```

////////////////////////////////////////////////////////////////
//                  MCC FY20 Data Cleaning File               //
//                          S&E Team                          //
////////////////////////////////////////////////////////////////

/*
The following file provides a step by step demonstration of the
process used by MCC's S&E Team to clean all of the raw data files
from indicator institutions and create the scorecards.  These steps
are outlined in MCC's FY20 Guide to the Indicators
(https://www.mcc.gov/resources/doc/guide-to-the-indicators-fy-2020)
and Data Notes
(https://www.mcc.gov/resources/doc/pub-fy2020-scorecard-data-notes)
This file is included to provide complete clarity as to exactly how
these indicators are constructed.

To use this file, you need to download the appropriate files from the
indicator institutions as described below, and set the global macros
to the correct locations.

This file will output individual files for each indicator.  You can run
sections of the program individually, but the section on GNI must be run
for the rest to work.

The official final data files are contained on MCC's open data site.
The purpose of this file is to provide systematic instructions explaining
the methodology used to create these files. Due to methodological changes
and historical data revisions by indicator institutions or MCC, this program
is not designed to recreate previous year’s scorecards. Rather it is designed
to clearly show MCC’s methodology in FY20. Updates will be made annually as needed.
*/

****************************
**  Preliminary Settings  **
****************************
** The following settings set up the necessary environment for the do file to run
** This includes programs which are run repeatedly in the file.

set more off
clear
////////////////////
//  Global Macros //
////////////////////

//To use this file you need to set the "globals" to the correct file paths on your computer
//raw_data is where you are storing the raw data, (instructions for download are included in each section below)
//created_data is where you would like the program to store temporary files
//final_data is where you would like the program to output the final files
global raw_data "WRITE_RAW_FILE_PATH_HERE" //Your filepath will probably be something like C:\Users\YOURUSERNAME\Documents\RAWDATAFOLDERNAME\
global created_data "WRITE_CREATED_FILE_PATH_HERE" //Your filepath will probably be something like C:\Users\YOURUSERNAME\Documents\CREATEDDATAFOLDERNAME\
global final_data "WRITE_FINAL_FILE_PATH_HERE" //Your filepath will probably be something like C:\Users\YOURUSERNAME\Documents\FINALDATAFOLDERNAME\


////////////////
//  Programs  //
////////////////
//The following sub-programs are used throughout the main program
//to rename countries and calculate percentiles.


**WBNaming
***********
//The following program is used to rename countries to a common set of names
//So that they can easily be merged together.

```

```
capture program drop WBNaming
program WBNaming
args country_variable

replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Côte d'Ivoire"
replace `country_variable' = "Sao Tome and Principe" if `country_variable' == "São Tomé and Principe"
replace `country_variable' = "Israeli Occupied Territories*" if `country_variable' == "Israeli Occupied Territories* " | `country_variable' == "Israeli-Occupied Territories*"
replace `country_variable' = "Kosovo" if `country_variable' == "Kosovo*"
replace `country_variable' = "Chechnya" if `country_variable' == "Chechnya*"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote dâ€™Ivoire"
replace `country_variable' = "Cyprus" if `country_variable' == "Cyprus "
replace `country_variable' = "Hong Kong*" if `country_variable' == "Hong Kong* "
replace `country_variable' = "Nagorno-Karabakh*" if `country_variable' == "Nagorno-Karabakh* "
replace `country_variable' = "Northern Cyprus" if `country_variable' == "Northern Cyprus*  "
replace `country_variable' = "Pakistani Kashmir" if `country_variable' == "Pakistani Kashmir*"
replace `country_variable' = "Palestinian Authority Administered Territories*" if `country_variable' == "Palestinian Authority Administered Territories* " | `country_variable' == "Palestinian Authority-Administered Territories*"
replace `country_variable' = "Puerto Rico" if `country_variable' == "Puerto Rico*"
replace `country_variable' = "South Ossetia" if `country_variable' == "South Ossetia*"
replace `country_variable' = "Transnistria" if `country_variable' == "Transnistria*"
replace `country_variable' = "Bahamas, The" if `country_variable' == "Bahamas"
replace `country_variable' = "Brunei Darussalam" if `country_variable' == "Brunei"
replace `country_variable' = "Cabo Verde" if `country_variable' == "Cape Verde"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo (Brazzaville)"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo (Kinshasa)"
replace `country_variable' = "Egypt, Arab Rep." if `country_variable' == "Egypt"
replace `country_variable' = "Iran, Islamic Rep." if `country_variable' == "Iran"
replace `country_variable' = "Kyrgyz Republic" if `country_variable' == "Kyrgyzstan"
replace `country_variable' = "Lao PDR" if `country_variable' == "Laos"
replace `country_variable' = "North Macedonia" if `country_variable' == "Macedonia"
replace `country_variable' = "Micronesia, Fed. Sts." if `country_variable' == "Micronesia"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "North Korea"
replace `country_variable' = "Russian Federation" if `country_variable' == "Russia"
replace `country_variable' = "Slovak Republic" if `country_variable' == "Slovakia"
replace `country_variable' = "Korea, Rep." if `country_variable' == "South Korea"
replace `country_variable' = "Syrian Arab Republic" if `country_variable' == "Syria"
replace `country_variable' = "Gambia, The" if `country_variable' == "The Gambia"
replace `country_variable' = "Venezuela, RB" if `country_variable' == "Venezuela"
replace `country_variable' = "Yemen, Rep." if `country_variable' == "Yemen"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Côte d'Ivoire"
replace `country_variable' = "Bahamas, The" if `country_variable' == "Bahamas"
replace `country_variable' = "Bolivia" if `country_variable' == "Bolivia (Plurinational State of)"
replace `country_variable' = "Hong Kong SAR, China" if `country_variable' == "China, Hong Kong Special Administrative Region"
replace `country_variable' = "Macao SAR, China" if `country_variable' == "China, Macao Special Administrative Region"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo"
replace `country_variable' = "Czech Republic" if `country_variable' == "Czechia"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Democratic People's Republic of Korea"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Democratic Republic of the Congo"
replace `country_variable' = "Egypt, Arab Rep." if `country_variable' == "Egypt"
replace `country_variable' = "Gambia, The" if `country_variable' == "Gambia"
replace `country_variable' = "Iran, Islamic Rep." if `country_variable' == "Iran (Islamic Republic of)"
replace `country_variable' = "Kyrgyz Republic" if `country_variable' == "Kyrgyzstan"
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao People's Democratic Republic"
replace `country_variable' = "Micronesia, Fed. Sts." if `country_variable' == "Micronesia (Federated States of)"
replace `country_variable' = "Korea, Rep." if `country_variable' == "Republic of Korea"
replace `country_variable' = "Moldova" if `country_variable' == "Republic of Moldova"
replace `country_variable' = "St. Kitts and Nevis" if `country_variable' == "Saint Kitts and Nevis"
replace `country_variable' = "St. Lucia" if `country_variable' == "Saint Lucia"
replace `country_variable' = "St. Vincent and the Grenadines" if `country_variable' == "Saint Vincent and the Grenadines"
replace `country_variable' = "Slovak Republic" if `country_variable' == "Slovakia"
replace `country_variable' = "United Kingdom" if `country_variable' == "United Kingdom of Great Britain and Northern Ireland"
replace `country_variable' = "Tanzania" if `country_variable' == "United Republic of Tanzania"
replace `country_variable' = "United States" if `country_variable' == "United States of America"
replace `country_variable' = "Venezuela, RB" if `country_variable' == "Venezuela (Bolivarian Republic of)"
replace `country_variable' = "Vietnam" if `country_variable' == "Viet Nam"
replace `country_variable' = "Yemen, Rep." if `country_variable' == "Yemen"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "Palestine"
replace `country_variable' = "Eswatini" if `country_variable' == "Swaziland"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "CÃƒÂ´te d'Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "CÃ´te d'Ivoire"
replace `country_variable' = "Syrian Arab Republic" if `country_variable' == "Syrian Arab Republic (the)"
replace `country_variable' = "Bahamas, The" if `country_variable' == "Bahamas (the)"
replace `country_variable' = "United Kingdom" if `country_variable' == "United Kingdom of Great Britain and Northern Ireland (the)"
replace `country_variable' = "Central African Republic" if `country_variable' == "Central African Republic (the)"
replace `country_variable' = "Comoros" if `country_variable' == "Comoros (the)"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo (the)"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Democratic People's Republic of Korea (the)"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Democratic Republic of the Congo (the)"
replace `country_variable' = "Dominican Republic" if `country_variable' == "Dominican Republic (the)"
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao People's Democratic Republic (the)"
replace `country_variable' = "Marshall Islands" if `country_variable' == "Marshall Islands (the)"
replace `country_variable' = "Netherlands" if `country_variable' == "Netherlands (the)"
replace `country_variable' = "Niger" if `country_variable' == "Niger (the)"
replace `country_variable' = "Philippines" if `country_variable' == "Philippines (the)"
replace `country_variable' = "Korea, Rep." if `country_variable' == "Republic of Korea (the)"
replace `country_variable' = "Moldova" if `country_variable' == "Republic of Moldova (the)"
replace `country_variable' = "Russian Federation" if `country_variable' == "Russian Federation (the)"
replace `country_variable' = "Sudan" if `country_variable' == "Sudan (the)"
replace `country_variable' = "United Arab Emirates" if `country_variable' == "United Arab Emirates (the)"
replace `country_variable' = "United States" if `country_variable' == "United States of America (the)"
replace `country_variable' = "Hong Kong SAR, China" if `country_variable' == "Hong Kong"
replace `country_variable' = "Sao Tome and Principe" if `country_variable' == "São Tomé and Príncipe"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Korea, Dem. Rep."
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo, The Democratic Republic of the"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo, Rep. of"
replace `country_variable' = "Tanzania" if `country_variable' == "Tanzania, United Republic of"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Korea, Democratic People's Republic of"
replace `country_variable' = "Micronesia, Fed. Sts." if `country_variable' == "Micronesia, Federated States of"
replace `country_variable' = "Bolivia" if `country_variable' == "Bolivia, Plurinational State of"
replace `country_variable' = "Moldova" if `country_variable' == "Moldova, Republic of"
replace `country_variable' = "Bosnia and Herzegovina" if `country_variable' == "Bosnia Herzegovina"
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao People's Dem. Rep."
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo, Democratic Republic"
replace `country_variable' = "Curaçao" if `country_variable' == "Curacao"
replace `country_variable' = "Timor-Leste" if `country_variable' == "East Timor"
replace `country_variable' = "Micronesia, Fed. Sts." if `country_variable' == "Federated States of Micronesia"
replace `country_variable' = "Kazakhstan" if `country_variable' == "Kazakstan"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Korea, Dem. People's Rep. of"
replace `country_variable' = "Libya" if `country_variable' == "Libyan Arab Jamahiriya"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "Palestinian Territory, Occupied"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "State of Palestine"
replace `country_variable' = "Reunion" if `country_variable' == "Réunion"
replace `country_variable' = "Saint-Barthelemy" if `country_variable' == "Saint-Barthélemy"
replace `country_variable' = "St. Martin (French part)" if `country_variable' == "Saint-Martin (French)"
replace `country_variable' = "St. Martin (French part)" if `country_variable' == "Saint-Martin (French part)"
replace `country_variable' = "Samoa" if `country_variable' == "Western Samoa"
replace `country_variable' = "North Macedonia" if `country_variable' == "The former Yugoslav Rep. of Macedonia"
replace `country_variable' = "Tanzania" if `country_variable' == "United Rep. of Tanzania"
replace `country_variable' = "Faroe Islands" if `country_variable' == "Faeroe Islands"
replace `country_variable' = "North Macedonia" if `country_variable' == "The former Yugoslav Republic of Macedonia"
replace `country_variable' = "Korea, Rep." if `country_variable' == "Korea, Republic of"
replace `country_variable' = "Macao SAR, China" if `country_variable' == "China Macao Special Administrative Region"
replace `country_variable' = "Hong Kong SAR, China" if `country_variable' == "China Hong Kong Special Administrative Region"
replace `country_variable' = "Virgin Islands (U.S.)" if `country_variable' == "United States Virgin Islands"
replace `country_variable' = "Reunion" if `country_variable' == "RÃ©union"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo, Republic of the"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo, Democratic Republic of"
replace `country_variable' = "Gambia, The" if `country_variable' == "Gambia (The)"
replace `country_variable' = "Central African Republic" if `country_variable' == "Central African Rep"
replace `country_variable' = "Cabo Verde" if `country_variable' == "Cabo Verde Republic of"
replace `country_variable' = "Bolivia" if `country_variable' == "Bolivia Plurinational States of "
replace `country_variable' = "Myanmar" if `country_variable' == "Burma"
replace `country_variable' = "West Bank and Gaza"  if `country_variable' ==  "Palestinian Authority Administered Territories*"
replace `country_variable' = "West Bank and Gaza"  if `country_variable' ==  "Palestinian Territories"
replace `country_variable' = "Caribbean Netherlands" if `country_variable' == "Bonaire Saint Eustatius and Saba"
replace `country_variable' = "South Sudan" if `country_variable' == "Republic of South Sudan"
replace `country_variable' = "Lebanon" if `country_variable' == "LebaNon"
replace `country_variable' = "Puerto Rico" if `country_variable' == "Puerto Rico (U.S.)"
