SINTAXIS SPSS

Cecilia Esparza Catalán

ÍNDICE

SINTAXIS SPSS

1.- INTRODUCCIÓN[1]

Existen dos formas de trabajar con el SPSS: seleccionando las tareas a realizar mediante el sistema de ventanas, o indicando las operaciones a efectuar mediante la sintaxis del programa (lenguaje de comandos).

La sintaxis de SPSS funciona a través de comandos, a los que se puede acceder desde los menús y cuadros de diálogo. Sin embargo, en ocasiones algunas de las posibilidades del SPSS solo están accesibles a través de la sintaxis. La ventaja que presenta trabajar con este lenguaje es que los archivos de sintaxis pueden guardarse y volver a ser ejecutados en sesiones diferentes.

Un archivo de sintaxis SPSS es simplemente un archivo de texto que contiene comandos. Aunque es posible abrir una ventana de sintaxis y escribir los comandos (*Archivo – Nuevo - Sintaxis*), suele ser más sencillo construir un archivo de sintaxis mediante uno de los siguientes métodos:

- Pegando la sintaxis de comandos desde los cuadros de diálogo.

  La mayor parte de los cuadros de diálogo a los que podemos acceder a través del menú de ventanas del SPSS incluyen un botón llamado “Pegar”. Dicho botón nos permite, una vez seleccionadas en el cuadro todas las opciones que nos interesan, pegar en un archivo la correspondiente sintaxis del comando.

- Copiando la sintaxis desde las anotaciones de los resultados.

  Es posible configurar SPSS para que la sintaxis correspondiente al procedimiento que realizamos aparezca en el visor junto con los resultados del mismo. Para ello solo hay que seleccionar la opción “Mostrar comandos en anotaciones” en la configuración del Visor (*menú Edición, Opciones, pestaña Visor*) antes de ejecutar el procedimiento.

- Copiando la sintaxis desde el archivo diario.

  Por defecto, todos los comandos que se han ejecutado durante una sesión se graban en un archivo de diario denominado spss.jnl. El archivo de diario es un archivo de texto en el que se añaden o se sobrescriben las acciones realizadas durante una sesión según cómo configuremos las opciones del programa. En la pestaña General de las opciones del menú

[1] OBSERVACIÓN: Todos los contenidos del presente manual han sido elaborados empleando la versión 13.0 del programa SPSS.

Edición podemos configurar las características de dicho archivo, así como localizar y/o modificar la ruta en la que se encuentra.

Para obtener información detallada sobre el lenguaje de comandos se puede consultar el SPSS Command Syntax Reference (disponible únicamente en inglés), al que podemos acceder a través del menú de ayuda del programa.

Una vez que hemos creado el archivo de sintaxis con los comandos que nos interesan, para obtener los resultados debemos ejecutarlo. Para ello debemos ir al menú *ejecutar* del Editor de sintaxis. Podemos elegir entre las siguientes opciones:

- Todo: se ejecuta el archivo de sintaxis completo.
- Selección: se ejecutan únicamente los comandos seleccionados.
- Actual: se ejecuta únicamente el comando actual (aquel en el que esté situado el cursor).
- Hasta el final: se ejecuta el archivo desde el comando actual hasta el final.

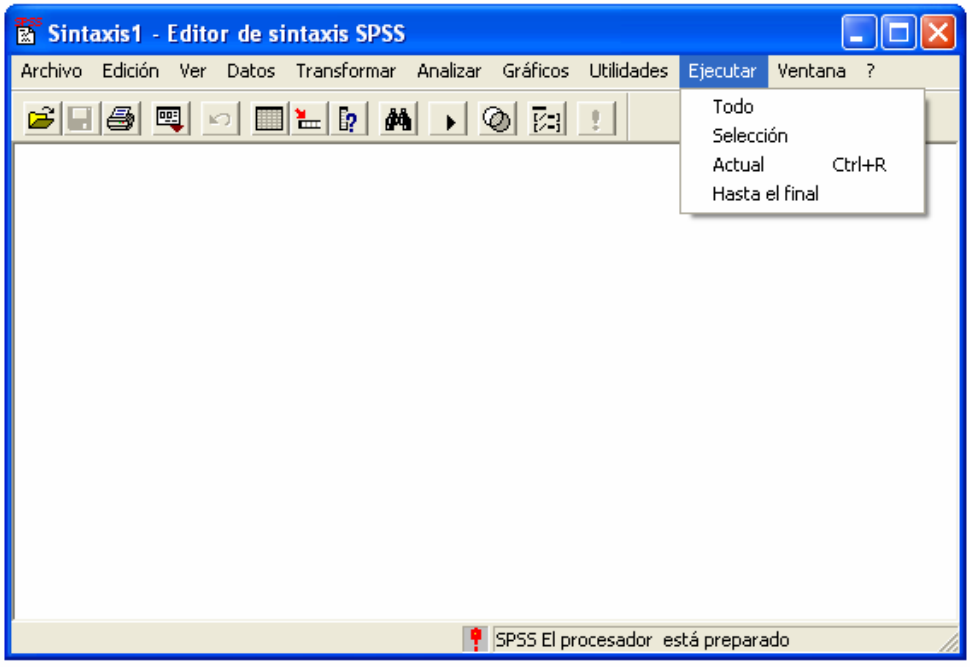

FIGURA 1: Pantalla del Editor de sintaxis.

2.- REGLAS DE SINTAXIS

Al editar y escribir la sintaxis de los comandos hay que tener en cuenta una serie de reglas:

- Cada comando debe empezar en una línea nueva y terminar con un punto (.).
- Los nombres de variable deben escribirse completos.
- El texto incluido entre apóstrofos o comillas debe ir contenido en una sola línea.
- La mayoría de los subcomandos están separados por barras inclinadas (/). La barra inclinada que precede al primer subcomando de un comando generalmente es opcional.
- Debe utilizarse un punto (.) para indicar decimales, independientemente de la configuración regional de Windows.
- Los nombres de variable que terminen en un punto pueden causar errores en los comandos creados por los cuadros de diálogo. No es posible crear nombres de variable de este tipo en los cuadros de diálogo y en general deben evitarse.

La sintaxis de comandos no distingue las mayúsculas de las minúsculas y permite el uso de abreviaturas de tres o cuatro letras en la mayoría de las especificaciones de los comandos. Podemos usar tantas líneas como deseemos para especificar un único comando. Podemos añadir espacios o líneas de separación en casi cualquier punto donde se permita un único espacio en blanco, como alrededor de las barras inclinadas, los paréntesis, los operadores aritméticos o entre los nombres de variable. Por ejemplo:

Ejemplo 1:

```
FREQUENCIES
   VARIABLES=raza sexo
   /PERCENTILES=25 50 75
   /BARCHART.
```

es equivalente a:

```
freq var=raza sexo /percent=25 50 75 /bar.
```

3.- COMANDOS

Los comandos que se van a tratar en el presente manual son mayoritariamente aquellos a los que no se puede acceder con la opción “pegar” que aparece en la mayor parte de los cuadros de diálogo al trabajar con ventanas.

