Departamento de Economía
Facultad de Ciencias Económicas
Universidad Nacional de La Plata

Introducción a Stata

Econometría I

Introducción a Stata

- El programa
- Base de datos
- Sintaxis
- Archivos DO y LOG

Introducción a Stata

- **El programa**
- Base de datos
- Sintaxis
- Archivos DO y LOG

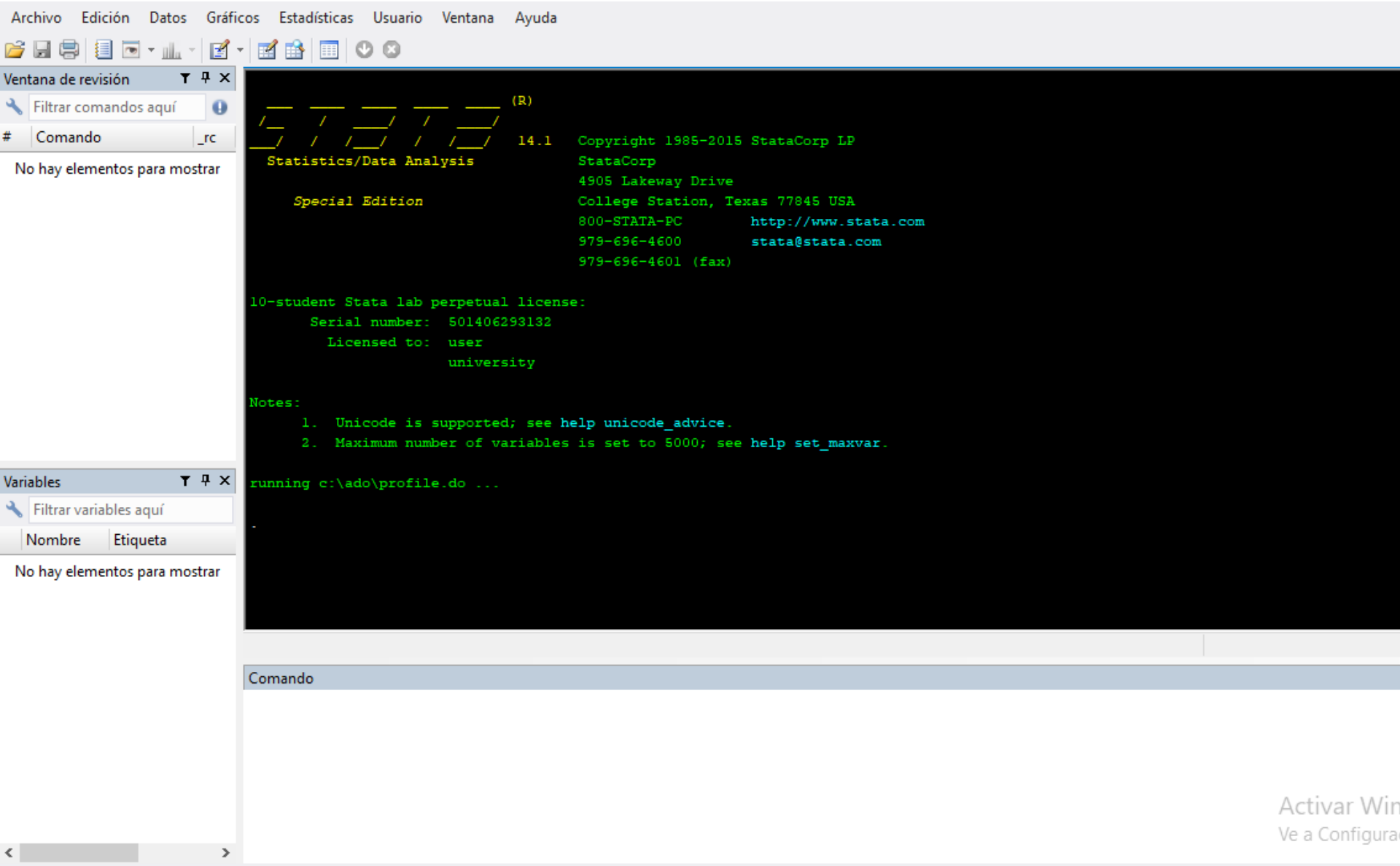
¿Qué es STATA?

- Stata es un programa que permite la gestión de base de datos y la realización de cálculos estadísticos y econométricos.
    - ¿Cuánto es en promedio el ingreso total familiar en Argentina?
    - ¿Cuál es la región argentina que presenta una población más envejecida?
- Se basa en un lenguaje de programación que respeta una sintaxis.
- Existen versiones de Stata para Windows, Linux y Mac.

