STATA 8 for Windows SAMPLE SESSION

Cross-Sectional Analysis

Short Course Training Materials
Designing Policy Relevant Research and
Data Processing and Analysis with STATA 8 for Windows
8th Edition

Department of Agricultural Economics, Michigan State University
East Lansing, Michigan
Revised January 2008

Components of the Cross-Sectional Training Materials

Section 0 - Introduction to the Window structures for STATA 8 for Windows (Stata Results, Review, Variables and Stata Command Windows as well as the Do-File Editor). This section must be read before starting the sample session.

Section 1 - Basic functions

Section 2 - Table Lookup & Aggregation

Section 3 - Tables & Multiple Response Questions and Other Useful Commands

Section 4 - Graphs, tables, publications and presentations, how to bring them into word processor, and use of Survey commands.

Annexes

I - Frequently used Stata commands.

II - Several pages from the socio-economic survey of the smallholder survey in the Province of Nampula, Mozambique (NDAE Working Paper 3, 1992).

Acknowledgments

Funding for this research was provided by the Food Security III Cooperative Agreement between the Department of Agriculture Economics at Michigan State University and the United States Agency for International Development, Global Bureau, Office of Agriculture and Food Security .

STATA 8 for Windows SAMPLE SESSION

SECTION 0 - File structure for STATA 8 for Windows (Data, Commands and Results windows)

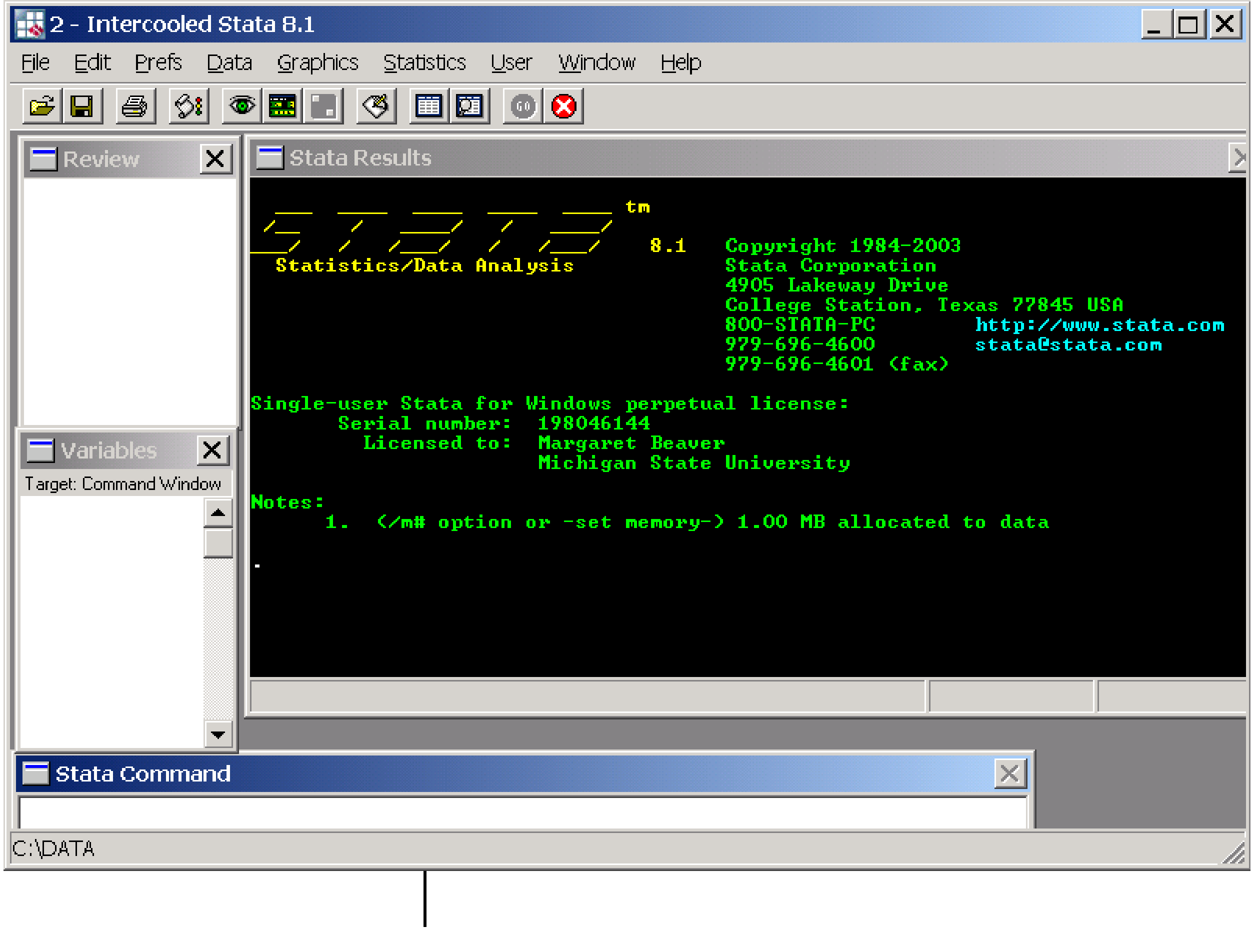
The following module introduces the basic concepts of levels, the notion of cross section analysis, and consequently, the methods of data organization. This module gives a brief description of the file structure of STATA for Windows, version 8. It is essential that you read through this module before starting the cross sectional session.

Overview

When you open STATA 8 for Windows for the first time, you will see four different windows within the program—

- the Stata Results window (results of a command are displayed in this window),
- the Review window(commands submitted to the processor appear in this window),
- the Variables window (the list of variable names in the data set that has been opened) and
- the Stata Command window (where commands can be typed, this is the "active" window at startup).

You can resize and reposition any of the windows in Stata. Below is an example of the default arrangement of the windows.



```
  ___  ____  ____  ____  ____ tm
 /__    /   ____/   /   ____/
___/   /   /___/   /   /___/   8.1   Copyright 1984-2003
  Statistics/Data Analysis           Stata Corporation
                                     4905 Lakeway Drive
                                     College Station, Texas 77845 USA
                                     800-STATA-PC        http://www.stata.com
                                     979-696-4600        stata@stata.com
                                     979-696-4601 (fax)

Single-user Stata for Windows perpetual license:
       Serial number:  198046144
         Licensed to:  Margaret Beaver
                       Michigan State University

Notes:
      1.  (/m# option or -set memory-) 1.00 MB allocated to data

.
```

If you wish to rearrange the windows and keep your new arrangement, from the Menu, select

Prefs /Save Windowing Preferences

or you can just close down the program. To go back to the original window settings, select

Prefs / Default Windowing

Other windows are available, but are not opened at startup. These windows are:

- Stata Viewer (used to view help files and log files, SMCL - markup and control language- files, and print log and other files. This window is not contained in the STATA 8 program window but stands alone and appears on the task bar as another icon.)
- Stata Data Editor (where you can view the data you have loaded into the program's memory)
- Stata Do-file Editor (text editor where you can build a "do" file, a file that contains commands that Stata can execute. This window is not contained in the STATA 8 window but stands alone and appears on the task bar as another icon.)

You can switch between the windows within Stata by using the **Window** choice from the Menu.

Version 8 of Stata provides menus to help the user. However, the user can also type all the commands in the Stata Command window. Throughout this tutorial, if the action desired can be done using the menus, directions will be given on how to use the menus. The Stata command that will do the same action will also be given so that you become familiar with the commands. Stata does not provide a mechanism to paste commands into a file that you can then execute. However, a method is provided to save the commands, that have been executed, into a "do" file where all the commands can be run at once or individually. You can also copy the commands from the Results window and paste them into the Do-file editor. You can also copy commands from the Command window and paste them into the Do-file editor.

How STATA uses memory:

a) The **set memory** command

A data file must be loaded into memory before any analysis can be done. Stata/SE uses 10 megabytes of memory for data, Intercooled Stata uses 1 megabyte of memory and Small Stata uses 300 kilobytes of memory for data. You cannot change the amount of memory used for Small Stata. For the other versions the amount of memory can be temporarily changed or permanently changed. The command to change the memory is:

```
set memory [amount of memory]
```

example:

```
set memory 5m
```

To check to see how much memory is being used and how much is remaining, use the following command:

```
memory
```

Before loading a file into memory, the result of this command in Intercooled Stata is:

```
--------------------------------------------------------------------
Details of set memory usage
    overhead (pointers)                             0        0.00%
    data                                            0        0.00%
                                        ----------------------------
    data + overhead                                 0        0.00%
    free                                    1,048,568      100.00%
                                        ----------------------------
    Total allocated                         1,048,568      100.00%
--------------------------------------------------------------------
Other memory usage
    system overhead                           745,090
    set matsize usage                          16,320
    programs, saved results, etc.                 105
                                        ---------------
    Total                                     761,515
-------------------------------------------------------
Grand total                                 1,810,083
```

After loading a small file, the results are:

```
. use "C:\docs\sample\c-q1a.dta", clear
. memory
                                                bytes
--------------------------------------------------------------------
Details of set memory usage
    overhead (pointers)                         6,096        0.58%
    data                                       67,056        6.40%
                                        ----------------------------
    data + overhead                            73,152        6.98%
    free                                      975,416       93.02%
                                        ----------------------------
    Total allocated                         1,048,568      100.00%
--------------------------------------------------------------------
Other memory usage
    system overhead                           745,090
    set matsize usage                          16,320
    programs, saved results, etc.               1,029
                                        ---------------
    Total                                     762,439
-------------------------------------------------------
Grand total                                 1,811,007
```

One megabyte can be used up fairly quickly, so it is recommended that you set the memory at the beginning of the session to a larger size, e.g.

```
set memory 10m
```

b) How to Set Memory When STATA is started from an icon on the desktop

If you wish to have the memory already set when you start the program, you can edit the command that starts the program and add the parameter for memory.

1. Highlight the icon on your desktop, right click and select **Properties** from the choices.

2. In the **Target:** box, add **/m20** (or whatever amount of memory you want to set it to) so that the command reads:

```
"C:\Stata8\wstata.exe" /m20
```

3. When Stata is installed, the directory to look for data is specified as the directory where the program was installed. (See "Start in" box.) However, Stata remembers where you last opened a file and will use that reference when the program is started the next time.

4. If you have made any changes, click on **Ok** to save the changes.

The next time you start STATA from the icon, the memory will be set and the default directory will be set to whatever directory you have specified. If you start the program from the **Start, All Programs** menu, the memory parameter will not be set unless you modify that shortcut as well within the Stata8 directory.

c) Increasing the Amount of Memory in the Middle of a STATA Session:

The **drop _all** command

If you want to increase the amount of memory in the middle of your session, you will not be able to do so unless you close the data file using the command

```
drop _all
```

Another option is to just close the STATA program and set the memory using the set memory command after you open the program and before you open a data file.

Types of Files Used by Stata and Their Extension Names

1. Data files

- files containing data (Extension *.DTA)

Data files have an extension of .dta From the STATA 8 window, you can open a data file.

From the Menu:

Select **File**, then **Open**.

If you are not in the directory where your files are, change to the appropriate directory. Only files with an extension name of ".dta" will be listed.

From the Command window (if you are working in the correct directory), you can type:

```
use "name of file", clear
```

2. Log files

- commands and output (Extension *.SMCL)
  Stata markup and control language
- commands and output (Extension *.log)
- ASCII text: commands only (Extension *.txt)

Stata can record a copy of the commands and the output from the commands in a “log” file. If you wish to record this information in a file, you must turn on the log. There are two types of logs:

1. **Log:** One records everything that you submit for execution and all the output resulting from the commands. You can specify one of two formats, either SMCL or ASCII text (log)

From the Menu: Select **File**, then **Log**, then **Begin**. You are prompted for a file name. The default extension is SMCL. The file is formatted in a the Stata markup and control language. Give the file a name and click on OK. If you prefer to record the information in ASCII text, then you would need to type the file extension of .log, e.g. session1.log.

The **log using** command

From the Command window, type:

```
log using session1, append
```

- opens a file to record the session and uses SMCL format
- this file can only be opened in STATA.

or type:

```
log using session1, append text
```

- opens a file to record the session and uses ASCII format
- this file can be opened in any text editor or word processor.

2. **cmdlog:** The other type of log file records only the commands, the STATA command is `cmdlog`. This command creates a file that records only the commands.

The **cmdlog using** command

In the Stata Command window, type:

```
cmdlog using session1, append
```

A file is opened which is named “session1", and information will be appended to anything that already exists in this file.

The **log close** command

To close the log, in the Command window, type

```
log close
```

<u>Reminder</u>: The log file that is written in SMCL format can only be opened in STATA. It is a specific format as mentioned earlier. If you want to share your commands and results from the log files with another person who might not have STATA, you should save your log files in the TEXT format with the extension of .log. Any editor or word processor can open this file. However, in the word processor, the font must be set to a fixed font, such as Courier New. Otherwise, the output will be difficult to read.

3. **Do files**

-Stata commands (Extension *.do)

A ".do" file contains commands that Stata can execute. The "do" file is created in the Do-file Editor. The user can type commands or paste commands into the editor. Other ways to create a do file are:

a) You can create a log file that contains only the commands, using the "cmdlog" command, see above.

b) You can select the Review window, click the right mouse button and select "Save Review contents". The extension .do will be automatically added to the file name you enter into the "File name" box.

c) You can copy commands from the Results windows into the Do-file Editor using <**Ctrl C**> to copy what you have blocked in the Results window and then switching to the Do-file Editor and pressing <**Ctrl V**> to paste the command that was copied from the Results window.

d) You can select the command from the Review window, which places it back into the Command window, where you can block the command, press <**Ctrl C**> , switch to the Do-file Editor and press **Ctrl V**> to paste the command.

Option d) may become your preferred method to build the do-file.

Comments to Document the Do-File Commands

Comments can be placed in the do-file as you copy and paste commands. Comments in a do file must start with /* and end with */ so that STATA will not think the comments are commands. An example of a comment is:

/* do file to examine variables using descriptives */

/* your name here and the date the file was created */

Within the Do-file Editor, you can submit several commands at once. (In the Command window, only one command at a time can be submitted for execution.) You cannot send commands directly from the Stata Command window to the Do-file Editor. The command must be copied.

How to Open a Do-File:

There are 2 ways to open the Do-File editor.

1. From the Button Bar, you can click on the "**Do-file Editor**" button 📝. Another window opens which is the Do-file editor.

The **doedit** command

2. From the Command window you can type:
   doedit

It is important to recognize the significance of the different types of files and to understand the various commands you use to create and access the files.

Discussion of the Windows used in STATA

A) The Do-file Editor

The Do-file Editor is the window where commands can be typed before they are submitted to the STATA processor. Commands can be **typed** directly into the Do-file Editor or you can copy the commands from the Results window and paste the commands into the Editor. There are four main uses of the Do-file Editor:

- To type commands directly into the Do-file Editor to be processed later by STATA,
- To send these commands to STATA 8 for Windows for processing,
- To write or save these commands to a file for future use, and
- To retrieve files of commands that you have saved previously.

It is important to understand that the commands you put in the Do-file Editor will not be executed (no output will be produced) until you send the commands to the processor. The Do-file Editor is simply an area that helps you prepare the commands. To send the commands to the processor, you use the **Do current file** button in the Do-file Editor window toolbar of STATA 8 or press <Ctrl-D>. The **Do** command executes the commands in the current do-file. Another button, called **Run current file**, also executes the commands in the current do-file but does not show any output in the Results window. Choosing either one sends all the command(s) to the processor, which reads the commands written in the Do-file Editor and executes them. To send only specific commands, block the commands you want to send and select **Tools / Do selection** or click on the "Do current file" button

When you have successfully completed each step in your analysis (or when you are ready to end a STATA 8 for Windows session, even if it was not completely successful) you should save the commands to a file for future use. To save the commands, make the Do-file Editor active and select **Save** from the **File** menu or click on the diskette symbol on the Tool Bar. A file created from the Do-file Editor is called the *command file*. It is a file containing only commands; it never contains any of the data you may be analyzing with the commands. You must save your data separately, as described in the following section. We suggest that you use the default *extension* of `.do` when naming command files. REP7.DO, DEM-ALL.DO, and SECTION1.DO are some examples.

By storing your commands to a .do file, you can retrieve, look at, or modify sets of commands and rerun them. To retrieve a do-file into the editor, open the Do-File Editor pull down the **File** menu and select **Open** or you can click on the "yellow" file folder 📂 on the tool bar in the editor. Select the file you wish to open and click on Open. Once you have opened a specific file, you can use the commands from the file, without having to recreate or type them again. If you make changes to the command file that you wish to keep, make sure you save them to disk again. Only one .do file can be open at a time.

Caution: From Windows Explorer, if you double-click on a ".do" file, the Stata program will open and all the commands in the do-file will be run immediately. The do-file will not be opened.

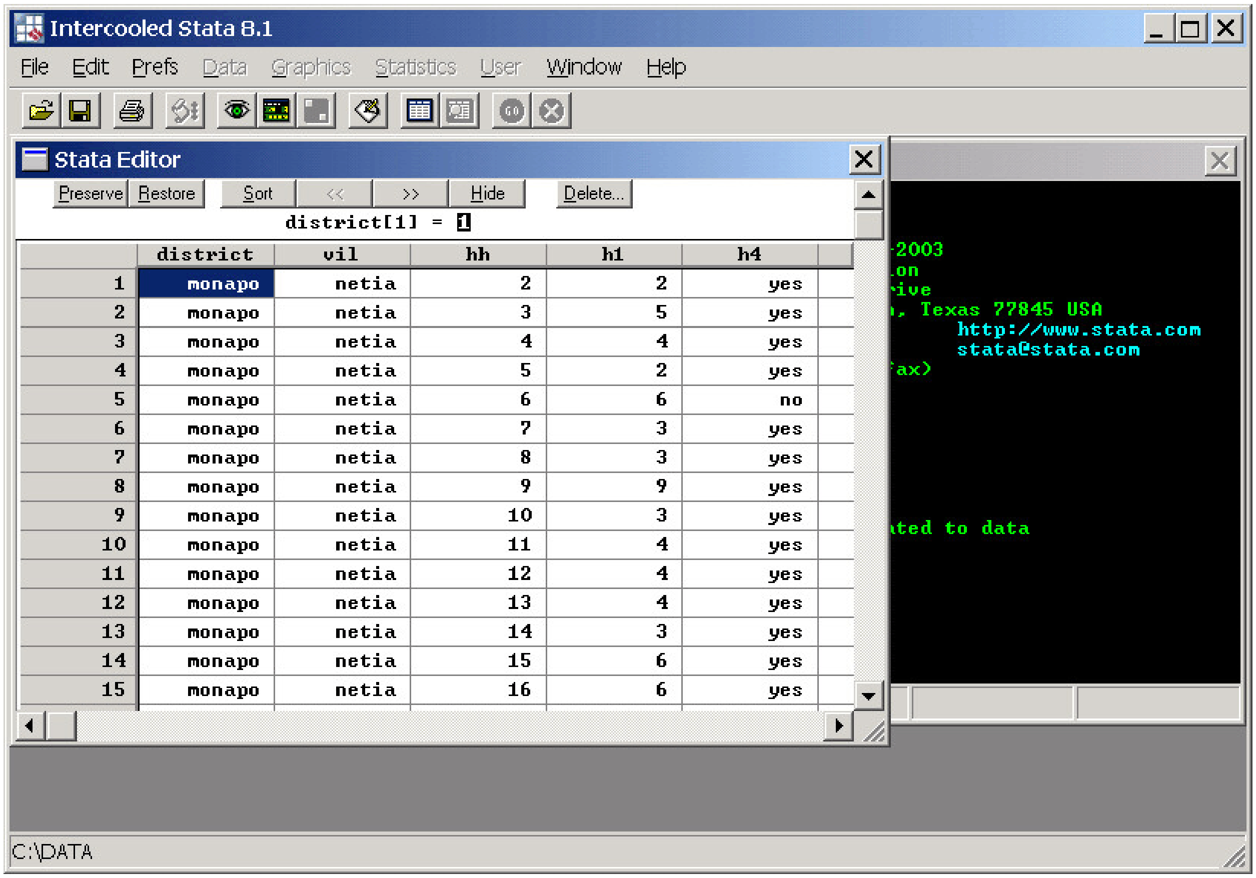
B) The Data Editor Window

STATA 8 for Windows stores your data in a *data file*. In addition to the values themselves, a data file contains such things as variable labels and value labels, formatting information, missing-value specifications, notes, etc. Before you can do any data analysis in STATA 8 for Windows, you must first tell STATA to open a Data file. First select **File** from the menu, select **Open**, highlight a data file (example: c-hh.dta) and click on the **Open** button. The command is immediately run. The data in the file are now available to be viewed in the Data Editor window. In the Review window you see the command that opened the data file. In the Variables window you see the list of variables that are available.

There are 2 methods that you can use to look at the data. The first opens the file in the Data Editor window. In this window you can manually change the data, so be careful when you use this method. The other method opens the data in a browser window where you cannot change any of the values, but you can sort and look at the data.

