Intro to the Census API and Tidycensus

Aaron Kessler

December 05, 2022

The Census API

How you would deal with the Census API directly:

https://www.census.gov/data/developers/guidance/api-user-guide.Example_API_Queries.html#list-tab-2080675447

https://api.census.gov/data/2020/acs/acs5/subject/examples.html

How you call the api:

`https://api.census.gov/data/2020/acs/acs5?get=NAME,B19013_001E&for=state:*&key=YOURKEYGOESHE`

Where do I find census table codes?

A good place to start: https://censusreporter.org/tables/B01002/

Good place to find details of codes within the table: https://www.socialexplorer.com/data/ACS2015_5yr/metad

Census itself: https://api.census.gov/data/2019/acs/acs5/variables.html

All Census API documentation: https://api.census.gov/data.html

Giving it a try for real

Let's do it for real, with a real API call. We'll ask for median income by state:

https://api.census.gov/data/2020/acs/acs5?get=NAME,B19013_001E&for=state:*&key=2a6f8c21a30d3024e0

We can copy and paste this directly into a web browser and see if it works. Does it work?

Now let's ask for median income county, for Virginia:

https://api.census.gov/data/2020/acs/acs5/subject?get=NAME,S0101_C01_001E&for=county:*&in=state:51

We can try to convert this output into a table using the jsonlite package directly in R:

```r
mytable <- fromJSON("https://api.census.gov/data/2020/acs/acs5/subject?get=NAME,S0101_C01_

mytable %>%
  head(25)
```

```
      [,1]                              [,2]             [,3]    [,4]
 [1,] "NAME"                            "S0101_C01_001E" "state" "county"
 [2,] "Accomack County, Virginia"       "32560"          "51"    "001"
 [3,] "Alleghany County, Virginia"      "15030"          "51"    "005"
 [4,] "Amelia County, Virginia"         "12970"          "51"    "007"
 [5,] "Appomattox County, Virginia"     "15814"          "51"    "011"
 [6,] "Arlington County, Virginia"      "236434"         "51"    "013"
 [7,] "Bath County, Virginia"           "4248"           "51"    "017"
 [8,] "Bedford County, Virginia"        "78965"          "51"    "019"
 [9,] "Botetourt County, Virginia"      "33440"          "51"    "023"
[10,] "Brunswick County, Virginia"      "16336"          "51"    "025"
[11,] "Buckingham County, Virginia"     "17087"          "51"    "029"
[12,] "Campbell County, Virginia"       "55406"          "51"    "031"
[13,] "Carroll County, Virginia"        "29911"          "51"    "035"
[14,] "Charles City County, Virginia"   "6965"           "51"    "036"
[15,] "Charlotte County, Virginia"      "11953"          "51"    "037"
[16,] "Clarke County, Virginia"         "14498"          "51"    "043"
[17,] "Craig County, Virginia"          "5103"           "51"    "045"
[18,] "Cumberland County, Virginia"     "9869"           "51"    "049"
[19,] "Dickenson County, Virginia"      "14524"          "51"    "051"
[20,] "Essex County, Virginia"          "10960"          "51"    "057"
[21,] "Fairfax County, Virginia"        "1149439"        "51"    "059"
[22,] "Floyd County, Virginia"          "15766"          "51"    "063"
[23,] "Fluvanna County, Virginia"       "26873"          "51"    "065"
[24,] "Frederick County, Virginia"      "88054"          "51"    "069"
[25,] "Giles County, Virginia"          "16760"          "51"    "071"
```

There are still some issues here with the header names we'd have to clean up. But we'd get there. Even so, it can get pretty cumbersome having to work with URL combinations every time we want to grab something, and can be confusing when you're new to working with raw API calls overall.

Tidycensus to the rescue

While we could deal with all the intricacies of the raw Census API, we thankfully don't have to.

Why we'll use tidycensus instead.

https://walker-data.com/tidycensus/index.html

Let's talk why it's so helpful.

Credentials

First step to using it is loading your API Key credential. You thankfully only have to do this one time on your computer and it will create a little file that remembers it each time.

```
# uncomment to run, then recomment it out so you don't run it every time

# census_api_key("", install=TRUE)
```

Why might we not want to put our key in code that will be shared or visible publicly? Let's talk about the risk there.

