**Guide to Stata Programs to Auto-populate Midline Table Shells**

The objective of this document is to explain the procedures and use of the Stata programs developed to auto-populate the phase two Zone of Influence (ZOI) Midline Indicator Assessment results table shells in Excel format. Currently, the midline indicator results table shells Excel file includes 105 tables. Because country-specific tables are customized from the midline table shells, the Stata programs and commands also need to be adjusted accordingly, which is also discussed in this document. To understand and use these Stata programs requires some knowledge of Stata programming.

The Stata programs use the “putexcel” command to export results to the table shells in the Excel file. The “putexcel” command is available in Stata version 13 or above, so Stata version 13 or above is required to run the Stata programs used for this task. These programs use two-user written Stata commands (lincomest and estpost), which may not come with the default Stata package. Install them before using these Stata programs.

To install estpost, run command: *ssc install estout, replace*

To install lincomest, run command: *ssc install lincomest, replace*

This guide is divided into three sections. Section 1 explains the creation of file directory structure for data, output, syntax files, and table shells in Excel format. Section 2 discusses the preparation of the Stata dataset used by the Stata programs. Section 3 discusses the preparation and use of Stata programs used to auto-populate the table shells.

# Create file directory structure

Populating the midline results tables involves several files. The proper functioning of the Stata programs to auto-populate the tables requires that these files are stored in carefully designated and easily accessible file directories.

Table 1.1 shows an example of the file directory structure used to test and run the Stata programs. To work efficiently and accurately when using the Stata programs, a similar file directory structure is suggested.

Table 1.1: File Directory Structure Used for Stata Programs

|  |  |
| --- | --- |
| **Directory** | **Description** |
| \Project | Project name |
| \Country | Survey country name |
| \Year | Year of survey |
| \Data | Store household and persons analytic data sets (Stata) |
| \Syntax | Store Stata do files |
| \Output | Save and store all intermediate data files (Stata) |
| \Log | Store Stata log files |
| \Table | Store results table (Excel) |

Because file directory names are usually long and complicated, Stata programs use global macros to refer to the longer file directory names for convenience and efficiency, and to reduce errors. In Stata, a global macro is something that is stored in memory and can be used anytime during a Stata session by referencing its name. For example, the standard Stata command to load the household analytic data file “FTF ZOI [country][year] household analytic data.dta” stored in “C:\Project\Country\Year\Data” folder is as follows:

*use “C:\Project\Country\Year\Data\ FTF ZOI [country][year] household analytic data", clear*

If a global macro (data) is defined to refer to the above directory path as follows:

*global data* “*C:\Project\Country\Year\Data”*

then, the household data file can be loaded with the following Stata command:

*use “$data\FTF ZOI [country][year] household analytic data", clear*

Table 1.2 shows the list of all global macros used to refer to the file folders in the Stata programs. The Stata programs use these global macros to access, process, and store data, so the same global macro names for the associated folder names MUST be used to run the Stata programs. These Stata commands defining the file directory structure and associated global macros must be added at the beginning of each Stata program (see Section 3 for more details). If the file directory structure is not specified properly as described in this guide, the Stata programs will give an error message.

Table 1.2: Stata Commands to Define Global Macros to Refer to File Directory Paths

|  |
| --- |
| *global data "C:\Project\Country\Year\Data"*  *global syntax "C:\Project\Country\Year\Syntax"*  *global output "C:\Project\Country\Year\Output"*  *global log "C:\Project\Country\Year\Log"*  *global table "C:\Project\Country\Year\Table"* |

# Prepare analytic data for comparative analysis

The table shells include tables for both comparative analysis of midline and baseline ZOI survey data and descriptive analysis of only midline ZOI survey data (i.e., data available only at midline, including climate, program participation, and water insecurity). The first step to prepare the midline and baseline analytic datasets for comparative analysis is to create all variables needed to calculate Feed the Future indicators, all relevant disaggregates, and any other results to be reported as specified in the table shells. All relevant analytic and survey design-related variables (strata, cluster, weight) must have the same names and value labels in both baseline and midline datasets. The next step is to combine baseline and midline datasets by appending them. The combined dataset must include a variable (survey) that identifies whether a record comes from the baseline or the midline survey. In the Stata programs, survey=“BL” for baseline data and survey=“ML” for midline data. The detailed procedures to prepare analytic data for tabulation are discussed as follows.