OPERACIONES CON ARCHIVOS

- Abrir un archivo SPSS:

```
GET
  FILE=’Ruta de acceso al archivo’.
```

Ejemplo 2:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
```

Hay que tener en cuenta que si en el archivo de sintaxis la ruta no está en una misma línea no se ejecuta el comando, salvo que indiquemos al programa que la ruta se encuentra en líneas diferentes. Para ello basta con encerrar entre comillas simples las distintas partes de la ruta y unirlas mediante un signo “+” colocado al final de cada una de las líneas excepto en la última.

Ejemplo 3:

```
GET
  FILE='C:\Archivos de programa\SPSS\En'+
 'cuesta general USA 1991.sav'.
```

Ejemplo 4:

```
GET
  FILE='C:\Archivos de programa\SPSS\En'+
 'cuesta genera'+
 'l USA 1991.sav'.
```

Utilizando los subcomandos DROP y KEEP dentro de GET FILE podemos abrir un fichero que contenga únicamente un conjunto de variables del archivo indicado. Con DROP= NombreVariable NombreVariable…. podemos especificar las variables que no queremos meter en el fichero y con KEEP= NombreVariable NombreVariable…. las que sí queremos incluir. Las variables no mencionadas en el subcomando KEEP no serán incluidas en el archivo.

Ejemplo 5:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'
  /KEEP=sexo raza región feliz.
```

En este ejemplo se abre un archivo que contiene únicamente las variables sexo, raza, región y feliz del archivo original “Encuesta general USA 1991.sav”.

Ejemplo 6:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'
  /DROOP=sexo raza región feliz.
```

En este ejemplo se abre un archivo que contiene todas las variables del archivo original, “Encuesta general USA 1991.sav”, excepto sexo, raza, región y feliz.

- Guardar cambios:

SAVE OUTFILE='nombre del archivo y ruta en la que se creará'
/COMPRESSED.

Ejemplo 7:

```
SAVE OUTFILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991 Modificada.sav'
 /COMPRESSED.
```

Hay que tener en cuenta que si en el archivo de sintaxis la ruta no está en una misma línea no se ejecuta el comando, salvo que indiquemos al programa que la ruta se encuentra en líneas diferentes. Para ello basta con encerrar entre comillas simples las distintas partes de la ruta y unirlas mediante un signo “+” colocado al final de cada una de las líneas excepto en la última.

Ejemplo 8:

```
SAVE OUTFILE='C:\Archivos de programa\SPSS\Encuesta'+
 'general USA 1991 Modificada.sav'
 /COMPRESSED.
```

Ejemplo 9:

```
SAVE OUTFILE='C:\Archivos de programa\SPSS\Encuesta'+
 'general USA 19'+
 '91 Modificada.sav'
 /COMPRESSED.
```

Mediante el subcomando UNSELECTED dentro de SAVE OUTFILE es posible guardar solamente un conjunto de casos (registros) previamente seleccionados. Con UNSELECTED = DELETE únicamente se salvan los casos seleccionados y con UNSELECTED = RETAIN se guardan todos los registros, así como el filtro de selección de casos. Este subcomando no realiza ninguna acción si el fichero de datos no contiene una selección de casos previa.

Ejemplo 10:

```
*Abrimos el archivo.
```

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
*Seleccionamos únicamente los casos correspondientes a mujeres.
USE ALL.
COMPUTE filter_$=(SEXO = 2).
VARIABLE LABEL filter_$ 'SEXO = 2 (FILTER)'.
VALUE LABELS filter_$  0 'No seleccionado' 1 'Seleccionado'.
FORMAT filter_$ (f1.0).
FILTER BY filter_$.
EXECUTE .
*Guardamos únicamente los casos correspondientes a mujeres.
SAVE OUTFILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991 reducida.sav'
 /UNSELECTED = DELETE
 /COMPRESSED.
*Abrimos el nuevo archivo para comprobar que sólo contiene los registros correspondientes a
*mujeres.
GET
    FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991 reducida.sav'.
```

Utilizando los subcomandos DROP y KEEP dentro de SAVE OUTFILE podemos guardar únicamente un conjunto de variables del archivo activo. Con DROP= NombreVariable NombreVariable…. podemos especificar las variables que no queremos salvar en el fichero y con KEEP= NombreVariable NombreVariable…. las que sí queremos guardar. Las variables no mencionadas en el subcomando KEEP no serán salvadas en el archivo.

Ejemplo 11:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
SAVE OUTFILE= 'C:\Archivos de programa\SPSS\Encuesta general USA 1991 Modifica-
da1.sav'
/KEEP=sexo raza región feliz
/COMPRESSED.
```

En este ejemplo se crea un nuevo archivo llamado “Encuesta general USA 1991 Modificada1.sav” que contiene únicamente las variables sexo, raza, región y feliz del archivo original “Encuesta general USA 1991.sav”.

Ejemplo 12:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
SAVE OUTFILE= 'C:\Archivos de programa\SPSS\Encuesta general USA 1991 Modifica-
da2.sav'
```

/DROOP=sexo raza región feliz

/COMPRESSED.

En este ejemplo se crea un nuevo archivo llamado “Encuesta general USA 1991 Modificada2.sav” que contiene todas las variables del archivo original “Encuesta general USA 1991.sav” excepto sexo, raza, región y feliz..

- Leer ficheros de texto:

El comando DATA LIST permite leer desde SPSS archivos de texto externos. Para que los datos incluidos en estos archivos de texto sean leídos correctamente por el SPSS con este comando, deben tener el siguiente formato:

- Cada una de las filas del fichero se referirá a un registro, es decir, a un único caso.
- En todos los registros los valores de cada una de las variables deben ocupar posiciones fijas de columna.

```
DATA LIST FILE='nombre del archivo de texto y ruta en la que se encuentra'
/NombreVariable_1 NºColumnasQueOcupa (TipoVariable)
……
NombreVariable_n NºColumnasQueOcupa (TipoVariable).
EXECUTE.
```

Al leer el archivo con el DATA LIST debemos indicar el nombre de cada una de las variables, así como el número de columnas que ocupa (por ejemplo: 3-6, es decir, de la columna 3 a la 6). A continuación, entre paréntesis indicamos el tipo de variable que es. Si no se incluye este paréntesis el programa entiende por defecto que la variable es numérica.

Para indicar que la variable es una variable de cadena debemos escribir la letra A dentro del paréntesis. F lee una variable numérica sin decimales y F,n indica variable numérica con n decimales. EDATE hace referencia a una variable de fecha con formato dd/mm/yy. Se pueden indicar otros muchos formatos de fecha como DATE (formato dd-mmm-yyyy), ADATE (formato mm/dd/yyyy) etc.

Estos son los principales formatos de variable que soporta SPSS, pero existen más. Para ver una lista completa de formatos podemos acudir a la sección “variables” del apartado “universals” del SPSS Command Syntax Reference.

