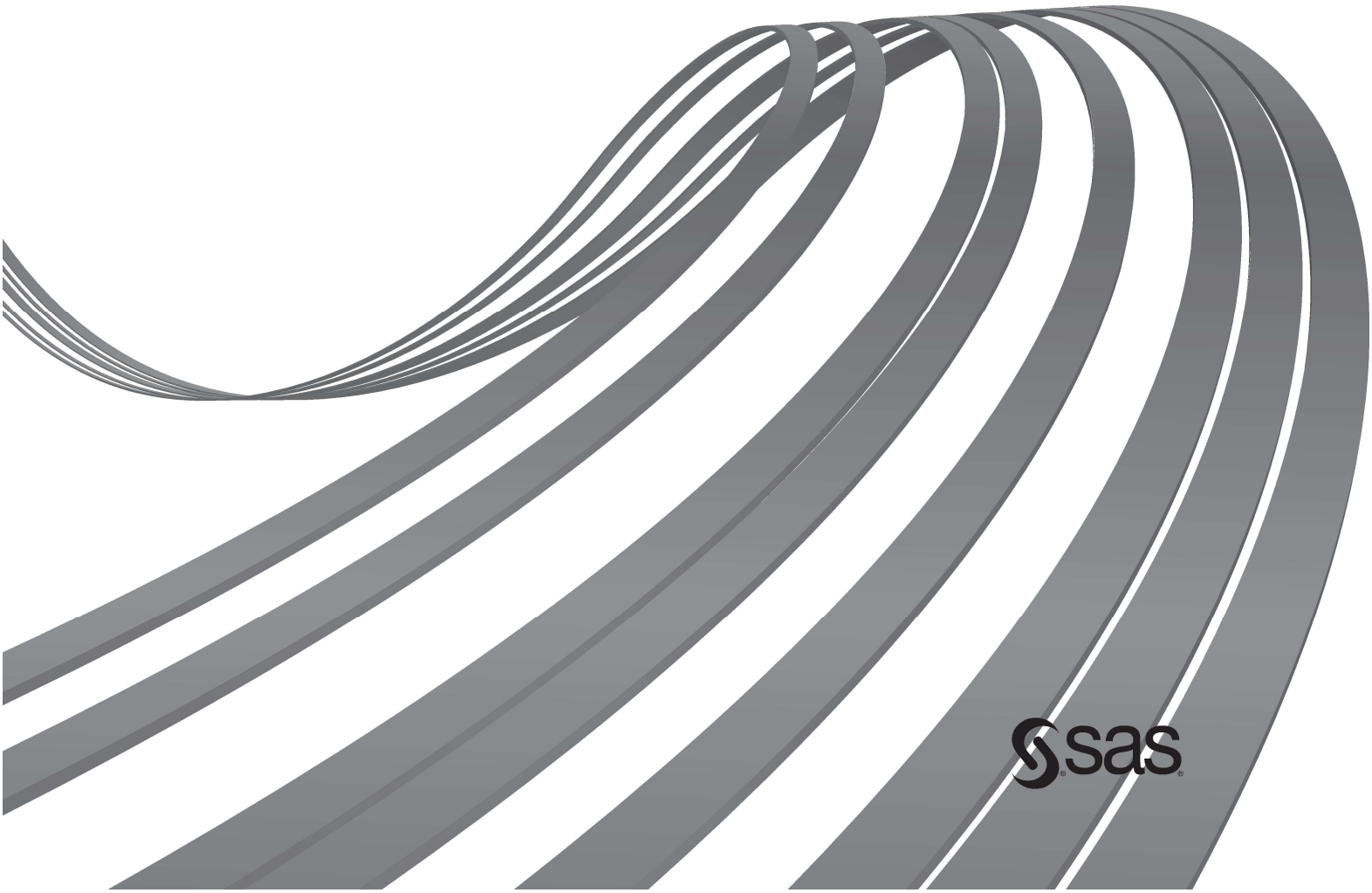
sas innovate

SAS®, SQL, R, and Python: We're All Friends

Lincoln H. Groves
Senior Manager, Analytical Education



Contents

SAS On-the-Job | Part 1

Starting with a Story

Role: Data Analyst within the Department of Health and Human Services

The coronavirus outbreak, or COVID-19, simultaneously created a health and economic crisis in the United States. As local regions locked down to slow the spread of the disease, millions of jobs were lost across the United States. Most schools also shuttered their doors and switched to remote learning. This meant that many working parents who retained their jobs found themselves as both remote workers and teacher's assistants during the lockdowns.

The impacts of COVID-19 have been very uneven, especially for mothers. A focus of public policy for decades has been to find a way to get mothers back into the labor force, while supporting them as they balance work and home responsibilities. In this SAS On-the-Job activity, we explore the pre- and post-COVID-19 trends in unemployment (UE) and labor force participation (LFP) rates among U.S. women aged 25 to 54.

More Details on your Job

You are a new Public Policy Analyst within the Department of Health and Human Services (HHS). You are placed into a department interested in public policies that promote labor force participation (LFP) by women – particularly low-income and less-educated women. Labor force participation simply means that women are engaged in the labor market - either they have a job or are looking for one. We'll also explore the more familiar unemployment rate, which captures women who would like a job, but don't currently have one.

During the height of the pandemic, a host of media articles highlighted the disproportionate impact of COVID-19 on women's employment. HHS leadership is interested in knowing whether these impacts are still being felt today - years after the official declaration of the pandemic. If yes, HHS leadership would consider enacting new, target policy response to support this potentially vulnerable group of workers. However, before considering policy responses, HHS leadership would like someone to examine whether there are lingering effects of COVID-19 on the current level of labor supply by women. And this someone is you! Drumroll, please!

As a new analyst, your goals in this set of exercises are to

- access the Jupyter notebooks created by your predecessor
- understand what the previous analyst did - particularly with the integration of SAS, SQL, Python, and R
- modify existing code (as appropriate).

And you'll be guided in this exercise by a very helpful onboarding buddy. Which is me. Your humble narrator.

Let's get started!

What We Cover

Beyond allowing you to play policy analyst for the day, this SAS On-the-Job (OTJ) provides an overview of SAS, SQL, Python, and R integration in Jupyter. More specifically, this SAS OTJ is an act in three parts, with the following flow:

(1) Start in a Python notebook - because the original coder loved Python. Perform a preliminary analysis of the data. Then use SQL to collapse the data - because it's a much more effective way to aggregate data than other approaches in Python. And finally, we'll ensure that the underlying data make sense.

(2) The second Jupyter notebook focuses on SAS code. We'll leverage some SAS code and macros to explore aggregated U.S. trends. We'll conclude by creating a state-level data set that we can then use to map trends in R (confession: R can produce some great maps!).

(3) The last notebook focuses on R code. We'll plot state-level trends over time for unemployment and labor force participation.

Python, SAS, and SQL Integration

American Community Survey (ACS) Data Import

Welcome to Day 1 as Public Policy Analyst at HHS. In your exploration of labor supply by women during COVID, the first thing we need to do is to get our data loaded into the environment and running. Toward that goal, let's first connect with the SAS Viya server - where we will run our entire analysis. After making this connection, we'll then use the "SASPy" Python package to establish a connection between SAS and Python - and run some SAS code within the Python kernel to load the American Community Survey Data.

And why start with the SAS code? Well, this project has used hackers from all backgrounds in the analysis - and the previous analyst was a big fan of using SAS to read in their data. But, know that reading in the data via Python would have been perfectly acceptable too!

In [10]:
```
import saspy

# Create a SAS session object
sas_session = saspy.SASsession()
```

```
Using SAS Config named: eduviya

SAS server started using Context SAS Studio compute context with
SESSION_ID=10d7040a-f841-4847-9471-e655a688b8a4-ses0000
```

With the connection established, we can now use the submit method (https://sassoftware.github.io/saspy/api.html#saspy.SASsession.submit) to run SAS code from Jupyter using a Python kernel. Notice how we use the LIBNAME statement to save the ACS file to disk - and while pulling the original data from GitHub. Moreover, we create a Python data frame at the end, so that we can forge on with the Python portion of the analysis.

```
results_dict = sas_session.submitLST(
              """
                 libname acs "/home/student/SIWWAF_SAS_SQL_R_Python";

                 filename acs_url url 'https://raw.githubusercontent.com/lincolngroves/SAS-OTJ-HHS/main/ACS_2015_2022_ltd.csv';

                    data acs.acs_2015_2022    ;
                         %let _EFIERR_ = 0; /* set the ERROR detection macro variable */

                         infile acs_url delimiter = ',' MISSOVER DSD  firstobs=2 ;

                         informat State_FIP best32. ;
                         informat State_Name $20. ;
                         informat YearQuarter YYQ6. ;
                         informat Race_Ethnic $21. ;
                         informat EDUC_LTD $19. ;
                         informat Child_Status $14. ;
                         informat Unemp best32. ;
                         informat in_LF best32. ;
                         informat WTFINL best32. ;

                         format State_FIP best12. ;
                         format State_Name $20. ;
                         format YearQuarter YYQ6. ;
                         format Race_Ethnic $21. ;
                         format EDUC_LTD $19. ;
                         format Child_Status $14. ;
                         format Unemp best12. ;
                         format in_LF best12. ;
                         format WTFINL best12. ;

                         input
                             State_FIP
                             State_Name  $
                             YearQuarter
                             Race_Ethnic  $
                             EDUC_LTD  $
                             Child_Status  $
                             Unemp
                             in_LF
                             WTFINL
                         ;
                         if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR detection macro variable */

                         run;
               """,
             )

acs_df = sas_session.sd2df(table="acs_2015_2022",libref="acs")
```

```
12   ods listing close;ods html5 (id=saspy_internal) options(bitmap_mode='inline') device=svg style=HTMLBlue; ods graphics on /
12 ! outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: sashtml.htm
13
14
15                  libname acs "/home/student/SIWWAF_SAS_SQL_R_Python";
NOTE: Libref ACS was successfully assigned as follows:
      Engine:        V9
      Physical Name: /home/student/SIWWAF_SAS_SQL_R_Python
16
17                  filename acs_url url 'https://raw.githubusercontent.com/lincolngroves/SAS-OTJ-HHS/main/ACS_2015_2022_ltd.csv';
18
19                     data acs.acs_2015_2022    ;
20                          %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
21
22                          infile acs_url delimiter = ',' MISSOVER DSD  firstobs=2 ;
23
24                          informat State_FIP best32. ;
25                          informat State_Name $20. ;
26                          informat YearQuarter YYQ6. ;
27                          informat Race_Ethnic $21. ;
28                          informat EDUC_LTD $19. ;
29                          informat Child_Status $14. ;
30                          informat Unemp best32. ;
31                          informat in_LF best32. ;
32                          informat WTFINL best32. ;
33
34                          format State_FIP best12. ;
35                          format State_Name $20. ;
36                          format YearQuarter YYQ6. ;
37                          format Race_Ethnic $21. ;
38                          format EDUC_LTD $19. ;
39                          format Child_Status $14. ;
40                          format Unemp best12. ;
41                          format in_LF best12. ;
42                          format WTFINL best12. ;
43
44                          input
45                              State_FIP
46                              State_Name  $
47                              YearQuarter
48                              Race_Ethnic  $
49                              EDUC_LTD  $
50                              Child_Status  $
51                              Unemp
52                              in_LF
53                              WTFINL
54                          ;
55                          if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR detection macro variable */
56
57                          run;
```

```
NOTE: The infile ACS_URL is:
      Filename=https://raw.githubusercontent.com/lincolngroves/SAS-OTJ-HHS/main/ACS_2015_2022_ltd.csv,
      Local Host Name=sas-compute-server-12ade943-9069-408b-a691-fd4723a36199-128,
      Local Host IP addr=10.42.5.35,
      Service Hostname Name=cdn-185-199-109-133.github.com,
      Service IP addr=185.199.109.133,
      Service Name=N/A,Service Portno=443,
      Lrecl=32767,Recfm=Variable

NOTE: 141515 records were read from the infile ACS_URL.
      The minimum record length was 50.
      The maximum record length was 102.
NOTE: The data set ACS.ACS_2015_2022 has 141515 observations and 9 variables.
NOTE: DATA statement used (Total process time):
      real time           1.69 seconds
      user cpu time       0.46 seconds
      system cpu time     0.22 seconds
      memory              1493.15k
      OS Memory           18532.00k
      Timestamp           02/28/2024 03:27:35 PM
      Step Count                        3  Switch Count  49
      Page Faults                       3
      Page Reclaims                     1011
      Page Swaps                        0
      Voluntary Context Switches        565
      Involuntary Context Switches      168
      Block Input Operations            1096
      Block Output Operations           33288


58
59   ods html5 (id=saspy_internal) close;ods listing;
60
```

Examine and Explore Our Data Frame

Alright - the data are now accessible in the notebook. What's typically next as an analyst? Well, let's examine the underlying data to become more familiar with the data and distribution of values!