**2a. Prepare the midline ZOI data**

Midline data are exported from CSPro as two files, a household-level data file and a person-level data file. Using the midline household and person-level data exported from CSPro into Stata format, prepare two analytic datasets, one for household-level data and another for person-level data, using the customized version of relevant Stata do files available in the ZOI Survey Methods Toolkit. In both cases, create all variables needed to calculate Feed the Future indicators, all relevant disaggregates, and any other results to be reported. Create a variable to identify the survey type. After creating all the necessary variables, save the two analytic datasets in the output folder. Stata commands to create midline person-level and household-level analytic datasets are as follows:

**Person-level data**

*use “$data\FTF ZOI Survey [Country] [YEAR] Person analytic data midline”, clear*

*gen survey=”ML”*

*gen all==1 ///Need this var in the program to auto populate table for all person*

*…..create all necessary variables*

*save “$output\FTF ZOI Survey [Country] [YEAR] Person analytic data\_ML”, replace*

**Household-level data**

*use “$data\FTF ZOI Survey [Country] [YEAR] Household analytic data midline”, clear*

*gen survey=”ML”*

*gen all==1 ///Need this var in the program to auto populate table for all household*

*…..create all necessary variables*

*save “$output\FTF ZOI Survey [Country] [YEAR] Household analytic data\_ML”, replace*

**2b. Prepare the baseline ZOI data**

Baseline data may be available in two data files like the midline data (person/individual-level and household-level datasets) or in several data files (household member/roster, women, children, farmer, A-WEAI, etc.).

If the baseline data come in two data files (household and person/individual), follow the same steps used for the midline, in Step 2a, to create two analytic datasets.

If the data come in several data files, merge the relevant files to create the household-level and person-‑level datasets and create a variable to identify the survey type.

The baseline data may contain derived/analytical variables that are similar or identical to those generated for the midline. The analyst should verify and validate the attributes of each analytic variable, such as variable name, label, and data type, and standardize them with the midline variables. If any analytic variables are missing in the baseline datasets, determine whether they need to be created or are not relevant. After generating all the necessary variables, save the two analytic datasets.

The steps to create baseline person-level and household-level analytic datasets are as follows:

1. If the baseline data come in two data files:

**Person-level data**

*use “$data\FTF ZOI Survey [Country] [YEAR] Person analytic data baseline”, clear*

*gen survey=”BL”*

*gen all==1 ///this var is used later*

*…..create or validate all necessary variables*

*save “$output\FTF ZOI Survey [Country] [YEAR] Person analytic data\_BL”, replace*

**Household-level data**

*use “$data\FTF ZOI Survey [Country] [YEAR] Household analytic data baseline”, clear*

*gen survey=”BL”*

*gen all==1*

*…..create or validate all necessary variables*

*save “$output\FTF ZOI Survey [Country] [YEAR] Household analytic data\_BL”, replace*

1. If baseline data come in several files:

**Person-level data**

*use “$data\FTF ZOI Survey [Country] [YEAR] module1 persons level analytic data baseline”, clear*

*merge 1:1 hhea hhnum m1\_line using “$data\FTF ZOI Survey [Country] [YEAR] module2 persons level analytic data baseline”*

*…merge module3 …..*

*gen survey=”BL”*

*gen all==1*

*…..create or validate all necessary variables*

*save “$output\FTF ZOI Survey [Country] [YEAR] Person analytic data\_BL”, replace*

**Household-level data**

*use “$data\FTF ZOI Survey [Country] [YEAR] module1 household level analytic data baseline”, clear*

*merge 1:1 hhea hhnum using “$data\FTF ZOI Survey [Country] [YEAR] module2 household level analytic data baseline”*

*…merge module5 ….*

*gen survey=”BL”*

*gen all==1*

*…..create or validate all necessary variables*

*save “$output\FTF ZOI Survey [Country] [YEAR] Household analytic data\_BL”, replace*

**2c. Append the baseline and midline datasets to create a combined dataset**

After baseline and midline data are prepared following the procedures discussed in Sections 2a and 2b, append both datasets to create two combined datasets, one for household level and the other for person level. The steps to append both datasets are as follows:

Create person-level combined data:

*use $output/FTF ZOI Survey [Country] [YEAR] Person analytic data\_ML, clear*

*append using $output/FTF ZOI Survey [Country] [YEAR] Person analytic data\_BL*

*tab survey*

*sort hhea hhnum m1\_line*

*save $output/FTF ZOI Survey [Country] [YEAR] Person analytic data\_ML\_BL, replace*

Create household-level combined data:

*use $output/FTF ZOI Survey [Country] [YEAR] Household analytic data\_ML, clear*

*append using $output/FTF ZOI Survey [Country] [YEAR] Household analytic data\_BL*

*tab survey*

*sort hhea hhnum*

*save $output/FTF ZOI Survey [Country] [YEAR] Household analytic data\_ML\_BL, replace*

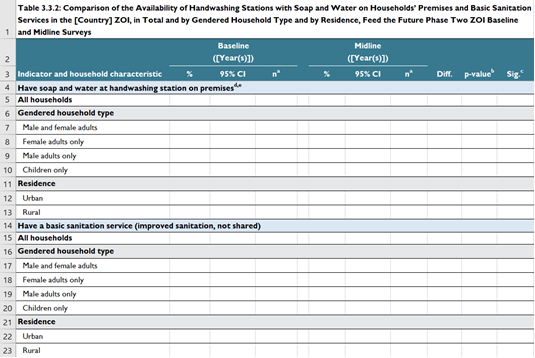
# Develop Stata programs to auto-populate table shells

Two types of Stata programs are used to auto-populate the table shells: (1) Stata programs to develop executable Stata commands, and (2) Stata programs to implement and run these Stata commands. The first type of Stata programs develops Stata commands, such as tabulate and summarize, to estimate indicator values and generate other results and export results. These programs do not include any variables or file names; rather, they include macros to represent the inputs that they require (e.g., file names, variables, Excel row numbers). The second type of Stata programs contain all the inputs to feed the macros used in the first type of programs—the actual file names, variables, Excel rows—when they are called and execute the commands to auto-populate table shells. This section discusses how these two types of Stata programs are used together to auto-populate a table.

**3a. Use Stata programs to develop Stata executable commands**

Currently, the table shells file includes 105 tables in Excel format. Each table shell displays what statistics are to be produced and in which Excel cells the results are to be posted. The title section of the table and table layout identify the indicators and types of indicators (binary, categorical, continuous). Based on this information, tables are grouped together, and one Stata program has been developed for each group (see Appendix Table A1). For example, the tables from Excel table shells ES1, T3.3.2, and T4.1.1 are grouped into TABTYPE\_A because they all have a similar table layout and have binary or continuous indicators. The Stata program that contains the syntax to auto-populate these tables is also called TABTYPE\_A. Each Stata program can handle as many disaggregate variables as needed.

For better understanding of the Stata programs, a step-by-step procedure of how to run the Stata programs is explained below using Table shell T3.3.2. Note that this table shell belongs to group TABTYPE\_A.



Because some tables in this group include both binary and continuous variables (e.g., ES1), the TABTYPE\_A program contains two Stata programs, one for binary variables (TABTYPE\_A\_PROP) and the other for continuous variables (TABTYPE\_A\_MEAN). However, Table shell table T3.3.2 includes two indicators (handwash and basic sanitation service), which are both binary variables, so, TABTYPE\_A\_PROP is used to estimate the proportions of indicators, confidence intervals, differences, and p-values of indicators for this table.

The step-by-step procedures of TABTYPE\_A\_PROP to estimate baseline and midline proportions and 95 percent confidence intervals, the midline-baseline difference, the associated p-value, and level of significance using weighted data and to export results to the designated Excel sheet are presented below. The information in Table 3.1 will help in understanding the use of macros in the Stata syntax.

