Introducción al manejo de Información en STATA 9.0 Con aplicación a la encuesta de hogares.

Versión 01

Autores:

Wilson Mayorga

Rafael Escalante

PARTE I. INTRODUCCION Y SINTAXIS BASICA

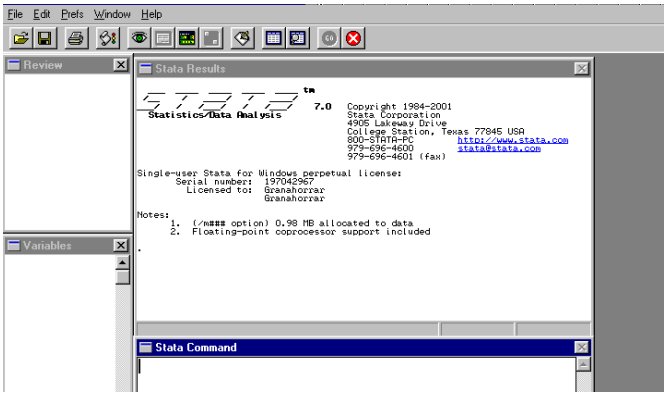
• INTRODUCCION

Stata es un paquete estadístico aplicado a problemas donde los datos son muestras de individuos, empresas, hogares, etc; a diferencia de otros programas más conocidos por los economistas, donde el interés principal es el estudio de datos de series de tiempo.

El diseño de ventanas del programa, permite que sea posible utilizar en ocasiones la interfase gráfica o el Mouse para realizar algunas operaciones, sin embargo, en casos complejos, ***debe escribirse la sintaxis correcta de comandos***. Este requerimiento hace que Stata se defina como un lenguaje de programación.

Las nuevas versiones, posteriores a la versión 8.0 difieren de las versiones previas en que permite realizar todas las operaciones estadísticas mediante menús interactivos, con lo cual el conocimiento exacto de la sintaxis deja de tener cierta relevancia.

La pantalla de inicio de Stata es:



En ocasiones la pantalla será negra, para cambiarla de color se puede tomar la siguiente ruta: Prefs-General Preferences y allí se utiliza el color que más se adecue al gusto del usuario. En

general, este menú PREFS, modifica las opciones de presentación del Programa, y es posible guardar de manera permanente las preferencias del respectivo usuario.

Los otros menús disponibles son:

- File: Abre, Guarda e Imprime los diferentes tipos de archivos disponibles en Stata.
- Edit: Copia y pega tablas de datos
- Data: Permite realizar operaciones entre tablas de datos y matrices
- Graphics: Es el listado completo de todas las operaciones gráficas que permite realizar Stata.
- Statistics: Es el menú principal. Contiene el listado de todas las operaciones estadísticas posibles de realizar en Stata 9.0.
- Window: Activa y desactiva las diferentes ventanas en que se distribuye el programa. Puede funcionar usando la tecla CTRL + un número que va de 1 hasta 8 según la ventana que se desea operar o desplazando utilizando el Mouse.

Las ventanas disponibles en Stata son:

CTRL + 1 Results : Aquí aparecen los resultados del uso de comandos y los mensajes de error, cuando hubieren.

CTRL + 2 Graph : Interfase donde aparecen los gráficos diseñados en el programa.

CTRL + 3 Viewer: Es como una forma de ingresar a la búsqueda de ayuda de comandos.

CTRL + 4 Command : En esta ventana se escriben los comandos que se deseen operar en Stata.

CTRL + 5 Review : En esta ventana aparece un listado de los comandos escritos por el usuario.

CTRL + 6 Variables : Aquí presenta el listado de variables con las que actualmente se está operando(pertenecen a un archivo de trabajo).

CTRL + 7 Data Editor: Es la hoja de ingreso de datos de forma manual.

CTRL + 8 Do File: Es la hoja de ingreso de comandos a un archivo tipo *.do.

Siguiendo en la barra de herramientas de izquierda a derecha observamos nuevas teclas, que

explicamos a continuación:

Permite abrir un *log*, esto es un archivo tipo texto (con extensión) .log) en donde se pueden ir

guardando las salidas que aparecen en el *Stata Results*.

Un editor de *do*, que es donde se pueden realizar *macros* (detalles más adelante).

Son el editor y el *browser*, el primero permite realizar modificaciones en la base de datos, el segundo sólo observar los datos.

El *break*, es una interrupción al programa a la función que esté realizando.

- **CONDICIONES INICIALES**

Debe tenerse en cuenta que ***Stata es un programa Sensible a las mayúsculas***, es decir, para el programa un comando con mayúsculas es diferente a uno en minúsculas. Todo se debe escribir en

minúsculas. Además, en casi todos los casos, Stata funciona con sólo escribir las tres primeras letras del comando respectivo.

Al escribir cada comando en Stata, aparecerá en pantalla Result el mensaje o resultado de respuesta del comando.

clear

Con este comando se borran los resultados y operaciones que de manera temporal se encuentran grabados en el programa

set memory

Permite aumentar la memoria que utiliza el programa dentro las posibilidades del equipo, por defecto Stata inicia su sesión con 5 *Megabytes*, por tanto si se requiere trabajar con más de esa capacidad (como para abrir una gran base de datos), por ejemplo con 10 Megas, se debe escribir: set memory 10m o también set memory 10000.

Un tercer comando, que evita que las salidas sean cortadas por el tamaño vertical de la pantalla es:

set more off

- **RECIBIENDO AYUDA DE STATA**

Si bien la presentación del menú HELP de Stata no es tan amigable como la de otros programas, el usuario cuenta con amplias posibilidades de información que le serán de muchísima utilidad.

HELP-CONTENTS : Presenta a manera de tabla de contenido, el listado de comandos disponibles en Stata. Dando click en alguno de ellos, se puede ingresar a una descripción del comando y algunos ejemplos de su uso.

HELP-STATA COMMAND: Ingresando el comando que genera dudas, se tiene una referencia de su uso y explicación.

HELP-SEARCH: Esta opción puede ser útil cuando se conoce el tema pero no el comando que podría usarse. Por ejemplo. SEARCH: logistic Regression.

También dentro del menú de Help se encuentran otras opciones de ayuda On line. Igualmente en la página www.stata.com se encuentran documentos y Stata Bulletin los cuales pueden ser de mucha utilidad para el usuario.

Escribiendo en la ventana de comandos la palabra HELP seguido del comando que se desea aplicar, genera como salida una explicación del significado, propiedades y descripción completa del procedimiento

Help logistic

- **SINTAXIS BASICA DE LOS COMANDOS**

La ejecución de una orden esta compuesta en general por tres partes: la primera, todo comando inicia con una o dos ***palabras clave*** (table, save, use, etc.); a continuación el usuario tiene que suministrar ciertos ***parámetros***, y por ultimo algunos procedimientos tienen ciertas ***opciones*** suministradas por el software.

Existen tres maneras de operar comandos en STATA:

- Utilizando el mouse y buscando la operación deseada dentro del menú Graphics, Data o Statistics.
- Escribiendo comando por comando en la ventana correspondiente y obteniendo el resultado en cada paso, dando Enter.
- Editando un archivo *.do, el cual es un archivo de texto, donde se escribe un conjunto de comandos. Al operar este archivo, se realizarán simultáneamente las operaciones allí definidas.

Con algunas pocas excepciones, la sintaxis básica de un comando en Stata es:

[by variable indicadora:] comando [lista de variables] [= expresión] [if expresión] [in rango] [weight] [, opciones específicas]

Donde entre brackets [] se presentan aquellas cosas que pueden ser opcionales al comando. Las palabras en inglés son de alguna manera “fijas”.

Lista de variables: Se refiere al nombre de la variable a la que se aplicará el comando. En caso que no aparezca se aplicará a todo el conjunto de datos.

Los componentes opcionales son:

- By lista de variables: Indica que el comando se repetirá para cada subconjunto de datos para los cuales los valores de la variable indicadora definida sean iguales. Para que funcione***, los datos deben ser primero ordenados por la variable indicadora.***

Ejemplo: Suponga que se tiene una variable llamada sexo, donde 1 es hombre y cero mujer. En ese caso es posible aplicar un comando que genere resultados para los hombres y para las mujeres así:

By sexo: comando [lista de variables]

- If Expresión: Esta opción restringe el comando a aquellas observaciones para los cuales la expresión condicional es verdadera.

Por ejemplo, si sólo se desea generar un resultado para los hombres, se tendría el siguiente comando:

Comando [lista de variables] if sexo==1

- In rango: Esta opción, restringe el alcance del comando a un rango de observaciones específico. La especificación de ese rango toma la forma numérica, así límite inferior[/límite superior], siendo estos límites posiciones.

Por ejemplo, se desea aplicar un comando entre el dato en la posición 5 y la posición 25 inclusive, la sintaxis será:

Comando [lista de variables] in 5/25

- = Expresión: Se usa cuando se desea dar algún valor específico al resultado. Se usa cuando se desea crear una nueva variable o reemplazar una variable(comandos generate o replace).

Generate suma = variable1 + variable2

- Weight: Se usa cuando se desea ponderar por alguna variable el resultado generado por un comando. Es útil por ejemplo, para calcular de manera rápida un promedio ponderado.

Summarize variable1 [weight=variable2]

- Opciones específicas: Cada comando, dependiendo de la operación que realice, tiene opciones propias. Estas se analizan en el momento en que cada comando sea estudiado. En el caso que una opción tenga más de un argumento, estos se separan por comas(,).

- **TIPOS DE ARCHIVOS**

Stata maneja varios tipos de archivos, de acuerdo a su función dentro del programa. Es así como existen archivos de comandos(*.do), archivos de datos(*.dta), gráficos(*.gph) y archivos de output o salida de resultados(*.log). Estos son los básicos en el manejo de Stata.

Otros archivos de gran utilidad son los archivos de programación como *.ado, y macros diseñadas por el usuario, los cuales serán analizados más adelante.

PARTE II. ADMINISTRACION DE DATOS

En general, todas las operaciones es posible realizarlas en Stata 9.0 mediante el uso del menú. La importación o exportación de datos es posible mediante la ruta file-import. La administración de datos que se explicará aquí mediante comandos, es posible realizarla utilizando el menú data-combine datasets.