In [12]:
```
# One Simple line will pull up the Python dataframe
acs_df
```

Out[12]:

	State_FIP	State_Name	YearQuarter	Race_Ethnic	EDUC_LTD	Child_Status	Unemp	in_LF	WTFINL
0	1.0	Alabama	2015-01-01	Black, Non-Hispanic	< HS	No Children	0.0	0.0	8709.3981
1	1.0	Alabama	2015-01-01	Black, Non-Hispanic	< HS	No Children	0.0	1.0	17350.5674
2	1.0	Alabama	2015-01-01	Black, Non-Hispanic	< HS	Older Children	0.0	0.0	13817.5321
3	1.0	Alabama	2015-01-01	Black, Non-Hispanic	< HS	Older Children	0.0	1.0	28839.7823
4	1.0	Alabama	2015-01-01	Black, Non-Hispanic	< HS	Older Children	1.0	1.0	5959.0752
...	...	...	...	...	...	...	...	...	...
141510	56.0	Wyoming	2022-10-01	All Other	Some College	No Children	0.0	1.0	4451.7602
141511	56.0	Wyoming	2022-10-01	All Other	Some College	Older Children	0.0	1.0	575.1048
141512	56.0	Wyoming	2022-10-01	All Other	College +	No Children	0.0	1.0	619.0565
141513	56.0	Wyoming	2022-10-01	All Other	College +	Older Children	0.0	1.0	618.7571
141514	56.0	Wyoming	2022-10-01	All Other	College +	Child < 5	0.0	1.0	2983.2854

141515 rows × 9 columns

From the data above, we can see that we'll have nine variables in our analysis. The current unit of analysis in this data is state, year-quarter, race-ethnic, and child-status. In simple English, this means that a row - that is, the observation - contains the unemployment rate and labor force participation rate for women of a particular race-ethnic group, education level, and child status - for a given state at a point in time. Yup, that's a mouthful, but still very important. Moreover, note that **Unemp** and **in_LF** are rounded in the window above - but are not in the underlying data. Finally, **WTFINL** is a weighting variable that represents the number of women represented in that row. So, 8709.3981 can be simply interpreted as ~8709 women. This variable will be very important when we aggregate the data.

Finally, **FIPS** are Federal Information Processing Standards, so the **State_Fip** is just a unique variable for that state. And it will help in the upcoming merges. Finally, **State** is just, well, a U.S. state.

With data now uploaded in Python, let's load some Python packages so that we can do some cooler things with the data!

In [13]:
```
# Classic Python packages that will make our Exploratory Data Analysis easier
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Summarize the Data

Let's summarize the data in Python - so that we can better understand the distribution of the numeric variables. As a wise person might or might not have stated: *gotta know thy data!*

In [14]:
```
# Perform the PROC MEANS equivalent in Python
summary = acs_df.describe()
summary = summary.round(2)  # Set the decimal places to 2

# Print the summary statistics
print(summary)
```

```
       State_FIP      Unemp      in_LF      WTFINL
count  141515.00  141515.00  141515.00   141515.00
mean       28.46       0.18       0.62    43356.65
std        15.68       0.38       0.49   102299.40
min         1.00       0.00       0.00      123.30
25%        15.00       0.00       0.00     4059.88
50%        29.00       0.00       1.00    11803.34
75%        41.00       0.00       1.00    36636.89
max        56.00       1.00       1.00  2366665.82
```

We can see from **Unemp** and **in_LF** above that we do have a distribution of values - and that the average (unweighted) unemployment rate over this period is 18% and the average (unweighted) labor force participation rate is 62%. Ideally, we'd want to weight these values statistically to get a true U.S. average - but this is a fine sanity check for now. The primary takeaway is that the data appear valid and - fun fact - there are no missing values.

How do we know the latter? Well, the count is the same across the four variables - and it also matches the number of variables in the data set. (See log above.)

Understand Distribution of Select Variables

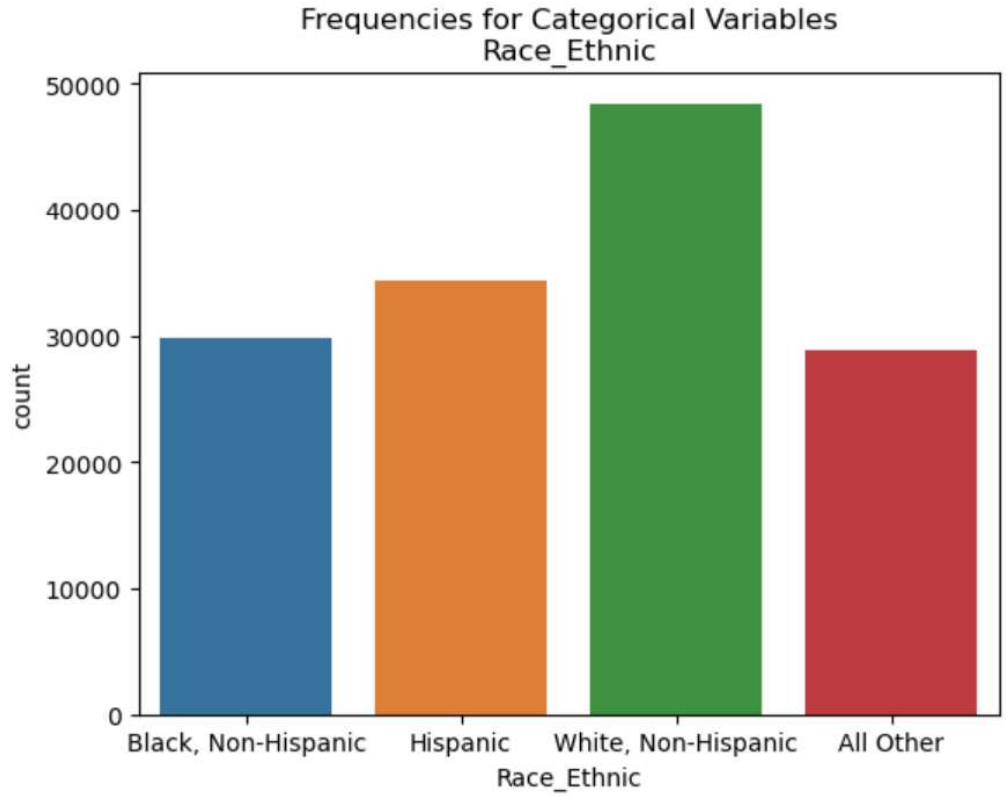
Data appear to be valid and clean. What a great way to start as an analyst... because - truth be told - this never happens.

Our next task is to examine the distribution of the categorical variables, via both tables and charts. For example, it would be useful to know how education, child status, and so on, are distributed in the data.

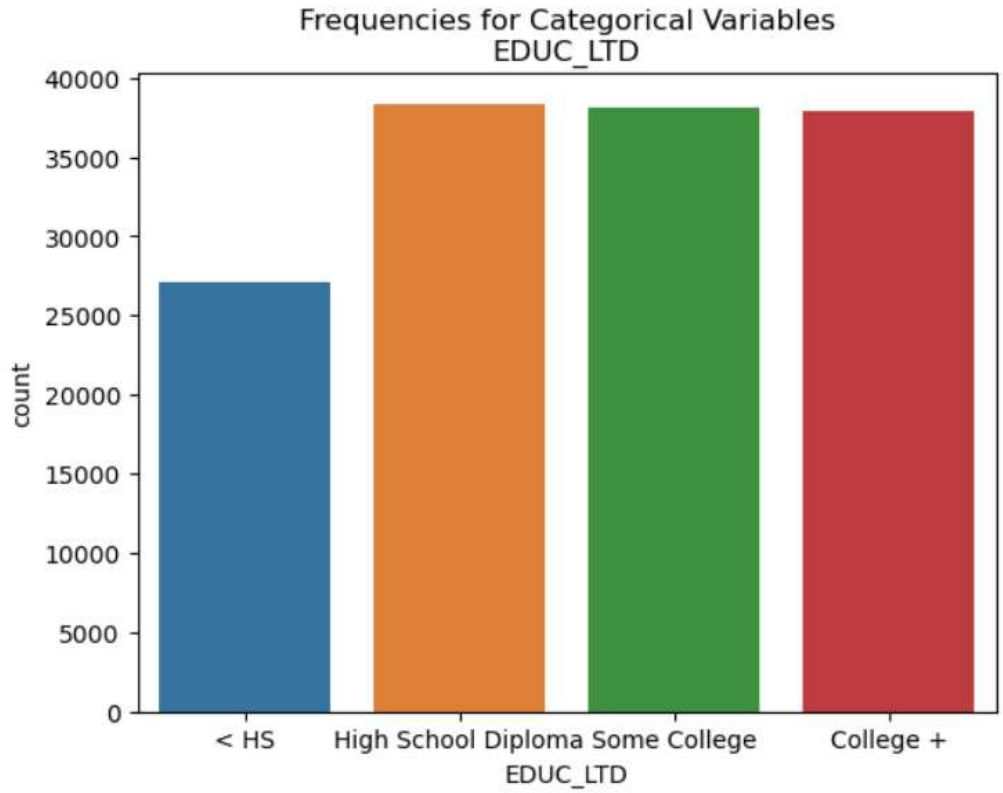
Let's use some classic Python procedures to do exactly that. We'll keep the analysis unweighted, which still allows us to examine the relative proportion of each variable in the analysis. This is perfectly fine test for this stage of the analysis. :)

In [15]:
```
# Set the title for the frequency analysis
title = "Frequencies for Categorical Variables"

# Perform frequency analysis
categorical_variables = ['Race_Ethnic', 'EDUC_LTD', 'Child_Status', 'Unemp', 'in_LF']
for variable in categorical_variables:
    freq_table = acs_df[variable].value_counts().reset_index()
    freq_table.columns = ['Value', 'Frequency']

    # Print the frequency table
    print(f"\n{title}")
    print(f"\n{variable}")
    print(freq_table)

    # Create frequency plot
    sns.countplot(data=acs_df, x=variable)
    plt.title(f"{title}\n{variable}")
    plt.show()
```

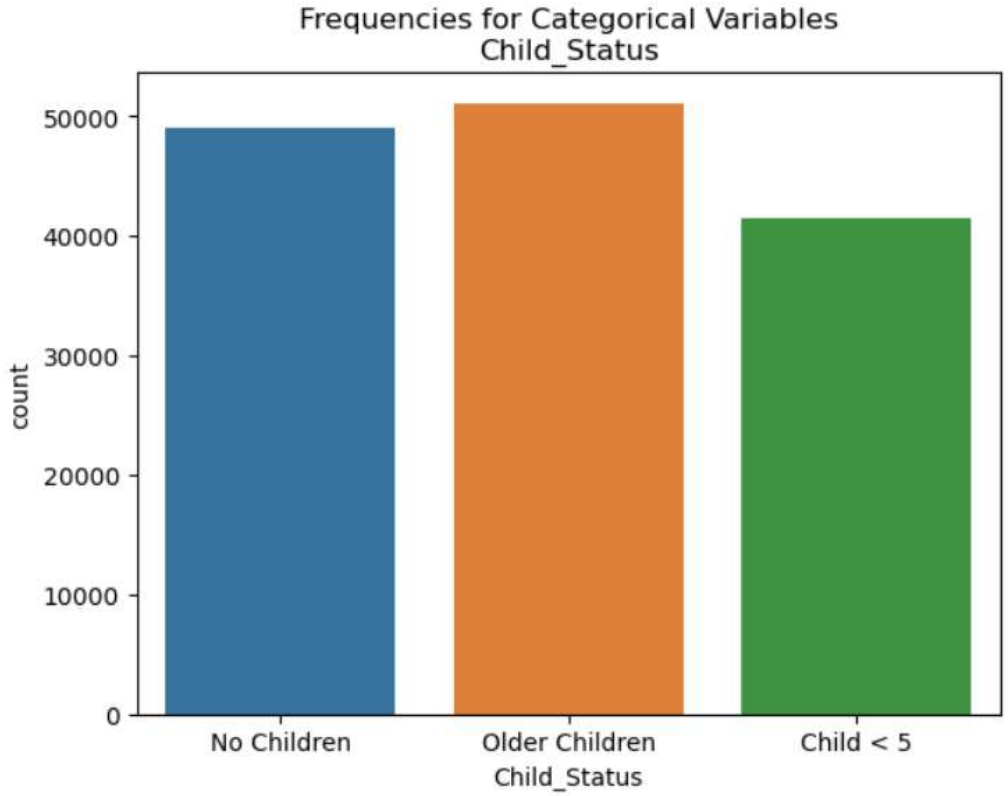
```
Frequencies for Categorical Variables

