

A Macro to Convert SAS® Data Sets into Lotus 123 Files

H. Ian Whitlock, Westat Inc.

Introduction

A common question is, "How do you make TAB or comma delimited files from SAS data for input into a spread sheet program?". The usual response is write a DATA step "PUT" program. The catch is that each new SAS data set requires a new "PUT" program. SAS/ACCESS® can be used to create DIF files, but this requires another SAS product.

In addition, one often wants a header record containing the names of the variables.

A one DATA step macro is presented to turn any SAS data set into a comma or tab delimited file with an optional header record naming the variables.

The Tools

The keywords `_NUMERIC_` and `_CHARACTER_` are used to put all variables of a SAS data set into one of two arrays without knowing any variable names.

`CALL VNAME`s are used to obtain the variable names.

In addition some `%IF - %THEN - %ELSE`, `%DO - %END` and `%LET` statements are used to control the process.

Parameters

The DATA step is divided into two sections - one is responsible for the header and the other for the data.

The header lists first the character variables (if any) followed by and then the numeric variables (if any). The form of the header is controlled by the parameter `HDR`.

When `HDR = _NONE_`, this line is omitted.

When `HDR = _NAMES_`, each variable name is enclosed in double quotes and separated by a character determined by the parameter `SEP`.

When `HDR = _LIST_` (the default), in addition to the above, the two blocks of character and numeric variables are enclosed in single quotes.

The presentation of data is controlled by the parameter `MISSING`. How should missing values be reported?

When `MISSING = _MISSING_`, missing values are printed as determined by the system option `MISSING`. (You must set this option before calling the macro or accept the SAS default values.) For example, when using comma as the delimiter:

```
"",ABC",,5,.
```

represents a five field record in which the first, third and fifth fields are missing. The first two fields are character and the remainder are numeric.

When `MISSING = _NULL_`, no value is written between the separators of a missing value. For the record above we have:

```
,"ABC",,5,
```

The parameter `SEP` defines the separating character. You may use the keywords `_TAB_` and `_COMMA_` or you may explicitly give a character in quotes such as `'#'` or `'05'X`. `SEP` is assigned `_TAB_` by default which is converted to the ASCII tab character `'09'X`, since the PC environment is assumed but not required.

The Macro

```
%macro sasto123(
  data= _last_ , /* sas data set for input */
  out= out , /* fileref for flat file output */
  hdr= _LIST_ , /* kywrd _NAMES_, _LIST_, _NONE_ */
  sep= _TAB_ , /* separator ',' '#' '09'x etc. */
               /* or keyword _TAB_ or _COMMA_ */
  missing= _NULL_ /* kywrds _NULL_, _MISSING_ */
);

  /* avoid data set variables beginning with double */
  /* underscore e.g. __nums __chars __name __i */

  %let missing = %upcase ( &missing ) ;
  %let hdr = %upcase ( &hdr ) ;
  %let sep = %upcase ( &sep ) ;
  %if &sep = _TAB_ %then %let mark = '09'x ;
  %else %if &sep = _COMMA_ %then %let mark = ',' ;
  %else %let mark = &sep ;

  data _null_ ;
    file &out notitles noprint mod ;
    set &data;

    /* arrays must follow set statement. */
    array __nums (*) _numeric_ ;
    array __chars (*) $ _character_ ;
```

```

/* length must follow array statements. */
%if &hdr ^= _NONE_ %then
  %str (length __name $ 8);

/* prepare to write line the variable names. */
%if &hdr ^= _NONE_ %then
%do ;
  if __n_ = 1 then
  do;
    %if &hdr = _LIST_ %then
    %do ;
      if dim (__chars) > 0 then put "" @;
    %end ;
    do __i = 1 to dim (__chars);
      call vname (__chars (__i), __name);
      put "" __name + (-1) "" @;
      if __i < dim (__chars) then
        put &mark @;
    end;
    %if &hdr = _LIST_ %then
    %do ;
      if dim (__chars) > 0 then put "" @;
    %end ;
    if dim (__nums) > 0 then
    do ;
      if dim (__chars) > 0 then put &mark @;
      %if &hdr = _LIST_ %then
        %str (put "" @);
      end;
      do __i = 1 to dim (__nums);
        call vname (__nums (__i), __name);
        put "" __name + (-1) "" @;
        if __i < dim (__nums) then
          put &mark @;
      end;
      %if &hdr = _LIST_ %then
      %do ;
        if dim (__nums) > 0 then put "" @;
      %end ;
      put;
    end;
  %end ;

/* write the variable values. */

do __i = 1 to dim (__chars);
  __chars (__i) =
    translate (__chars (__i), "", "");
end;

%if &missing = _NULL_ %then
%do ;
  do __i = 1 to dim (__chars);
    if __chars (__i) ^= '' then
      put "" __chars (__i) $ + (-1) "" @;

```

```

    if __i < dim (__chars) or dim (__nums) > 0 then
      put &mark @;
    end ;
    do __i = 1 to dim (__nums);
      if __nums (__i) ^= . then
        put __nums (__i) + (-1) @;
      if __i < dim (__nums) then
        put &mark @;
    end ;
    put ;
  %end ;
  %else
  %do ;
    put @1 "" (__chars *) ($ + (-1) ",") + (-1)
      "" (__nums *) ($ + (-1) ',');
  %end ;
run;
%mend sasto123;

```

Test Code and Usage

The code was tested under SAS for Windows version 6.08 TS407 using code such as:

```

filename out 'c:\sesug94\sasto123.dat';
%sasto123 (data = w)

```

All combinations of the parameter values were tested with W containing only character data, only numeric data, and both types of data. In each case the data was taken from:

a very long lineeeee	charchar	1	2
short	char	3	4
.	c	.	5
.	.	.	.

Conclusion

It is not hard to develop a useful general macro for converting SAS data sets into files accepted by the main spread sheet programs.

The author can be contacted by mail or e-mail

H. Ian Whitlock
Westat
1650 Research Boulevard
Rockville, MD 20850-3129

Whitloi1%westat@mcimail.com

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.