En Stata las observaciones son numeradas secuencialmente desde 1 hasta N. En el formato de número de Stata no existen las comas; es decir cinco mil será 5000 y el punto determinará decimales, por ejemplo 5.3. También pueden escribirse números negativos(-5.3) o números en notación científica(5e+3).

En el caso que existan missing values, serán denotadas por un punto(.) dentro del archivo de datos, y para Stata siempre el missing value será mayor que cualquier número; sin embargo, al realizar algún cálculo estadístico, el programa obviará estos missing values.

Los números pueden ser guardados en uno de cinco tipos: byte, int, long, float(es el default) o double. La discriminación depende del número de bytes de memoria necesarios para guardar cada observación(es así como byte requiere un byte, int, que es diferente de integer, requiere 2 bytes así sucesivamente).

Las variables alfanuméricas son denominadas string y deben ir dentro de comillas. En Stata son denominadas str1, str2,…

En cuanto al formato de visualización, es decir, la forma en que los datos son presentados, es el siguiente:

Para variables númericas, viene expresado como %w.d seguido de uno de estos tres formatos: e, f, g. Con w denotamos un número entero que especifica la anchura del formato, mientras que d india el número de dígitos que siguen al punto decimal. Para variables alfanuméricas, el formato es %ws, donde s indica que es string y w es un número entero que indica la anchura dada a la variable. Por defecto, el formato de cada variable es:

Byte	%9.0g
Int	%9.0g
long	%12.0g
float	%9.0g
double	%10.0g
Str#	%#s(o %9s si la anchura es menor de 9 caracteres)

Es posible cambiar el formato de visualización de las variables con el comando format. Por ejemplo, supongamos que una variable es del tipo float y se desea que tenga formato %10.2g, la sintaxis será:

format variable %10.2g

- **CREACION DE VARIABLES Y ARCHIVOS DE DATOS**

Los archivos de datos tienen extensión *.dta. A su vez, están formados por variables. Cuando se nombra una variable dentro del programa Stata, la misma debe tener como máximo ocho

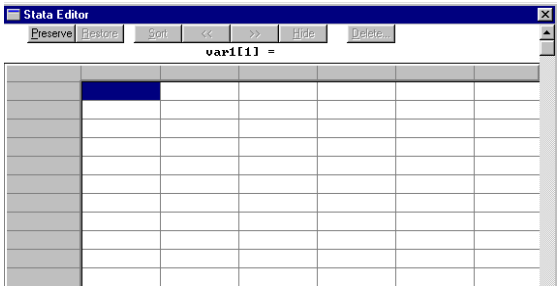
caracteres (pueden ser letras o números y debe empezar siempre con una letra). Las variables pueden ser: numéricas, alfabéticas, de fecha o de tiempo.

El ingreso de Datos a Stata puede realizarse directamente utilizando el EDITOR o mediante lectura de archivos externos(ASCII), como Lotus o Excel o similares.

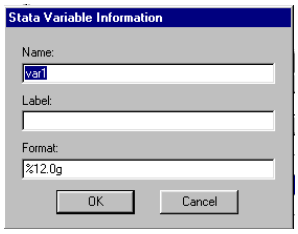
- ***Ingreso Manual de Datos***

En la barra de comandos se da click en el ***Data Editor*** con lo cual aparece una hoja similar a Excel, donde se digitan los datos correspondientes, siendo cada columna una nueva variable, o se copian y pegan desde algún otro archivo(por ejemplo Excel o Word).

En la parte superior de la ventana aparecen los nombres de las variables, encabezando cada una de las columnas en las que se introducirán sus valores.



Una vez digitados los valores de cada celda (números, nombres o frases), se procede a definir las variables: nombre, el formato y la etiqueta. Haciendo doble click en cualquier celda, nos muestra un cuadro de diálogo ofrecido para definir las características de la variable.



Una vez ingresados los nombres, se puede cerrar el editor y el programa preguntará si se desean mantener estos nuevos cambios. Aquí se dará click en OK y las nuevas variables aparecerán en la ventana de DATA EDITOR.

Esta operación genera un archivo temporal, apenas terminaremos la sesión el desaparece perdiendo así la información. Para guardar la base de datos como archivo permanente con extensión *.dta, se elige FILE-SAVE AS donde se abrirá un cuadro de diálogo que permitirá especificar tanto la ruta de acceso como el nombre del archivo de datos que se desea grabar.

- ***Ingreso de Datos Externos***

Stata importa datos provenientes de archivos externos de tipo ASCII (conocidos también como archivos planos). Existen 3 comandos disponibles para realizar importación de este tipo de datos: infix, infile e insheet.

En estos archivos la información puede venir de diversas formas:

1. ***En columnas***: De esta manera se conoce la posición donde se encuentra el valor de cada uno de los ítems.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
7	9	8	3	8	7	6	2	W	I	L	S	O	N	M	A	Y	1	2	6	1	7	7	.	2	1	7	0	.	2

En la tabla anterior hay seis variables: id, nombre, sexo, edad, altura y peso. Los datos se encuentran en formato de columna, y todas las observaciones de cada variable ocupan el mismo número de posiciones. Por ejemplo la variable id empieza en la columna 1 y termina en la columna 8.

Cuando los archivos provienen de un archivo externo tipo ASCII, con estas características, se le debe indicar a Stata con la comandos ***infix*** y luego se le debe dar la forma que debe leer las base, teniendo en cuenta el siguiente orden para definir una variable:

formato nombre posición inicial - posición final

La línea de comandos para este conjunto de datos es de la siguiente forma:

```
infix str8 id 1-8  str9 nombre 9-17 byte sexo 18 byte edad 19-20  float altura 21-26  float peso 27-
30  using " c:\stata\ejemplo.raw "          ( Enter)
```

Cuando se utiliza este formato, las posiciones de las variables deben ser fijas; es decir, en ***infix*** se indica que la variable nombre está en la columna 9 a 17, los datos para esa variable deben estar siempre entre esas posiciones.

Este tipo de formato permite leer todo o parte del registro. En este ejemplo se lee todo el registro. Sin embargo, es posible leer sólo algunas variables, por ejemplo, si quisiera leer sólo el nombre y la edad, la sintaxis sería:

```
infix str8 id 1-9  str9 nombre 9-17  byte edad 19-20  using "c:\stata\ejemplo.raw "
```

El comando ***infix*** espera la ruta de ubicación del archivo ASCII, por lo tanto es necesario ubicarlo de donde debe tomar el archivo, por lo tanto la línea de comandos viene acompañada de otra palabra clave ***using***.

2. ***Forma de lista***: Se listan las variables en el orden en el cual aparecen en el registro de entrada. Las siguientes consideraciones se deben tener en cuenta para esta forma de ordenamiento de los datos:

- Las variables en el archivo por leer deben estar en el orden en el que aparecen en la línea de comandos.
- Los valores de las variables deben estar separados por alguno de las siguientes alternativas: espacio en blanco, por tabuladores o por comas.
- En los valores para las variables alfanuméricas no se permiten espacios en intermedios. La longitud por defecto para esta clase de variables es de ocho dígitos.
- Los espacios en blanco causan que los nombres de las variables y sus valores se desfasen. Los valores faltantes en este tipo de formato se indican con un punto (.).

Continuando con el ejemplo anterior, pero ahora se tiene la información en forma de lista:

```
79838762    WILSON MAY          1   26   177.2    70.2
```

La línea de comandos para este conjunto de datos es de la siguiente manera:

```
infile str8  id str9 nombre sexo edad  altura  peso  using  "c:\stata\ejemplo.raw "
```

En el uso del comando ***infile*** para variables alfanuméricas tiene que ser definido el formato, por ejemplo, en el ejercicio anterior la variable *nombre* debe ir antecedido por el formato a utilizar, es decir ***str9 nombre.***

Además, en el uso del comando infile se supone que no existen espacios entre las palabras; por ejemplo si el nombre fuese Juan Carlos, debería incluirse como Juan_carlos.

Si la separación es por comas, tenemos la alternativa de utilizar dos comandos, ***infile*** y ***insheet*** La línea de comandos para este conjunto de datos es de la siguiente manera:

insheet id nombre sexo edad altura peso using "c:\stata\ejemplo.raw"

Este comando carga en memoria temporal archivos de datos que están en formato separados por comas, espacios o tabuladores(archivos planos o *.txt y similares). Su sintaxis es:

Insheet lista de variables using nombre del archivo , opciones.

Las opciones son:

Names = indica si en la primera fila están los nombres de las variables

Comma = indica si las variables están separadas por comas.

Tab = Indica si las variables están separadas por tabulador

Clear = especifica que se desea quitar de la memoria los datos anteriores y dejar estos.

insheet ejemplo1 ejemplo2 using "c:\data\ejemplo.raw", names

Si las columnas en los datos no tienen nombre, entonces se usa nonames.

Obsérvese que utilizando este comando no fue necesario definir los formatos de las variables alfanuméricas, además no es necesario agregar el simbolo (_), para separar el nombre de Juan_carlos.

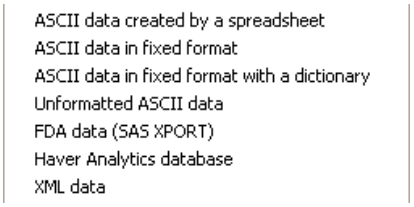
Si se usan tabuladores utilizaremos ***insheet o infile*** de la misma manera que el ejemplo anterior. Suponga estos datos

Wilson	23	45
Andres	56	78
Julian	34	44

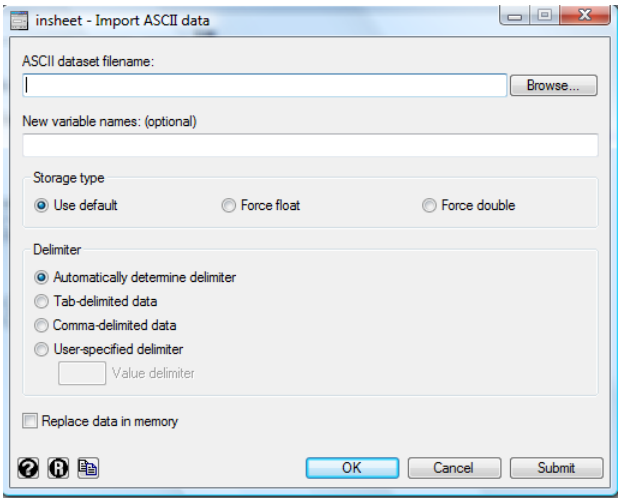
Se pueden ingresar a STATA, suponiendo que están separados por tabulador y sin nombres de la siguiente manera:

```
insheet using “c:\data\ejemplo2.txt”, nonames tab
```

Utilizando el menu interactivo del programa es posible realizar las operaciones anteriores, mediante la ruta FILE – IMPORT, la cual despliega el siguiente conjunto de opciones:



Al revisar la pantalla del comando Insheet (primera opción del menú) aparecen en su orden, las opciones para cargar los datos externos que se van a importar, luego el nombre de las variables (el cual es opcional) y finalmente el tipo de delimitador que separa los datos.