Race_Ethnic
                Value  Frequency
0  White, Non-Hispanic      48448
1             Hispanic      34399
2  Black, Non-Hispanic      29758
3            All Other      28910
```



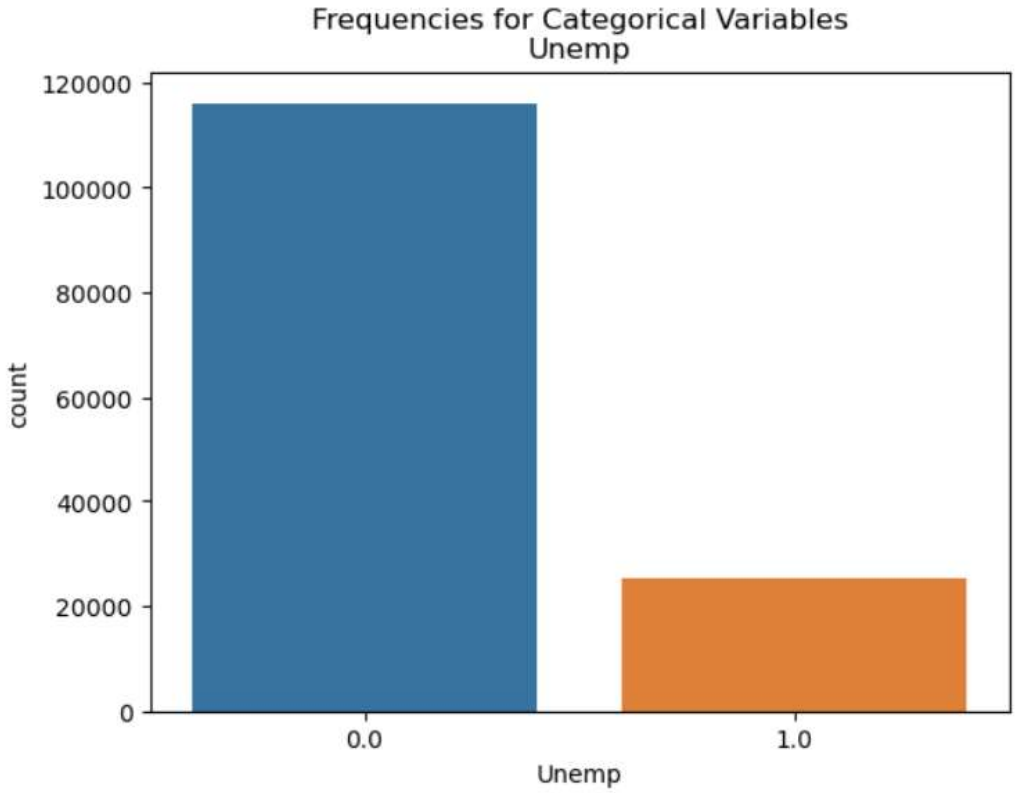
```
Frequencies for Categorical Variables

EDUC_LTD
                Value  Frequency
0  High School Diploma      38366
1         Some College      38104
2            College +      37932
3                 < HS      27113
```



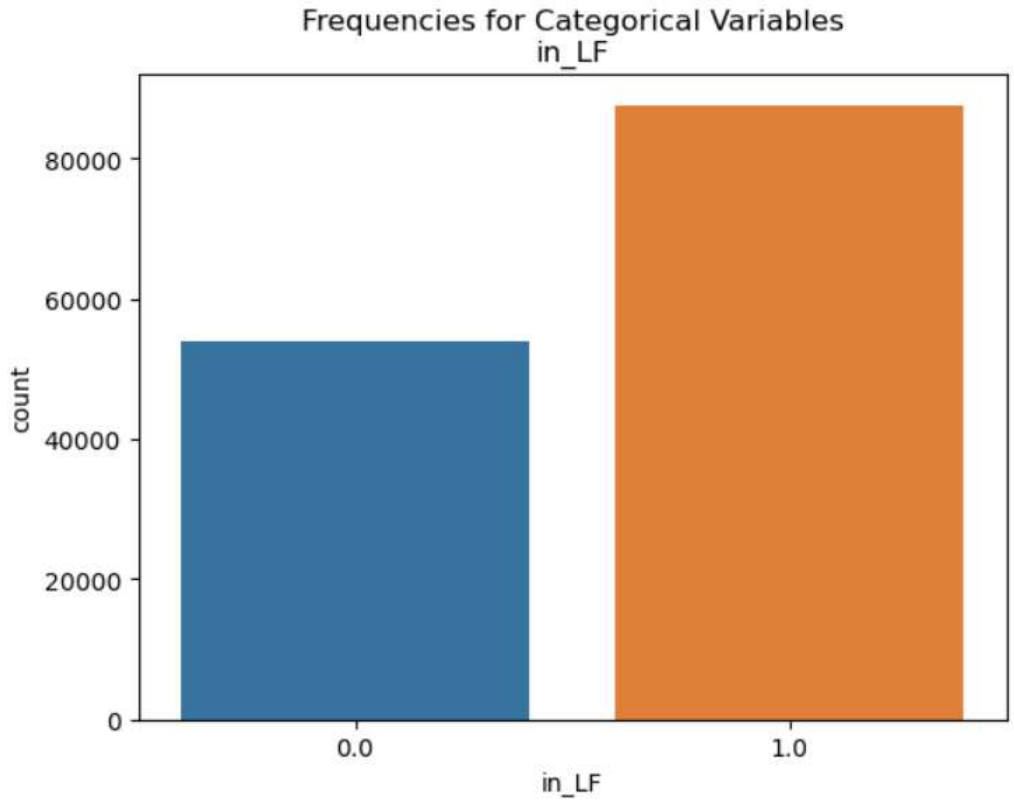
```
Frequencies for Categorical Variables

Child_Status
            Value  Frequency
0  Older Children      51117
1     No Children      48984
2       Child < 5      41414
```



```
Frequencies for Categorical Variables

Unemp
   Value  Frequency
0    0.0     116058
1    1.0      25457
```



```
Frequencies for Categorical Variables

in_LF
   Value  Frequency
0    1.0      87679
1    0.0      53836
```



Do you see any interesting trends in the underlying data? Again, these data aren't weighted, so they don't represent U.S. averages. But they're a useful statistic nonetheless.

My interesting findings: (1) White, Non-Hispanic are in, by far, the most states across the United States. (2) There are several states in the U.S. that we couldn't calculate estimates for individuals with less than a high school level of education, because sample sizes were either too small or not available. (3) Most women in the data set (a) had children, (b) were in the labor force, and (c) were employed.

Use SQL Code to Collapse Data to Get U.S. Estimates

As noted above, an observation in the data is a particular demographic group residing in a particular state in a given year. Because we have the weighed value for each cell, we can use SQL to aggregate the data to the country level and give us a more precise U.S. average over time. Ready to see the how elegantly SQL can collapse data? Well, I am!

```
results_dict = sas_session.submit(
    """
      libname acs  "/home/student/SIWWAF_SAS_SQL_R_Python";

       proc sql;
           create      table acs.covid_labor_supply_us as
           select      distinct
                    yearquarter ,

       /*****************************************************************  Labor Force Status | All  */
                    sum( ( unemp=1 ) * WTFINL )                                                                    / sum( ( in_LF=1 ) *   WTFINL )
                    sum( ( in_LF=1 ) * WTFINL )                                                                    / sum(                                WTF


       /*****************************************************************  Labor Force Status | By Race  */

                    /*****************************************************  Unemployment */
                    sum( ( race_ethnic="Black, Non-Hispanic" ) * ( unemp=1 ) * WTFINL )         / sum( ( race_ethnic="Black, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )   as UE_
                    sum( ( race_ethnic="Hispanic" ) * ( unemp=1 ) * WTFINL )                           / sum( ( race_ethnic="Hispanic" ) * ( in_LF=1 ) * WTFINL )
                    sum( ( race_ethnic="White, Non-Hispanic" ) * ( unemp=1 ) * WTFINL )         / sum( ( race_ethnic="White, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )   as UE_
                    sum( ( race_ethnic="All Other" ) * ( unemp=1 ) * WTFINL )                          / sum( ( race_ethnic="All Other" ) * ( in_LF=1 ) * WTFINL )

                    /*****************************************************  LFP */
                    sum( ( race_ethnic="Black, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )         / sum( ( race_ethnic="Black, Non-Hispanic" ) * WTFINL )
                    sum( ( race_ethnic="Hispanic" ) * ( in_LF=1 ) * WTFINL )                           / sum( ( race_ethnic="Hispanic" ) * WTFINL )
                    sum( ( race_ethnic="White, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )         / sum( ( race_ethnic="White, Non-Hispanic" ) * WTFINL )
                    sum( ( race_ethnic="All Other" ) * ( in_LF=1 ) * WTFINL )                          / sum( ( race_ethnic="All Other" ) * WTFINL )


       /*****************************************************************  Labor Force Status | By Education  */

                    /*****************************************************  Unemployment */
                    sum( ( educ_ltd="< HS" ) * ( unemp=1 ) * WTFINL )                                      / sum( ( educ_ltd="< HS" ) * ( in_LF=1 ) * WTFINL )
                    sum( ( educ_ltd="High School Diploma" ) * ( unemp=1 ) * WTFINL )            / sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )
                    sum( ( educ_ltd="Some College" ) * ( unemp=1 ) * WTFINL )                          / sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )
                    sum( ( educ_ltd="College +" ) * ( unemp=1 ) * WTFINL )                             / sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )

                    /*****************************************************  LFP */
                    sum( ( educ_ltd="< HS" ) * ( in_LF=1 ) * WTFINL )                                      / sum( ( educ_ltd="< HS" ) * WTFINL )
                    sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )            / sum( ( educ_ltd="High School Diploma" ) * WTFINL )
                    sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )                          / sum( ( educ_ltd="Some College" ) * WTFINL )
                    sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )                             / sum( ( educ_ltd="College +" ) * WTFINL )


       /*****************************************************************  Labor Force Status | By Child Status  */

                    /*****************************************************  Unemployment */
                    sum( ( child_status="No Children" ) * ( unemp=1 ) * WTFINL )                       / sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )
                    sum( ( child_status="Older Children" ) * ( unemp=1 ) * WTFINL )             / sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )
                    sum( ( child_status="Child < 5" ) * ( unemp=1 ) * WTFINL )                         / sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )

                    /*****************************************************  LFP */
                    sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )                       / sum( ( child_status="No Children" ) * WTFINL )
                    sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )             / sum( ( child_status="Older Children" ) * WTFINL )
                    sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )                         / sum( ( child_status="Child < 5" ) * WTFINL )
```

```
           from        acs.acs_2015_2022
           group       by 1
           order       by 1 ;
       quit;

    """,
)

covid_labor_supply_us_df = sas_session.sd2df(table="covid_labor_supply_us",libref="acs")
```

Note: The code above has been truncated to fit the page.

Lots of good stuff in that statement above. Note that we're collapsing to the **yearquarter** level - and incorporating the sample weight (that is, **WTFINL**) in the analysis. Notice how succinct that code is - which would take many traditional Python or SAS commands if we went that route instead.

Let's now summarize the data one more time, to ensure that we aggregated the data properly. One bad sign would be to see unemployment rate and labor force participation rates either less than 0 or greater than 1. Because those things shouldn't happen.

Sanity Check: Do the Data Make Sense?

Let's print the data just to ensure that our data seem reasonable. The Python code:

```
In [17]: # Print the Data
         covid_labor_supply_us_df
```

Out[17]:

	YearQuarter	UE_Women	LFP_Women	UE_BlackWomen	UE_HispanicWomen	UE_WhiteWomen	UE_OtherWomen	LFP_BlackWomen	LFP_HispanicWo
0	2015-01-01	0.047211	0.738310	0.087219	0.063629	0.034865	0.040917	0.769755	0.65
1	2015-04-01	0.045522	0.738932	0.075385	0.062764	0.034821	0.041100	0.770039	0.66
2	2015-07-01	0.048815	0.734344	0.077023	0.067676	0.038039	0.044745	0.766467	0.66
3	2015-10-01	0.042647	0.743679	0.074805	0.060602	0.030634	0.042491	0.766863	0.66
4	2016-01-01	0.044473	0.744559	0.083346	0.055370	0.032719	0.046326	0.758997	0.66
5	2016-04-01	0.041264	0.742937	0.064624	0.054351	0.031966	0.045198	0.760563	0.66
6	2016-07-01	0.046789	0.744580	0.073069	0.055856	0.036961	0.055377	0.773851	0.68
7	2016-10-01	0.040895	0.749199	0.068410	0.051406	0.031419	0.042302	0.782778	0.67
8	2017-01-01	0.041431	0.749183	0.071577	0.053231	0.032228	0.035445	0.767590	0.67
9	2017-04-01	0.038418	0.751382	0.064231	0.046475	0.030096	0.039340	0.779143	0.67
10	2017-07-01	0.041737	0.751989	0.067395	0.049934	0.034223	0.036773	0.788550	0.67
11	2017-10-01	0.034527	0.751999	0.060091	0.046555	0.025240	0.033678	0.784759	0.67
12	2018-01-01	0.036484	0.751639	0.066816	0.046814	0.026355	0.036118	0.779314	0.68
13	2018-04-01	0.032264	0.754139	0.047910	0.042908	0.025821	0.029685	0.778637	0.69
14	2018-07-01	0.036136	0.757496	0.055848	0.046951	0.028697	0.032393	0.786638	0.69
15	2018-10-01	0.031217	0.764470	0.050949	0.042892	0.023536	0.028238	0.789246	0.70
16	2019-01-01	0.034355	0.758455	0.052966	0.043908	0.026655	0.036996	0.783327	0.69
17	2019-04-01	0.029075	0.757178	0.044322	0.035683	0.023850	0.026770	0.785704	0.69
18	2019-07-01	0.034164	0.761716	0.047183	0.041776	0.029697	0.028835	0.782763	0.69
19	2019-10-01	0.029583	0.776167	0.046503	0.038723	0.022436	0.031692	0.799629	0.71
20	2020-01-01	0.032782	0.768584	0.049942	0.048650	0.023085	0.038160	0.785616	0.70
21	2020-04-01	0.119946	0.742715	0.141014	0.163660	0.100107	0.131798	0.753963	0.66
22	2020-07-01	0.083675	0.749040	0.111939	0.103573	0.068076	0.101604	0.760583	0.67
23	2020-10-01	0.057499	0.754180	0.080323	0.082113	0.043167	0.066333	0.771861	0.67
24	2021-01-01	0.058201	0.752554	0.084852	0.084390	0.043133	0.063640	0.772730	0.67
25	2021-04-01	0.051745	0.747654	0.081171	0.071486	0.037894	0.055946	0.780887	0.67
26	2021-07-01	0.049911	0.752037	0.077457	0.059001	0.040836	0.047518	0.776056	0.67
27	2021-10-01	0.035978	0.761384	0.058800	0.049634	0.026262	0.036320	0.772184	0.69
28	2022-01-01	0.034349	0.763540	0.052194	0.046943	0.026161	0.033603	0.788942	0.69
29	2022-04-01	0.029718	0.761229	0.049441	0.038964	0.022403	0.027092	0.792820	0.69
30	2022-07-01	0.033675	0.765952	0.057269	0.035262	0.026981	0.035544	0.792027	0.70
31	2022-10-01	0.030569	0.769284	0.054214	0.033785	0.023564	0.031758	0.789987	0.70

32 rows × 25 columns

Note: The output above has been truncated to fit the page.

Let's further probe the data:

In [18]:
```
# Summarize the Data
summary_us = covid_labor_supply_us_df.describe()
summary_us = summary_us.round(3)  # Set the decimal places to 3