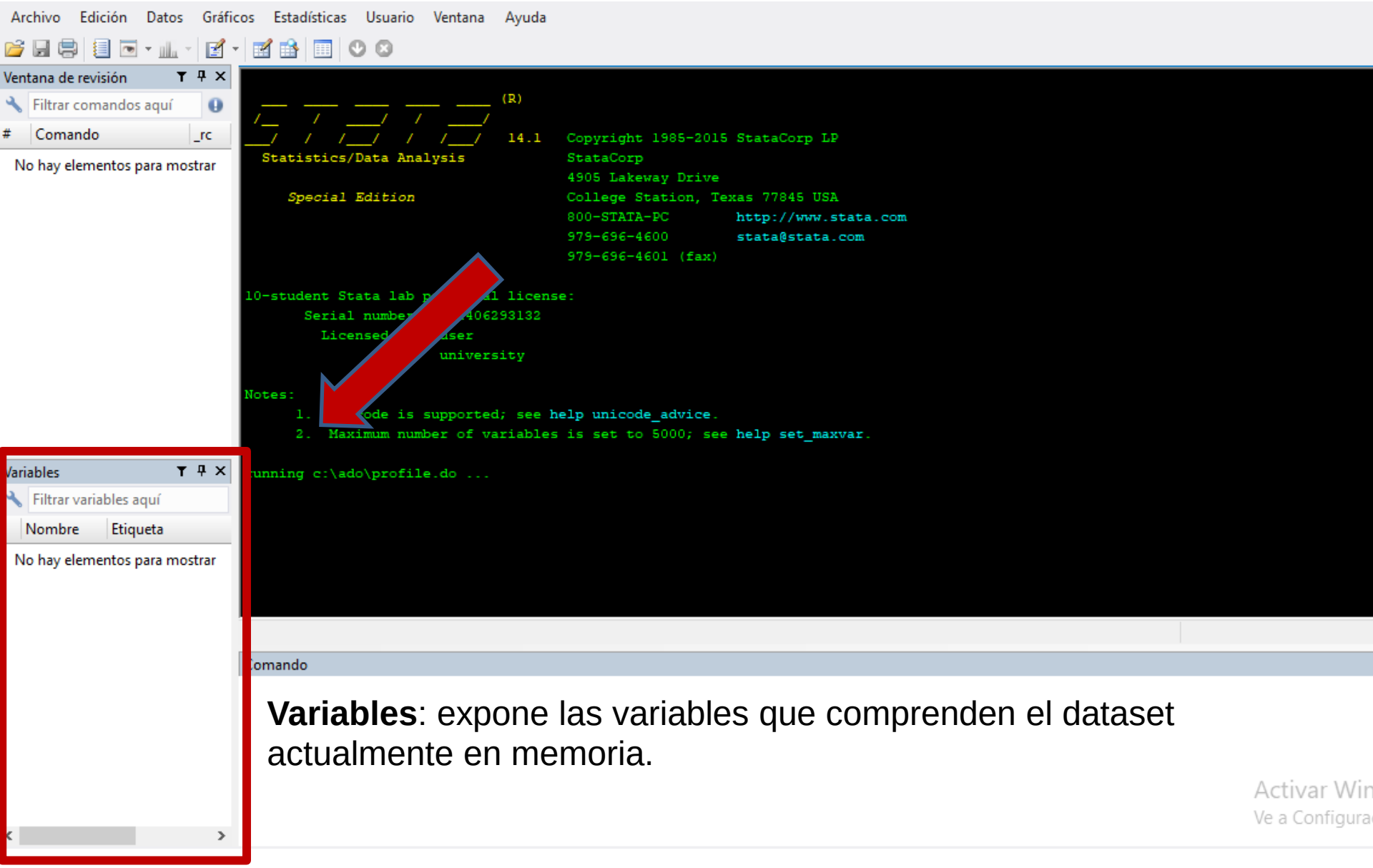
¿Dónde se aprende?

- Stata ofrece varias alternativas:
  - Para una primera aproximación es suficiente con el User's Guide.
  - Manuales detallados por comandos.
  - Stata Press: libros sobre temas específicos (estadística, econometría, demografía, etc.)
  - Stata Journal: artículos sobre nuevos métodos, comandos y otros tópicos de programación.
  - En la web: blogs, videos tutoriales, etc.