**Table 3.1: Macros Used in the TABTYPE\_A\_PROP Program**

| **Macros used in**  **TABTYPE\_A\_PROP** | **Information needed to replace macro to auto-populate table** | **Example relevant to Table shell table T3.3.2** |
| --- | --- | --- |
| `1’ | Data set name | *“FTF ZOI Survey [Country] [YEAR] Household analytic data\_ML\_BL”* |
| `2’ | Data subset condition | c0\_59m==1 |
| `3’ | Weight variable | wgt\_c5 |
| $tabname | Excel file name | “Results table” |
| `4’ | Excel sheet name to export results | T3.3.2 |
| `5’ | Row number in Excel to past result | 212 |
| `6’ | Explanatory variable name | csex |
| `7’ | indicator name | chn\_stunted |

The following steps follow the steps in the Stata program TABTYPE\_A\_PROP.

1a. Set the system.

*set more off*

*clear all*

*set maxvar 30000*

1b. Set global macro for directory path.

*global data "C:\Project\Country\Year\Data"*

*global log “C:\Project\Country\Year\Log"*

*global output " C:\Project\Country\Year\Output"*

*global syntax "C:\ C:\Project\Country\Year\Syntax"*

*global table "C:\ C:\Project\Country\Year\Table"*

1c. Set global macro for working dataset.

*global hhdata "C:\Project\Country\Year\ FTF ZOI Survey [Country] [YEAR] Household data analytic\_ML.dta"*

*global psdata "C:\Project\Country\Year\ FTF ZOI Survey [Country] [YEAR] Person data analytic\_ML.dta"*

*global tabname "C:\Project\Country\Year\ FTF ZOI Survey [Country] [YEAR] Results table"*

1d. Set working directory.

*cd "C:\Project\Country\Year"*

2. Load data.

*use "`1'" if `2', clear*

[Local macro `1’ represents dataset name and `2’ represents command for data subset. Use of local macros allows the Stata program to use different datasets and subsets when needed. The end of this section shows when and how to use and change the dataset or data subset to auto-populate a table.]

3. Declare survey design for dataset.

*egen hhea2=group(survey hhea)*

*svyset hhea2 [pw=`3'], singleunit(scaled)*

[Local macro `3’ represents the weight variable. See Appendix Table A3 for the list of weight variables and relevant indicators for which each weight is used.]

4. Define Excel file to export results.

*putexcel set "$tabname", sheet(`4') modify*

[Global macro `tabname’ represents the Excel table shells file name (Results table) as defined in Step 1c. Local macro `4’ represents Excel sheet name where the results will be pasted. For Table shell table T3.3.2, `4’ represents “T3.3.2”.]

5. Define the row to write results for each indicator in the Excel sheet.

*local row = `5'*

[Local macro `5’ represents the row number in Excel where the results will be posted.

6. Define disaggregate.

*levelsof `6', local(subcat)*

[The ‘levelsof’ command identifies all values in a variable (`6’) and puts those values in a macro (subcat).]

7. Estimate proportion and the 95 percent confidence interval for the indicators at baseline.

*foreach var of varlist `7' {*

*foreach i in `subcat' {*

*capture estpost svy,subpop(if `6'==`i'): tab `var' if survey=="BL", col se ci*

*cap mat est=e(b)*

*cap mat lb=e(lb)*

*cap mat ub=e(ub)*

*gen obs1=e(N\_sub)*

*cap gen est1=string(est[1,2]\*100, "%3.1f")*

*cap destring est1,replace*

*cap gen lb1 =string(lb[1,2]\*100, "%3.1f")*

*cap gen ub1 =string(ub[1,2]\*100, "%3.1f")*

*\*\*Set confidence interval in [lb - ub] format*

*egen CI1 = concat(lb1 ub1), punct(", ")*

*cap mat drop est lb ub*

*}*

Where,

`7’ refers to indicator

`6’ refers to the disaggregate variable, `i’ refers to the categories of disaggregate.

estpost posts results from various Stata commands in e().

[Capture (or cap) is useful in do-files and programs because their execution terminates when a command issues a nonzero return code. Preceding commands with the word capture allows the do-file or program to continue if an indicator cannot be estimated for some reason.]