Ejemplo 13:

Supongamos que tenemos un archivo de texto llamado DataList1.txt con los siguientes datos:

```
7,0 2 21/03/91 Pérez     01/06/2006
2,1 1 01/01/90 Jiménez   01/06/2006
5,5 1 11/06/91 González 01/06/2006
9,7 2 22/02/90 López     01/06/2006
```

La sintaxis SPSS que debemos ejecutar para leer el archivo de texto anterior es:

```
DATA LIST FILE='C:/Documents and Settings/Cecilia/Mis documentos/Manuales/SPSS/SPSS Sintaxis/DataList1.txt'
/NotaMed 1-3 (F,1)
 Sexo 5
 FechaNac 7-14 (EDATE)
 Apellido 16-23 (A)
 FechaExam 25-34 (DATE).
EXECUTE.
```

Cuando queremos leer desde SPSS un archivo de texto con datos de variables únicamente numéricas, dicho archivo puede tener un formato menos rígido. Empleando el subcomando FREE no es necesario que cada variable ocupe posiciones de columna fijas. Los valores de cada una de las variables no deben estar en posiciones fijas, sino únicamente separados por un espacio. Tampoco es necesario que cada registro se encuentre en una línea del archivo, sino que es posible colocarlos en una misma línea, aunque deben estar ordenados respecto a las variables de la misma forma en todos los registros (siguiendo el orden de variables indicado en el comando).

```
DATA LIST FILE='nombre del archivo de texto y ruta en la que está' FREE
/NombreVariable_1 …… NombreVariable_n.
EXECUTE.
```

Ejemplo 14:

Supongamos que tenemos un archivo de texto llamado DataList2.txt con los siguientes datos:

9 2 1997 2,5 10 1 1996 5 9 2 1997 9,15 8 1 1998 8 10 2 1996 6,5 9 1 1997 4

correspondientes a los valores de las variables edad, sexo, año de nacimiento y nota media.

La sintaxis SPSS que debemos ejecutar para leer el archivo de texto anterior es:

```
DATA LIST FILE='C:/Documents and Settings/Cecilia/Mis documentos/Manuales/SPSS/SPSS Sintaxis/DataList2.txt' FREE
/Edad Sexo AnyoNac NotaMed.
EXECUTE.
```

- Crear archivos de datos:

```
DATA LIST
/NombreVariable_1 NºColumnasQueOcupa (TipoVariable)…… NombreVariable_n NºColumnasQueOcupa (TipoVariable).
BEGIN DATA
…… Valores de las variables en sus correspondientes posiciones de columna fijas y con un registro por línea……
END DATA.
```

Este comando permite crear archivos de datos desde la sintaxis SPSS. Dentro del comando BEGIN DATA tenemos que incluir los datos que contendrá el archivo. Estos datos deben tener el siguiente formato:

- Cada una de las filas se referirá a un registro, es decir, a un único caso.
- En todos los registros los valores de cada una de las variables deben ocupar posiciones fijas de columna.

Al igual que en el apartado anterior, al crear un archivo con el DATA LIST debemos indicar el nombre de cada una de las variables, así como el número de columnas que ocupa (por ejemplo: 3-6, es decir, de la columna 3 a la 6). A continuación, entre paréntesis indicamos el tipo de variable que es. Si no se incluye este paréntesis el programa entiende por defecto que la variable es numérica.

Para indicar que la variable es una variable de cadena debemos escribir la letra A dentro del paréntesis. F lee una variable numérica sin decimales y F,n indica variable numérica con n decimales. EDATE hace referencia a una variable de fecha con formato dd/mm/yy. Se pueden indicar otros muchos formatos de fecha como DATE (formato dd-mm-yyyy), ADATE (formato mm/dd/yyyy) etc.

Estos son los principales formatos de variable que soporta SPSS, pero existen más. Para ver una lista completa de formatos podemos acudir a la sección “variables” del apartado “universals” del SPSS Command Syntax Reference.

Ejemplo 15:

La sintaxis SPSS que debemos ejecutar para crear el archivo de datos utilizado en el ejemplo 13 es:

```
DATA LIST
/NotaMed 1-3 (F,1) Sexo 5 FechaNac 7-14 (EDATE) Apellido 16-23 (A) FechaExam 25-34
(DATE).
BEGIN DATA
7,0 2 21/03/91 Pérez    01/06/2006
2,1 1 01/01/90 Jiménez  01/06/2006
5,5 1 11/06/91 González 01/06/2006
9,7 2 22/02/90 López    01/06/2006
END DATA.
```

Cuando queremos crear un archivo de datos con variables únicamente numéricas, los valores especificados en el BEGIN DATA pueden tener un formato menos rígido. Empleando el subcomando FREE no es necesario que cada variable ocupe una posición de columna fija. Los valores de cada una de las variables no deben estar en posiciones fijas, sino únicamente separados por un espacio. Tampoco es necesario que cada registro se encuentre en una línea, sino que es posible colocarlos en la misma, aunque deben estar ordenados respecto a las variables de la misma forma en todos los registros (en el orden indicado en el comando DATA LIST).

```
DATA LIST FREE
/NombreVariable_1 …… NombreVariable_n .
BEGIN DATA
…… Valores de las variables ordenados y separados por un espacio……..
END DATA.
```

Ejemplo 16:

La sintaxis SPSS que debemos ejecutar para leer el archivo de texto anterior es:

```
DATA LIST FREE
/Edad Sexo AnyoNac NotaMed.
BEGIN DATA
9 2 1997 2,5 10 1 1996 5 9 2 1997 9,15 8 1 1998 8 10 2 1996 6,5 9 1 1997 4
END DATA.
```

INFORMACIÓN SOBRE EL ARCHIVO

- Mostrar documentos:

```
DOCUMENT texto.
```

El comando DOCUMENT almacena un texto de cualquier longitud en el fichero de datos SPSS activo. Podemos ver el contenido de los documentos de texto generados empleando el comando DISPLAY DOCUMENT. Este comando muestra todos los documentos que se hayan creado. También podemos borrar toda la información utilizando el comando DROP DOCUMENT. Hay que tener en cuenta que borra todos los documentos existentes en el archivo de trabajo, no es posible seleccionar alguno en concreto para eliminar solo ése.

Ejemplo 17:

```
DOCUMENT Este es el contenido de document.
DISPLAY DOCUMENT.
```

Al ejecutar este ejemplo obtenemos el siguiente resultado en pantalla:

Documento

1(a)	DOCUMENT Este es el contenido de document.

a Introducido 05-Sep-2006

Hay que tener en cuenta que mientras no ejecutemos el comando DROP, “Este es el contenido de document” seguirá siendo el valor de document. Además, si ejecutamos un nuevo DOCUMENT con otro texto este se añadirá al anterior y al hacer DISPLAY DOCUMENT aparecerán ambos. Hagámoslo:

DOCUMENT Introducimos un nuevo documento de texto.

DISPLAY DOCUMENT.

Obtenemos: **Documento**

1(a)	DOCUMENT Este es el contenido de document.
2(a)	DOCUMENT Introducimos un nuevo documento de texto.

a Introducido 05-Sep-2006

*Eliminamos todos los datos de DOCUMENT:

DROP DOCUMENT.

DISPLAY DOCUMENT.

Obtenemos:

Advertencia

El archivo de trabajo no contiene documentos.

- Mostrar diccionario:

DISPLAY DICCTIONARY.

Los ficheros de datos SPSS son archivos que contienen, además de los datos, un diccionario. El diccionario incluye el nombre de cada variable incluida en el archivo de datos, así como sus características (etiquetas, valores perdidos, etc.).