replace `country_variable' = "North Macedonia" if `country_variable' == "Macedonia, FYR"
replace `country_variable' = "Taiwan" if `country_variable' == "Taiwan, China"
replace `country_variable' = "San Marino" if `country_variable' == "San MariNo"
replace `country_variable' = "Timor-Leste" if `country_variable' == "Timor Leste"
replace `country_variable' = "West Bank and Gaza"  if `country_variable' ==  "Occupied Palestinian Territory"
replace `country_variable' = "Czech Republic" if `country_variable' == "Czechia"
replace `country_variable' = "Guinea-Bissau" if `country_variable' == "Guinea Bissau"
replace `country_variable' = "Guinea-Bissau" if `country_variable' == "GuineaBissau"
replace `country_variable' = "North Macedonia" if `country_variable' == "The former Yugoslav Republic of Macedonia"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "State of Palestine"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "West Bank and Gaza Strip"
replace `country_variable' = "Timor-Leste" if `country_variable' == "TimorLeste"
replace `country_variable' = "United States" if `country_variable' == "United States of America"
replace `country_variable' = "Venezuela, RB" if `country_variable' == "Venezuela (Bolivarian Republic of)**"
replace `country_variable' = "Venezuela, RB" if `country_variable' == "Venezuela (Bolivarian Republic of)"
replace `country_variable' = "Northern Cyprus" if `country_variable' == "Northern Cyprus*"
replace `country_variable' = "Northern Cyprus" if `country_variable' == "Cyprus (Turkish)"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Ivory Coast"
replace `country_variable' = "Iran, Islamic Rep." if `country_variable' == "Islamic Republic of Iran"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "The Democratic Republic Of The Congo"
replace `country_variable' = "Northern Cyprus" if `country_variable' == "Cyprus North"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Korea, North"
replace `country_variable' = "Korea, Rep." if `country_variable' == "Korea, South"
replace `country_variable' = "Macao SAR, China" if `country_variable' == "Macau"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Republic of the Congo (Brazzaville)"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Dem. Rep. of the Congo"
replace `country_variable' = "Sao Tome and Principe" if `country_variable' == "São Tomé e Príncipe"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Dem. Rep. of Congo"
replace `country_variable' = "Antigua and Barbuda" if `country_variable' == "Antigua"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote d’Ivoire"
replace `country_variable' = "Tibet" if `country_variable' == "Tibet*"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Korea, Dem. People’s Rep."
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo, Democratic Republic of the Congo"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo, Republic of"
replace `country_variable' = "Korea, Dem. People's Rep." if `country_variable' == "Korea, North "
replace `country_variable' = "Taiwan" if `country_variable' == "Taiwan "
replace `country_variable' = "Eswatini" if `country_variable' == "eSwatini"
replace `country_variable' = "Bosnia and Herzegovina" if `country_variable' == "Bosnia"
replace `country_variable' = "Bosnia and Herzegovina" if `country_variable' == "Bosnia & Herz."
replace `country_variable' = "Bosnia and Herzegovina" if `country_variable' == "Bosnia And Herzegovina"
replace `country_variable' = "Bosnia and Herzegovina" if `country_variable' == "Bosnia-Herzegovina"
replace `country_variable' = "Myanmar" if `country_variable' == "Burma (Myanmar)"
replace `country_variable' = "Myanmar" if `country_variable' == "Burma/Myanmar"
replace `country_variable' = "Central African Republic" if `country_variable' == "Central Afr. Rep."
replace `country_variable' = "China" if `country_variable' == "China Mainland"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo, Dem. Rep. (Zaire)"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Congo, Democratic Republic Of (Kinshasa)"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo, Rep"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo, Republic Of (Brazzaville)"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote D'Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote D’Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote d'Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote d’Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "CôTe D'Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote d’Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Cote d’Ivoire"
replace `country_variable' = "Cyprus" if `country_variable' == "Cyprus (Greek)"
replace `country_variable' = "Czech Republic" if `country_variable' == "Czech Rep."
replace `country_variable' = "Czech Republic" if `country_variable' == "Czech Republic"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Dem. Rep. Congo"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Democratic Republic Of Congo"
replace `country_variable' = "Congo, Dem. Rep." if `country_variable' == "Democratic Republic Of The Congo"
replace `country_variable' = "Vietnam" if `country_variable' == "Democratic Republic Of Vietnam"
replace `country_variable' = "Dominican Republic" if `country_variable' == "Dom. Rep."
replace `country_variable' = "Dominican Republic" if `country_variable' == "Dominican Rep."
replace `country_variable' = "Dominican Republic" if `country_variable' == "Dominican Republic"
replace `country_variable' = "Guinea-Bissau" if `country_variable' == "Guinea Bissao"
replace `country_variable' = "Hong Kong SAR, China" if `country_variable' == "Hong Kong Sar, China"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "Israeli-Occupied Territories And Palestinian Authority"
replace `country_variable' = "Korea, Rep." if `country_variable' == "Korea"
replace `country_variable' = "Kyrgyz Republic" if `country_variable' == "Kyrgyz Rep."
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao PDR"
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao Pdr"
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao, Pdr"
replace `country_variable' = "Macao SAR, China" if `country_variable' == "Macao"
replace `country_variable' = "North Macedonia" if `country_variable' == "Macedonia, Fyr"
replace `country_variable' = "Micronesia, Fed. Sts." if `country_variable' == "Micronesia, Fs"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "Palestine/Gaza"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "Palestine/West Bank"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Republic Of The Congo"
replace `country_variable' = "Romania" if `country_variable' == "Roumania"
replace `country_variable' = "St. Vincent and the Grenadines" if `country_variable' == "Saint Vincent And The Grenadine"
replace `country_variable' = "St. Vincent and the Grenadines" if `country_variable' == "Saint Vincent And The Grenadines"
replace `country_variable' = "Sao Tome and Principe" if `country_variable' == "Sao Tome And Pr."
replace `country_variable' = "Serbia & Montenegro" if `country_variable' == "Serbia"
replace `country_variable' = "Serbia & Montenegro" if `country_variable' == "Serbia And Montenegro"
replace `country_variable' = "St. Kitts and Nevis" if `country_variable' == "St. Kitts And Nev."
replace `country_variable' = "St. Vincent and the Grenadines" if `country_variable' == "St. Vincent And The Grenadines"
replace `country_variable' = "South Sudan" if `country_variable' == "Sudan South"
replace `country_variable' = "Sao Tome and Principe" if `country_variable' == "SãO Tomé And Principe"
replace `country_variable' = "Trinidad And Tobago" if `country_variable' == "Trinidad And Tob."
replace `country_variable' = "United Arab Emirates" if `country_variable' == "UAE"
replace `country_variable' = "United Arab Emirates" if `country_variable' == "Uae"
replace `country_variable' = "Uraguay" if `country_variable' == "Uruguay *"
replace `country_variable' = "United States" if `country_variable' == "USA"
replace `country_variable' = "United States" if `country_variable' == "Usa"
replace `country_variable' = "Venezuela, RB" if `country_variable' == "Venezuela *"
replace `country_variable' = "Venezuela, RB" if `country_variable' == "Venezuela, Rb"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "West Bank Gaza"
replace `country_variable' = "West Bank and Gaza" if `country_variable' == "West Bank And Gaza"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Republic of Congo"
replace `country_variable' = "North Macedonia" if `country_variable' == "FYR Macedonia"
replace `country_variable' = "Hong Kong SAR, China" if `country_variable' == "Hong Kong SAR"
replace `country_variable' = "Lao PDR" if `country_variable' == "Lao P.D.R."
replace `country_variable' = "Taiwan" if `country_variable' == "Taiwan Province of China"
replace `country_variable' = "Bahamas, The" if `country_variable' == "The Bahamas"
replace `country_variable' = "Macao SAR, China" if `country_variable' == "Macao SAR"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "Côte d’Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "CÔTe D'Ivoire"
replace `country_variable' = "Congo, Rep." if `country_variable' == "Congo-Brazaville"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "CÃƒÂ´te d'Ivoire"
replace `country_variable' = "Cote d'Ivoire" if `country_variable' == "CÃ´te d'Ivoire"
replace `country_variable' = "Curaçao" if `country_variable' == "CuraÃ§ao"

end
```

```


**PercentRankInc
****************
//The following program mimics Excel's PercentRankInc function
//For indicators with a static cutoff (Civil Liberties, Political Rights,
//Immunization Rates for the higher scorecard pool and Inflation) this is done normally.
//However, for indicators with a median based cutoff, countries at the median are set to
//the 50th percentile, countries at the maximum as set to the 100th percentile
//and countries at the minimum are set to the 0th percentile

capture program drop PercentRankInc
program PercentRankInc
args vartest

generate over = 300  //Dummy value for now, this will be the number of countries scoring greater than or equal to a given country
generate under = 300  //Dummy value for now, this will be the number of countries scoring less than a given country
global count = _N //Number of observations
global vartest = "`vartest'"

forvalues line_change = 1(1)$count {  //This loops once for each country.
    global overcount = -1  //This counts the number of countries over the country in line_change.  It starts at -1 as we do not count the country itself as being greater than or equal to itself.
    forvalues line_test = 1(1)$count {  //This loops over all the countries again, comparing them each to the selected country from line_change
        if `=${vartest}[`line_change']' <= `=${vartest}[`line_test']' & `=${vartest}[`line_test']' != . {  //This if statement augments the overcount whenever a country from line_test is greater than or eqaul to the selected country from line_change
            global overcount = ${overcount} + 1
        }
    replace over = ${overcount} in `line_change' //This replaces the variable over with the overcount macro
    replace over = . in `line_change' if missing(`=${vartest}[`line_change']')  //This sets over to missing if the original value is missing.
    }
}

forvalues line_change = 1(1)$count { //This loops once for each country.
    global undercount = 0 //This counts the number of countries over the country in line_change. It starts at 0 since we are not testing less than or equal to, so we will not count the country against itself.
    forvalues line_test = 1(1)$count { //This loops over all the countries again, comparing them each to the selected country from line_change
        if `=${vartest}[`line_change']' > `=${vartest}[`line_test']' & `=${vartest}[`line_test']' != . {  //This if statement augments the undercount whenever a country from line_test is less than the selected country from line_change
            global undercount = ${undercount} + 1
        }
    replace under = ${undercount} in `line_change'  //This replaces the variable under with the undercount macro
    replace under = . in `line_change' if missing(`=${vartest}[`line_change']') //This sets over to missing if the original value is missing.
    }
}
gen percentile = under/(over+under)  //This calculates percentiles for all observations using the under and over variables.

//The if statement below forces countries at the median to the 50th percentile, at the max to the 100th percentile and at the min to the 0th percentile
//For all the median based indicators (see above for more information).
if "`vartest'" != "Civil_Liberties" & "`vartest'" != "Political_Rights" & "`vartest'" != "Inflation" & "`vartest'" != "Immunization_Rate_HISP" {
egen median = median(`vartest')
egen min = min(`vartest')
egen max = max(`vartest')
replace percentile = 0.5 if median == `vartest'
replace percentile = 1 if max == `vartest'
replace percentile = 0 if min == `vartest'
}
gen Score_In_FY20 = `vartest'
end

*****************************
**  Import/Clean Raw Data  **
*****************************
** The following sections take the necessary data files for MCC's Scorecard,
** import them, clean them and save the necessary files.  This section
** includes instructions for downloading these files as well as processing them.
** in comments before each section.

///////////
//  GNI  //
///////////
//Countries are divided into two scorecard pools based on GNI per capita.