How might we get around that? Well there are a few ways, but one of the best and most straightforward to is store the actual key in what's called the .REnviron file on your computer…and then just pull from that place in the code. That way you can do something like this:

```
# uncomment the line below to run - this assumes your key is saved in your .Renviron file

# census_api_key(Sys.getenv("MYCENSUSAPIKEY"), install=TRUE)
```

How do we find the .Renviron file? Using the usethis package, it's super easy, barely an inconvenience. (Kudos to anyone who gets that reference.) Run this line in the console and it will automatically locate the file on your computer for you:

`usethis::edit_r_environ()`

If you never put anything there before, it will just be blank. That's ok. Put in a line that includes the name you want to call your saved secret variable, and then the value with it:

`MYCENSUSAPIKEY='tktktktktktk'`

Then save the file, restart the R session, and you're done.

Census codes/variables

And of course, remember trying to find those Census variables? There's tidycensus itself which helps gather them together for you too!

```
censusvariables <- load_variables(2020, "acs5", cache = TRUE)
```

Let's get started pulling some data

```
#choose some census measures
medincome <- "B19013_001"
```

Make the call for ACS data, which default to the latest ACS5 (in this case 2016-2020.

```
get_acs(geography = "county",
        variables = medincome,
        state = "VA")
```

```
Getting data from the 2016-2020 5-year ACS
```

```
# A tibble: 133 x 5
   GEOID NAME                          variable   estimate   moe
   <chr> <chr>                         <chr>         <dbl> <dbl>
 1 51001 Accomack County, Virginia     B19013_001    46178  2575
 2 51003 Albemarle County, Virginia    B19013_001    84643  3217
 3 51005 Alleghany County, Virginia    B19013_001    48513  4275
 4 51007 Amelia County, Virginia       B19013_001    63918  6276
 5 51009 Amherst County, Virginia      B19013_001    57368  4546
 6 51011 Appomattox County, Virginia   B19013_001    55457  5246
 7 51013 Arlington County, Virginia    B19013_001   122604  2627
 8 51015 Augusta County, Virginia      B19013_001    65076  3262
 9 51017 Bath County, Virginia         B19013_001    55481 15306
10 51019 Bedford County, Virginia      B19013_001    67136  3303
# ... with 123 more rows
```