1. Open Data Editor window

1. The first method to view the data is to open the Data Editor window. Click on the Data Editor button ▦ or in the Command window, type edit and press <Enter>. If value labels have been assigned to the values in a variable, you will see the value label rather than the actual value. Below is an example of a data file with value labels displayed for some variables and values only for other variables. If you wish to see the values rather than the labels, type edit, nolabel in the Command window.



	district	vil	hh	h1	h4
1	monapo	netia	2	2	yes
2	monapo	netia	3	5	yes
3	monapo	netia	4	4	yes
4	monapo	netia	5	2	yes
5	monapo	netia	6	6	no
6	monapo	netia	7	3	yes
7	monapo	netia	8	3	yes
8	monapo	netia	9	9	yes
9	monapo	netia	10	3	yes
10	monapo	netia	11	4	yes
11	monapo	netia	12	4	yes
12	monapo	netia	13	4	yes
13	monapo	netia	14	3	yes
14	monapo	netia	15	6	yes
15	monapo	netia	16	6	yes

```
-2003
.on
ive
n, Texas 77845 USA
         http://www.stata.com
         stata@stata.com
ax)







ated to data
```

Stata makes a backup copy of the data when you enter the Data Editor.

The **Preserve** button will update the backup copy of the data if you have made changes and want to continue to work in the data editor.

The **Restore** button will replace the current data in the editor with the backup copy if you decide you don’t want to keep any changes you’ve made.

The **Sort** button will sort the variable that you have selected. It will not sort multiple variables, just the variable where you cursor is situated.

Exercise: Place your cursor in the variable column labeled **District** and click on the sort button.

The **<<** button will shift the current variable to be the first variable.

The **>>** button will shift the current variable to be the last variable.

The **Hide** button will hide the current variable. It will still be in the data set, it just will not be visible.

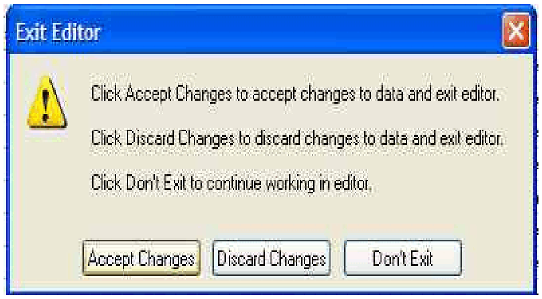
The **Delete** button will give you 3 choices as to what you want to delete. You can delete the variable, you can delete the current observation or you can delete all observations that have the same value as the current observation.

Exercise: Change the value to 1 in the hh column where hh = 3

To exit the Data Editor, click on the "x" in the upper right hand corner of the Stata Editor.

Exiting the Data Editor

When you exit the Stata Editor, a dialog box appears: Click OK to save the changes, Click Cancel to restore the data to its original backup or Preserved copy.



Click Accept Changes to accept changes to data and exit editor.

Click Discard Changes to discard changes to data and exit editor.

Click Don't Exit to continue working in editor.

We do not want to keep the changes, so click on "Discard Changes".

You will often get a data file, compute new variables, make transformations, and finally save the modified set of data to a new name to be used at another time. For example, you might retrieve a data file with land area per crop, add to it production per crop from another file, and then calculate yield. If you want to use these new production and yield variables at a later time, you must make sure that the data file is saved with the new variables in it.

To save a data file, close the Data Editor and then:

Saving the STATA Data File

From the menus select
 File, Save As... and enter the name.

From the Command window, you can also type

```
save "newfilename"
```

The **save, replace** command

or, if you want to use the same name, type

```
save, replace
```

and it will use the same name as the original file.

In the Results window you will see documentation of the Stata commands that are equivalent to what you did in the Data Editor, e.g.

```
- preserve
- sort district
- replace hh = 1 in 2
- restore
```

The dash indicates the change was made in the Data Editor. The data were preserved. We sorted by district, then we changed the value in the variable called "hh" to 1 for the second record. The last line "restore" did not save any of the changes that were made, but restored the data to the original data before we entered the Data Editor.

2. Look at data using the "BROWSE" method

The second method to look at the data is to use the "**Browse**" mode. You cannot modify the dataset if you use this method. This method will prevent you from accidently modifying the data. If you did not close the Stata Editor, you must close it first, before you go into the Browse mode. Click on the browse button. In this window you can sort the data and also hide columns if you wish. To exit the Browser, click on the "x" in the upper right hand corner of the Stata Browser.

Note: *If the Browser is open or the Data Editor is open, the Command window is not available. You must close the editor or browser before you can type any commands or choose from the menus.*

C) The Stata Results Window

STATA 8 for Windows automatically writes all messages and output to the Results Window that are from the execution of your commands. For example, if you run a tabulate command, then the frequency table you specify will be written to the Results window. Similarly, if you generate a table, the table will appear in the Results window. If you wish to save the information in the Results window you must remember to turn a log file on. See the explanation above under "Log files" -page 5.

D) The Command Window

The command window is used to type commands directly. If you use the menus, the command is run immediately. The command is placed in the Review window. If you want to rerun a command that is in that window, click on the command. The command is placed in the Command window. To execute the command, press <Enter>.

Helpful keystrokes within this window.

<PgUp> recalls the last command run and places it in the Command window. If you continue to press <PgUp>, the next command above will be placed in the window.

<PgDown> moves back down through the commands that appear in the Review window.
<esc> clears the contents of the Command window

E) The Viewer

The Viewer in Stata is used to view help files and log files and to print these files. To enter the Viewer, click on **File, View....** A "**Choose File to View**" dialog window opens. You can type the name of the file or click on the **Browse** button. By default, the file type extension name is: SMCL Files (**.smcl**). Select the file and click on **Open**. The file name is pasted into the initial dialog box where you can then click on **Ok** If you decide to use Help from the menus, the Help files are opened in the Viewer.

F) Stata Graph window

A graph is opened in its own window and is not stored in the Results window. If you wish to keep a graph, you can copy the graph to a word processing document or you can save the graph to a file. Right-click on the graph to see these options. A graph file has the extension **.gph**.

Summary of the Basic File Types

Do-file files

Do-file files (or command files) contain commands saved in the Do-file Editor. They do not contain output or data—only commands. Do files are made accessible to STATA for Windows if you open the Do-file editor. Within the Do-file editor you can open a do file.

Log files

Log files contain statistical output, data information and presentation generated by the STATA 8 for Windows processor, given selected commands. They do not contain data. Log files are made accessible to STATA for Windows with a **File, View** command. The extension is *.smcl.

Data files

Data files contain data, including original survey variables plus new variables created through various STATA 8 for Windows commands such as the generate command. Data files are made accessible to STATA for Windows with a **File, Open** command.

STATA 8 for Windows SAMPLE SESSION

SECTION 1 - Basic functions: STATA files, Descriptives and Data Transformation

Introduction

This is a self-paced training aid designed to introduce the commands needed for some typical statistical survey analyses using **STATA 8.0 for Windows**. This tutorial is intended to be a stand-alone training tool. To use it most effectively, you should ask a knowledgeable STATA for Windows user to help you get started and to answer questions as you work independently through the session. It can also be used as a guide for classroom training.

A copy of the questionnaire on which the data is based can be found in the Mozambique project 1992 **NDAE Working Paper 3: A Socio-economic survey of the smallholder survey in the province of Nampula: Research Methods**. Three tables were made available and can be found at the end of the manual in Annex 2 at the end of this document (for further information please contact Dr. Michael Weber at webermi@msu.edu). Four portions of the questionnaire are referenced, each of which has a corresponding STATA for Windows data file. Two other STATA for Windows data files are required for conversion of units of measure.

Questionnaire Section	STATA for Windows Data File
Main Household Section	C-HH.DTA
Table IA: Household Member Characteristics	C-Q1A.DTA
Table IV: Characteristics of Production	C-Q4.DTA
Table V: Sales of Farm Products	C-Q5.DTA
Conversion factors for computing kilograms	CONVER.DTA
Conversion factors for computing calories	CALORIES.DTA

This training consists of four sections, each of which should take approximately two hours. We recommend that you complete each section in a single sitting. These tutorial materials make the following assumptions:

- You know how to use Windows with a mouse
- The six data files listed above are stored in the directory C:\docs\sample on your hard disk. If you have not done so already, you need to copy the files from sample_Stata.zip to this directory.

Important: *Always remember to SAVE the changes to the data after each exercise and module, using a* ***new*** *file name. Also, you may want to save Review window contents to a .do file if you have not been copying commands to a do file already. You may also want to save your log file created during each session.*

Open your STATA software. If you have not read or completed **Section 0,** please do so now to clarify the concept of the Command Window, the Review Window the Results Window, the Do-file Editor and the Viewer.

Data Files and the Working File

Data from questionnaires that has been entered into STATA 8 for Windows are stored in what are called *data files*. If we want to work with a set of data, we must open the corresponding data file so that it is available to the program.

Opening a Data File: The **use** command

When a data file is opened, it is loaded from the disk into memory (the computer's "RAM"), making it the working file. This means that the data from this file is now available for you to use. Let's start with the questionnaire for Table IA: Household Member Characteristics. The data file that corresponds to it is c-q1a.dta. To open this file, perform the following steps:

1. From the File menu, select Open...
   This will open the Open File *dialog box.*

2. Change to the directory where your sample session data are and select the file c-q1a.dta.

3. Click on the **Open** button to open the file. The command appears in the Review window.
   In the Review *Window you will see the text*

```
use "C:\docs\sample\c-q1a.dta", clear
```

4. We want to create a do-file to save our commands. Copy the command, that was just executed in the Results window (using <Ctrl-C>), click on the button in the Tool Bar to open the Do-File Editor and paste the command into this file (<Ctrl-V>). Note also that the use command you just ran has been written to the Review window. You could press <PgUp> which places the command in the Command window where you can copy it and then switch to the do-file editor and paste the command.

5. We want to add comments to define what the do file is about. Above the command that you just pasted, insert some lines. You can type:

```
/* session 1 - basic functions, descriptives */
/* "your name here" - "the current date here" */
   (example:  /* beaver - 10 Oct 2003 */)
/* member level file */
```

Describing the Contents of a Data File:
The **describe** command

The household-member data file is now in memory.

One key thing we often want to know about a data file is what variables it contains. We can find this out, along with other information, by using the **Describe data** command on the **Data** menu:

1. From the **Data** menu select **Describe data...**

2. There are several choices under this option: Select **Describe variables in memory**. A dialog box opens:

There are several options in the dialog box. At the bottom of the dialog box, you have the choices to click on Ok, Cancel or Submit. If you choose **Submit** the dialog box remains open so that you select another option within the dialog box without having to open the box again. If you choose **Ok** the dialog box closes. The command is executed, whichever one you choose. To look at all the variables, leave the variables box empty.

Click on **Ok**

In the Results window, you will see the beginning of the description of the variables. You also see **–more–** at the bottom of the screen. It indicates there is more information to be displayed, but the display has paused so that you can view the information. To continue to the next screen, click the <spacebar> or you could also click on the green button on the tool bar - GO. This button is green only if there is more to be viewed in the Results window.

To obtain the same results from the Command window type

```
describe
```

Copy the command from the Results window and paste it into the do-file.

The output shows the file name, the number of observations, the number of variables, the size and then information about each of the variables, the storage type, the display format the value label and variable label.

```
Contains data from C:\docs\sample\c-q1a.dta
  obs:         1,524
 vars:            11
 size:        73,152 (93.0% of memory free)
-----------------------------------------------------------------
                storage display     value
variable name     type  format      label       variable label
-----------------------------------------------------------------
district          float %9.0g       district    district
vil               float %9.0g       vil         village
hh                float %9.0g                   household
mem               float %9.0g                   member number
ca1               float %9.0g       ca1         does this person work?
ca2               float %9.0g       ca2         relation to head
ca3               float %9.0g                   age
ca4               float %9.0g       ca4         sex
ca5               float %9.0g       ca5         level of schooling
ca6               float %9.0g       ca6         marital status
univ              float %9.0g       univ        where entered
-----------------------------------------------------------------
Sorted by:
```

Data storage types

An explanation of each of the columns follows:

Storage type: Stata has 6 storage types:

Float - real numbers, 8.5 digits of precision, width of 8 with 5 decimals, default unless another type is specified
Double - real numbers, 16.5 digits of precision, width of 16 with 5 decimals
byte - integer between -127 and 100
int - integer between -32,767 and 32,740
long - integer between -2,147,483,647 and 2,147,483,620
strX - string indicating number of characters, Intercooled Stata maximum size is 244.

Display format: The display format is the third column which describes how the data are to be displayed. Stata will make an assumption with new variables so it is not always necessary to specify the format. Format information always begins with a percent sign "%", to indicate the start of the format information. Refer to the User's Guide, Chapter 15.5 for more details. In this example, the 9 describes the width of the variable. After the decimal the 0 indicates no fixed number of decimals will be displayed. If you wished to see only 2 decimals, the example would be %9.2g. The letter following indicates what type of format:

e - scientific notation, e.g. 1.00e+03
f - fixed format, e.g. 1000.03
g - general format
c - optional along with either f or g will display a comma, e.g. 1,000.03

Variable label: Label describing the variable.

Value label: If the variable has value labels the name of the label appears in this column. Stata assigns a name to the label which contains the values and labels. The label is then applied to the variable.

There are several ways to view the labels and values:

Documenting variables and labels:

The **labelbook** command
The **label list** command
The **codebook** command

One example: If you wish to see what labels have been defined for specific values for the variables that have value labels as indicated above, you can run the command to create a codebook of the labels. From the menus:

1. From the **Data** menu select **Labels /Label values**

2. Select **Produce codebook describing value labels** and click on **Ok**.

In the Command window you can also type

```
labelbook
```

to obtain the same results. This command describes only those variables with value labels. It is a good command to use to document these variables.

Another example: You can select specific variables to see the labels. From the menus:

1. From the **Data** menu select **Labels /Label values**
2. Select **List value labels**
3. A dialog box opens. Click on each label you want to see. The name will appear in the top part of the dialog box.
4. Click on **Ok**.

The listing shows you what values are assigned to a label. A label name can be assigned to multiple variables.

In the Command window you can also type

```
label list ca1 ca2
```

To document all the variables, another command is available:

1. From the **Data** menu select **Describe data...**
2. Select **Describe data contents (codebook)** and click on **Ok**.

In the Command window you can also type

```
codebook
```

In this output each variable is listed with the type of variable, the range of values and gives descriptive statistics for variables based on whether it thinks the variables are continuous or categorical. Stata cannot always tell if the variable is categorical, so it does not also display a frequency table for a categorical variable.

Generating descriptive statistics:

The **summarize** and **tabulate** commands

One of the first things to do at the beginning of analysis is to run descriptive statistics (e.g. counts, averages, maximum, minimum, and standard deviations) for all variables. This type of analysis helps you to find data entry errors. It also gives you a "feel" for what kind of data are in the file, to see that missing values have been defined correctly, etc. It may be tempting to skip this step for some data sets or for some variables, but this is an important step that will almost always save time later and improve analysis. For example, finding out the average age of all respondents may not be something you are interested in knowing, but if the average age turns out to be 91.3 yrs, this would alert you that something is probably wrong.

Basic descriptive statistics can be obtained from two common Stata for Windows commands—**Summarize** and **Tabulate**. **Summarize** is used for continuous variables, while **Tabulate** is used for categorical variables.

There are three types of variables.

1. A *continuous variable* is a variable that does not have a fixed number of values. It measures something, e.g. age, weight, population.

2. A *categorical variable* is a variable that has a limited number of values that form categories or groups to which something belongs, e.g. geographic location, relation to head. For example, look at the Annex 2 - Table IA: Household Member questionnaire. Variable

ca3 (age) is a continuous variable because age can take on many different values. Variable **ca2** (relation to head) is a categorical variable because its values are limited to the categories 1-6.

3. An *indicator variable* is a special type of categorical variable. This type of variable denotes whether something is true, e.g. yes/no questions, or whether a person is male or female. This type of variable contains only 2 categories, i.e., it divides the data into 2 groups.

Using one variable

Start by examining the data in the file. Use the Data Editor window to scroll through your data file. To do this, perform the following steps:

1. Click on the Data Editor button on the Tool Bar or in the Command Window, type **edit** and press <Enter>.

   You could, instead, click on the Browse button since we only want to look at the data.

2. Scroll through the data.

   A period in a field indicates a missing value or system missing value. In Stata you can specify up to 27 different missing values, e.g. .a or .b and are called “extended” missing values. Extended missing values are used to identify specific reasons why there are no data, e.g. person refused to answer, or question was not asked.

Scrolling through the data will give you a "feel" for what your data are like. It might also help point out obvious errors, e.g. a variable whose values are missing for all listed cases.

Decide which of the variables in this file are continuous and which are categorical (normally you would refer to the questionnaire to make this decision). You need to know this in order to select the right procedure to use for each variable. If you mistakenly perform a **Tabulate** on a continuous variable, you will probably get more output than you really want, with possibly hundreds of different "categories", one for each different value found. If you perform a **Summarize** on a categorical variable, you will usually get meaningless results, since the average value of a variable that consists of categories has no real significance.

By examining the data, you should have found that variable **ca3** (age) is continuous and the remaining variables are categorical. To run descriptives on **ca3,** do the following:

Descriptives

The **summarize** command

1. From the **Statistics** menu select
   Summaries, tables & tests
   Summary statistics
   Summary statistics
   This will open the summarize - Summary Statistics *dialog box.*
2. Click in the **variables** box so that your cursor rests there. Then click on **ca3** in the Variables windows. In the **options** section below the variable box, note that "Standard Display" is the default selection for output.
3. Click on the **Submit** button to run the command. The dialog box will remain open.

The output appears in the Stata Results window. You will see that the mean for age (**ca3**) is 21.33602 years.

The Stata command is

```
summarize ca3
```

The Results window displays:

```
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |      1524    21.33602    17.69252         .5         81
```

If we wanted more statistics, in the summarize - Summary Statistics dialog box (which is still open)

4. Click on the radio button next to "Display additional statistics"
5. Click on the **Ok** button to run the command. The dialog box will close.

The results are:

```
                            age
-------------------------------------------------------------
      Percentiles      Smallest
 1%            1             .5
 5%            1             .6
10%            3              1       Obs                1524
25%            7              1       Sum of Wgt.        1524

50%           16                      Mean           21.33602
                        Largest       Std. Dev.      17.69252
75%           32             75
90%           48             76       Variance       313.0252
95%           57             78       Skewness       .9152221
99%           69             81       Kurtosis        3.00135
```

The median age is 16 (50% - Percentile).

The Stata command is

```
summarize ca3, detail
```

Copy this command to the Do-File Editor.

Frequencies

Since the variables ca1 (work on a farm or not), ca2 (relation to head), ca4 (sex), ca5 (level of schooling) and ca6 (marital status) are categorical, we will run a Tabulate on them. To run a tabulation, do the following:

The **tab1** Command

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   Multiple one-way tables
   The* tab1 - One-way tables *dialog box opens.
2. Make sure that your cursor is in the Variables box which is found under the tab labeled Main. Then you can select the variables you want from the Variables window.
   ca1 ca2 ca4 ca5 ca6
3. Click on the **Ok** button.
4. The command will be executed. Copy the command to the Do-File Editor.

You will see in the Stata Results window. If you have turned "more" off, then you will want to scroll up through the Results window to find **ca1**. For **ca1** 70.67% of the household members work on a farm. The results for **ca6** show that 37.99% of those surveyed are in monogamous marriages.

The Stata command is:

```
tab1 ca1 ca2 ca4 ca5 ca6
```

Note: *to produce a tabulation (frequency) of just one variable, you can use the **tabulate** command. However, if you want to list several variables in the frequency command, you must use the **tab1** command. Below, you will see that if you use the tabulate command and list 2 variables, you produce a cross-tabulation.*

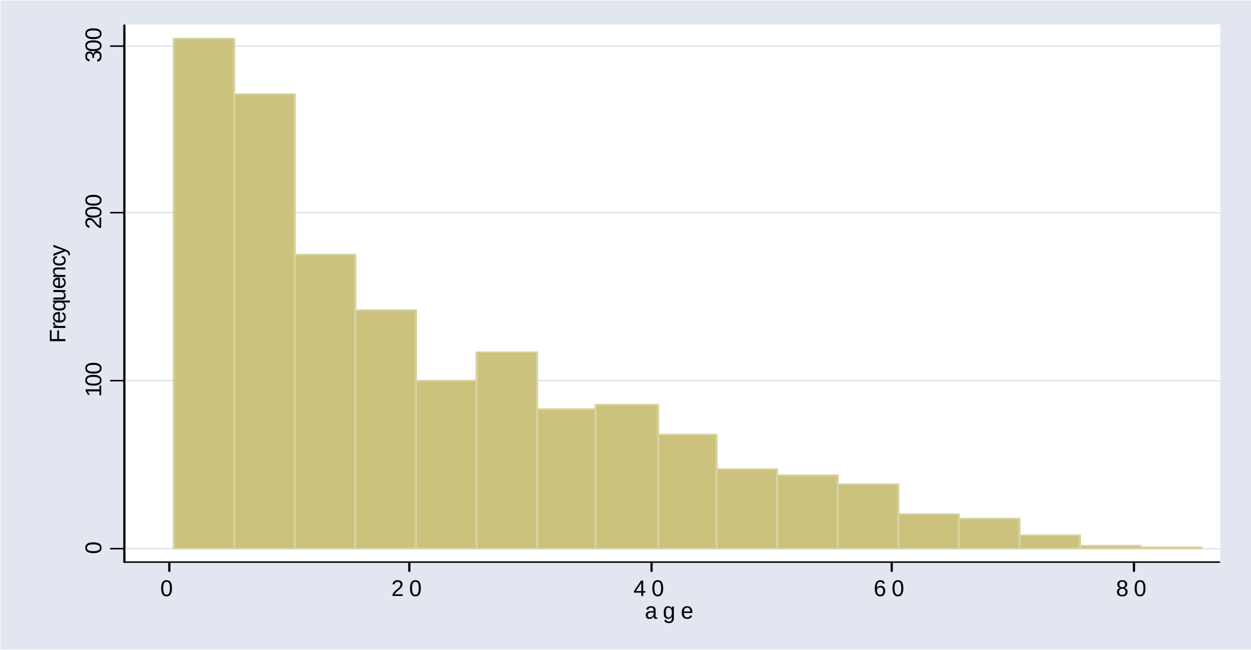
The **histogram** command

Another useful way to examine a continuous variable is to Graph the variable to view the distribution of the values. From the menus select **Graphics, Histogram**

1. Be sure your cursor is in the correct place and select ca3 from the Variables window.
2. Check the box ✓ for **Width of bins** and type in **5** in the box next to this option. The ages will be grouped into 5 year ranges.
3. For the Y-axis click on the radio button next to "Frequency" so we will see the number of cases in the age groups.
3. Click on **Ok** to run the command.