Por su parte, la pantalla del comando infix muestra como elementos fundamentales la opción de uso de un "archivo diccionario" o la opción de especificación. Como se mencionó previamente, para usar el archivo infix el conjunto de datos no deben tener ningún separador y el usuario debe conocer el número de posiciones (longitud) con la que se asocia cada variable. Si se conoce dicho número de posiciones es posible digitar la información en la opción de "specifications" tal como lo muestra el ejemplo.

Cuando el número de variables es notorio, se sugiere crear un archivo diccionario. Este archivo es uno tipo especial de archivo de stata con extensión dht, el cual tiene la información de cada variable y su longitud. Este tipo de archivos se crea en la ventana de archivos do, la cual se explicará más adelante. La sintaxis básica de este archivo es:

Infix dictionary {

Variable1 posicion_inicio posición_final

Variable2 posicion_inicio posición_final

}

Siguiendo con el ejemplo anterior, tendremos:

Infix dictionary {

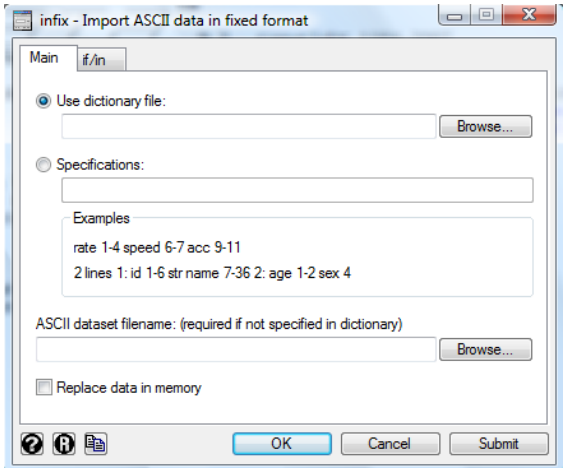
id 1-9

nombre 9-17

edad 19-20

}

Este archivo se guarda con extensión dht y se ingresa en la posición que lo exige la ventana del comando infix, para así realizar el proceso de importación de los archivos planos.

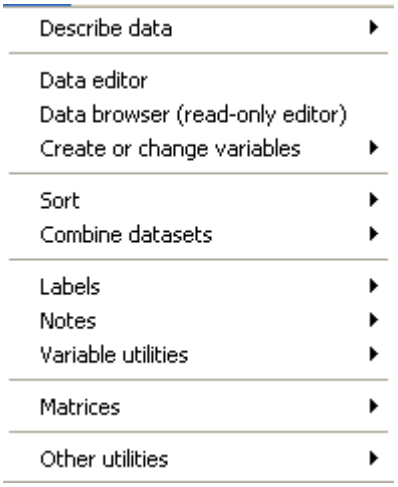


- **OPERACIONES ENTRE ARCHIVOS STATA**

Una vez importados o creados los datos, es posible operar con ellos. La nueva versión de Stata permite realizar estas operaciones bien sea por comandos o mediante la elección de las opciones

correspondientes con el Mouse en el menú DATA-COMBINE DATASETS. A continuación se explicará el uso de comandos.

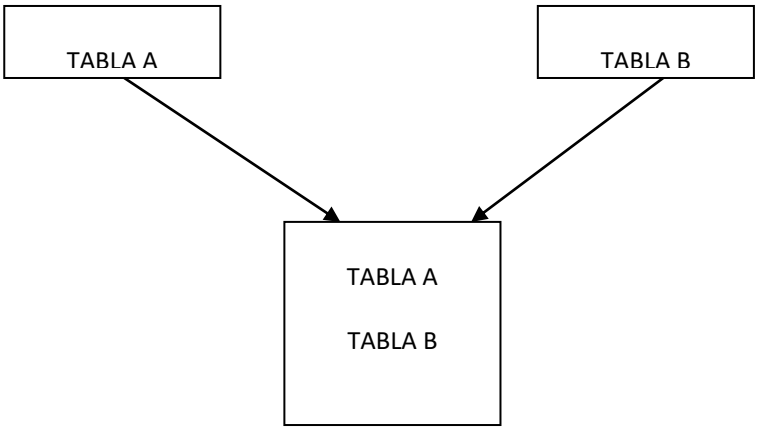
En general, todos los comandos disponibles para realizar operaciones dentro la base de datos disponible en memoria se pueden encontrar siguiendo la ruta FILE-DATA. Aquí se encuentra el siguiente conjunto de opciones:



El primer grupo de opciones, agrupadas dentro del menú COMBINE DATASETS, permiten unir varias tablas de datos en una sola. Aquí se analizan 3 opciones: Unión Vertical(APPEND), Unión Horizontal(MERGE) y Selección de Datos(JOINBY).

1. CONCATENAR VERTICALMENTE

Cuando se van a unir archivos que tienen variables comunes de diferentes individuos, esto se conoce como concatenar vertical. De la siguiente Manera:



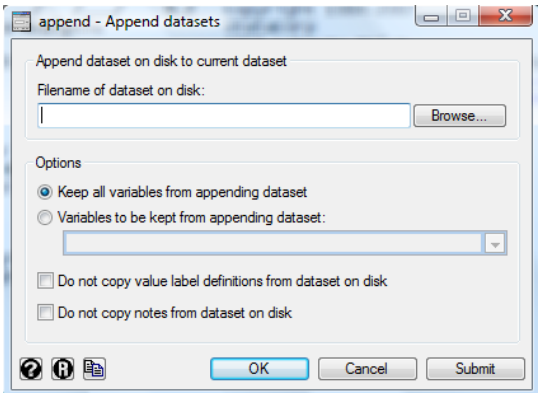
Supongamos que se tienen dos bases de datos. Para generar el nuevo archivo que consiste en la unión de database1.dta y database2.dta, tenemos que seguir los siguientes pasos.

1. Tener activo alguno de ellos, por ejemplo database1.dta
2. Usar el comando append de la siguiente forma:

```
append using "c:\stata\database2.dta"
```

Para mantener los cambios debe guardarse esta nueva base de datos utilizando el comando save o mediante el menú FILE-SAVE AS.

Para realizar esta operación mediante la interfase gráfica de STATA se sigue la ruta data-combined datasets –append datasets, mediante lo cual aparecerá la siguiente ventana:



La opción "Filename dataset on disk" corresponde a la Tabla B del gráfico anterior, en tanto las opciones siguientes permite mantener todas las variables de dicha tabla en el resultado final de la importación o apenas el listado seleccionado con la opción "variables to be kept from appending dataset".

2. CONCATENAR HORIZONTALMENTE

Este caso corresponde a unir valores de diferentes variables correspondientes a los mismos individuos. Se utiliza el comando ***merge*** y adicionalmente se requiere una variable común o de enlace (Variable llave). Los archivos que se concatenan tienen que estar ordenados por las variables de enlace.

Las siguientes líneas de comandos concatena horizontalmente los dos archivos anterior utilizando como variable de enlace ***id*** .

Recuerde que alguna de las bases tiene que estar activa en el programa.

- Abrir el archivo database3.dta

```
use "C:\stata\database3.dta", clear
```

- Ordenar este archivo teniendo en cuenta la variable enlace(variable llave)

```
sort variable_llave
```

- Realizar la concatenación.

```
merge id using "C:\stata\database4.dta"
```

En el caso que los archivos concatenados no tengan exactamente el mismo numero de observaciones, se generarán missing values en las posiciones donde falte información.

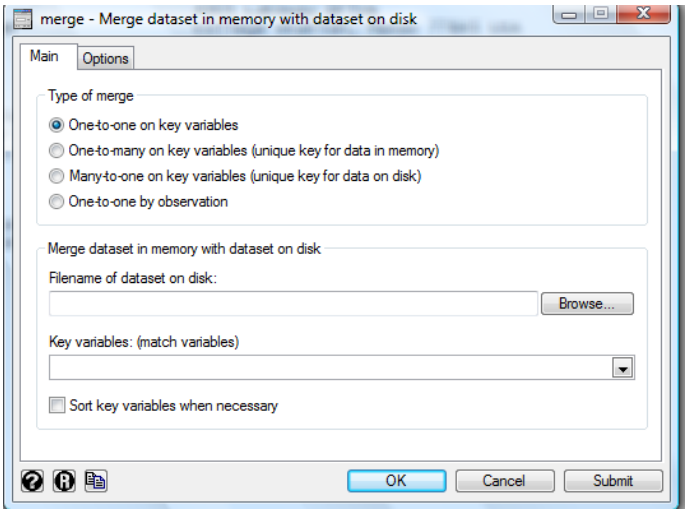
El procedimiento genera una nueva variable (***_merge***), que aporta la información sobre lo realizado, utilizando los siguientes códigos:

_merge=1 Corresponde a los individuos (según la variable enlace) que están contenidos en la base activa y no están contenidos en la otra base.

_merge=2 Individuos contenidos en la base dos no contenidos en la base activa

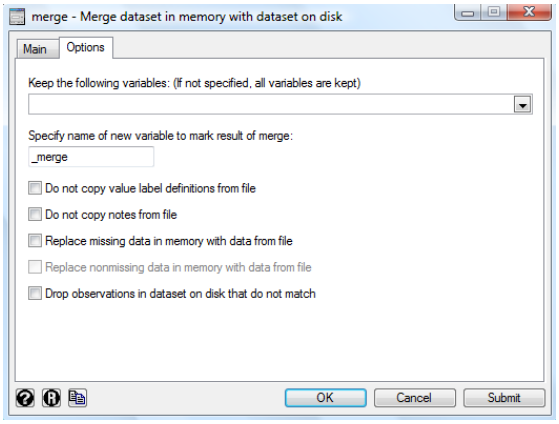
_merge=3 Individuos comunes en las dos bases.