# Print the summary statistics
print(summary_us)
```

```
       UE_Women  LFP_Women  UE_BlackWomen  UE_HispanicWomen  UE_WhiteWomen  \
count    32.000     32.000         32.000            32.000         32.000
mean      0.044      0.753          0.068             0.057          0.034
std       0.018      0.010          0.020             0.025          0.015
min       0.029      0.734          0.044             0.034          0.022
25%       0.034      0.747          0.053             0.044          0.026
50%       0.040      0.752          0.066             0.050          0.030
75%       0.047      0.761          0.077             0.061          0.035
max       0.120      0.776          0.141             0.164          0.100

       UE_OtherWomen  LFP_BlackWomen  LFP_HispanicWomen  LFP_WhiteWomen  \
count         32.000          32.000             32.000          32.000
mean           0.044           0.778              0.683           0.777
std            0.022           0.011              0.015           0.010
min            0.027           0.754              0.659           0.759
25%            0.033           0.770              0.671           0.771
50%            0.038           0.779              0.679           0.776
75%            0.045           0.786              0.695           0.783
max            0.132           0.800              0.712           0.801

       LFP_OtherWomen  ...  LFP_Women_LTHS  LFP_Women_HS  LFP_Women_SCollege  \
count          32.000  ...          32.000        32.000              32.000
mean            0.717  ...           0.506         0.676               0.766
std             0.018  ...           0.015         0.012               0.008
min             0.672  ...           0.474         0.637               0.750
25%             0.709  ...           0.494         0.670               0.761
50%             0.714  ...           0.507         0.675               0.766
75%             0.729  ...           0.514         0.686               0.772
max             0.749  ...           0.540         0.696               0.785

       LFP_Women_CollegeP  UE_Women_NoKids  UE_Women_OlderKids  \
count              32.000           32.000              32.000
mean                0.835            0.044               0.042
std                 0.009            0.018               0.017
min                 0.818            0.028               0.026
25%                 0.829            0.034               0.032
50%                 0.833            0.039               0.039
75%                 0.840            0.046               0.045
max                 0.855            0.122               0.116
```

	UE_Women_YoungKids	LFP_Women_NoKids	LFP_Women_OlderKids	\
count	32.000	32.000	32.000	
mean	0.047	0.787	0.761	
std	0.018	0.010	0.009	
min	0.030	0.769	0.745	
25%	0.038	0.782	0.756	
50%	0.044	0.787	0.759	
75%	0.052	0.794	0.767	
max	0.125	0.810	0.782	

	LFP_Women_YoungKids
count	32.000
mean	0.661
std	0.013
min	0.638
25%	0.653
50%	0.658
75%	0.670
max	0.693

[8 rows x 24 columns]

What do you see? My musings: (1) There are 32 observations in the aggregated data set. The math: 8 years * 4 quarters. (2) The (weighted) average unemployment rate is 0.044 - or 4.4%. This value is much lower than the 0.18 (18%) value reported above - which shows that weighting is consequential and likely higher in the smaller states. (3) We see that, in general, the unemployment rate decreases, and the labor force participation rate increases, as the level of education increases. All these nuggets make sense!

Part 1 Recap

Our project is off to a great start. We've imported the data, performed some preliminary data analysis using Python code, and collapsed the data using SQL. But, we really haven't answered any of the interesting questions for HHS Leadership: namely (1) what are the UE and LFP trends over time and (2) are the levels back to "normal" after a COVID adjustment period?

We'll tackle these questions better in Parts 2 and 3 of this analysis.

SAS On-the-Job | Part 2

The Story Continues

In Part 1, we did a nice job of better understanding the objective at hand - and our data. We used a variety of tools - including Python, SQL, and SAS - to perform an exploratory data analysis and now can explore data at both the state and aggregated U.S. analysis. The (data) world is our oyster... yay!

In this section, we'll take a deeper dive into the trends in unemployment and labor force participation for women aged 25-54 (that is, “prime-aged”) in the United States. In particular, this section has three objectives.

- Plot aggregated U.S. trends over time | by demographic groups | by quarter.
- Examine state-level unemployment rates and labor force participation rates | by year.
- Create data sets to be used in Part 3 of our analysis.

Note: The output in this section is very long. For this printout, we’ve included only a portion of the output to save trees. The full output can be found in the GitHub project repository here: https://github.com/lincolngroves/SAS-OTJ-HHS

Time Series Plots Using SAS GPLOT and SAS Macros

Task 1 | Examine U.S. Trends Over Time

Let's get right to it. With our first task, let’s use GPLOT to explore trends in unemployment rates and labor force participation rates in the U.S. In particular, let's look at the trends by race and ethnicity, education level, and child status. Oh, and you'll gain exposure to SAS macros, so you can see how macros greatly simplify the amount of repetitive code that needs to be written.

Assign the Library to Pull the Data

We'll again read and write data to our SAS Viya environment. So, let's assign a link to that location on our drive.

```
In [1]: libname acs "/home/student/SIWWAF_SAS_SQL_R_Python";
```

```
SAS server started using Context SAS Studio compute context with SESSION_ID=14ea919b-b9a1-43ec-9df5-206c45c2b1aa-ses0000
12   ods listing close;ods html5 (id=saspy_internal) options(bitmap_mode='inline') device=svg style=HTMLBlue; ods graphics on /
12 ! outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: sashtml.htm
13
14   libname acs "/home/student/SIWWAF_SAS_SQL_R_Python";
NOTE: Libref ACS was successfully assigned as follows:
      Engine:        V9
      Physical Name: /home/student/SIWWAF_SAS_SQL_R_Python
15   ods html5 (id=saspy_internal) close;ods listing;
16
```

Set Up SAS Code for Pretty Graphs

One advantage to coding is that you can specify the exact options that you'd like to see in your graphs. One downside to coding is that you need to specify the exact options in your graph... otherwise you'll be stuck with some potentially less than ideal defaults. So, let's adjust the graph setting to something that will work for us.

```
In [ ]: *-------------------------------------------------------------------------------*
        |                                                  Examine Current Trends
        |                                          Part I: Adjust Graph Settings
        *-------------------------------------------------------------------------------*;

        ********************************************  Assign colors and symbols to plot lines;
        symbol1 interpol=join line=1    color=bl        ;
        symbol2 interpol=join line=2    color=b         ;
        symbol3 interpol=join line=3    color=br        ;
        symbol4 interpol=join line=4    color=g         ;
        symbol5 interpol=join line=5    color=p         ;

        ********************************************  Format Axis;
        legend1 position=(top center inside)label=none mode=share frame;
```

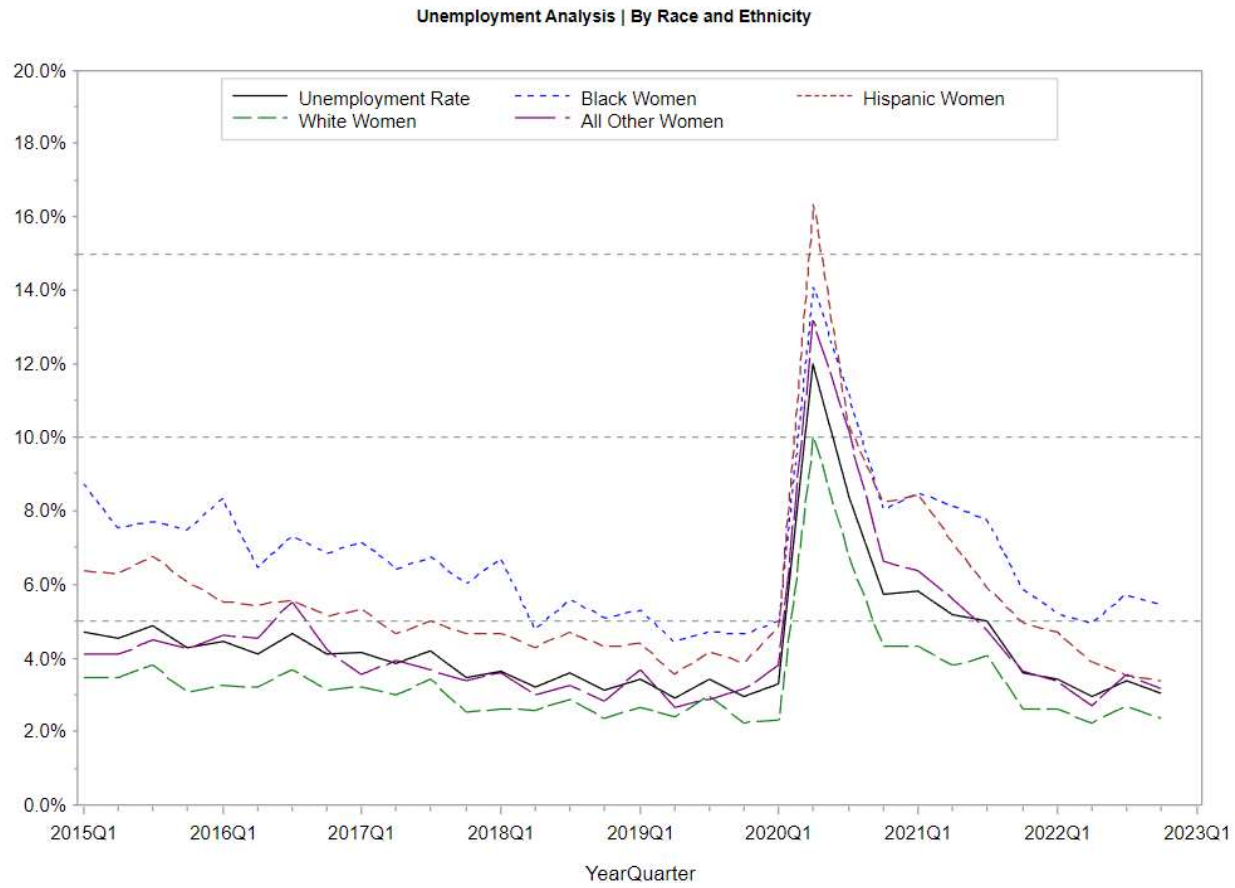
```
17   ods listing close;ods html5 (id=saspy_internal) options(bitmap_mode='inline') device=svg style=HTMLBlue; ods graphics on /
17 ! outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: sashtml1.htm
18
19   *-------------------------------------------------------------------------------*
20   |                                                  Examine Current Trends
21   |                                          Part I: Adjust Graph Settings
22   *-------------------------------------------------------------------------------*;
23
24   ********************************************  Assign colors and symbols to plot lines;
25   symbol1 interpol=join line=1       color=bl        ;
26   symbol2 interpol=join line=2       color=b         ;
27   symbol3 interpol=join line=3       color=br        ;
28   symbol4 interpol=join line=4       color=g         ;
29   symbol5 interpol=join line=5       color=p         ;
30
31   ********************************************  Format Axis;
32   legend1 position=(top center inside)label=none mode=share frame;
33   ods html5 (id=saspy_internal) close;ods listing;
34
```

U.S. Plots for Unemployment and Labor Force Participation | by Race/Ethnicity, Education Level, and Child Status | Quarterly

Ready or not - it's SAS macro loop time! We'll use the PROC GPLOT data below to produce six charts of labor supply trends, over time.

In [ ]:
```
*-------------------------------------------------------------------------------------*
|                                                 Part II: Macro Loops
*-------------------------------------------------------------------------------------*;

********************************** Define Variable List to Simplify Coding ;
%macro hilfe(from,to,by,var1,varlist2,title);

        ********************************************** Format axis for current range ;
        axis2   order=(&from to &to by &by ) offset=(0,0) label=none minor=(n=1);


        ********************************************** Overlay Plot ;
        title3 h=1.75pct &title;
        proc gplot data=acs.covid_labor_supply_us;
           plot ( &var1         )       * YearQuarter
                ( &varlist2 )   * YearQuarter
                 /      overlay
                        legend=legend1
                vaxis=axis2
                vref=&from to &to by .05
                lvref=2;
        run;
        quit;

%mend;


********************************************** Unemployment ;
%hilfe(0,.20,.02,UE_Women,UE_BlackWomen UE_HispanicWomen UE_WhiteWomen UE_OtherWomen,                "Unemployment Analysis | By Rac
%hilfe(0,.20,.02,UE_Women,UE_Women_LTHS UE_Women_HS UE_Women_SCollege UE_Women_CollegeP,             "Unemployment Analysis | By Edu
%hilfe(0,.20,.02,UE_Women,UE_Women_NoKids UE_Women_OlderKids UE_Women_YoungKids,                        "Unemployment Analysis


