu, na.rm=TRUE)*100,
    p_latinx_zip = first(pct_latinx_zip),
  )
#> # A tibble: 556 x 12
#>    zip5  tot_prosp p_mul~1 p_mul~2 p_white p_whi~3 p_api p_api~4 p_black p_bla~5
#>    <chr>     <int>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>   <dbl>   <dbl>
#>  1 20008         1     0      2.17   100      71.4 0        5.94   0        7.62
#>  2 98001       506    44.5    5.47    45.1    61.8 1.58    14.2    0.198    7.16
#>  3 98002       347    41.8    4.79    35.4    56.5 1.15     8.61   1.15     5.13
#>  4 98003       487    45.8    5.62    32.2    46.8 3.90    14.6    1.03    11.7
#>  5 98004       741    51.6    5.22    44.0    60.1 0.945   28.8    0.270    1.58
#>  6 98005       456    54.6    5.90    36.0    49.2 3.73    34.4    0.439    3.28
#>  7 98006      1514    59.6    4.09    35.1    53.7 1.85    33.3    0.198    1.74
#>  8 98007       360    53.6    2.95    30      41.7 3.61    42.2    0.833    2.99
#>  9 98008       573    44.7    3.66    47.6    60.8 2.27    24.0    0        3.34
#> 10 98010        93    17.2    1.85    79.6    79.2 2.15     2.68   0        1.65
#> # ... with 546 more rows, 2 more variables: p_latinx <dbl>, p_latinx_zip <dbl>,
#> #   and abbreviated variable names 1: p_multirace, 2: p_multirace_zip,
#> #   3: p_white_zip, 4: p_api_zip, 5: p_black_zip
#> # i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

Once finished, knit to (pdf) and upload both .Rmd and pdf files to class website under the week 4 tab
Remeber to use this naming convention "lastname_firstname_ps4"