La ruta para realizar esta operación siguiendo la interfase gráfica es data-combined datasets-merge two datasets. También existe la posibilidad de realizar el pegado simultáneamente de varias tablas.



El primer grupo de opciones marcadas como “one to one on key variables”, “one to many…” , many to one…” “one to one…” indican si se desea pegar cada observación de una fila de una tabla a otra fila (unicamente una fila) de la otra tabla, o al contrario, si lo que se desea es de una tabla replicar la información para varias filas de la otra tabla.

Este comando incluye una pestaña de opciones, la cual permite elegir, entre otras cosas cuáles variables se desea incluir dentro de la tabla final de resultados del proceso de pegado, así como definir el nombre de la variable que va a detallar el resultado del proceso para cada fila de la tabla de resultado (si la correspondiente fila pertenecía a ambos archivos base o sólo a uno de ellos).



3. INCLUSION DE INFORMACION ADICIONAL

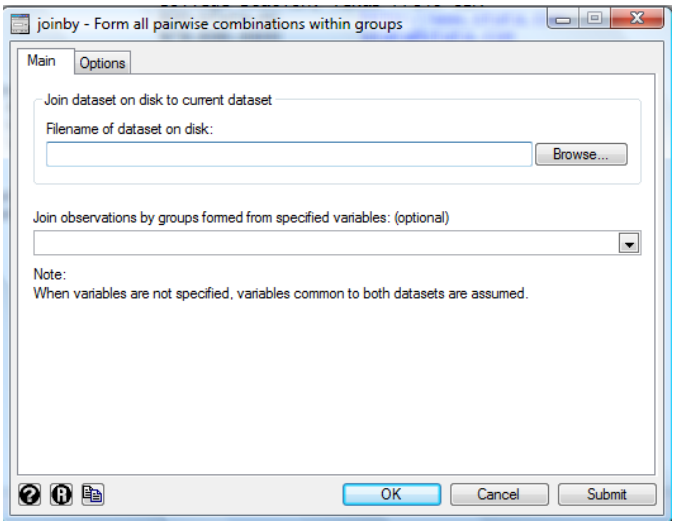
Supongamos que se tiene una tabla llamada personas y otra llamada ocupados. La primera tabla tiene más individuos que la segunda, pero se desea tomar de la tabla de personas únicamente la información relacionada para los individuos que pertenezcan al grupo de ocupados; es decir la tabla final debe tener tantas filas como el número de la tabla ocupados y tantas columnas como la suma de las dos tablas. Este proceso se realiza mediante el comando joinby.

El procedimiento necesita variables enlace (Variables llaves), es decir, variables comunes entre los dos archivos. Los archivos que se concatenan tienen que estar ordenados por las variables de enlace.

Las siguientes instrucciones realizan la operación propuesta, donde la variable de enlace es ***sexo***.

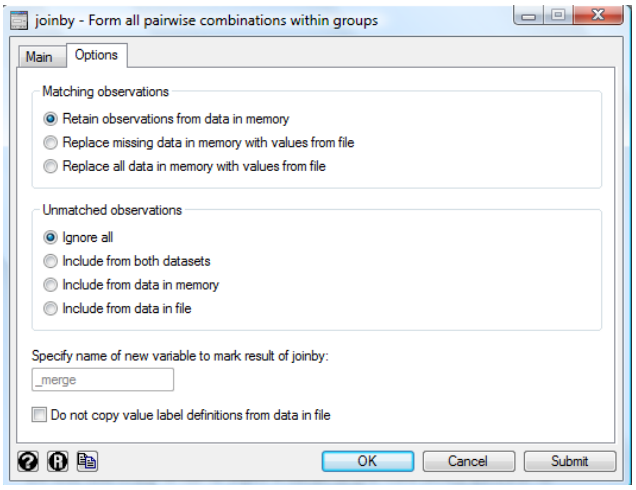
- Abrir el archivo Stata *de personas.dta*
- Ordenar el archivo activo teniendo en cuenta la variable de enlace.
- Grabar el archivo *database1.dta* con la modificación realizada en el segundo paso.
- Abrir la segunda base, ocupados*.dta*.
- Ordenar la base activa por la variable llave o de enlace
- Unir la información de la tabla personas a la tabla ocupados, dejando únicamente la información de los individuos que coincidan entre las dos tablas.

La forma más sencilla de realizar este proceso (una vez las tablas de datos se encuentren ordenadas según las variables llave) es utilizando el menú data y específicamente la opción form all pairwise combinatios. Al seguir esta ruta se abrirá la siguiente ventana:



En esta primera pestaña se debe ingresar (utilizando el botón Browse) la ruta donde se encuentra el archivo que se va a pegar (la tabla B, siendo la tabla la que se encuentra activa en memoria) y posteriormente, se ingresa el listado de variables llave que indican la relación entre las tablas (puede ser una o varias de ellas).

Es fundamental, antes de continuar con el proceso, revisar las opciones de la ventana Options, ya que éstas determinarán el tipo de pegado a realizar. En esta nueva pestaña aparecerán las siguientes alternativas:



El primer grupo de opciones corresponde al tipo de operación que se realizará con aquellas observaciones que correspondan a individuos que pertenezcan tanto a la tabla A como a la tabla B. Las opciones son mantener solos las observaciones de la tabla en memoria(la que se encuentra abierta en STATA) o reemplazar los datos de una tabla por los de la otra.

El segundo grupo de opciones, corresponde a la operación que se realizará con los datos que aparezcan en una tabla pero que no aparezcan en otra. Siguiendo con el ejemplo que inició esta sección, donde teníamos una tabla de personas con más filas que la otra tabla llamada ocupados, y si deseamos dejar únicamente tantas filas como la tabla de ocupados pero con más información por columnas; debemos utilizar este grupo de opciones.

Si se elige la opción “ignore all”, la tabla final dejará únicamente la información de los individuos (filas) que se encuentren exactamente en ambas tablas. Las demás opciones permiten dejar toda la información por columnas de los individuos de una tabla o de la otra o de ambas. Es necesario tener claro que la tabla llamada “in memory” corresponde a aquella que se encuentre activa en STATA, mientras la tabla llamada “in file” corresponde a aquella tabla que se está pegando que se encuentra guardada en el disco duro.

4. DISEÑO DE FILTROS: COMANDOS KEEP y DROP

Estas instrucciones se utilizan cuando se va a trabajar sólo con una parte de un archivo al seleccionar un grupo de variables u observaciones. Estos comandos restringen toda la base de datos, luego no es posible discriminar filtros para variables en particular.

En caso que se desee aplicar el menú, la ruta es data-variable utilities-eliminate variables or observations.

El comando ***keep*** mantiene en el archivo, sólo las variables que son listadas junto al comando, borrando el resto.

keep [lista de variable]

keep sexo altura

Para restringir por observaciones se puede utilizar In rango o if expresión, como se analizó anteriormente:

keep in 1/3

keep if sexo==1 & var3>20

El comando ***drop*** elimina del archivo activo, las variables que son listadas en la comandos.

drop [lista de variable]

drop sexo altura

Igualmente, es posible utilizar el comando ***drop*** junto a las condiciones de rango y de expresión:

drop in rango

drop if condición

En algunas ocasiones, el interés no es restringir la base de datos, sino simplemente realizar algún análisis estadístico puntual sobre una parte de ella. En esos casos, es útil utilizar las opciones in rango, if condición o incluso by [lista de variables], de esta manera no se afecta la extensión de la base de datos.

comando [lista de variables] if variable1> 25 in 12/75

Con la sintaxis anterior, se restringe a hacer alguna operación en el rango de la 12ava observación hasta la 75ava y además si alguna variable es mayor a 25.

5. OTRAS OPERACIONES RELACIONADAS CON DATOS

- ***REEMPLAZAR VARIABLES***

En este caso se usa el comando replace

Replace var1=1.5 in 1/10

- ***USO DE DATOS YA CREADOS***

Para usar tablas de datos ya disponibles se utiliza el comando use

Use “c:\nombre tabla”, replace

Otra forma es usar el menú y allí dar File Open.

- ***CREANDO ARCHIVOS LOG***

Un archivo *.log guarda los resultados que aparezcan en la ventana de STATA RESULTS. Para crearlo se debe seguir la siguiente sintaxis:

Log using “c:\nombre archivo.log”

Si ya se creó este archivo en un ejercicio anterior y se desea escribir sobre él, entonces se adiciona luego de una coma el comando append.

Si lo que se desea es SOBREESCRIBIR no se coloca append sino replace. Para dejar de grabar se escribe el comando log close

Para detener de forma temporal la grabación se usa log off; y para reiniciar la grabación se usa log on. Estos dos comandos son útiles para evitar llenar el archivo de muchos pasos o muchas salidas. Para abrir archivo *.log, puede usarse la siguiente ruta:

File-log-view

Esta ruta abrirá el archivo log y podrán verse los resultados guardados.

5. FUNCIONES Y EXPRESIONES

Antes ya vimos algunos usos de expresiones. Un listado más detallado aparece a continuación:

+ , - , /, ^(elevado), <, <=, >=, ==, ~=(diferente), ~(diferente),|(o), &(y)

Este conjunto de expresiones se utilizan ampliamente en el proceso de generar nuevas variables, por ejemplo variables dummy.

Ejemplo:

list sexo if((social_c~=8 & social_c~=9) & edad<23 & civil==0)

list if ingreso>1000 | ingreso>5000 & edad<25

En el caso de caracteres alfanuméricos se escriben dentro de comillas.

If pais ~= “COL”.

En el caso que se deseen generar nuevas variables, bien sea utilizando los anteriores operadores o funciones, se debe anteponer SIEMPRE el comando gen o generate.

Generate var4 = var1 + var2

Generate var5 = var1/var2 if dum==1

En el caso que no se cumpla la condición se generará un missing value(.).

También es posible usar este comando antecediendo el separador bysort:

bysort: sexo generate variable 4 = variable4*5;

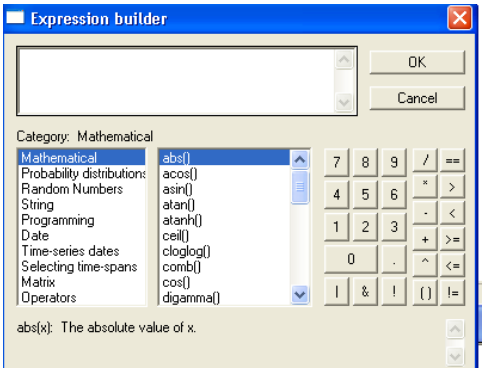
El conjunto de opciones de generar nuevas variables se encuentra en la ruta FILE-DATA-CREATE OR CHANGE VARIABLES. En particular, la opción gen aparecerá en CREATE NEW VARIABLES.