********************************************** Labor Force Participation ;
%hilfe(0.6,1,.05,LFP_Women,LFP_BlackWomen LFP_HispanicWomen LFP_WhiteWomen LFP_OtherWomen,           "Labor Force Participation Analy
%hilfe(0.6,1,.05,LFP_Women,LFP_Women_LTHS LFP_Women_HS LFP_Women_SCollege LFP_Women_CollegeP,  "Labor Force Participation Analysis | By
%hilfe(0.6,1,.05,LFP_Women,LFP_Women_NoKids LFP_Women_OlderKids LFP_Women_YoungKids,                "Labor Force Participation Analy
```

Note: The code above has been truncated to fit the page.



Note: The above is a portion of the output.

Notice the effect of COVID on these variables? Unemployment rates were greatly affected by the COVID19 pandemic - particularly in the period starting in Q2 of 2020. Women of all demographic groups were affected by the COVID19 lockdowns, but higher-education women didn't experience the same spike in unemployment (likely because they could still do their job remotely). Interestingly, there really wasn't much variation in UE rates by child status - and the UE levels have returned - essentially - to the same level that they were pre-COVID.

Let's pause on that last point a bit. UE being relatively unaffected by child status during the pandemic - and now being essentially at the same level as before the pandemic - would mean that HHS leadership shouldn't have to push for additional supports for working mothers. At least in terms of the unemployment rate.

But, labor force participation rate is another (and perhaps better) measure of labor supply. LFP essentially measures individuals' desire to be part of the labor force, whether they have a job or not. In the last chart, we see that women without children have the highest LFP rates. And women with young children (that is, those less than 5 years old) have the lowest levels of LFP. That all makes sense. The interesting part is at the end - starting in roughly Q1 of 2022. To me, it appears that LFP rates have rebounded to where they were before the pandemic... and might be slightly higher.

Again, for this policy analyst, I see no need for targeted policies, given that rates are either at or above where they were before the pandemic. But, you're the new analyst - what other interesting trends do you notice?

Produce Summary Tables of Underlying Data

To round out this section, let's ensure that the data are what we expect them to be. We can do this with a simple PROC PRINT statement in SAS. And this can also be thought of as the "show your work" portion of the programming.

In [ ]:
```
*-------------------------------------------------------------------------------------*
|                                                                        Backup Tables
*-------------------------------------------------------------------------------------*;

%macro backup(var,title);
        title3 h=1.75pct &title;
        proc print data=acs.covid_labor_supply_us noobs label;
                var YearQuarter ue_: ;
        run;
%mend;

%backup(ue_:,"Backup Tables - Unemployment Rates");
%backup(lfp_:,"Backup Tables - Labor Force Participation Rates";);
```

The SAS System

YearQuarter	Unemployment Rate	Black Women	Hispanic Women	White Women	All Other Women	EDUC < HS	EDUC = HS	Some College	College +	No Children	Older Children	Young Children
2015Q1	4.7%	8.7%	6.4%	3.5%	4.1%	11.7%	6.0%	5.5%	2.5%	4.5%	4.7%	5.4%
2015Q2	4.6%	7.5%	6.3%	3.5%	4.1%	11.2%	6.3%	4.6%	2.5%	4.5%	4.3%	5.4%
2015Q3	4.9%	7.7%	6.8%	3.8%	4.5%	12.1%	6.9%	4.6%	3.0%	4.6%	4.9%	5.6%
2015Q4	4.3%	7.5%	6.1%	3.1%	4.2%	8.9%	6.1%	4.6%	2.5%	4.1%	4.1%	5.0%
2016Q1	4.4%	8.3%	5.5%	3.3%	4.6%	10.1%	6.5%	4.6%	2.5%	4.4%	4.3%	5.0%
2016Q2	4.1%	6.5%	5.4%	3.2%	4.5%	10.0%	5.9%	4.2%	2.3%	4.0%	4.0%	4.7%
2016Q3	4.7%	7.3%	5.6%	3.7%	5.5%	10.1%	5.9%	5.0%	3.1%	4.8%	4.4%	5.0%
2016Q4	4.1%	6.8%	5.1%	3.1%	4.2%	10.0%	6.1%	4.0%	2.3%	4.2%	3.9%	4.4%
2017Q1	4.1%	7.2%	5.3%	3.2%	3.5%	10.1%	6.2%	4.1%	2.4%	4.0%	4.1%	4.5%
2017Q2	3.8%	6.4%	4.6%	3.0%	3.9%	7.7%	5.6%	4.3%	2.2%	3.7%	3.9%	4.0%
2017Q3	4.2%	6.7%	5.0%	3.4%	3.7%	8.2%	5.6%	4.7%	2.6%	4.1%	4.2%	4.4%
2017Q4	3.5%	6.0%	4.7%	2.5%	3.4%	7.3%	5.1%	3.7%	1.9%	3.4%	3.2%	4.3%
2018Q1	3.6%	6.7%	4.7%	2.6%	3.6%	7.6%	5.5%	3.8%	2.2%	3.7%	3.3%	4.5%
2018Q2	3.2%	4.8%	4.3%	2.6%	3.0%	6.7%	4.7%	3.5%	2.0%	3.2%	3.1%	3.8%
2018Q3	3.6%	5.6%	4.7%	2.9%	3.2%	6.7%	4.7%	4.0%	2.5%	3.6%	3.5%	4.2%
2018Q4	3.1%	5.1%	4.3%	2.4%	2.8%	7.8%	4.4%	3.3%	1.8%	3.1%	3.1%	3.2%
2019Q1	3.4%	5.3%	4.4%	2.7%	3.7%	8.2%	4.7%	3.7%	2.1%	3.6%	3.1%	4.0%

Note: The above is a portion of the output.

Data look legit to me! And, tables like this - though long - can be a great way to spot outliers. Notice the cluster around the time of the pandemic?

Task 2 | Examine State-Level UE and LFP Estimates | Yearly

The United States is a very large and diverse country. Economic conditions in Montana might not be the same as they are in North Carolina. And that's the beauty of Census data - we can parse out those individual trends and examine them over time. Let's make a few data modifications and then produce some PROC SGPANEL plots.

First Step: Collapse to State-Year Data Using SQL

We'll need to aggregate the data to the state + year level. As shown in Part 1, this can be done very succinctly using the powerful SQL language. Let's see how we can collapse our data in a single PROC SQL step - all while producing weighted (and more accurate) estimates of UE and LFP.

In [ ]:
```
*-------------------------------------------------------------------------------*
|                                                    Collapse Data
|                          Produce State-Level Estimates                        |
*-------------------------------------------------------------------------------*;

*****************************************************  By State ;
proc sql;
        create  table acs.covid_labor_supply2 as
        select  distinct state_fip, state_name,
            year(yearquarter) as Year format 9.,

/*****************************************************************  Labor Force Status | All  */
                        sum( ( unemp=1 ) * WTFINL )
                        sum( ( in_LF=1 ) * WTFINL )


/*****************************************************************  Labor Force Status | By Education  */

                        /*****************************************************  Unemployment */
                        sum( ( educ_ltd="High School Diploma" ) * ( unemp=1 ) * WTFINL )                / sum( ( educ_ltd="High Sch
                        sum( ( educ_ltd="Some College" ) * ( unemp=1 ) * WTFINL )                               / sum( ( educ_ltd="
                        sum( ( educ_ltd="College +" ) * ( unemp=1 ) * WTFINL )                                  / sum( ( educ_ltd="

                        /*****************************************************  LFP */
                        sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )                / sum( ( educ_ltd="High Sch
                        sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )                               / sum( ( educ_ltd="
                        sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )                                  / sum( ( educ_ltd="


/*****************************************************************  Labor Force Status | By Child Status  */

                        /*****************************************************  Unemployment */
                        sum( ( child_status="No Children" ) * ( unemp=1 ) * WTFINL )                    / sum( ( child_status="No Ch
                        sum( ( child_status="Older Children" ) * ( unemp=1 ) * WTFINL )                 / sum( ( child_status="Olde
                        sum( ( child_status="Child < 5" ) * ( unemp=1 ) * WTFINL )                              / sum( ( child_stat

                        /*****************************************************  LFP */
                        sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )                    / sum( ( child_status="No Ch
                        sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )                 / sum( ( child_status="Olde
                        sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )                              / sum( ( child_stat


        from    acs.acs_2015_2022
        group   by 1,2,3
        order   by 1,2,3 ;
quit;
```

Note: The code above has been truncated to fit the page.

```
95   ods listing close;ods html5 (id=saspy_internal) options(bitmap_mode='inline') device=svg style=HTMLBlue; ods graphics on /
95 ! outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: sashtml4.htm
96
97   *-------------------------------------------------------------------------------------*
98   |                                                          Collapse Data
99   |                         Produce State-Level Estimates                               |
100  *-------------------------------------------------------------------------------------*;
101
102  ************************************************** By State ;
103  proc sql;
104     create  table acs.covid_labor_supply2 as
105     select  distinct state_fip, state_name,
106              year(yearquarter) as Year format 9.,
107
108  /************************************************************** Labor Force Status | All  */
109                    sum( ( unemp=1 ) * WTFINL )
as UE_Women                              label="Unemployment
109! Rate"      format percent9.1              ,
110                    sum( ( in_LF=1 ) * WTFINL )
as LFP_Women                      label="LFP Rate"                        format percent9.1              ,
111
112
113  /************************************************************** Labor Force Status | By Education  */
114
115                    /************************************************** Unemployment */
116                    sum( ( educ_ltd="High School Diploma" ) * ( unemp=1 ) * WTFINL )               / sum( ( educ_ltd="High
116!  WTFINL )  as UE_Women_HS                  label="EDUC <= HS"             format percent9.1               ,
117                    sum( ( educ_ltd="Some College" ) * ( unemp=1 ) * WTFINL )                            / sum( ( educ_lt
117!            as UE_Women_SCollege    label="Some College"    format percent9.1              ,
118                    sum( ( educ_ltd="College +" ) * ( unemp=1 ) * WTFINL )                               / sum( ( educ_lt
as
118! UE_Women_CollegeP  label="College +"              format percent9.1              ,
```

Note: The above is a portion of the output.

I still think that SQL trick is cool. Notice how we turn select variables into 1/0 variables and then pass them through our SQL collapsing equation. For example, the code (**child_status="No Children"**) resolves to *1* if the child status is "No Children" and *0* otherwise. So, we essentially have a bunch of 1s and 0s multiplied together on a line-by-line basis, with a weight, which we then aggregate up to a state-year level. Now that's cool!

Transpose Data to Prepare for SGPANEL Plots

The data aren't quite where they need to be for the SGPANEL plots. But we can do that easily with a little arranging/rearranging of the data.

In [ ]:
```
proc transpose data=acs.covid_labor_supply2 out=tran1 (rename=(_label_=Group));
        by state_fip state_name year ;
run;


data    ue      (keep=state_fip state_name year group ue_rate)
                lfp (keep=state_fip state_name year group lfp_rate);
        set tran1 ;

        label   group="Group" ;

        if      index(_name_,"UE_")=1 then do;
                UE_Rate                 = col1 ;
                label   UE_Rate = "Unemployment Rate" ;
                format  UE_Rate percent9.1 ;
                output  ue ;
        end;

        else if index(_name_,"LFP_")=1 then do;
                LFP_Rate                = col1 ;
                label   LFP_Rate= "Labor Force Participation Rate" ;
                format  LFP_Rate percent9.1 ;
                output  lfp ;
        end;
run;
```

```
146  ods listing close;ods html5 (id=saspy_internal) options(bitmap_mode='inline') device=svg style=HTMLBlue; ods graphics on /
146! outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: sashtml5.htm
147
148  proc transpose data=acs.covid_labor_supply2 out=tran1 (rename=(_label_=Group));
149     by state_fip state_name year ;
150  run;

NOTE: There were 408 observations read from the data set ACS.COVID_LABOR_SUPPLY2.
NOTE: The data set WORK.TRAN1 has 5712 observations and 6 variables.
NOTE: PROCEDURE TRANSPOSE used (Total process time):
      real time           0.02 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      memory              2584.50k
      OS Memory           23472.00k
      Timestamp           08/09/2023 01:55:51 PM
      Step Count                        12  Switch Count  1
      Page Faults                       1
      Page Reclaims                     81
      Page Swaps                        0
      Voluntary Context Switches        14
      Involuntary Context Switches      6
      Block Input Operations            336
      Block Output Operations           1424


