Public Policy Analyst for a Day:
A SAS Viya Workbench Tour with SAS, SQL, Python + R

SAS On-the-Job Activity

Purpose

This SAS On-the-Job activity shows learners how they can seamlessly switch from SAS, SQL, R, and Python within SAS Viya Workbench – under the mantra of *use the right tool for the job*… or more simply, *use the tool you know*. This activity follows a new policy analyst at the Department of Health and Human Services, who is tasked with understanding how the coronavirus, i.e., COVID19, impacted the labor supply of prime-aged women across the United States.

Software

This SAS On-the-Job activity uses SAS Viya Workbench for Learners. We will use both the Visual Studio and Jupyter integrated development environments (IDEs), while leveraging 3 separate kernels: (1) Python, (2) SAS, and (3) R. And while this training is delivered via SAS Viya Workbench, know that the SAS code – if configured properly – could run in a SAS9 or SAS Viya environment.

Industry Alignment

This On-the-Job activity aligns most closely with economic and public policy analysis. We'll explore aggregated U.S. and state-level trends in the unemployment and labor force participation rates, through a series of descriptive statistics, charts, and maps.

Table of 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, targeted 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 notebooks created by your predecessor
- understand what the previous analyst did - particularly with the integration of SAS, SQL, Python, and R
- modify existing code to explore more (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 a single unified project. 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.

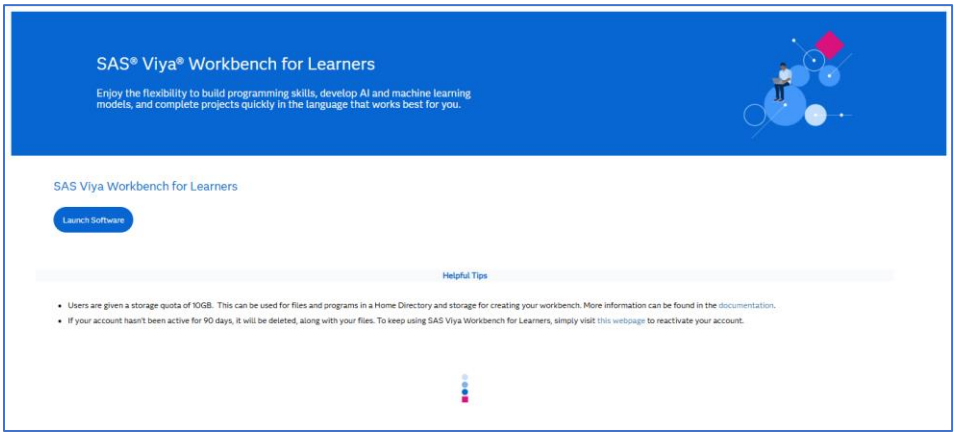
(2) The second notebook focuses on SAS code. We'll leverage some SAS code and macros to explore aggregated U.S. trends. We'll conclude by creating a state-level data set that we can then use to map trends in R (confession: R can produce some great maps!).

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

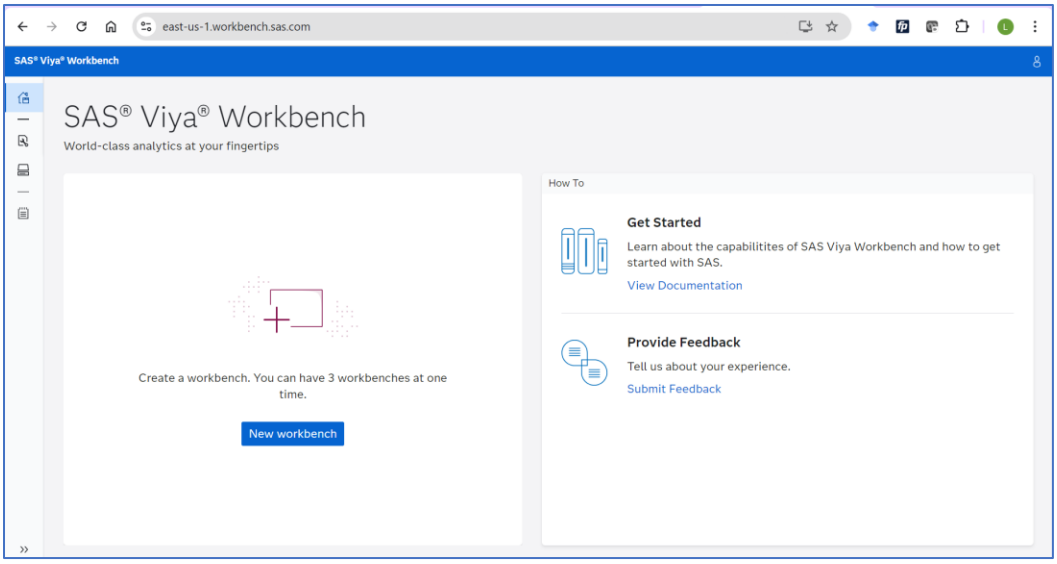
Setting up SAS Viya Workbench for Learners

To begin this multilingual adventure, we first need to access SAS Viya Workbench for Learners and successfully set you up in the environment. So, this section is truly the start – and needs to be followed exactly as specified below. Once all the pieces are in place, you are then just a few clicks away from running all the programs required to analyze public policy trends using SAS, SQL, R, and Python.

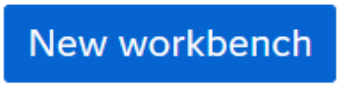
- To access SAS Viya Workbench for Learners, start on our product landing page here: www.sas.com/workbenchforlearners
- Once you have a **SAS profile** and accept the terms of use, you should be taken to the SAS Viya Workbench for Learners product page, here: Course: SAS Viya Workbench for Learners. It looks like this:



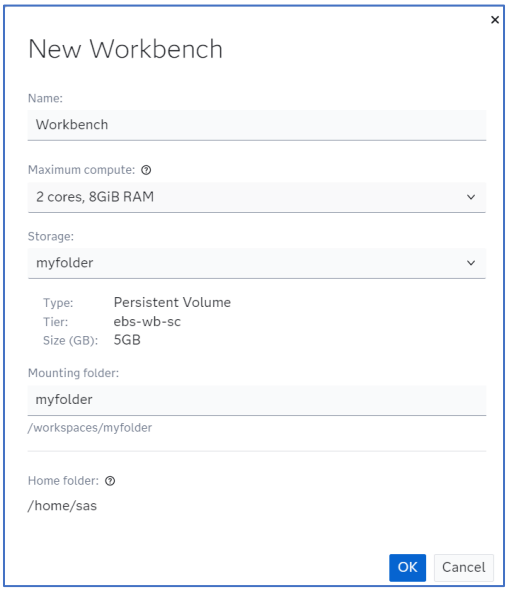
- Click the **Launch Software** button above to get started!
- If this is your first time in SAS Viya Workbench, you'll need to create a **New workbench**. And creation is just a few button clicks away:



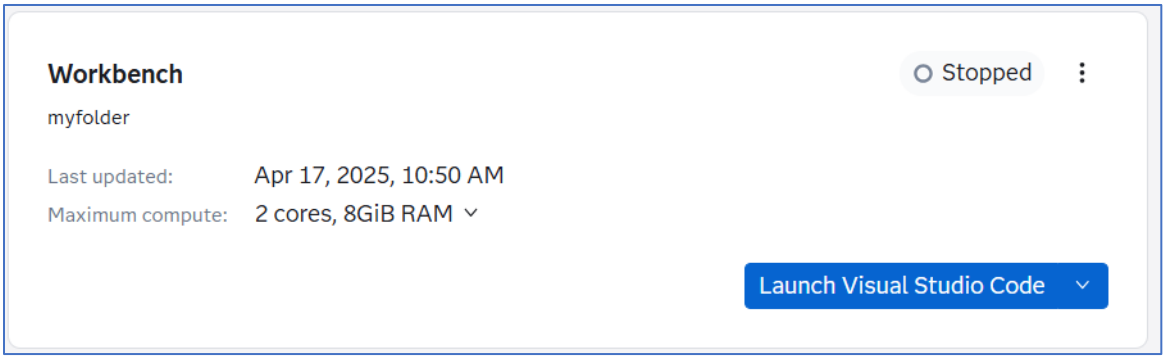
- But, before getting right to it, spend a little time getting acquainted with the WFL landing page. For example, check out the **Get Started** resources, or the options under the **Workbenches** and **Storage** icons on the left side. In other words, just become more and more comfortable with the resources available on the WFL landing page. And when you're satiated, click that **New workbench** button:



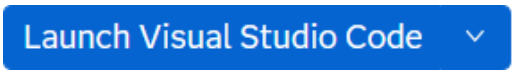
- The following appears:



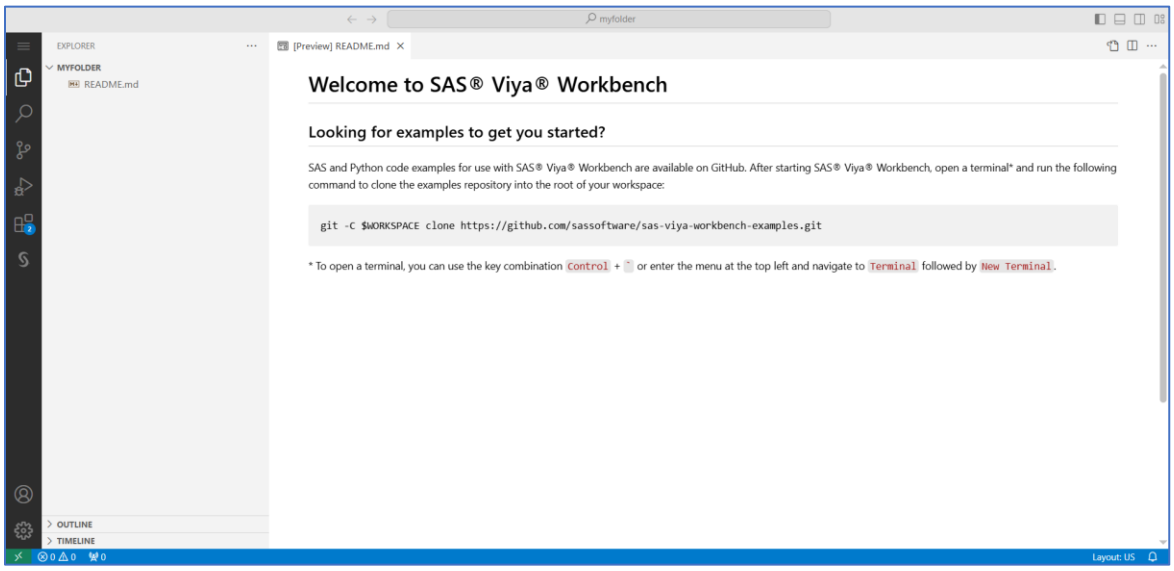
- For **Name**, let's just keep **Workbench** to simplify. And, in fact, just keep the other settings at their default and then click **OK**. Success looks like this:



- Next, in our new workbench, click the **Launch Visual Studio Code** button, here:

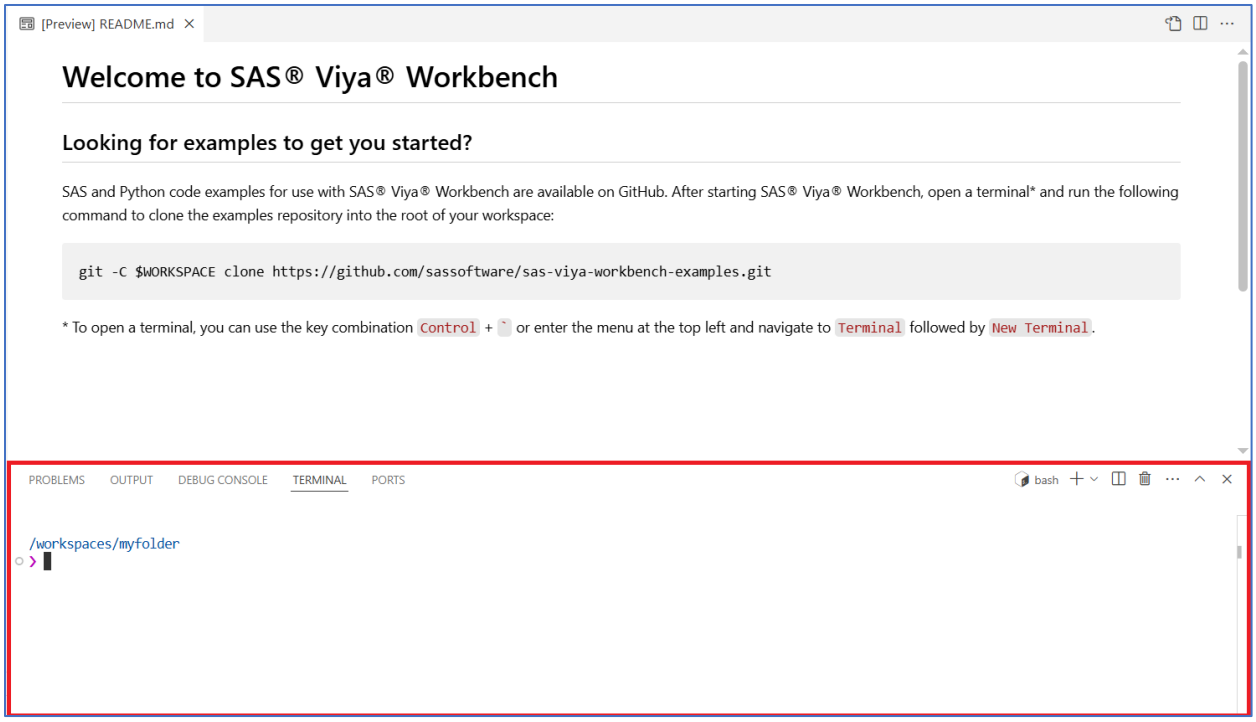


- Your workbench container loads! And welcome to **Visual Studio**!



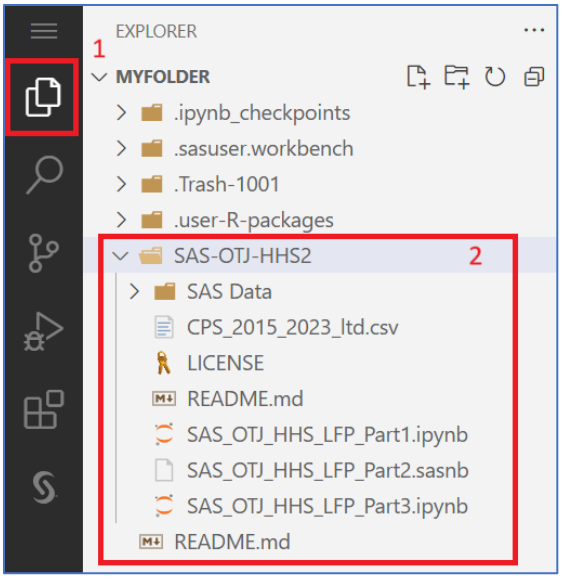
- If this is your first time in the environment, explore the Visual Studio environment for a bit. Because we're all about exploring and learning here.
- Once you're more comfortable with Visual Studio, the last step in this section is to access the data for this SAS On-the-Job activity. You'll want to access the **Terminal** – which is likely hidden by default. No worries – it can be activated via the **Toggle Panel** if it's not already up. And that button is in the upper-right corner, here:

- And a happy terminal will look like the following:



- You're almost there! Simply type the following line of code into the Terminal:
  git clone https://github.com/lincolngroves/SAS-OTJ-HHS2.git
- Then hit **Enter**. It's that easy to clone a GitHub repo and success looks like this:

- One last important item for this section. Let's find the notebooks and data left behind by our predecessor! Access the **Explorer** button on the left pane and then expand **the SAS-OTJ-HHS2** folder. Like so:



- The three notebooks we will use in our analysis are *SAS_OTJ_HHS_LFP_Part1*, *SAS_OTJ_HHS_LFP_Part2* and – you guessed it – *SAS_OTJ_HHS_LFP_Part3*. That's the way we'll walk through the programs. See you in the next section!

Python, SAS, and SQL Integration

Current Population Survey (CPS) Data Import

Welcome to Day 1 as Public Policy Analyst at HHS! With access to software and your data setup, we can officially begin our analysis. Yay!

Note: you can follow along in the notebook or – if you're feeling really brave – recreate the notebook from the narrative below. Either way, learning is the objective here!

The first thing we'll need to do in our analytics adventure is to load the data into the environment. Toward that goal, let's first connect with the SAS Viya Workbench 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 Current Population Survey Data.

And why start with SAS code in a Python program? Well, this project has used hackers from all backgrounds in the analysis - and one of the previous analysts 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!

The first step: ensure that saspy is installed:

```
!pip install saspy
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting saspy
  Downloading saspy-5.103.0-py3-none-any.whl.metadata (3.9 kB)
Downloading saspy-5.103.0-py3-none-any.whl (10.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.0/10.0 MB 125.4 MB/s eta 0:00:00
Installing collected packages: saspy
```

Successfully installed saspy-5.103.0

Next, import SASpy and create a SAS session:

```
import saspy

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

Using SAS Config named: default

SAS server started using Context 0001 with SESSION_ID=0001-ses0000

[*Note: I'll stop printing the log after this one. But the log is very important, so please read!*]

With the connection established, we can now use the submit method to run SAS code using a Python kernel. Notice how we pull the original data directly from GitHub using a traditional SAS programmer's approach. We could have certainly used a Python procedure to load the data, but I'm simply showing you another way here. 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 cps "/workspaces/myfolder/SAS-OTJ-HHS2/SAS Data" ;

    filename git_url url 'https://raw.githubusercontent.com/lincolngroves/SAS-OTJ-HHS2/main/CPS_2015_2023_ltd.csv';

    /* Use PROC IMPORT to read the CSV file */
    proc import datafile=git_url
                out=cps.cps_2015_2023
                dbms=csv
                replace;
                guessingrows=5000;
    run;

    """,
)

# Convert SAS data file to Python data frame
cps_df = sas_session.sd2df(table="cps_2015_2023",libref="cps")
```

[4] ✓ 2.8s Python

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!

```
# One Simple line will pull up the Python dataframe
cps_df
```

	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
...	...	...	...	...	...	...	...	...	...
157780	56.0	Wyoming	2023-10-01	All Other	Some College	Older Children	0.0	1.0	979.8808
157781	56.0	Wyoming	2023-10-01	All Other	College +	No Children	0.0	1.0	2851.1120
157782	56.0	Wyoming	2023-10-01	All Other	College +	Older Children	0.0	0.0	1240.7009
157783	56.0	Wyoming	2023-10-01	All Other	College +	Older Children	0.0	1.0	588.7811
157784	56.0	Wyoming	2023-10-01	All Other	College +	Child < 5	0.0	1.0	305.8516

157785 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 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!

```
# 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 may – or may not – have stated: *gotta know thy data!*

```
# Perform the PROC MEANS equivalent in Python
summary = cps_df.describe()
summary = summary.round(2)  # Set the decimal places to 2

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

[ ]

```
       State_FIP                    YearQuarter      Unemp      in_LF  \
count  157785.00                         157785  157785.00  157785.00
mean       28.47  2019-04-09 16:21:24.624013824       0.18       0.62
min         1.00            2015-01-01 00:00:00       0.00       0.00
25%        15.00            2017-01-01 00:00:00       0.00       0.00
50%        29.00            2019-04-01 00:00:00       0.00       1.00
75%        41.00            2021-07-01 00:00:00       0.00       1.00
max        56.00            2023-10-01 00:00:00       1.00       1.00
std        15.68                            NaN       0.38       0.49

           WTFINL
count   157785.00
mean     43753.89
min        123.30
25%       4122.79
50%      11952.13
75%      37107.80
max    2366665.82
std     103258.75
```

We can see from **Unemp** and **in_LF** above that we have a good range 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 the Distribution of Select Variables

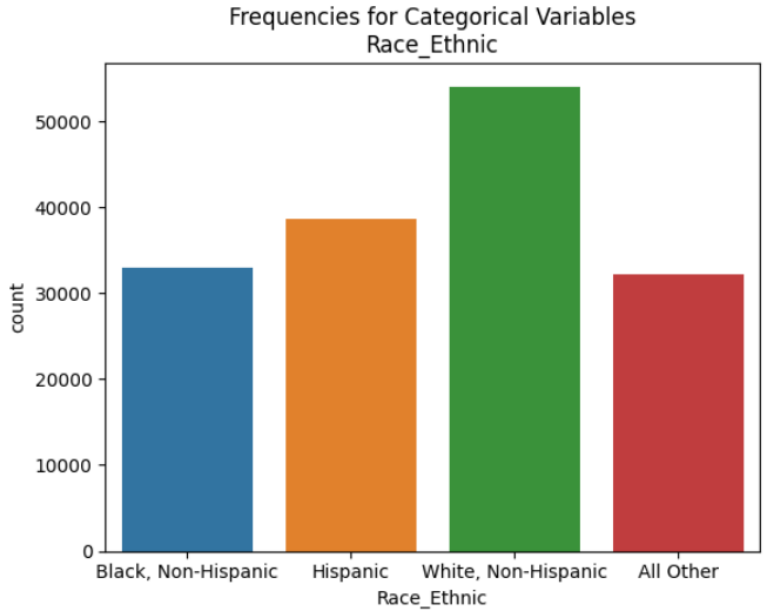
Data appear to be valid and clean. What a great way to start as an analyst... because - truth be told - this never happens.

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.