- ***FUNCIONES***

Para operar funciones, es decir, para crear nuevas variables utilizando funciones, se debe anteponer el comando generate, con la sintaxis:

gen nombre = comando(inputs).

En caso que se desee utilizar el menú para la generación de variables, es posible seguir la ruta data-create or change variables. En cada operación es posible tener acceso a la calculadora de stata que trae el listado de funciones y operaciones disponibles con gen y egen, así como a todos los símbolos que se desee utilizar, dando clic en create(…).



- ***VARIABLES INDICADORAS Y VARIABLES CATEGORICAS***

Una variable Indicadora o Dummy asigna 1 o 0 según se cumpla o no cierta característica. Una variable categórica divide a la muestra en grupos dependiendo de cierta característica (por ejemplo mucho, poco, nada). Toda variable indicadora es categórica, pero lo contrario no es cierto.

STATA permite convertir variables continuas en categóricas o indicadoras, y convertir variables categóricas en variables indicadoras. A continuación veremos la estructura básica para generar dichas variables.

a. Construcción de Variable indicadora a partir de variable continua:

```
Generate d1 = (variable>25)
```

D1 tomará el valor de 1 si la condición es cierta y cero si es falsa. Si deseamos no tener en cuenta los missing values en el anterior ejercicio hacemos:

```
Gen d1 = (variable>25) if variable~=.
```

Una forma "indirecta" de crear la anterior variable hubiera sido la siguiente:

```
Gen d1 = 1 if variable>25 & variable ~=.

Replace d1=0 if variable<25
```

b. Construcción de Variable Categórica a partir de Variable Continua

Vamos a generar una variable d2 cuyos valores estén en función de los valores de var1 de la siguiente forma: d2 vale 0 si var1<=1.5, vale 1 si var1 está entre 1.5 y 3.0 y vale dos si es mayor o igual a 3.0.

```
Gen d2 = 0 if var1<=1.5

Replace d2 =1 if var1>1.5 & var1<=3.0

Replace d2=2 if var1>3.0 & var1~=.
```

PARTE III. GENERACION DE REPORTES

El conjunto de opciones disponibles para crear tablas de resumen, mediante las cuales se puedan presentar de manera descriptiva los datos disponibles en memoria se halla en STATISTICS-SUMMARIES, TABLES, TEST – TABLES.

El diseño de tablas en stata puede ser de una vía o de dos vías. En el primer caso se tiene una variable categórica por filas y en cada columna la variable continua a la cual se desee calcular estadísticas de resumen. En el caso de tablas a dos vías(two way) se deben tener dos variables categórica que se van a relacionar: Una por filas y otra por columnas.

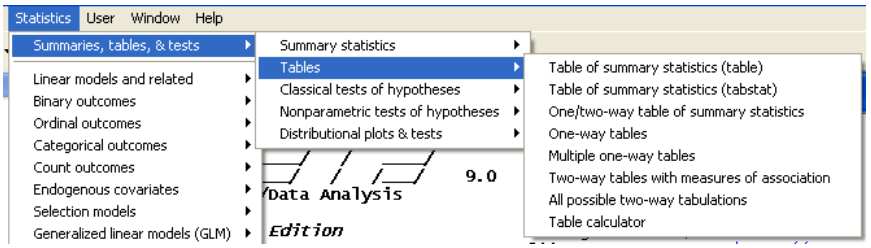
Los comandos más importantes para presentar una rápida revisión de los datos(además de los presentados en la sección de administración de datos) son los siguientes:

Table

Summarize

Tabulate

Tabstat



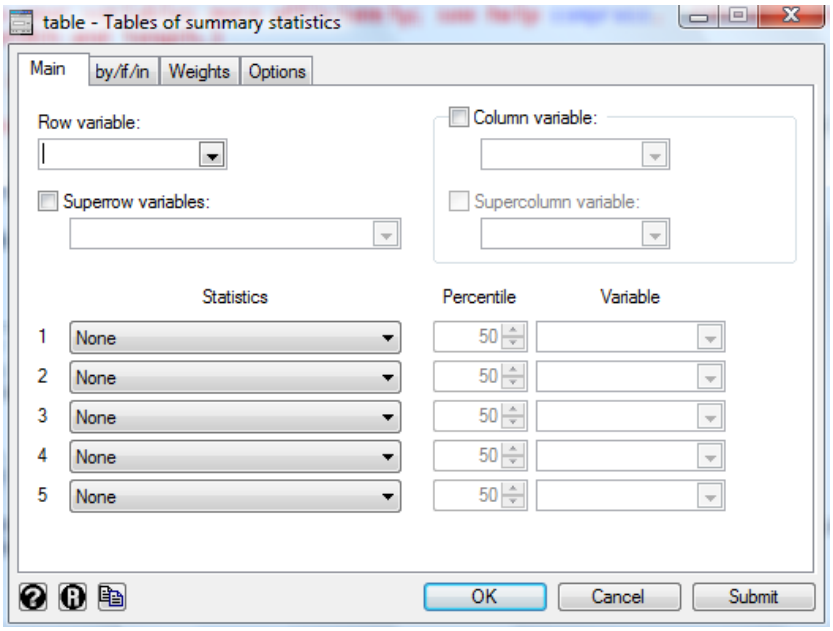
- ***COMANDO TABLE***

El primer comando, table, presenta estadísticas básicas en una tabla. Su sintaxis básica es:

Table variablefila variablecolumna if expresión in rango , contents(posibilidades) by(categórica)

Este comando permite realizar las siguientes operaciones:

- Freq = frecuencia(default)
- Mean variable
- Sd variable
- Sum variable
- Count variable
- Max variable
- Min variable
- Median variable
- P# varname = este es el percentil, por ejemplo p95.



En principio, todos los comandos para generación de tablas o reportes funcionan de manera similar. Una tabla consistirá en información calculada según las alternativas anteriores, desagregando en variables llamadas fila(row variable) o columna(column variable). Este tipo de variables corresponden a categorías para las cuales se deseen realizar los cálculos. Ejemplos de variables categóricas que servirán como variables fila o columna serán el sexo, el nivel educativo, el rango de edad (definida como variable categórica) entre otras.

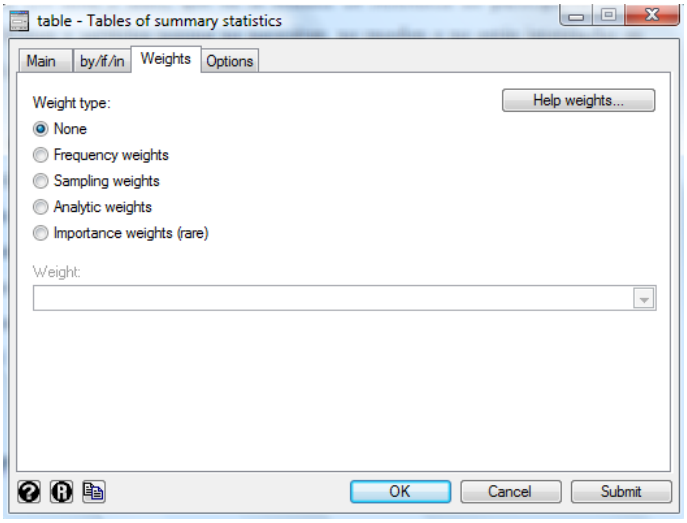
Las variables superfila o supercolumna corresponden a divisiones o categorías mayores en las cuales se desee agrupar la información.

La segunda pestaña de los comandos de tabla corresponden al grupo de opciones by/if/in. La primera alternativa (by) permitirá repetir la tabla para diversos segmentos de población. Por ejemplo, si lo que se desea es realizar el mismo cálculo para todas las ciudades de Colombia, se deberá usar esta opción “by” seguida de la variable que indica la ciudad.

La opción IF, como se mencionó previamente, corresponde a la generación de filtros, es decir, la generación de un reporte para una categoría particular. Por ejemplo, si del total de información sólo se desea generar un reporte para la ciudad de Bogotá, se utilizará “if” seguido de la condicion que la variable ciudad tome los valores únicamente asociados a la ciudad de Bogotá.

La pestaña weight, permitirá elegir el FACTOR DE EXPANSION. Toda encuesta es una muestra aleatoria de alguna población con características similares, esto quiere decir que en las encuestas no se entrevistan a todos los hogares ni personas (para el caso de la Encuesta de Hogares). Por esta razón toma gran importancia la variable dentro de la base de datos llamada factor de expansión. El factor de expansión, se define como el inverso de la probabilidad de selección, en los casos de inclusión forzosa como el censo de población, el factor de expansión toma valor de uno. En otras palabras es la estimación del número de individuos poblacionales que cada individuo entrevistado representa.

Si no se incluye esta variable de expansión, el reporte corresponderá al análisis de las observaciones que estrictamente se incluyen dentro de la base de datos. Al expandir este resultado muestral se tendrá una estimación del comportamiento poblacional.



La mejor explicación posible de los diferentes tipos de factor de expansión disponibles en STATA se encuentra siguiendo la ruta help-stata command y buscando por weight.

Finalmente, la pestaña de opciones, incluye las diferentes opciones de presentación del reporte, tales como qué tipo de total de filas o de columnas se desea reportar o si se desea eliminar algún tipo de celda de la tabla reporte.