Ejemplo 18:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
DISPLAY DICCTIONARY.
```

La información incluida en el diccionario puede mostrarse ordenada alfabéticamente por nombre de variable, y también es posible especificar variables concretas sobre las que deseamos conocer sus características. Para ello basta con emplear los subcomandos SORTED y VARIABLES=NombreVariable1 …. NombreVariableN respectivamente.

Cuando no se incluye el subcomando SORTED, la información sobre las variables incluida en el diccionario aparece en el orden en el que aparecen dichas variables en el archivo.

Ejemplo 19:

```
DISPLAY SORTED DICCTIONARY.
```

Ejemplo 20:

```
*El siguiente comando solicita la información contenida en el diccionario referida únicamente a
*las variables sexo y raza.
DISPLAY DICCTIONARY VARIABLES= sexo raza.
```

OPERACIONES CON VARIABLES

- Crear variables cadena:

```
STRING NombreVariable (ANºcaracteres).
EXECUTE.
```

Ejemplo 21:

*Creamos una variable cadena llamada VarCadena con una longitud de 25 caracteres.
STRING VarCadena (A25).
EXECUTE.
*Creamos en un mismo comando dos variables cadena con una longitud de 5 caracteres.
STRING VarCadena1 VarCadena2 (A5).
EXECUTE.
*Creamos en un mismo comando dos variables cadena con diferentes longitudes.
STRING VarCadena3 (A7) VarCadena4(A3).
EXECUTE.

- Crear variables numéricas:

```
NUMERIC NombreVariable (FNºcifras.Nºdecimales).
EXECUTE.
```

Ejemplo 22:

*Creamos una variable numérica llamada VarNum de anchura 8 y con dos decimales.
NUMERIC VarNum (F8.3).
EXECUTE.
*Creamos en un mismo comando dos variables numéricas con el mismo formato (anchura 8
*y ningún decimal).
NUMERIC VarNum1 VarNum2 (F8.0).
EXECUTE.
*Creamos en un mismo comando dos variables numéricas con diferentes formatos.
NUMERIC VarNum3 (F7.1) VarNum4 (F4.2).
EXECUTE.

- Borrar variables:

```
DELETE VARIABLES NombreVariable.
```

Ejemplos 23:

*Eliminamos la variable VarCadena del archivo de datos.

DELETE VARIABLES VarCadena.

*Eliminamos en un mismo comando distintas variables de diferentes tipos.

DELETE VARIABLES VarCadena1 VarCadena2 VarCadena3 VarCadena4 VarNum1 VarNum2 VarNum3 VarNum4.

- Renombrar variables:

RENAME VARIABLES (NombreAntiguo=NombreNuevo).

Ejemplo 24:

*Renombramos la variable municipio y le damos el nombre de municipi.

RENAME VARIABLES (hermanos=hnos).

Es posible renombrar varias variables en el mismo comando de sintaxis de la siguiente forma:

RENAME VARIABLES (NombreAntiguo1=NombreNuevo1)…… (NombreAntiguoN=NombreNuevoN).

- Cambiar el formato de representación de una variable existente:

FORMATS NombreVariable (NuevoFormato).

Ejemplo 25:

*Cambiamos el formato de representación de la variable región.

*Le quitamos los decimales.

FORMATS región (F8.0).

Este comando no se puede emplear con variables cadena. Tampoco sirve para cambiar una variable numérica a cadena o viceversa. No es posible efectuar estas operaciones con comandos de sintaxis sin, por ejemplo, recodificar la variable en una nueva de diferente tipo.

-Cambiar el nivel de medida de una variable:

VARIABLE LEVEL NombreVariable (tipoMedida).

En el programa SPSS podemos seleccionar tres diferentes niveles de medida para las variables: escala, nominal y ordinal. Por regla general escala es el nivel seleccionado cuando la variable es cuantitativa, nominal cuando es categórica y ordinal cuando se trata de una variable categórica tal que es posible ordenar todas sus distintas categorías.

Ejemplo 26:

*Modificamos el nivel de medida de la variable VarNum (originalmente escala) para pasarlo a *ordinal.

VARIABLE LEVEL VarNum (Ordinal).

-Etiquetar variables:

```
VARIABLE LABELS NombreVariable 'Etiqueta'.
```

Este comando nos permite etiquetar variables, es decir, nos permite asignar a las variables un texto explicativo que aporte más información que su nombre.

Ejemplo 27:

*Etiquetamos la variable VarNum.

VARIABLE LABELS VarNum 'Variable de prueba'.

- Etiquetar las categorías de una variable:

```
VALUE LABELS
/NombreVariable
  1 'etiqueta1'
  2 'etiqueta2'
  ……..
  N 'etiquetaN'.
```

Empleando este comando podemos etiquetar los distintos valores que toma una variable categórica, es decir, podemos nombrar las distintas cualidades cuyo valor no sea directamente interpretable.

Ejemplo 28:

*Etiquetamos las categorías de la variable prob2.

VALUE LABELS

/prob2

1 'salud'
2 'Financieros'
3 'Falta de servicios básicos'
4 'Familia'
5 'Personales'
6 'Legales'.

Cuando empleamos el comando VALUE LABELS se suprimen todas las etiquetas de valor especificadas en la variable a la que se aplica el comando. Sin

embargo existe otro comando que nos permite añadir o cambiar etiquetas de valor sin afectar a las ya definidas. Este comando es el siguiente:

```
ADD VALUE LABELS
/NombreVariable
  1 'etiqueta1'
  2 'etiqueta2'
  ……..
  n 'etiqueta_n'.
```

Ejemplo 29:

```
*Añadimos la etiqueta de la séptima categoría de la variable prob2.
ADD VALUE LABELS
/prob2
 7 'Diversos'.
```

Es posible etiquetar categorías iguales de varias variables en el mismo comando de sintaxis.

Ejemplo 30:

```
*Etiquetamos las categorías de las variables prob3 y prob4.
VALUE LABELS
/prob3 prob4
  1 'salud'
  2 'Financieros'
  3 'Falta de servicios básicos'
  4 'Familia'
  5 'Personales'
  6 'Legales'
  7 'Diversos'.
```

-Establecer valores perdidos:

```
MISSING VALUES NombreVariable (ValorADeterminarComoPerdido, ValorADeterminarComoPerdido,….).
```

Este comando permite definir como valores perdidos por el usuario determinados valores o categorías de una variable. De esta forma el tratamiento de dichos valores o categorías de la variable en los análisis que se realicen será el de missing, no siendo incluidos en los mismos.

Ejemplo 31:

```
*Calculamos la tabla de frecuencias de la variable región.
```

```
FREQUENCIES
  VARIABLES=región
  /ORDER=  ANALYSIS .
```

Obtenemos el siguiente resultado:

región

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válidos	1 Nor-Este	679	44,8	44,8	44,8
	2 Sur-Este	415	27,4	27,4	72,1
	3 Oeste	423	27,9	27,9	100,0
	Total	1517	100,0	100,0	

```
*Declaramos como valor perdido definido por el usuario a la categoría oeste de la variable re-
*gión.

MISSING VALUES región (3).

*Calculamos la tabla de frecuencias de la variable región tras declarar la categoría oeste como
*valor perdido por el usuario.

FREQUENCIES
  VARIABLES=región
  /ORDER=  ANALYSIS .
```

Obtenemos el siguiente resultado:

región

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válidos	1 Nor-Este	679	44,8	62,1	62,1
	2 Sur-Este	415	27,4	37,9	100,0
	Total	1094	72,1	100,0	
Perdidos	3 Oeste	423	27,9		
Total		1517	100,0		

Empleando el comando MISSING VALUES también podemos eliminar todos o parte de los valores perdidos definidos por el usuario.

Ejemplo 32:

```
*Eliminamos el  valor perdido 9 definido por el usuario de la variable feliz:

MISSING VALUES feliz (0,8).
```

Ejemplo 33:

```
*Borramos todos los valores perdidos definidos por el usuario para la variable feliz.

MISSING VALUES feliz ().

*Borramos todos los valores perdidos definidos por el usuario para la variable región.