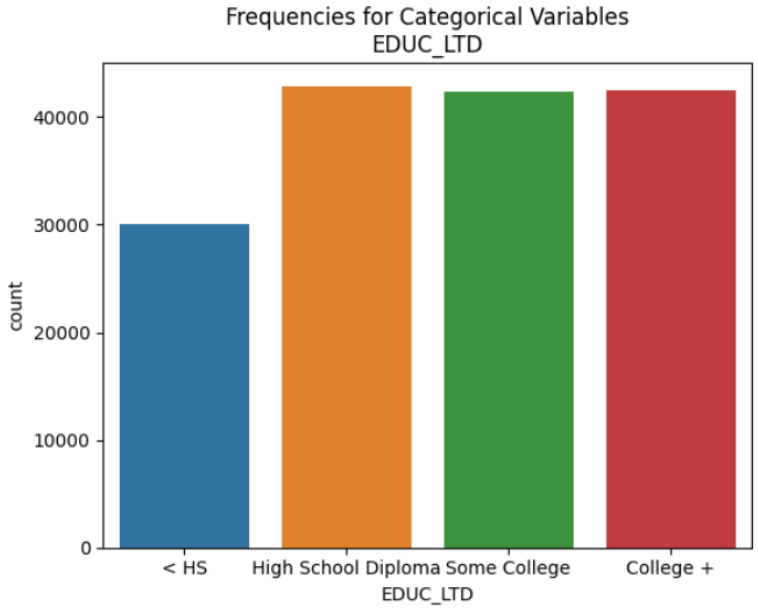
Let's use some classic Python tools to do exactly that. We'll keep the analysis unweighted, which still allows us to examine the relative proportion of each variable in the analysis. This is a perfectly fine test for this stage of the analysis. :)

```
[ ]
# 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 = cps_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=cps_df, x=variable)
    plt.title(f"{title}\n{variable}")
    plt.show()
```

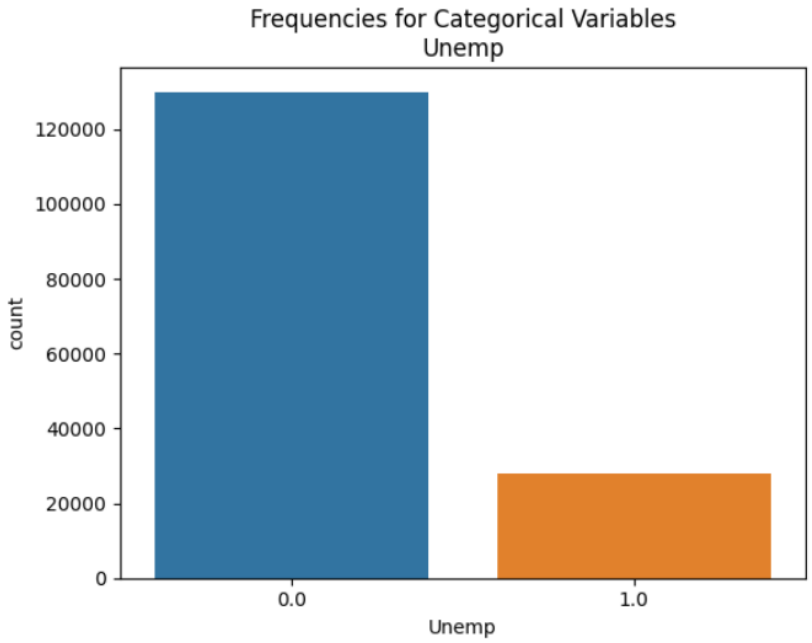
```
Frequencies for Categorical Variables

Race_Ethnic
                Value  Frequency
0  White, Non-Hispanic      54059
1             Hispanic      38569
2  Black, Non-Hispanic      33029
3            All Other      32128
```



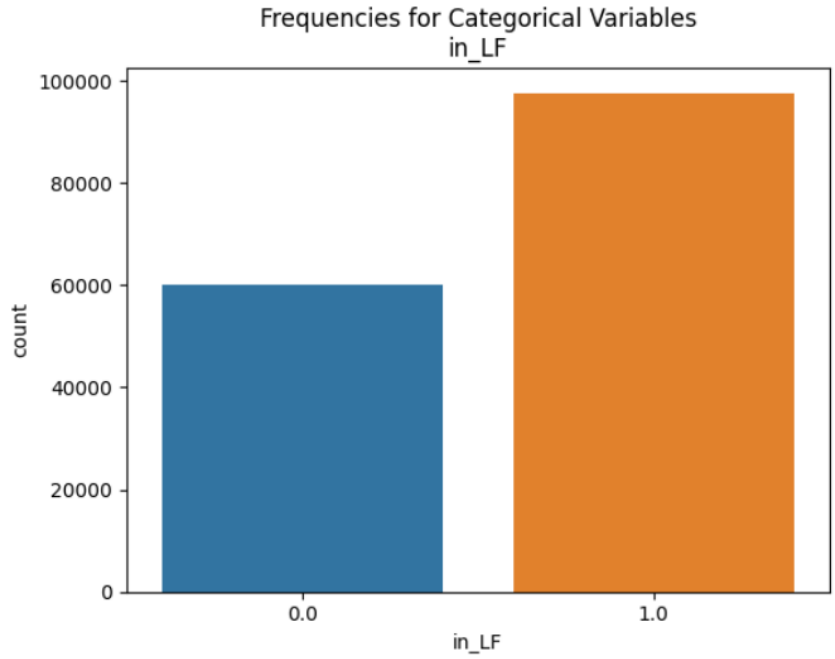

```
Frequencies for Categorical Variables

EDUC_LTD
                Value  Frequency
0  High School Diploma      42882
1            College +      42502
2         Some College      42372
3                < HS      30029
```



```
Frequencies for Categorical Variables

Unemp
   Value  Frequency
0    0.0     129930
1    1.0      27855
```




```
Frequencies for Categorical Variables

in_LF
   Value  Frequency
0    1.0      97678
1    0.0      60107
```

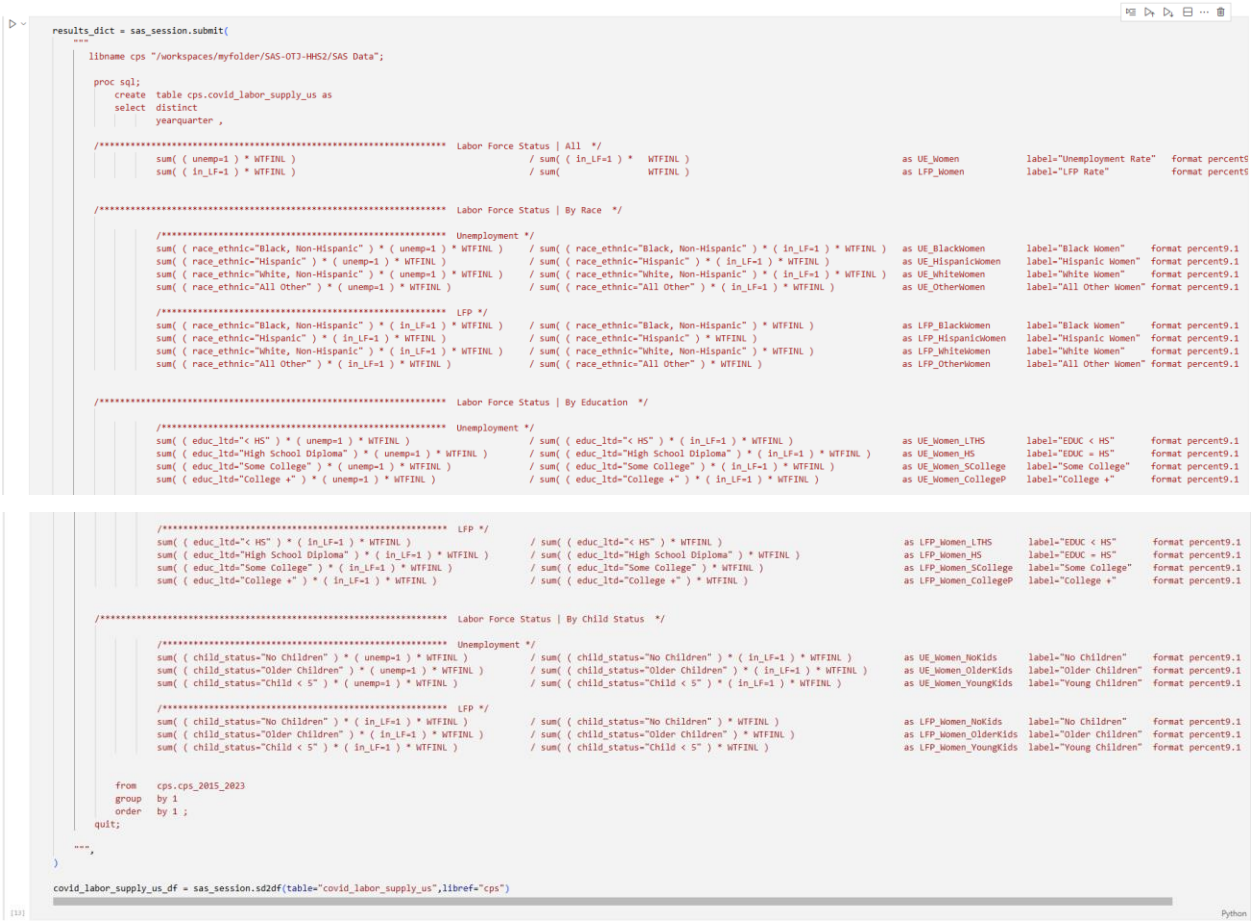


Do you see any interesting trends in the underlying data? Again, these data aren't weighted, so they don't represent U.S. averages. But they're a useful check, nonetheless.