Can set it to wide format too. That's easier for us to work with here.

```
get_acs(geography = "county",
        variables = medincome,
        state = "VA",
        output = "wide")
```

```
Getting data from the 2016-2020 5-year ACS
```

```
# A tibble: 133 x 4
   GEOID NAME                           B19013_001E B19013_001M
   <chr> <chr>                                <dbl>       <dbl>
 1 51001 Accomack County, Virginia            46178        2575
 2 51005 Alleghany County, Virginia           48513        4275
 3 51007 Amelia County, Virginia              63918        6276
 4 51011 Appomattox County, Virginia          55457        5246
 5 51013 Arlington County, Virginia          122604        2627
 6 51017 Bath County, Virginia                55481       15306
 7 51019 Bedford County, Virginia             67136        3303
 8 51023 Botetourt County, Virginia           72719        4231
 9 51025 Brunswick County, Virginia           46111        3642
10 51029 Buckingham County, Virginia          48603        4866
# ... with 123 more rows
```

```
#let's add another variable to the mix
medage  <- "B01002_001"

#run it again for both income and age
get_acs(geography = "county",
        variables = c(medincome, medage),
        state = "VA")
```

```
Getting data from the 2016-2020 5-year ACS
```

```
# A tibble: 266 x 5
   GEOID NAME                        variable   estimate    moe
   <chr> <chr>                       <chr>         <dbl>  <dbl>
 1 51001 Accomack County, Virginia   B01002_001     45.8    0.5
 2 51001 Accomack County, Virginia   B19013_001  46178   2575
 3 51003 Albemarle County, Virginia  B01002_001     39.4    0.4
 4 51003 Albemarle County, Virginia  B19013_001  84643   3217
 5 51005 Alleghany County, Virginia  B01002_001     48      0.6
 6 51005 Alleghany County, Virginia  B19013_001  48513   4275
 7 51007 Amelia County, Virginia     B01002_001     45      1.8
 8 51007 Amelia County, Virginia     B19013_001  63918   6276
 9 51009 Amherst County, Virginia    B01002_001     44.9    0.2
10 51009 Amherst County, Virginia    B19013_001  57368   4546
# ... with 256 more rows
```

```r
#now we can see why wide is different
get_acs(geography = "county",
        variables = c(medincome, medage),
        state = "VA",
        output = "wide")
```

```
Getting data from the 2016-2020 5-year ACS
```

```
# A tibble: 133 x 6
   GEOID NAME                          B19013_001E B19013_001M B01002_001E B0100~1
   <chr> <chr>                               <dbl>       <dbl>       <dbl>   <dbl>
 1 51001 Accomack County, Virginia           46178        2575        45.8     0.5
 2 51005 Alleghany County, Virginia          48513        4275        48       0.6
 3 51007 Amelia County, Virginia             63918        6276        45       1.8
 4 51011 Appomattox County, Virginia         55457        5246        42.7     1.2
 5 51013 Arlington County, Virginia         122604        2627        34.8     0.2
 6 51017 Bath County, Virginia               55481       15306        48.9     7.1
 7 51019 Bedford County, Virginia            67136        3303        46.6     0.4
 8 51023 Botetourt County, Virginia          72719        4231        47.4     0.5
 9 51025 Brunswick County, Virginia          46111        3642        44.6     0.7
10 51029 Buckingham County, Virginia         48603        4866        42.9     1
# ... with 123 more rows, and abbreviated variable name 1: B01002_001M
```

```r
#we can also save a series of variables like this, to make our lives easier
myvars <- c(totalpop = "B01003_001",
            medincome = "B19013_001",
            medage = "B01002_001"
)
```

```r
#watch what happens now - note the column names that persist...
va_counties <- get_acs(geography = "county",
                       variables = c(myvars),
                       state = "VA",
                       output = "wide")
```

```
Getting data from the 2016-2020 5-year ACS
```

```r
#remove MOE columns - they all end with "M"
va_counties <- va_counties %>%
  select(-ends_with("M"))

va_counties
```

```
# A tibble: 133 x 5
   GEOID NAME                           totalpopE medincomeE medageE
   <chr> <chr>                              <dbl>      <dbl>   <dbl>
 1 51001 Accomack County, Virginia          32560      46178    45.8
 2 51005 Alleghany County, Virginia         15030      48513    48
 3 51007 Amelia County, Virginia            12970      63918    45
 4 51011 Appomattox County, Virginia        15814      55457    42.7
 5 51013 Arlington County, Virginia        236434     122604    34.8
 6 51017 Bath County, Virginia               4248      55481    48.9
 7 51019 Bedford County, Virginia           78965      67136    46.6
 8 51023 Botetourt County, Virginia         33440      72719    47.4