Interface

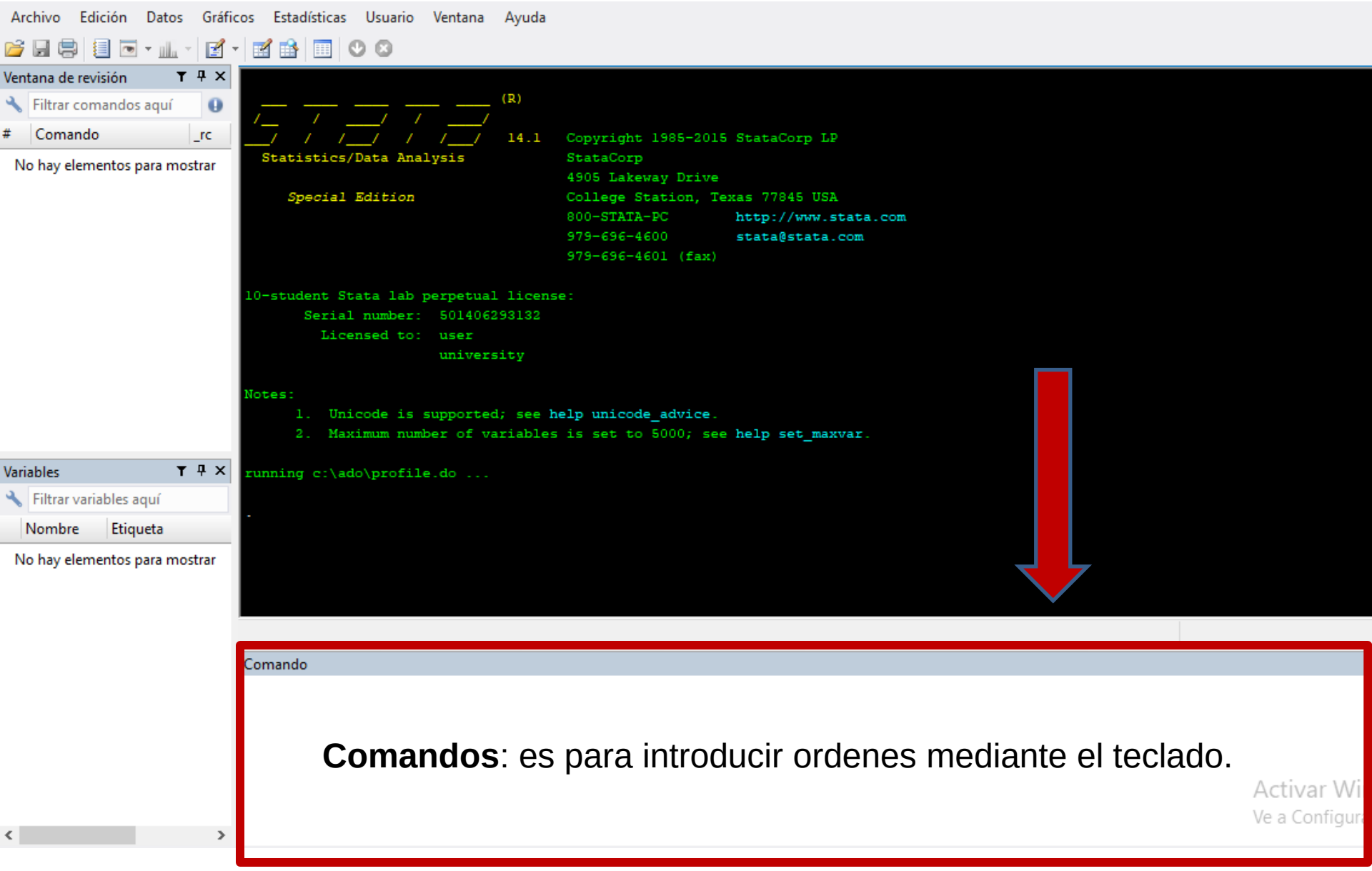

Archivo
Edición
Datos
Gráficos
Estadísticas
Usuario
Ventana
Ayuda
Ventana de revisión
Filtrar comandos aquí
Comando _rc
No hay elementos para mostrar
Variables
Filtrar variables aquí
Nombre Etiqueta
No hay elementos para mostrar
(R)
14.1
Statistics/Data Analysis
Special Edition
Copyright 1985-2015 StataCorp LP
StataCorp
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)
10-student Stata lab perpetual license:
Serial number: 501406293132
Licensed to: user
university
Notes:
1. Unicode is supported; see help unicode_advice.
2. Maximum number of variables is set to 5000; see help set_maxvar.
running c:\ado\profile.do ...
.
Comando

Interface



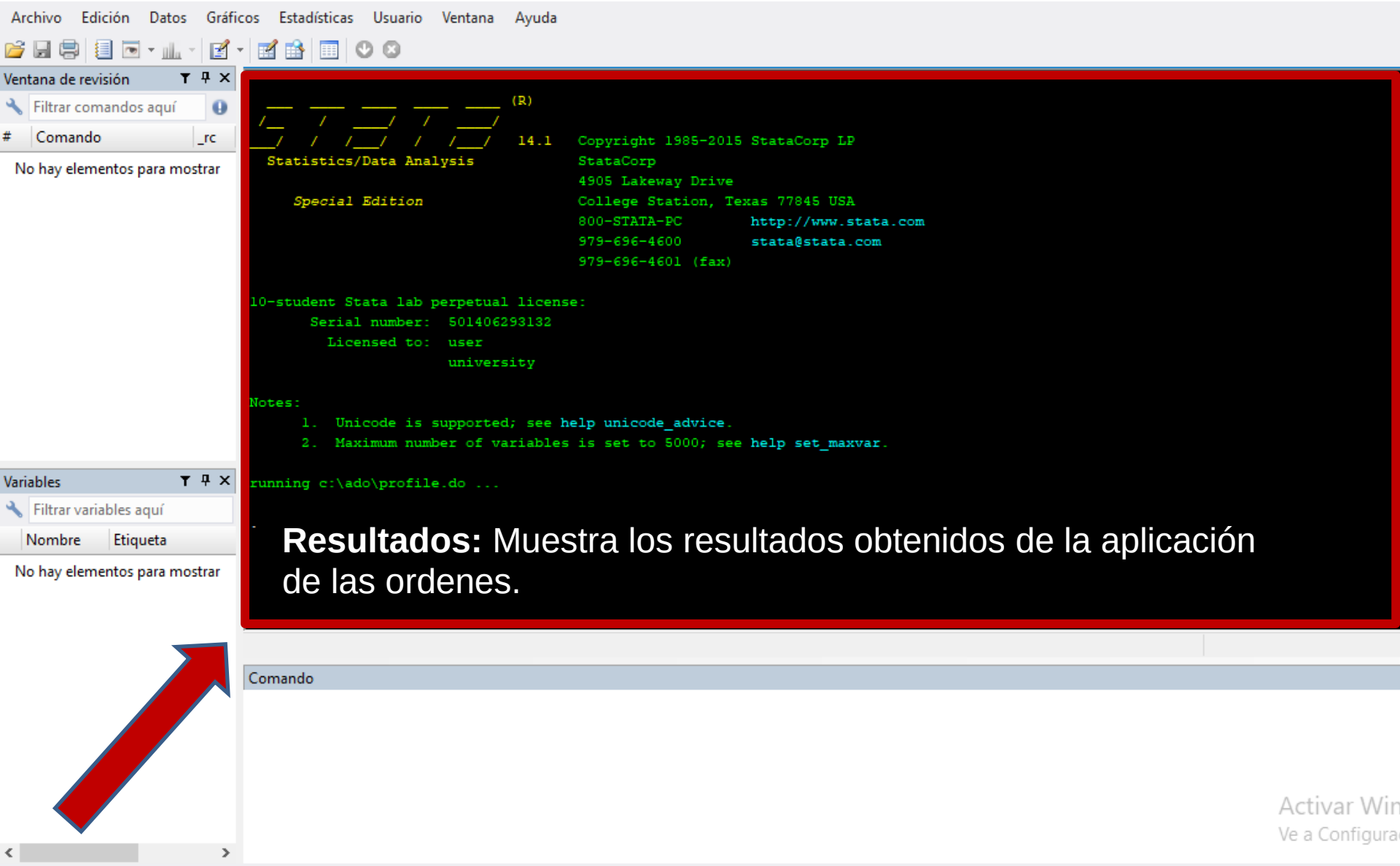
Variables: expone las variables que comprenden el dataset actualmente en memoria.