MISSING VALUES región ().
```

-Referencias a variables consecutivas:

NombrePrimeraVariable TO NombreÚltimaVariable

SPSS permite hacer referencia a un conjunto de variables sin necesidad de escribir los nombres de cada una de ellas, sin embargo dichas variables deben ser consecutivas en el archivo de trabajo. De esta manera solo necesitamos escribir los nombres de la primera y la última de las variables del conjunto.

Ejemplo 34:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
SAVE OUTFILE= 'C:\Archivos de programa\SPSS\Encuesta general USA 1991 Modifica-
da2.sav'
/DROOP=sexo TO feliz
/COMPRESSED.
```

En este ejemplo se crea un nuevo archivo llamado “Encuesta general USA 1991 Modificada.sav” que contiene todas las variables del archivo original “Encuesta general USA 1991.sav” excepto sexo, raza, región y feliz.

```
*Abrimos el nuevo archivo para comprobar que sólo contiene los registros correspondientes a
*mujeres.
GET
    FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991 Modificada2.sav'.
```

Ejemplo 35:

```
*Eliminamos las variables salud1 hasta salud9.
DELETE VARIABLES salud1 TO salud9.
```

Ejemplo 36:

```
*Creamos 9 variables numéricas llamadas hijos1, hijos2,…, hijos9 de anchura 2 y sin
*decimales.
NUMERIC hijos1 TO hijos9 (F2.0).
```

OTROS

- Mostrar sintaxis:

SET printback=yes/off.

Este comando nos permite indicar al programa que la sintaxis correspondiente al procedimiento empleado aparezca (printback=yes) o no (printback=off) en el visor junto con los resultados del mismo.

- Titular páginas de resultados:

TITLE 'Texto que queremos que figure'.

O equivalentemente:

TITLE Texto que queremos que figure.

El comando TITLE inserta el texto incluido a continuación del mismo en el encabezado de cada página de resultados de SPSS.

El título puede incluir cualquier carácter, aunque solo puede tener una longitud de 60 caracteres.

Ejemplo 37:

```
TITLE 'Ejemplos de sintaxis del curso'.
```

**Una vez ejecutado el comando TITLE de este ejemplo, al imprimir los resultados el texto "Ejemplos de sintaxis del curso" aparecerá en el encabezado de cada página impresa.

- Insertar comentarios:

***Comentario.**

O equivalentemente:

COMMENT Comentario.

Ejemplo 38:

```
*Este es un comentario.
COMMENT Este también es un comentario.
```

Hay que tener cuidado con los comentarios, puesto que si no acabamos un comentario con un punto, el programa entiende que las líneas que siguen al mismo continúan siendo comentarios hasta que encuentra el terminador del comando (el punto). De esta forma, si olvidamos el punto final de un comentario y a continuación incluimos algún comando, este no se ejecutará.

Cuando queremos incluir un comentario dentro de un comando debemos utilizar la siguiente sintaxis:

/* Comentario */

Ejemplo 39:

```
FREQUENCIES
  VARIABLES=hermanos
  /PERCENTILES=25 /*Comentario dentro de un comando*/ 50 75
  /BARCHART.
```

- Transformaciones temporales:

TEMPORARY

Este comando indica el comienzo de transformaciones temporales que solo tendrán efecto para el proceso inmediatamente posterior. De esta manera las variables creadas inmediatamente después del comando TEMPORARY son variables temporales. Asimismo, cualquier modificación realizada sobre variables del archivo nada más ejecutar el comando que nos ocupa también será temporal.

Ejemplo 40:

```
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.
TEMPORARY.
RECODE
  región
  (1=1)  (2=1)  (3=2)  INTO  EsteOeste .

FREQUENCIES
  VARIABLES=EsteOeste
  /ORDER=  ANALYSIS .

FREQUENCIES
  VARIABLES=EsteOeste
  /ORDER=  ANALYSIS .
```

Ejemplo 41:

```
SORT CASES BY sexo .
TEMPORARY.
SPLIT FILE
  LAYERED BY sexo .

FREQUENCIES
  VARIABLES=raza
  /ORDER=  ANALYSIS.

FREQUENCIES
  VARIABLES=raza
  /ORDER=  ANALYSIS.
```

- Execute:

EXECUTE.

El comando EXECUTE fuerza la lectura de los datos y ejecuta transformaciones pendientes. Existen comandos de SPSS que deben ir seguidos del EXECUTE, puesto que de otra manera quedan transformaciones pendientes y podemos tener problemas a la hora de ejecutar otros comandos.

Ejemplo 42:

STRING VarCadena (A25).

DELETE VARIABLES sexo.

Al ejecutar estos dos comandos obtenemos el siguiente resultado:

Advertencia

No se pueden eliminar variables si hay transformaciones pendientes.
Este comando no se ha ejecutado.

Ejemplo 43:

```
DATA LIST FILE='C:/Documents and Settings/Cecilia/Mis documentos/Manuales/SPSS/SPSS Sintaxis/DataList2.txt' FREE
/Edad Sexo AnyoNac NotaMed.
```

Si queremos ejecutar varias veces consecutivas un mismo comando que necesita ir seguido de EXECUTE, no es necesario incluirlo al final de cada uno de ellos. Basta con incluirlo tras el último de los comandos a ejecutar. De hecho es más recomendable utilizar en estos casos un único EXECUTE, ya que de otra manera se lee todo el archivo de datos cada vez que aparece el comando y, si el archivo es grande, el tiempo de ejecución será mayor.

Ejemplo 44:

```
STRING VarCadena (A25).
STRING VarCadena1 VarCadena2 (A5).
STRING VarCadena3 (A7) VarCadena4(A3).
EXECUTE.
```

- Print:

PRINT / NombreVariable1 NombreVariable2 ……. NombreVariableN.

EXECUTE.

El comando PRINT muestra el valor de cada uno de los casos de las variables incluidas en el mismo. Todos los valores que toman las variables indicadas en el comando serán presentados en el visor de resultados.

Ejemplo 45:

```
PRINT / Edad AnyoNac.
EXECUTE.
```

Si queremos que el comando PRINT nos muestre los valores de todas las variables del archivo de datos basta con sustituir la enumeración de variables por el subcomando ALL.

Ejemplo 46:

```
PRINT / ALL.
EXECUTE.
```

Con el subcomando RECORDS indicamos el número de líneas que ocupará cada caso en el visor de resultados.

Ejemplo 47:

```
PRINT RECORDS=3 / ALL.
EXECUTE.
```

Al ejecutar este ejemplo obtenemos la siguiente salida:

```
    9,00     2,00  1997,00     2,50      .


   10,00     1,00  1996,00     5,00      .


    9,00     2,00  1997,00     9,15      .


    8,00     1,00  1998,00     8,50      .


   10,00     2,00  1996,00     6,00      .


    9,00     1,00  1997,00     4,00      .
```

En este ejemplo indicamos al programa que cada caso debe ocupar tres líneas. En la primera fila aparecen los valores de todas las variables del archivo, y las dos líneas siguientes no presentan dato alguno, están vacías.

Existe la posibilidad de indicar las variables cuyos valores queremos que aparezcan en cada una de las líneas indicadas en RECORDS. Para ello basta con incluir un slash (/) por línea, y a continuación las variables cuyos valores queremos que se generen en la misma.