//The following is the construction of that measure from World Bank data.

//Download Instructions:
/*
Download the xls file from:
https://datacatalog.worldbank.org/dataset/gni-capita-ranking-atlas-method-and-ppp-based
Check if columns and sheet names still match
Check if the footnotes match the income categories (When there is no GNI data
available for a country, in some cases the World Bank still provides a determination
of which income group that country would be in, there are captured in footnotes in the GNIPC file.)

Save in the ${raw_data} folder as GNIPC.xls

Note that MCC uses the July release.  Sometimes the World Bank will change data
 in between the July and the September release.  The version from the July release
 is available on MCC's Open Data Portal
*/

import excel "${raw_data}\GNIPC.xls", sheet("GNIpc") clear
//According the the World Bank, these countries fit MCC's definition of LISP and HISP
//Therefore the footnotes are translated into income categories
//D is country name, E is GNI, F is a footnote for countries missing GNI
//Sometimes the world bank will change which footnotes correspond to which income categories, check this carefully.
//LISP stands for Lower Income Scorecard Pool
//HISP stands for Higher Income Scorecard Pool
//These are divided by the Historic IDA threshold.
replace E = "LISP" if F =="k"
replace E = "HISP" if F =="l"
replace E = "UMIC" if F =="i"
replace E = "HIC" if F =="j"
keep D E
// The following are not listed as "Independent States" by the State Department,
// and therefore do not receive scorecards.  In the following sections, we limit the
// dataset by only keeping countries that match a country in this dataset.

drop if D == "" | D == "Economy" | D =="World" | D =="East Asia & Pacific" /*
*/| D =="Europe & Central Asia" | D =="Latin America & Caribbean" /*
*/ | D =="Middle East & North Africa" | D =="North America" | D =="South Asia" /*
*/| D =="Sub-Saharan Africa" | D =="Low income" | D =="Lower middle income" /*
*/| D =="Upper middle income" | D =="High income" | D=="West Bank and Gaza"

//These are placeholder values over the threshold for countries without data.
replace E = "15000" if E == "HIC"
replace E = "6000" if E == "UMIC"
replace E = "0" if E == "LISP"
replace E = "2500" if E == "HISP"
rename D Country
rename E GNI

//Now that all of the values for GNI are numeric, we can create a string variable "Income" Sorting them by threshold
```

```
//These threshold categories can be found in the following file labeled "Threshold"
//They can be different every year, and so will need to be changed for future iterations
destring, replace
generate Income = "LISP"
replace Income = "HISP" if GNI > 1925
replace Income = "UMIC" if GNI > 3995
replace Income = "HIC" if GNI > 12375

//This program standardizes all of the country names, so that they can be matched with other data files
WBNaming Country

save "${created_data}\GNI_FY20.dta", replace
export excel using "${final_data}\FY20_GNI.xlsx", firstrow(variables) replace

//////////////////
//  Thresholds  //
/////////////////
//Countries are divided into two scorecard income pools based on GNI per capita.
//The cutoff for these pools is the Historic IDA threshold from the World Bank
//The following section demonstrates how these are constructed from the World Bank Data.
//This is where the cutoffs for the Income variable in the previous section come from.

//Download Instructions:
/*
Download the historical classification by income in XLS format from
https://datahelpdesk.worldbank.org/knowledgebase/articles/906519-world-bank-country-and-lending-groups
Check if columns and sheet names still match
Save in the ${raw_data} folder as OGHIST.xls
*/
import excel "${raw_data}\OGHIST.xls", sheet("Thresholds") clear
keep A S-AU
keep if A == "Date:" |A == "Bank's fiscal year:" | A == "II. IDA Eligibility /2  /7" | A == "Lower middle income" | A == "Upper middle income"
sort A
replace A = "Threshold" if A == "Bank's fiscal year:"
//You may need to install renvars. Enter "help renvars" and install the update.
//This command renames the variables with the first row of observations.
renvars, map (word(@[1],1))
drop in 1
foreach var of varlist _all {
replace `var' = strreverse(substr(strreverse(`var'),1,4)) in 1  //Takes the year (last 4 digits) only from each value in the first row
replace `var' = substr(`var',(strpos(`var'[2],"=")+2),.) in 2  //Isolates the cutoff value for the second row (the cutoff between the Lower Income Scorecard Pool and the Higher Income Scorecard Pool)
replace `var' = substr(`var',(strpos(`var'[3],"-")+1),.) in 3  //Isolates the cutoff value for the third row (the cutoff between LMICs and UMICs)
replace `var' = substr(`var',(strpos(`var'[4],"-")+1),.) in 4  //Isolates the cutoff value for the fourth row (the cutoff between UMICs and HICs)
replace `var' = subinstr(`var',",","",.)  //Removes commas from all numbers so they can be converted to numeric values.
}
destring, replace

xpose, clear varname  //Reshapes data
label variable v1 "Year"
label variable v2 "LISP"
label variable v3 "HISP"
label variable v4 "UMIC"
label variable _varname "Fiscal Year"
drop in 1
order _varname
save "${created_data}\Threshold_FY20.dta", replace
export excel using "${final_data}\FY20_Threshold.xlsx", firstrow(varlabels) replace

//////////////////
//  Population  //
//////////////////
```

```
//Population is not used to determine scorecard passage.
//However, it is included on the scorecard as a descriptive statistic in the upper left corner,
// so the method for constructing it is listed below

//Download Instructions:
/*
Go to https://data.worldbank.org/indicator/SP.POP.TOTL
Download the excel file
Save it as World Bank Population Data.xls in the raw data folder
*/
import excel "${raw_data}\World Bank Population Data.xls", sheet("Data") clear
//Dropping unnecessary observations with datasource information
drop in 3
drop in 2
drop in 1
//Formatting the first row of variables to use renvars
foreach var of varlist _all{
label var `var' "`=`var'[1]'"
replace `var' = "Y" +`var'[1] in 1
}
foreach var of varlist _all{
replace `var' = subinstr("`=`var'[1]'"," ", "_",.) in 1
}
renvars, map (word(@[1],1))
drop in 1
destring, replace
rename YCountry_Name Country
drop YCountry_Code YIndicator_Name YIndicator_Code
WBNaming Country
//This takes out all of the things that are not considered "Independent States" by the State Department
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keepusing() keep(3) //This removes those observations which were not included in the GNI data above.
drop GNI
keep Country Y2019
export excel using "${final_data}\FY20_Population.xlsx", firstrow(varlabels) replace


/////////////////////////////////////////
//  Democratic Rights Indicators (FH)  //
/////////////////////////////////////////
//The Democratic Rights Indicators are Political Rights and Civil Liberties
//They are both sourced from Freedom House's Freedom In The World report.
//Download Instructions
/*
Download the Aggregate Category and Subcategory Scores, 2003-2019 (Excel) file from
https://freedomhouse.org/report-types/freedom-world
Check if columns and sheet names still match the lines below (i.e. is the sheet still
named "FIW06-19" and the variable for country/territory still named CT)
Save in the ${raw_data} folder as Aggregate Category and Subcategory Scores FIW2003-2018.xlsx
*/
set more off
import excel "${raw_data}\Aggregate_Category_and_Subcategory_Scores_FIW2019.xls", sheet("FIW06-19") firstrow clear
keep if CT == "c" //Only keeping countries
keep if Edition == 2019 //Only keeping current year
keep CountryTerritory PR CL //Only keeping country name and relevant data values (PR=Political Rights, CL=Civil Liberties)
rename CountryTerritory Country

**To Merge with GNI data
************************
WBNaming Country //Renaming countries to match GNI data
save "${created_data}\FH_FY20.dta", replace

merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)  //This only keeps countries included in the GNI data

keep Country Income CL PR
```

```
rename CL Civil_Liberties
rename PR Political_Rights
save "${created_data}\FH Full.dta", replace

//Percentile are not used for determining passage on this indicator, but they are included on the scorecard
//The following computes the percentile ranks for the Democratic Rights Indicators
//It is worthy of note that MCC's general rule of forcing countries at the median to the 50th percentile and
// countries at the max to the 100th percentile does not apply here since percentiles are not used in the calculation of passage

global FH_vars "Civil_Liberties Political_Rights"
foreach var of global FH_vars {
//Percentile Ranks for LISPs
preserve
keep if Income == "LISP"
PercentRankInc `var'  //This program is listed above, and calculates the percentile rank using MCC's Methodology
save "${created_data}\FH `var' LISP.dta", replace
restore

//Percentile Ranks for HISPs
preserve
keep if Income == "HISP"
PercentRankInc `var'

//Appending other scorecard pool data
append using "${created_data}\FH `var' LISP.dta"
keep Country percentile Score_In_FY20 Income
gen Indicator = "`var'"
rename percentile Percentile
if "`var'" == "Civil_Liberties" {  //We need to use an if statement here because we can't put a local macro at the beginning of the filename
save "${created_data}\Civil Liberties.dta", replace
}
else {
save "${created_data}\Political Rights.dta", replace
}

export excel "${final_data}\FY20_`var'.xlsx", firstrow(variables) replace
restore
}


////////////////////////////
//  Education Indicators  //
////////////////////////////

//Download Instructions:
/*
Go to http://data.uis.unesco.org/
On the left panel select Education > Education > Education (full dataset)
Click Customise > Selection > Indicator
Click Unselect Items > Unselect all then Collapse all
Click Indicator > Financial Resources > Government Expenditure on Education >
> Government Expenditure on Education as a percentage of GDP
> Government Expenditure on Primary Education as a Percentage of GDP (%)

Then select Completion > Completion and Graduation Ratios >
> Gross Intake Ratio to Last Grade > Gross intake ratio to the last grade of primary education, female (%)

Then select Participation > Enrollment Ratios >
> Gross Enrollment Ratio by Level of Education >
> Gross enrollment ratio, lower secondary, female (%)

Click Time > Select latest data > Select last 50 years

Click View Data
```

```
Go to Export > Text File (CSV) > Download

Check if columns and sheet names still match
Save in the ${raw_data} folder as Education Data YEAR.csv.xls
*/
set more off
import delimited "${raw_data}\Education Data 2019.csv", clear
drop v6
//These countries would be removed when the GNI data is merged anyway, but we need to drop them early due to the way the reshaping process handles duplicates
drop if country == "World" | country == "Latin America and the Caribbean" | country == "Small Island Developing States" /*
*/ | country == "Africa (Northern)" | country == "Africa (Sub-Saharan) " | country == "Arab States" /*
*/ | country == "Arab States" | country == "Asia (Central and Southern)" | country == "Asia (Central)" /*
*/ | country == "Asia (Eastern and South-eastern)" | country == "Asia (Eastern)" | country == "Asia (South-eastern)" /*
*/ | country == "Asia (Southern)" | country == "Asia (Western)" | country == "Africa (Sub-Saharan)" | country == "East Asia and the Pacific" /*
*/ | country == "Central Asia" | country == "Central and Eastern Europe" | country == "High income countries" | country == "Landlocked Developing Countries" /*
*/ | country == "Least Developed Countries" | country == "Low income countries" | country == "Lower middle income countries" | country == "Middle income countries" /*
*/ | country == "North America and Western Europe" | country == "Northern America" | country == "Northern America and Europe" | country == "Oceania" /*
*/ | country == "Oceania (Australia/New Zealand)" | country == "Oceania (excl. Australia/New Zealand)" | country == "South and West Asia" | country == "Sub-Saharan Africa" /*
*/| country == "Upper middle income countries" | country == "Western Asia and Northern Africa" | country == "Europe" | country == "Northern America" | country == "Northern America "