Interface

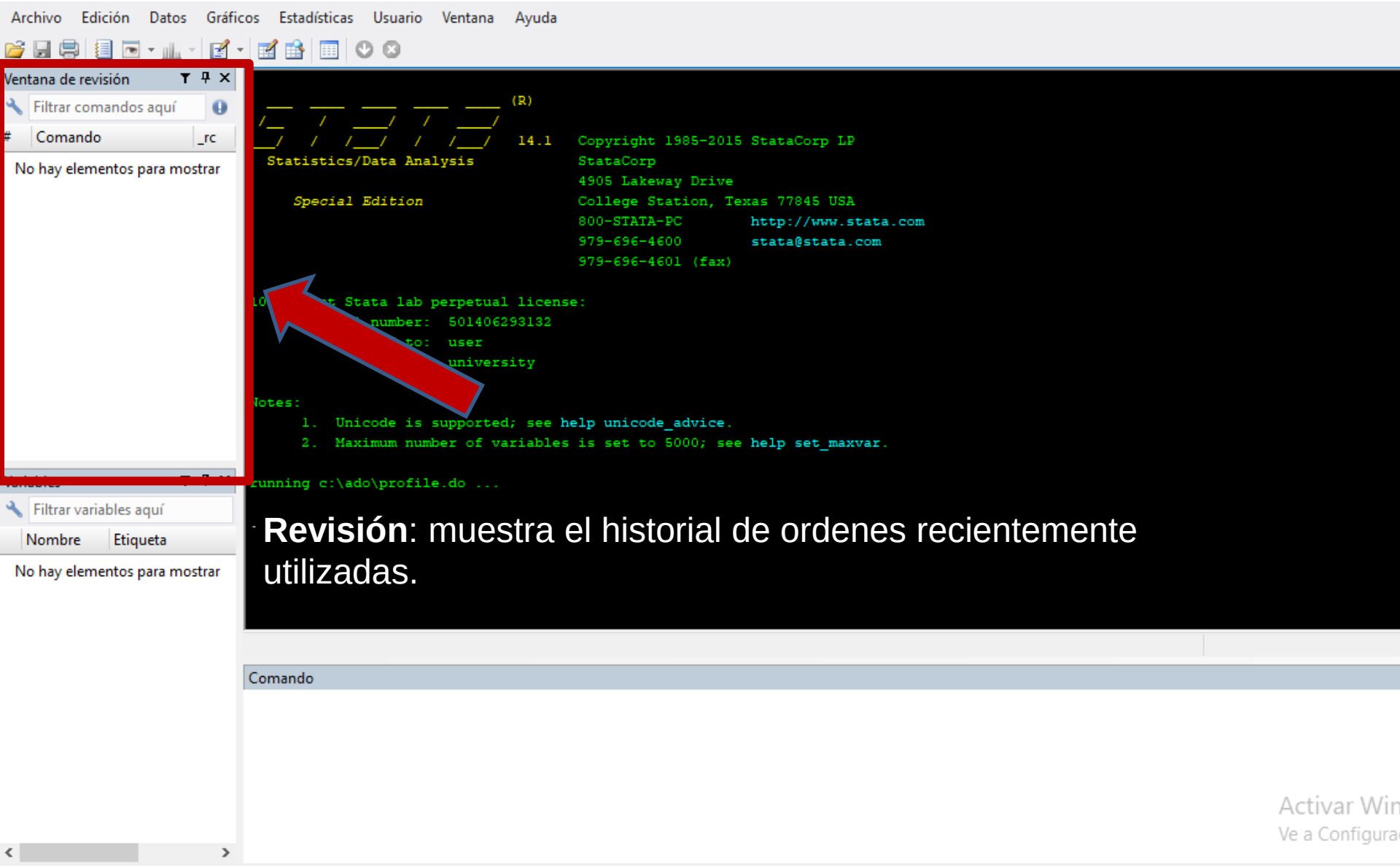


Comandos: es para introducir ordenes mediante el teclado.

Interface


Archivo Edición Datos Gráficos Estadísticas Usuario Ventana Ayuda
Ventana de revisión
Filtrar comandos aquí
Comando _rc
No hay elementos para mostrar
Variables
Filtrar variables aquí
Nombre Etiqueta
No hay elementos para mostrar
(R)
14.1 Copyright 1985-2015 StataCorp LP
Statistics/Data Analysis StataCorp
4905 Lakeway Drive
Special Edition College Station, Texas 77845 USA
800-STATA-PC http://www.stata.com
979-696-4600 stata@stata.com
979-696-4601 (fax)
10-student Stata lab perpetual license:
Serial number: 501406293132
Licensed to: user
university
Notes:
1. Unicode is supported; see help unicode_advice.
2. Maximum number of variables is set to 5000; see help set_maxvar.
running c:\ado\profile.do ...
Resultados: Muestra los resultados obtenidos de la aplicación de las ordenes.
Comando

Interface



- **Revisión**: muestra el historial de ordenes recientemente utilizadas.