My interesting findings: (1) White, Non-Hispanic women 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 to Collapse Data and 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 how elegantly SQL can collapse data? Well, I am!



```
results_dict = sas_session.submit(
    """
    libname cps "/workspaces/myfolder/SAS-OTJ-HHS2/SAS Data";

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

        /*************************************************************  Labor Force Status | All  */
                sum( ( unemp=1 ) * WTFINL )                             / sum( ( in_LF=1 ) *   WTFINL )                                 as UE_Women             label="Unemployment Rate"   format percentS
                sum( ( in_LF=1 ) * WTFINL )                             / sum(                 WTFINL )                                 as LFP_Women            label="LFP Rate"            format percentS


        /*************************************************************  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_BlackWomen       label="Black Women"     format percent9.1
                sum( ( race_ethnic="Hispanic" ) * ( unemp=1 ) * WTFINL )                / sum( ( race_ethnic="Hispanic" ) * ( in_LF=1 ) * WTFINL )              as UE_HispanicWomen    label="Hispanic Women"  format percent9.1
                sum( ( race_ethnic="White, Non-Hispanic" ) * ( unemp=1 ) * WTFINL )     / sum( ( race_ethnic="White, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )   as UE_WhiteWomen       label="White Women"     format percent9.1
                sum( ( race_ethnic="All Other" ) * ( unemp=1 ) * WTFINL )               / sum( ( race_ethnic="All Other" ) * ( in_LF=1 ) * WTFINL )             as UE_OtherWomen       label="All Other Women" format percent9.1

                /*********************************************  LFP */
                sum( ( race_ethnic="Black, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )     / sum( ( race_ethnic="Black, Non-Hispanic" ) * WTFINL )                 as LFP_BlackWomen       label="Black Women"     format percent9.1
                sum( ( race_ethnic="Hispanic" ) * ( in_LF=1 ) * WTFINL )                / sum( ( race_ethnic="Hispanic" ) * WTFINL )                            as LFP_HispanicWomen    label="Hispanic Women"  format percent9.1
                sum( ( race_ethnic="White, Non-Hispanic" ) * ( in_LF=1 ) * WTFINL )     / sum( ( race_ethnic="White, Non-Hispanic" ) * WTFINL )                 as LFP_WhiteWomen       label="White Women"     format percent9.1
                sum( ( race_ethnic="All Other" ) * ( in_LF=1 ) * WTFINL )               / sum( ( race_ethnic="All Other" ) * WTFINL )                           as LFP_OtherWomen       label="All Other Women" format percent9.1


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

                /*********************************************  Unemployment */
                sum( ( educ_ltd="< HS" ) * ( unemp=1 ) * WTFINL )                       / sum( ( educ_ltd="< HS" ) * ( in_LF=1 ) * WTFINL )                     as UE_Women_LTHS        label="EDUC < HS"       format percent9.1
                sum( ( educ_ltd="High School Diploma" ) * ( unemp=1 ) * WTFINL )        / sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )      as UE_Women_HS          label="EDUC = HS"       format percent9.1
                sum( ( educ_ltd="Some College" ) * ( unemp=1 ) * WTFINL )               / sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )             as UE_Women_SCollege    label="Some College"    format percent9.1
                sum( ( educ_ltd="College +" ) * ( unemp=1 ) * WTFINL )                  / sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )                as UE_Women_CollegeP    label="College +"       format percent9.1

                /*********************************************  LFP */
                sum( ( educ_ltd="< HS" ) * ( in_LF=1 ) * WTFINL )                       / sum( ( educ_ltd="< HS" ) * WTFINL )                                   as LFP_Women_LTHS       label="EDUC < HS"       format percent9.1
                sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )        / sum( ( educ_ltd="High School Diploma" ) * WTFINL )                    as LFP_Women_HS         label="EDUC = HS"       format percent9.1
                sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )               / sum( ( educ_ltd="Some College" ) * WTFINL )                           as LFP_Women_SCollege   label="Some College"    format percent9.1
                sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )                  / sum( ( educ_ltd="College +" ) * WTFINL )                              as LFP_Women_CollegeP   label="College +"       format percent9.1


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

                /*********************************************  Unemployment */
                sum( ( child_status="No Children" ) * ( unemp=1 ) * WTFINL )            / sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )          as UE_Women_NoKids      label="No Children"     format percent9.1
                sum( ( child_status="Older Children" ) * ( unemp=1 ) * WTFINL )         / sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )       as UE_Women_OlderKids   label="Older Children"  format percent9.1
                sum( ( child_status="Child < 5" ) * ( unemp=1 ) * WTFINL )              / sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )            as UE_Women_YoungKids   label="Young Children"  format percent9.1

                /*********************************************  LFP */
                sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )            / sum( ( child_status="No Children" ) * WTFINL )                        as LFP_Women_NoKids     label="No Children"     format percent9.1
                sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )         / sum( ( child_status="Older Children" ) * WTFINL )                     as LFP_Women_OlderKids  label="Older Children"  format percent9.1
                sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )              / sum( ( child_status="Child < 5" ) * WTFINL )                          as LFP_Women_YoungKids  label="Young Children"  format percent9.1


            from    cps.cps_2015_2023
            group   by 1
            order   by 1 ;
        quit;

    """,
)

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

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

Lots of good stuff in that cell 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 commands or SAS statements 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:

```
[ ]
#  Print the Data
covid_labor_supply_us_df
```

	YearQuarter	UE_Women	LFP_Women	UE_BlackWomen	UE_HispanicWomen	UE_WhiteWomen	UE_OtherWomen	LFP_BlackWomen	LFP_HispanicWomen	LFP_WhiteWomen	...	LFP_Women_LTHS	LFP_Women_HS	LFP_Women_SCollege
0	2015-01-01	0.047211	0.738310	0.087219	0.063629	0.034865	0.040917	0.769755	0.658631	0.761507	...	0.491811	0.668360	0.761317
1	2015-04-01	0.045522	0.738932	0.075385	0.062764	0.034821	0.041100	0.770039	0.669844	0.761003	...	0.494933	0.674909	0.766443
2	2015-07-01	0.048815	0.734344	0.077023	0.067676	0.038039	0.044745	0.766467	0.661662	0.758963	...	0.488201	0.672994	0.753138
3	2015-10-01	0.042647	0.743679	0.074805	0.060602	0.030634	0.042491	0.766863	0.666811	0.767981	...	0.495868	0.675396	0.763120
4	2016-01-01	0.044473	0.744559	0.083346	0.055370	0.032719	0.046326	0.758997	0.667180	0.771075	...	0.507608	0.672918	0.765526
5	2016-04-01	0.041264	0.742937	0.064624	0.054351	0.031966	0.045198	0.760563	0.663884	0.771219	...	0.510625	0.670497	0.761224
6	2016-07-01	0.046789	0.744580	0.073069	0.055856	0.036961	0.055377	0.773851	0.680854	0.766116	...	0.520268	0.678013	0.760150
7	2016-10-01	0.040895	0.749199	0.068410	0.051406	0.031419	0.042302	0.782778	0.679291	0.771722	...	0.509389	0.681654	0.769555
8	2017-01-01	0.041431	0.749183	0.071577	0.053231	0.032228	0.035445	0.767590	0.679051	0.773372	...	0.502559	0.670492	0.772951
9	2017-04-01	0.038418	0.751382	0.064231	0.046475	0.030096	0.039340	0.779143	0.677983	0.774767	...	0.505116	0.686433	0.766346
10	2017-07-01	0.041737	0.751989	0.067395	0.049934	0.034223	0.036773	0.788550	0.676314	0.774134	...	0.523662	0.682852	0.766432
11	2017-10-01	0.034527	0.751999	0.060091	0.046555	0.025240	0.033678	0.784759	0.678561	0.774776	...	0.512919	0.687826	0.768748
12	2018-01-01	0.036484	0.751639	0.066816	0.046814	0.026355	0.036118	0.779314	0.689551	0.772807	...	0.513919	0.672965	0.771012
13	2018-04-01	0.032264	0.754139	0.047910	0.042908	0.025821	0.029885	0.778637	0.694838	0.774984	...	0.516066	0.677920	0.773822
14	2018-07-01	0.036136	0.757496	0.055848	0.046951	0.028897	0.032393	0.786638	0.690584	0.778083	...	0.514762	0.688247	0.771640
15	2018-10-01	0.031217	0.764470	0.050949	0.042892	0.023536	0.028238	0.789246	0.702501	0.787812	...	0.534594	0.688909	0.778568
16	2019-01-01	0.034355	0.758455	0.052966	0.043908	0.026655	0.036996	0.783327	0.694364	0.781927	...	0.509407	0.683143	0.771557
17	2019-04-01	0.029075	0.757178	0.044322	0.035683	0.023850	0.026770	0.785704	0.692376	0.780635	...	0.491038	0.686646	0.772079
18	2019-07-01	0.034164	0.761716	0.047183	0.041776	0.029697	0.028835	0.782763	0.696946	0.784596	...	0.522425	0.689121	0.776300
19	2019-10-01	0.029583	0.776167	0.046503	0.038723	0.022436	0.031692	0.799629	0.711747	0.800829	...	0.539971	0.695639	0.784795
20	2020-01-01	0.032782	0.768584	0.049942	0.048650	0.023085	0.038160	0.785616	0.702340	0.795131	...	0.502722	0.696470	0.774904
21	2020-04-01	0.119946	0.742715	0.141014	0.163660	0.100107	0.131798	0.753963	0.664120	0.771113	...	0.484847	0.636924	0.755187
22	2020-07-01	0.083675	0.749040	0.111939	0.103573	0.068076	0.101604	0.760583	0.675646	0.777798	...	0.491291	0.660191	0.754504
23	2020-10-01	0.057499	0.754180	0.080323	0.082113	0.043167	0.066333	0.771861	0.678268	0.783910	...	0.505496	0.662835	0.763768
24	2021-01-01	0.058201	0.752554	0.084852	0.084390	0.043133	0.063640	0.772730	0.671112	0.780266	...	0.481014	0.657484	0.757796
25	2021-04-01	0.051745	0.747654	0.081171	0.071486	0.037894	0.055946	0.780887	0.671366	0.770881	...	0.474402	0.661417	0.750424
26	2021-07-01	0.049911	0.752037	0.077457	0.059001	0.040836	0.047518	0.776056	0.674238	0.776888	...	0.508709	0.665302	0.750815

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

Let's further probe the data:

```
[ ]
# 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)
```

```
...                 YearQuarter  UE_Women  LFP_Women  UE_BlackWomen  \
count                    36    36.000     36.000         36.000
mean   2019-05-17 06:40:00     0.042      0.756          0.066
min    2015-01-01 00:00:00     0.028      0.734          0.044
25%    2017-03-09 12:00:00     0.033      0.749          0.051
50%    2019-05-16 12:00:00     0.036      0.753          0.062
75%    2021-07-24 00:00:00     0.046      0.764          0.076
max    2023-10-01 00:00:00     0.120      0.779          0.141
std                    NaN     0.017      0.012          0.020

       UE_HispanicWomen  UE_WhiteWomen  UE_OtherWomen  LFP_BlackWomen  \
count            36.000         36.000         36.000          36.000
mean              0.055          0.033          0.043           0.781
min               0.034          0.021          0.027           0.754
25%               0.042          0.025          0.032           0.771
50%               0.048          0.029          0.037           0.782
75%               0.059          0.035          0.045           0.789
max               0.164          0.100          0.132           0.813
std               0.024          0.015          0.021           0.013

       LFP_HispanicWomen  LFP_WhiteWomen  ...  LFP_Women_LTHS  LFP_Women_HS  \
count             36.000          36.000  ...          36.000        36.000
mean               0.687           0.779  ...           0.508         0.678
min                0.659           0.759  ...           0.474         0.637
25%                0.674           0.772  ...           0.496         0.670
...
max                  0.786               0.706
std                  0.011               0.018

[8 rows x 25 columns]
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

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

What do you see? My musings: (1) There are 36 observations in the aggregated data set. The math: 9 years * 4 quarters. (2) The (weighted) average unemployment rate is 0.042 - or 4.2%. 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. Get all output by running the SAS Notebook in WFL.*

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 Workbench environment. So, let's assign a link to that location on our drive.

```
[ ]
libname cps "/workspaces/myfolder/SAS-OTJ-HHS2/SAS Data";
```