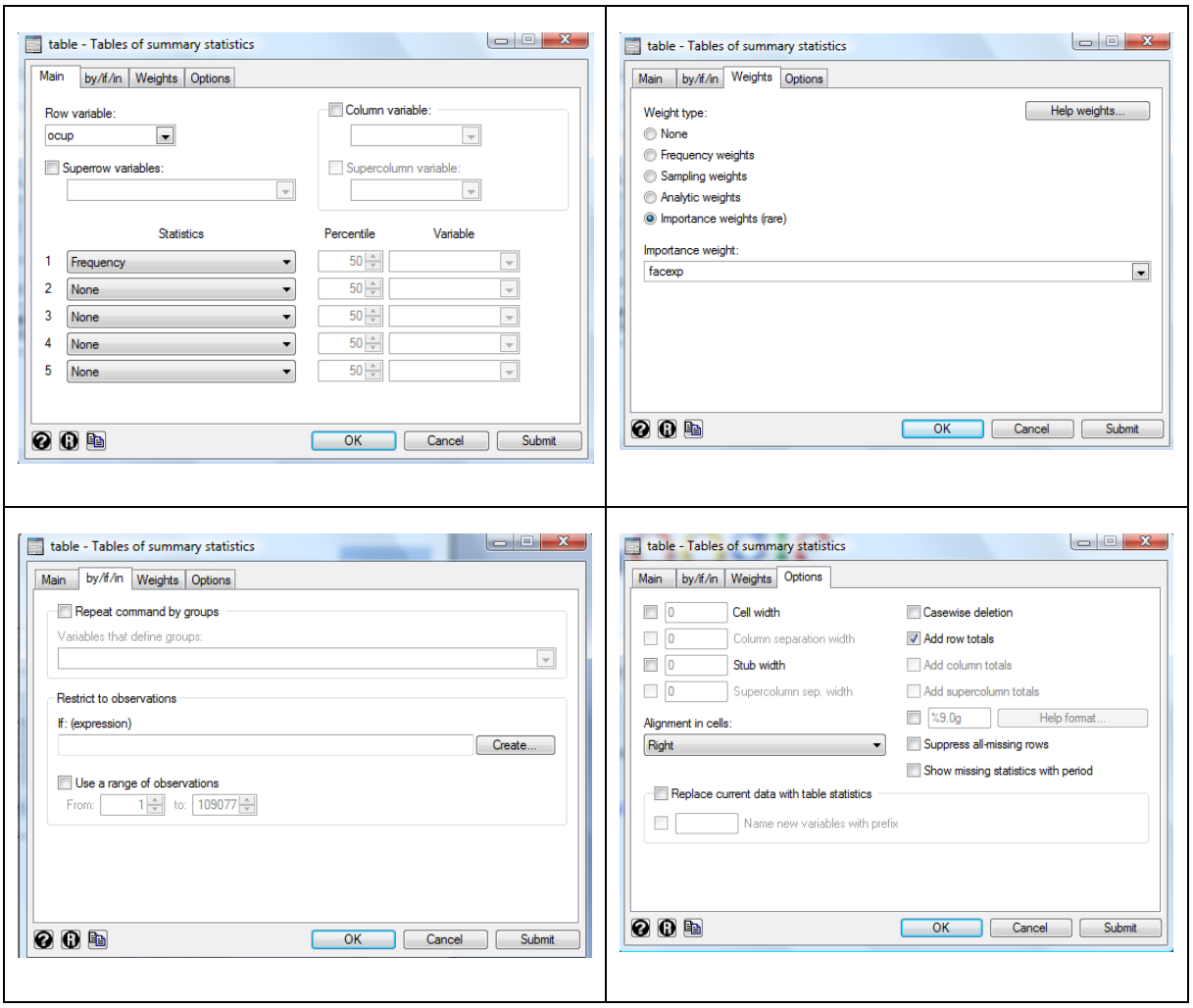
Ejemplo:

Tomando una de las tablas de la Encuesta de Hogares, digamos la correspondiente al segundo trimestre de 2006, se desea calcular el número de personas ocupadas y desocupadas.

Como se requiere el número de elementos, necesitamos calcular la “freq” y como deseamos desagregar por ocupados y desocupados requerimos una variable categórica que indique este hecho.

Adicionalmente, si se desea estimar los elementos en la población, se deberá ajustar en la pestaña de factores de expansión la opción iweight y tomar, de esta base de datos, la variable correspondiente a los factores de expansión.

Realizando estas operaciones, las pestañas deberán quedar de esta manera:



El resultado se verá de la siguiente forma:

```
. table ocup [iweight = facexp], contents(freq ) row
```

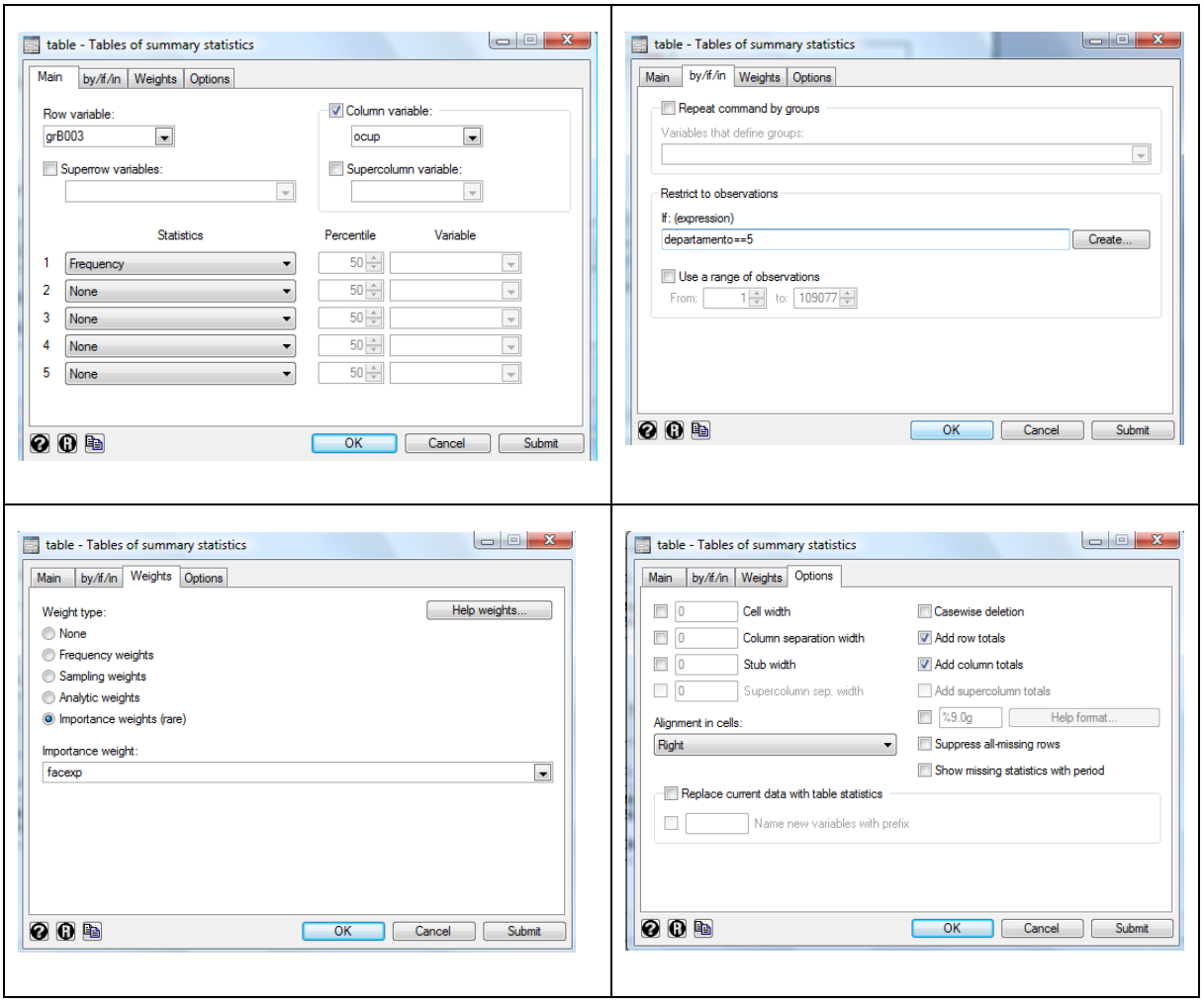
Poblacion Ocupada	Freq.
Desempleados	**1240646**
Empleados	**8479160**
Total	**9719806**

.

Este tipo de reporte se denomina de una entrada, por cuanto sólo aparece una variable categórica (la variable fila). En el próximo ejemplo haremos una tabla de doble entrada.

Ejemplo:

En un caso más complejo, se desea generar una tabla de reporte, donde las filas sean las personas ocupadas y desocupadas, en tanto, las columnas sean los rangos de edad de dichas personas. Para hacerlo más complejo, se requiere este cálculo únicamente para el departamento 5.



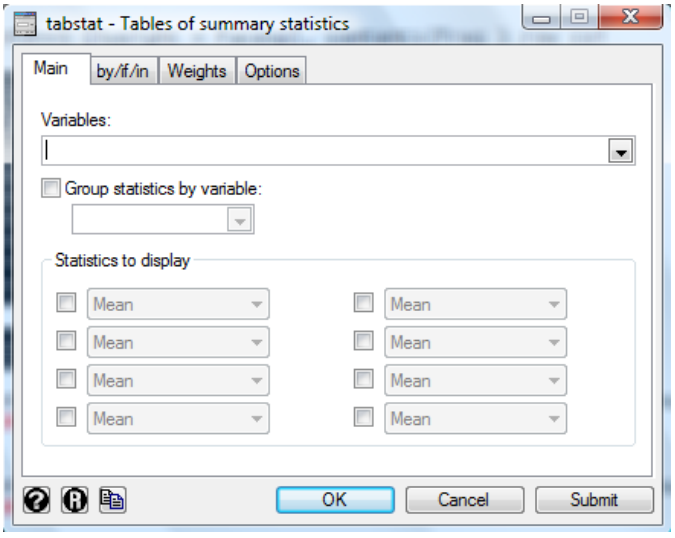
El resultado se verá de esta manera:

Grupos de edad	Poblacion ocupada		
	Desempleados	Empleados	Total
menores de 15		2,514.33	2,514.33
de 15 a 24	74,520	185,689	260,209
de 25 a 34	48,954.3	317,825	366,779
de 35 a 44	36,101.3	366,812	402,913
de 45 a 54	27,844.3	225,787	253,631
de 55 a 64	6,704.33	88,973.3	95,677.7
65 anos y mas	689	22,230.7	22,919.7
Total	194,813	1209831	1404644

El aspecto más relevante que debe tenerse en cuenta para diseñar una tabla de reporte es que tanto la variable fila como la variable columna deben ser categorías, de lo contrario el reporte puede no tener sentido práctico.