151
152
153  data       ue      (keep=state_fip state_name year group ue_rate)
154             lfp (keep=state_fip state_name year group lfp_rate);
155     set tran1 ;
156
157     label   group="Group" ;
158
159     if      index(_name_,"UE_")=1 then do;
160             UE_Rate                 = col1 ;
161             label   UE_Rate = "Unemployment Rate" ;
162             format  UE_Rate percent9.1 ;
163             output  ue ;
164     end;
```

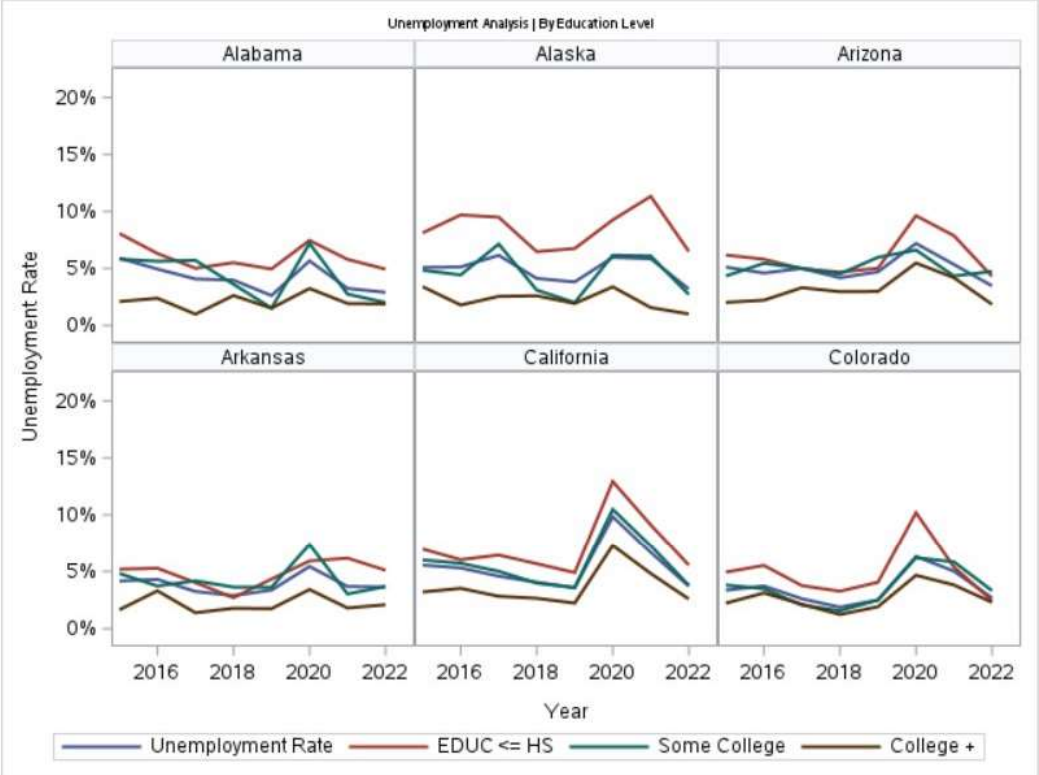
Note: The above is a portion of the output.

Now the data are prepared - and the way we need them to be for the SGPANEL plots. Onward!

Unemployment Rates

Let's first examine the unemployment rates, by education level and then by child status. And you know the follow-up questions: (1) what looks interesting and (2) are there any noticeable trends by child status over time?

In [ ]:
```
*-------------------------------------------------------------------------------------------*
|                                                 Unemployment Rate Analysis
*-------------------------------------------------------------------------------------------*;

*****************************************  By Education Level;
title3 h=1.75pct "Unemployment Analysis | By Education Level";
proc sgpanel data=ue;
        where group in ("Unemployment Rate" "EDUC <= HS" "Some College" "College +");
        panelby state_name / columns=3 rows=2 novarname ROWHEADERPOS=right ;
        series y=UE_Rate  x=Year               / group=group lineattrs=(thickness=2 pattern=solid);
        keylegend / title="" position=bottom;
        colaxis fitpolicy=thin valuesformat=best5. ;
run;
quit;


*****************************************   By Child Status;
title3 h=1.75pct "Unemployment Analysis | By Child Status";
proc sgpanel data=ue;
        where group in ("Unemployment Rate" "No Children" "Young Children" "Older Children");
        panelby state_name / columns=3 rows=2 novarname ROWHEADERPOS=right ;
        series y=UE_Rate  x=Year               / group=group lineattrs=(thickness=2 pattern=solid);
        keylegend / title="" position=bottom;
        colaxis fitpolicy=thin valuesformat=best5. ;
run;
quit;
```

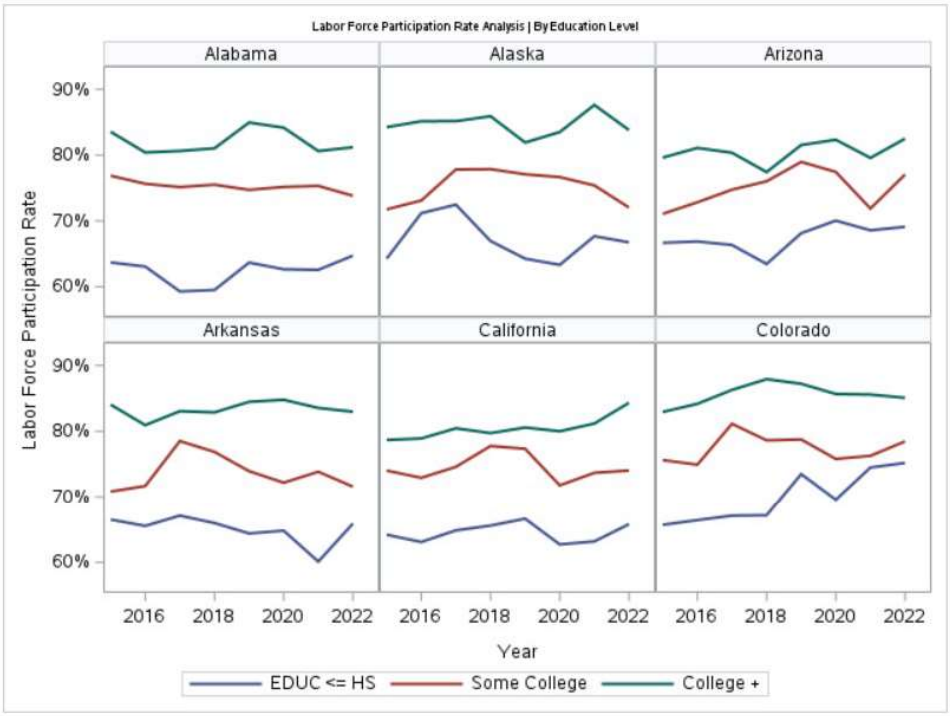


Note: The above is a portion of the output.

Labor Force Participation

Now that we're in the coding spirit of things, let's also run the LFP analysis. We can then share our deep thoughts on the entire analysis.

In [ ]:
```
*-------------------------------------------------------------------------------------------*
|                                                                          LFP Rate Analysis
*-------------------------------------------------------------------------------------------*;

*****************************************  By Education Level;
title3 h=1.75pct "Labor Force Participation Rate Analysis | By Education Level";
proc sgpanel data=lfp;
        where group in ("Labor Force Participation Rate Rate" "EDUC <= HS" "Some College" "College +");
        panelby state_name / columns=3 rows=2 novarname ROWHEADERPOS=right ;
        series y=lfp_Rate  x=Year               / group=group lineattrs=(thickness=2 pattern=solid);
        keylegend / title="" position=bottom;
        colaxis fitpolicy=thin valuesformat=best5. ;
run;
quit;


*****************************************   By Child Status;
title3 h=1.75pct "Labor Force Participation Rate Analysis | By Child Status";
proc sgpanel data=lfp;
        where group in ("Labor Force Participation Rate Rate" "No Children" "Young Children" "Older Children");
        panelby state_name / columns=3 rows=2 novarname ROWHEADERPOS=right ;
        series y=lfp_Rate  x=Year               / group=group lineattrs=(thickness=2 pattern=solid);
        keylegend / title="" position=bottom;
        colaxis fitpolicy=thin valuesformat=best5. ;
run;
quit;
```



Note: The above is a portion of the output.

My deep thoughts: UE and LFP rates differ markedly across states and demographic groups. But nearly all trends appear to have reverted to their pre-COVID trends.

Part 2 Recap

If the labor market has returned to pre-COVID conditions, there likely isn't a need to design programs to combat the effects of COVID. Because the market already handled that. Now if there are other issues that policy makers want to address, like making it easier for mothers with small children to enter the labor force, that's an entirely separate topic for another time.

And what do you see - our new policy maker extraordinaire? Do you agree?

SAS On-the-Job | Part 3

The Final Chapter! Map Time!

R you ready for the R portion of our analysis? And do you love a bad pun? Then you're in just the right place!

In Part 3 of our HHS adventures, we return to our state-level data on labor supply by prime-aged women. Our analytical objective in this section is to use the cool geographic mapping procedures in R to examine changes in the unemployment rate and the labor force participation rate from 2015 to 2022. As is the consistent theme, the general goal here is to see whether pre-COVID trends are back. Because if they're not, HHS leadership could advocate for policies to support segments of women in the labor force, such as women with small children.

Let's finish our analysis strong!

Prepare the R Environment

The first step is to load the requisite libraries in R. Note that a library is a different thing in R than SAS. In R, it's a package of tools, rather than a pointer to a data location. That's a fun fact for your next trivia night!

In [1]:
```
# Load required libraries
library(ggplot2)
library(reshape2)
library(grid)
library(scales)
library(mapproj)
library(viridis)
library(maps)
library(haven)
```

```
Loading required package: maps

Loading required package: viridisLite


Attaching package: 'viridis'


The following object is masked from 'package:maps':

    unemp


The following object is masked from 'package:scales':

    viridis_pal
```

Access Map Data in R

Next, we'll want to leverage data that's already included as part of the R install. We can then link this data to our data set - and ensure that data gets assigned properly in our upcoming maps.

In [2]:
```
# R comes with some pre-installed map data. Let's leverage that in this analysis

# Load the U.S. state map data
map_data <- map_data("state")

# Ensure Proper Casing for Merge
map_data$region = toupper(map_data$region)
```

It's never a bad thing to explore your data. So, let's check out the first 10 observations in **map_data** to ensure that it's in a form that we recognize.

In [3]:
```
head(map_data,10)
```

A data.frame: 10 × 6

	long	lat	group	order	region	subregion
	<dbl>	<dbl>	<dbl>	<int>	<chr>	<chr>
1	-87.46201	30.38968	1	1	ALABAMA	NA
2	-87.48493	30.37249	1	2	ALABAMA	NA
3	-87.52503	30.37249	1	3	ALABAMA	NA
4	-87.53076	30.33239	1	4	ALABAMA	NA
5	-87.57087	30.32665	1	5	ALABAMA	NA
6	-87.58806	30.32665	1	6	ALABAMA	NA
7	-87.59379	30.30947	1	7	ALABAMA	NA
8	-87.59379	30.28655	1	8	ALABAMA	NA
9	-87.67400	30.27509	1	9	ALABAMA	NA
10	-87.81152	30.25790	1	10	ALABAMA	NA

And that's just a big ole data set, which is why we limited it.

Now let's prepare **covid_labor_supply2** for a merge. Note that we need to match case on the **State_Name**, which is why we've got that extra line of code.

In [4]:
```
# Prepare the data from Part 2 for a merge
data1 <- read_sas('covid_labor_supply2.sas7bdat')