Introducción a Stata

- El programa
- **Base de datos**
- Sintaxis
- Archivos DO y LOG

Base de datos

- Utilizaremos el archivo arg_2020_ephc_t3.dta (vamos a trabajar en una carpeta denominada “Intro Stata”).

- Este dataset contiene datos provenientes de la Encuesta Permanente de Hogares de Argentina, del tercer trimestre del 2020.

- ¿Cómo abrimos una base?:
    a. Forma rápida (no aconsejable): Doble click en el archivo.
    b. Forma aconsejable:
        i. Aclararle a Stata el directorio de nuestra carpeta:

```
cd "C:\Intro a stata"
```

        ii. Abrir la base de datos:

```
use "arg_2020_ephc_t3.dta"
```

Base de datos

Notemos:

- Las comillas (“ ”) son necesarias si hay espacios en la ruta o en el nombre del archivo. Ejemplo:

```
cd C:\Intro_Stata
```
✔

```
cd C:\Intro Stata
```
✖

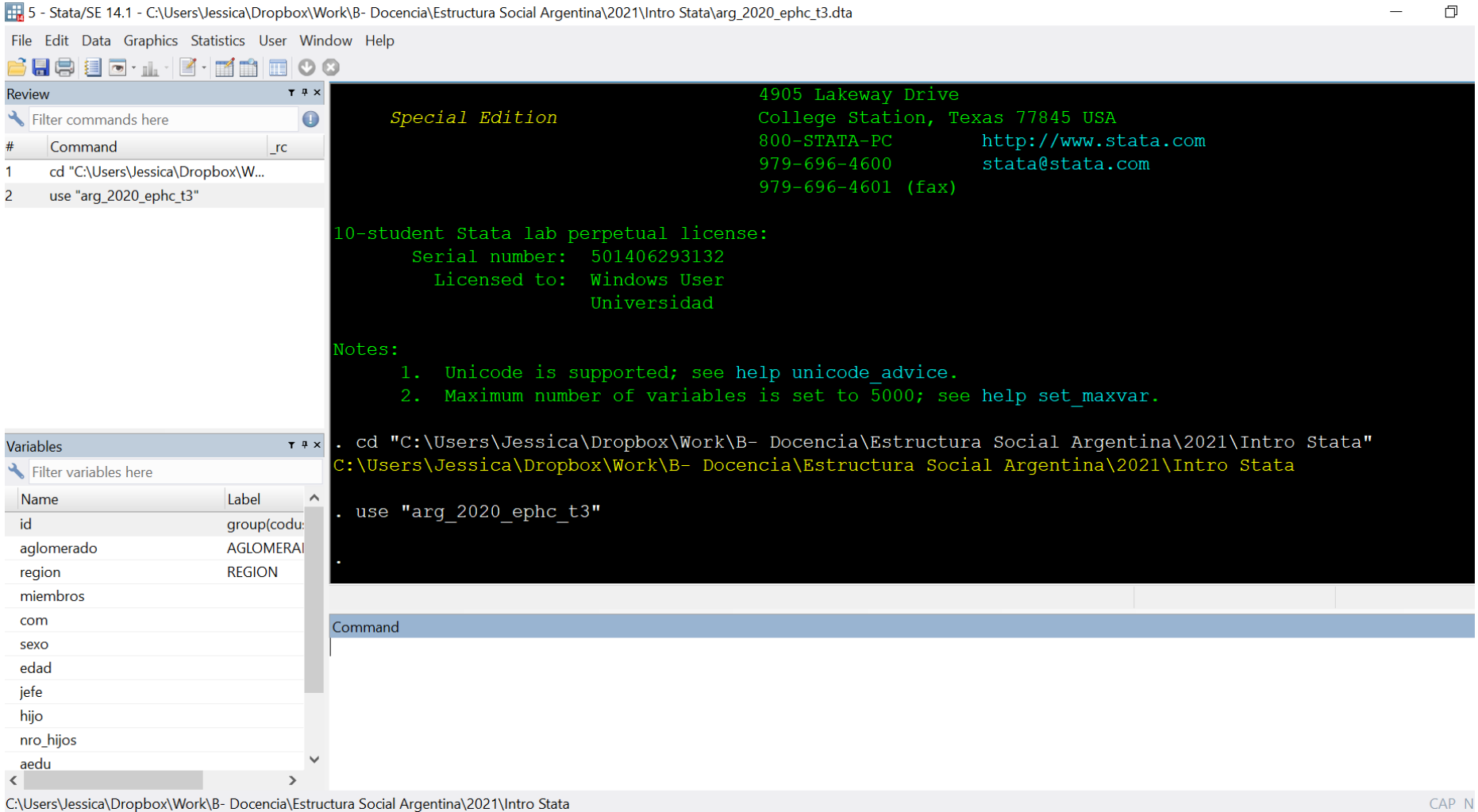
```
cd “C:\Intro Stata”
```
✔

- No debe haber un dataset previo en memoria. Si lo hay, opción “clear” (más adelante)
- Se pueden resumir los dos pasos en la siguiente oración:

```
use “C:\Intro Stata\arg_2020_ephc_t3”
```

Base de datos

- Si la carga del dataset fue exitosa, veremos que las ventanas Revisión, Variables y Resultados se modificaron:



Estructura del dataset

- El contenedor de datos en Stata se denomina DATASET. Se puede visualizar mediante el comando `browse`.

	id	aglomerado	region	miembros	com	sexo	edad	jefe
1	1	19	40	2	1	1	35	1
2	1	19	40	2	2	2	29	0
3	3	26	42	5	1	1	56	1
4	3	26	42	5	2	2	56	0
5	3	26	42	5	3	1	21	0
6	3	26	42	5	4	1	29	0
7	3	26	42	5	5	1	31	0
8	4	17	44	2	1	2	65	1
9	4	17	44	2	2	1	22	0

- Es una tabla de doble entrada donde las columnas se denominan variables y las filas observaciones.