Set Up SAS Code for Pretty Graphs

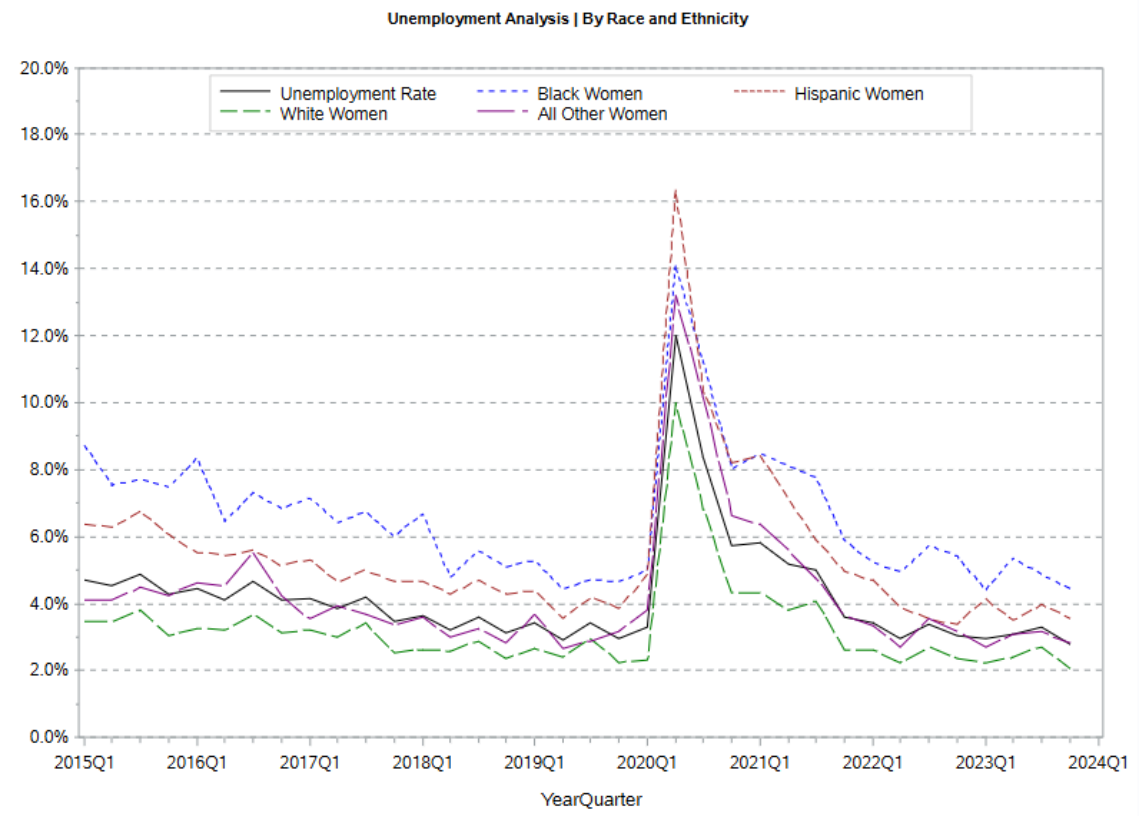
One advantage to coding is that you can specify the exact options that you'd like to see in your graphs. One downside to coding is that you need to specify the exact options in your graph... otherwise you'll be stuck with some potentially less than ideal defaults. So, let's adjust the graph setting to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

```
[ ]
*-------------------------------------------------------------------------------*
|                          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;
```

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.

```
*-------------------------------------------------------------------------------*
|                          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=cps.covid_labor_supply_us;
       plot ( &var1     )   * YearQuarter
            ( &varlist2 )   * YearQuarter
            /  overlay
               legend=legend1
               vaxis=axis2
               vref=&from to &to by &by
               lvref=2;
    run;
    quit;

%mend;


********************************************** Unemployment ;
%hilfe(0,.20,.02,UE_Women,UE_BlackWomen UE_HispanicWomen UE_WhiteWomen UE_OtherWomen,          "Unemployment Analysis | By Race and Ethnicity");
%hilfe(0,.20,.02,UE_Women,UE_Women_LTHS UE_Women_HS UE_Women_SCollege UE_Women_CollegeP,      "Unemployment Analysis | By Education Level");
%hilfe(0,.20,.02,UE_Women,UE_Women_NoKids UE_Women_OlderKids UE_Women_YoungKids,              "Unemployment Analysis | By Child Status");


********************************************** Labor Force Participation ;
%hilfe(0.6,1,.05,LFP_Women,LFP_BlackWomen LFP_HispanicWomen LFP_WhiteWomen LFP_OtherWomen,    "Labor Force Participation Analysis | By Race and Ethnicity");
%hilfe(0.6,1,.05,LFP_Women,LFP_Women_LTHS LFP_Women_HS LFP_Women_SCollege LFP_Women_CollegeP, "Labor Force Participation Analysis | By Education Level");
%hilfe(0.6,1,.05,LFP_Women,LFP_Women_NoKids LFP_Women_OlderKids LFP_Women_YoungKids,          "Labor Force Participation Analysis | By Child Status");
```



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-educated 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.

```
*-------------------------------------------------------------------------------*
|                                Backup Tables                                  |
*-------------------------------------------------------------------------------*;

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

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

[ ]

YearQuarter	Unemployment Rate	Black Women	Hispanic Women	White Women	All Other Women	EDUC < HS	EDUC = HS	Some College	College +
2015Q1	4.7%	8.7%	6.4%	3.5%	4.1%	11.7%	6.0%	5.5%	2.5%
2015Q2	4.6%	7.5%	6.3%	3.5%	4.1%	11.2%	6.3%	4.6%	2.5%
2015Q3	4.9%	7.7%	6.8%	3.8%	4.5%	12.1%	6.9%	4.6%	3.0%
2015Q4	4.3%	7.5%	6.1%	3.1%	4.2%	8.9%	6.1%	4.6%	2.5%
2016Q1	4.4%	8.3%	5.5%	3.3%	4.6%	10.1%	6.5%	4.6%	2.5%
2016Q2	4.1%	6.5%	5.4%	3.2%	4.5%	10.0%	5.9%	4.2%	2.3%
2016Q3	4.7%	7.3%	5.6%	3.7%	5.5%	10.1%	5.9%	5.0%	3.1%
2016Q4	4.1%	6.8%	5.1%	3.1%	4.2%	10.0%	6.1%	4.0%	2.3%
2017Q1	4.1%	7.2%	5.3%	3.2%	3.5%	10.1%	6.2%	4.1%	2.4%
2017Q2	3.8%	6.4%	4.6%	3.0%	3.9%	7.7%	5.6%	4.3%	2.2%
2017Q3	4.2%	6.7%	5.0%	3.4%	3.7%	8.2%	5.6%	4.7%	2.6%
2017Q4	3.5%	6.0%	4.7%	2.5%	3.4%	7.3%	5.1%	3.7%	1.9%
2018Q1	3.6%	6.7%	4.7%	2.6%	3.6%	7.6%	5.5%	3.8%	2.2%
2018Q2	3.2%	4.8%	4.3%	2.6%	3.0%	6.7%	4.7%	3.5%	2.0%
2018Q3	3.6%	5.6%	4.7%	2.9%	3.2%	6.7%	4.7%	4.0%	2.5%
2018Q4	3.1%	5.1%	4.3%	2.4%	2.8%	7.8%	4.4%	3.3%	1.8%
2019Q1	3.4%	5.3%	4.4%	2.7%	3.7%	8.2%	4.7%	3.7%	2.1%
2019Q2	2.9%	4.4%	3.6%	2.4%	2.7%	6.8%	3.8%	3.2%	1.9%
2019Q3	3.4%	4.7%	4.2%	3.0%	2.9%	7.0%	4.5%	3.7%	2.4%
2019Q4	3.0%	4.7%	3.9%	2.2%	3.2%	6.4%	4.6%	3.1%	1.8%
2020Q1	3.3%	5.0%	4.9%	2.3%	3.8%	8.1%	4.7%	3.5%	2.1%
2020Q2	12.0%	14.1%	16.4%	10.0%	13.2%	24.6%	17.5%	14.2%	7.7%

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.

```
*-------------------------------------------------------------------------------*
|                                Collapse Data                                  |
|                          Produce State-Level Estimates                        |
*-------------------------------------------------------------------------------*;

************************************************** By State ;
proc sql;
   create  table cps.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 )                        as UE_Women           label="Unemployment Rate"   format percent9.1
           sum( ( in_LF=1 ) * WTFINL )                                  / sum(                  WTFINL )                        as LFP_Women          label="LFP Rate"            format percent9.1


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

           /********************************************** Unemployment */
           sum( ( educ_ltd="High School Diploma" ) * ( unemp=1 ) * WTFINL )     / sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )  as UE_Women_HS          label="EDUC <= HS"      format percent9.1      ,
           sum( ( educ_ltd="Some College" ) * ( unemp=1 ) * WTFINL )            / sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )         as UE_Women_SCollege    label="Some College"    format percent9.1      ,
           sum( ( educ_ltd="College +" ) * ( unemp=1 ) * WTFINL )               / sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )            as UE_Women_CollegeP    label="College +"       format percent9.1      ,

           /********************************************** LFP */
           sum( ( educ_ltd="High School Diploma" ) * ( in_LF=1 ) * WTFINL )     / sum( ( educ_ltd="High School Diploma" ) * WTFINL )                as LFP_Women_HS         label="EDUC <= HS"      format percent9.1      ,
           sum( ( educ_ltd="Some College" ) * ( in_LF=1 ) * WTFINL )            / sum( ( educ_ltd="Some College" ) * WTFINL )                       as LFP_Women_SCollege   label="Some College"    format percent9.1      ,
           sum( ( educ_ltd="College +" ) * ( in_LF=1 ) * WTFINL )               / sum( ( educ_ltd="College +" ) * WTFINL )                          as LFP_Women_CollegeP   label="College +"       format percent9.1      ,


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

           /********************************************** Unemployment */
           sum( ( child_status="No Children" ) * ( unemp=1 ) * WTFINL )         / sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )      as UE_Women_NoKids      label="No Children"     format percent9.1      ,
           sum( ( child_status="Older Children" ) * ( unemp=1 ) * WTFINL )      / sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )   as UE_Women_OlderKids   label="Older Children"  format percent9.1      ,
           sum( ( child_status="Child < 5" ) * ( unemp=1 ) * WTFINL )           / sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )        as UE_Women_YoungKids   label="Young Children"  format percent9.1      ,