# Adjust Casing for Merge
data1$State_Name = toupper(data1$State_Name)
```

Let's explore the data one more time - to ensure that we've got the right one:

In [5]:
```
data1
```

State_FIP	State_Name	Year	UE_Women	LFP_Women	UE_Women_HS	UE_Women_SCollege	UE_Women_College
<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl
1	ALABAMA	2015	0.05903514	0.7120473	0.08071777	0.05814288	0.0209415
1	ALABAMA	2016	0.04944853	0.7001156	0.06312331	0.05633199	0.0238247
1	ALABAMA	2017	0.04078107	0.6901163	0.05017575	0.05711202	0.0098586
1	ALABAMA	2018	0.03973231	0.7044006	0.05505923	0.03617405	0.0262299
1	ALABAMA	2019	0.02619557	0.7215607	0.04950904	0.01513010	0.0150844
1	ALABAMA	2020	0.05666085	0.7270269	0.07461227	0.07200355	0.0323963
1	ALABAMA	2021	0.03245672	0.7124474	0.05794357	0.02744716	0.0192508
1	ALABAMA	2022	0.02901746	0.7142394	0.04939848	0.02008251	0.0188361
2	ALASKA	2015	0.05086221	0.7352526	0.08122067	0.04844351	0.0340012
2	ALASKA	2016	0.05125439	0.7620422	0.09696339	0.04429963	0.0178837
2	ALASKA	2017	0.06153648	0.7744281	0.09499766	0.07141625	0.0254987
2	ALASKA	2018	0.04122818	0.7621233	0.06473406	0.03100966	0.0259670
2	ALASKA	2019	0.03817501	0.7373931	0.06748475	0.02001494	0.0192393
2	ALASKA	2020	0.05973402	0.7406971	0.09259542	0.06160915	0.0338688
2	ALASKA	2021	0.05849297	0.7643210	0.11329813	0.06094058	0.0156290
2	ALASKA	2022	0.03190973	0.7421370	0.06488792	0.02718140	0.0100487
4	ARIZONA	2015	0.05116736	0.6937292	0.06177393	0.04338595	0.0201976
4	ARIZONA	2016	0.04597584	0.7091518	0.05807917	0.05450170	0.0221216
4	ARIZONA	2017	0.05000491	0.7177334	0.04968274	0.05023507	0.0331178
4	ARIZONA	2018	0.04180351	0.7197809	0.04677784	0.04517954	0.0295687
4	ARIZONA	2019	0.04699075	0.7543774	0.05018665	0.05976588	0.0298046
4	ARIZONA	2020	0.07196077	0.7495849	0.09628768	0.06613261	0.0547377
4	ARIZONA	2021	0.05342827	0.7229600	0.07869764	0.04335906	0.0418356
4	ARIZONA	2022	0.03472001	0.7535760	0.04302705	0.04743424	0.0184731
5	ARKANSAS	2015	0.04157140	0.7122765	0.05194079	0.04820517	0.0164943
5	ARKANSAS	2016	0.04298992	0.7119860	0.05275789	0.03725223	0.0330151
5	ARKANSAS	2017	0.03245451	0.7437986	0.04019476	0.04172991	0.0138726
5	ARKANSAS	2018	0.02887766	0.7302433	0.02703314	0.03633172	0.0177971
5	ARKANSAS	2019	0.03365544	0.7289428	0.04332022	0.03609976	0.0174455
5	ARKANSAS	2020	0.05429468	0.7235322	0.05911489	0.07365971	0.0341658
⋮	⋮	⋮	⋮	⋮	⋮	⋮	

Note: The output above has been truncated to fit the page.

53	WASHINGTON	2017	0.04237425	0.7402962	0.06222927	0.04016177	0.0293931
53	WASHINGTON	2018	0.03290793	0.7429658	0.03987669	0.03747069	0.0255725
53	WASHINGTON	2019	0.03536918	0.7737592	0.04300997	0.03812705	0.0254853
53	WASHINGTON	2020	0.06855711	0.7669504	0.10452287	0.07016393	0.0456262
53	WASHINGTON	2021	0.04346125	0.7665704	0.06913848	0.04681172	0.0262652
53	WASHINGTON	2022	0.04039190	0.7517061	0.07675749	0.02630231	0.0292339
54	WEST VIRGINIA	2015	0.05112208	0.6732519	0.05630622	0.06533021	0.0230089
54	WEST VIRGINIA	2016	0.05022583	0.6802121	0.07631215	0.05359652	0.0133044
54	WEST VIRGINIA	2017	0.03976826	0.6927771	0.06614172	0.03873594	0.0111468
54	WEST VIRGINIA	2018	0.03954694	0.6963568	0.05238028	0.03824651	0.0241528
54	WEST VIRGINIA	2019	0.03981524	0.7221303	0.06653061	0.03192802	0.0181296
54	WEST VIRGINIA	2020	0.06729936	0.7354362	0.09715844	0.06814199	0.0425112
54	WEST VIRGINIA	2021	0.04452219	0.7139289	0.06057270	0.04176168	0.0286558
54	WEST VIRGINIA	2022	0.03678607	0.7141047	0.04968335	0.03010867	0.0216450
55	WISCONSIN	2015	0.04441969	0.8257414	0.06156519	0.05700261	0.0204447
55	WISCONSIN	2016	0.03236035	0.8522440	0.03687559	0.04181084	0.0177896
55	WISCONSIN	2017	0.02619686	0.8340237	0.03657280	0.02625294	0.0151945
55	WISCONSIN	2018	0.02723953	0.8429349	0.04743418	0.02125453	0.0213638
55	WISCONSIN	2019	0.03045636	0.8436876	0.04566106	0.03270783	0.0206400
55	WISCONSIN	2020	0.05839944	0.8187473	0.10275351	0.06101152	0.0385566
55	WISCONSIN	2021	0.03841932	0.8044515	0.08529478	0.03141257	0.0213684
55	WISCONSIN	2022	0.02416327	0.7891549	0.05925166	0.01322186	0.0150217
56	WYOMING	2015	0.03095979	0.7595377	0.04420998	0.03711122	0.0146186
56	WYOMING	2016	0.03575319	0.7766703	0.03887426	0.04355085	0.0244288
56	WYOMING	2017	0.03077252	0.7711445	0.03619215	0.02936118	0.0274293
56	WYOMING	2018	0.04121874	0.7702229	0.05764346	0.03937832	0.0245109
56	WYOMING	2019	0.03103349	0.7989543	0.04143910	0.03040207	0.0163182
56	WYOMING	2020	0.03597091	0.7895148	0.03844135	0.04024423	0.0280347
56	WYOMING	2021	0.03231824	0.7873136	0.03819420	0.04110280	0.0180874
56	WYOMING	2022	0.03101467	0.7772843	0.05796820	0.02967962	0.0163739

Note: The output above has been truncated to fit the page.

Merge Time, Merge Time!

Are you ready to pull together the two pieces? Well, I used an exclamation point, so you know that I'm excited. Use the following code to combine **map_data** from R with our aggregated ACS data set. Data populated all the way across the row tells us that the merge was done properly - so check for that:

In [6]:
```
# Merge the data with map data
state_map_data <- merge(map_data, data1, by.x = "region", by.y = "State_Name", all.x = TRUE)

# Let's check out the first 20 observations of the merged data!
head(state_map_data,20)
```

A data.frame: 20 × 22

	region	long	lat	group	order	subregion	State_FIP	Year	UE_Women	LFP_Women	⋯	UE_Women_Cc
	<chr>	<dbl>	<dbl>	<dbl>	<int>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	
1	ALABAMA	-87.46201	30.38968	1	1	NA	1	2019	0.02619557	0.7215607	⋯	0.01
2	ALABAMA	-87.46201	30.38968	1	1	NA	1	2020	0.05666085	0.7270269	⋯	0.03
3	ALABAMA	-87.46201	30.38968	1	1	NA	1	2021	0.03245672	0.7124474	⋯	0.01
4	ALABAMA	-87.46201	30.38968	1	1	NA	1	2022	0.02901746	0.7142394	⋯	0.01
5	ALABAMA	-87.46201	30.38968	1	1	NA	1	2018	0.03973231	0.7044006	⋯	0.02
6	ALABAMA	-87.46201	30.38968	1	1	NA	1	2015	0.05903514	0.7120473	⋯	0.02
7	ALABAMA	-87.46201	30.38968	1	1	NA	1	2016	0.04944853	0.7001156	⋯	0.02
8	ALABAMA	-87.46201	30.38968	1	1	NA	1	2017	0.04078107	0.6901163	⋯	0.00
9	ALABAMA	-87.48493	30.37249	1	2	NA	1	2019	0.02619557	0.7215607	⋯	0.01
10	ALABAMA	-87.48493	30.37249	1	2	NA	1	2020	0.05666085	0.7270269	⋯	0.03
11	ALABAMA	-87.48493	30.37249	1	2	NA	1	2021	0.03245672	0.7124474	⋯	0.01
12	ALABAMA	-87.48493	30.37249	1	2	NA	1	2022	0.02901746	0.7142394	⋯	0.01
13	ALABAMA	-87.48493	30.37249	1	2	NA	1	2018	0.03973231	0.7044006	⋯	0.02
14	ALABAMA	-87.48493	30.37249	1	2	NA	1	2015	0.05903514	0.7120473	⋯	0.02
15	ALABAMA	-87.48493	30.37249	1	2	NA	1	2016	0.04944853	0.7001156	⋯	0.02
16	ALABAMA	-87.48493	30.37249	1	2	NA	1	2017	0.04078107	0.6901163	⋯	0.00
17	ALABAMA	-87.52503	30.37249	1	3	NA	1	2019	0.02619557	0.7215607	⋯	0.01
18	ALABAMA	-87.52503	30.37249	1	3	NA	1	2020	0.05666085	0.7270269	⋯	0.03
19	ALABAMA	-87.52503	30.37249	1	3	NA	1	2021	0.03245672	0.7124474	⋯	0.01
20	ALABAMA	-87.52503	30.37249	1	3	NA	1	2022	0.02901746	0.7142394	⋯	0.01

Note: The output above has been truncated to fit the page.

Data look good to me. How are things on your end? Do you see data all the way across the row?

One More Housekeeping Item

The last analysis preparation item is to think a bit more about how the map of the U.S. will appear. And though the exact details of **theme_map** can be handled on another day, let's steal some code to create a pretty theme map for our upcoming plots.

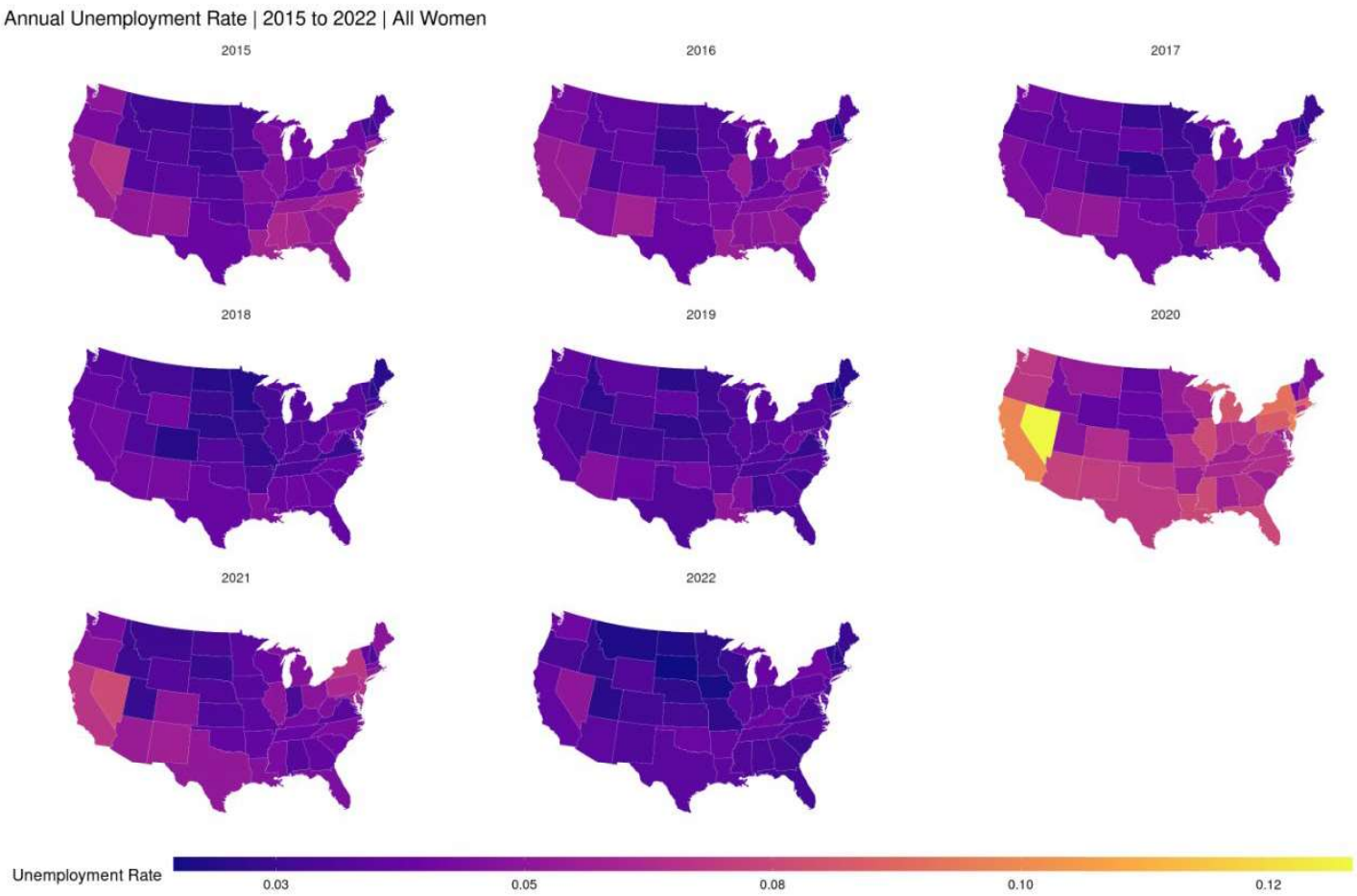
In [7]:
```
# Check this out (but don't ask me what each piece means)
# Source Code: https://socviz.co/index.html#preface ==> this book is great!
theme_map <- function(base_size=20, base_family="") {
  require(grid)
  theme_bw(base_size=base_size, base_family=base_family) %+replace%
    theme(axis.line=element_blank(),
          axis.text=element_blank(),
          axis.ticks=element_blank(),
          axis.title=element_blank(),
          panel.background=element_blank(),
          panel.border=element_blank(),
          panel.grid=element_blank(),
          panel.spacing=unit(0, "lines"),
          plot.background=element_blank(),
          legend.justification = c(0,0),
          legend.position = c(0,0),
          legend.key.width = unit(20,"lines")
    )
}