The Stata command is:

```
histogram ca3, width(5) frequency
```



Saving a graph to a file:

If you want to save this graph to a word processing document, you can right click on the graph, select “copy graph”, then switch to your word processor and paste it into the document. If you want to save the graph to disk, right click and choose “save graph”.

Note: *Only one graph appears in the graph window at a time. If you run multiple graph commands at one time from a do-file, only the last graph will be visible. You must run one graph command, then save or copy the graph before you run the next graph command.*

For a more detailed description of the sub-commands available for **Summarize** and **Tabulate** refer to the **Guide for STATA References S-Z.**

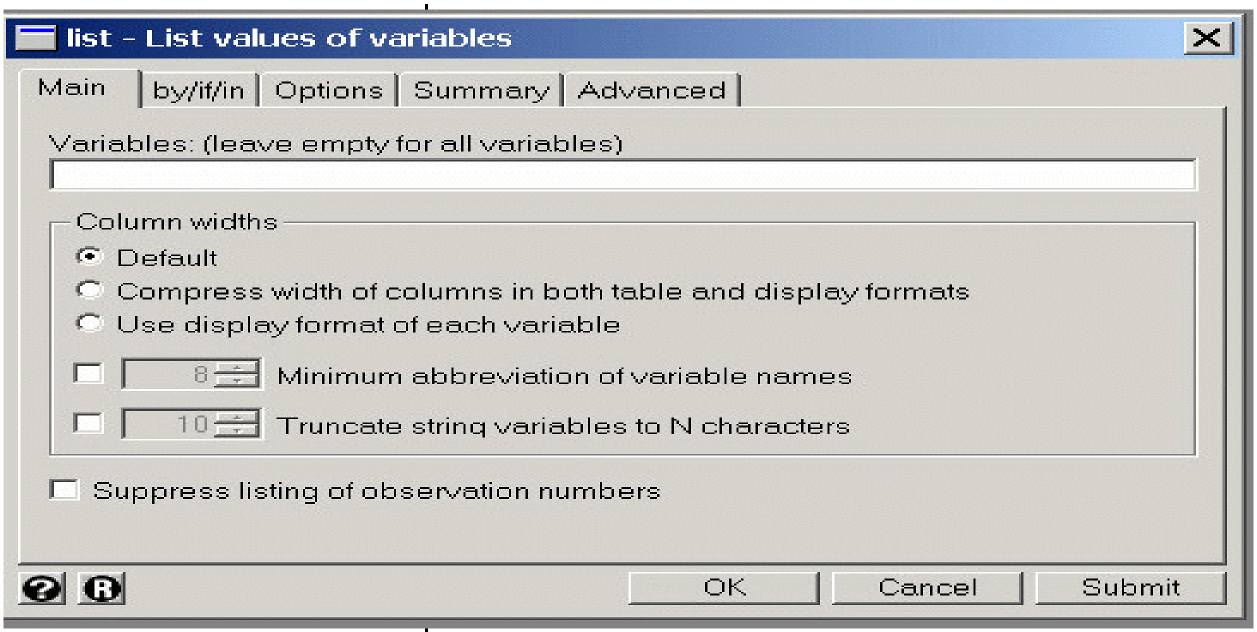
The **list** command

You may want to look at the data selecting only specific cases rather than scrolling down through the data set to find a specific case or cases. The list command gives you the option to select all or specific cases.

1. From the **Data, Describe Data** menu select **List Data**

The list dialog box - List values of variables has 5 tabs where you can set specific parameters for the data that you want to list.

On the **Main** tab you can specify the variables to be listed or leave it blank to list all variables. The default column width separates each variable by 5 spaces and shows the variables in "display" format. Below is an example:


list - List values of variables
Main
by/if/in
Options
Summary
Advanced
Variables: (leave empty for all variables)
Column widths
● Default
○ Compress width of columns in both table and display formats
○ Use display format of each variable
□ 8 Minimum abbreviation of variable names
□ 10 Truncate string variables to N characters
□ Suppress listing of observation numbers
OK
Cancel
Submit

2. Place your cursor in the variables box and select the variables from the Variables window
   district vil hh mem ca1 ca2 ca3 ca4 ca5 ca6
   Note: if you wished to include all variables, leave the box empty.
3. Click on the tab labeled "by/if/in"
   In this tab we can limit the number of cases that are displayed.
4. Check the box ✓ next to "Obs in range". Specify the range to be from 1 to 10.
5. Click on the "Options" tab. Under "Table options" check the box ✓ next to "**Force a clean table**".
6. Click on **Ok** to run the command.
7. In the Results window you see a list of the observations. If the information for each observation is wrapping to the next line, you can resize the Results windows so that it is wider. Place your mouse pointer on the right border of the window and when you see a double arrow, click the <Left Mouse Button>, hold it and drag the right side out to make the window wider.

You see the -More- at the bottom of the Results window. To see the next screen, there are several methods you can use:

"More" options

Press <Enter>
Press any key
Click on the **More** button on the tool bar
Click on the –more– at the bottom of the Results window

If you wish to interrupt a Stata command, you can

click on the **Break** ⊗ button on the Tool bar or
press <Ctrl-Break> or
type q (the letter q for quit) in the Command window.

If you wish to turn the -More- off so that the output in the "results" window is shown completely, you can turn "more" off. The command is:

```
set more off
```

8. To rerun the command you just ran, click on the last command in the Review windows. You see the command is now in the Command window. Press <Enter> to run the command.
9. Copy the command to the Do-file editor and add comments to explain what you have done.

The Stata command should look like

```
list district vil hh mem ca1 ca2 ca3 ca4 ca5 ca6 in
1/10, clean
```

If you wish to, you can type the list command in the Command window. If you are typing in the command window, you can pick the variables from the Variable window and the name will be pasted into the Command window.

Note that to list a subset of observations, Stata uses the key word "**in**", e.g. in 1/10. The key word "IN" restricts the list to a range of observations. Examples are:

list in 1	lists first observation
list in -1	lists last observation
list in 2/4	lists observations 2 through 4
list in -3/-2	lists 2 observations starting with the 3rd from the last observation.

To limit the listing to a specific criterion use the "if" key word. Examples are:

```
list district vil hh mem ca3 if ca3 > 70
list district vil hh mem ca2 ca3 if ca3 < 15 & ca2 < 3
```

If the variables to be listed are in the order you want to see, you can type the first variable, then a dash (-), then the last variable in the list. In this listing we are looking for cases where the age is less than 15 and the relationship to head is either head or spouse.

```
list district-ca3 if ca3 < 15 & ca2 < 3
```

If we want to see the observations with the five lowest values and five highest values, we would first sort by that variable and list the first five cases and the last five cases. For example, if the question is: What is the age of the 5 youngest head of households and what is the age of the 5 oldest head of households?

Stata commands:

```
sort ca2 ca3
list district vil hh mem ca1 ca2 ca3 in 1/5
gsort -ca2 +ca3
list district vil hh mem ca1 ca2 ca3 in -5/-1
```

Note: missing values sort as higher values.
<u>Reminder</u>: after any command is run, we will copy the command into the Do-file Editor window.

Apply what you've just learned about descriptive statistics by doing the following exercise.

Exercise 1.1

Run descriptive statistics on another sample file. Use the production questionnaire - Table IV, whose data is in file C-Q4.DTA.

Hints:

a. make C-Q4.DTA your working data file.
b. Use the **Summarize** command for continuous variables, and **Tabulate or tab1** for categorical variables.
c. **Prod** is a categorical variable.
d. Quantities (**p1b**, **p2b**, ...) are continuous variables.

e. Units (**p1a**, **p2a**, ...) are categorical variables.
f. **p4** (month in which stocks ran out last year) **&** **p6** (month in which stocks will run out this year are categorical variables.

A small sampling of what you should find from running these frequencies and descriptive statistics follows:

```
Tabulate:
              product |      Freq.     Percent        Cum.
----------------------+-----------------------------------
               cotton |         83        4.90        4.90
              peanuts |        144        8.51       13.41
           rough rice |        155        9.16       22.56
              bananas |         50        2.95       25.52
         sweet potato |         12        0.71       26.23
        cashew liquor |         24        1.42       27.64
    sugar cane liquor |         11        0.65       28.29
         dried cashew |          2        0.12       28.41
           sugar cane |         13        0.77       29.18
           cashew nut |        130        7.68       36.86
              coconut |         45        2.66       39.52
                beans |        279       16.48       56.00
       manteiga beans |          7        0.41       56.41
            sunflower |          5        0.30       56.70
              oranges |         13        0.77       57.47
         cashew fruit |         44        2.60       60.07
               manioc |        338       19.96       80.04
              sorghum |        124        7.32       87.36
                maize |        192       11.34       98.70
             "ossura" |          5        0.30       99.00
              tobacco |          4        0.24       99.23
               tomato |         13        0.77      100.00
----------------------+-----------------------------------
                Total |      1,693      100.00

Summarize:

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
         p1b |      1670    26.35286    163.4359          0       5000
         p2b |      1598    22.81508    159.5101         .5       5000
         p3b |       173    2.523121    4.574581          0         30
         p5b |      1231    15.61243    86.10356          0       1460
         p7b |       869    4.938435    6.875536          0        100
```

Descriptive Statistics - involving two or more variables

Two-way Tables with Categorical Variables (Cross-tabulation)

We wish to produce a table that shows the distribution of cases according to their values using two or more categorical variables.

Look at the household member questionnaire in the annex section, Annex Table IA. One thing you might be interested to know is how the gender of the respondents varied by relationship to the head of household. This would tell you, for example, how many females are heads of households. The **Tabulate** command will produce this

The **tabulate** command

type of summary. Make the household member file, c-q1a.dta, the working data file.

1. Click on the yellow open folder tool at the top left of the Toolbar
2. Select the file c-q1a.dta.
3. Click on **Open** to open the file.
4. Copy the command for opening the file which appears in the Results window, into the Do-file Editor window.

Reminder: You should add comments to your do-file so that you can remember what and why you were doing specific commands when you developed the do-file. Several days or weeks from now you may not remember. Comments in a do-file start with slash and then an asterisk and end with an asterisk and a slash:

```
/* this is a comment */
```

Stata will not run a line as a command if it begins with these symbols.

To create a two-way table do the following:

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   Two-way tables with measures of association
   The tabulate2 - Two-way tables dialog box opens.
2. In the **Row Variable** box choose **ca2** from the drop down list.
3. In the **Column Variable** box choose **ca4** from the drop down list.

We would like to see row percentages and column percentages.

4. Under Cell Contents click in the box next to **Within column relative frequencies** to put a ✓.
5. Click in the box ✓ next to **Within row relative frequencies**.
6. Click on the **Ok** button. The command will be executed.
7. Copy the command from the Review window into the Do-File Editor and write a comment to explain what the command does.

The Stata command is:

```
tabulate ca2 ca4, column row
```

The Key box in the Review window specifies which statistics appears on each row in the cells.

```
+-------------------+
| Key               |
|-------------------|
|     frequency     |
|  row percentage   |
| column percentage |
+-------------------+

   relation to |          sex
          head |         m          f |     Total
---------------+----------------------+----------
          head |       321         21 |       342
               |     93.86       6.14 |    100.00
               |     41.42       2.88 |     22.74
---------------+----------------------+----------
  wife/husband |         2        306 |       308
               |      0.65      99.35 |    100.00
               |      0.26      41.98 |     20.48
---------------+----------------------+----------
  son/daughter |       374        336 |       710
               |     52.68      47.32 |    100.00
               |     48.26      46.09 |     47.21
---------------+----------------------+----------
 mother/father |         1          5 |         6
               |     16.67      83.33 |    100.00
               |      0.13       0.69 |      0.40
---------------+----------------------+----------
other relative |        77         61 |       138
               |     55.80      44.20 |    100.00
               |      9.94       8.37 |      9.18
---------------+----------------------+----------
         Total |       775        729 |     1,504
               |     51.53      48.47 |    100.00
               |    100.00     100.00 |    100.00
```

In this case we wanted counts, row percentages, and column percentages. Row percentages sum to 100 across all the cells in a row, while column percentages sum to 100 across all the cells in a column. The table produced by this command tells you that there are 21 female heads of households, and that 6.14% of the total number of heads of households are female (row percent). Of those who are female, 2.88% are head of household (column percent).

Summary statistics on a continuous variable for each value in a categorical variable

The **bysort ...: summarize** command

For this analysis the same command is used as for general summary statistics with a slight modification. This command will show how the mean and other statistics for a continuous variable differ by the values of one or more categorical variables.

Suppose we want to know how the age of the members varied by their relationship to the head of household. If we did this with **Tabulate** we would get a table with dozens of cells for the different ages represented, which would be in an unusable format. Instead we will use **Summarize** using the "by" key word.

1. From the **Statistics** menu select
   Summaries, tables & tests
   Summary statistics
   Summary statistics
   The summarize - Summary Statistics dialog box opens.
2. With your cursor in the "**variables**" box select **ca3** from the Variables window..
3. Be sure that the under "Options" in this tab, **Standard Display** has been selected.
4. Click on the "**by/if/in**" tab.
5. Click in the box "**Repeat command for groups defined by**"
6. In the box below this option, select **ca2**
7. Click the **Ok** button. The command will be executed.

This command calculates the means of the variable **ca3** (age) separately for each different value of the variable **ca2** (relation to head) including the system missing value.

The Stata command is:

```
bysort ca2: summarize ca3
```

Note that the command begins with "**bysort**". This command is first sorting the data by **ca2** before it runs the summarize command. You could also sort the file by **ca2** first and then just use the "**by**" key word, e.g.

```
sort ca2
by ca2: summarize ca3
```

```
______________________________________________________________________________
-> ca2 = head
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |       343     41.5277    14.12719         18         81

______________________________________________________________________________
-> ca2 = wife/husb
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |       310     33.1871    11.80466         13         76

______________________________________________________________________________
-> ca2 = son/daugh
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |       718    8.133844    5.797507         .5         48

______________________________________________________________________________
-> ca2 = mother/fa
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |         6    48.16667    22.09449         20         69

______________________________________________________________________________
-> ca2 = other rel
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |       143    12.55245    10.06785          1         75

______________________________________________________________________________
-> ca2 = .
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         ca3 |         4          15    12.24745          6         33
```

From this output you find that the average age of the head of household is 41.5277 years while the average age of a spouse is 33.1871 years.

Data Transformations

After examining the results of the descriptive statistics you will often want to do data transformations. A data transformation is an operation that takes an existing variable and either changes values in a systematic way or uses the values to calculate a new variable. The following example shows a common data transformation: the conversion of a continuous variable to a categorical variable.

The information we received from the **summarize** command is interesting, but it might also be useful to see the actual distribution of the ages into groups or categories, so we can tell, for example, how many heads of household are older than 60. Since the age variable, **ca3**, is continuous, we cannot do this directly—first we have to transform it. Let's suppose we're interested in four categories: 0-10 years old, 11-19 years, 20-60 years, and over 60 years of age.

Converting continuous variables to categorical variables

The **generate** command
The **replace** command
The **label variable** command
The **label define** command

To categorize a variable, we can use the **generate** command. Categorizing a continuous variable makes detailed information more general. To keep the detailed information as well as the new general information, you must recode the variable into a new variable. If you recode into the same variable the original values will be lost.

There are several methods that can be used to recode a continuous variable.

First method:

First method: If you wish to see the category values of 1, 2, 3, and 4 where

1 = 0-10,
2 = 11-19,
3 = 20-60 and
4 = over 60

you can do the following:

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable**
   The generate - Generate a new variable dialog box opens.
3. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **age_gp**
4. For the **Contents** box, type in
   1
   This is the value that you want the new variable to have.
5. Click on the **New variable(s) type** drop down box and change to **byte**.
6. Click on the **if/in** tab.
7. In the **Restrict to observations if** box, type in **ca3 >=0 & ca3 <=10**
   Note: you must use the ampersand symbol (&), not the word "and".
8. Click on **Ok**

The Stata command is:

```
generate byte age_gp= 1 if ca3 >=0 & ca3 <=10
```

In the Results window you can see below the command the statement: (949 missing values generated)

It tells you that of the 1524 cases, 949 have not been assigned a value for the new variable **age_gp**.

Now that the new variable has been created, another command is used assign the codes for the other age groups. That command is the **Replace** command.

9. Select **Create or change variables** from the **Data** menu
10. Select **Change contents of variable**.
    The replace-Replace contents of variables dialog box opens.
11. In the **Variables** box pick the new variable that was just created from the drop down list - **age_gp**
12. Type **2** in the **Contents** box
13. Click on the **if/in** tab.
14. In the **Restrict to observations if** box, type in **ca3 >10 & ca3 <=19**
15. Click on **Submit**. The dialog box remains open and the command is run. In the Results window you see: (271 real changes made)
16. In the dialog box in the **Restrict to observations if** box, change the criteria to: **ca3 >19 & ca3 <=60**
17. Click on the **Main** tab.
18. Type **3** in the **Contents** box
19. Click on **Submit**. The dialog box remains open and the command is run. In the Results window you see: (629 real changes made)
20. In the dialog box, type 4 in the **Contents** box
21. Click on the **if/in** tab.
22. In the **Restrict to observations if** box, change the criteria to: **ca3 >60**
23. Click on **Ok** . In the Results window you see: (49 real changes made)

The Stata commands created and run are:

```
generate byte age_gp= 1 if ca3 >=0 & ca3 <=10
(949 missing values generated)

replace age_gp = 2 if ca3>10 & ca3 <=19
(271 real changes made)

replace age_gp = 3 if ca3>19 & ca3 <=60
(629 real changes made)

replace age_gp = 4 if ca3>60
(49 real changes made)
```

The Results states how many observations were modified after each command was run.

The next step is to verify that the change were made correctly. Run the **Tabulate** command on the new variable.

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   One-way tables
   The tabulate1 - One-way tables dialog box opens.
2. Select the variable
   age_gp
   to place it in the **Variables** box. Click on the **Ok** button.
3. The command will be executed.

The Stata command is:

```
tabulate age_gp
```

There should be 4 codes in the frequency table—1, 2, 3, and 4. We can use the Data Browser to check to see what changes were made. Click on the **Data Browser** button. Close the window when you are finished.

The values do not have any labels to define what 1, 2, 3, and 4 represent. We want to add value labels as well as a variable label.

To assign a variable label:

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the variable: **age_gp**
3. In the **Attach label to variable** (up to 80 characters) box, type
   Age group
4. Click on the **Ok** button.

The Stata command is:

```
label variable age_gp "Age group"
```

To assign value labels to a variable:

1. Click on **Data**, then **Labels,** then **Label values**, then **Define value label**.

Remember, Stata assigns a name to a group of value labels.

2. In the **Define value labels** dialog box, click on the button **Define**
3. In the **Define new label** box, type **age_gp** and click on the **Ok** button.
4. In the next dialog box, **Add value**, type 1 in the **Value** box and in the **Text** box type **0 to 10**
5. Click on the **Ok** button. The dialog box to add values remains open.
6. Type 2 in the **Value** box and in the **Text** box type 11 to 19, and click on the **Ok** button.
7. Type 3 in the **Value** box and in the **Text** box type 20 to 60, and click on the **Ok** button.
8. Type 4 in the **Value** box and in the **Text** box type 61 and older, and click on the **Ok** button. Then click on **Cancel**
9. The values have been assigned value labels to the labeled called "age_gp". Click on the **Close** button to close the **Define value labels** dialog box.

The Stata commands are:

```
label define age_gp 1 "0 to 10"
label define age_gp 2 "11 to 19", add
label define age_gp 3 "20 to 60", add
label define age_gp 4 "61 and older", add
```

The first command creates a label name and defines the label for the first value. The next 3 commands add to the label name and define the labels for the next 3 values.

The last step is to assign this label to the variable - age_gp.