Ejemplo 48:

```
PRINT RECORDS=3 / edad sexo
                  / AnyoNac NotaMed
                  / .
EXECUTE.
```

Ahora obtenemos la siguiente salida:

```
   9,00     2,00
1997,00     2,50

  10,00     1,00
1996,00     5,00

   9,00     2,00
1997,00     9,15

   8,00     1,00
1998,00     8,50

  10,00     2,00
1996,00     6,00

   9,00     1,00
1997,00     4,00
```

En este nuevo ejemplo, los datos de las variables edad y sexo aparecen en la primera línea, el año de nacimiento y la nota media en la segunda, y la tercera fila no presenta valores, está vacía.

Es recomendable dejar la última fila de cada caso vacía para poder distinguir fácilmente un registro de otro.

ESTRUCTURAS DE PROGRAMACIÓN

- Do repeat:

```
DO REPEAT Stand-inVariable = NombreVariable1…..NombreVariableN.
Comando1.
……..
Comando n.
END REPEAT.
EXECUTE.
```

La estructura DO REPEAT nos permite repetir varias veces el mismo grupo de transformaciones (comandos) sobre un conjunto de variables, reduciendo el número de comandos que es necesario escribir. Stand-inVariable es una variable que recoge el conjunto de variables al que se van a aplicar las transformaciones. Hay que tener en cuenta que esta variable solo existe dentro del comando DO REPEAT.

Ejemplo 49:

```
*Creamos un Nuevo fichero.
DATA LIST LIST /var1 var2 var3 var4 var5 var6.
BEGIN DATA
3 3 3 3 3 3
2 2 2 2 2 2
1 1 1 1 1 1
0 0 0 0 0 0
END DATA.

*Empleamos el commando Do Repeat para cambiar los valores de las variables var1 y var4.
DO REPEAT v=var1 var6.
  COMPUTE v=99.
END REPEAT.
EXECUTE.
```

Hay que tener en cuenta que no se reduce el número de comandos que SPSS ejecuta, solo el número de comandos que escribimos. Si queremos ver el número de comandos que se ejecutan basta con incluir el subcomando PRINT entre END REPEAT y el punto indicador de fin de comando.

Ejemplo 50:

```
*Creamos un Nuevo fichero.
DATA LIST LIST /var1 var2 var3 var4 var5 var6.
BEGIN DATA
3 3 3 3 3 3
2 2 2 2 2 2
1 1 1 1 1 1
0 0 0 0 0 0
END DATA.

*Empleamos el commando Do Repeat para cambiar los valores de las variables var1 y var4.
DO REPEAT v=var1 var6.
  COMPUTE v=99.
END REPEAT PRINT.
EXECUTE.
```

También es posible definir en el mismo comando varias variables Stand-in, teniendo en cuenta que tendrán que estar separadas unas de otras mediante un slash (/).

Ejemplo 51:

```
DO REPEAT v13=var1 TO var3
                / v46=var4 TO var6.
  COMPUTE v13=99.
  COMPUTE v46=00.
END REPEAT PRINT.
EXECUTE.
```

Además es posible definir una Stand-in variable que, en lugar de un conjunto de variables, contenga un conjunto de valores. De esta forma, en los comandos podemos asignar una a otra.

Ejemplo 52:

```
DATA LIST LIST /var1 var2 var3 var4.
BEGIN DATA
3 3 3 3
2 2 2 2
1 1 1 1
END DATA.

DO REPEAT v=var1 TO var4 /val=1 3 5 7.
  COMPUTE v=val.
END REPEAT PRINT.
EXECUTE.
```

Un ejemplo de la utilidad de este comando es su empleo para definir e inicializar (dar valores) un conjunto de nuevas variables numéricas.

Ejemplo 53:

```
DO REPEAT R=var5 to var10.
  COMPUTE R=0.
END REPEAT PRINT.
EXECUTE.
```

Existe la posibilidad de incluir todas las variables del archivo en una variable Stand-in, sin más que asignarle ALL en lugar de los nombres de todas las variables del fichero.

Ejemplo 54:

```
DO REPEAT v=ALL.
  COMPUTE v=999.
END REPEAT PRINT.
EXECUTE.
```

Los comandos SPSS más utilizados que se pueden emplear dentro de una estructura DO REPEAT son los siguientes: COMPUTE, RECODE, IF, COUNT, SELECT IF, VECTOR, STRING, NUMERIC, DATA LIST, MISSING VALUES, LOOP, DO IF, PRINT y FORMATS.

- If:

IF (Expresión_lógica) variable=expresión.

EXECUTE.

Este comando ejecuta una transformación (variable=expresión) en función de la evaluación de una expresión lógica. La transformación puede crear una nueva variable o modificar los valores de una ya existente.

Una expresión lógica es toda aquella expresión cuya evaluación da como resultado verdadero o falso[2] (Ejm: var1 < var2, var1 <> var2, etc.). La transformación se efectúa solo en aquellos registros en los que la expresión lógica da como resultado verdadero.

Es posible crear o modificar variables tanto numéricas como de cadena, aunque en este último caso si la variable no existe en el archivo de datos hay que crearla primero empleando el comando STRING.

Ejemplo 55:

```
*Abrimos el archivo Encuesta general USA 1991.sav.
GET
  FILE='C:\Archivos de programa\SPSS\Encuesta general USA 1991.sav'.

*Creamos una variable de cadena.
STRING TipoFamilia (A20).
EXECUTE.

*Asignamos el valor "Familia numerosa" a la variable TipoFamilia cuando la variable hijos sea
mayor o igual que 3.
IF(hijos>=3)  TipoFamilia='Familia numerosa'.
EXECUTE.
```

[2] Operadores que pueden ser empleados en las expresiones lógicas:

EQ o = “Igual”	NE o <> “Distinto”	LT o < “Menor que”
Le o <= “Menor o igual que”	GT o >”Mayor que”	GE o >= “Mayor o igual que”
AND o & “Operador y lógico”	OR o | “Operador o lógico”	Not “Negación lógica

```
*Asignamos el valor "Familia NO numerosa" a la variable TipoFamilia cuando la variable hijos
sea menor que 3.
IF(hijos<3)  TipoFamilia='Familia NO numerosa'.
EXECUTE.
```

Ejemplo 56:

```
*Creamos una nueva variable numérica con valor 1 si el individuo dice haber tenido algún tipo
*de problema (es decir si no encontramos un valor perdido en la variable prob1) y 0 si no lo ha
*tenido (la variable prob1 presenta un valor perdido).
IF(not missing(prob1))  Problemas=1.
EXECUTE.

IF(missing(Problemas))  Problemas=0.
EXECUTE.
```

- Do if:

```
DO IF (Expresión_lógica).
Comando_1
…….
Comando_n
END IF.
```

El comando DO IF ejecuta una serie de comandos en función de la evaluación de una expresión lógica. Si dicha expresión lógica no es verdadera no se ejecuta ninguno de los comandos.

Si los comandos incluidos en el DO IF requieren EXECUTE, este comando deberá incluirse una sola vez a continuación del END IF.