//Renaming countries for consistency
WBNaming country
rename time year

////Reshaping the data into a "wide" format so it is easier to use
**EdExpense
preserve
keep if indicator == "Government expenditure on primary education as a percentage of GDP (%)"
keep country year value
reshape wide value, i(country) j(year)
save "${created_data}\EdExpense FY20.dta" , replace
restore

**Primary
preserve
keep if indicator == "Gross intake ratio to the last grade of primary education, female (%)"
keep country year value
reshape wide value, i(country) j(year)
save "${created_data}\Primary FY20.dta" , replace
restore

**Secondary
keep if indicator == "Gross enrolment ratio, lower secondary, female (%)"
keep country year value
reshape wide value, i(country) j(year)
save "${created_data}\Secondary FY20.dta" , replace


**Merging
*********
//For these indicators, MCC takes the most recent data point from the last 7 years of data (including the current year, 2019)
//Other documentation states that we take the last 6 years of data, this is because only one country has data for 2019
//Therefore it is the 6 most recent years of data starting in 2018 or the 7 most recent
```

```
starting in 2019
//The following code takes the most recent value for each indicator in the last 7 years (since 2013)
set more off
//Ed Expenditures
use "${created_data}\EdExpense FY20.dta", clear
keep country value2013-value2019
egen EdExpense = rowlast(value*)  //Using the last (most recent) value, note that the variables must be sorted chronologically for this to work
keep country EdExpense
//Primary Completion
merge 1:1 country using "${created_data}\Primary FY20.dta" , nogen
keep country EdExpense value2013-value2019 //Limiting to the last 7 years
egen Primary = rowlast(value*) //Taking the most recent data
keep country Primary EdExpense
//Secondary Enrollment
merge 1:1 country using "${created_data}\Secondary FY20.dta", nogen
keep country Primary EdExpense value2013-value2019
egen Secondary = rowlast(value*)
keep country Secondary EdExpense Primary
rename country Country
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta" , nogen keep(3)

global EdIndicators "Secondary Primary EdExpense"
save "${created_data}\Ed Full.dta", replace

//The following sections calculate percentile ranks for each of the three education indicators and generate the files:
preserve
keep if Income == "LISP"
PercentRankInc EdExpense
save "${created_data}\Ed EdExpense LISP.dta", replace
restore
preserve
keep if Income == "HISP"
PercentRankInc EdExpense
append using "${created_data}\Ed EdExpense LISP.dta"
keep Country Income percentile Score_In_FY20
gen Indicator = "Primary_Education_Expenditures"
rename percentile Percentile
save "${created_data}\Primary Education Expenditures.dta", replace
export excel using "${final_data}\FY20_Primary_Education_Expenditures.xlsx", firstrow(variables) replace
restore

preserve
keep if Income == "LISP"
PercentRankInc Primary
keep Country Income percentile Score_In_FY20
gen Indicator = "Primary_Girls_Education_Completion"
rename percentile Percentile
save "${created_data}\Primary Girls Education Completion.dta", replace
export excel using "${final_data}\FY20_Primary_Girls_Education_Completion.xlsx", firstrow(variables) replace
restore

preserve
keep if Income == "HISP"
PercentRankInc Secondary
keep Country Income percentile Score_In_FY20
gen Indicator = "Secondary_Girls_Education_Enrollment"
rename percentile Percentile
save "${created_data}\Secondary Girls Education Enrollment.dta", replace
export excel using "${final_data}\FY20_Secondary_Girls_Education_Enrollment.xlsx", firstrow(variables) replace
restore


//////////////////////////////////
```

```
// Immunization Rate Indicator  //
///////////////////////////////////
//Download Instructions
/*
Download from https://www.who.int/immunization/monitoring_surveillance/data/en/
Under 4. Immunization coverage or administered doses, select:
4.6 WHO/UNICEF Estimates of National Immunization Coverage (excel)

Check if columns and sheet names still match (i.e. is the sheet still called MCV1 and the Country variable still called Cname)
Save in the ${raw_data} folder as WHO coverage_estimates_series.xls
*/
//This indicator is an average of two measures of immunization: MCV1 and DTP3
//The following code performs this calculation and outputs the final file.
set more off
import excel "${raw_data}\WHO coverage_estimates_series.xls", sheet("MCV1") clear

//MCV1
foreach var of varlist E-AQ {  //This loop allows the first row to be renamed as Y{year} (e.g. Y2016) since Stata cannot take numbers as variable names
replace `var' = "Y" +`var'[1] in 1
}
renvars, map (word(@[1],1))
drop in 1
destring, replace
rename Cname Country
keep Country Y*
WBNaming Country
order _all, sequential
keep Country Y2018
save "${created_data}\Immunization MCV FY20.dta" , replace

//DPT3
import excel "${raw_data}\WHO coverage_estimates_series.xls", sheet("DTP3") clear
foreach var of varlist E-AQ { //This loop allows the first row to be renamed as Y{year} (e.g. Y2016) since Stata cannot take numbers as variable names
replace `var' = "Y" +`var'[1] in 1
}
renvars, map (word(@[1],1))
drop in 1
destring, replace
rename Cname Country
keep Country Y*
WBNaming Country
order _all, sequential
keep Country Y2018
save "${created_data}\Immunization DTP FY20.dta" , replace

**Merging
*********
//Merging both files and then merging them with the GNI data
set more off
use "${created_data}\Immunization MCV FY20.dta", clear
keep Country Y2018
rename Y2018 MCV1
merge 1:1 Country using "${created_data}\Immunization DTP FY20.dta"
keep Country Y2018 MCV1
rename Y2018 DTP3
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

egen Score = rowmean(MCV1 DTP3)  //Taking the average of the two values

save "${created_data}\Imm Full.dta", replace

preserve
keep if Income == "LISP"
rename Score Immunization_Rate_LISP
PercentRankInc Immunization_Rate_LISP
save "${created_data}\Imm LISP.dta", replace
```

```

restore
keep if Income == "HISP"
rename Score Immunization_Rate_HISP
PercentRankInc Immunization_Rate_HISP
append using "${created_data}\Imm LISP.dta"
keep Country Income percentile Score_In_FY20
gen Indicator = "Immunization_Rate"
rename percentile Percentile
save "${created_data}\Immunization Rate.dta", replace
export excel using "${final_data}\FY20_Immunization_Rate.xlsx", firstrow(variables) replace


///////////////////////////
//  Health Expenditures  //
///////////////////////////

//Download Instructions
/*
Go to http://apps.who.int/nha/database/Select/Indicators/en
Select Indicators > Financing Sources >
Domestic General Government Health Expenditure (GGHE-D) as % of Gross Domestic Product (GDP)
Click Next, select all countries, click next
Select all years, click next
Select % Gross domestic product (GDP)
click view data and build report
Click Excel Workbook

Save as NHA indicators.xlsx in the raw data folder
check to see if the columns and rows still match.
*/
set more off
import excel "${raw_data}\NHA indicators.xlsx", sheet("Table") clear
drop B C
drop in 2

foreach var of varlist D-T {  //replacing the first row with values that have characters in them so that variables can be renamed using renvars
replace `var' = "Y" +`var'[1] in 1
}
renvars, map (word(@[1],1))
drop in 1
rename Countries Country
WBNaming Country
destring, replace
save "${created_data}\Health FY20.dta", replace

**Merging
*********
keep Country Y2016  //Keep the most recent year
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)
save "${created_data}\Health Full.dta", replace

preserve
keep if Income == "LISP"
PercentRankInc Y2016
save "${created_data}\Health LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc Y2016
save "${created_data}\Health HISP.dta", replace
append using "${created_data}\Health LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Health_Expenditures"
rename percentile Percentile
save "${created_data}\Health Expenditures.dta", replace
export excel using "${final_data}\FY20_Health_Expenditures.xlsx", firstrow(variables) replace

```

```
/////////////////////////////
//  Gender in the Economy //
/////////////////////////////


//Download Instructions:
/*
Go to Women, Business and the Law.  Go to their reports page: https://wbl.worldbank.org/en/reports
Find the most recent report with new data, and click on data
Save as the file referenced below in the raw data folder

*/

set more off
import excel "${raw_data}\wblrawdata2010201829march2018.xlsx", sheet("WBL2018") clear
set more off
tostring _all, replace
replace AT = "Can an unmarried woman be HoH like an unmarried man?" in 1
replace AU = "Can a married woman be HoH like a married man?" in 1

 //The following loop cleans up the first row into variable names that are not too long for Stata to handle

foreach var of varlist _all {
if "`=`var'[1]'" == "." {
drop `var'
}
else {
//drop `var' if `var' == "."
replace `var' = subinstr("`=`var'[1]'", `""""', "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "If customary law is recognized as a valid source of law under the constitution, is it invalid if it violates constitutional provisions on nondiscrimination or equality?", "If vld cust law, does const invldate if disc?",.) in 1
replace `var' = subinstr("`=`var'[1]'", "If personal law is recognized as a valid source of law under the constitution, is it invalid if it violates constitutional provisions on nondiscrimination or equality?", "If vld personal law, does const invldate if disc?",.) in 1


replace `var' = subinstr("`=`var'[1]'", " (reserved seats) in place", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "legislative", "Lgs.",.) in 1
replace `var' = subinstr("`=`var'[1]'", "sanctions for noncompliance", "sanc. for ncomp.",.) in 1
replace `var' = subinstr("`=`var'[1]'", "with mandated quotas", "w/quotas",.) in 1
replace `var' = subinstr("`=`var'[1]'", "(e.g. financial)", "($)",.) in 1
replace `var' = subinstr("`=`var'[1]'", "political parties", "pol. part.",.) in 1
replace `var' = subinstr("`=`var'[1]'", "women on candidate lists", "w. cands",.) in 1
replace `var' = subinstr("`=`var'[1]'", "nondiscrimination", "ndiscrm",.) in 1
replace `var' = subinstr("`=`var'[1]'", "constitution", "const",.) in 1
replace `var' = subinstr("`=`var'[1]'", "customary", "cust",.) in 1
replace `var' = subinstr("`=`var'[1]'", "explicitly", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "recognized", "recg",.) in 1
replace `var' = subinstr("`=`var'[1]'", "valid", "vld",.) in 1
replace `var' = subinstr("`=`var'[1]'", "explicitly", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "unmarried", "umar",.) in 1
replace `var' = subinstr("`=`var'[1]'", "married", "mar",.) in 1
replace `var' = subinstr("`=`var'[1]'", " or pursue a trade or profession", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "children", "kids",.) in 1
replace `var' = subinstr("`=`var'[1]'", "non-national", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "legal", "leg",.) in 1
replace `var' = subinstr("`=`var'[1]'", "responsibility", "resp",.) in 1
replace `var' = subinstr("`=`var'[1]'", "financially", "fin",.) in 1
replace `var' = subinstr("`=`var'[1]'", "husband", "hus",.) in 1
replace `var' = subinstr("`=`var'[1]'", "administers", "admns",.) in 1
replace `var' = subinstr("`=`var'[1]'", "property", "prop",.) in 1
replace `var' = subinstr("`=`var'[1]'", "formal", "frml",.) in 1
replace `var' = subinstr("`=`var'[1]'", "their decisions be appealed", "they be appealed",.) in 1
replace `var' = subinstr("`=`var'[1]'", "recognize", "recog",.) in 1
replace `var' = subinstr("`=`var'[1]'", "personal", "prsnl",.) in 1
replace `var' = subinstr("`=`var'[1]'", "court", "crt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "private credit bureau", "priv cred bur",.) in 1
replace `var' = subinstr("`=`var'[1]'", "public credit registry", "pub cred reg",.) in 1
replace `var' = subinstr("`=`var'[1]'", "public credit registries", "pub cred regs",.) in 1
replace `var' = subinstr("`=`var'[1]'", "information", "info",.) in 1
replace `var' = subinstr("`=`var'[1]'", "retailers", "rtlrs",.) in 1
replace `var' = subinstr("`=`var'[1]'", "microfinance", "microfin",.) in 1