8. Estimate proportion and 95 percent confidence interval for the indicators at midline.

*foreach var of varlist `7' {*

*foreach i in `subcat' {*

*capture estpost svy,subpop(if `6'==`i'): tab `var' if survey=="ML", col se ci*

*cap mat est=e(b)*

*cap mat lb=e(lb)*

*cap mat ub=e(ub)*

*gen obs2=e(N\_sub)*

*cap gen est2=string(est[1,2]\*100, "%3.1f")*

*cap destring est2,replace*

*cap gen lb2 =string(lb[1,2]\*100, "%3.1f")*

*cap gen ub2 =string(ub[1,2]\*100, "%3.1f")*

*\*\*Set confidence interval in [lb - ub] format*

*egen CI2 = concat(lb2 ub2), punct(", ")*

*cap mat drop est lb ub*

*}*

9. Calculate the difference in proportions between two surveys.

*cap gen diff=string((est2-est1), "%3.1f")*

*cap destring diff,replace*

10. Conduct a Rao Scott chi-squared test of difference in proportion.

*cap svy,subpop(if `6'==`i'): tab `var' survey, col se ci*

*cap mat list e(b)*

*cap lincomest \_b[p22] -\_b[p21]*

where,

b[p22] = proportion of an indicator in midline

b[p21] = proportion of an indicator in baseline

lincomest calculates confidence intervals and P-values for linear combinations of model coefficients and saves the results as estimation results.

11. Capture the p-value.

*gen pval=est[1,4]*

*gen pvalx=string(pval, "%4.2f")* ///Converted to string for programming convenience

11a. If (pval>0.01), set p-values to 2 decimal places.

*if (pval>0.01) {*

*replace pvalx= string(pval, "%4.2f")*

*}*

11b. If (pval>=0.001 & pval<0.010) | (pval>=0.045 & pval<0.055), set p-values to 3 decimal places.

*If (pval>=0.001 & pval<0.010) | (pval>=0.045 & pval<0.055) {*

*replace pvalx= string(pval, “%4.3f”)*

*}*

11c. If (pval<0.001), set p-values to “<0.001."

*if (pval<0.001) {*

*replace pvalx="<.001"*

*}*

11d. Destring the variable pvalx for better display in Excel. [Destring the var for prog. convenience.]

*destring pvalx,replace*

11e. Define the level of significance that corresponds with the p-value.

*gen sig="" if pval==.*

*replace sig="\*" if pval<.05*

*replace sig="\*\*" if pval<.01*

*replace sig="\*\*\*" if pval<.001*

*replace sig="n/s" if pval>=.05*

12. Export results to the Excel table shell file, formatted to be centered in the cell, to use size 10 Gill Sans MT font, and to follow the reporting conventions summarized in Appendix Table A2.

*putexcel B`row'=(est1), hcenter font("Gill Sans MT", 10) nobold nformat(0.0)*

*putexcel C`row'=(CI1) D`row'=(obs1), hcenter font("Gill Sans MT", 10) nobold*

*putexcel F`row'=(est2), hcenter font("Gill Sans MT", 10) nobold nformat(0.0)*

*putexcel G`row'=(CI2) H`row'=(obs2), hcenter font("Gill Sans MT", 10) nobold*

*putexcel I`row'=(diff), hcenter font("Gill Sans MT", 10) nobold nformat(0.0)*

*putexcel J`row'=(pvalx), hcenter font("Gill Sans MT", 10) nobold nformat(#.00)*

*putexcel K`row'=(sig), hcenter font("Gill Sans MT", 10) nobold*

*putexcel L`row'=("`var'\_`6'`i'"), left*

13. Update the values of the indicator estimates, confidence intervals, baseline-midline difference, p-value, and significance level based on guidelines as shown in Appendix Table A2.

For example, if obs=0, set est to "—"

*if (obs1==0) {*

*putexcel B`row'=("—"), hcenter font("Gill Sans MT", 10) nobold*

*}*

*if (obs2==0) {*

*putexcel F`row'=("—"), hcenter font("Gill Sans MT", 10) nobold*

*}*

**3b. Use Stata programs to implement and run the Stata executable commands**

As discussed in Section 3a, the Stata executable commands programs cannot auto-populate a table alone, because it does not contain any real variables or file names. It needs another Stata program to run the Stata commands. For this purpose, a Stata program has been developed to feed the Stata executable commands program all the relevant information to auto-populate the table shells. The Stata program to implement and run the Stata executable command programs is called “GEN\_TAB.” GEN\_TAB calls the relevant Stata program and the blank table shells and executes the commands to auto-populate the relevant table associated with the Stata program. This section explains what is included in the “GEN\_TAB” program and how it runs the Stata executable commands program to auto-populate a table.

