Exploring Census Data

Emma Johnson

December 2025

S1201: Marital Status

```
browseURL("https://data.census.gov/table/ACSST1Y2024.S1201?g=010XX00US$0400000")
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   3.5.2     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.1.0
## -- 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 errors
```

```
library(tidycensus)
```

```
## Warning: package 'tidycensus' was built under R version 4.5.2
```

Here is the typical set-up to fetch census data from the United States Census Bureau:

```
#census_api_key(API_key, install = TRUE)
## Use your API key ^ here

marital_status <- get_acs(geography = "state", table = "S1201")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```
## Loading ACS5/SUBJECT variables for 2023 from table S1201. To cache this dataset for faster access to
```

```
## Using the ACS Subject Tables
## Using the ACS Subject Tables
## Using the ACS Subject Tables
## Using the ACS Subject Tables
## Using the ACS Subject Tables
## Using the ACS Subject Tables
## Using the ACS Subject Tables
## Using the ACS Subject Tables
```

```r
glimpse(marital_status)
```

```
## Rows: 9,984
## Columns: 5
## $ GEOID    <chr> "01", "01", "01", "01", "01", "01", "01", "01", "01", "01", "~
## $ NAME     <chr> "Alabama", "Alabama", "Alabama", "Alabama", "Alabama", "Alaba~
## $ variable <chr> "S1201_C01_001", "S1201_C01_002", "S1201_C01_003", "S1201_C01~
## $ estimate <dbl> 4124301, 1978662, 172319, 488248, 303517, 303401, 321761, 389~
## $ moe      <dbl> 773, 1057, 1324, 1207, 985, 1050, 569, 474, 873, 1196, 1346, ~
```

Notice how the variables are not informative under their current codes, so we will now fetch more descriptive names using `load_variables()`. To keep the data as concise as possible, we will use these variable names in our main dataframe, `ms` (marital status).

```r
ms <- marital_status %>%
  left_join(
    load_variables(2024, "acs1/subject"),
    join_by(variable == name),
    keep = FALSE,
    relationship = "many-to-one"
            ) %>%
  select(-concept)

glimpse(ms)
```

```
## Rows: 9,984
## Columns: 6
## $ GEOID    <chr> "01", "01", "01", "01", "01", "01", "01", "01", "01", "01", "~
## $ NAME     <chr> "Alabama", "Alabama", "Alabama", "Alabama", "Alabama", "Alaba~
## $ variable <chr> "S1201_C01_001", "S1201_C01_002", "S1201_C01_003", "S1201_C01~
## $ estimate <dbl> 4124301, 1978662, 172319, 488248, 303517, 303401, 321761, 389~
## $ moe      <dbl> 773, 1057, 1324, 1207, 985, 1050, 569, 474, 873, 1196, 1346, ~
## $ label    <chr> "Estimate!!Total!!Population 15 years and over", "Estimate!!T~
```

By taking a glimpse of the data, you may notice that the variable names are long and inefficiently named. This is because the data has levels of headers, which can make calling a variable quite difficult. So, the easiest way I found to fix this was to ask ChatGPT to clean it up. Otherwise, feel free to have at it yourself.

```r
## With a little clean-up help from ChatGPT:
ms$clean <- ms$label %>%
  sub("^Estimate!!", "", .) %>%
  str_split("!!") %>%
  sapply(function(parts) {
    parts <- trimws(tolower(parts))

    # ---- 1. marital status ----
    status <- parts[1]

    # normalize "now married (except separated)" to "married"
    status <- gsub("^now married.*", "married", status)

    # ---- 2. last meaningful piece ----
```

```r
    last <- parts[length(parts)]

    # ---- 3. detect sex (male/female) anywhere ----
    sex_word <- NA_character_
    if (any(grepl("\\bmales?\\b", parts)))   sex_word <- "male"
    if (any(grepl("\\bfemales?\\b", parts))) sex_word <- "female"

    # ---- 4. remove male/female from final chunk to avoid duplicates ----
    if (!is.na(sex_word)) {
      last <- gsub("\\bmales?\\b",   "", last)
      last <- gsub("\\bfemales?\\b", "", last)
      label <- paste(status, sex_word, last)
    } else {
      label <- paste(status, last)
    }

    # ---- 5. snake_case cleanup ----
    label <- gsub("[^a-z0-9]+", "_", label)
    label <- gsub("_+", "_", label)
    label <- gsub("^_|_$", "", label)

    label
  })

ms <- ms %>%
  select(geo_id = GEOID, name = NAME, variable = clean, estimate, error_margin = moe)

glimpse(ms)
```

```
## Rows: 9,984
## Columns: 5
## $ geo_id       <chr> "01", "01", "01", "01", "01", "01", "01", "01", "01", "01~
## $ name         <chr> "Alabama", "Alabama", "Alabama", "Alabama", "Alabama", "A~
## $ variable     <chr> "total_population_15_years_and_over", "total_male_15_year~
## $ estimate     <dbl> 4124301, 1978662, 172319, 488248, 303517, 303401, 321761,~
## $ error_margin <dbl> 773, 1057, 1324, 1207, 985, 1050, 569, 474, 873, 1196, 13~
```