           /********************************************** LFP */
           sum( ( child_status="No Children" ) * ( in_LF=1 ) * WTFINL )         / sum( ( child_status="No Children" ) * WTFINL )                    as LFP_Women_NoKids     label="No Children"     format percent9.1      ,
           sum( ( child_status="Older Children" ) * ( in_LF=1 ) * WTFINL )      / sum( ( child_status="Older Children" ) * WTFINL )                 as LFP_Women_OlderKids  label="Older Children"  format percent9.1      ,
           sum( ( child_status="Child < 5" ) * ( in_LF=1 ) * WTFINL )           / sum( ( child_status="Child < 5" ) * WTFINL )                      as LFP_Women_YoungKids  label="Young Children"  format percent9.1


   from    cps.cps_2015_2023
   group   by 1,2,3
   order   by 1,2,3 ;
quit;
```

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.

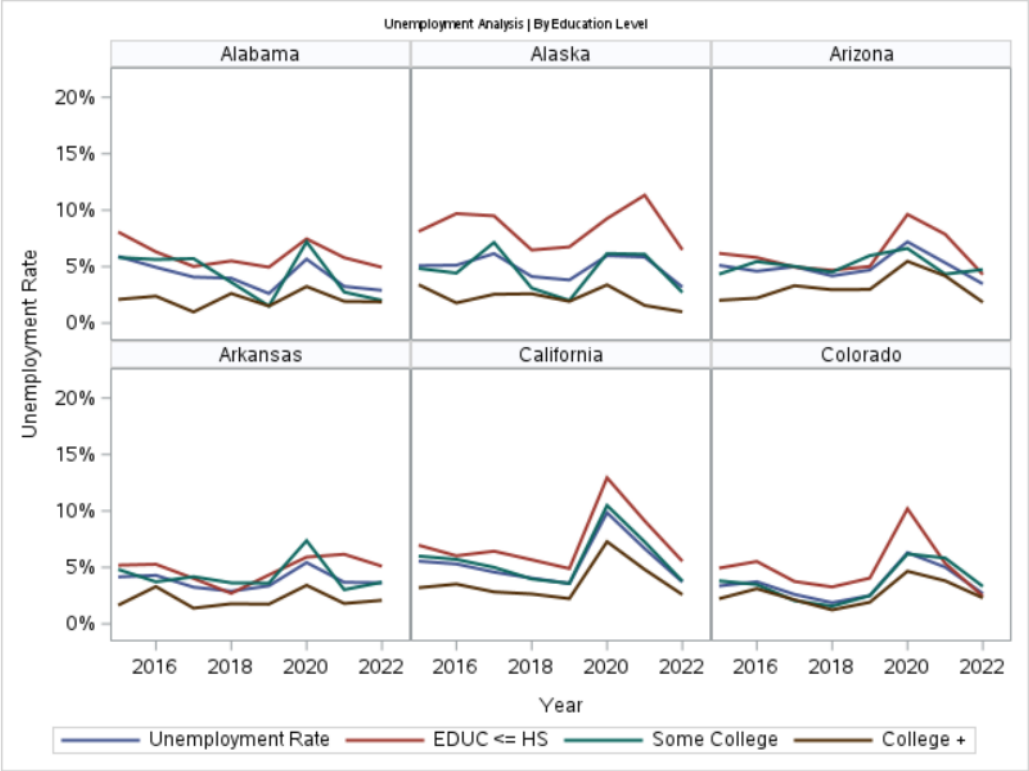
```
proc transpose data=cps.covid_labor_supply2 out=tran1 (rename=(_label_=Group));
    by state_fip state_name year ;
run;


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;
```

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?

```
*-------------------------------------------------------------------------------------------*
|                              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;
```

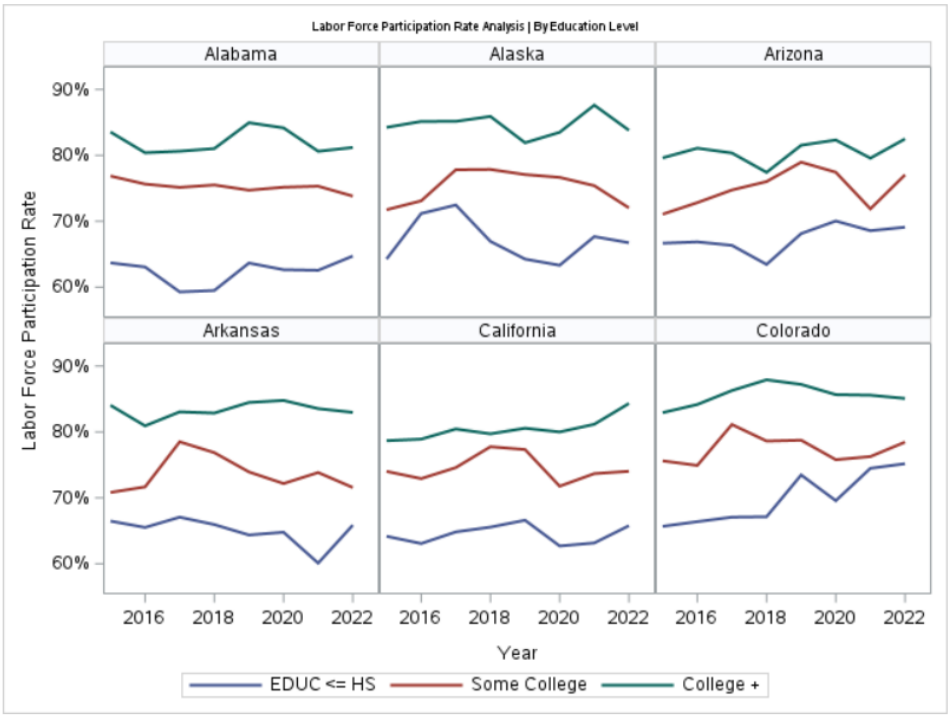


Note: *The above is a portion of the output.*

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.

```
*-------------------------------------------------------------------------------------*
|                                    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!

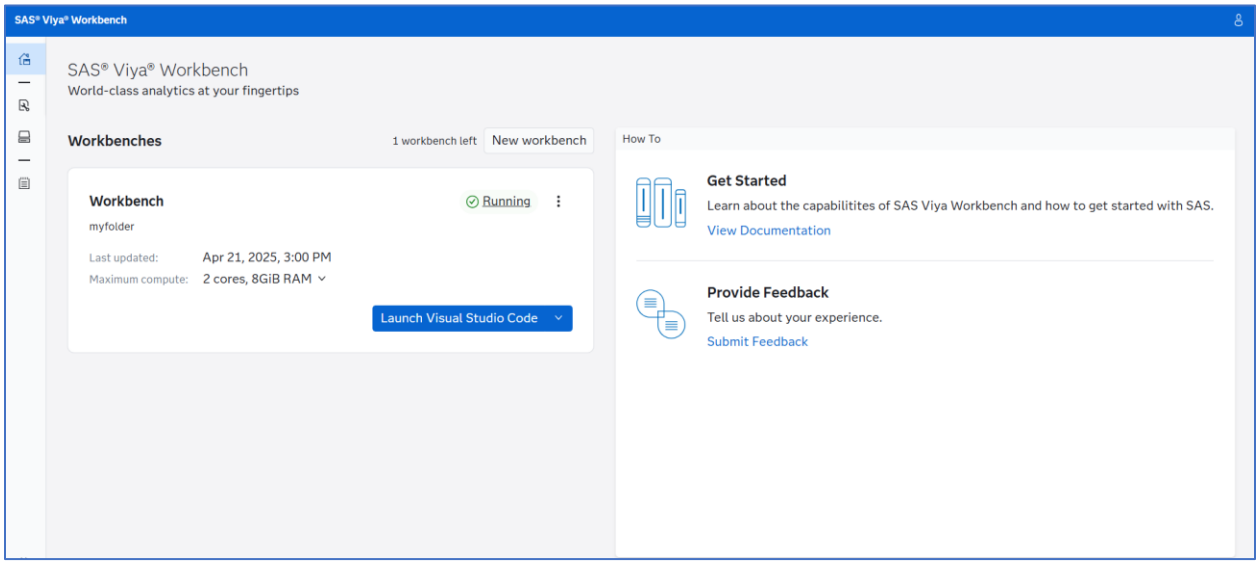
R you ready for the R portion of our analysis? And do you love a bad pun? Then you're in just the right place!

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 2023. 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.

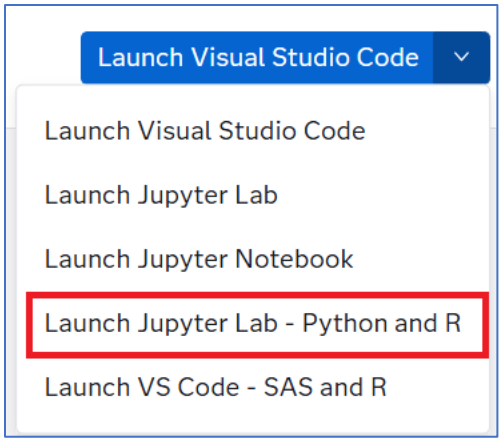
Let's finish our analysis strong!

Prepare the R Environment

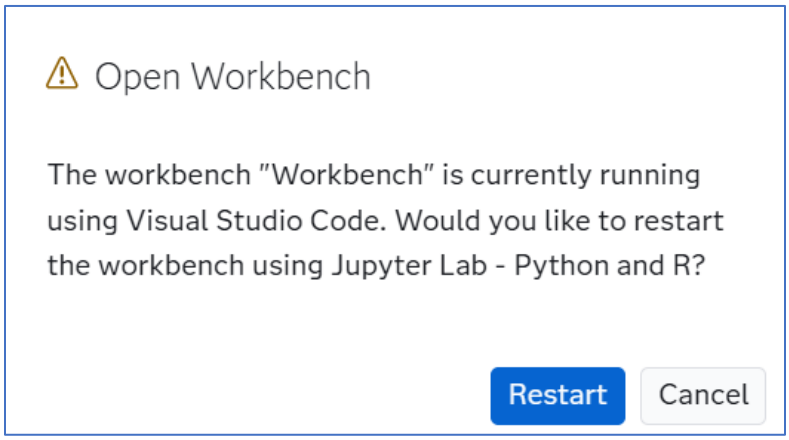
Unfortunately, SAS can't legally ship the R and Python kernels in the same IDE environment in SAS Viya Workbench for Learners… yet. Stay tuned on that front. So, for now, we'll need to continue our adventure by closing out of the existing notebook – in the usual way that you'd close any web browser. Return to the **SAS Viya Workbench for Learners** landing page:



Find that **Launch button** again and – just for the fun of it – let's choose the **Launch Jupyter Lab – Python and R** button, here:

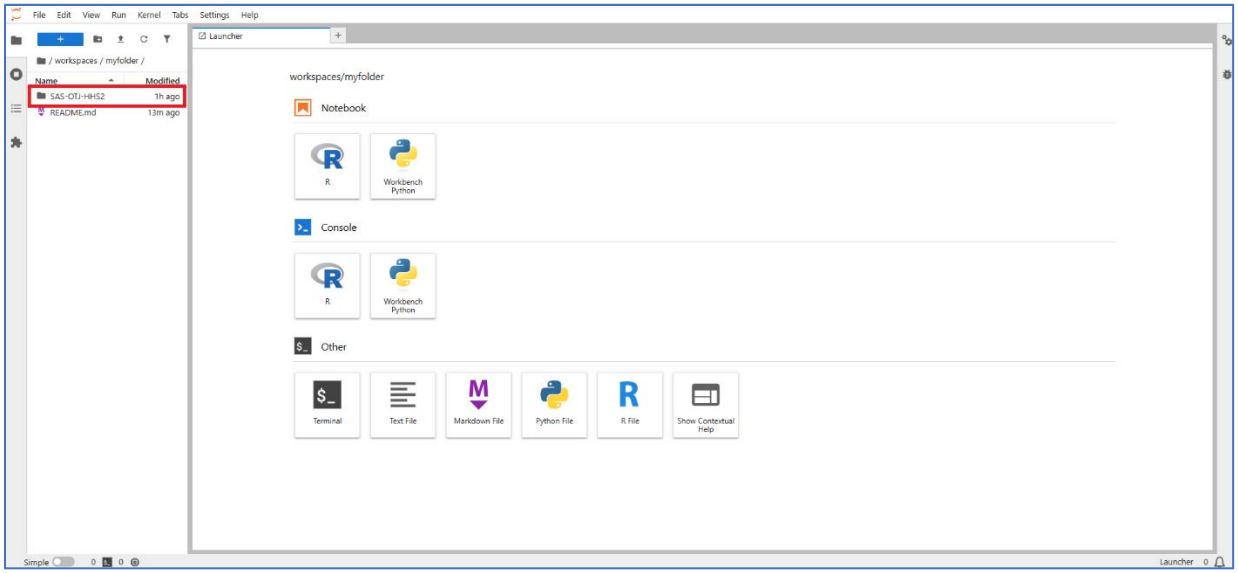


You'll likely get this warning message:

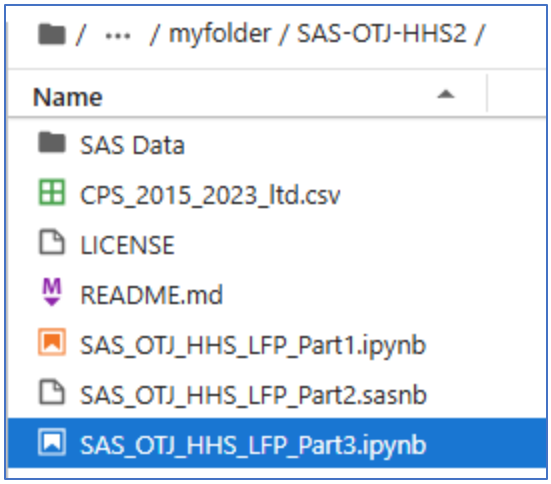


And, yes, it's perfectly fine to **Restart**!

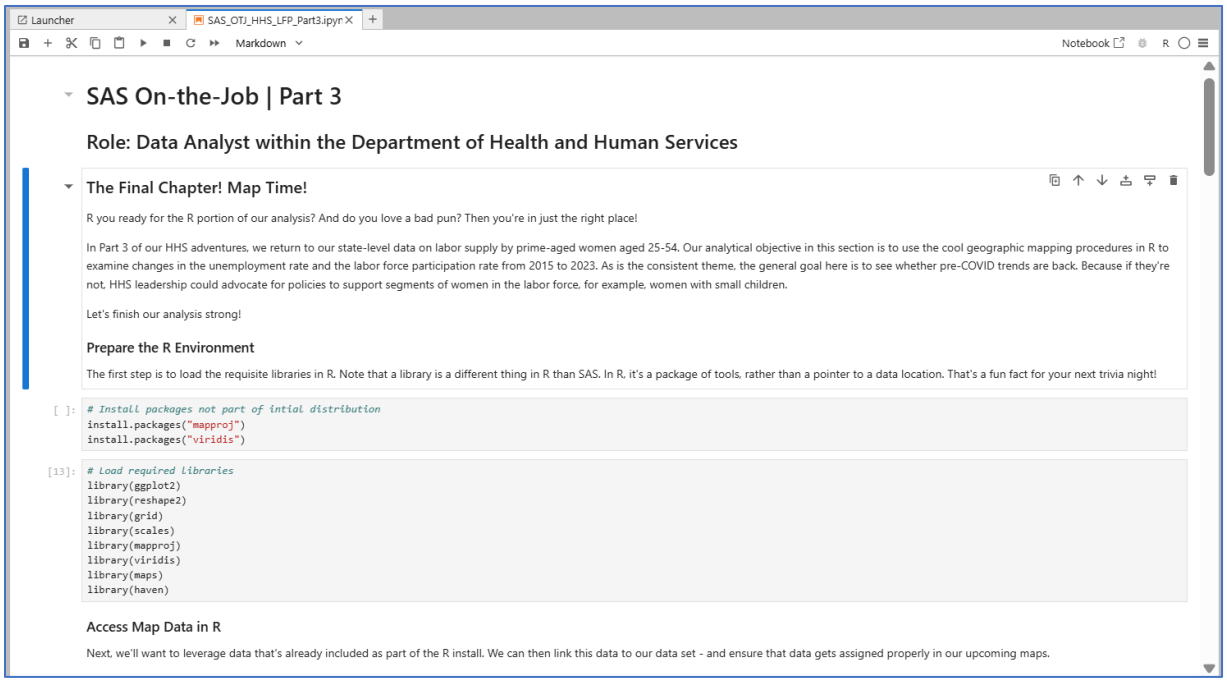
Let's check out a second beloved IDE – Jupyter Lab! And the **SAS-OTJ-HHS2** folder should be waiting for you:



Drill into **SAS-OTJ-HHS2** and find the *SAS_OTJ_HHS_LFP_Part3* file here:



And open it! Now we are rolling!



Once an R program is ready to go, the next step is to install some packages that weren't part of the original WFL distribution. A couple of lines of code will fix that:

```
[ ]: # Install packages not part of intial distribution
     install.packages("mapproj")
     install.packages("viridis")
```

Next, let's 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!

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

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 locations are assigned properly in our upcoming maps.

```
# 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.

```
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.

```
# Prepare the data from Part 2 for a merge
data1 <- read_sas('../SAS-OTJ-HHS2/SAS Data/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:

```
data1
```

State_FIP	State_Name	Year	UE_Women	LFP_Women	UE_Women_HS	UE_Women_SCollege	UE_Women_CollegeP	LFP_Women_HS	LFP_Women_SCollege	LFP_Women_CollegeP
<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	ALABAMA	2015	0.05903514	0.7120473	0.08071777	0.05814288	0.020941541	0.6368446	0.7686367	0.8354862
1	ALABAMA	2016	0.04944853	0.7001156	0.06312331	0.05633199	0.023824741	0.6306299	0.7564194	0.8039712
1	ALABAMA	2017	0.04078107	0.6901163	0.05017575	0.05711202	0.009858660	0.5927814	0.7514234	0.8062791
1	ALABAMA	2018	0.03973231	0.7044006	0.05505923	0.03617405	0.026229953	0.5949498	0.7550574	0.8102880
1	ALABAMA	2019	0.02619557	0.7215607	0.04950904	0.01513010	0.015084462	0.6365994	0.7471347	0.8495096
1	ALABAMA	2020	0.05666085	0.7270269	0.07461227	0.07200355	0.032396354	0.6264771	0.7516498	0.8416940
1	ALABAMA	2021	0.03245672	0.7124474	0.05794357	0.02744716	0.019250824	0.6256072	0.7531953	0.8061328
1	ALABAMA	2022	0.02901746	0.7142394	0.04939848	0.02008251	0.018836172	0.6472077	0.7381429	0.8119118
1	ALABAMA	2023	0.02414300	0.7300202	0.03631050	0.02379195	0.012366240	0.6452797	0.7684687	0.8428726
2	ALASKA	2015	0.05086221	0.7352526	0.08122067	0.04844351	0.034001258	0.6426811	0.7174740	0.8424400
2	ALASKA	2016	0.05125439	0.7620422	0.09696339	0.04429963	0.017883771	0.7117701	0.7309908	0.8514743
2	ALASKA	2017	0.06153648	0.7744281	0.09499766	0.07141625	0.025498735	0.7246573	0.7781355	0.8515758
2	ALASKA	2018	0.04122818	0.7621233	0.06473406	0.03100966	0.025967039	0.6694269	0.7786207	0.8590649
2	ALASKA	2019	0.03817501	0.7373931	0.06748475	0.02001494	0.019239370	0.6425734	0.7708428	0.8191131

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 CPS data set. Data populated all the way across the row tells us that the merge was done properly - so check for that:

```
# 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)
```

	region	long	lat	group	order	subregion	State_FIP	Year	UE_Women	LFP_Women	···	UE_Women_Co
	<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.

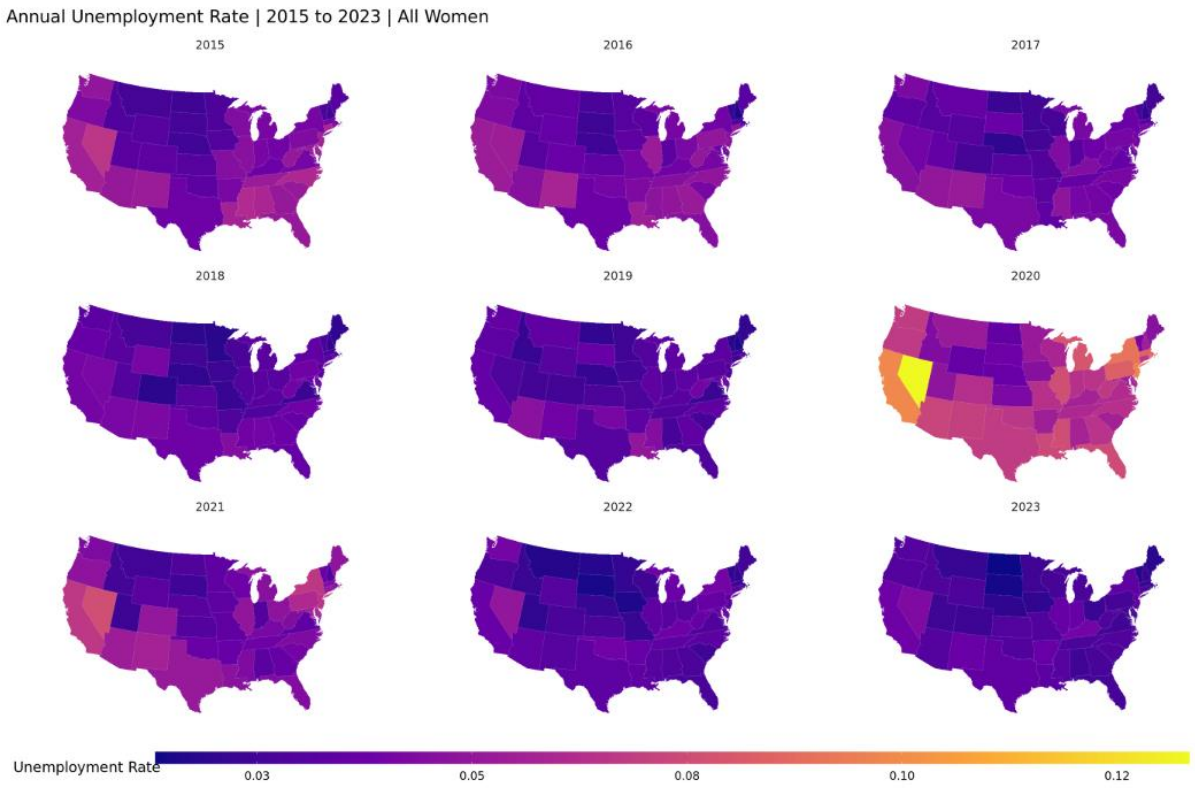
```
# Check this out (but don't ask me what each piece means)
# Source Code: https://socviz.co/index.html#preface ==> this book is great!
theme_map <- function(base_size=20, base_family="") {
  require(grid)
  theme_bw(base_size=base_size, base_family=base_family) %+replace%
    theme(axis.line=element_blank(),
          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, and we'll start with trends for all prime-aged women:

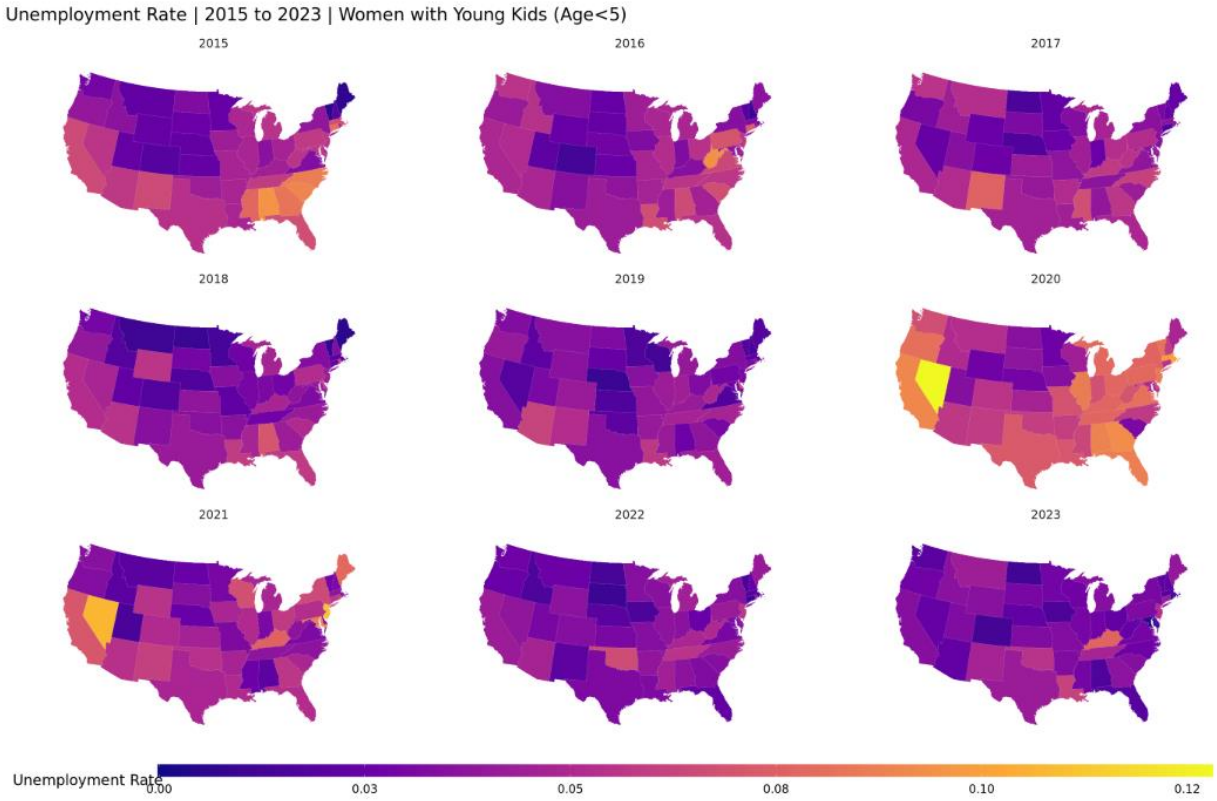
```
p0 <- ggplot(data = subset(state_map_data),
             mapping = aes(x = long, y = lat,
                           group = group,
                           fill = UE_Women))

p1 <- p0 + geom_polygon(color = "gray90", linewidth = 0.05) +
  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 2023 | All Women")
```



What do you see in these charts? 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 2023 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:

```
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 = 0.05) +
  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 2023 | Women with Young Kids (Age<5)")
```

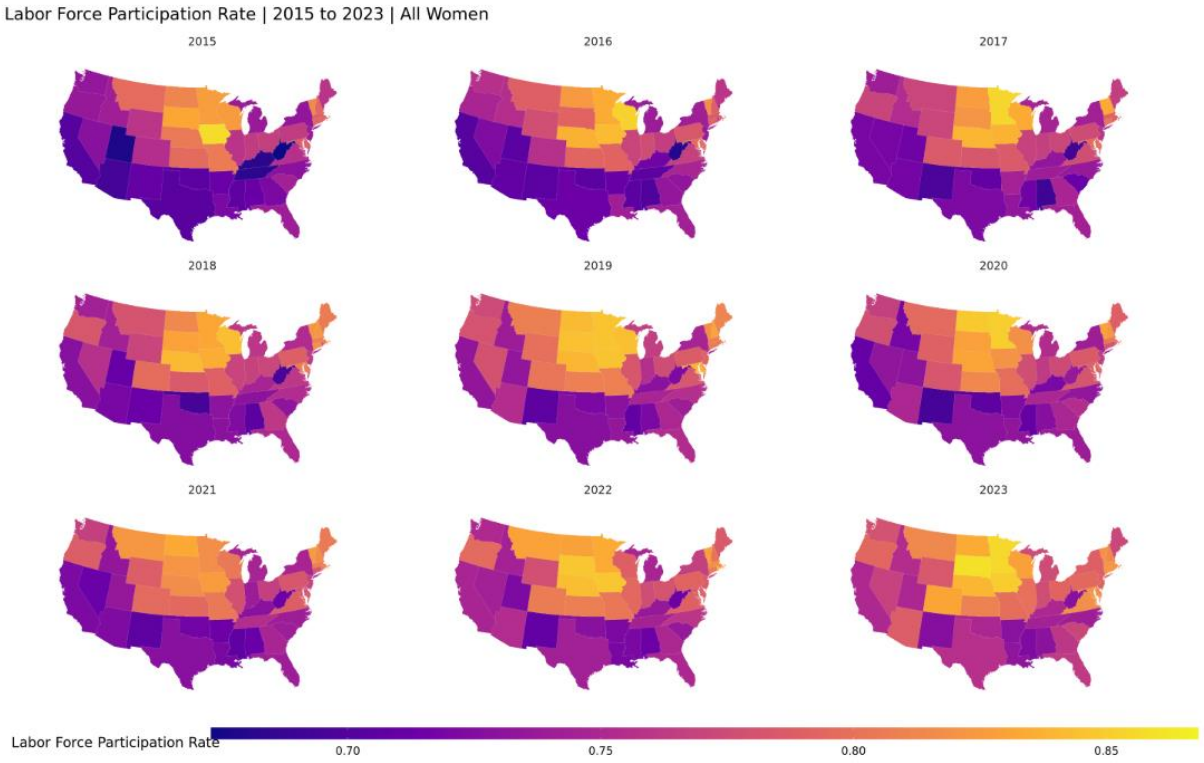


Again, a similar story plays out. Unemployment rates have returned to their pre-pandemic norms... and might be even better than they were in the early period.

Labor Force Participation Rate Analysis

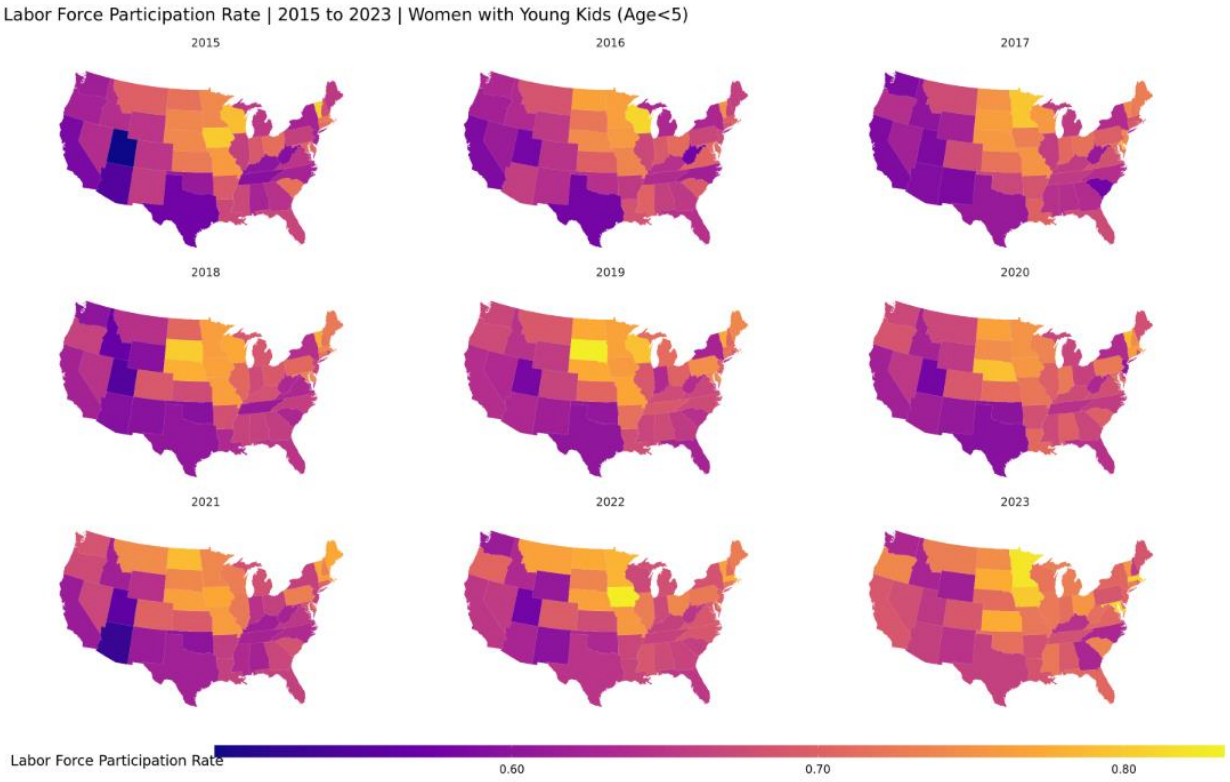
Because this code is so easy to modify after it's written, let's examine the labor force participant rate for all women. The story shouldn't change, but it is interesting to see the variation over time within states.

```
# 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 = 0.05) +
  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 = 3) +
  theme(legend.position = "bottom",
        strip.background = element_blank()) +
  labs(fill = "Labor Force Participation Rate ",
       title = "Labor Force Participation Rate | 2015 to 2023 | 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.

```
# 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 = 3) +
  theme(legend.position = "bottom",
        strip.background = element_blank()) +
  labs(fill = "Labor Force Participation Rate ",
       title = "Labor Force Participation Rate | 2015 to 2023 | 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!