replace `var' = subinstr("`=`var'[1]'", "institutions", "insts",.) in 1
replace `var' = subinstr("`=`var'[1]'", "institution", "inst",.) in 1
replace `var' = subinstr("`=`var'[1]'", "discrimination", "disc",.) in 1
replace `var' = subinstr("`=`var'[1]'", "on the basis of", "on",.) in 1
replace `var' = subinstr("`=`var'[1]'", "nonpregnant and nonnursing", "npreg/nonurse",.) in 1
replace `var' = subinstr("`=`var'[1]'", "protection", "prtctn",.) in 1
replace `var' = subinstr("`=`var'[1]'", "distance", "dist",.) in 1
replace `var' = subinstr("`=`var'[1]'", "maintaining", "maintng",.) in 1
replace `var' = subinstr("`=`var'[1]'", "cover prohibition", "prhbt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "authorizing or knowingly entering into", "authrzng/knwinly marrying",.) in 1
replace `var' = subinstr("`=`var'[1]'", "penalties in the law", "penalties",.) in 1

label var `var' "`=`var'[1]'"

replace `var' = subinstr("`=`var'[1]'", " ", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "/", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", ",", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", ".", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "?", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "(", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", ")", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", `"""', "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "'", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "`", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "-", "",.) in 1


replace `var' = subinstr("`=`var'[1]'", "Does_the_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "What_are_the_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "women_representatives", "w_cnds",.) in 1
replace `var' = subinstr("`=`var'[1]'", "in_elections_for_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Are_there_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "for_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Can_an_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Can_a_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "citizenship", "ctzsh",.) in 1
replace `var' = subinstr("`=`var'[1]'", "confer", "cfr",.) in 1
replace `var' = subinstr("`=`var'[1]'", "prohibit", "prbt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "creditor", "crdr",.) in 1
replace `var' = subinstr("`=`var'[1]'", "What_percentage", "perc",.) in 1
replace `var' = subinstr("`=`var'[1]'", "wages", "wgs",.) in 1
replace `var' = subinstr("`=`var'[1]'", "paid", "pd",.) in 1
replace `var' = subinstr("`=`var'[1]'", "How_many_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "parental", "prtl",.) in 1
replace `var' = subinstr("`=`var'[1]'", "leave", "lv",.) in 1
replace `var' = subinstr("`=`var'[1]'", "must_be_taken_by_the_", "mst_tak",.) in 1
replace `var' = subinstr("`=`var'[1]'", "mandate", "mdt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "based_on", "on",.) in 1
replace `var' = subinstr("`=`var'[1]'", "What_is_the_age", "age",.) in 1
replace `var' = subinstr("`=`var'[1]'", "at_which_a_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "and_recieve_full", "full",.) in 1
replace `var' = subinstr("`=`var'[1]'", "and_recieve_partial", "partial",.) in 1
replace `var' = subinstr("`=`var'[1]'", "can_retire", "retire",.) in 1
replace `var' = subinstr("`=`var'[1]'", "What_is_the_mandatory_", "mndt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "retirement", "retrmt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Can_npregnonurse_women", "nprg_nse_w",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Does_domestic_violence_legislation", "dom_vio_leg",.) in 1
replace `var' = subinstr("`=`var'[1]'", "legislation", "leg",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Is_there_", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "sexual_harassment", "sx_hrsmt",.) in 1
replace `var' = subinstr("`=`var'[1]'", "What_is_the_leg", "leg",.) in 1
replace `var' = subinstr("`=`var'[1]'", "What_is_the_minimum", "min",.) in 1
replace `var' = subinstr("`=`var'[1]'", "marriage", "marr",.) in 1
replace `var' = subinstr("`=`var'[1]'", "with", "w",.) in 1
replace `var' = subinstr("`=`var'[1]'", "judicial", "jud",.) in 1
replace `var' = subinstr("`=`var'[1]'", "authorization", "authztn",.) in 1
replace `var' = subinstr("`=`var'[1]'", "__", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "__", "_",.) in 1
replace `var' = abbrev(`var', 31) in 1
replace `var' = subinstr("`=`var'[1]'", "~", "",.) in 1
}
}

renvars, map (word(@[1],1))

//The following section keeps only the variables that MCC uses to construct the Gender in
the Economy Indicator.
keep Economy /*
Accessing Institutions
*/    umar_woman_obtain_a_national__ mar_woman_obtain_a_national_Ic /*
*/    umar_woman_travel_outside_theo   mar_woman_travel_outside_the_u   /*
*/    umar_woman_travel_outside_hero   mar_woman_travel_outside_her_m    /*
*/    umar_woman_get_a_job_in_the_se   mar_woman_get_a_job_in_the_sa_     /*
*/    umar_woman_sign_a_contract_inh   mar_woman_sign_a_contract_in_e       /*
*/    umar_woman_register_a_businesi   mar_woman_register_a_businessn     /*
*/    umar_woman_open_a_bank_accouni   mar_woman_open_a_bank_accountn      /*
*/    umar_woman_choose_where_to_li_   mar_woman_choose_where_to_livi         /*
*/    umar_woman_cfr_ctzsh_to_her_ks   mar_woman_cfr_ctzsh_to_her_ki_   /*
*/    umar_woman_be_HoH_like_an_umam   mar_woman_be_HoH_like_a_mar_man      /*
*/    If_vld_cust_law_does_const_ind   Do_mar_couples_jointly_share_g     /*
Using Property
*/    law_provide_the_valuation_of_n   Who_legly_admns_marital_prop     /*
*/    Do_umar_men_and_umar_women_ha_   Do_mar_men_and_mar_women_haveq      /*
*/    Do_sons_and_daughters_have_eql   Do_female_and_male_surviving_o   /*
Going to Court
*/    Does_a_womans_testimony_carryh      /*
Getting a Job
*/    nprg_nse_w_work_the_same_nighh   nprg_nse_w_do_the_same_jobs_am     /*
Protecting Women From Violence
*/    domestic_violence_leg     clear_criminal_penalties_domei   /*
*/    a_specialized_crt_or_procedurc   leg_that_specifically_address_          /*
*/    criminal_penalties_sx_hrsmt_ie   leg_age_of_marr_girls      /*
*/    any_exceptions_to_the_leg_agef   law_prbt_or_invldate_child_ora   /*
*/    penalties_authrzngknwinly_mari
rename Economy Country

drop in 1
destring, replace
//Some indicators count for half a point, others count for a full point.  The following
global macros split the variables above into these two categories
global halfpoint_indicators "umar_woman_obtain_a_national__
mar_woman_obtain_a_national_Ic   umar_woman_travel_outside_theo
mar_woman_travel_outside_the_u   umar_woman_travel_outside_hero
mar_woman_travel_outside_her_m    umar_woman_get_a_job_in_the_se
mar_woman_get_a_job_in_the_sa_     umar_woman_sign_a_contract_inh
mar_woman_sign_a_contract_in_e       umar_woman_register_a_businesi
mar_woman_register_a_businessn     umar_woman_open_a_bank_accouni
mar_woman_open_a_bank_accountn      umar_woman_choose_where_to_li_
mar_woman_choose_where_to_livi         umar_woman_cfr_ctzsh_to_her_ks
mar_woman_cfr_ctzsh_to_her_ki_   umar_woman_be_HoH_like_an_umam
mar_woman_be_HoH_like_a_mar_man   Do_umar_men_and_umar_women_ha_
Do_mar_men_and_mar_women_haveq  "
global fullpoint_indicators "If_vld_cust_law_does_const_ind
Do_mar_couples_jointly_share_g  law_provide_the_valuation_of_n
Who_legly_admns_marital_prop Do_sons_and_daughters_have_eql
Do_female_and_male_surviving_o Does_a_womans_testimony_carryh
nprg_nse_w_work_the_same_nighh   nprg_nse_w_do_the_same_jobs_am domestic_violence_leg
clear_criminal_penalties_domei a_specialized_crt_or_procedurc
leg_that_specifically_address_ criminal_penalties_sx_hrsmt_ie
law_prbt_or_invldate_child_ora penalties_authrzngknwinly_mari"

```

```
//The following assigns 0.5 points to any halfpoint indicator with a "No" or ".." value
foreach halfpoint of global halfpoint_indicators {
replace `halfpoint' = "0.5" if `halfpoint' == "No"
replace `halfpoint' = "0" if `halfpoint' == "Yes"
replace `halfpoint' = "0" if `halfpoint' == "yes"
replace `halfpoint' = "." if `halfpoint' == "N/A"
replace `halfpoint' = "0.5" if `halfpoint' == ".."
}
destring, replace

//The following assigns 1 point to answers of "No", "Husband", "Wife", or ".."
foreach fullpoint of global fullpoint_indicators {
replace `fullpoint' = "1" if `fullpoint' == "No"
replace `fullpoint' = "0" if `fullpoint' == "Yes"
replace `fullpoint' = "0" if `fullpoint' == "yes"
replace `fullpoint' = "1" if `fullpoint' == "Husband"
replace `fullpoint' = "1" if `fullpoint' == "husband"
replace `fullpoint' = "1" if `fullpoint' == "Wife"
replace `fullpoint' = "1" if `fullpoint' == "wife"
replace `fullpoint' = "0" if `fullpoint' == "Other"
replace `fullpoint' = "0" if `fullpoint' == "other"
replace `fullpoint' = "0" if `fullpoint' == "Both must agree"
replace `fullpoint' = "0" if `fullpoint' == "both must agree"
replace `fullpoint' = "0" if `fullpoint' == "Original owner"
replace `fullpoint' = "0" if `fullpoint' == "original owner"
replace `fullpoint' = "0" if `fullpoint' == "Original Owner"
replace `fullpoint' = "0" if `fullpoint' == "Separate with spousal consent"
replace `fullpoint' = "0" if `fullpoint' == "separate with spousal consent"
replace `fullpoint' = "." if `fullpoint' == "N/A"
replace `fullpoint' = "1" if `fullpoint' == ".."
}

//The following assigns 1 point to the indicator for answers of "..", "Yes", or values under 18  Note that these indicators are reversed from other indicators in the loops above
replace leg_age_of_marr_girls = "1" if leg_age_of_marr_girls == ".."
replace any_exceptions_to_the_leg_agef = "1" if any_exceptions_to_the_leg_agef == "Yes"
replace any_exceptions_to_the_leg_agef = "1" if any_exceptions_to_the_leg_agef == "yes"
replace any_exceptions_to_the_leg_agef = "0" if any_exceptions_to_the_leg_agef == "No"
replace any_exceptions_to_the_leg_agef = "0" if any_exceptions_to_the_leg_agef == "no"
replace any_exceptions_to_the_leg_agef = "1" if any_exceptions_to_the_leg_agef == ".."
destring, replace
replace leg_age_of_marr_girls = 1 if leg_age_of_marr_girls < 18
replace leg_age_of_marr_girls = 0 if leg_age_of_marr_girls >= 18

//The following creates a macro of all of the relevant varaibles, then sums those across.
unab Allvars : _all
unab exclude : Country
local Addlist : list Allvars - exclude
egen Gender_score = rowtotal(`Addlist')
label variable Gender_score "2019"
keep Country Gender_score
WBNaming Country
save "${created_data}\Gender FY20.dta", replace

**Merging
*********

use "${created_data}\Gender FY20.dta", clear
keep Country Gender_score
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Gender Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc Gender_score
replace percentile = 1-percentile //Note this is different from other indicators due to the inverse scale
save "${created_data}\Gender LISP.dta", replace
restore
```

```
keep if Income == "HISP"
PercentRankInc Gender_score
replace percentile = 1-percentile //Note this is different from other indicators due to the inverse scale
save "${created_data}\Gender HISP.dta", replace
append using "${created_data}\Gender LISP.dta"


keep Country Income percentile Score_In_FY20
gen Indicator = "Gender_In_The_Economy"
rename percentile Percentile
save "${created_data}\Gender In The Economy.dta", replace
export excel using "${final_data}\FY20_Gender_In_The_Economy.xlsx", firstrow(variables) replace

//////////////////////////////////
//  Natural Resource Protection //
//////////////////////////////////

//Download Instructions:
/*
Go to CIESIN's Natural Resource Management Index https://sedac.ciesin.columbia.edu/data/collection/nrmi/sets/browse
Click on Data Sets, and select the most recent release of the Natural resource protection and Child Health Indicators
Click on Data Download.  You may need to log into NASA's website to view the data.
Select the zip release
Save as the file referenced below in the raw data folder
*/


import excel "${raw_data}\NRPI.xlsx", sheet("NRPI_proximity") firstrow clear
replace NRPI_v2019_19 = 100 if CountryName == "Morocco" //Morocco's value was incorrectly entered as 99.9999999999999 but should be 100.