There is lots of redundancy in the data due to grouping by state. That is, columns `geo_id` and `name` (i.e. state/territory name) are repeated for each of their corresponding observations. To improve efficiency, we will want to use this grouping to our advantage, rather than having to make the system re-group every time we want to look at specific states. So, we will nest each territory's data into it's own dataframe using `group_nest` and `group_keys` for lookup.

```r
ms_by_state <- ms %>%
  group_by(geo_id, name) %>%
  group_nest(keep = FALSE)

state_keys <- ms %>%
  group_by(geo_id, name) %>%
  group_keys()

# e.g. Alabama
ms_by_state$data[[1]]
```

```
## # A tibble: 192 x 3
##    variable                             estimate error_margin
##    <chr>                                   <dbl>        <dbl>
##  1 total_population_15_years_and_over  4124301          773
##  2 total_male_15_years_and_over        1978662         1057
##  3 total_male_15_to_19_years            172319         1324
##  4 total_male_20_to_34_years            488248         1207
##  5 total_male_35_to_44_years            303517          985
##  6 total_male_45_to_54_years            303401         1050
##  7 total_male_55_to_64_years            321761          569
##  8 total_male_65_years_and_over         389416          474
##  9 total_female_15_years_and_over      2145639          873
## 10 total_female_15_to_19_years          169532         1196
## # i 182 more rows
```

It's looking much more digestible already. In addition to the grouping by state, there is also latent groupings of variable type. To optimize efficiency once more, I created an index for the types of variables, so finding them will be easier later.

```
variable_index <- list(
  total_pop = ms$variable[1:15],
  race = ms$variable[16:26],
  labor_force = ms$variable[27:30],
  ratio_sex = ms$variable[31],

  marital_status_percentage = ms$variable[32],
  married_pop = ms$variable[33:47],
  married_race = ms$variable[48:58],
  married_work_force = ms$variable[59:62],

  widowed_pop = ms$variable[65:79],
  widowed_race = ms$variable[80:90],
  widowed_work_force = ms$variable[91:94],

  divorced_pop = ms$variable[97:111],
  divorced_race = ms$variable[112:122],
  divorced_work_force = ms$variable[123:126],

  separated_pop = ms$variable[129:143],
  separated_race = ms$variable[144:154],
  separated_work_force = ms$variable[155:158],

  never_married_pop = ms$variable[161:175],
  never_married_race = ms$variable[176:186],
  never_married_work_force = ms$variable[187:190]
)
```

As an example of what we've cleaned up so far, here is how we can fetch a group of variables for a specific state, or even for all states in a comparison:

```
# Widowed population variables in Alabama (2 ways):
ms_by_state$data[[1]] %>%
  filter(variable %in% variable_index$widowed_pop)
```

```
## # A tibble: 16 x 3
##    variable                               estimate error_margin
##    <chr>                                     <dbl>        <dbl>
##  1 widowed_population_15_years_and_over        6.9          0.1
##  2 widowed_male_15_years_and_over              3.3          0.1
##  3 widowed_male_15_to_19_years                 0            0.1
##  4 widowed_male_20_to_34_years                 0.1          0.1
##  5 widowed_male_35_to_44_years                 0.4          0.1
##  6 widowed_male_45_to_54_years                 1.5          0.2
##  7 widowed_male_55_to_64_years                 3.5          0.3
##  8 widowed_male_65_years_and_over             12.2          0.4
##  9 widowed_female_15_years_and_over           10.2          0.2
## 10 widowed_female_15_to_19_years               0            0.1
## 11 widowed_female_20_to_34_years               0.3          0.1
## 12 widowed_female_35_to_44_years               1.2          0.2
## 13 widowed_female_45_to_54_years               3.4          0.4
## 14 widowed_female_55_to_64_years               9.3          0.4
## 15 widowed_female_65_years_and_over           34.5          0.5
## 16 widowed_population_15_years_and_over        6.9          0.1
```

```r
# OR
ms_by_state %>%
  filter(name == "Alabama") %>%
  unnest(cols = everything()) %>%
  filter(variable %in% variable_index$widowed_pop)
```

```
## # A tibble: 16 x 5
##    geo_id name    variable                               estimate error_margin
##    <chr>  <chr>   <chr>                                     <dbl>        <dbl>
##  1 01     Alabama widowed_population_15_years_and_over        6.9          0.1
##  2 01     Alabama widowed_male_15_years_and_over              3.3          0.1
##  3 01     Alabama widowed_male_15_to_19_years                 0            0.1
##  4 01     Alabama widowed_male_20_to_34_years                 0.1          0.1
##  5 01     Alabama widowed_male_35_to_44_years                 0.4          0.1
##  6 01     Alabama widowed_male_45_to_54_years                 1.5          0.2
##  7 01     Alabama widowed_male_55_to_64_years                 3.5          0.3
##  8 01     Alabama widowed_male_65_years_and_over             12.2          0.4
##  9 01     Alabama widowed_female_15_years_and_over           10.2          0.2
## 10 01     Alabama widowed_female_15_to_19_years               0            0.1
## 11 01     Alabama widowed_female_20_to_34_years               0.3          0.1
## 12 01     Alabama widowed_female_35_to_44_years               1.2          0.2
## 13 01     Alabama widowed_female_45_to_54_years               3.4          0.4
## 14 01     Alabama widowed_female_55_to_64_years               9.3          0.4
## 15 01     Alabama widowed_female_65_years_and_over           34.5          0.5
## 16 01     Alabama widowed_population_15_years_and_over        6.9          0.1
```

```r
# Ratio of Unmarried Men 15 to 44 years per 100 unmarried women 15 to 44 years
ms %>%
  filter(variable %in% variable_index$ratio_sex) %>%
  select(name, estimate, error_margin)
```

```
## # A tibble: 52 x 3
##    name                 estimate error_margin
```

```
##    <chr>                   <dbl>         <dbl>
##  1 Alabama                 104.            0.7
##  2 Alaska                  131.            4.2
##  3 Arizona                 115.            0.6
##  4 Arkansas                110.            1
##  5 California              113.            0.2
##  6 Colorado                120.            0.7
##  7 Connecticut             108.            0.7
##  8 Delaware                102.            1.4
##  9 District of Columbia     87.3           1
## 10 Florida                 110.            0.4
## # i 42 more rows
```