options(repr.plot.width = 24, repr.plot.height = 16)
```

Data are now loaded into R - and the theme map is ready to go. Let's get our mapping on!

Unemployment Rate Analysis

Let's now explore the unemployment rate across time in the U.S. We'll just focus on the continental U.S. (to simplify), but let's examine the trends for all prime-aged women:

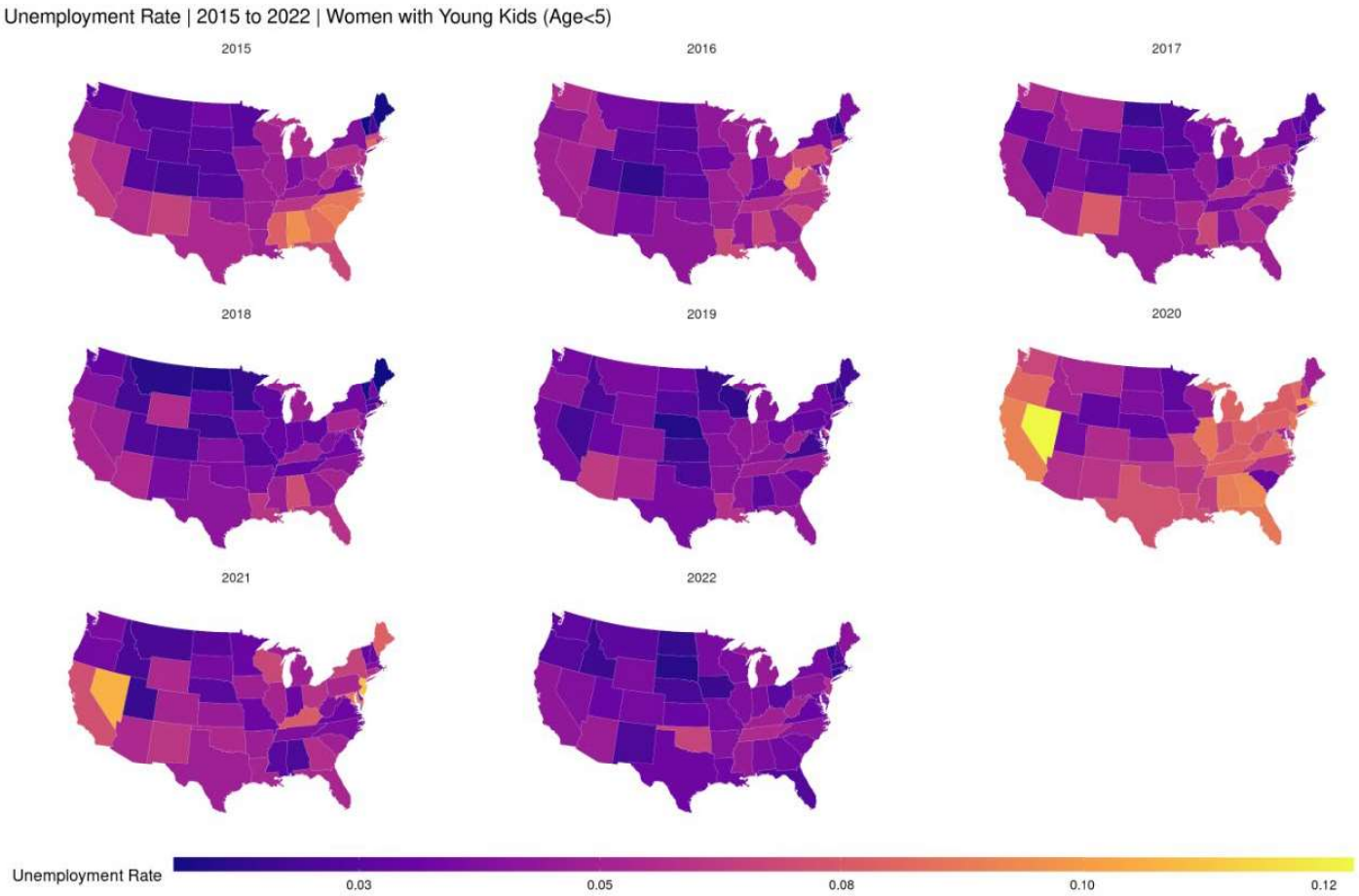
In [8]:
```
p0 <- ggplot(data = subset(state_map_data),
             mapping = aes(x = long, y = lat,
                           group = group,
                           fill = UE_Women))

p1 <- p0 + geom_polygon(color = "gray90", linewidth = 0.10) +
  coord_map(projection = "albers", lat0 = 39, lat1 = 45)

p2 <- p1 + scale_fill_viridis_c(option = "plasma", label=function(x) sprintf("%.2f",x))

p2 + theme_map() + facet_wrap(~ Year, ncol = 3) +
  theme(legend.position = "bottom", strip.background = element_blank()) +
  labs(fill = "Unemployment Rate ",
       title = "Annual Unemployment Rate | 2015 to 2022 | All Women")
```



What do you see in these trends? Again, I see that trends have returned to the pre-pandemic levels. And, dare I say it, it looks like employment prospects are better in 2022 than they were in the earlier period.

Because it's easy, let's just modify the code above and change the variable **UE_Women** to **UE_Women_YoungKids**. The code:

In [9]:
```
p0 <- ggplot(data = subset(state_map_data),
             mapping = aes(x = long, y = lat,
                           group = group,
                           fill = UE_Women_YoungKids))

p1 <- p0 + geom_polygon(color = "gray90", linewidth = 0.10) +
  coord_map(projection = "albers", lat0 = 39, lat1 = 45)

p2 <- p1 + scale_fill_viridis_c(option = "plasma", label=function(x) sprintf("%.2f",x))

p2 + theme_map() + facet_wrap(~ Year, ncol = 3) +
  theme(legend.position = "bottom",
        strip.background = element_blank()) +
  labs(fill = "Unemployment Rate ",
       title = "Unemployment Rate | 2015 to 2022 | Women with Young Kids (Age<5)")
```

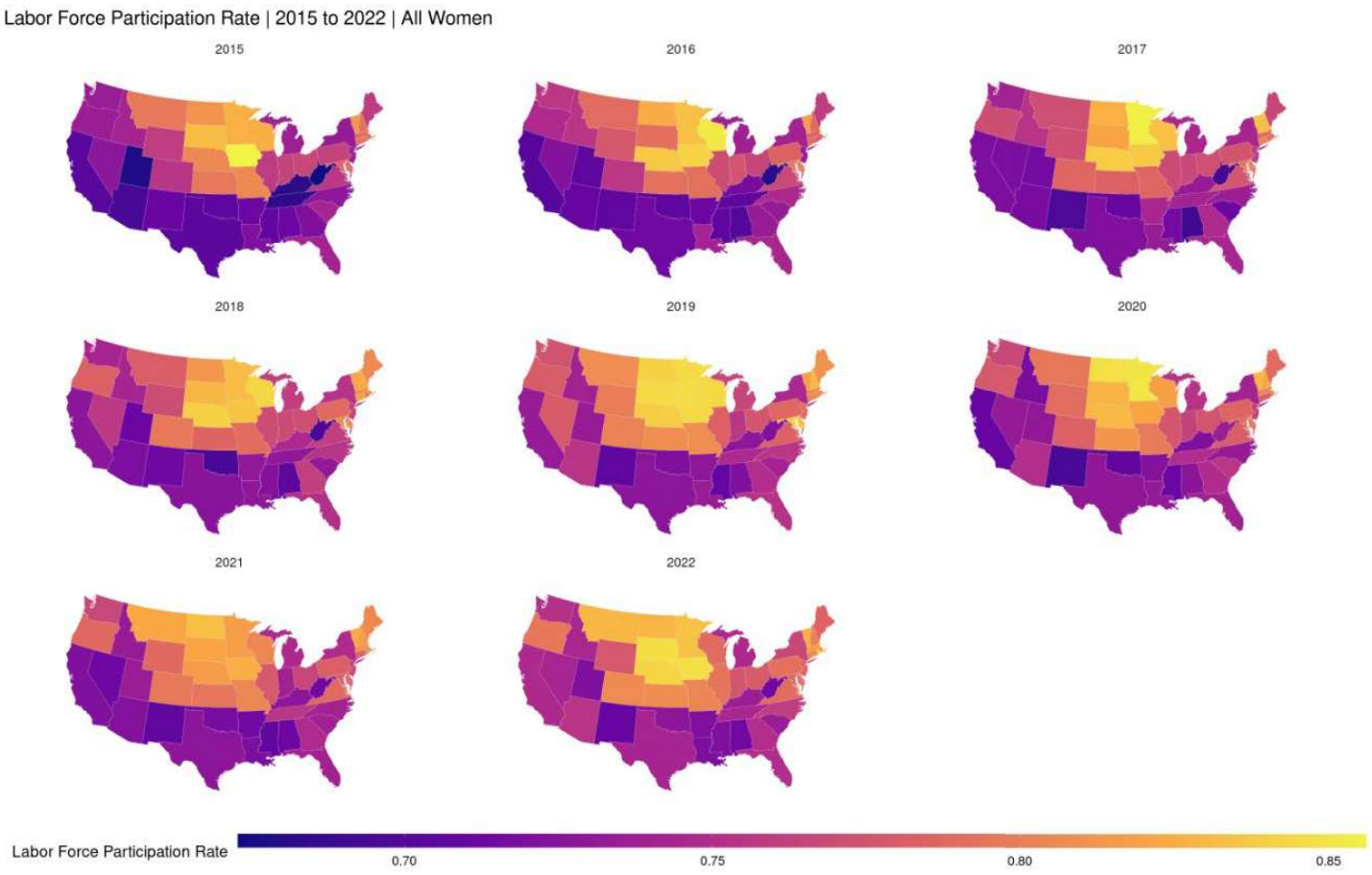


Again, a similar story plays out. Unemployment rates have returned to their pre-pandemic norms... and might be even better than they were in the early period.

Labor Force Participation Rate Analysis

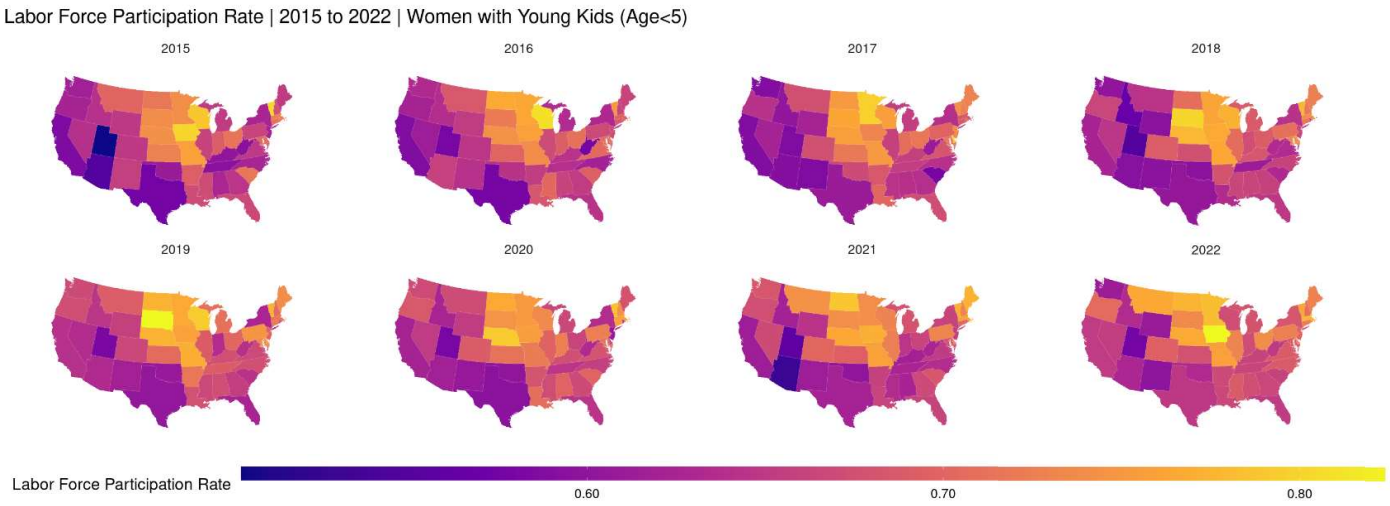
Because this code is so easy to modify after it's written, let's examine the labor force participant rate for all women. The story shouldn't change, but it is interesting to see the variation over time within states.

In [10]:
```
# Labor Force Participation Rate
r0 <- ggplot(data = subset(state_map_data),
             mapping = aes(x = long, y = lat,
                           group = group,
                           fill = LFP_Women) )

r1 <- r0 + geom_polygon(color = "gray90", linewidth = 0.10) +
  coord_map(projection = "albers", lat0 = 39, lat1 = 45)

r2 <- r1 + scale_fill_viridis_c(option = "plasma", label=function(x) sprintf("%.2f",x))

r2 + theme_map() + facet_wrap(~ Year, ncol = 3) +
  theme(legend.position = "bottom",
        strip.background = element_blank()) +
  labs(fill = "Labor Force Participation Rate ",
       title = "Labor Force Participation Rate | 2015 to 2022 | All Women")
```



The story repeats. Things were good. Then they got really bad. And they're either the same as the pre-pandemic levels - or better now. Let's also take a peek at LFP rates for women with young children.

In [11]:
```
# Labor Force Participation Rate, Women with young children
r0 <- ggplot(data = subset(state_map_data),
             mapping = aes(x = long, y = lat,
                           group = group,
                           fill = LFP_Women_YoungKids) )

r1 <- r0 + geom_polygon(color = "gray90", linewidth = 0.05) +
  coord_map(projection = "albers", lat0 = 39, lat1 = 45)

r2 <- r1 + scale_fill_viridis_c(option = "plasma", label=function(x) sprintf("%.2f",x))

r2 + theme_map() + facet_wrap(~ Year, ncol = 4) +
  theme(legend.position = "bottom",
        strip.background = element_blank()) +
  labs(fill = "Labor Force Participation Rate ",
       title = "Labor Force Participation Rate | 2015 to 2022 | Women with Young Kids (Age<5)")
```



Yet again a similar pattern. Good. Then Bad. Then good again.

My recommendation is that HHS leadership take no action on designing supports specifically targeting women affected by the pandemic. Whether segments of women - that is, those with young children - should receive other supports to encourage their labor supply is a great research question for another time. For now, congrats on completing your first day as a Department of Health and Human Services Policy Analyst!