WBNaming CountryName
label var CountryName "Country"
keep CountryName NRPI*
save "${created_data}\NRP FY20.dta", replace

**Merging
*********

use "${created_data}\NRP FY20.dta", clear
rename CountryName Country
keep Country NRPI_v2019_19
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)
save "${created_data}\NRP Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc NRPI_v2019_19
save "${created_data}\NRP LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc NRPI_v2019_19
save "${created_data}\NRP HISP.dta", replace
append using "${created_data}\NRP LISP.dta"


keep Country Income percentile Score_In_FY20
gen Indicator = "Natural_Resource_Protection"
rename percentile Percentile
save "${created_data}\Natural Resource Protection.dta", replace
export excel using "${final_data}\FY20_Natural_Resource_Protection.xlsx", firstrow(variables) replace

//////////////////////////
//      Child Health    //
//////////////////////////
//Download Instructions:
```

```
//This should be contained in the same download as the NRP data above

import excel "${raw_data}\CHI_2019.xlsx", sheet("CHI_2019_v3") firstrow clear
keep Country CHI_v2019_18 CHI_v2019_17 CHI_v2019_16 CHI_v2019_15  CHI_v2019_14 CHI_v2019_13 CHI_v2019_12 CHI_v2019_11 CHI_v2019_10
WBNaming Country

save "${created_data}\Child FY20.dta", replace

**Merging
*********
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Child Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc CHI_v2019_18
save "${created_data}\Child LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc CHI_v2019_18
save "${created_data}\Child HISP.dta", replace
append using "${created_data}\Child LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Child_Health"
rename percentile Percentile
save "${created_data}\Child Health.dta", replace
export excel using "${final_data}\FY20_Child_Health.xlsx", firstrow(variables) replace


////////////////////////////////////
// Land Rights and Access (IFAD) //
////////////////////////////////////

//Download Instructions
/*
Go to https://webapps.ifad.org/members/gc
click on "All Sessions" on the Right to go here https://webapps.ifad.org/members/gc/sessions
Find the most recent Session, and look for "Programme of work and budget of IFAD and its Office of Evaluation for YEAR"
If this is not present, check a previous Session
Scroll to the bottom of this document to the section titled "YEAR Rural Sector Performance Assessment (RSPA) scores"
Find the indicator for Access to Land (3.3 in 2018)
Copy all the numbers into a txt file, replace all spaces with commas.  Save as a .csv file
Open CSV in excel, the numbers should be in different columns.
Copy the data, use special paste to reformat it so that it is in a single column.
Copy the country labels in one group at a time.  Make sure you use the same order as the original data.
(You may need to open the file in Adobe Acrobat instead of a web browser to do this).
Delete any cells where the country title is on more than one line. Shift Cells up.
Check to make sure that all of the country data is lined up correctly.
Label the columns "Country" and "Data", Label the sheet IFAD Data

Save as a Excel file (not a CSV) named IFAD Data.xlsx


If data is not available in the current year for a country, MCC pulls forward data from the previous report (back to 2014 for FY20).
This report can be found by going to https://webapps.ifad.org/members/gc/sessions, selecting the next previous session (check for the 2015 RSPA at the bottom of the document
then selecting the agenda item with the "IFAD's 2017 results-based programme of work and regular and capital budgets, the IOE results-based work programme and budget for 2017 and indicative plan for 2018-2019, and the HIPC and PBAS progress reports"
Once again scroll to the bottom of the document and repeat the steps above for the indicator Access to Land (B.i in this document).  This should be the 2015 Rural Sector Performance Assessment
Label the columns "Country" and "Data", label the sheet IFAD Data
Save as IFAD_2015.xlsx
```

```

Repeat this process, for the 2015 session (38th) of the governing council.  These numbers should be from the 2014 RSPA.
Save as IFAD_2014.xlsx

*/

set more off
import excel "${raw_data}\IFAD Data.xlsx", clear firstrow sheet("IFAD Data")
WBNaming Country
rename Data Y2020
save "${created_data}\IFAD_Data.dta", replace

import excel "${raw_data}\IFAD_2015.xlsx", clear firstrow sheet("IFAD Data")
WBNaming Country
rename Data Y2015
merge 1:1 Country using "${created_data}\IFAD_Data.dta", nogen
replace Y2020 = Y2015 if missing(Y2020)
save "${created_data}\IFAD_Data.dta", replace

import excel "${raw_data}\IFAD_2014.xlsx", clear firstrow sheet("IFAD Data")
WBNaming Country
rename Data Y2014
merge 1:1 Country using "${created_data}\IFAD_Data.dta", nogen
replace Y2020 = Y2014 if missing(Y2020)


egen Max_Score2020 = max(Y2020)
egen Min_Score2020 = min(Y2020)
gen IFAD_Score2020 = 1-((Max_Score2020-Y2020)/(Max_Score2020-Min_Score2020))

//keep Country IFAD_Score*
save "${created_data}\IFAD_Score_Normalized.dta", replace


//This indicator is only half of the Land Rights and Access Indicator
//We need to add in the Doing Business Registering Property data in order to create the full indicator.



///////////////////////
//   Doing Business  //
///////////////////////

//Download Instructions:
/*
Go to https://www.doingbusiness.org/en/data and select historical data create a custom dataset
This should take you to: https://www.doingbusiness.org/en/custom-query
Choose all Economies
Choose Topics "Starting a Business" "Getting Credit" and "Registering Property"
Select years back to 2015
Click Create report
Click the excel button to download a report
Save as the file below
*/


import excel "${raw_data}\Doing Business 2020.xlsx", clear
//The following changes variable names that are too long for Stata
foreach var of varlist _all {
label var `var' "`=`var'[1]'"
replace `var' = subinstr("`=`var'[1]'", " ", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'", ")", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "(", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "-", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "%", "pc",.) in 1
replace `var' = subinstr("`=`var'[1]'", "?", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "__", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "__", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "doing_business", "do_biz",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Registering_property_", "Rg_prp_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Registering_Property_", "Rg_Prp_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Quality_of_land_administration", "Ql_ld_adm",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Starting_a_Business", "Start_Biz",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Getting_Credit", "Get_Cr",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Strength_of", "St",.) in 1
replace `var' = subinstr("`=`var'[1]'", "Depth_of", "Dp",.) in 1
replace `var' = subinstr("`=`var'[1]'", "index", "",.) in 1
replace `var' = subinstr("`=`var'[1]'", "__", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'", "__", "_",.) in 1
replace `var' = abbrev(`var', 27) in 1
replace `var' = subinstr("`=`var'[1]'", "~", "",.) in 1
//Coding "No Practice" as "-999999999", this will be changed back to a real value below
replace `var' = "-999999999" if `var' == "No Practice"
}
renvars, map (word(@[1],1))
drop in 1

//Keeping only the necessary variables and observations
replace Year = subinstr(Year,"DB","",.)
destring, replace
drop if Year < 2015
drop if missing(Year) & Economy == ""

//Setting variable names up so that they can be reshaped, and reshaping the data to a wide format
unab Allvars : _all
unab exclude : Economy Year
local Addlist1 : list Allvars - exclude

foreach var of varlist `Addlist1' {
rename `var' `var'_
local `var'lb : variable label `var'
}

unab Allvars : _all
unab exclude : Economy Year
local Addlist2 : list Allvars - exclude
reshape wide `Addlist2' , i(Economy) j(Year)

//Labeling Variables
foreach var of local Addlist1 {
forvalues k = 2015(1)2020 {
label var `var'_`k' "``var'lb' `k'"
}
}

//Generating a maximum for each variable, and putting it in place of the "No Practice" values
unab Allvars : _all
unab exclude : Economy
local Addlist3 : list Allvars - exclude

foreach var of varlist `Addlist3' {
egen x`var' = max(`var')
replace `var' = x`var' if `var' == -999999999
drop x`var'
}

//Merging
WBNaming Economy
rename Economy Country
label var Country "Country"
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)
drop GNI
save "${created_data}\All_Doing_Business_Formatted.dta" , replace

//This file will be used in what follows to create three indicators: Business Start Up, Access to Credit, and (with IFAD) Land Rights and Access.
```

```


////////////////////////
//  Access to Credit  //
////////////////////////
use "${created_data}\All_Doing_Business_Formatted.dta", clear
keep Country Get_Cr_St_legal_rights_152_20* Get_Cr_Dp_credit_inform08__20*
//The following computes access to credit scores for a given country by normalizing both components and then taking an average.
forvalues k=15(1)20 {
gen Norm_Depth_`k' = Get_Cr_Dp_credit_inform08__20`k' * 12
gen Norm_Legal_`k' = Get_Cr_St_legal_rights_152_20`k' * 8
gen Access_score_`k' = (Norm_Depth_`k' + Norm_Legal_`k')/2
}
save "${created_data}\Access Calculated.dta" , replace


**Merging
*********

use "${created_data}\Access Calculated.dta", clear
keep Country Access_score_20
rename Access_score_20 Access_score
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Access Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc Access_score
save "${created_data}\Access LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc Access_score
save "${created_data}\Access HISP.dta", replace
append using "${created_data}\Access LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Access_To_Credit"
rename percentile Percentile
save "${created_data}\Access To Credit.dta", replace
export excel using "${final_data}\FY20_Access_To_Credit.xlsx", firstrow(variables) replace


/////////////////////////
//  Business Start-Up  //
/////////////////////////


use "${created_data}\All_Doing_Business_Formatted.dta", clear
keep Country Start_Biz_Cost_Men_pc_of___20*  Start_Biz_Cost_Women_pc_om_20* Start_Biz_Time_Men_days_20*  Start_Biz_Time_Women_days_20*
forvalues k= 15(1)20 {
egen Time_Average`k' = rowmean(Start_Biz_Time_Men_days_20`k'  Start_Biz_Time_Women_days_20`k') //Taking an average of time for men and women
egen Cost_Average`k' = rowmean(Start_Biz_Cost_Men_pc_of___20`k' Start_Biz_Cost_Women_pc_om_20`k')  //Taking an average of cost for men and women
//The following finds the min and max of these variables
egen Max_Time`k' = max(Time_Average`k')
egen Max_Cost`k' = max(Cost_Average`k')
egen Min_Time`k' = min(Time_Average`k')
egen Min_Cost`k' = min(Cost_Average`k')
//The following normalizes these variables to a common scale using the min and max
gen Time_Score`k' = (Max_Time`k'-Time_Average`k')/(Max_Time`k'-Min_Time`k')
gen Cost_Score`k' = (Max_Cost`k'-Cost_Average`k')/(Max_Cost`k'-Min_Cost`k')
gen Business_score_`k' = (Time_Score`k' + Cost_Score`k')/2
}
keep Country Time_Score* Cost_Score* Business_score_*
save "${created_data}\Business Calculated.dta" , replace

```

```
**Merging
**********
use "${created_data}\Business Calculated.dta", clear
keep Country Business_score_20
rename Business_score_20 Business_score
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Business Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc Business_score
save "${created_data}\Business LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc Business_score
save "${created_data}\Business HISP.dta", replace
append using "${created_data}\Business LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Business_Start_Up"
rename percentile Percentile
save "${created_data}\Business Start Up.dta", replace
export excel using "${final_data}\FY20_Business_Start_Up.xlsx", firstrow(variables) replace


///////////////////////////////////
//  Land Rights and Access (WB)  //
///////////////////////////////////

set more off
use "${created_data}\All_Doing_Business_Formatted.dta", clear
keep Country Rg_Prp_Time_days_20* Rg_Prp_Cost_pc_of_properte_20*

forvalues k= 15(1)20 {
egen Max_Time`k' = max(Rg_Prp_Time_days_20`k')
egen Max_Cost`k' = max(Rg_Prp_Cost_pc_of_properte_20`k')
egen Min_Time`k' = min(Rg_Prp_Time_days_20`k')
egen Min_Cost`k' = min(Rg_Prp_Cost_pc_of_properte_20`k')
gen Time_Score`k'  = (Max_Time`k'-Rg_Prp_Time_days_20`k')/(Max_Time`k'-Min_Time`k')
gen Cost_Score`k'  = (Max_Cost`k'-Rg_Prp_Cost_pc_of_properte_20`k')/(Max_Cost`k'-Min_Cost`k')
gen Rg_Prp_score_`k'  = (Time_Score`k' + Cost_Score`k')/2
}

**Merging
*********
merge 1:1 Country using  "${created_data}\IFAD_Score_Normalized.dta"
gen LRA_score = (Rg_Prp_score_20 + IFAD_Score2020)/2

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\LRA Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc LRA_score
save "${created_data}\LRA LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc LRA_score
save "${created_data}\LRA HISP.dta", replace
append using "${created_data}\LRA LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Land_Rights_And_Access"
rename percentile Percentile
save "${created_data}\Land Rights And Access.dta", replace
export excel using "${final_data}\FY20_Land_Rights_And_Access.xlsx", firstrow(variables) replace

```

```

////////////////////////////////////
//   WGI Control of Corruption   //
////////////////////////////////////
//The following four indicators are drawn from the World Governance Indicators.