- En cada celda se pueden guardar datos de diferentes tipos.

Tipos de datos

- Tipos de datos en Stata: (i) números, (ii) palabras y (iii) fechas.

- Datos numéricos: admite varios formatos.

Nombre	Tipo de números	Límite inferior	Límite superior
Byte	Enteros	-127	100
Integer	Enteros	-32,767	32,740
Long	Enteros	-2,147,483,647	2,147,483,620
Float	Con decimales	-1.70141E+38	1.70141173319*10^38
Double	Con decimales	-8.9885E+307	8.9884656743*10^307

- Datos en palabras (*string*): se pueden almacenar palabras con hasta un máximo de 2045 caracteres.
- Fechas: son números con un formato especial de visualización.

Introducción a Stata

- El programa
- Base de datos
- **Sintaxis**
- Archivos DO y LOG

Sintaxis

- Stata trabaja mediante la especificación por parte del usuario de órdenes que se denominan comandos.

- Los comandos conforman un lenguaje de comunicación con el programa, por lo que existe una determinada sintaxis que tiene la siguiente estructura general:

```
[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight]
   [,opciones]
```

- Los corchetes indican elementos opcionales. Existen comandos que comprenden sólo una palabra.

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Un ejemplo de comando que funciona sólo invocando su nombre es el comando `describe`, que muestra una descripción de la base de datos y la lista de variables que contiene:

```
describe
```

- Un comando muy útil para conocer de forma rápida la cantidad de observaciones en una base es `count` :

```
count
```

- Otros casos son los comandos `browse`, que muestra toda la base de datos, y `summarize`, que muestra estadísticas descriptivas:

```
browse
summarize
```

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Hace referencia a una o más variables involucradas en un comando. Por ejemplo, estadísticas descriptivas de una variable:

```
summarize edad
```

- Para un grupo de variables:

```
summarize edad aedu
```

- Lista de variables (en este caso todas las variables que están entre edad y aedu)

```
summarize edad-aedu
```

- Variables que empiecen con la letra i:

```
summarize i*
```

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Se utiliza cuando se quiere limitar la aplicación de un comando a observaciones que cumplen ciertas restricciones.

- Para ello se utiliza el "si condicional" (**if** en inglés)

- Por ejemplo, descripción estadística de la variable edad para las observaciones correspondientes a mujeres.

```
summarize edad if …
```

- Operadores de comparación:

  - *Igual*: ==
  - *Distinto*: !=
  - *Mayor (menor):* > (<)
  - *Mayor o igual (menor o igual)*: >= (<=)

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Se utiliza cuando se quiere limitar la aplicación de un comando a observaciones que cumplen ciertas restricciones.

- Para ello se utiliza el "si condicional" (**if** en inglés)

- Por ejemplo, descripción estadística de la variable edad para las observaciones correspondientes a mujeres.

```
summarize edad if sexo==2
```

- Operadores de comparación:

  - *Igual*: ==
  - *Distinto*: !=
  - *Mayor (menor):* > (<)
  - *Mayor o igual (menor o igual)*: >= (<=)

- Operadores lógicos:
  - *Y*: &
  - *O*: |

- Operador jerárquico: el paréntesis determina el orden de aplicación de las sentencias condicionales.

- Algunos ejemplos más complejos de sentencias condicionales son los siguientes:

```
summarize edad if sexo==2
summarize edad if (sexo==2 & aglomerado==2) | sexo==1
summarize edad if sexo==2 & (aglomerado==2  | sexo==1)
summarize edad if !(sexo==2) & (aglomerado==2 |sexo==1)
```

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Permite aplicar el comando a un rango de observaciones, de acuerdo al orden del dataset.

- Ejemplo 1: descripción estadística de las 10 primeras observaciones del dataset

```
summarize edad in 1/10
```

- Ejemplo 2: aplicar el comando a las últimas 10 observaciones

```
summarize edad in -10/-1
```

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- El componente **[in range]** depende del ordenamiento del dataset.
- Los comandos para ordenar un dataset son `sort` y `gsort`.
- El comando `sort` permite ordenar sólo de manera ascendente de acuerdo a la variable que se especifica:

```
sort itf
```

- El comando gsort permite ordenar en cualquier sentido.
- De manera descendente según la población de cada país:

```
gsort -itf
```

- De manera ascendente (por default, no es necesario el “+”):

```
gsort +itf
```

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Se utiliza generalmente con dos comandos: `generate` y `replace`.

```
generate nueva = 0
```

- Permite crear una nueva variable en el dataset. Es requisito indicar la definición de dicha variable.

- En este caso la variable se llama “nueva” y tiene valor 0 en todas las observaciones.

- Se pueden generar nuevas variables en base a otras variables (próxima clase)

- También puede crearse una variable con valores nulos (en Stata se indica “.”)

```
generate nulo = .
```

- O variables que contienen texto (*string*):

```
generate region2 = ""
```

- ¿Cuál es el posible contenido de una *string?*:
    - Datos identificatorios: esta información no puede ser utilizada directamente en el análisis estadístico.
    - Valores no categóricos: se trata de variables con números en formato *string*. Para esto es útil el comando `destring`.