- ***COMANDO TABSTAT***

Este comando genera tablas de estadísticas descriptivas. Se diferencia de los reportes del comando Table que no necesariamente se requiere que existan variables categóricas, aunque admite al menos una de ellas y las columnas del reporte corresponden a estadísticas para diferentes variables, que en este caso serán continuas. La interfase gráfica de este reporte es:



En la opción de “variables” se eligen las variables continuas paras las cuales se deseen generar algún tipo de estadística. Se pueden elegir varias de ellas.

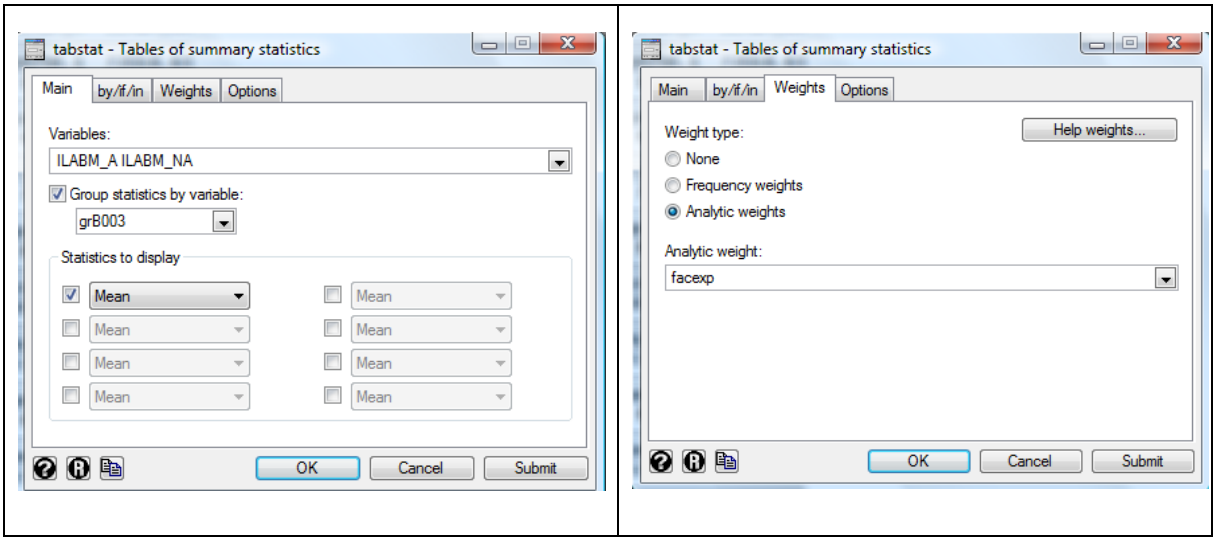
En la opción “group statistics by variable” se elige alguna variable categórica para la cual se desee agrupar la generación de resultados.

Ejemplo:

Se desea estimar la media para las variables ingreso laboral mensual de los asalariados e ingreso mensual de los no asalariados.

En este caso, el comando Tabstat permite generar dos columnas del reporte, uno para cada variable.

Si además se desea realizar el cálculo segmentando por los grupos de edad, la interfase gráfica de STATA debería ser de esta manera:



El resultado se verá como el siguiente reporte:

```
Summary statistics: mean
  by categories of: grB003 (Grupos de edad)

       grB003 |   ILABM_A  ILABM_NA
--------------+--------------------
menores de 15 |  166294.1  72018.63
   de 15 a 24 |  418894.8    295975
   de 25 a 34 |  672396.5  569399.2
   de 35 a 44 |  788418.8  613809.6
   de 45 a 54 |  919226.7  666730.4
   de 55 a 64 |  992567.6  619662.5
65 anos y mas |  768984.2  660824.2
--------------+--------------------
        Total |  704802.4  590082.3
```

PARTE III. ANALISIS GRAFICO

STATA presenta una manera eficiente de realizar diferentes gráficos, muchos de ellos asociados a datos categóricos. Los tipos de gráficos que se analizan en esta sección son:

scatter = dispersión

Histograms = histograma

Oneway = única variable

Box = box-plot

Bar = gráfico de barras

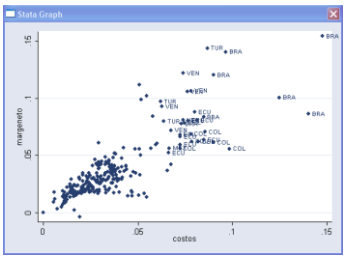
Pie = gráfico de pastel

Todas las opciones para el diseño de gráficos se encuentran en el menú GRAPHICS. Gracias a la la versatilidad del Stata 9.0 es posible editarlos muy fácilmente a partir del menú diseñado para tal fin.

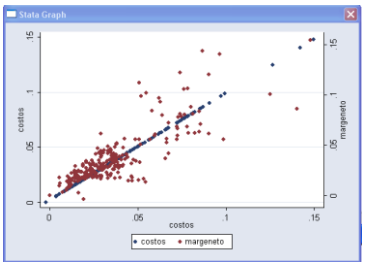
Los gráficos más sencillos se realizan en la ruta graphics-easy graphs. Cuando se desean adicionar algunas características particulares, es preferible realizarlos mediante las opciones principales graphics-twoway o overlay twoway.

Ejemplo:

twoway (scatter margeneto costos, mlabel(atipico))



twoway (scatter costos costos) (scatter margeneto costos, yaxis(2))



- ***GUARDANDO GRAFICOS***

Stata como base presenta un gráfico a la vez, sin embargo dando clic derecho sobre el gráfico es posible guardarlo en el disco duro, eligiendo la opción save graph. Supongamos que hemos creado 3 gráficos con la forma anterior, para llamarlos y verlos simultáneamente la sintaxis es:

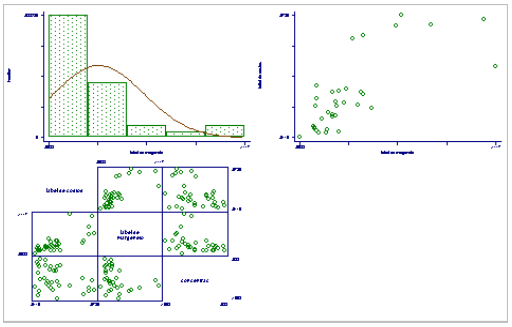
graph combine grafico1 grafico2 grafico3

También es posible, mediante la ruta graphics-manage graphics-save graphics in memory copiar dichos graficos en la memoria temporal de Stata y luego utilizarlos en una ventana mediante la ruta graphics-table of graps. La sintaxis que imprimirá el programa será:

graph copy g1

graph copy g2

graph combine g1 g2



PARTE IV. ARCHIVOS DO

Estos archivos de texto, corresponden a un conjunto de comandos que el usuario desea correr de manera simultánea y evita que se esté escribiendo cada comando en la ventana correspondiente. La forma más sencilla de crear este archivo, es eligiendo la opción “save review contents” luego de dar clic derecho sobre la ventan de review de los comandos de stata, o también, puede generarse este archivo de comandos en la medida que se opera cada comando en la interfase de STATA y dado que en cada operación se genera el comando respecto, se puede copiar cada comando en un nuevo archivo *.do línea por línea.

Puede crearse en la ventana correspondiente de stata(editor Do) o en cualquier programa de texto como word o notepad. Se recomienda cerrar un archivo *.do con exit. Esto hará que no afecte el resto de programación.

Si las líneas son muy largas dentro de un archivo *.do, entonces se puede realizar la siguiente sintaxis al inicio y al final de las líneas que se deseen, usando el comando #delimit:

#delimit ;

Líneas que se desee que finalicen una vez el usuario incluya un (;).

#delimit cr

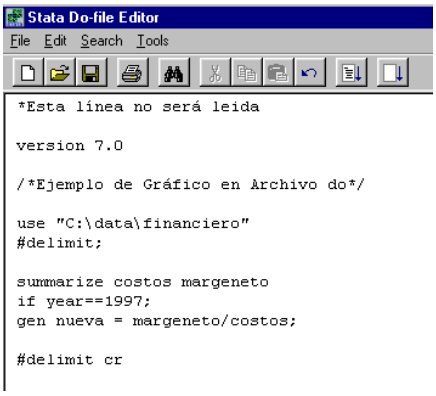
Lo que hace es reemplazar el Enter como indicador de finalización de un comando por el punto y coma(;).

Otra forma de continuar en otra línea sin perder el sentido es usar los comentarios, como por ejemplo:

```
summarize ... if (var1>0 | var1==.) & var2>0 & var3>0 & /*
*/ var1/var2>100
```

Cuando exista un error en el archivo *.do creado el programa generará un mensaje de error y no aplicará los comandos del archivo; a menos que el usuario indique desde el inicio de la creación del archivo *.do que no se detenga. Para realizar esto se adiciona el comando ***nostop***.

Ejemplo de Uso de archivos do:



Adicionalmente, puede crearse parte o todo lo que se corre en una archivo do con un comando log, de la misma forma en que se explicó en secciones anteriores;

```
Capture log close
Log using archivo, replace
…operaciones
log close
```

La línea capture log evita que el haber definido anteriormente un archivo log genere algún problema.

Igualmente es posible llamar otros archivos do, dentro de un do, simplemente se escribe dentro del código

Do nombre archivo.

Por último, existen dos formas de correr un archivo do. Usar la opción do nombre del archivo y run nombre del archivo. La diferencia entre ellas es que la opción run no imprime las salidas que pudieren generarse de comandos que aparezcan en el archivo do.

PARTE V. ELEMENTOS DE PROGRAMACION

Un programa es un archivo que se aplica en Stata diseñado por el usuario para realizar operaciones definidas por el mismo usuario.

Cuando se escribe un comando no reconocido por Stata el programa "pensará" que es un programa y buscará su nombre. Si lo encuentra lo aplica. La sintaxis básica de un programa es:

```
Program define nombre del programa
        Comandos y operaciones de Stata
End
```

Es recomendable iniciar el archivo con la versión de Stata sobre la cual desea operar(version 9.0 para este caso). Además del comando capture program drop, para que cada vez que se inicie el programa escrito se borre la memoria temporal de Stata.

Capture program drop nombre del programa

Si se desea que alguna línea no sea leída se antepone un asterisco(*); y si se desea crear comentarios, se inician con (/*) y se finaliza con(*/). Estos comentarios pueden colocarse en cualquier parte, incluso entre líneas o entre comandos. Es importante anotar que estos comentarios sólo pueden hacerse en este tipo de archivos.