Ejemplo 57:

```
DO IF (sexo=1).
COUNT
  NumProbSaludFem = salud1 salud2 salud3 salud4 salud5 salud6 salud7 salud8 salud9  (1)  .
VARIABLE LABELS   NumProbSaludFem 'Número de problemas de salud que tienen las mujeres entrevistadas' .
COUNT
  NumProblFem = prob2 prob1 prob3 prob4  (1 thru 7)  .
VARIABLE LABELS NumProblFem 'Número de problemas que tienen las mujeres entrevistadas' .
END IF.
EXECUTE.
```

Existe la posibilidad de indicar al programa que ejecute una serie de comandos alternativos cuando la expresión lógica evaluada no es cierta. Para ello basta con incluir dichos comandos alternativos a continuación del comando ELSE tal y como se muestra a continuación.

```
DO IF (Expresión_lógica).
Comando_1
…….
Comando_n
ELSE.
Comando_n+1
…….
Comando_m
END IF.
```

Ejemplo 58:

```
*Creamos una variable cadena.
STRING TipoFamilia(A15).

*Asignamos valores a la variable cadena en función de los valores de la variable hijos.
DO IF (hijos<3).
 COMPUTE TipoFamilia='No numerosa'.
ELSE.
 COMPUTE TipoFamilia='Numerosa'.
END IF.

EXECUTE.
```

Ejemplo 59:

```
DO IF (sexo=1).
COUNT
  NumProbSaludFem = salud1 salud2 salud3 salud4 salud5 salud6 salud7 salud8 salud9  (1)  .
VARIABLE LABELS   NumProbSaludFem 'Número de problemas de salud que tienen las muje-
res entrevistadas' .
COUNT
  NumProblFem = prob2 prob1 prob3 prob4  (1 thru 7)  .
VARIABLE LABELS NumProblFem 'Número de problemas que tienen las mujeres entrevista-
das' .
ELSE.
  COMPUTE  NumProbSaludFem = 9999.
  COMPUTE  NumProblFem = 9999.
END IF.
EXECUTE.
```

- Loop:

```
LOOP #NombreContador=n TO m.
Comando_1
…….
Comando_n
END LOOP.
```

La estructura básica del comando LOOP lo que hace es ejecutar los comandos incluidos dentro del mismo tantas veces como valores distintos toma el contador (#NombreContador).

El contador va incrementando su valor. En primer lugar vale n, luego se incrementa en una unidad, n+1, y así sucesivamente hasta alcanzar el valor final, es decir, m. Los comandos incluidos dentro del comando LOOP se ejecutan, además de la primera vez, cada vez que el contador cambia de valor, es decir, se ejecutan un total de m-n+1 veces.

Ejemplo 60:

```
*Creamos una variable x con valor 1 en todos los casos.
COMPUTE x=1.

*Ejecutamos el bucle.
LOOP #K=1 TO 5.
COMPUTE x=x+1.
END LOOP.
EXECUTE.
```

En este ejemplo la ejecución del bucle hace que el valor de la variable x se incremente en una unidad cada vez. Originalmente la variable x tiene valor 1 en todos los casos. Cuando comienza a ejecutarse el LOOP el contador #K vale 1 y se realiza el primer incremento de x, es decir, x pasa a tomar valor 2 para todos los individuos. De esta forma:

cuando #K=1 entonces x=2
cuando #K=2 entonces x=3
cuando #K=3 entonces x=4
cuando #K=4 entonces x=5
cuando #K=5 entonces x =6

4.- UNIDAD DE PRODUCCIÓN

La unidad de producción de SPSS nos permite agrupar varios archivos de sintaxis en un único trabajo de producción y ejecutarlos conjuntamente.

Para acceder a la unidad de producción debemos seleccionar *Inicio – Todos los programas – SPSS para Windows –SPSS 13.0 Utilidad de producción.*

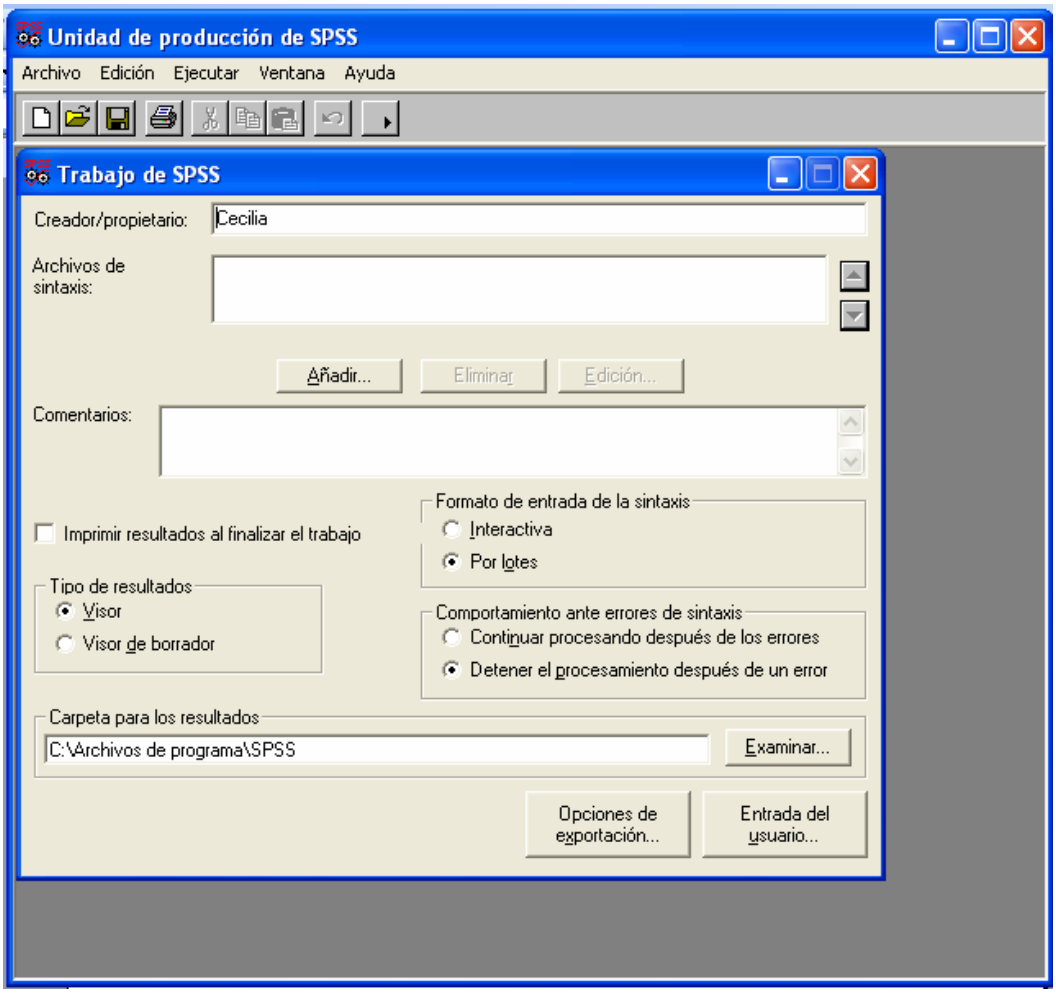

FIGURA 2: Pantalla que aparece al abrir la Unidad de Producción.

En la casilla archivos de sintaxis debemos incluir los ficheros de sintaxis que queremos que se ejecuten en el orden en que nos interesa hacerlo. Para ello basta con pinchar sobre el botón *Añadir…* y buscar el archivo que queremos incluir. Accionando los botones *Eliminar* y *Edición…* podemos, respectivamente, suprimir un archivo y editarlo (abrirlo y modificarlo).

Hay que tener en cuenta que la primera línea del primero de los ficheros de sintaxis que se va a ejecutar debe abrir un fichero de datos sobre el que co-

menzarán a ejecutarse los comandos, de otra manera obtendremos un error. No basta ejecutar el trabajo de producción con un fichero de datos activo como cuando ejecutamos un archivo de sintaxis. La unidad de producción es independiente, funciona sin tener abierto el editor de datos.