 9 51025 Brunswick County, Virginia         16336      46111    44.6
10 51029 Buckingham County, Virginia        17087      48603    42.9
# ... with 123 more rows
```

```r
#remove that trailing "E"
colnames(va_counties) <- sub("E$", "", colnames(va_counties)) # $ means end of string only

va_counties
```

```
# A tibble: 133 x 5
   GEOID NAM                            totalpop medincome medage
   <chr> <chr>                             <dbl>     <dbl>  <dbl>
 1 51001 Accomack County, Virginia         32560     46178   45.8
 2 51005 Alleghany County, Virginia        15030     48513   48
 3 51007 Amelia County, Virginia           12970     63918   45
 4 51011 Appomattox County, Virginia       15814     55457   42.7
 5 51013 Arlington County, Virginia       236434    122604   34.8
 6 51017 Bath County, Virginia              4248     55481   48.9
 7 51019 Bedford County, Virginia          78965     67136   46.6
 8 51023 Botetourt County, Virginia        33440     72719   47.4
 9 51025 Brunswick County, Virginia        16336     46111   44.6
10 51029 Buckingham County, Virginia       17087     48603   42.9
# ... with 123 more rows
```

```r
### what if we want the mapping/geospatial boundaries too?
va_counties_withgeo <- get_acs(geography = "county",
                          variables = c(myvars),
                          state = "VA",
                          output = "wide",
                          geometry = TRUE)
```

```
Getting data from the 2016-2020 5-year ACS
```

```
Downloading feature geometry from the Census website.  To cache shapefiles for use in future
```

```
  |
  |                                                                      |   0%
  |
  |                                                                      |   1%
  |
  |=                                                                     |   1%
  |
  |=                                                                     |   2%
  |
  |==                                                                    |   2%
  |
  |==                                                                    |   3%
  |
  |===                                                                   |   4%
  |
  |====                                                                  |   5%
  |
  |====                                                                  |   6%
  |
  |=====                                                                 |   7%
  |
  |======                                                                |   9%
  |
  |=======                                                               |  10%
  |
  |=======                                                               |  11%
  |
  |========                                                              |  11%
  |
```

```
|=========                                                              |  13%
|
|==========                                                             |  14%
|
|==========                                                             |  15%
|
|===============                                                        |  21%
|
|===============                                                        |  22%
|
|====================                                                   |  28%
|
|====================                                                   |  29%
|
|=====================                                                  |  30%
|
|=========================                                              |  36%
|
|============================                                           |  40%
|
|============================                                           |  41%
|
|==================================                                     |  53%
|
|====================================                                   |  56%
|
|============================================                           |  67%
|
|=============================================                          |  70%
|
|=============================================                          |  71%
|
|==============================================                         |  72%
|
|==============================================                         |  73%
|
|================================================                       |  78%
|
|=================================================                      |  81%
|
|==================================================                     |  82%
|
|========================================================               |  83%
```

```
  |
  |=============================================================          |  84%
  |
  |==============================================================         |  85%
  |