10. Click on **Data**, then **Labels,** then **Label values**, then **Assign value label to variable**.
    The label values - Attach value label *dialog box opens.*
11. In the Variable: box select **age_gp** from the drop down list.
    This is the name of the variable that we want to attach a label to.
12. In the Attach value label box, select the label "**age_gp**"
13. Click on the **Ok** button.

The Stata command is:

```
label values age_gp age_gp
```

Run a tabulate on the variable to see the results.

Another method we can use, which is much easier, is to generate the new variable, assign the new values and assign the labels for the values in one step:

Second method:

1. Select **Create or change variables** from the **Data** menu
2. Select **Other variable transformation commands**
3. Select **Recode categorical variable**
4. In the "**Main**" tab, place your cursor in the **Variables** box and select **ca3** from the Variables window
5. In the **Required** box, specify the range you want and the new value to be assigned as well as the label for that new value, e.g.
   (0/10.999 = 1 "0 to 10")
6. In the Optional boxes continue to specify the ranges and value to be assigned, e.g.
   (11.00/19.999 = 2 "11 to 19")
   (20.00/60.999 = 3 "20 to 60")
   (61.00/max = 4 "61 and greater")
   Note: examples on how to specify the value can be see if you click on drop down arrow for each of the Optional boxes.
7. Click on the "**Options**" tab. Click on the radio button next to "**Generate new variables**".
8. In the box, type the name of the new variable: **age_gp1**
9. We can also specify a name for the value labels. Click on the check box next to "**Specify a name for the value label defined by the transformation rules**".
10. In the box, type "**age_label**".
11. Click on **Ok**

Let's add a variable label to the new variable: The Stata command is:

```
label variable age_gp1 "Age group - second method"
```

Now, compare the **age_gp** variable with the **age_gp1** variable. Use a cross tabulation.

```
tab age_gp age_gp1
```

Variation on the second method

The **recode** function

The same results can be achieved by using one command the recode() function in conjunction with the Generate command. The recode() function takes three or more arguments. The first argument is the variable name that you want to categorize. The rest of the arguments are used to determine how to code the new variable.

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable**
3. Click on the reset button in the lower left hand corner of the dialog box - R
4. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **agecat**
5. Click on the **New variable(s) type** drop down box and change to **byte**.
6. For the **Contents** box, click on the **Create** button.
7. In the **Expression builder** box, under **Categories**, select **Programming**
8. A list of available functions are displayed. Scroll down to the **recode()** function and highlight that function. You will see a description of the function at the bottom of the dialog box.
9. Double click on the this function. The function will be pasted in the window at the top of the dialog box so that you see:
   recode(**x**,x1,x2,...,xn)
   with the first “x” highlighted. Replace the first “x” with the variable name, **ca3**, so that the expression now looks like:
   recode(**ca3**,x1,x2,...,xn)
   Replace the “x1" with the value of the highest age that you want to recoded for the first group, e.g.
   recode(ca3,**10**,x2,...,xn)
   Continue replacing the values with the next group to be coded until all groups are defined, e.g..
   recode(ca3,10,**19,60,100**)
   Stata will use the value as the code assigned to all cases that fall within that group. The value of 10 will be assigned to all observations with ages between 0 and 10, the value of 19 will be assigned to all observations that fall between ca3 >10 and <=19, and so on.
10. Click on **Ok** to close the dialog box.
11. Click on **Ok** to run the command.

The Stata command is:

```
generate byte agecat= recode(ca3,10,19, 60,100)
```

Run a **tabulate** on the new variable - **agecat** - and compare the number of cases in each category between the new variable and the age_gp variable.

These new variables are not yet part of the data file stored on disk. We must save the file in order for these variables to be included permanently in a new data file. It is a good practice to save a file under a different name in case we want to go back to a previous version of a file. For this reason we will use the **Save As** command from the **File** menu with the new file name **Q1A-AGE.DTA**.

1. From the **File** menu select **Save As...**
   The cursor should be in the box under File name: *above* the Save as type: *Stata data (*.DTA) drop-down box. Since *.dta in the File name: area is blocked, you can immediately start typing the new file name.*
2. Type **q1a-age** (The .DTA extension will be added automatically.)
3. Click on **Save** to run the command.

The Stata command is:

```
save "C:\Docs\sample\q1a-age.dta", replace
```

Now each time the data file Q1A-AGE.DTA is opened, the **age_gp** variable as well as the other two **age_gp1** and **agecat**, will be included.

You might want to use this new categorical variable to determine how many people in each age group are heads of households, spouses, or children.

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   Two-way tables with measures of association
   The* tabulate2 -Ttwo-way tables *dialog box opens.
2. **Use age_gp for Row variable and ca2** (relation to head) for Column variable.
3. Check the proper selections in the Cell content choices, for we want both Row and Column percentages.

4. Click on **Ok** to run the command.

The Stata command is:

```
tabulate age_gp ca2, column row
```

From the table you can see that 11.95% of heads of households are 61 years of age or older. Also, of the people 61 years or older, 83.67% are heads of households.

Compare the information you get from this **Two-way table** analysis with the information from the **Generate** command performed on **ca3** (age) earlier.

Apply what you have learned about data transformations and descriptive statistics in the following exercise.

Exercise 1.2

Using the Household Data and Questionnaire (latter available in the annex), find out the number of households in each district that have 1-4, 5-7, and more than 7 persons per household. One way to find out this information is to create the following table.

Hints:

a. Use the file c-hh.dta.
b. Recode **h1** into **hhsize** using the following groups: (1 thru 4) (5 thru 7) (8 thru Highest).
c. Add a variable label and value labels.
d. Run a **two-way table - Tabulate** on this variable by **district**

Looking at the results, you can see 34,76% of all 1 to 4 member households are found within Monapo and that 60,75% of all households in Monapo have 1 to 4 members in a household.

```
+-------------------+
| Key               |
|-------------------|
|     frequency     |
|  row percentage   |
| column percentage |
+-------------------+

   Household |            district
        size |    monapo     ribaue    angoche |     Total
-------------+---------------------------------+----------
 1-4 members |        65         48         74 |       187
             |     34.76      25.67      39.57 |    100.00
             |     60.75      40.34      64.35 |     54.84
-------------+---------------------------------+----------
 5-7 members |        39         56         36 |       131
             |     29.77      42.75      27.48 |    100.00
             |     36.45      47.06      31.30 |     38.42
-------------+---------------------------------+----------
8-12 members |         3         15          5 |        23
             |     13.04      65.22      21.74 |    100.00
             |      2.80      12.61       4.35 |      6.74
-------------+---------------------------------+----------
       Total |       107        119        115 |       341
             |     31.38      34.90      33.72 |    100.00
             |    100.00     100.00     100.00 |    100.00
```

Before exiting Stata for Windows we should save the do-file. The file contains all of the commands. It is useful to keep this file so you can rerun the commands if you want review the commands and the output that is produced.

1. Make the Do-file Editor the active window using its icon in the Windows taskbar.
2 From the **File** menu select **Save As...**
3 Enter the filename **session1**
   The .do extension will be added to the name automatically.
4. Click on **Save**

To exit Stata for Windows:

1. From the **File** menu select **Exit**
   A dialog box will open to say that "Data have been changed without being saved. Do you really want to exit?"
2. Click on Yes
   We do not need to save the newly created categorical variable. We will not be using it again.

STATA 8 for Windows SAMPLE SESSION

SECTION 2 - Restructuring Data Files - Table Lookup & Aggregation

Restructuring Data Files

For some types of analysis the data files may need to be restructured to a different level. The data from the four questionnaires—household, member, production and sales—are in four separate data files because the data are at different levels. The household data is at the most general, or highest, level - one case per household. The other three files contain more detailed data, which is usually thought of as being at a lower level - there are multiple cases per household. If you are not familiar with the concept of levels of data, read "Computer Analysis of Survey Data -- File Organization for Multi-Level Data" by Chris Wolf, before continuing on with this section. This paper is available at http://www.aec.msu.edu/fs2/survey/index.htm

The analysis we did in Section 1 was done at each level separately, using just the variables in a single file at a time. However, other types of analysis require combining data from more than one file. Let's look at an example.

Suppose we want to create a table of calories per adult equivalent produced per day from the principal food crops. Furthermore, we want to see how this varies by district and calorie-production quartile.

TABLE:1 Food Production in calories per adult equivalent per day

	Calorie Production Quartile			
Districts	1	2	3	4
Monapo				
Ribaue				
Angoche				

The data in their current form cannot answer the question; therefore, many transformations are required to produce this table. This is a typical example of the complications you will encounter in real-world data analysis. This entire section will be devoted toward the goal of creating this table.

To begin, let's first take a look back at some of the files that we have and at the variables we need to use from each of these:

- c-q1a.dta: This file contains data on household member characteristics. It is at the household-member level. We need to use the variables **ca3** (age) and **ca4** (sex) in this exercise to compute the number of adult equivalents per household.
- c-q4.dta: This file contains data on crops produced by the household. The variables we need to calculate the total production of the household are:

    a. **prod** - contains the codes for the agricultural crop produced.
    b. **p1a** - contains the codes for the unit in which the production was measured (100 kg sack, 50 kg sack, etc).
    c. **p1b** - contains the number of units produced for the year.

Note that the unit of production is not a standard unit for each crop. For example, a "100 kg sack", as the term is used in Mozambique, weighs 100 kg only when the sack is filled with maize. When it is filled with manioc root, it weighs much less than 100 kg. Thus, we need *conversion factors* to be able to convert each of the units in which production was actually measured to our standard unit, which is the kilogram.

- conver.dta: This is a *table-lookup file*. This file was created specifically to handle the problem of converting non-standard units to a standard unit. For each product-unit combination there is a conversion factor to convert the measurement to equal the weight in kilograms. In other words, there is a different conversion factor for each product-unit combination. For example, the conversion factor for a 50 kg sack of rough rice is 39.44; for a 50 kg sack of cotton it is 17.5, while a 50 kg sack of peanuts is 41.67. The variables in this file are:
    a. **prod** - product (crop) code
    b. **unit** - unit of measure
    c. **conver** - conversion factor (equal to the number of actual kilograms for the combination of **prod** and **unit**)

Below, a sample of data from CONVER.DTA shows that:

rice (**prod**=7) measured in a 20 liter can (**unit**=8) weighs 19 kg;
rice (**prod**=7) measured in a 50 kg bag (**unit**=24) weighs 53 kg;
beans (**prod**=30) measured in a 20 liter can weighs 17 kg;
beans (**prod**=30) measured in a 50 kg bag weighs 47 kg.

prod	**unit**	**conver**
(Product)	(unit)	(conversion factor)
...	...	...
7	8	19
7	24	53
...	...	...
30	8	17
30	24	47
...	...	...

- calories.dta: This also is a *table-lookup file*, created for convert kilograms of food into calories of food. It contains two variables:

  a. **prod** - the product (crop)
  b. **calories** - number of calories per kilogram of each of the crops

To arrive at the final table, we need to combine the data from different files. There are different methods to use to combine files, depending on what is desired. In Stata, we can

1. **Append** datasets. Appending data sets means that the data in different files have the same variables and the desire is to add one data set of observations to another data set (or append one file to the end of another file). An example would be that you entered data for harvest in a file for one district, entered data for harvest into another file for another district. In the end you want the data for all districts in the same file. To do that, you would use the append command.

2. **Merge** datasets. Merge combines datasets horizontally matching corresponding observations. An example is a survey asking questions about the household in Part 1 and another set of questions about the household in Part 3. Each Part of the survey is entered into a different data file. To combine Part 1 and Part 3 (both at the household level), use the merge command.

3. **Joinby** datasets. This type of merge combines datasets horizontally matching all pairwise combinations possible. An example is a set of data on parents and a set of data on children. Joinby would match the parents to every observation of the children within that family. The key word "unmatched" is used and within parentheses the type of join is specified. There are four types of joins with merge:

none - all unmatched observations are ignored (this is the default), i.e. if there is not a matching observation in both files, the observation is dropped from the final dataset.

both - unmatched observations from the "master" (or file that is in memory) and "using" (file that is not in memory) data

master - unmatched observations from the "master" data are included

using - unmatched observations from the "using" data are included

4. **Cross** datasets. In this type of merge, the first observation in the first file is joined horizontally with every observation in the second data set. The second observation in the first file is then joined with every observation in the second data set and so on. This type of file combination is rarely used.

In this tutorial we will use the "**merge**" and the "**joinby**" commands.

With this information in hand, we can now think about the specific steps we must take to create the table we want. Logically, there are three steps:

1. We need to know how many calories each household produced for the year. We can generate a file with this information using data we have stored in three files—the production file, c-q4.dta, and two table-lookup files, conver.dta and calories.dta.

2. We need to know how many adult equivalents are in each household. We can generate a file with this information using data from the member file, c-q1a.dta.

3. We need to combine the results from steps 1 and 2 into one file so we can compute calories produced per adult equivalent per day.

Step 1: Generate a household level file containing the number of calories produced per household.

In executing this step, we must keep three things firmly in mind.

First, all production is currently measured in non-standard units. Each unit can have a different weight for each of the products. Thus, we must first convert all production into kilograms.

Second, we want to know many calories are produced by each household, not kilograms. Thus, after converting all production to kilograms, we must convert kilograms to calories.

Third, an examination of the file shows that we have data for each product produced by the household. But we want to know the total calories of staplel food that was produced by the household, not the total calories from each separate product. After we convert all production to calories, we can select the staple foods and then sum the calories within each household to arrive at the household total.

Let's begin by opening c-q4.dta, the production file.

1. Select **File**/**Open...**
2. Select the file name c-q4.dta
3. Click on **Ok** to run the command.
4. Open up the Do-File Editor (<Ctrl+8> or click on the button on the tool bar) and copy the command into a new do-file.
5. Type in comments at the beginning the do-file, e.g. purpose of the do-file, your name and the date.
6. Save the do-file to the name session2.do

Convert production to kilograms

First we must convert all production of the crops into kilograms. To find the conversion factor appropriate for each case in the production file (c-q4.dta), we need to look up the product and unit in the conver.dta file. We will create a new file where each observation has both the data from the production file and a variable containing the conversion factor for that product-unit combination. In Stata for Windows we want to use the “joinby” command, which can be found through the menus with the following choice:

Data
Combine datasets
Form all pairwise combinations within groups.

The input files for a merge must be sorted by the key variable(s) (those variables you are using to match the cases). Since we have a unique conversion factor for each product-unit combination, both our product variable and our unit variable are the key variables. The CONVER.DTA file is already sorted by **prod** and **unit**. We must sort the currently working production file the same way, while taking account of the fact that the unit variable is named **p1a** and not **unit**. To sort the cases:

sort data by key variables

1. From the **Data** menu select **Sort** then **Sort Data**
   The Sort - Sort data *dialog box will come up.*
2. In the **Variables:** box select **prod** and **p1a** and click on **Ok**.
3. Copy the command and paste it into the do-file.

The Stata command is

```
sort prod p1a
```

Let's look at the two variables using the tab1 command. If we wanted to know whether data were missing, we would need to add a subcocmmand to the tab1 command to show how many cases had missing values. We can type in the Command window

```
tab1 prod p1a, missing
```

There are 1,693 cases. We have many products. For the tabulation of p1a we see 2 values that have no labels (0 and 1) and note that there are 23 cases that do not have a value in p1a. If we were still cleaning this file, we would see that there are problems and would need to research the errors and missing data. If it were possible, we would want to make corrections before proceeding with the analysis.

Rename any key variables in both files to the same name

We cannot merge the two files unless the variables that we want to merge by have the same names. We will rename p1a to unit.

1. From the **Data** menu select **Variable utilities** then **Rename variable**

The rename - Rename variables *dialog box will come up.*

2. The first choice “rename variables” has been selected. In the **Existing Variables** box select **p1a**. In the **New variable name** box type **unit**.
3. Click on **Ok**.
4. Copy the command and paste it into the do-file.

The Stata command is

```
rename p1a unit
```

The files are now ready to be merged. We are doing a “File - Table” merge where the second file is our “Lookup Table”. We want to keep all records in the “master” file (or the file in memory) and keep only those records in the “using” file that match. The file created will become the working data file, replacing the current one.

The **joinby** command

1. From the **Data** menu select **Combine datasets**, then select **Form all pairwise combinations within groups**

   The joinby - Form all pairwise combinations within groups *dialog box will open.*
2. For the **Filename of dataset on disk**: click on **Browse** Select the filename conver.dta and click on **Open**. The name C:\docs\sample\conver.dta appears in the box.
3. In the box below **Join observations by groups formed from specified variables**: select **prod unit**
4. Click on the “Options” tab.
5. Under “Unmatched Observations”, select

   Include from current dataset

   This option keeps all cases in the original data set (c-q4a.dta) even if there is no match in the lookup data set (conver.dta).
6. Click on **Ok**.
7. Copy the command and paste it into the do-file.

The Stata command is

```
joinby prod unit using "C:\docs\sample\conver.dta",
unmatched ( master )
```

The above command tells Stata for Windows to merge the working data file or “master” (the file in memory) and the conver.dta file or “using” data file, (using conver.dta as a table lookup) to add the “conver” variable that is in the conver.dta to our working data file. We renamed **p1a** to **unit**.

Key variables (or variables to match by) are required in any procedure to merge two files when one of the files is being used as a keyed table. Our key variables specify doing the lookup by product and unit (the grouping variables), because we have a different conversion factor for each product-unit combination. If we had used only **prod**, Stata would expect each product to have only a single conversion factor, with the same value regardless of the unit of measurement used. For example, it would expect the same conversion factor for rice whether it was in a 100 kg bag or a 20 liter can. This would be incorrect.

The new working file produced by the join now contains the needed conversion factor variable, **conver**. For every product-unit combination, **conver** is equal to the number of kilograms in that unit.

Check the resulting data file

It is always important to verify that the join worked as you expected it to work. Click on the Data Browser button to look at some cases to verify that the conversion factors match the products. We could also use the list command to see if a 20-liter can filled with maize grain has a conversion value of 18 kilograms (prod = 47 unit = 8).

The Stata command is:

```
list prod unit conver if prod==47 & unit ==8
```

Note: ***Two equal signs (==) are required.***

The two equal signs distinguish relational equality from the =*exp* assignment phrase. For example, if you want to create a variable where you will be assigning values to that variable, you will use an expression (*exp*) and need only 1 equal sign (example: gen newvar = oldvar*2.5). In the above example, prod already has values and we want to see only records where **prod** has a specific value. Therefore, it is a relational equality and we must use 2 equal signs (in words, we are saying show only records where prod has a value of 47 and unit has a value of 8).

STATA creates a variable called _merge. This variable contains values to show how the merge worked. We should run a tabulate on the _merge variable to look at how the merge was done:

```
tab1 _merge
```

From the output you can see there are same number of records as before the merge, i.e. 1693.

```
-> tabulation of _merge
                      _merge |  Freq.  Percent        Cum.
-----------------------------+-----------------------------
          only in master data |     27     1.59        1.59
both in master and using data |  1,666    98.41      100.00
-----------------------------+-----------------------------
                        Total |  1,693   100.00
```

Note that there are 27 cases where there was no match for the prod-unit combination in the look-up file. You would want to investigate further to see if the records without a look-up value are crops that you want to have included in the analysis you are doing, and, if they are, correct the lookup file and run your procedure again.

Compute total kilograms produced

We can now calculate total kilograms by multiplying the number of units (**p1b**) by this conversion factor.

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable**
   The generate - Generate a new variable dialog box opens.
3. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **qprod_tt**
. For the **Contents** box, type in **p1b * conver**
5. Click on **Ok**
6. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The Stata command is:

```
generate float qprod_tt= p1b * conver
```

Note that 49 cases do not have a value of qprod_tt. Why are there more cases with no value in this variable, if only 27 cases had no lookup conversion value?

Convert total kilograms produced to calories

For the next part of Step 1, we need to look up the number of calories in a kilogram for each product. This information is in the table-lookup file calories.dta. This file has two variables — product and number of calories per kilogram. The key variable is product (**prod**). In order to add the calorie-conversion variable to the working data file we need to do another merge with keyed table lookup (**joinby**). This time the key variable only needs to be the product variable. The data file has already been sorted by product (see the previous merge), so we don't need to sort it again. Stata will reuse the _merge variable again with the next join, so we should drop this variable first since we no longer need it. The Stata command is:

```
drop _merge
```

Now we are ready for the next join:

The **joinby** command

1. From the **Data** menu select **Combine datasets**, then select **Form all pairwise combinations within groups**
   The joinby - Form all pairwise combinations within groups *dialog box opens.*
2. Select **Groups formed by variables**. In the box below select **prod** .
3. For the **Dataset filename**: click on **Browse** Select the filename calories.dta and click on **Open** The name C:\docs\sample\calories.dta appears in the **Dataset filename** box.
4. Click on the “Options” tab.
5. Under “**Unmatched Observations**”, select **Include from current dataset**
   This option will keep all cases in the data set in memory that do not have a match in the lookup data set (calories.dta).
6. Click on **Ok**.
7. Copy the command and paste it into the do-file and add a comment.

The Stata command is:

```
joinby prod using "C:\docs\sample\calories.dta",
unmatched( master )
```

The new working data file produced by the merge now contains the needed calorie variable, **calories**, but check to make sure. Maize grain (PROD=47) should have 3590 calories per kilogram in the **calories** variable. We can browse the data and or we can use the list command again.

