# DEBUGGING AND DATA CLEANING TECHNIQUES WITH SPSS[1]

When working with large files, debugging can be time consuming. You can wait a long time for SPSS to tell you (in a secret code that you must learn to decipher) that you have misplaced a semicolon or misspelled a word. If you have an error in logic, SPSS probably will not be able to identify this. You must coax SPSS into helping you find these problems. In this document, I will describe a variety of ways that can help you debug a program, and you will begin to learn to interpret SPSS's secret code.

On most platforms, SPSS can be run in batch mode or interactive mode. (See the document "SPSS 10 Production Facility" for information on how to run in batch mode.) The rules for running SPSS in interactive and batch mode are somewhat different. It is a good practice to write SPSS syntax files so that they conform to both sets of rules.

**Batch Mode Rules and Features**

All commands in the command file must begin in column 1. You can use plus (+) or minus (-) signs in the first column if you want to indent the command specification to make the command file more readable.

If multiple lines are used for a command, column 1 of each continuation line must be blank.

Command terminators (periods) are optional. (Interactive mode requires terminators.)

An asterisk (*) in the first column indicates a comment line (see the COMMENT command).

It is possible to use the "Edit" command in batch mode. If the word EDIT is put at the top of a syntax file, SPSS will check the syntax for you, without getting the data to run the program. This is a helpful feature.

When you run a program in batch mode, it writes the log file to disk. Command line numbers are included. This makes debugging easier.

A "FINISH" line can be used. Lines of code past the finish line will not be processed.

## Configuring SPSS to Give More Information

By default, SPSS PC gives very little information about how things have gone in running a program. You can configure SPSS so that it will give more. Please see the following document for information about how to configure SPSS so that it will provide more complete information.

http://staff.washington.edu/glynn/config_spss.pdf

## Look at Your Program

Document your program, and make it neat. Have things line up so that visual checking will be easier.

[1]Prepared by Patty Glynn, University of Washington. April 12, 2001, updated 6/16/02 C:\all\help\helpnew\debugsp.wpd

## Debug with Fewer Cases

**EDIT.**

If the first line of your program in batch mode is edit the syntax will be checked without using data. With large data sets, this can save a lot of time.

**N OF CASES 100 .**

You can limit the number of cases with an N OF CASES command. If “100" is used, the first 100 cases will be used for your analysis.

Example from SPSS documentation.

```
TEMPORARY.
N 25.
FREQUENCIES VAR=SALARY.
N 50.
FREQUENCIES VAR=AGE.
```

The first N OF CASES command is temporary. Only 25 cases are used in the first FREQUENCIES procedure. The second N OF CASES command is permanent. The second frequency table and the report are based on 50 cases .... The working data file now contains 50 cases (assuming the original working file had at least that many).

CAUTION - if you save a file after using an N OF CASES command, you will lose cases!

**DRAWING A RANDOM SAMPLE**

Sometimes data sets are organized in such a way so that you do not get the variation you need if you select only the cases that are at the beginning of the file. Another option is to create a random sample. SPSS has a several random number generating functions, one of which is “uniform”. The following line would assign a random number that ranges in value from 0 to 1 (1 because that is the number inside the parentheses) to each case. The following lines would create a 10 percent random sample (approximately).

```
USE ALL.
COMPUTE filter_$=(uniform(1)<=.10).
VARIABLE LABEL filter_$ 'Approximately 10 % of cases (SAMPLE)'.
FORMAT filter_$ (f1.0).
FILTER BY filter_$.
EXECUTE .
```

Another option is to create a very small sample data that you can use to figure out what is going on. For example.

```
** Test.sps - sample program for testing problems.
DATA LIST FREE / ID * name (A9) AGE (f3) hinch (f3) weight (f3) gender (A) .
BEGIN DATA
1 George 22 72 203 m
2 Frank 67 65 180 m
3 Sally 27 62 120 f
4 Michelle 33 66 145 f
END DATA.
variable label
 name   'First name'
 age    'Age in years'
 hinch  'Height in inches'
```

```
 weight 'Weight in pounds' .

list var = all.
freq var = all .
```

## Identifying Some Common Errors

Look in the log for errors and warnings. Debug from top to bottom. Once there is an error, SPSS gets totally confused and is an unreliable witness regarding what might be right or wrong in the rest of the program. SPSS will let you know if it finds something wrong with an error message. The error message may not occur right below the problem, making it hard to figure out exactly where the error is. I changed the line `list var = all.` to `list var = names.` ran the program in interactive mode and got the following message:

```
>Error # 701 in column 12.  Text: NAMES
>An undefined variable name, or a scratch or system variable was specified
>in a variable list which accepts only standard variables.  Check spelling
>and verify the existence of this variable.
>This command not executed.
```

The problem is that the variable name is “NAME” not “NAMES”. But, the column number of the error is given, but not the line number. But, by running the program a different way, you coax SPSS into giving you more information. You can run the program in batch mode - or, you can use **Windows Explorer** to find the SPSS program. **Right-click on the program**, and then highlight **Run**. This will open SPSS and run the program - and line numbers will be included in your output, and the line number will be included with the error message - as seen below.

```
INCLUDE FILE='C:\all\spssclass\progs\test.sps'.
   6  ** Test.sps - sample program for testing problems.
   7  DATA LIST FREE / ID * name (A9) AGE (f3) hinch (f3) weight (f3) gender (A) .
   8  BEGIN DATA
  12  END DATA.
  13  variable label
  14   name   'First name'
  15   age    'Age in years'
  16   hinch  'Height in inches'
  17   weight 'Weight in pounds' .
  18
  19  list var = names .

>Error # 701 on line 19 in column 12.  Text: NAMES
>An undefined variable name, or a scratch or system variable was specified
>in a variable list which accepts only standard variables.  Check spelling
>and verify the existence of this variable.
>This command not executed.
```

At my last count, there were over a million kinds of mistakes that I have personally made in SPSS. I won’t be able to show all of them to you’re here, but will try to expose you to a few so that you will recognize them.