- Stata no permite dos variables con el mismo nombre

```
generate region2 = "GBA"
variable region already defined
```

- Podemos borrar variables mediante el comando `drop`

```
drop region2
```

- U observaciones…

```
drop if aglomerado==2
```

- A las variables existentes se les puede crear una etiqueta (label):

```
label var region "Region del pais"
```

- También puede etiquetar los valores de una variable, especialmente si la variable es categórica. Se procede en dos pasos:
  - Paso 1: crear las etiquetas de los valores

```
label define etiqueta 1 "GBA" 40 "NOA" 41 "NEA" 42 "Cuyo"
```

  - Paso 2: le aplico la etiqueta definida anteriormente a los valores de la variable (o las variables) que corresponda:

```
label values region etiqueta
```

- Las variables con valores etiquetados aparecen en color azul al hacer **browse**

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Permite sistematizar la aplicación del comando por grupos de observaciones.
- Los grupos están definidos por los valores de la variable indicada.
- Es requisito ordenar el dataset por la variable que se va a usar en el **[by varlist:]**

```
sort region
```

- Luego:

```
by region: summarize edad
```

- Una alternativa en una sola línea de código es `bysort`

```
bysort region: summarize edad
```

- Lo mismo se podría haber obtenido haciendo:

```
summarize edad if region==1
summarize edad if region==40
summarize edad if region==41
summarize edad if region==42
summarize edad if region==43
summarize edad if region==44
```

- Debe notarse que este método es muy engorroso si la variable que agrupa los datos tiene muchas opciones.

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- En la mayoría de los casos, utilizamos bases de datos que no representan la población total de referencia, sino una muestra.
- Un ponderador permite expandir los resultados obtenidos para el total de la población.

	id	aglomerado	com	sexo	edad	pondera
1	1	19	1	1	35	80
2	1	19	2	2	29	80
3	3	26	1	1	56	158
4	3	26	2	2	56	158
5	3	26	3	1	21	158
6	3	26	4	1	29	158
7	3	26	5	1	31	158
8	4	17	1	2	65	169
9	4	17	2	1	22	169
10	5	19	1	1	61	353
11	5	19	2	2	61	353
12	5	19	3	1	23	353

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Calculamos la edad media de la muestra:

```
summarize edad
```

- Calculamos la edad media de la población de referencia:

```
summarize edad [weights=pondera]
```

[by varlist:] comando [varlist] [=exp] [if exp] [in range] [weight] [, opciones]

- Existen comandos que aceptan opciones adicionales.
- Éstas son especificadas en la sintaxis luego de una coma.
- Por ejemplo, resumen estadístico más detallado

```
summarize edad, detail
```

- De esta manera, el comando summarize ahora brinda una descripción estadística distinta a la que hace por defecto.

- Vimos que para abrir una base mediante el comando `use`, es necesario que no haya un dataset previo abierto en el programa.

- Con la opción `clear` evitamos tener que cerrar y abrir el programa al momento de abrir una base

```
use "arg_2020_ephc_t3", clear
```

Abreviaturas

- Los comandos y variables usados pueden ser abreviados.

- La regla es que la abreviatura puede realizarse siempre que no se confunda con otro comando (o variable). Por ejemplo:

```
sum edad
```

- Existen algunas excepciones a esta regla:

  - Los comandos “destructivos” no se abrevian: *drop*, *clear*
  - Existe el comando `describe` que se abrevia con d, a pesar de confundirse con otros

Introducción a Stata

- El programa
- Base de datos
- Sintaxis
- **Archivos DO y LOG**

Archivos DO y LOG

- Hasta ahora la interacción con Stata ha sido mediante el tipeo de comandos en la ventana de comandos de Stata.
- Archivo “DO” (do-file): son archivos de texto que contienen una secuencia de comandos.
- Al ejecutar dicha secuencia, los resultados serán visualizados en Stata, pero no guardados.
- Una forma de guardar esos resultados es utilizando un archivo LOG.
- Archivos “LOG” (log-file): son archivos de texto en donde se almacena una copia de todo lo visualizado en la ventana Results de Stata.

Proyecto

- Está compuesto por todos los archivos que intervienen en nuestra interacción con Stata.

- En resumen, un proyecto simple contiene los siguientes archivos:

Archivo	¿Qué hace?
dta	contiene los datos necesarios
do	ejecuta una secuencia de comandos
log	guarda los resultados

- Esta estructura de proyecto resulta muy útil para la resolución de gran parte de los TPs.

Archivo DO

- Un archivo DO es de tipo texto plano (sin formatos).

- La idea central es que contenga una secuencia de comandos que nos permita obtener ciertos resultados.

- Para la creación de un DO-FILE tenemos dos alternativas:
  1) Editor de textos que tiene incluido Stata.
  2) Editor de textos externo.