Check the data file

The Stata command is:

```
list prod calories if prod==47
```

There are many cases; we can break the list command after looking at the first screen of data - click on “Break” icon (stop sign with x).

Also check the _merge variable to see how the merge was done:

```
tab1 _merge
```

There are 87 cases with no match in the calories.dta file. Verify that those cases are not staple foods.

Calculate the total calories produced

We can now compute total calories produced.

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable**
   The generate - Generate a new variableForm all pairwise combinations within groups *dialog box opens.*
3. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **cprod_tt**
4. For the **Contents** box, type in **qprod_tt * calories**
5. Click on **Ok**
6. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The Stata command is:

```
generate float cprod_tt= qprod_tt * calories
```

Note that 131 missing values were generated. Check to see why.

Assign variable labels

The two new variables do not yet have variable labels. To assign a variable label:

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the first variable: **qprod_tt**
3. In the **Attach label to variable** (up to 80 characters) box, type
   Total production in kgs
4. Click on the **Submit** button.
   Clicking on the “submit” button leaves the dialog box open so we can then define the label for the cprod_tt variable without having to select it again from the menus.

5. In the **Variables**: box, select the name of the second variable: **cprod_tt**
6. In the **Attach label to variable** (up to 80 characters) box, type
   Total calories produced
7. Click on the **Ok** button.
8. Copy the commands from the Results window and paste them into the do-file and add a comment to explain what you have done.

The Stata commands are:

```
label variable qprod_tt "Total production in kgs"
label variable cprod_tt "Total calories produced"
```

Select only staple food products

This gives us a new working data file with total calories produced per product for each household. The final output table asks only for information about the staple food crops:

peanuts	(prod=5))
rice	(prod=6)
nhemba bean	(prod=30)
manteiga bean	(prod=31)
manioc	(prod=41)
sorghum	(prod=44)
maize	(prod=47)

We can find the product code by looking at **prod** in the questionnaire. Since we are only interested in those products, we need to exclude the rest of the records about other crops. Stata uses the "keep" command. Once you run this command you will no longer have the complete data set available. You must remember that you should never save a file to the same name after you have selected out a set of data. You will overwrite the original data and no longer have the complete set. To select just a subset of records:

The **keep if** command

1. Click on **Data**, then **Variable Utilities**, then **Eliminate variables or observations**.
   You should see the Drop and keep - eliminate variables/observations *dialog box.*
2. Under the **Main** tab select the round button next to Keep
3. Under the **by/if/in** tab, in the **Restrict to observations if** box type

```
prod == 5 | prod == 6 | prod == 30 |
prod == 31 | prod == 41 | prod ==44 |
prod == 47
```

The "|" is a symbol for the word OR. We are telling Stata to select all cases with **prod** *equal to 47 <u>or</u>* **prod** *equal 30 <u>or</u>* **prod** *equal 31...*

4. Click on **Ok**
5. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The Stata command is:

```
keep if prod==5 | prod==6 | prod==30 |
prod==31 | prod==41 | prod==44 | prod==47
```

Only cases with these product codes will now be used for analysis. Note that 454 observations were dropped. You can use the **tabulate** command to verify that you now have only 7 crops in the file. In the Command window you can easily type

```
tab prod
```

There are a total of 1239 cases. Now we need to know how many calories were produced <u>per household</u> for all staple food products combined. To do this, we need to sum, for each household, the values of **cprod_tt** for all of the food crops the household produced. In other words, we need to create a new household-level file from the current household-product level file where there is only one case per household. STATA uses the command "collapse" to aggregate the number of cases at one level to a new level. We will sum all the cases for household-product to create just one case for household.

Create a new file which is a household level file rather than a household-product level file

The **collapse** command

To create the new household-level file, we use **collapse**. It always uses the working data file as the file to be collapsed.

1. From the **Data** menu select **Create or change variables** then select **Other variable transformation commands** then select **Make dataset of means, medians, etc.**
   The collapse - Make dataset of means, medians, etc. *dialog box will appear.*
2. Click on the **Options** tab and in the **Grouping variables** box, select **district vil hh** in that order because those variables represent the identification of an individual household.
   The Grouping variable(s) *is used to specify the variables to be used for combining cases in*

*the collapsed file. Any cases from the original file that have identical values for all of the grouping variables will be combined into a single case in the collapsed file. We want the collapsed file to have one case per household, so we use the variables that identify a household in our survey–***district***,* **vil***, and* **hh***.*

3. Click on the **Main** tab and in the **Collapse list** box type **(sum) cprod_tt** . You can see other examples of how you can specific what you want in the Examples box below.
4. Click on **Ok**
5. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The Stata command is

```
collapse (sum) cprod_tt, by(district vil hh)
```

Look at the resulting file (click on the **Data Browser** tool). You should see four variables with only one case per household. The collapse command created a new variable **cprod_tt**, which we calculated by summing **cprod_tt**, total calories produced, across all cases (all the different food crops) for each household. The only variables which are contained in a collapsed file are the grouping variables and any new collapsed computed variables created (e.g. **cprod_tt**). Remember to close the browser before you continue.

You can look at the variable definitions using the **describe** command. The computed variable **cprod_tt** does not have a very descriptive label any more so we need to change the label to reflect what the variable is.

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the first variable: **cprod_tt**
3. In the **Attach label to variable** (up to 80 characters) box, type
   Calories produced in staple foods
4. Click on the **Ok** button.
5. Copy the command and paste it into the do-file.
6. Run the **describe** command again.

The Stata commands are:

```
describe
label variable cprod_tt "Calories produced in staple foods"
describe
```

The new working data file now contains what we need, total number of calories from staple foods produced per household. We can also look at this variable by doing a descriptives. Use the "summarize" command to run a mean on the new variable **cprod_tt**. You should find that the average number of calories produced per household per year is 4,483,965.

Save this data file using the **Save As...** command.

1. Use **Save As...** from the **File** menu
2. Name the file **hh-file1**
3. Click on **Ok**
4. Copy the command and paste it into the do-file and add a comment to explain what you have done.

Remember to save your do-file regularly. You must be in the Do-file editor to save the do-file.

Step 2: Generate a household level file containing the number of adult equivalents per household.

The data needed to calculate adult equivalents per household is in the member file, C-Q1A.DTA.

1. Click on the "Open Folder" button on the STATA Taskbar
2. Select the file name c-q1a.dta and open the file.
3. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The rules we will use for calculating adult equivalents for this survey are:

Males, 10 years and older	= 1.0
Females, 10 to 19 years old	= 0.84
Females, 20 years and older	= 0.72
Children, under 10 years old	= 0.60

The adult equivalent value says that, on average, a female 10 to 19 years old needs only 84% as many calories as a male 10 years or older, and that children under 10 need only 60% as many calories as the typical male 10 years and older. Thus, for example, a child (male or female) under age 10 is counted as .60 adult equivalents. For each person (observation) in the member file we need to look at the

Create a variable with the adult equivalent for each person

variables sex, **ca4**, and age, **ca3**, to calculate adult equivalents.

The **generate…. if** command

The **Generate.../If...** command allows us to do this. We will name the adult equivalent variable that we will create to be **ae**.

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable**
   The generate - Generate a new variable *dialog box opens.*
3. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **ae**
4. For the **Contents** box, type the value of **1**
5. Click on the **if/in** tab.
6. Type the statement **ca4 == 1 & ca3 >= 10**
7. Click on **OK**

Now that the new variable has been created, another command is used to assign the codes for the other adult equivalent groups, the **replace** command.

8. Select **Create or change variables** from the **Data** menu
9. Select **Change contents of variable**.
   The replace-Replace contents of variables *dialog box opens.*
10. In the **Variables** box select the name of the variable that was just created: **ae**
11. Type .84 in the **Contents** box
12. Click on the **If/In** tab.
13. In the **Restrict to observations if** box, type in

ca4==2 & (ca3 >=10 & ca3 <=19)

14. Click on **Submit**. The dialog box remains open and the command is run.
15. In the **Restrict to observations if** box, change the criteria to: **ca4 == 2 & ca3 >=20**
16. Click on the **Main** tab.
17. Type .72 in the **Contents** box .
18. Click on **Submit**. The dialog box remains open

and the command is run.

19. Type .6 in the **Contents** box
20. Click on the **If/In** tab.
21. In the **Restrict to observations if** box, change the criteria to: **ca3 <10**
22. Click on **Ok**

Numeric Expression	If... Statement
0.84	**ca4 == 2 & (ca3 >= 10 & ca3 <= 19)**
0.72	**ca4 == 2 & ca3 >= 20**
0.6	**ca3 < 10**

23. Copy the 4 commands from the Results window and paste them into the do-file and add a comment to explain what you have done.

The new variable does not yet have a variable label. To assign a variable label:

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the variable name: **ae**
3. In the **Attach label to variable** (up to 80 characters) box, type
   Adult equivalents
4. Click on the **Ok** button.
5. Copy the commands from the Results window and paste them into the do-file and add a comment to explain what you have done.

The Stata commands are:

```
generate byte ae= 1 if ca4 == 1 & ca3 >=10
replace ae = .84 if ca4==2 & ca3 >=10 & ca3 <=19
replace ae = .72 if ca4==2 & ca3 >=20
replace ae = .6 if ca3 < 10
label variable ae "Adult equivalents"
```

To verify that the new adult equivalent variable, **ae**, has been calculated, display a frequency table for it.

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   One-way tables
   The tabulate1 - One-way Tables *dialog box opens.*

2. Select the variable name
   ae
   in the **Categorical Variables** box which is found under the tab labeled **Main**.
3. Check the box ✓ next to **Treat missing values like other values**
4. Click on the **Ok** button.
5. Copy the command and paste them into the do-file and add a comment to explain what you have done.

The Stata command is:

```
tabulate ae, missing
```

You should see there are 1524 total cases. Ideally there should be four values represented in the table —1, .72, .84, and .60— and no missing cases. You can see we have nine missing cases. This tells us that our data file is missing either the age or the sex for nine people. This problem should have been identified during the cleaning process. At this point it would be ideal for the researcher to go back to the original questionnaires to determine the reason why these data are missing. Since we can't do this, we will use an alternative method.

Replace “missing values” with a mean value

If we leave these values missing, the total adult equivalents of those households will appear to be slightly smaller than they actually are, which will distort the results. We could avoid this problem by eliminating the households of those nine individuals from our analysis, but then we can't use the information about the food production from those households. Instead, we will try to make a reasonable assumption about those nine missing members. We know that the adult-equivalent values range from a low of .6 for children to a high of 1.0 for adult males, which is not a very wide range. We can determine the mean adult-equivalent value for the whole sample and use that value to fill in the missing data. To find out the average adult-equivalent value for our sample...

1. **Statistics/Summaries, tables and tests/Summary Statistics/ Summary Statistics**
2. Variable is **ae**
3. Don’t forget to copy the command into the do-file editor.

The Stata command is:

```
summarize ae
```

We can see that the mean value of **ae** for all individuals is .79, with a standard deviation of only .17. We will assume that the nine individuals with missing age or sex codes are all "average" individuals, and assign them the adult-equivalent value of .79. (Warning: be very cautious about "filling in" missing data this way. Careless use of this technique can give you misleading results. We are using this example to illustrate the use of Stata commands and not recommending that you do this routinely to compensate for missing data.)

We will use the **Replace** command to change the system missing values (.) in the **ae** variable to .79.

1. **Data**/**Create or change variables**/**Change contents of variable**
   The replace - Replace into same variable *dialog box will appear.*
2. Under the **Main** tab, select **ae** in the Variable: box
3. In the Contents box type .79
4. Under the **if/in** tab in the **Restrict to observations if:** box type
   ae==.
   The "period" represents system missing.
5. Click on **Ok**
6. Don't forget to copy the command into the do-file editor.
7. Check the results of your **replace** command by rerunning the **tabulate** command.

You should see 9 cases in the frequency with a value of .79.

The Stata command is:

```
replace ae = .79 if ae==.
tabulate ae, missing
```

Calculate the adult equivalents for the household

Now we need to calculate the number of adult equivalents for each household. The current file is at the member level, but we need values for the household . Again we use **Collapse** to go from the member level to the household

The **collapse** command

level. The new variable **ae** will be calculated by summing **ae** across all members of a household.

<u>Reminder</u>: The Grouping variable(s) specify the variables to be used for combining cases in the collapsed file. Any cases from the original file that have identical values for all of the grouping variables will be combined into a single case in the collapsed file. We want the collapsed file to have one case per household, so we use the variables that identify a household in our survey—**district**, **vil**, and **hh**.

1. From the **Data** menu select **Create or change variables** then select **Other variable transformation commands** then select **Make dataset of means, medians, etc.**
   The collapse - Make dataset of means, medians, etc. *dialog box will appear.*
2. Click on the **Options** tab and in the **Grouping variables** box, select **district vil hh** in that order because those variables represent the identification of an individual household.
3. Click on the **Main** tab and in the **Collapse list** box type **(sum) ae** . You can see other examples of how you can specific what you want in the Examples box below.
4. Click on **Ok**
5. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The Stata command is

```
collapse (sum) ae, by(district vil hh)
```

Collapse creates a new working file. The new working data file is at the household level, with one case per household. The variable **ae** is the total adult equivalents for that household. Look at the resulting file (click on the Data Browser tool). You should see four variables with only one case per household. You can also look at the variable definitions using the **describe** command. The computed variable **ae** does not have a very descriptive label any more so we need to change the label to reflect what the variable is.

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the first variable: **ae**

3. In the **Attach label to variable** (up to 80 characters) box, type
   Adult equivalents per household
4. Click on the **Ok** button.
5. Copy the command and paste it into the do-file.
6. Run the **describe** command again.

To verify that this variable was created, **summarize** the variable **ae**.

1. **Statistics/Summaries, tables and tests/Summary Statistics/ Summary Statistics**
2. Variable is **ae**
3. Don't forget to copy the command into the do-file editor.

You should find that the average adult equivalent over all households is 3.49.

The Stata commands are:

```
label variable ae "Adult equivalents per household"
summarize ae
```

This completes step 2. Save this file as HH-FILE2.DTA.

1. Click on **File**/**Save As...**
2. Filename is hh-file2
3. Click on the **Ok** button.
4. Copy the command to the do-file.

The Stata command is:

```
save "C:\docs\sample\hh-file2.dta", replace
```

Step 3:

We need to merge the two files created in steps 1 & 2 together to compute calories produced per adult equivalent.

The **merge** command

We have created two files: hh-file1.dta, which contains the calorie-production data for all households, and hh-file2.dta, which contains the adult-equivalent data for all households. We need to combine these files case-by-case matching by district, village and household, to get both sets of data in a single file. To do this, we use **Combine datasets / Merge datasets** under the **Data** menu choice

We noted earlier that key variables are required for any merge. When you're joining two files at the same level, as we're about to do, it may not seem important to include key variables, but it is. The key variables determine which observations are to be combined.

Note: *You should never use* ***Combine datasets*** *without Key Variables because without them you have no guarantee that Stata will combine the cases correctly.*

The command will execute without any warnings or error messages, but the results may be incorrect.

At this point, if you have not closed STATA, hh-file2.dta is still the working file.

A very important point: Stata cannot merge two datasets unless they are both sorted in the order of the key variables. One way to check to see if Stata knows the file is sorted is to use the **Describe** command. In the Results window you can see at the end of the list of variables, the words "sorted by" and the list of variables that the file is sorted by. Because we created hh-file1.dta by collapsing a file, it is already sorted by district, vil and hh. hh-file2.dta was also created by collapsing a file so it is sorted by district, vil and hh. We are ready to merge the two files.

1. Select **Data** / **Combine datasets** / **Merge datasets**
2. Select the file on disk to merge. Click on the **Browse** button. Select the file hh-file1.dta and click on **Open**
   The Merge - Merge datasets *dialog box will appear.*
3. In the Variables to match merge (optional) box, select
   district vil hh
   These are the Key Variables

4.. Click on the **Options** tab. Under this tab, you see the box labeled **Specify new name of variable to mark result of merge** The default name is _merge. This variable received a code of 1 or 2 or 3 to describe what type of merge occurred. The code definition is:
   1 = observation is from file in memory
   2 = observation is from file on disk
   3 = observations are from both files
   It is very important to look at the values in this variable after you have run the merge.

5. Click on the **Ok** button.
6. Copy the command to the do-file.

The Stata command is:

```
merge district vil hh using "C:\docs\sample\hh-file1.dta"
```

Check the _merge variable to be sure the merge was done correctly.

Now that you have run the merge, run a tabulate on the _merge variable. You can abbreviate the name to "_m", e.g. **tabulate _m**. You should see only the value of "3" for 343 observations. That means that there was an observation for each "district - vil - hh" combination in each of the two files.

Calculate the total calories produced per adult equivalent per household for the year

Merge Files created a new working data file. The two variables you need to compute calories produced per adult equivalent are now in the working file. Total calories produced (**cprod_tt**) per household for the year divided by total adult equivalents per household (**ae**) divided by 365 days per year gives us calories produced per adult equivalent per day (**cprod_ae**).

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable**
   The generate - Generate a new variable dialog box opens.
3. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **cprod_ae**
   For the **Contents** box, type in **cprod_tt/ae/365**
5. Click on **Ok**
6. Copy the command and paste it into the do-file and add a comment to explain what you have done.

The new variable does not yet have a variable label. To assign a variable label:

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the first variable: **cprod_ae**
3. In the **Attach label to variable** (up to 80 characters) box, type
   Calories produced per adult equivalent per day
4. Click on the **Ok** button.
5. Copy the command and paste them into the do-file and add a comment to explain what you have done.

The Stata commands are:

```
generate float cprod_ae= cprod_tt/ae/365
label variable cprod_ae "Calories per adult equivalent per day"
```

Computing quartiles

Before we can produce the table we want, we have to create one more variable, denoting which calorie-production quartile each household falls into within each

The **xtile** command using **if**

district. The Stata command to use is called **xtile**. This command is not available through the menus. To look at the structure of the command, we can use the Help menu.

1. Click on **Help / Stata command**.
2. In the **Command:** box, type **xtile** and click on **Ok**.

Under the Description heading, the definition of xtile is that it is a the command that categorizes a variable into the specified quantiles and places the information into a new variable. Examples can be found under the Examples: heading. Since we want to divide the data into 4 quartiles within each district, we can use the "if" subcommand, e.g.,

```
xtile quart = cprod_ae if district = 1, nq(4)
```

where
quart is the new variable that is created
cprod_ae is the variable used to rank the data
district is the controlling variable
nq(4) is short for nquantiles(number) which specifies the number of quantiles to use.

You must type this command in the Command window and press <Enter>.

Using the "if" subcommand works if you do not have very many codes to control by. We have 3 districts so it would not be a problem to use the **if** subcommand. What if we had 20 districts? This method would be a bit cumbersome.

Another method that is a bit easier is to use a counter.

```
for z in num 1/3: xtile quartz = cprod_ae if
district==z, nq(4)
```

The **foreach** looping command
The **levels** command

Stata provides another looping command that we can use to compute the new ranking variable. The coding used is not available through the menus. The looping command can be found in the Programming manual and is called **foreach**. Stata added a new command in April, 2003, called **levels** . The values are stored in temporary variables called r(levels). That information can be stored in a local variable and the variable used to cycle through the values.

1. Type the following command in the Command window:

```
levels district
```

```
The results should display the values of
the districts, e.g. 1 2 3
```

2. Now let’s store that information in a local variable. To make a temporary local level, we include the word “local” which means the variable only exists with the do-file. We need this command to be placed in the do-file. Switch to the do-file editor and type

```
levels district, local(levels)
```

3. We can now create variables containing the rank of the household within each district. We must type these commands into the do-file because the command is multiple lines. You are already in the do-file editor . Type:

```
foreach z of local levels {
  xtile quart`z’ = cprod_ae if (district == `z’), nq(4)
}
```

‘z’ is a local macro name which is set to each value in the variable “levels”. The values we know are 1, 2, and 3. In the first loop of this programming command z is equal to 1, in the second loop z is equal to 2, etc.

quart`z’ refers to a variable name where the contents of z is appended to the name quart, e.g. quart1, quart2, quart3, etc.

district = `z’, means that for the first loop district is equal to 1, for the second loop district is equal to 2, etc.

Very important note: The macro name `z’ must be surrounded by a “left” single quote (found in the upper left hand corner or the keyboard to the left of the key with the number 1) and a “right” single quote (found on the key to the left of the <Enter> key). If you do not use the left single quote, you will see an error message that says in red:

```
' invalid name
```

Be sure that you end the first line with a left curly brace, e.g. { and that you place on a line by itself after all commands that you want to be included in the loop, a right curly brace, e.g. }

Since we have 3 districts, 3 new variables are created with names of quart1, quart2, quart3.

4. We want the information in just one variable so we will create another variable and fill it with the information from the variables created above. The next step you are familiar with. We create a new variable and fill it with system missing.

```
gen quart = .
```

5. We now replace the data in **quart** with the data in the temporary variables. Remember, we must rerun the levels command as well since the data are temporarily stored in memory. Type the following lines, block and run them.

```
/*replace values with information from temporary
variable */
levels district, local(levels)
foreach z of local levels {
   replace quart=quart`z' if district==`z'
}
```

This command cycles through the values for z and replaces the contents of quart with the contents of quart1 if district is equal to 1 in the first loop, then replaces the contents of quart with the contents of quart2 if district is equal to 2 in the second loop, then replaces the contents of quart with the contents of quart3 if district is equal to 3 for the final loop.