//The code for each follows the same procedures
/*
//Download instructions:
Go to https://info.worldbank.org/governance/wgi/ and download the excel file
Name the file wgidataset.xlsx and save in the raw data folder
*/

import excel "${raw_data}\wgidataset.xlsx", sheet("ControlofCorruption") clear
forvalues k = 1(1)13 {  //Dropping extraneous header information
drop in 1
}
foreach var of varlist _all {
replace `var' = `var'[2] +`var'[1] in 1 //Combining the year and the type (Estimate, Standard Deviation, etc.) into a single variable
replace `var' = "." if `var' == "#N/A"  //Setting #N/A as missing
replace `var' = "Country" if `var' == "Country/Territory"
}
drop in 2  //Dropping type now that it is incorporated into the first row
renvars, map (word(@[1],1))
drop in 1
destring, replace
save "${created_data}\Control of Corruption Full.dta" , replace

**Merging
*********

keep Country Estimate2018
rename Estimate2018 Corruption_Score

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)
save "${created_data}\Corruption Full.dta", replace

//Calculating Percentile Ranks
preserve
keep if Income == "LISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(Corruption_Score)
replace Corruption_Score = Corruption_Score - median
drop median
PercentRankInc Corruption_Score
save "${created_data}\Corruption LISP.dta", replace
restore
keep if Income == "HISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(Corruption_Score)
replace Corruption_Score = Corruption_Score - median
drop median
PercentRankInc Corruption_Score
save "${created_data}\Corruption HISP.dta", replace
append using "${created_data}\Corruption LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Control_Of_Corruption"
rename percentile Percentile
save "${created_data}\Control Of Corruption.dta", replace
export excel using "${final_data}\FY20_Control_Of_Corruption.xlsx", firstrow(variables) replace


///////////////////////////////////
//  WGI Government Effectiveness //
///////////////////////////////////

import excel "${raw_data}\wgidataset.xlsx", sheet("GovernmentEffectiveness") clear
```

```
forvalues k = 1(1)13 {  //Dropping extraneous header information
drop in 1
}
foreach var of varlist _all {
replace `var' = `var'[2] +`var'[1] in 1 //Combining the year and the type (Estimate, Standard Deviation, etc.) into a single variable
replace `var' = "." if `var' == "#N/A"  //Setting #N/A as missing
replace `var' = "Country" if `var' == "Country/Territory"
}
drop in 2  //Dropping type now that it is incorporated into the first row
renvars, map (word(@[1],1))
drop in 1
destring, replace
save "${created_data}\Government Effectiveness Full.dta" , replace

**Merging
***********

keep Country Estimate2018
rename Estimate2018 Gov_Eff_Score

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

//Calculating Percentiles
preserve
keep if Income == "LISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(Gov_Eff_Score)
replace Gov_Eff_Score = Gov_Eff_Score - median
drop median
PercentRankInc Gov_Eff_Score
save "${created_data}\Gov_Eff LISP.dta", replace
restore
keep if Income == "HISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(Gov_Eff_Score)
replace Gov_Eff_Score = Gov_Eff_Score - median
drop median
PercentRankInc Gov_Eff_Score
save "${created_data}\Gov_Eff HISP.dta", replace
append using "${created_data}\Gov_Eff LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Government_Effectiveness"
rename percentile Percentile
save "${created_data}\Government Effectiveness.dta", replace
export excel using "${final_data}\FY20_Government_Effectiveness.xlsx", firstrow(variables) replace


///////////////////////////////////
//       WGI Rule of Law         //
///////////////////////////////////

import excel "${raw_data}\wgidataset.xlsx", sheet("RuleofLaw") clear
forvalues k = 1(1)13 {  //Dropping extraneous header information
drop in 1
}
foreach var of varlist _all {
replace `var' = `var'[2] +`var'[1] in 1 //Combining the year and the type (Estimate, Standard Deviation, etc.) into a single variable
replace `var' = "." if `var' == "#N/A"  //Setting #N/A as missing
replace `var' = "Country" if `var' == "Country/Territory"
}
drop in 2  //Dropping type now that it is incorporated into the first row
renvars, map (word(@[1],1))
drop in 1
destring, replace
save "${created_data}\Rule of Law Full.dta" , replace
```

```

keep Country Estimate2018
rename Estimate2018 RuleofLaw_Score

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

//Calculate percentiles
preserve
keep if Income == "LISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(RuleofLaw_Score)
replace RuleofLaw_Score = RuleofLaw_Score - median
drop median
PercentRankInc RuleofLaw_Score
save "${created_data}\RuleofLaw LISP.dta", replace
restore
keep if Income == "HISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(RuleofLaw_Score)
replace RuleofLaw_Score = RuleofLaw_Score - median
drop median
PercentRankInc RuleofLaw_Score
save "${created_data}\RuleofLaw HISP.dta", replace
append using "${created_data}\RuleofLaw LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Rule_Of_Law"
rename percentile Percentile
save "${created_data}\Rule Of Law.dta", replace
export excel using "${final_data}\FY20_Rule_Of_Law.xlsx", firstrow(variables) replace

/////////////////////////////////////
//    WGI Regulatory Quality       //
/////////////////////////////////////

import excel "${raw_data}\wgidataset.xlsx", sheet("RegulatoryQuality") clear
forvalues k = 1(1)13 {  //Dropping extraneous header information
drop in 1
}
foreach var of varlist _all {
replace `var' = `var'[2] +`var'[1] in 1 //Combining the year and the type (Estimate, Standard Deviation, etc.) into a single variable
replace `var' = "." if `var' == "#N/A"  //Setting #N/A as missing
replace `var' = "Country" if `var' == "Country/Territory"
}
drop in 2  //Dropping type now that it is incorporated into the first row
renvars, map (word(@[1],1))
drop in 1
destring, replace
save "${created_data}\Regulatory Quality Full.dta" , replace

keep Country Estimate2018
rename Estimate2018 RegQuality_Score

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

//Calculating Percentiles
save "${created_data}\RegQuality Full.dta", replace
preserve
keep if Income == "LISP"
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(RegQuality_Score)
replace RegQuality_Score = RegQuality_Score - median
drop median
PercentRankInc RegQuality_Score
save "${created_data}\RegQuality LISP.dta", replace
restore
keep if Income == "HISP"
```

```
//For all the WGI indicators, we shift all values so that 0 is the median
egen median = median(RegQuality_Score)
replace RegQuality_Score = RegQuality_Score - median
drop median
PercentRankInc RegQuality_Score
drop median
save "${created_data}\RegQuality HISP.dta", replace
append using "${created_data}\RegQuality LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Regulatory_Quality"
rename percentile Percentile
save "${created_data}\Regulatory Quality.dta", replace
export excel using "${final_data}\FY20_Regulatory_Quality.xlsx", firstrow(variables) replace




///////////////////////////////////
//    Freedom of Information     //
///////////////////////////////////
// The Freedom of the Press indicator is a composite of three sub indicators
// The main sub indicator is Freedom of the Press, with two minor sub indicators focused on Right to Information, and Key Internet controls
//The following code imports, cleans and combines these files to create scores for each country.

////////////////////////////
//  Freedom of the Press  //
////////////////////////////
// In previous years, the Freedom of the Press Indicator was drawn from Freedom House's Freedom of the Press Report
// However, Freedom House stopped producing this indicator in 2017.  This required MCC to find another source for these data
// This year, MCC began using the Reporters Without Borders (RSF) Press Freedom Index in place of Freedom House.
// However, RSF was missing data for several small island nations, therefore MCC uses percentile matching as shown below
// to impute RSF scores for these countries based on data from previous years of Freedom House.
//Download Instructions:
/*
Go to the RSP website.  Go to the Index details (https://rsf.org/en/ranking_table), and select download the dataset
Save as "RSF dataset 2019.csv"
For the historical files: the "index_format_upload 2017.csv" can be found at the bottom of the page here "https://rsf.org/en/2017-press-freedom-index-ever-darker-world-map"
And the classment2016 file can be found at the bottom of this page: "https://rsf.org/en/2016-world-press-freedom-index-leaders-paranoid-about-journalists"
Go to the Freedom House website.  Go to the Freedom of the Press Report (https://freedomhouse.org/report-types/freedom-press), and select "Freedom of the Press Data"
Save as FOTP1980-FOTP2017_Public-Data.xlsx
*/

// The Following Sections format the RSF files for the Press Freedom Index
//RSF Data Formatting
set more off
//RSF 2019
import delimited "${raw_data}\RSF Dataset 2019.csv", clear
keep en_country score2019
rename en_country Country
rename score2019 RSFscore

replace RSFscore = subinstr(RSFscore,",",".",.)
destring, replace
WBNaming Country
save "${created_data}\RSF_Scores_2019.dta", replace

//RSF 2018
import delimited "${raw_data}\RSF Dataset 2019.csv", clear
```

```
keep en_country score2018
rename en_country Country
rename score2018 RSFscore

replace RSFscore = subinstr(RSFscore,",",".",.)
destring, replace
WBNaming Country
save "${created_data}\RSF_Scores_2018.dta", replace

//RSF 2017
import delimited "${raw_data}\index_format_upload 2017.csv", delimiter(";") clear
//This file is mislabeled.  The data that is labeled as "overallscore2016" is actually 2017 data (according to the main map).  The score2015 data is actually 2016 data (as it matches data from the 2016 file).  This has been confirmed with RSF.
keep en_country overallscore2016  //This is actually 2017 data
rename en_country Country
rename overallscore2016 RSFscore

replace RSFscore = subinstr(RSFscore,",",".",.)
destring, replace
WBNaming Country
save "${created_data}\RSF_Scores_2017.dta", replace

//RSF 2016
import delimited "${raw_data}\index_format_upload 2017.csv", delimiter(";") clear
//This file is mislabeled.  The data that is labeled as "overallscore2016" is actually 2017 data (according to the main map).  The score2015 data is actually 2016 data (as it matches data from the 2016 file).
keep en_country score2015  //This is actually 2016 data
rename en_country Country
rename score2015 RSFscore

replace RSFscore = subinstr(RSFscore,",",".",.)
destring, replace
WBNaming Country
save "${created_data}\RSF_Scores_2016.dta", replace

//RSF 2015
import delimited "${raw_data}\classement2016.csv", delimiter(";") clear
keep en_country score2015
rename en_country Country
rename score2015 RSFscore

replace RSFscore = subinstr(RSFscore,",",".",.)