Un programa y un archivo *.do no son muy diferentes para Stata(los lee igual), pero existen algunas diferencias. Estas son:

- Un archivo *.do se aplica con do nombre del archivo, un programa se aplica con sólo su nombre.

- Los programas deben ser cargados antes de ser aplicados. Para cargar un programa, la forma “simple” es primero usar run seguido del nombre del archivo donde quedó guardado el programa y luego sí aplicar el programa(simplemente dando su nombre).

Run nombre archivo

Nombre del programa

- Un archivo *.do imprime en la ventana de resultados las operaciones y los comandos usados. Un programa sólo imprime resultados.

Al igual que los archivos *.do un programa puede llamar otros programas (incluso un programa puede usar archivos *.do y viceversa).

Los programas pueden crearse en el editor de archivos *.do o en cualquier editor de texto y pueden guardarse como archivos *.do o como archivos *.ado.

Una forma más eficiente de ahorrarse este problema entre archivos es guardar directamente el programa como si fuera un archivo *.ado, y esto se hace directamente desde el editor de *.do

EJEMPLO DE UN PROGRAMA TIPICO

```
Capture log close

log using nombre archivo log, replace

set more off

Capture program drop nombre del programa

program define nombre del programa

…operaciones…

end

log close

exit
```

RECOMENDACIÓN IMPORTANTE: Se recomienda no complicar al programa con rutas de archivos muy largas. Lo ideal es tener todos los archivos referentes a un ejercicio en una única carpeta(Y SÓLO UNA CARPETA). Y también debe guardarse cada archivo *.ado en algún sitio que Stata lo pueda encontrar(por ejemplo el listado de rutas donde stata por default).

- ALGUNOS COMANDOS UTILES EN PROGRAMACION

- COMANDO SET:

Set puede utilizarse de la siguiente manera:

Set memory

Set textsize

Pero tiene otros muchos usos. Dentro de un archivo *.do puede usarse para:

Set more esta línea hace que el programa se detenga en cada pantallazo y así con un enter el usuario va leyendo.

Set rmsg off Esta línea evita que aparezca la sintaxis de cada comando en la pantalla de resultados.

Set output error Suprime todos los outputs que pudieran aparecer producto de comandos, excepto los mensajes de error.

- COMANDO DISPLAY

Sirve para imprimir títulos.

Display “título”

- COMANDO ASSERT

Este comando asegura o verifica que algún condicional sea verdad. Si lo es, no producirá output. Si no lo es, imprimirá “assertion is false” y detendrá el archivo *.do donde se encuentre. La sintaxis básica es:

By categórica: assert expresión if condición in rango

Ejemplos:

Assert costos>=0.2 | costos <=0.8 if year==1997

Assert costos ~=.

- COMANDO PRESERVE

Este comando permite que se modifique una tabla de datos dentro de un programa, pero que una vez finalizado el programa se mantenga la tabla original sin modificaciones.

Program define nombre

Operaciones

Preserve

Comandos que “destruyen” los datos originales y realizan cálculos

End

El comando preserve mantiene la tabla una vez el programa finalice. Si se desea que cargue nuevamente la tabla original desde el mismo programa, en alguna línea específica se usa además de preserve el comando restore.

Program define nombre

Operaciones

Preserve

Comandos que “destruyen” los datos originales y realizan cálculos

Restore, preserve

Otros cálculos sobre la tabla original

End

- COMANDO QUIETLY

Este comando impide que sea impreso el output de un comando. Su sintaxis es:

```
Program define nombre

Quietly regress `1’ `2’

End

Program define nombre

Quietly {

Regress `1’ `2’

Predict resid, resid

Sort resid

Summarize resid, detail

}

list `1’ `2’

drop resid

end
```

- MACROS

NOTA: En esta sección se hará uso de la siguiente sintaxis, la cual es muy importante cumplir. Cuando se hable de comillas individuales realmente corresponde a `1'. Es decir, el primer símbolo es la tilde y el segundo es la comilla individual.

Una macro dentro de Stata corresponde a las variables utilizadas en un programa de Stata. Pueden ser locales o globales. Las macros locales deberán tener máximo una letra menos que los comandos (en la versión 6.0 eran 7 letras, en la versión 7.0 son 31).

Ejemplo

Local lista1 "variable1 variable2"

Hemos creado una macro que hace que cada vez que se diga

List lista1 imprimirá variable1 variable2

Otra forma de llamar macros es con:

`lista1'

local lista variable1 variable2 variable3

regress dependiente `lista'

Las macros locales o globales pueden tener el mismo nombre y Stata las reconocerá como diferentes.

Cuando se desea que la macro deje de ser alfanúmerica(aplicable a títulos) y funcione como operaciones, lo único que debe hacerse es definir la operación:

Ejemplo:

Local dos = 2 + 2

En este caso se define una macro llamada dos la cual evalúa la operación y tendrá valor de 4.

Ejemplo:

Suponga que ya existe una macro llamada i y se le desea sumar un número 3. La sintaxis será:

Local i = `i' + 1

Nótese que para definir variables alfanuméricas se usan dobles comillas(" ") y para operar con macros ya creadas se usa comillas sencillas (` ').

Ejemplo:

If "`respuesta'" =="si" {

}

else {

}

Nótese el uso de dobles comillas: “” y también de ‘ ‘ dentro del llamado a la macro de nombre respuesta. La razón es que, primero las comillas sencillas se refieren al uso de la macro y las comillas dobles se refieren a que el contenido de esta macro es alfanumérico.

- ARGUMENTOS EN PROGRAMAS

Suponga que llama un programa de la siguiente forma:

Programa variable1 variable2

En este caso tanto variable1 como variable2 son argumentos del programa y se incluyen como macros para ser operados dentro del programa.

Stata por default define macros usando números cuando se colocan variables seguidos de un programa o un archivo *.do.

Por ejemplo:

Do ejemplo1 var1 var2 var3 var4

Se definen automáticamente las siguientes macros:

`0’ que será igual a todo lo que el usuario escribió desde var1 hasta var4 con espacios comillas, todo.

`1’ la primera palabra que corresponde a var1

`2’ la segunda palabra que corresponde a var2

etc..

```
capture program drop tester

program define tester

        display "argument 1 is |`1'|"

        display "argument 2 is |`2'|"

        display "argument 3 is |`3'|"

        display "argument 4 is |`4'|"

end

exit
```

Una mejor forma de llamar los argumentos dentro de un archivo do o un programa es usando el comando args. Con este comando se asignan los nombres a las posiciones llenadas por el usuario al llamar el programa.

Ejemplo:

```
Program define programa1

Args var1 var2 var3 var4

…operaciones…

end
```

Entonces var1 var2 var3 var4 se convierten en cuatro variables locales que corresponden a `1’ `2’ `3’ `4’ respectivamente.

Otro ejemplo puede ser el siguiente:

```
Program define ejemplo1
        Args n a b
Drop _all                   /*se usó una variable del sistema que borra todas
 las variables en memoria*/
        set obs = `1'
        generate x = (_n-1)/(_N-1)*(`b'-`a')+`a'
end
```

- USO DE CONDICIONALES

Los principales comandos que podemos usar en programación son:

If – else

While

Foreach

Forvalues

- COMANDO IF

Su sintaxis es:

```
If expression {

Comandos Stata

}

else if expression {

comandos stata

}

else {

comandos stata

}
```

Ejemplo:

Este programa toma 2 argumentos. El primero es una variable existente(x). El segundo argumento es un número(n). El programa creará una variable llamada z. Si n>0, z = x^n; si n=0, entonces z=log(x) y si n<0 entonces z= -x^n. Esta es la transformación de box-cox.

```
capture program drop potencia

program define potencia

if `2'>0 {

        generate z =`1'^`2'
```

```
        label variable z "`1'^`2'"
        }
else if `2'==0 {
        generate z = log(`1')
        label variable z = "log(`1')"
}
else {
        generate z = -(`1'^(`2'))
        label variable z "`1'^(`2')"
}
end
```

Aquí introducimos la sintaxis para los corchetes. Estos indican operaciones repetitivas(loop) todo lo que vaya dentro de corchetes se hará de manera que cumpla la condición impuesta, bien sea por un condicional if, como en este caso, o por otros condicionales como los que siguen a continuación.

- COMANDO FOR

Este comando permite realizar operaciones repetitivas. Su sintaxis básica es:

For variable in rango tipo de lista : comando condición en función de X.

Siendo x un letra que siempre debe ir; en este caso particular(únicamente con este comando) la X es fija como sintaxis; y también nótese que la X va en mayúscula.

En general estas variables comodin(en mayúscula para escribir la operación correspondiente) son:

X, Y, Z, A, B, C, D, E mejor dicho, cualquier cosa que vaya en MAYUSCULA

Se pueden tener varios tipos de lista separados por \; de igual manera se pueden tener varias condiciones separadas por \

Los tipos de lista pueden ser:

Var	Un listado de variables: var1 var2 var3
New	una nueva variable
Num	Una secuencia de números(por ejemlo 4/10)
Any	para cualquier variable

Ejemplos:

- Creación de 100 variables con números uniformes:

(debe haberse definido el tamaño de la tabla previamente. Esto se hace con la instrucción set obs número)

set obs 100

For new u1-u10: gen x=uniform()

- Reemplaza en cualquier variable un missing value:

For any . : replace z=. If y= X

- Genera nuevas 5 variables que corresponden a los poderes de x2, x3, x4, x5.

For new x2-x5 \ num 2/5: gen X =variable^Y

- COMANDO FORVALUES

La sintaxis para este comando es:

```
forvalues nueva_macro = rango {
        comandos referidos a nueva_macro
}
```

Donde nueva macro es el nombre de una nueva macro local y rango especifica el rango de valores para los cuales se desea operar.

- Ejemplo: Construcción de un contador hasta 10

```
program define diez
        forvalues i = 1(1)10 {
                display "`i'"
        }
end
```

Nótese la sintaxis del rango la macro se llama i y empieza en 1(va de 1 en 1)hasta 10.

```
program define ejemplo
        local N = _N
        forvalues i = 1(1)`N' {
                display name[`i']
                display _column(10) "genero " genero[`i']
        }
        end
```

- COMANDO WHILE

La sintaxis para este comando es:

```
while condicion {
                comandos Stata
        }
```

- Ejemplo: Construcción de un contador hasta 10

```
program define diez_w
                local i = 1
                while `i' <= 10 {
                        display `i'
                        local i = `i' + 1
                }
        end
```