En la casilla comentarios podemos introducir explicaciones sobre el trabajo de producción.

Si activamos la casilla *Imprimir resultados al finalizar el trabajo*, una vez que se han ejecutado todos los archivos de sintaxis y se ha generado un fichero con todos los resultados, el programa lo imprime.

En *Tipo de resultados* podemos elegir el visor en el que queremos que se generen los resultados[3].

En *Comportamiento ante errores de sintaxis* podemos elegir la forma de controlar el tratamiento de los errores que se pueden producir en la ejecución:

- *Continuar procesando después de los errores*. Los errores no detienen el procesamiento de los comandos. Cuando se produce un error en un comando los siguientes se ejecutan normalmente.
- *Detener el procesamiento después de un error*. El procesamiento de comandos se detiene si se detecta un primer error en el archivo de trabajo de producción, de manera que el resto de los comandos no se ejecutan.

En *Carpeta para los resultados* debemos indicar la ruta de la carpeta en la que queremos que se grabe el fichero de resultados generado por la unidad de producción, sin más que pinchar sobre el botón *Examinar*. El programa denomina el archivo con el nombre del trabajo de producción, solo que en lugar de tener extensión .spp (la propia de los trabajos de producción) acabará en .spo (extensión correspondiente a ficheros de resultados SPSS).

En el botón *Opciones de exportación* se pueden exportar los resultados en distintos formatos. Es posible exportar el documento completo de resultados (con tablas, texto y gráficos), el documento sin gráficos o únicamente los gráficos.

[3] En el Visor es posible editar todos los resultados. El Visor de borrador es la ventana en la que se muestran los resultados como texto normal, de manera que las tablas y los gráficos no se pueden editar, únicamente es posible editar los resultados de texto.

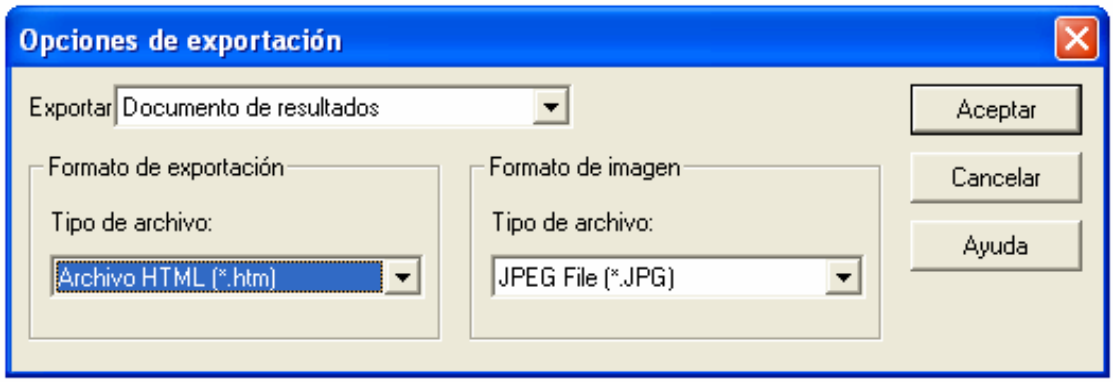

FIGURA 3: Cuadro de diálogo que aparece al pinchar sobre el botón *Opciones de exportación*.

Las tablas pivote y los resultados de texto se pueden guardar en formato HTML, de texto, Word/RTF y Excel, y los gráficos en una amplia variedad de formatos comunes utilizados por otras aplicaciones.

Una vez que determinamos en el cuadro de diálogo lo que queremos exportar, debemos seleccionar el formato de exportación y el formato de imagen (si procede) que nos interesa.

El o los ficheros exportados se grabarán en la misma carpeta en la que se guarda el archivo de resultados, con el mismo nombre y la extensión que corresponda. Cuando se exporta más de un gráfico, los nombres se basarán en el del archivo del trabajo de producción seguido de un número secuencial y la extensión del formato seleccionado.

La Unidad de Producción también nos permite sustituir nombres de archivo, variables o conjuntos de variables por símbolos de macro (parámetros), de manera que cada vez que ejecutamos el trabajo de producción el programa nos solicita que introduzcamos los valores de los mismos.

El nombre de un símbolo de macro debe comenzar por @, y en el botón *Entrada del usuario…* debemos indicar las características de la macro a la que invoca (el procedimiento al que llama):

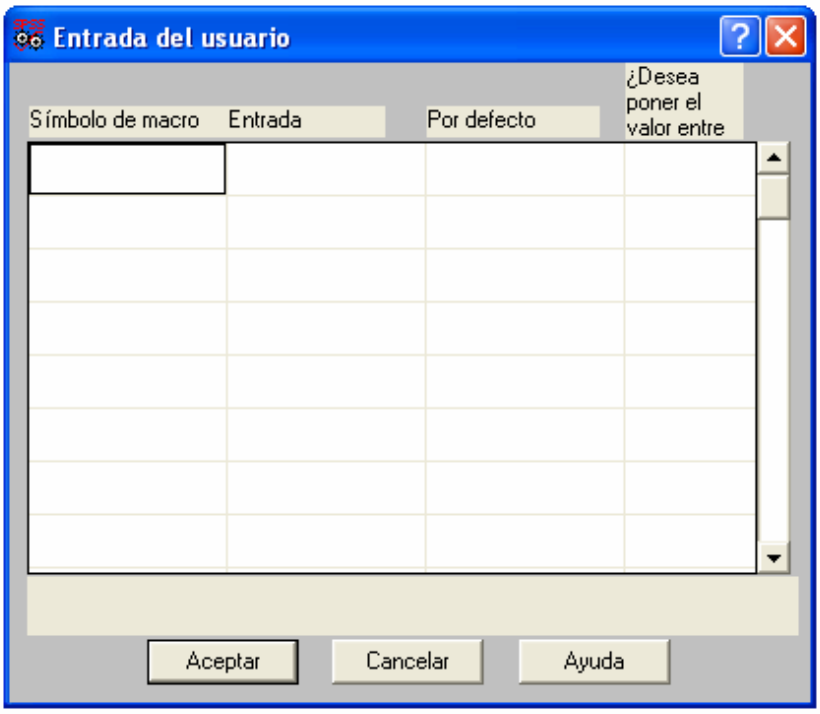

FIGURA 4: Cuadro de diálogo que aparece al pinchar sobre el botón *Entrada del usuario*.

- En *Símbolo de macro* incluimos el nombre utilizado en el archivo de sintaxis de comandos para invocar la macro que solicita información al usuario.

- En *Entrada* irá la etiqueta descriptiva que se muestra cuando el trabajo de producción solicita al usuario que introduzca información. Por ejemplo, se podría emplear la frase "¿Qué archivo de datos desea utilizar?" para identificar un campo que requiera el nombre de un archivo de datos.

-En la casilla *Por defecto* corresponde incluir el valor que utiliza por defecto el trabajo de producción si no se introduce uno nuevo. Este valor se muestra cuando el trabajo de producción solicita información al usuario, y se puede sustituir o modificar en el momento de la ejecución.

- En *¿Desea poner el valor entre comillas?* debemos introducir Y o Yes (Sí) si deseamos que el valor aparezca entre comillas. En caso contrario, tendremos que dejar el campo en blanco o introducir N o No. Para especificar un nombre de archivo se debe escribir Yes (Sí), ya que las especificaciones de nombres de archivos tienen que ir entre comillas.