4. The next step is to delete the temporary variables. Type the following, block and run the commands:

```
/*delete temporary variables */
        levels district, local(levels)
        foreach z of local levels {
           drop quart`z'
        }
```

Examples of the use of the **foreach** command are:

Computing new variables:

More examples of the **foreach** command

```
foreach var of varlist inc1-inc12 {
  generate tax`var' = `var' * .10
}
```

Collapsing across variables:

compute the quarterly income variables **incqtr1-incqtr4** using the **foreach** command.

```
foreach qtr of numlist 1/4 {
  local m3 = `qtr'*3
  local m2 = (`qtr'*3)-1
  local m1 = (`qtr'*3)-2
  generate incqtr`qtr' = inc`m1' + inc`m2' + inc`m3'
}
```

Always check any new variables to see if the values are what you expect to see. We can use the **tabulate** command with 2 variables - district and quart - to check the variables

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   Two-way tables with measures of association
   The tabulate2 - Two-way tables *dialog box opens.*
2. In the **Row Variable** box select **quart**
3. In the **Column Variable** box select **district**
4. Click on the **Ok** button.
5. Copy the commands starting with **for z in...** and ending with the tabulate command from the Review window into the Do-File Editor and write a comment to explain what the commands are doing.

The number of cases in each cell should be very close to being the same, plus or minus a case or two, e.g.

```
           |              district
     quart |    monapo     ribaue    angoche |     Total
-----------+---------------------------------+----------
         1 |        28         30         29 |        87
         2 |        27         30         29 |        86
         3 |        27         30         29 |        86
         4 |        27         29         28 |        84
-----------+---------------------------------+----------
     Total |       109        119        115 |       343
```

The Stata commands are:

```
/* there are 3 districts so we want to loop 3 times */
for z in num 1/3: xtile quartz=cprod_ae if district == z, nq(4)

/*initialize variable */
gen quart=.

/*replace values with information from temporary variable */
for z in num 1/3: replace quart=quartz if district==z
for z in num 1/3: drop quartz

/* check results  - should see equal number of cases in each
category */
tabulate quart district
```

The new variable requires a label:

1. Click on **Data**, then **Labels**, then **Label variable**.
2. In the **Variables**: box, select the name of the first variable: **quart**
3. In the **Attach label to variable** (up to 80 characters) box, type
   Calorie production quartile
4. Click on the **Ok** button.
5. Copy the command and paste them into the do-file.

The Stata command is:

```
label variable quart "Calorie production quartile"
```

Display the final table

We can now display a table showing the average caloric production in quartiles for each of the districts.

1. From the menus click on
   Statistics
   Summaries, tables & tests
   Summary statistics
   Summary statistics
   The summarize - Summary statistics *dialog box opens.*
2. In the Variable(s): box select **cprod_ae**.
3. Click on the "**by/if/in**" tab.
4. Click in the box "**Repeat command for groups defined by**"
5. In the box below this option, select **district quart**
6. Click on **Ok** to run the command.
7. Copy the command from the Results window and paste it into the do-file and add a comment to explain what you have done.

The Stata command is:

```
bysort district quart: summarize cprod_ae
```

You should note that the mean for the 2nd quartile in Monapo is 2539.364. The output from the **summarize** command gives you the numbers necessary for the table. However the output is difficult to read. There is another command, **table**, which can also be used to produce the final table. We will discuss this command in Section 3.

Before you save the file, you should sort the file by the key variables and then save this file as hh-file3,dta.

1. Sort the file by the key variables. Type in the Command window:
   Sort district vil hh
2. We no longer need the variable _merge so it should be dropped. Type in the Command window:
   drop _merge
3. Click on **File**/**Save As...**
4. Filename is hh-file3
5. Click on **Ok**
6. Copy the commands and paste them into the do-file.

Remember to save the contents of the Do-file Editor to a permanent file so you can use it another time.

1. Make the Do-File Editor the active window
2. Click on **File**/**Save As...**
3. Use the filename session2
   The .do extension will be added automatically.

This file now contains all the commands that you pasted either from the Command window or from the Review window.

Note: *Whenever you do any substantial amount of work, you should always copy the commands to a do-file and save the file so that you have documentation on what analysis you have done and so you can repeat the analysis without building all the commands again.*

Document the do-file with comments

Documenting the do-file with comments can save you much time trying to remember what analysis you did and why.

So now let's see how you would retrieve the command file you just created. To exit STATA for Windows:

1. Click on **File**/**Exit**
   Stata will prompt you if you have not saved the data file and will give you an opportunity to return to the program to save the data file. If you do not want to save your data file, click on "yes" to exit.

Start STATA for Windows again. To open our do-file:

1. **Window** / **Do-file editor** or press **<CTRL 8>**
   The Do-file editor window will open.
2. Click on the yellow file folder tool and select the file session2.do
3. Click on **OK**

You can then re-execute these same commands or edit them as you wish.

Your SESSION2.DO should look similar to lines below, with the exception that documentation comments have been added to this example, using an "/*" at the beginning of each comment and ending each comment with a */

```
/*STATA do file */
/* Purpose:  Calculate food production in calories per adult equivalent per day */
/* M Beaver - August 2003 */

/* Stata recommends you include the version that the do file was written in */

/* version 8.2 */

/*turn on the log */
log using session2, replace
```

```
/*turn off "more" so the whole file will run" */
set more off

/* open production data file */
use "C:\docs\sample\c-q4.dta", clear

/* sort by variables to match by */
sort prod p1a
tab1 prod p1a
/* rename the p1a variable to unit to match the conver data file */
rename p1a unit
joinby prod unit using "C:\docs\sample\conver.dta", unmatched( master )
tab1 _merge
/* check to see if got what was expected using list command */
list prod unit conver if prod==47 & unit ==8

generate float qprod_tt= p1b * conver

/* merge in the lookup conversion value for calories and calculate total calories */
joinby prod using "C:\docs\sample\calories.dta", unmatched( master )

generate float cprod_tt= qprod_tt * calories

/* add variable labels */
label variable qprod_tt "Total production in kgs"
label variable cprod_tt "Total calories produced"

/* select only staple crops */
keep if prod == 5 | prod == 6 | prod == 30 | prod == 31 | prod == 41 | prod ==44 | prod == 47

/* check to see that there are only 7 crops listed */
tabulate prod

/* need to sum all calories produced by the household */
collapse (sum) cprod_tt, by(district vil hh)
label variable cprod_tt "Calories produced in staple foods"
describe

/* verify you have the right average calories produced over whole sample */
summarize cprod_tt

/*  save the file */
save "C:\docs\sample\hh-file1.dta", replace

/* calculating adult equivalents */
use "C:\docs\sample\c-q1a.dta", clear

generate byte ae= 1 if ca4 == 1 & ca3 >=10
replace ae = .84 if ca4==2 & ca3 >=10 & ca3 <=19
replace ae = .72 if ca4==2 & ca3 >=20
replace ae = .6 if ca3 < 10

label variable ae "Adult equivalents"

/* check the variable */
tabulate ae, missing

/* calculate mean to determine average ae across the whole population */
summarize ae

/* replace all system missing with the value of .79 */
replace ae = .79 if ae==.
tabulate ae, missing

/* need to sum the adult equivalents for each household */
collapse (sum) ae, by(district vil hh)
```

```
label variable ae "Adult equivalents per household"
summarize ae

* save file for later use */
save "C:\docs\sample\hh-file2.dta", replace

use "C:\docs\sample\hh-file2.dta", clear
/* now combine both the hh-filel with hh-file2 */
/* both files are already sorted by key variables */
merge district vil hh using "C:\docs\sample\hh-file1.dta"

/* calculate the calories per adult equivalent per day */
generate float cprod_ae= cprod_tt/ae/365
label variable cprod_ae "Calories per adult equivalent per day"

/* rank the new variable by district */
/*check for number of districts */
tabulate district

/* there are 3 districts so we want to loop 3 times */
for z in num 1/3: xtile quartz=cprod_ae if district == z, nq(4)

/*initialize variable */
gen quart=.
/*replace values with information from temporary variable */
for z in num 1/3: replace quart=quartz if district==z
for z in num 1/3: drop quartz

/* check results  - should see equal number of cases in each category */
tabulate quart district
label variable quart "Calorie production quartile"
/* produce the table */
bysort district quart: summarize cprod_ae
/* sort file by key variables and drop _merge */
sort district vil hh
drop _merge
save "C:\Docs\sample\hh-file3.dta", replace
log close
```

Exercise 2.1

Produce similar output using calories retained (production minus sales) instead of calories produced. It will show calories retained per adult equivalent per day from the total of the same six food crops. The output should be broken down by district and calorie production quartile.

Hints:

a. The procedure is very similar to the work that we just completed. Open a new do-file to save your commands for this exercise.

b. Sales come from c-q5.dta.

c. Check the file for the appropriate variable for the quantity of sold production. Note that the product codes are the same as for c-q4.dta. Also check for the variables by which to sort.

d. You can start from a blank file and build all the commands necessary to produce the calories retained, or you can copy the commands used to generate the table from section 2 and adjust the commands as necessary to calculate the calories retained. Changes must be made for file names and variables.

e. Computing the calories sold involves the same basic steps as computing the calories produced. (Step 1)

f. Merge this newly created file, (the file containing calories sold), with the file containing calories produced, hh-file3.dta. Check the _merge variable (tab _merge) and explain why you see more than one value.

g. Keep in mind that only 256 households sold products, but all 343 households produced and retained calories. If the “calories sold” variable is missing, it means the household did not sell food, so it should be recoded to zero.

h. Compute calories retained = calories produced - calories sold. The average calories retained per adult equivalent for the whole population should be 3044.233

I. Rank into quartiles.

j. Use the **Tabulate** command to show calories retained by **district** and **quartile**.

k. Save the data file to the name, hh-file4.dta.

l. Save the contents of the do-file editor to a new name reflecting the name of the exercise.

Below is an example of the output you should produce:

```
-> district = monapo, quarts = 1
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     cret_ae |        28    1171.574    420.7985   224.4898   1806.867

-> district = monapo, quarts = 2
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     cret_ae |        27    2239.088    199.4202    1888.33   2554.892

-> district = monapo, quarts = 3
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     cret_ae |        27    3343.003    461.9159   2685.971   4303.122

-> district = monapo, quarts = 4
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     cret_ae |        27    7619.101    3557.135   4359.737   20873.97
```

```
-> district = ribaue, quarts = 1
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        30    1251.391    358.8783   429.2929   1790.432

-> district = ribaue, quarts = 2
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        30    2171.697    205.3644   1835.298   2566.006

-> district = ribaue, quarts = 3
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        30    3165.192    330.2283   2578.604   3731.045

-> district = ribaue, quarts = 4
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        29     5828.97      1632.9   3825.879   9464.901

-> district = angoche, quarts = 1
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        29    929.4182    388.3228   207.9077   1395.962

-> district = angoche, quarts = 2
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        29    1718.789    166.1601   1447.059   1984.059

-> district = angoche, quarts = 3
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        29    2442.247    347.8035   1997.711   3063.996

-> district = angoche, quarts = 4
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------------
     cret_ae |        28     5022.29    2443.454   3134.742   12674.86
```

STATA 8 for Windows SAMPLE SESSION

SECTION 3 - Tables & Other Types of Analyses

Tables

The **table** command

Using the **Table** command you can calculate various statistics and present them in a variety of ways that are completely under your control. **Table** allows you to choose how you want to assemble variables and statistics for display in rows, columns, and super-columns or super-rows. A super-column or super-row has a variable nested below it. Variables can be stacked or nested. Nested means that all of the values for one variable are displayed below the individual values of another variable. You can manipulate table structure, content, and presentation format.

With this command there a few limitations:

- up to 4 variables can be specified in the by()
- up to 5 statistics can be displayed in each cell
- the sum of the number of rows, columns, super-columns, and super-rows is called the number of margins. A table may contain up to 3000 margins, e.g. a one-way table may contain 3000 rows, a two-way table may contain 2998 rows and 2 columns, or 2997 rows and 3 columns and so forth

Commands that produce similar results are:

tabstat - displays summary statistics for a series of numeric variables in a single table
tabsum - produces one- and two-way tables of means and standard deviations - this command is faster, but the table command is more flexible
tabulate - one- and two-way tables of frequencies
tab1 produces one-way tabulation for each variable
tab2 produces two-way tabulations of all combinations of the variables

Let's compare the **tabulate** command with the **table** command if we want to create two-way tables.

Open the member file we created from Section 1 that contains the age variable, Q1A-AGE.DTA.

1. **File**/**Open...**
2. Select q1a-age.dta
3. Click on **Ok**

4. Copy the command, paste it into a new do-file and add comments. Add the commands to turn more off and start the log file for this session.

The **tabulate** command

First, do a simple two-way table using the **tabulate**.

1. From the menus click on
   Statistics
   Summaries, tables & tests
   Tables
   Two-way tables with measures of association
   The tabulate2 - Two-way tables *dialog box opens.*
2. In the **Row Variable** box select **ca2**
3. In the **Column Variable** box select **age_gp**
4. Under Cell Contents click in the box next to **Within <u>column</u> relative frequencies** to put a ✓.
5. Click in the box ✓ next to **Within <u>row</u> relative frequencies**.
6. Click on the **Ok** button. The command will be executed.
7. Copy the command into the Do-File Editor and write a comment to explain what the command does.

The Stata command is:

```
tabulate ca2 age_gp, column row
```

Below is the output.

```
+-------------------+
| Key               |
|-------------------|
|     frequency     |
|  row percentage   |
| column percentage |
+-------------------+

   relation to |                  Age group
          head |    0 to l0   11 to 19   20 to 60   61 and ol|     Total
---------------+--------------------------------------------+----------
          head |         0          6        296         41 |       343
               |      0.00       1.75      86.30      11.95 |    100.00
               |      0.00       2.22      47.13      83.67 |     22.57
---------------+--------------------------------------------+----------
  wife/husband |         0         25        280          5 |       310
               |      0.00       8.06      90.32       1.61 |    100.00
               |      0.00       9.26      44.59      10.20 |     20.39
---------------+--------------------------------------------+----------
  son/daughter |       503        184         31          0 |       718
               |     70.06      25.63       4.32       0.00 |    100.00
               |     87.78      68.15       4.94       0.00 |     47.24
---------------+--------------------------------------------+----------
 mother/father |         0          0          5          1 |         6
               |      0.00       0.00      83.33      16.67 |    100.00
               |      0.00       0.00       0.80       2.04 |      0.39
---------------+--------------------------------------------+----------
other relative |        70         55         16          2 |       143
               |     48.95      38.46      11.19       1.40 |    100.00
               |     12.22      20.37       2.55       4.08 |      9.41
---------------+--------------------------------------------+----------
         Total |       573        270        628         49 |     1,520
               |     37.70      17.76      41.32       3.22 |    100.00
               |    100.00     100.00     100.00     100.00 |    100.00
```

The **table** command

Let's use **table** to produce a similar table. However with the table command we cannot ask for row or column percentages - this command is generally used for summary statistics. Frequency and Totals are possible to select from this command.

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   Table of summary statistics (table).
2. Under the **Main** tab select **ca2** in the Row variable: box
3. Click in the box ✓ next to **Column variable** and select **age_gp** in the box below.
4. In the **Statistics section, #1**, select **Frequencies** from the drop down box.
5. Under the **Options** tab check ✓ **Add row totals** and also check ✓ **Add column totals**
6. Click on the **Ok** button.
7. Copy the command into the Do-File Editor and write a comment to explain what the command does.

The Stata command is:

```
table ca2 age_gp, contents( freq ) row col
```

The results are:

```
----------------------------------------------------------------------------------
relation to     |                          Age group
head            |     0 to 10     11 to 19     20 to 60  61 and older        Total
----------------+-----------------------------------------------------------------
           head |                        6          296            41          343
   wife/husband |                       25          280             5          310
   son/daughter |         503          184           31                        718
  mother/father |                                     5             1            6
 other relative |          70           55           16             2          143
                |
          Total |         573          270          628            49        1,520
----------------------------------------------------------------------------------
```

Note: the word "contents" can be abbreviated to "c", e.g. c(freq).

Comparison of the commands **summarize**, **tabulate** and **table**

The following is a comparison of computing averages using **summarize**, **tabulate** and **table**, based on an example from section 2.

1. **File**/**Open**
2. Select hh-file3.dta, Click on **Ok** and copy the command into the Do-File editor.

First we will use the **summarize** command:

1. From the **Statistics** menu select
   Summaries, tables & tests
   Summary statistics
   Summary statistics
   The summarize - Summary statistics *dialog box opens.*
2. Select **cprod_ae** in the "**variables**" box.
3. Be sure that the under "Options" in this tab, **Standard Display** has been selected.
4. Click on the "**by/if/in**" tab.
5. Click in the box "**Repeat command for groups defined by**"
6. In the box below this option, select **district quart**
7. Click the **Ok** button.

For each combination of district and quart, we see the summary statistics. This output is difficult to read.

Next we will use the **tabulate** command:

1. From the menus click on
   Statistics
   Summaries, tables & tests
   Tables
   One/two-way table of summary statistics
   The tabsum - One/two-way table of summary statistics *dialog box opens.*
2. In the Variable 1: box select **district**
3. In the Variable 2 (optional): box select **quart**.
4. In the Summarize Variable: box select **cprod_ae**.
5. For output we are only interested in the mean, so check the boxes next to
   ✓ Suppress standard deviation
   ✓ Suppress frequencies
   ✓ Suppress number of observations
6. Click on **Ok** to run the command.

In the Results window we see:

```
               Means of Calories per adult equivalent per day

           |       Calorie production quartile
  district |         1          2          3          4 |     Total
-----------+--------------------------------------------+----------
    monapo | 1248.5475  2539.3641  3997.4884  9150.0217 | 4206.4673
    ribaue |  1502.242   2554.488  4062.3014   7607.719 | 3900.7966
   angoche | 1297.9691   2465.509   3698.807    8495.49 | 3950.2608
-----------+--------------------------------------------+----------
     Total | 1352.5022  2519.7353  3919.3795  8399.3828 | 4014.5181
```

Notice that the number of decimals is not uniform. We can fix that with the **table** command.

1. From the menus click on
   Statistics
   Summaries, tables & tests
   Tables
   Table of summary statistics (table).
2. Press the Reset button R to clear the boxes .
3. Under the **Main** tab select **district** in the Row variable: box
4. Click in the box ✓ next to **Column variable** and select **quart** in the box below.
5. In the **Statistics section, #1**, select **Mean** from the drop down box.
6. In the box to the right specify the variable to use for the **Mean** statistic - **cprod_ae**

We would also like to see the minimum and maximum values

7. Click on the drop down box next to #2 and scroll down to **Maximum** and select that statistic. For the variable select **cprod_ae**
8. Click on the drop down box next to #3 and scroll down to **Minimum** and select that statistic. For the variable select **cprod_ae**
9. Under the **Options** tab check ✓ **Add row totals** and also check ✓ **Add column totals**
10. To format the numbers, check ✓ next to the format box and change the contents to read:
    %9.3f
    The Help format... button shows different formats that can be specified. This format says to use a width of 9 with 3 decimals in fixed (f) format.
11. Click on the **Ok** button.
12. Copy the three different commands from the Results window into the Do-File Editor and write a comment to explain what each command does differently.

The three Stata commands are:

```
bysort district quart: summarize cprod_ae
tabulate district quart, summarize(cprod_ae)
nostandard nofreq noobs
table district quart, contents( mean cprod_ae max
cprod_ae min cprod_ae ) row col format(%9.3f)
```

For each district, the first row is the mean, the second row is the max and the third row is the min.

```
----------------------------------------------------------------
          |            Calorie production quartile
 district |        1          2          3          4      Total
----------+-----------------------------------------------------
   monapo | 1248.547   2539.364   3997.488   9150.021   4206.467
          | 1972.673   3175.779   5066.724  28465.750  28465.750
          |  294.101   1984.114   3225.948   5107.123    294.101
          |
   ribaue | 1502.242   2554.488   4062.302   7607.719   3900.797
          | 2030.398   3141.388   4983.722  13123.971  13123.971
          |  429.293   2082.420   3190.407   5151.591    429.293
          |
  angoche | 1297.969   2465.509   3698.807   8495.490   3950.261
          | 2023.654   2996.365   4691.524  20485.100  20485.100
          |  353.882   2037.201   3009.462   5021.753    353.882
          |
    Total | 1352.502   2519.735   3919.379   8399.383   4014.518
          | 2030.398   3175.779   5066.724  28465.750  28465.750
          |  294.101   1984.114   3009.462   5021.753    294.101
----------------------------------------------------------------
```

The **table** command permits you to specify more than one variable to summarize and also permits formatting of the contents of the table.

Print a table from the Viewer

A simple way to print a table you have just created, is to open the **Viewer**, select the table and print.

1. Open the Viewer - Click on **File / View**. A dialog box opens, asking for the name of the file
2. Click on the **Browse** button and change to the directory where you are writing the log file and select the file session3.smcl and click on **Ok**
3. Scroll down to the table you want to print and block it.
4. Click on **File**/**Print Viewer**. The **Print** dialog box opens. Under **Page Range** click on the radio button next to **Selection**. Then click on **Print** .
5. Another dialog box opens called **Printer Settings** In this box you can specify a Header, a Name and a Project. If you do not want line numbers and the Stata logo to print, you should uncheck the boxes next to Print Line #'s and Print Logo.
6. Click on **Ok** to print the selection.