  |===============================================================        |  86%
  |
  |=================================================================      |  88%
  |
  |====================================================================   |  90%
  |
  |======================================================================= |  92%
  |
  |========================================================================|  93%
  |
  |========================================================================|  94%
  |
  |========================================================================|  94%
  |
  |========================================================================|  95%
  |
  |========================================================================|  95%
  |
  |========================================================================|  96%
  |
  |========================================================================| |  98%
  |
  |========================================================================|| 100%
```

```
va_counties_withgeo
```

```
Simple feature collection with 133 features and 8 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -83.67539 ymin: 36.54074 xmax: -75.24247 ymax: 39.46601
Geodetic CRS:  NAD83
First 10 features:
   GEOID                         NAME totalpopE totalpopM medincomeE
1  51035      Carroll County, Virginia     29911        NA      44518
2  51089        Henry County, Virginia     51032        NA      38511
3  51015      Augusta County, Virginia     75754        NA      65076
```

```
4  51143 Pittsylvania County, Virginia     60867        NA      49520
5  51175  Southampton County, Virginia     17829        NA      63034
6  51185     Tazewell County, Virginia     41201        NA      42207
7  51117  Mecklenburg County, Virginia     30726        NA      50224
8  51091     Highland County, Virginia      2202        NA      51831
9  51041 Chesterfield County, Virginia    348500        NA      84645
10 51047     Culpeper County, Virginia     51935        NA      80663
   medincomeM medageE medageM                       geometry
1        3022    48.3     0.6 MULTIPOLYGON (((-81.03406 3...
2        2064    48.1     0.4 MULTIPOLYGON (((-80.09514 3...
3        3262    45.1     0.3 MULTIPOLYGON (((-79.53273 3...
4        2158    47.6     0.4 MULTIPOLYGON (((-79.71235 3...
5        5660    46.7     0.7 MULTIPOLYGON (((-77.50192 3...
6        3080    45.3     0.3 MULTIPOLYGON (((-81.90089 3...
7        2884    48.5     0.4 MULTIPOLYGON (((-78.74027 3...
8        3904    59.5     4.8 MULTIPOLYGON (((-79.81015 3...
9        1549    39.0     0.2 MULTIPOLYGON (((-77.87823 3...
10       3922    39.8     0.7 MULTIPOLYGON (((-78.22915 3...
```

```r
#all counties in the US?
all_counties_withgeo <- get_acs(geography = "county",
                        variables = c(myvars),
                        output = "wide",
                        geometry = TRUE)
```

```
Getting data from the 2016-2020 5-year ACS

Downloading feature geometry from the Census website.  To cache shapefiles for use in future
```

```r
all_counties_withgeo
```

```
Simple feature collection with 3221 features and 8 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -179.1489 ymin: 17.88328 xmax: 179.7785 ymax: 71.36516
Geodetic CRS:  NAD83
First 10 features:
   GEOID                          NAME totalpopE totalpopM medincomeE
1  01053       Escambia County, Alabama     36775        NA      35558
2  01129     Washington County, Alabama     16336        NA      42331
```

```
3  01113        Russell County, Alabama     57938        NA      42208
4  01107        Pickens County, Alabama     20049        NA      40362
5  01119         Sumter County, Alabama     12595        NA      26150
6  04027           Yuma County, Arizona    211931        NA      48790
7  04001         Apache County, Arizona     71714        NA      33967
8  04017         Navajo County, Arizona    110271        NA      43140
9  05131     Sebastian County, Arkansas    127670        NA      47878
10 06037 Los Angeles County, California  10040682        NA      71358
   medincomeM medageE medageM                       geometry
1        2902    39.4     0.6 MULTIPOLYGON (((-87.61558 3...
2        4919    43.8     1.3 MULTIPOLYGON (((-88.46443 3...
3        2255    36.7     0.5 MULTIPOLYGON (((-85.43472 3...
4        2740    42.5     0.5 MULTIPOLYGON (((-88.34043 3...
5        2033    35.7     0.3 MULTIPOLYGON (((-88.42082 3...
6        1981    34.8     0.1 MULTIPOLYGON (((-114.8163 3...
7        2108    35.4     0.2 MULTIPOLYGON (((-110.0007 3...
8        2357    38.2     0.3 MULTIPOLYGON (((-110.7507 3...
9        2067    37.9     0.3 MULTIPOLYGON (((-94.4476 34...
10        336    36.7     0.1 MULTIPOLYGON (((-118.6044 3...
```