Editor de textos incluido en Stata:
Se puede abrir desde el Menú o mediante el siguiente comando:

```
doedit
```

Archivo DO

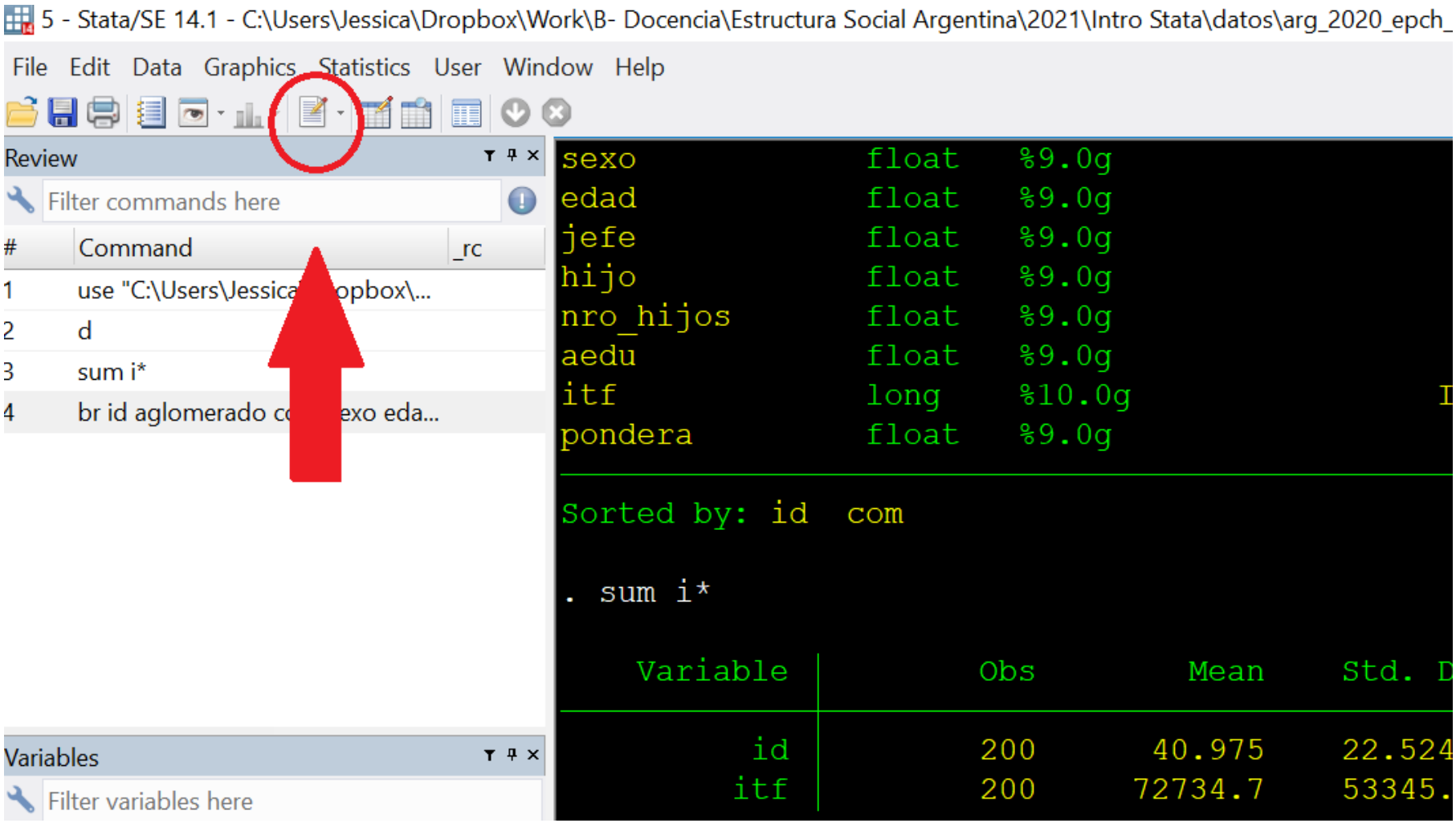
Editor de textos externo

- Hay varias opciones disponibles:
  - Notepad++: http://notepad-plus-plus.org/
  - Editplus: http://www.editplus.com/
  - Textpad: http://www.textpad.com/
  - Crimson: http://www.crimsoneditor.com/

- Cada uno requiere instalar un archivo para que reconozca la sintaxis de Stata.

Mi primer archivo DO

- Abrimos nuestro archivo do:



Mi primer archivo DO

- Escribimos los comandos que queremos guardar:

```
cd "Intro Stata"
use "arg_2020_ephc_t3", clear
describe
```

- Guardamos el archivo con el nombre clase1.do en la carpeta C:\Intro Stata.

- En Stata ejecutamos el do-file clase1. Hay dos maneras:
  1) Utilizando el botón "Execute (do)"
  2) Mediante el comando do:

```
do clase1.do
```

Mi primer archivo DO

- Si todo funcionó bien, habremos ejecutado nuestro primer archivo DO y en la ventana Stata Result estará el resultado del comando `describe`.
- Nota: pueden incorporarse comentarios dentro del archivo DO de la siguiente manera: /* Este es un comentario */. También es un comentario una línea iniciada con asterisco *

```
/* Fijamos el directorio
  Abrimos la base de datos */
 cd "Intro Stata"
 use "arg_2020_ephc_t3", clear
 * Describimos base de datos
 describe
```

Mi primer archivo LOG

- Los resultados que se registran en la ventana de resultados pueden ser almacenados en un archivo de texto de extensión .log

- El código a agregar para obtener un archivo LOG es el siguiente:

```
cd "Intro Stata"
use "arg_2020_ephc_t3", clear
capture log close
log using clase1.log, replace
describe
log close
```

Mi primer archivo LOG

Funcionamiento:

- log using hace que se empiecen a registrar los resultados en el archivo clase1.log mientras que log close los cierra.
- replace implica que en cada nueva ejecución del programa los resultados se sobrescriben.
- Solo quedan registrados los resultados entre el log using y el siguiente log close.

Comentarios:

- Existen otras formas de exportar resultados (en forma de tablas, texto y gráficos).
- El archivo LOG es una de las más primitivas pero la más simple (y por lo tanto útil para principiantes).