The following steps show how the GENERATE\_TAB program is used to populate Table shell table T3.3.2.

1. Set global var for directory path and data.

*global data "C:\Project\Country\Year\Data"*

*global syntax "C:\Project\Country\Year\Syntax"*

*global output "C:\Project\Country\Year\Output"*

*global log "C:\Project\Country\Year\Log"*

*global table "C:\Project\Country\Year \Table"*

*global hhdata "C:\Project\Country\Year\Output\ FTF ZOI Survey [Country] [YEAR] Person analytic data ML\_BL"*

*global psdata "C:\Project\Country\Year\Output \ FTF ZOI Survey [Country] [YEAR] Household analytic data ML\_BL"*

*global tabname "C:\Project\Country\Year \Table \ FTF ZOI Survey [Country] [YEAR] Report Template"*

2. Load TABTYPE\_A.

*do "$syntax\TABTYPE\_A"*

3. Run TABTYPE\_A\_PROP.

*TABTYPE\_A\_PROP "$hhdata" "all==1" wgt\_hh “T3.3.2” 5 all "handwash"*

*TABTYPE\_A\_PROP "$hhdata" "all==1" wgt\_hh “T3.3.2” 7 genhhtype\_dj "handwash"*

*TABTYPE\_A\_PROP "$hhdata" "all==1" wgt\_hh “T3.3.2” 12 ahtype "handwash"*

*TABTYPE\_A\_PROP "$hhdata" "all==1" wgt\_hh “T3.3.2” 15 all "san\_impnotshared"*

*TABTYPE\_A\_PROP "$hhdata" "all==1" wgt\_hh “T3.3.2” 17 genhhtype\_dj "san\_impnotshared"*

*TABTYPE\_A\_PROP "$hhdata" "all==1" wgt\_hh “T3.3.2” 22 ahtype "san\_impnotshared"*

These six lines of Stata syntax will auto-populate Table shell table T3.3.2. Now, let us examine each line in Step 3 to understand how this Stata program works.

Line 1: The statement after TABTYPE\_A\_PROPcommand contains *"$hhdata"* and refers to the “household analytic data\_ML\_BL.dta” file. Macro `1’ (in TABTYPE\_A) picks up this dataset when running the program. The second statement contains “all==1,” which is asking Stata to consider all data for calculation. Macro `2’ picks up this dataset or subset when running the program. The third statement (macro `3’) contains the name of a household sampling weight variable “wgt\_hh,” which is the name of the relevant weight variable for the analysis. The fourth statement (macro `4’) contains the name of the tab (Excel sheet) in the Excel table shell “T3.3.2,” where results are posted to populate the table. The fifth statement (macro `5’) contains “5” which refers to the row number in Table shell table T3.3.2 where all the results are posted. The argument in the fifth space along with Step 13 of the Stata executable command programs in Section 3a determines which results will be posted in which row and column. The sixth statement contains variable name “all.” Because “all” has only one value (=1), it will read the “*svy, subpop(if `6'==`i')”* command as “*svy, subpop(if all==1)”* when implementing the TABTYPE\_A\_PROP command for line 1 to estimate indicators for all household. The seventh statement contains “handwash,” which is the name of the indicator (a binary variable). Note that line 1 contains all the information needed to populate row 5 of Table shell table T3.3.2. So if we run line 1, it will auto‑populate row 5 of T3.3.2.

Line 2: The Stata syntax in line 2 of Step 3 will auto-populate rows 7 through 10. The Stata program for this line will calculate proportion and other statistics by gender household type (genhhtype\_dj).

Line 3: The Stata syntax in line 3 will auto-populate rows 12 and 13. The Stata program for this line will calculate proportion and other statistics by residence type (ahtype).

Lines 4, 5, and 6: The Stata syntax in these lines will auto-populate the table from rows 15 through 23.

Table ES1 also belongs to TABTYPE\_A group, so two more examples from Table ES1 are provided to show the use and flexibility of this Stata program.

Example1: The following two commands show the Stata code to populate rows 42 through 47 of Table ES1. This part of the table contains calculation of depth of poverty of the poor indicator (povdepth190). Because povdepth190 is a continuous variable Stata command, TABTYPE\_A\_MEAN is used in this case. Note that weight variable for this indicator is also different (wgt\_hm).