Making a detailed county-level demographics table

```
myvars <- c(totalpop = "B01003_001",
            medincome = "B19013_001",
            medage = "B01002_001",
            natborn_total = "B05012_001",
            natborn_foreign = "B05012_003",
            military_total = "B21001_001",
            military_veteran = "B21001_002",
            originrace_total_all = "B03002_001",
            originrace_whitealone = "B03002_003",
            education_total = "B06009_001",
            education_bachelors = "B06009_005",
            education_gradprofess = "B06009_006")
```

```
#Getting data for every state
allcounties_wide <- get_acs(geography = "county",
                            variables = c(myvars),
                            output = "wide",
                            geometry = TRUE)
```

```
Getting data from the 2016-2020 5-year ACS

Downloading feature geometry from the Census website.  To cache shapefiles for use in future
```

```
#remove MOE columns - they all end with "M"
allcounties_wide <- allcounties_wide %>%
  select(-ends_with("M"))

names(allcounties_wide)
```

```
 [1] "GEOID"                  "NAME"                   "totalpopE"
 [4] "medincomeE"             "medageE"                "natborn_totalE"
 [7] "natborn_foreignE"       "military_totalE"        "military_veteranE"
[10] "originrace_total_allE"  "originrace_whitealoneE" "education_totalE"
[13] "education_bachelorsE"   "education_gradprofessE" "geometry"
```

```
# #cleaning up and splitting NAME into component parts
allcounties_wide <- allcounties_wide %>%
  mutate(
    county_name = str_split(NAME, ",", simplify = TRUE)[ ,1],
    state_name = str_split(NAME, ",", simplify = TRUE)[ ,2],
    state_name = str_trim(state_name)
  )
```

```
glimpse(allcounties_wide)
```

```
Rows: 3,221
Columns: 17
$ GEOID                 <chr> "01053", "01129", "01113", "01107", "01119", "0~
$ NAME                  <chr> "Escambia County, Alabama", "Washington County,~
$ totalpopE             <dbl> 36775, 16336, 57938, 20049, 12595, 211931, 7171~
$ medincomeE            <dbl> 35558, 42331, 42208, 40362, 26150, 48790, 33967~
$ medageE               <dbl> 39.4, 43.8, 36.7, 42.5, 35.7, 34.8, 35.4, 38.2,~
$ natborn_totalE        <dbl> 36775, 16336, 57938, 20049, 12595, 211931, 7171~
$ natborn_foreignE      <dbl> 308, 217, 1815, 745, 243, 56611, 992, 2867, 110~
$ military_totalE       <dbl> 28536, 12857, 42683, 16076, 10161, 154574, 5247~
$ military_veteranE     <dbl> 2152, 865, 6300, 1202, 457, 15553, 3206, 7254, ~
$ originrace_total_allE <dbl> 36775, 16336, 57938, 20049, 12595, 211931, 7171~
$ originrace_whitealoneE <dbl> 22091, 10653, 26096, 10752, 3131, 64843, 12993,~
```

```
$ education_totalE       <dbl> 25703, 11148, 39280, 14386, 7657, 134749, 45607~
$ education_bachelorsE   <dbl> 2252, 700, 4416, 1385, 761, 13574, 3496, 6898, ~
$ education_gradprofessE <dbl> 961, 907, 2455, 690, 649, 7225, 2209, 4690, 701~
$ geometry               <MULTIPOLYGON [°]> MULTIPOLYGON (((-87.61558 3..., MU~
$ county_name            <chr> "Escambia County", "Washington County", "Russel~
$ state_name             <chr> "Alabama", "Alabama", "Alabama", "Alabama", "Al~
```

```r
#bring new columns forward
allcounties_wide <- allcounties_wide %>%
  select(GEOID,
         state_name,
         county_name,
         everything(),
         -NAME)

glimpse(allcounties_wide)
```

```
Rows: 3,221
Columns: 16
$ GEOID                  <chr> "01053", "01129", "01113", "01107", "01119", "0~
$ state_name             <chr> "Alabama", "Alabama", "Alabama", "Alabama", "Al~
$ county_name            <chr> "Escambia County", "Washington County", "Russel~
$ totalpopE              <dbl> 36775, 16336, 57938, 20049, 12595, 211931, 7171~
$ medincomeE             <dbl> 35558, 42331, 42208, 40362, 26150, 48790, 33967~
$ medageE                <dbl> 39.4, 43.8, 36.7, 42.5, 35.7, 34.8, 35.4, 38.2,~
$ natborn_totalE         <dbl> 36775, 16336, 57938, 20049, 12595, 211931, 7171~
$ natborn_foreignE       <dbl> 308, 217, 1815, 745, 243, 56611, 992, 2867, 110~
$ military_totalE        <dbl> 28536, 12857, 42683, 16076, 10161, 154574, 5247~
$ military_veteranE      <dbl> 2152, 865, 6300, 1202, 457, 15553, 3206, 7254, ~
$ originrace_total_allE  <dbl> 36775, 16336, 57938, 20049, 12595, 211931, 7171~
$ originrace_whitealoneE <dbl> 22091, 10653, 26096, 10752, 3131, 64843, 12993,~
$ education_totalE       <dbl> 25703, 11148, 39280, 14386, 7657, 134749, 45607~
$ education_bachelorsE   <dbl> 2252, 700, 4416, 1385, 761, 13574, 3496, 6898, ~
$ education_gradprofessE <dbl> 961, 907, 2455, 690, 649, 7225, 2209, 4690, 701~
$ geometry               <MULTIPOLYGON [°]> MULTIPOLYGON (((-87.61558 3..., MU~
```

```r
#clean up column names to remove trailing E
#we do this here and not above to avoid losing the E in NAME until it's split and discarde
colnames(allcounties_wide) <- sub("E$", "", colnames(allcounties_wide)) # $ means end of s

names(allcounties_wide)
```

```
 [1] "GEOID"                "state_name"           "county_name"
 [4] "totalpop"             "medincome"            "medage"
 [7] "natborn_total"        "natborn_foreign"      "military_total"
[10] "military_veteran"     "originrace_total_all" "originrace_whitealone"
[13] "education_total"      "education_bachelors"  "education_gradprofess"
[16] "geometry"
```

```
#percentage calculations
#-- tricky, since demo groups differ in columns
#-- this might have to be done individually for each demographic grouping

allcounties_wide <- allcounties_wide %>%
  mutate(
    pct_born_foreign = round_half_up(natborn_foreign / natborn_total * 100, 2),
    pct_mil_veteran = round_half_up(military_veteran / military_total * 100, 2),
    pct_race_white = round_half_up(originrace_whitealone / originrace_total_all * 100, 2),
    pct_race_nonwhite = 100 - pct_race_white,
    pct_ed_college_all = round_half_up((education_bachelors + education_gradprofess) / edu
  )

#remove unneeded columns
allcounties_wide <- allcounties_wide %>%
  select(-natborn_total,
         -natborn_foreign,
         -education_total,
         -education_bachelors,
         -education_gradprofess,
         -military_total,
         -military_veteran,
         -originrace_total_all,
         -originrace_whitealone
  )
```

```
# save results for next steps
saveRDS(allcounties_wide, here("data", "allcounties_wide_demographics.rds"))
```