Exercise 3.1

Produce a similarly formatted table using calories retained as you did in Exercise 2.1. **Include totals** by production quartile. Your table should look similar to the table below:

```
----------------------------------------------------------------
          |              Calories retained quartile
 district |        1          2          3          4      Total
----------+-----------------------------------------------------
   monapo | 1171.574   2239.088   3343.003   7619.102   3570.975
          | 1806.867   2554.892   4303.122  20873.971  20873.971
          |  224.490   1888.330   2685.971   4359.737    224.490
          |
   ribaue | 1251.391   2171.697   3165.192   5828.970   3081.416
          | 1790.432   2566.006   3731.045   9464.901   9464.901
          |  429.293   1835.298   2578.604   3825.879    429.293
          |
  angoche |  929.418   1718.789   2442.247   5022.290   2506.498
          | 1395.962   1984.059   3063.996  12674.862  12674.862
          |  207.908   1447.059   1997.711   3134.742    207.908
          |
    Total | 1118.378   2040.130   2977.233   6135.476   3044.233
          | 1806.867   2566.006   4303.122  20873.971  20873.971
          |  207.908   1447.059   1997.711   3134.742    207.908
----------------------------------------------------------------
```

Multiple Response Questions

One type of question used in survey research expects the respondent to select multiple answers. A single variable cannot record the answers to this type of question adequately, because a variable can have only one value for each case. The solution is to record each possible response in a different variable. The responses can be analyzed separately using commands you have already seen (**tabulate**), but ideally we want to analyze these related variables jointly.

Multiple dichotomy (yes/no questions)

<u>Multiple dichotomy - yes/no questions</u>: If a survey question asks the respondent to "check all that apply" from a set of ten choices, ten variables must be used to code the responses. A separate variable is required for each of the ten responses. Each variable has a value to indicate whether the response was checked (1) or yes, or not checked (2) or no. An example of this type of question can be found in the household level survey questions (see appendix), Section V - Agricultural Sales, question 64 - have you increased the quantities sold over the last five years? All of the variable names associated with this question begin with H64.

Open the file:

1. **File**/**Open**
2. Select c-hh.dta
3. Click on **Ok**
4. Copy the command to the Do-file editor.

In this survey 1 = yes and 2 = no. Questions you might ask are:

A. <u>How many respondents increased sales quantities of maize?</u>

To answer this question you can count the number of times the value of 1 appears in the variable associated with maize. To count the number of times a value appears in the variable. The command is **count**

The **count** command

```
count if h64a == 1
```

In the Results window you see the value of 86. You could also run a frequencies:

```
tabulate h64a
```

The tabulate shows that 147 did not increase sales of maize as well as 86 households who did. Now we change the question.

The **egen** command

B. How many crops increased in sales within the household?

For this question we can ask to sum the number of 1's in those variables using the **egen** command. We need to recode the value of 2 to 0.

Recode:

1. Select **Create or change variables** from the **Data** menu
2. Select **Other variable transformation commands** **Recode categorical variable**
   The **recode - Recode categorical variable** *dialog box opens.*
3. Under the **Main** tab, click in the Variables box and select all the variables that start with h64, e.g. h64a h64b h64c h64d h64e h64f h64g h64h
4. In the box for **Required**: type **(2=0)**
5. Click on **Ok**

The values in the h64 variables are either 0=no or 1=yes. We are ready to count the number of crops that increased in sales:

1. Select **Create or change variables** from the **Data** menu
2. Select **Create new variable (extended)**
   The **egen - Extensions to generate** *dialog box opens.*
3. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **ncrops**
4. In the box for **Generate variable type as**: select **integer**
5. For the **egen function** box, scroll down to **Row sum** and select that.
6. Click on the **egen function argument - Variables** box type **h64***
7. Click on **Ok**

The **tabstat** command

The Stata command is:

```
egen ncrops = rsum(h64*)
```

Now you can do a frequencies on the new variable:

```
tabulate ncrops
```

Two households had increased sales on all crops (ncrop = 8), 136 households had not increased sales on any of the crops (ncrop = 0).

C. <u>What is the distribution of the crops?</u>

For this question we can use the **summarize** command, but we could also use the **tabstat** command:

1. From the menus click on
   Statistics..
   Summaries, tables & tests
   Tables
   Table of summary statistics (tabstat).
2. Under the **Main** tab type **h64*** in the Variables: box
3. Under “Statistics to display” place a tick in the first box and select **Sum** as the statistic.
4. Under the **Options** tab in the **Use as Columns** change to **Statistics**.
5. Click on **Ok**

The Stata command is:

```
tabstat h64*, statistics( sum ) columns(statistics)
```

You see all the h64 variables with a count of the number of cases where yes was specified. Manioc (h64b) was the most frequent crop for which households had increased sales (115), sorghum (h64g) was the least (14).

Multiple response

<u>Multiple Response</u>: On the other hand, if the survey question asks the respondent to "list up to 4 choices" from a set of ten, four variables must be used to code the responses. The set of possible responses are assigned sequential values and are used for each of the 4 variables. The respondent must record a different value in each of the 4 variables. These types of variables are called *multiple response* variables.

Question 35 in the household questionnaire is an example of a multiple response question. It asks about crops grown principally to be sold. Each household is asked to specify up to three main crops which are coded into variables **h35a**, **h35b**, and **h35c**. Codes are provided for five of the most common crops. The question is left open-ended, however, since a code of 6 is allowed for a crop not on the list. The name of the crop is written down on the questionnaire and later assigned a code.

Because the question was open ended, more categories were added to these variables than what appears in the annex. After the data are collected, the researcher assigns a code to each of the crops specified for "6-other" - this procedure is called “post-coding”. Codes and value labels are entered into the data file and the data changed from the value of 6 to the appropriate code. As you will see, using the **tabulate** command, eleven different crops were specified for question 35.

Stata does not have an official command that will tabulate data collected in this format. We can do frequencies of each variable or develop commands to pull out specific information. There is a user-written command called tabu (Peter Sasieni, STB-25; Stata 3.1). For each variable in a list of 2 variables, this command will tabulate the number of times it takes on the values 0, 1, ..., 9; the number of times it is missing; and the number of times it is equal to some other value. String variables are not tabulated but are identified at the end of the displayed table. To download this ado file, connect to the Stata website to the STB - 25 site.

Using this type of analysis you could state the following: Of the households that sold more than one crop, rice was the primary cash crop, peanuts were the next most important, 22 households sold rice and peanuts. Looking at peanuts as the primary crop, 37 households also sold rice as the secondary crop. This analysis does not show the total that sold rice as a primary crop. The **tab1** command would give you that information.

For now, you can run a **tab1** command on the three variables to look at the frequencies.

Other Types of Analyses

Weights

Stata provides for a method to analyze data using different types of weights. The type of weight that is to be used with a set of data will depend on the type of sampling that has been used.

See the table below for an explanation of the available weight types.

Sub-Command	**type of weight**	**Definition**
fweight or frequency	frequency weights	indicates duplicate observations, this value is always an integer. If the fweight associated with an observation is 5, that means there are really 5 such observations, each identical
pweight	Sampling weights	inverse of the probability that this observation is included in the sample due to the sampling design. A pweight of 100 indicates that this observation represents 100 subjects in the population. There are qualifications to this weight when used with survey analysis commands
aweight or cellsize	analytic weights	inversely proportional to the variance of an observation. The observations typically represent averages and the weights are the number of elements that produced the average
iweight	Importance weights	relative "importance" of the observation. This weight is generally used by programmers who want to produce a specific computation.

To read more about weights look at the User manual, section 23.16 - weighted estimation. If you use the generic "weight" sub-command, Stata will tell you which weight it assumes you want to use. Not all commands will allow a weight to be included. The format is

[typeofweight=variable].

Let's use one of the Stata's data files to explore this sub-command.

1. **File / Open**
2. Change to the C:\Stata8\ado\base\c and select the file census.dta.
3. Click on Ok . - Remember to copy the command and paste it to the editor.

Use the **Browse** button to look at the data. There is one observation for each state. The variable called **pop** is the total population for the state. The variable called **medage** is the median age of the population. First lets get the population-weighted mean.

The (a)weight subcommand (analytic)

1. From the **Statistics** menu select
   Summaries, tables & tests
   Summary statistics
   Summary statistics
   The Summarize - Summary Statistics *dialog box opens.*
2. Select **medage** in the “**variables**” box.
3. Click on the “**weights**” tab.
   Note that only 3 types of weights are available to choose from. There is also a help button on weights.
4. Select **Analytic weights**
   Each observation represents the mean for the state, so analytic weight is chosen.
5. In the box below **Weight**, select **pop**
6. Click the **Submit** button.

In the output, the sum of the weight is 225,907,472, which is the population of the U.S. in the 1980 census. The weighted mean is 30.11.

7. Now, back in the dialog box, click on **None** under the **Weight** tab.
8. Click on **Ok**

The unweighted mean is 29.54

The Stata commands are:

```
use "C:\Stata8\ado\base\c\census.dta", clear
summarize medage [aweight=pop]
summarize medage
```

Survey weights are discussed in the next section.

Indicator variables

An indicator variable is a special case of a categorical variable. An indicator variable has two groups only, whereas other categorical variables can have more than two groups. Usually the values are 0 and 1 or no/yes.

Examples of indicator variables are:

Is a person a citizen of the U.S.? (no/yes).
Does a farmer use fertilizer? (no/yes).

Stata can convert continuous variables to categorical and indicator variables and it can also convert categorical variables to indicator variables.

Suppose we want to create a new variable that indicates whether a person is 18 years old or older. You could have generated a

Converting continuous variables to indicator variables

new variable and assigned it a value of 1 if ca3 > =18. Then you would need a second step to recode the system missing to 0. There is another way to create this variable.

We will use the file c_q1a.dta. Open the file and then create a new variable using the **generate** command following the steps below:

1. **File / Open** Select c_q1a.dta and click on **Open**.
2. Check to see if there are any missing values in the age variable **- ca3**. Use the list command
   list if ca3 >=.
   We are checking to see if there are missing values because Stata considers missing values to be greater than any number.
3. Select **Create or change variables** from the **Data** menu
4. Select **Create new variable**
   The generate - Generate a new variable dialog box opens.
5. Under the **Main** tab, type the name of the new variable in the **Generate Variable** box: **age18p**
6. For the **Contents** box, type in
   ca3>=18
7. Click on the **Generate variable as type** drop down box and change to **byte**.
8. Click on **Ok** .
9. Run a **tabulate** to look at the results.
   Note: if there had been a missing value for an observation, that observation would have been assigned a value of 1.

It would have been better to put a qualifier on the command to assign the values to cases where ca3 was not missing (e.g. ca3 < .).

```
generate byte age18p = ca3>=18 if ca3 < .
```

Then, any missing values in **ca3** would also be missing in the new variable **age18p**.

Converting categorical variables to indicator variables

Suppose that you want to do regression analysis and control for effects of the different geographic regions. We have a variable called district which has 3 categories. We want to create indicator variables for the three districts. These types of variables are also called dummy variables. First let's run the describe command to look at the contents of the file:

```
describe
```

Next let's look at the values and labels for the variable **district**:

```
label list district
```

To make 3 indicator variables we can type:

```
tabulate district, generate(district)
```

Now, run the **describe** command again:

```
describe
```

Three new variables have been created, called district1, district2, and district3. We can examine the variables using the **tab1** command.

```
tab1 district*
```

STATA 9.1 for Windows SAMPLE SESSION

SECTION 4 - Table and Graphs - how to bring them into a word processor, and Survey estimation, accounting for design effects

How to move Stata results into other applications

The objective of this module is to give you the tools necessary to prepare reports, i.e. to learn how to move Stata results into other applications. The method is simple: once a graph or a table has been produced, it can be printed or incorporated into reports prepared using word processors or publishing programs. Incorporating tables and graphs from Stata can be done using a the copy and paste procedure. You should save the log file as well in case you need other tables that were created. Find the table in the session3.scml file that showed the count of the "relation of head" to "age group" cross-tabulation:

Copying tables from the Viewer

1. Click on **File / Log / View...** In the **Choose File to View** dialog box, click on the **Browse** button.
2. Select session3.smcl .
3. Click on **Open**, Then click on **Ok**
4. Locate a table that you want to copy to your word processor. Use your mouse to block the table.
5. Press **Ctrl-C** (copy). This key sequence copies what you have blocked.
6. Now open your word processor software if it is not already open.
7. Place your cursor where you want the table to appear. Press Ctrl-V (paste) to paste the table.
8. In your word processor, block the text that you just pasted. Now change the font to a fixed font, e.g. Courier New or Letter Gothic. Click on **Format, Font**, and select the font. The size of the font may need to be adjusted depending on the margins of your paper. The default will be 12 and you may want to select 10 or 9 or 8.

Below is an example of a table copied into a word processor before the font is changed to a fixed pitch:

```
--------------------------------------------------------------------------------
relation to   |                       age_gp
head          |    0 to 10     11 to 19     20 to 60  61 and older       Total
---------------+----------------------------------------------------------------
         head |                  6          296          41          343
  wife/husband |                       25          280          5          310
   son/daugher |        503          184          31                      718
 mother/father |                                     5          1          6
other relative |        70          55          16          2          143
              |
        Total |       573          270          628          49        1,520
--------------------------------------------------------------------------------
```

Below is the same table after the font is changed to a "fixed pitch" and the font size is adjusted so that the table will fit on the page.

```
-------------------------------------------------------------------------------------
relation to    |                        Age group
head           |     0 to 10      11 to 19      20 to 60  61 and older         Total
---------------+---------------------------------------------------------------------
          head |                         6           296            41           343
  wife/husband |                        25           280             5           310
  son/daugher  |         503           184            31                         718
 mother/father |                                       5             1             6
other relative |          70            55            16             2           143
               |
         Total |         573           270           628            49         1,520
-------------------------------------------------------------------------------------
```

Copying tables from the Results window

You can also copy the information from the **Results** window into your word processor. Stata provides three choices from the Edit menu for copying tables. Click on **Edit** to look at the choices.

1. Copy text - Ctrl-C - copies the table as straight text.

2. Copy table - Shift-Ctrl-C - copies the table and includes tabs where it thinks there should be tabs

3. Copy table as HTML - Shift-Ctrl-Alt-C - copies the table into HTML format.

You may encounter problems with the second and third options if you use these. Stata determines if there should be tabs and may not make the correct decision. You might need to increase the width of the columns in the output to make sure that tabs are included. Below is an example of the same table using the Shift-Ctrl-C

```
relation to                               Age group

head  0  to 10                  11  to 19   20 to 60    61    and    older  Total

head        6                   296              41     343
wife/husband                                25   280                 5      310
son/daughter                       503           184    31
    718
mother/father                                    5                   1      6
other relative                     70            55     16                  2
    143

Total     573                      270      628                49    1,520
```

Quite a bit of editing is required to make the above table presentable.

Exercise 4.1.

Select another table from your Session3.SMCL file. Repeat steps 1 to 8 as outlined above.

Graphs

The process to copy output to a word processor is basically the same for Graphics, such as pie charts and histograms, but there is more flexibility in the ways to save the file, along with more difficulties in getting just the look you want. As an example, we will look at the distribution of cashew tree ownership across households in the Mozambique data, using a histogram.

Open a new do file and place the requisite information at the top, e.g.

```
/* Purpose:  copying output to word processors, graphs */
/* Your name - date */

/* Stata recommends you include the version that the do file was
written in.

version 8.2
set memory 20M

/*turn on the log */
log using c:\docs\sample\session4, replace

/*turn off "more" so the whole file will run" */
set more off
```

Save this do file under the name session4.do.

We are now ready to open the household file that contains the tree ownership variable, c-hh.dta.

1. **File**/**Open**
2. Select c-hh.dta and click on **Ok**
3. Paste the command from the Results window to the do file editor.

Histogram

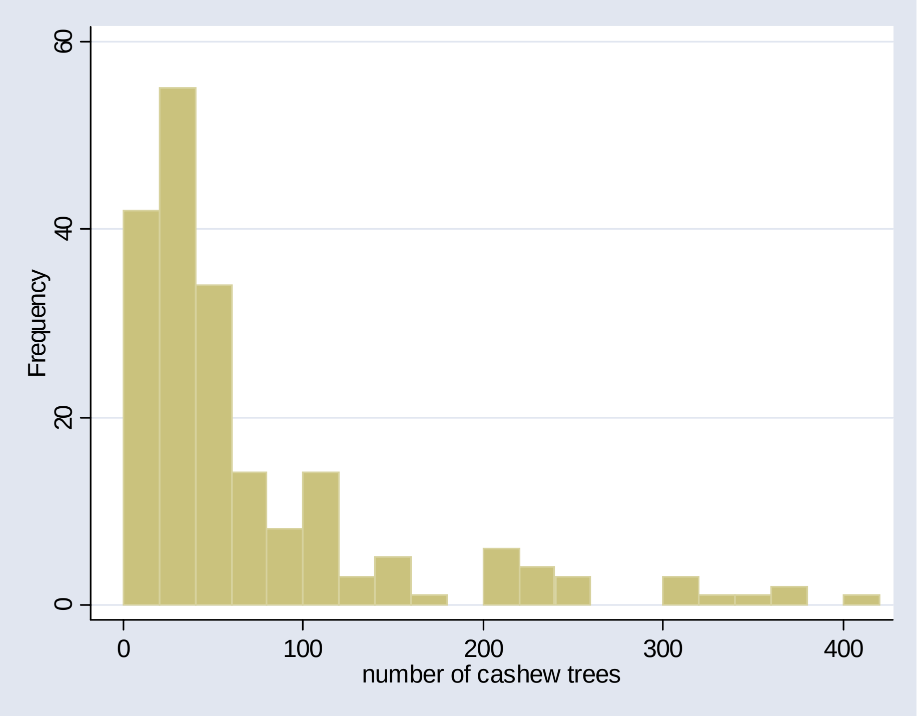
Create the Histogram chart using the variable **H57** (number of trees owned):

4. Select **Graphics / Easy graphs / Histogram**.
5. In the Variable box select **H57** (Number of cashew trees) .
   Note: you can specify if the variable is continuous or discrete.
6. Click on the **Options** tab. Under this tab, ✓ check the box next to **width of bin** and type **20**
7. Under **Y-Axis** click on the radio button next to **Frequency**
8. Click on **Ok**. Copy the command to the Do-file editor.

The Stata command is:

```
histogram h57, frequency width(20)
```

You should get a histogram chart that looks like this:



To copy this graph to your word processor,

1. Click on **Edit / Copy graph.**
   `You could also right-click on the graph itself and select "Copy Graph".`
2. Open your wordprocessor and click on **Edit / Paste**

You will not be able to edit this graph, other than the size, placement, wrapping of text and other basic aspects allowed by your word processor.

You can also save a chart as a separate file.

1. Click on **File / Save graph**.
2. A dialog box opens where you can type the name of the file. The default extension is .gph which is the format that Stata recognizes as a graph file. If you save the file with a .gph extension, you can then open the graph again within Stata.
3. You can also save the graph into different formats, e.g. windows metafile (wmf). Wordprocessors can import a graph with this extension into a graphic box.

Once a graph window has been closed, you cannot reopen it unless you have saved the graph to a file. You can rerun the command that created the graph to see the graph again.

Scatter plot using "by" subcommand

Let's look at another graph. We will use the file created in the last session, hh-file4.dta. We can plot adult equivalents per household with total calories produced.

1. **File**/**Open**
2. Select hh-file4.dta and click on **Ok**
3. Select **Graphics / Two-way graphs (scatterplot, lines, etc.).**
4. In the dialog box, the type of plot by default is **scatter**. If you click on the drop down box next to the type you can see the choices of line, connected and area, bar, etc.
5. For the **X axis**, select the variable **cprod_tt**. For the **Y axis** select the variable **ae**
6. Click on the **Submit** button to view the graphic.
7. Close the graph and return to the dialog box. If we want to see the distribution by district, click on the "**By**" tab. In the **Variables** box select **district**
8. Click on the **Ok** button to view the graphic.
   What are these graphs telling you?

Close the graph.

Overlaid graphs

Graphs can also be overlaid.

1. Select **Graphics / Overlaid two-way graphs** .
2. In the dialog box, under the tab "**Plot 1**" select type of plot to be **scatter**.
3. For the **X axis**, select the variable **cprod_tt**. For the **Y axis** select the variable **ae**
4. Under the tab "**Plot 2**" select type of plot to be **lfit** (linear prediction plot)
5. For the **X axis**, select the variable **cprod_tt**. For the **Y axis** select the variable **ae**
6. Click on the **Submit** button to view the graphic.
   What are these graphs telling you?
7. Change "**Plot 2**" to be the type of **qfitci** (quadratic prediction plot with CI's).
8. Click on the **Submit** button to view the graphic.
   What are these graphs telling you?
9. If we want to see the distribution by district, click on the "**By**" tab. In the **Variables** box select **district**
10. Click on the **Ok** button to view the graphic.
    What are these graphs telling you?