*TABTYPE\_A\_MEAN "$hhdata" "all==1" wgt\_hm "$sheet" 42 all povdepth190*

*TABTYPE\_A\_MEAN "$hhdata" "all==1" wgt\_hm "$sheet" 44 genhhtype\_dj povdepth190*

Example 2: The following Stata commands populate rows 190 through 193 of Table ES1. This part of the table contains the calculation of women of reproductive age consuming a diet of minimum diversity (*whn\_mdd\_w)* indicator, which is a binary variable, calculated for women of reproductive age, and this indicator comes from the person-level data. Thus, the Stata commands are modified accordingly. The first statement after the command (TABTYPE\_A\_PROP) contains “person analytic data\_ML\_BL.dta” file. The second statement contains the subset “*wra*==1” (wra=women of reproductive age). The third statement is the weight variable with *wgt\_w*, the weight variable for women of reproductive age.

*TABTYPE\_A\_PROP "$psdata" "wra==1" wgt\_w "ES1" 190 all whn\_mdd\_w*

*TABTYPE\_A\_PROP "$psdata" "wra==1" wgt\_w "ES1" 192 wra\_agecat whn\_mdd\_w*

# Appendix: Sample Report Template Tables, Reporting Conventions, and Sampling Weight Variables

Table A1: Report Template Types, Associated Stata Macro Syntax File Names, and Report Template Numbers

|  |  |  |
| --- | --- | --- |
| **Report template type** | **Name of Stata macro** | **Tables covered by the macro** |
| Graphical user interface, text  Description automatically generated | TABTYPE\_A | ES1, T332, T411, T412, T413, T423, T511, T531, T541, T542, T543, T544, T545, T811, T831, T911 |
| A picture containing graphical user interface  Description automatically generated | TABTYPE\_B | T322, T331, T333, T523, T631, T632, T633, T642, T645 |
| Graphical user interface, website  Description automatically generated | TABTYPE\_C | T311, T312, T421, T521, T641, T643, T644, T646, T812 |
| Graphical user interface, text  Description automatically generated | TABTYPE\_D | T121, T122, T123 |
| Text  Description automatically generated with medium confidence | TABTYPE\_E | T341, T351, T561, T751, T821, T841 |
| Graphical user interface  Description automatically generated | TABTYPE\_F | TA1341, TA1342, TA1343, TA1344 |
| Text  Description automatically generated | TABTYPE\_G | T422, T424 |
| A picture containing shape  Description automatically generated | TABTYPE\_A11 | TA11 |
|  | TABTYPE\_H | T412, T413 |
|  | TABTYPE\_I | T532, T533 |
|  | TABTYPE\_J | T512 |
|  | TABTYPE\_K | T411 |

Table A2: Reporting Conventions Summary Table



Table A3: Weight Variables and Relevant Group or Indicators

|  |  |  |
| --- | --- | --- |
| **Weight variable name** | **Weight variable description** | **Relevant indicators/results** |
| wgt\_hh | Household | Water, sanitation, and hygiene; climate; comparative wealth index; household-level program participation results; dwelling characteristics; resilience |
| wgt\_fpdm | Primary adult female decision-makers | Abbreviated Women’s Empowerment in Agriculture Index |
| wgt\_c5 | Children under 5 years | Children’s health and nutrition program participation |
| wgt\_c2 | Children under 2 years (baseline only) | Extended breastfeeding, minimum acceptable diet |
| wgt\_w | Women of reproductive age (15-49 years) | Minimum dietary diversity—women, women’s health and nutrition program participation |
| wgt\_f | All targeted value chain commodity (VCC) farmers | Promoted improved practices |
| wgt\_v1 | Crop VCC #1 farmers (maize) | Promoted improved practices |
| wgt\_v2 | Crop VCC #2 farmers (millet) | Promoted improved practices |
| wgt\_v3 | Crop VCC #3 farmers (okra) | Promoted improved practices |
| wgt\_lv | Livestock VCC #1 farmers (sheep) | Promoted improved practices |
| wgt\_hm or wgt\_hhmem | Household members | Poverty, wealth index, Food Insecurity Experience Scale |