destring, replace
WBNaming Country
save "${created_data}\RSF_Scores_2015.dta", replace


//The following section formats the Freedom House data
//FH Data Formatting
import excel "${raw_data}\FOTP1980-FOTP2017_Public-Data.xlsx", clear sheet(Data)
replace A = "Country" in 5
//The following section fills in missing year values for observations which were merged in the original document
qui ds  //List all variables
local variable_tally : word count `r(varlist)'  //Count the number of variables
local tally_minus = `variable_tally' - 1 //Create a macro which is one less than the full variable count
forvalues k = 1(1)`tally_minus' {
local left : word `k' of `r(varlist)'  //Finding the "left" value for the year observation
local right : word `=`k'+1' of `r(varlist)' //Finding the "right" value for the year observation
replace `right' = `left' if `right'[3] == "" in 3  //Filling forward the year values when the values are missing
}
//Dropping Statuses and Sub-Scores
foreach var of varlist _all {
if !(`var'[5] == "Print" | `var'[5] == "Broadcast" | `var'[5] == "Status" | `var'[5] == "Total Score" | `var'[5] == "Country")  {
 drop `var'
}
}

//Dropping extraneous headings
drop in 4
drop in 2
drop in 1
replace A = "Country" in 1
//The following section of code combines the year headings with the data type headings
foreach var of varlist _all {
label var `var' "`=`var'[1]'"
replace `var' = "Y`=`var'[1]'_`=`var'[2]'" in 1
replace `var' = subinstr("`=`var'[1]'","-","_",.) in 1
}
replace A = "Country" in 1

renvars, map (word(@[1],1))
drop in 1
drop in 1

destring, replace
WBNaming Country
save "${created_data}\Freedom_House_Press_Formatted.dta", replace


**Imputation
************
//Now that we have formatted the two files, we need to merge them and
//use the freedom house data to impute the new scores using percentile
//matching

set more off
//This loop takes each RSF year and uses the freedom house data to impute the correct scores.
forvalues RSF_year = 2015(1)2019 {

use "${created_data}\Freedom_House_Press_Formatted.dta", clear

merge 1:1 Country using "${created_data}\RSF_Scores_`RSF_year'.dta", nogen keep(1 3)
//Since the most recent Freedom House data is from 2017, when the RSF year is 2017, 2018, or 2019,
// we use that Freedom House data to impute the scores.  However in earlier years,
// we use matching year data:
if "`RSF_year'" == "2019" | "`RSF_year'" == "2018" | "`RSF_year'" == "2017" {
rename Y2017_Total FHScore

}
else {
rename Y`RSF_year'_Total FHScore
}

replace FHScore = "." if FHScore == "-"
keep Country RSFscore FHScore
destring, replace
//In this line we are making the decision to only consider countries that are in the GNI dataset as described above.
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)
//In order to calculate the percentile matching scores, we need to calculate the number
//of Freedom House Scores in the full dataset.
egen n = count(FHScore)
//We also need to calculate a ranking (1+ the number of scores that are lower)
egen i = rank(FHScore), track
//The Percentile rank in the full Freedom House dataset is then calculated as the rank minus 1, divided by the count minus 1
gen pcrank_FH = (i - 1) / (n - 1)
drop i n
egen RSF_count = count(RSFscore)
//Now, we use the following calculations to determine where the rank of the freedom house percentile
//in the RSF dataset, and split it into its whole number and decimal part
```

```
gen FH_Rank_in_RSF = (pcrank_FH*(RSF_count-1))+1
gen FH_floor = floor(FH_Rank_in_RSF)
gen FH_decimal = FH_Rank_in_RSF - FH_floor

sort RSFscore
local RSF_cnt = `=RSF_count[1]'
//This code creates a set of global macros corresponding to the ranking of the RSF scores
qui forvalues k=1(1)`RSF_cnt' {
local k_plus = `k' + 1
global RSF_function_`k' = `=RSFscore[`k']'
noisily di "${RSF_function_`k'}  `k'"
}
gen function_x = .
gen function_x_plus = .
local count = _N
//The following code finds the RSF scores which rank on either side of
//The Freedom House ranking, and then adding the product of the decimal
//part of the Freedom House ranking and the difference of the two RSF scores
//on either side.
qui forvalues k = 1(1)`count' {
replace function_x = ${RSF_function_`=FH_floor[`k']'} in `k'
local floor_plus = `=FH_floor[`k']' + 1
local count_minus = `count' -1
if "`=FH_floor[`k']'" == "`RSF_cnt'" {
noi di "`k'"
local floor_plus = `=FH_floor[`k']'
noisily di "${RSF_function_`floor_plus'}"
}
replace function_x_plus = ${RSF_function_`floor_plus'} in `k'
}
gen New_RSF_Score = function_x + (FH_decimal*(function_x_plus-function_x))
//This replaces the missing RSF scores with the newly imputed scores based on the FH data
replace RSFscore = New_RSF_Score if missing(RSFscore)
drop New_RSF_Score function_x_plus function_x FH_decimal FH_floor FH_Rank_in_RSF RSF_count pcrank_FH GNI FHScore
rename RSFscore Y`RSF_year'_Total
label var Y`RSF_year'_Total "`RSF_year'"
gen Y`RSF_year'_Status = "NF"
label var Y`RSF_year'_Status "`RSF_year'"
WBNaming Country
save "${created_data}\Freedom_of_The_Press_Imputed_`RSF_year'.dta", replace
}
//Merging each of the year files together
use "${created_data}\Freedom_of_The_Press_Imputed_2015.dta", clear
forvalues RSF_year = 2016(1)2019 {
merge 1:1 Country using "${created_data}\Freedom_of_The_Press_Imputed_`RSF_year'.dta", nogen
}
save "${created_data}\FOP FY20.dta", replace

//Before we can create the final file, we need to process the Key Internet Controls and the Right to Information files.

///////////////////////////
// Key Internet Controls //
///////////////////////////
//Now that the main indicator has been processed, we must process the other two minor indicators
//Countries gain 1 point (higher score indicates worse performance) for each key internet control
//on Freedom House's Key Internet controls metric up to 9.
/*
Download Instructions:
Go to the report here https://www.freedomonthenet.org/sites/default/files/2019-11/11042019_Report_FH_FOTN_2019_final_Public_Download.pdf
On Page 28, there is a table with a list of countries and the number of key internet controls that they employed over the last year
Copy the first two columns of this table into an excel file, label one column Country and the other KIC
Label the sheet KIC table.  Save the document as Key Internet Controls
```

```

*/
//
import excel "${raw_data}\Key Internet Controls.xlsx", sheet("KIC table") firstrow clear
WBNaming Country
label var Country "Country"
//Not all countries have KIC scores
drop if Country == "" & KIC == .
save "${created_data}\Key_Internet_Controls.dta", replace

//Before we can create the indicator, we need to process the Right to Information file.

////////////////////////////
//  Right to Information  //
////////////////////////////
//Download Instructions: Go to the Country data page: https://www.rti-rating.org/country-data/
// Click on "Excel" download.  Save as RTI Ratings in the raw data folder
import excel "${raw_data}\RTI Ratings.xlsx", sheet("Sheet1") firstrow clear
keep Country Total
WBNaming Country
save "${created_data}\Right_To_Information.dta", replace


**Freedom of Information Merging
**************************************
//Merging files together:
use "${created_data}\Right_To_Information.dta", clear
merge 1:1 Country using "${created_data}\Key_Internet_Controls.dta", nogen
merge 1:1 Country using "${created_data}\Freedom_of_The_Press_Imputed_2019.dta", nogen
keep Country Total KIC Y2019_Total
gen RTI_score = 4 if !missing(Total)
replace KIC = 9 if !missing(KIC) & KIC>9
recode  RTI_score KIC (.=0)
gen FOI_score = (Y2019_Total - RTI_score) + KIC

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)
save "${created_data}\FOI Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc FOI_score
replace percentile = 1-percentile  //Note this is different for FOI due to the inverted scale
save "${created_data}\FOI LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc FOI_score
replace percentile = 1-percentile  //Note this is different for FOI due to the inverted scale
save "${created_data}\FOI HISP.dta", replace
append using "${created_data}\FOI LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Freedom_Of_Information"
rename percentile Percentile
save "${created_data}\Freedom Of Information.dta", replace
export excel using "${final_data}\FY20_Freedom_Of_Information.xlsx", firstrow(variables) replace


/////////////////////////
//    Trade Policy     //
/////////////////////////
/* Download Instructions:
As of the publication of this do file, Heritage's scores for trade freedom are not public for all MCC candidate countries
However, when they become public they will be available here https://www.heritage.org/index/download
Select "Download Raw Data" and save the file as index2020_data.xls in the raw data folder
*/

```

```
set more off
import excel "${raw_data}\index2020_data.xls", clear firstrow
keep CountryName TradeFreedom
replace TradeFreedom = "." if TradeFreedom == "N/A"
destring, replace
keep CountryName TradeFreedom
rename CountryName Country
rename TradeFreedom Trade_Policy_Score
WBNaming Country

**Merging
************
WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Trade Policy Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc Trade_Policy_Score
save "${created_data}\Trade Policy LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc Trade_Policy_Score
save "${created_data}\Trade Policy HISP.dta", replace
append using "${created_data}\Trade Policy LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Trade_Policy"
rename percentile Percentile
save "${created_data}\Trade Policy.dta", replace
export excel using "${final_data}\FY20_Trade_Policy.xlsx", firstrow(variables) replace


////////////////////////
//    WEO/IMF Data     //
////////////////////////
/*
Download instructions
Use the October 2019 release
Go to https://www.imf.org/external/pubs/ft/weo/2019/02/weodata/index.aspx
Select by countries
Select all countries
Select Inflation, average consumer prices (Percent change) and General government net lending/borrowing (Percent of GDP)
Select Start 1980, end 2024

*/
set more off
import excel "${raw_data}\WEO_Data.xlsx", clear
//Cleaning up variables for renvars
foreach var of varlist F-AX {
replace `var' = "Y`=`var'[1]'" in 1
}
foreach var of varlist _all {
replace `var' = subinstr("`=`var'[1]'"," ", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'","/", "_",.) in 1
replace `var' = subinstr("`=`var'[1]'","-", "_",.) in 1

replace `var' = "." if `var' == "n/a"
replace `var' = "." if `var' == "--"
}
renvars, map (word(@[1],1))
drop in 1
destring, replace
drop Y2019 Y2020 Y2021 Y2022 Y2023 Y2024 Estimates_Start_After
foreach var of varlist Y* {
lab var `var' `=subinstr("`var'", "Y","",.)'
}
save "${created_data}\WEO_IMF_Data_Full.dta" , replace

```

```

//////////////////////
//  Fiscal Policy   //
//////////////////////
//This pulls directly from an IMF indicator
set more off
use "${created_data}\WEO_IMF_Data_Full.dta", clear
keep if Subject_Descriptor == "General government net lending/borrowing"
keep Country Y*
WBNaming Country

**Merging
*********
//Fiscal Policy is a three year rolling average
egen Fiscal_Policy_Score = rowmean(Y2016 Y2017 Y2018)
keep Country Fiscal_Policy_Score

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Fiscal Policy Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc Fiscal_Policy_Score
save "${created_data}\Fiscal Policy LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc Fiscal_Policy_Score
save "${created_data}\Fiscal Policy HISP.dta", replace
append using "${created_data}\Fiscal Policy LISP.dta"

keep Country Income percentile Score_In_FY20
gen Indicator = "Fiscal_Policy"
rename percentile Percentile
save "${created_data}\Fiscal Policy.dta", replace
export excel using "${final_data}\FY20_Fiscal_Policy.xlsx", firstrow(variables) replace

//////////////////////
//    Inflation     //
//////////////////////
//This pulls directly from an IMF indicator
set more off
use "${created_data}\WEO_IMF_Data_Full.dta", clear
keep if Subject_Descriptor == "Inflation, average consumer prices"
keep Country Y*
WBNaming Country

**Merging
*********
rename Y2018 Inflation_Rate
keep Country Inflation_Rate

WBNaming Country
merge 1:1 Country using "${created_data}\GNI_FY20.dta", nogen keep(3)

save "${created_data}\Inflation Full.dta", replace
preserve
keep if Income == "LISP"
PercentRankInc Inflation_Rate
replace percentile = 1-percentile  //Note this is different for Inflation due to the inverted scale
save "${created_data}\Inflation LISP.dta", replace
restore
keep if Income == "HISP"
PercentRankInc Inflation_Rate
replace percentile = 1-percentile  //Note this is different for Inflation due to the inverted scale
save "${created_data}\Inflation HISP.dta", replace
append using "${created_data}\Inflation LISP.dta"

```

```
keep Country Income percentile Score_In_FY20
gen Indicator = "Inflation"
rename percentile Percentile
save "${created_data}\Inflation.dta", replace
export excel using "${final_data}\FY20_Inflation.xlsx", firstrow(variables) replace
```