The Stata commands are:

```
twoway (scatter ae cprod_tt)

twoway (scatter ae cprod_tt), by(district)

twoway (scatter ae cprod_tt) (lfit ae cprod_tt)

twoway (scatter ae cprod_tt) (lfit ae cprod_tt),
by(district)

twoway (scatter ae cprod_tt) (qfitci ae cprod_tt)

twoway (scatter ae cprod_tt) (qfitci ae cprod_tt),
by(district)
```

Survey Estimation - Accounting for Design Effects

Stata provides statistical commands that have been developed specifically for survey analyses. Chapter 30 in the User's Guide discusses these commands as well as the manual called Survey Data. Most of these commands begin with the letters **svy**. There are a few of the survey commands that do not begin with these letters.

Per these manuals - survey data generally have three importance characteristics:

1. The weights applied to survey data are sampling weights - also called probability weights
2. The sample is clustered
3. Stratification is used in selecting the sample

Briefly, sampling weights are used in analysis to give estimators that are approximately unbiased for whatever is being estimated for the whole population, i.e. one observation represents many elements in the population from which the sample is drawn.

Clustering by districts or villages is used in almost all survey sampling rather than selecting an independent sample. Further subsampling may occur within a district or a village as well. Units at the first level of sampling are called the "*primary sampling unit*" or "PSU". In Zambia the earlier sampling method divided the districts into census supervisory areas (CSA). Within the CSA, Standard Enumerator Areas (SEA) were set up. The "PSU" for this sample was determined to be the SEA. To identify each SEA as being unique the three variables, district, CSA and SEA, must be merged into one variable.

Different groups of clusters may further be sampled separately which is called strata. In the Zambia example the province-district is considered the strata. Strata are considered to be statistically independent and can be analyzed as such.

To summarize, weights are used to get the correct point estimates. Clustering and stratification are used to get the correct standard errors.

The svy commands also calculate the design effects of deff and deft. ***Deff** is equal to the design-based variance estimate divided by an estimate of the variance that would have been obtained if the survey was carried out using simple random sampling.* ***Deft** is approximately equal to the square root of deff.* Further explanation of these two terms can be found in the Survey Data manual under the command **svymean.**

We will use a data set from Zambia for the Post harvest 2001/2002 season where the area for the specific types of crops is tested.

1. **File**/**Open**
2. Select zam_areatest.dta and click on **Ok**
3. Paste the command to the do-file editor.

Use the browse command to look at the data.

```
browse, nolabel
```

Variables have been computed from the dist, csa, and sea variables to create stratcom (prov & dist) and clust (csa & sea). Close the browser.

First we will look at how to define stratified random sampling to be able to use the survey commands, to account for weighting, clustering and stratification. Before we can run any analysis, the weight, strata and PSU identifier variables must be set.

4. Click on **Statistics / Survey data analysis**
5. Then click on **Setup & utilities / Set variables for survey data**
6. Under the **Strata:** box select **stratcom**
7. Under the **Primary sampling unit:** box select **clust**
8. Under the **Sampling weights:** box select **hhwgt**
9. Click on **Ok**
10. Paste the command to the do-file editor.

The Stata command is:

```
svyset [pweight=hhwgt], strata(stratcom) psu(clust)
```

We can use the **svytotal** command to look at the total estimates.

11. Click on **Statistics / Survey data analysis**
12. Then click on **Univariate estimates / Totals for survey data**

13. In the **Variables** box select **maisea ricea milleta sunfa**
14. Click on **Submit**
15. Paste the command to the do-file editor.

Let's run the same analysis with only the weight specified to see the difference.

17. Click on the **Survey settings** button.
18. ✓ check the box next to **Clear all previous settings**
19. ✓ check the box next to **Use simple random sample (SRS)**
20. Click on **Ok**
21. You are now back in the dialog box for defining svytotal. Click on **Ok**

Note, we have gotten the same point estimate as the design-based estimate, but the standard errors are much smaller. The second table does not account for the sampling design.

```
. svyset [pweight=hhwgt], strata(stratcom) psu(clust) clear(all)
pweight is hhwgt
strata is stratcom
psu is clust

. svytotal maizea ricea milleta sunfa, available

Survey total estimation
pweight:  hhwgt                                    Number of obs    =      6601
Strata:   stratcom                                 Number of strata =        69
PSU:      clust                                    Number of PSUs   =       394
                                                   Population size  = 807413.58
------------------------------------------------------------------------------
   Total |   Estimate    Std. Err.   [95% Conf. Interval]        Deff
---------+--------------------------------------------------------------------
  maizea |   649230.9    25105.89    599840.3    698621.5    4.771045
   ricea |   14472.95    2360.009    9830.125    19115.77    4.410844
 milleta |   61770.91    7346.125    47318.95    76222.87    7.506339
   sunfa |   24319.15    3418.858    17593.26    31045.04    3.851162
------------------------------------------------------------------------------

. svyset [pweight=hhwgt], srs clear(all)
pweight is hhwgt
simple random sample (SRS)

. svytotal maizea ricea milleta sunfa, available

Survey total estimation
pweight:  hhwgt                                    Number of obs    =      6601
Strata:   <one>                                    Number of strata =         1
PSU:      <observations>                           Number of PSUs   =      6601
                                                   Population size  = 807413.58
------------------------------------------------------------------------------
   Total |   Estimate    Std. Err.   [95% Conf. Interval]        Deff
---------+--------------------------------------------------------------------
  maizea |   649230.9    14013.13    621760.6    676701.2    1.486391
   ricea |   14472.95    1327.559     11870.5    17075.39    1.395732
 milleta |   61770.91    3942.684    54041.97    69499.84    2.162198
   sunfa |   24319.15    1907.919    20579.01    28059.29     1.19936
------------------------------------------------------------------------------
```

STATA for Windows SAMPLE SESSION

Annexes

Annex I

This annex provides a brief reference guide and to explain the various functions of the STATA commands most commonly used. This annex was developed by Ellen Payongayong.

The commands in the table below do not contain the full Stata syntax.

Note that commands can be abbreviated. In the Help Syntax Viewer, the syntax explanation will show how much of the command must be typed, e.g. "Summarize" can be shortened to "su" or "sum". In this Help viewer, the letters that are required for the command are underlined.

Command	Description
`pwd`	tells you which directory you're in
`cd` *`{c | d | e...)`*`:`	cd c: changes drives to c drive
`cd ..`	changes directory one level higher
`cd` *`(path)`*	changes current directory to that specified in path
`cd\`	takes you to the root directory
`dir`	lists contents of current directory
`use` *`filename1`*	loads file into memory
`save` *`filename2`*	saves current file in memory into *filename1*. if filename already exists, stata will not let you overwrite it
`save` *`filename2`*`,` **`replace`**	saves current file in memory into *filename2*, overwriting any file in working directory that is currently named *filename2*
`save``,` **`replace`**	saves current file in memory into filename of that which is currently in memory
`edit`	brings up the data editor
`browse`	brings up the same data "editor'' as in **edit,** but will not allow you to change data
`describe`	gives a description of the data file: number of observations, number of variables, list of variables, variable type and width, variable labels (if any)
`summarize`	gives basic summary statistics: number of valid observations, mean, standard deviation, minimum value and maximum value
`list`	lists observations
`keep`	retains in memory only those variables **or** cases specified
`drop`	discards from memory all variables **or** cases specified
`tabulate`	generates one- and two-way frequency tables
`tab1`	generates one-way table for each variable specified after the command.
`log using` *`filename`*	saves **all commands and related output** into specified file. the default format is SMCL for Stata Markup and Control Language. file is given extension .smcl

Command	Description
log using *filename, text*	saves all commands and related output into an ASCII file with extension .txt.
log { off | on | close}	**off** temporarily suspends the log file (switches it "off"); **on** switches the log "on" and **close** closes the log file
log using *filename*, **append**	adds subsequent commands to an existing log
log using filename, **replace**	saves all commands and related output into the specified file, overwriting said file if it already exists

By opening a log file with **cmdlog** *instead of* **log**, *you record only what you type in the command window (results are suppressed). The same basic syntax applies for both cmdlog and log. You can open both an smcl file and a log file.*

Command	Description
clear	clears data set from memory
help *command*	accesses help feature of Stata
exit	exits stata
sort *varlist*	sorts observations in ascending order according to the specified variable.
(1) **note:** *"..."* (2) **note** *varname* **:** *"..."* (3) **notes**	(1) allows you to enter notes about the dataset (2) allows you to enter notes about variable varname (3) calls up all notes in memory. Notes are saved in the dataset.
label variable *varname "lblnamel"*	assigns a variable label to variable specified
(1) **label define** *lblname # "label1" [# "label2"]...* (2) **label values** *varname1 lblname*	(1) assigns labels to integers (#) and stores these in the value label *lblname* (2) associates the value label lblname to the variable v*arname1* e.g. label define gender 1 "female" 2 "male" label values sexhead gender
label list	lists all value labels
recode	modifies the value of a variable using rules specified
generate	creates a new variable
set memory	changes the amount of memory allocated to the data area; Stata suggests setting the memory to at least one and half times the size of the file you want to load in the memory of the computer.
replace	changes the value of an existing variable
count	when used with **if,** it counts the number of observations that meet the specified condition; otherwise, it counts the number of observations in the dataset
rename	changes the name of an existing variable
collapse	converts the data file in memory into another data set of means, medians, etc.
merge *varlist* **using** filename	merge joins corresponding observations from the dataset currently in memory (called the master dataset) with those from the Stata-format dataset stored as *filename* (called the using dataset) into single observations; performs a match merge on *varlist* when these are specified.

Command	Description
	the variable _merge, which gives information on the results of the merge command, is added to the file. _merge==1 obs. from master data _merge==2 obs. from using data _merge==3 obs. from both master and using data
`merge` *`varlist`* `using filename, nokeep`	"nokeep" causes merge to ignore observations in the using data that have no corresponding observation in the master.
`do`	executes a do-file
`assert`	assert verifies that an expression is true. if it is, the command produces no output; if it is not, assert informs you that the "assertion is false".
`append using`	append appends a STATA-format dataset stored on disk to the end of the dataset in memory.
`mvencode` *`varlist`*`,` **`mv (#),`** `[override]`	changes all occurrences of missing to # in the variable listing specified. override specifies the protection provided by mvencode is to be overridden. without this option, mvencode refuses to make the requested change if # is already used in the data.
`mvdecode` *`varlist`*`,` **`mv (#)`**	changes all occurrences of # to missing in the variable list
`egen`	creates a new variable equal to the specified functions and its arguments
`regress` *`depvar varlist`*	regress estimates a model of the dependent variable on variables in *varlist*
`xi: regress ....i.variable`	constructs categorical dummy variables for variables omitting the first category.
`predict` `variable`	stores the predicted values from the regression in *variable*. what this command can do is determined by the previous command.
`probit`	probit estimates maximum-likelihood probit models.
`search`	searches the keyword database. Use search when you are not certain of the command, e.g., search string shows all commands associated with strings.
`tables`	calculates and displays tables of statistics.
`reshape`	converts data from wide to long form and vice versa. 'wide' and 'long' refer to how data are organized. See **reshape** notes below.
`fillin` `varlist`	adds observations with missing data so that all combinations of *varlist* exist, thus rectangularizing the file. the variable _fillin is added to the data. _fillin is 1 for created observations and 0 for previously existing observations.
`(`**`svy`** `commands)`	these are commands prefixed with 'svy' and they pertain to commands used in analyzing survey data.
`tables`	calculates and displays tables of statistics.
`format` `varlist %fmt`	formats numeric variables as follows--number before the decimal indicates the length of the variable, number after the decimal indicates number of decimal places: %#.#g - general numeric format (%5.0g) %#.#f - fixed numeric format (e.g., %5.2f) %#.#e -base 10 power strings are formatted as follows and can be 81 chars long: %#s (e.g., %10s)

Reshape notes: The **reshape** command is particularly useful for files such as that shown in the following example:

Households were asked about the number of livestock owned for three types of livestock coded 330, 331 and 335. To save on data entry time, only those entries reporting any livestock were entered. Missing livestock codes in the file therefore means that the household did not own the livestock associated with the code. The file looks like this.

```
 hh  animcode      num
206       331       70
217       331       65
217       335        8
221       330     1200
221       331      200
```

The above file could have been organized such that each household has only one line of information, and the three animal types appear as three different variables. Such a file would be the wide form of the data. The file as it is organized now is the long form of the data.

The following reshape command converts the file from long to wide form such that each animal code is now a variable, and the file becomes a household-level file.

```
. reshape wide num, i (hh) j (animcode)
. list, nol nod noo

  hh   num330   num331   num335
 206        .       70        .
 217        .       65        8
 221     1200      200        .
```

When followed by this next command, the file is re-converted from wide to long. But note that the file has become rectangularized, that is, the three animal codes now appear for each household.

```
. reshape long num, i (hh) j (animcode)
. list, nol nod noo

 hh   animcode      num
206        330        .
206        331       70
206        335        .
217        330        .
217        331       65
217        335        8
221        330     1200
221        331      200
221        335        .
```

The command **fillin** would have also generated the same rectangularized file as in the preceding example.

<u>Do-file suggested commands</u> to place at the beginning of a do-file to set the parameters before starting to work:

1. Commands in a do-file may be delimited by a carriage return or a semi-colon. To set the semi-colon as the delimiter, the command is:

   #delimit ;

   This command will only work in a do-file. The delimiter cannot be changed from the console.

   If you wish to revert back to the carriage return as the delimiter, the command is:

   #delimit cr

2. The next command will clear the memory:

   clear ;

3. There are several “set” commands that are useful to put at the beginning of the do-file as well.

   set memory 70000; (sets the size of memory)
   set more off ; *(turns the “more” off in the Results window)*
   set matsize 100 ; *(limits number of variables that can be specified in an estimation command)*

ANNEX II

Socio-Economic Survey of Family Sector Farms in the Province of Nampula (Angoche, Monapo e Ribaúe)

July/August 1991
Departamento de Preços e Mercados
Food Security Project

Name of Household Head ____

Household Number ____ HH

Aldeia ____ VIL

Distrito ____ DIST

(Subset of questions from original questionnaire)

I. **HOUSEHOLD CHARACTERÍSTICS**

H1 _______ 1. How many persons are in this household?

H4 _______ 4. Has your family always lived in this village?
1=yes 2=no

H8 _______ 8. Is your family registered as "deslocada"?
1=yes 2=no

H19 ______ 19. Do you presently have lands in fallow?
1=yes 2=no

H21 ______ 21. What is the total area of these fallowed parcels? (hectares)

H24 ______ 24. Do you have lands that you have completely abandoned?
1=yes --> question 25 2=no --> question 27

H25 ______ 25. What is the total area of these abandoned lands? (hectares)

H26 ______ 26. What was the principal motive for abandoning these lands?
1=no security
2=lands lost fertility
3=lack of labor
4=insect attacks
5=other

[**We would like to ask you about the food crops you grow**.]

H29 ______ 29. Over the last five years, have you increased or decreased the amount of land in food crops?
1=increased 2=decreased 3=no change

H31 ______ 31. During a normal year, is your farm production sufficient to feed your entire family?
1=yes 2=no

[**We would like to ask you about the cash crops you grow on your farm?**]

H34 ______ 34. Do your grow any crops that are principally destined for the market?

1=yes 2=no

35. Which crops are grow principally to be sold? (List the three most important)

H35A ____ 1=cotton 4=sunflower
H35B ____ 2=peanuts 5=rice
H35C ____ 3=sesame 6=other

H36 ______ 36. Over the last five years, have you changed the area grown in these cash crops?
1=increased
2=decreased
3=no change

H39 ______ 39. Do you normally grow cotton?
1=yes 2=no

H52 _____ 52. Since your involvement with the cotton companies, have you reduced your area dedicated to food crops, such as maize and manioc?
1=yes 2=no

IV. **PRODUCTION**

H56 ______ 56. Do you have cashew trees?
1=yes 2=no

H57 ______ 57. How many trees do you presently have? (number)

H57A ____ 57A. Of these trees, from how many did you harvest during the last year? (number)

V. AGRICULTURAL SALES

We would like to ask about the marketing of your agricultural products since August of 1990.

64. Over the last five years, have you increased the quantities marketed of the following crops:

H64A ____ a. maize 1=yes 2=no
H64B ____ b. manioc 1=yes 2=no
H64C ____ c. rice 1=yes 2=no
H64D ____ d. cotton 1=yes 2=no
H64E ____ e. peanuts 1=yes 2=no
H64F ____ f. beans 1=yes 2=no
H64G ____ g. sorghum 1=yes 2=no
H64H ____ h. cashew nuts 1=yes 2=no

H65 ______ 65. Compared with five years ago, has the marketing of these products been more difficult or easier?
1=more difficult --> question 66
2=easier --> question 67

H66 ______ 66. If more difficult, why?
1=fewer buyers
2=transportation problems
3=security problems
4=low prices
5=lack of consumer goods
6=other ___________________________

H67 ______ 67. If easier, why?
1=more buyers
2=better transportation
3=better security
4=attractive prices

5=more consumer goods
6=other___________________________

H83 ______ 83. Does your family usually receive traditional gifts or participate in exchange relations?
1=yes 2=no

H84 ______ 84. If yes, how often?
1=only when there is a lack of food
2=only during feasts and rituals
3=frequently

XI. **TYPICAL CONSUMPTION PATTERNS**.

H86 ______ 86. How many meals did these people have yesterday? (Number of meals)

H89 ______ 89. Do you consider these meals adequate to maintain the health of all the household members?
1=yes 2=no

We would also like to ask you about your diet during the hungry period (January to May).

H91 ______ 91. How meals do you customarily prepare daily during hungry period?

H92 ______ 92. In general, are these hungry period meals adequate to maintain the health of all household members?
1=yes 2=no

H96 ______ 96. During the hungry period, was there always food available to purchase from the market or from your neighbors?
1=yes 2=no

I. HOUSEHOLD CHARACTERISTICS

Table IA: Household Characteristics

Name	Family Member Number	This person works on-farm or off-farm 1=yes 2=no	Relation to Head 1=head 2=spouse 3=child 4=parent 5=other kin 6=other	Age	Sex 1=m 2=f	Level of Schooling (enter the last completed year) 0=illiterate 12=post-high school 98=no formal schooling but literate	Marital Status 1=monogamous 2=polygamous 3=single 4=widowed 5=divorced 6=emigrant wife (husband out longer than six months
	MEM	**CA1**	**CA2**	**CA3**	**CA4**	**CA5**	**CA6**
	1		Head				
	2						
	3						
	4						
	5						
	6						
	7						
	8						
	9						
	10						
	11						

IV. PRODUCTION

Table IV: Characteristics of Production

Product		**Quantity harvested**		**Quantity harvested in a normal year**		**Existing stocks at harvest time**		**Month in which last year's stock ran out**	**Amount to be stored from this year's harvest for consumption**		**How long will this year's stocks last?**	**Quantity reserved for seed**	
1=corn 2=beans 3=manteiga beans 4=manioc 5=rice 6=sorghum 7=cotton	8=peanuts 9=cashew nuts 10=cashew drink 11=cane drink 12=coconut 13=coconut drink others	Unit 1=sack 100 2=sack 50 3=kilo 4=liter 5=can 20	Qt	Unit 1=sack 100 2=sack 50 3=kilo 4=liter 5=can 20	Qt	Unit 1=sack 100 2=sack 50 3=kilo 4=liter 5=can 20	Qt	(enter the month)	Unit 1=sack 100 2=sack 50 3=kilo 4=liter 5=can 20	Qt	(enter the month or "all year", if appropriate)	Unit 1=sack 100 2=sack 50 3=kilo 4=liter 5=can 20 other	Qt
PROD		**P1A**	**P1B**	**P2A**	**P2B**	**P3A**	**P3B**	**P4**	**P5A**	**P5B**	**P6**	**P7A**	**P7B**

V. AGRICULTURAL SALES

Table V: Sales of Farm Products

Sale	Crop	Quantity sold		Period of sale	Motive for sale at this time	Buyer	Locale of sale	Distance from the farm	Why sold to this buyer	Value of Sales		Who in the household is responsible for the sale
	1=corn 2=manteiga bean 3=beans 4=manioc 5=rice 6=cotton 7=peanuts 8=cashew nut 9=cashew drink 10=cocos others	Units 1=sack 100 2=sack 50 3=kilo 4=liter 5=can 20	No. of Unit	1= planting (Aug-Dec.) 2= hungry period (Jan-April) 3=this year's harvest 4= various times	1=needed money 2=buyers available 3=consumer goods available 4=attractive price	1=lojista 2=wholesaler 3=AGRICOM 4=ambulante 5=brigada 6=company	1=farmgate/ house 2=village 3=locality 4=district 5=province	(enter the kms between farmer and point of sale)	1=the only one available 2=always sell to this one 3=best price 4=transportation provided 5=carries consumer goods	meticais	Unit 1=unit price 2=total value	1=husband 2=wife
VE	**V1**	**V2A**	**V2B**	**V3**	**V4**	**V5**	**V6**	**V7**	**V8**	**V9A**	**V9B**	**V10**
1												
2												
3												
4												
5												
6												
7												
8												
9												

N.B. Not all of the variables that appear in the printed table are in file c-q5.dta. Only variables **VEN, V2a, V2b, V9a** and **V9b** were kept for this exercise. The **PROD** variable replaces the **V1** variable.