As I mentioned above, the rules for SPSS in batch mode and interactive mode are slightly different. It is possible to write syntax so that it conforms to both sets of rules, and I recommend doing so.

In batch mode, spacing matters. Each command line must begin in column one. This is what SPSS

uses as an indication of when one command line ends, and another begins. A continuation of a command line onto the next line must NOT begin on column one (or there may be a + or - for readability). Failure to follow this rule is a common error.

In interactive mode, periods (terminators) matter.

SPSS also uses is a slashes and parentheses as punctuation within commands. For examples:

```
aggregate outfile = *
   / break   = allv
   / meanage = mean( age )
   / nage    = n( age )
   / sdage   = sd( age ) .

MATCH FILES TABLE= ‘c:\full\path\tmpone.sav’
           / FILE= ‘c:\full\path\tmpmany.sav
           / BY state county tract
           / MAP .
```

Failure to use these properly will result in errors.

Failing to properly identify a directory or file name when reading or writing a file will also result in an error. In the following example, the file name should have been `'c:/progs/test1.sav'` For example :

```
save out = 'c:/prog/test1.sav' .

>Error # 61 in column 12.  Text: c:/prog/test1.sav
>The filename is not valid.
>This command not executed.
```

If you try to merge data sets that are not sorted in the right order, you will get a message like the following:

```
** Test.sps - sample program for testing problems.
DATA LIST FREE / ID * name (A9) AGE (f3) hinch (f3) weight (f3) gender (A) .
BEGIN DATA .
1 George 22 72 203 m
2 Frank 67 65 180 m
3 Sally 27 62 120 f
4 Michelle 33 66 145 f
END DATA .
save out = 'c:/all/spssclass/progs/test1.sav' .

** note that the following data are not in order of ID.
DATA LIST FREE / ID exam1 .
BEGIN DATA .
1  22
3  27
2  67
4  33
END DATA .
save out = 'c:/all/spssclass/progs/test2.sav' .

match files file = 'c:/all/spssclass/progs/test1.sav'
          / file = 'c:/all/spssclass/progs/test2.sav'
          / by id .
```

```
list var = all.

      ID NAME      AGE HINCH WEIGHT GENDER    EXAM1

    1.00 George     22   72    203  m         22.00
    2.00 Frank      67   65    180  m           .
File #2
     KEY:        2

>Error # 5130
>File out of order.  All the files in MATCH FILES must be in non-descending
>order on the BY variables.  Use SORT CASES to sort the file.
>This command not executed.
```

So, the message is telling us that the second file listed is not sorted in ascending order of the "by varible". Adding "sort cases by id." before `"save out = 'c:/all/spssclass/progs/test2.sav' .'` would solve this problem.

Another error message that you might find is "`Incorrect variable name: either the name is more than 8 characters`".

```
DATA LIST FREE / ID exam1 .
BEGIN DATA .
1  22
2  33
END DATA .

compute x1 = exam2 .

>Error # 4285 in column 14.  Text: EXAM2
>Incorrect variable name: either the name is more than 8 characters, or it
>is not defined by a previous command.
>This command not executed.
```

This may be because the variable was not included in a keep statement, or because it has been misspelled.

## SPSS Can't Find Some Problems – it Is up to You!

1. Check you N of cases. When you match files, rarely should you end up with more cases than you had in any of the files you merged. If you end up with more cases, probably something went wrong.

2. Be suspicious. Look for face validity. If things seem too odd to be true, it might be because of a programming error, or dirt in your data.

3. There are many problems that SPSS cannot find for you. If you make a mistake in logic when creating a variable, SPSS won't know it. A simple error in a keystroke can make your variable completely wrong. RUN TESTS, LOOK AT OUTPUT! For example, there is a mistake in the following code. The value of 9 is missing, and should not be included in the non-south. When you create new variables, examine the values closely. Descriptives can be used to make sure the minimum and maximums are within the expected range, and means are about what you would expect. Compare means for different groups. Do they conform to what you expect?

   Are there variables in the data set from which you can internal checks? Are there ways to check for

inconsistencies? With SPSS, you can use IF statements to create flags to alert you to problems. The following tests to see if anyone reports that they are currently married, and have never been married.

```
DATA LIST FREE / curmar nevmar .
begin data.
1 1
1 0
1 0
end data.
if curmar = 1 and nevmar = 1 flagmar = 1 .
freq var = flagmar.
```

While it is possible in SPSS for the PC to change values directly in the data, I DO NOT recommend it. If you change data in this way, you do not have a record of changes made. Instead, I recommend creating a program that gets the original data, makes corrections, document the changes, and saves a new data set. For example:

```
* MARNEW.SPS.
* GET MAROLD, MAKE CORRECTIONS, AND SAVE MARNEW.
* DOCUMENT ERROR WAS DETECTION AND CORRECTION.

GET FILE = 'C:\DL\MAROLD.SAV' .

* ERROR FOUND AND CORRECTED ON 4/1/2001.
* AGE OF RESP WITH ID 2001 SHOULD BE 21 .
* WAS CODED AS 211 .
IF ID = 2001 AGE = 21 .

DESC VAR = AGE .

SAVE OUTFILE = 'C:\DL\MARNEW.SAV' .
```

**FLAGMAR**

| | | Frequency | Percent | Valid Percent | Cumulative Percent |
|---|---|---|---|---|---|
| Valid | 1.00 | 1 | 33.3 | 100.0 | 100.0 |
| Missing | System | 2 | 66.7 | | |
| Total | | 3 | 100.0 | | |