Creating Column Delimited Files with Little or No Effort

J. Charles Gober, Beekeeper Enterprises, Alexandria, Virginia

ABSTRACT

When working with large SAS® datasets containing hundreds of variables and millions of records clients will often ask you to create column delimited files to be used on other platforms. Usually to be processed by a third generation language such as Cobol®, PL/I®, FORTRAN®, etc., languages that have been around longer than most database technologies. In many cases the client will ask you to modify your files by including only certain variables, converting SAS dates to Gregorian or Julian formats, stripping off unwanted decimals. The client may also want you to eliminate white space to decrease file size. And there is always the inevitable change of mind after you are all done, variables that were originally dropped are now to be included, kept variables excluded. States are now to be spelled out instead of abbreviated, and certain variables are now to be placed at the beginning of the record.

INTRODUCTION

As a contractor you are constantly moving from client to client. Unless you are lucky enough to be assigned to a startup project, you will be working with data files someone else created. If these files were create by a less experienced programmer, certain SAS defaults were probably taken for. When variables were created they might have been allowed to default to the SAS default values of eight bytes for numerics and eight bytes for characters, or if the programmer was uncertain of the data, they could have been heavily padded to allow for entries of unordinary variable length. If you were to create a file delimited by a comma or some other character this would not be a problem for SAS will automatically trims all variables when using just the vanilla put statement. Example: **PUT A ',' B ',' C ',' ...** *etc.*;. This gives you a nice neat tight file with each variable (and delimiter) separated by only one blank. Excess white space has been removed. Not so easy if you want to have everything lined up in nice neat columns.

HISTORY OF THE PROCESS

Many of us, at one time or another, have thought about writing routines that would make our programming lives simpler. Few of us ever find the time and opportunity to do so. This process was created in such a circumstance. One week of spare time before a contract renewal. A contract to convert large SAS datasets to column delimited text files.

Our first attempt at automating this process was to execute a **PROC CONTENTS** to gather variable names and lengths, feed this information into a program and let the program create the **PUT** statements. This worked on '*cleaned*' datasets containing only character variables but left excessive white space when the variables were not properly assigned lengths. Variables that only had the values of '**Y**' and '**N**' were assigned eight bytes when they really only needed one. Numeric variables could not be done since the lengths reported by the CONTENTS routine represented bytes of storage. For example the number '**123456.78910**' will take up twelve positions on a flat file but might be reported as having a (byte) length of only eight.

Our second attempt was to create a routine to scan every variable of every record and then pass the maximum lengths into the macro routine. This process worked so well that we decide to wrap the process in a macro routine with parameters allowing us to feed in the name of the input file and the name of the output file. Unfortunately the down fall was that this took extensive CPU time to carry out the billions of operations necessary to complete all of the comparisons. definitely not something to use if your are being charged for CPU usage.

After a few trial runs and as with all seemingly good ideas, we were asked to improve the macro and add a parameter to either **DROP** or **KEEP** only certain variables. Then came a request to accommodate format changes for SAS date and time formats and any other user formats to be applied. Another user wanted the ability to have the **PUT** statements stored for latter execution. Someone needed to have certain variables at the beginning of the record layout to act as keys. Someone else wanted all character variables to be placed before the numerics. There also had to be assurances that the datasets internally used by the macro did not accidentally have names that were already in use. And finally someone just wanted the process converted to create comma delimited files.

TECHNIQUES

None of the techniques and ideas used to create this macro were hard to envision or create. It was just a matter of having the available time on your hands to apply them. In fact most of the routines were modifications of examples taken directly from SAS books. Since this routine was designed to be called from other programs a macro with name-style invocations[a] was the obvious choice for packaging. We also borrowed techniques to help our macro write a program to write another program[b] The creation of arrays[c] is heavily documented along with using length statements[d], counting the number of words in a string[e], etc. Because of the length of the code involved it is impossible to explain each section in the allowable six pages but hopefully by breaking down the macro into small sections We have made it is easier to comprehend.

CONCLUSION

The final macro has eleven parameters that should fit most everyone's needs. More could be added but it was already getting too customized. It is designed to eliminate programmer error and to free up time for more challenging tasks. It is capable of running on almost any platform than SAS is designed for. It works quite well when using the SAS **AUTOCALL00** option. It only takes five minutes to set up and can save hours of frustration. The following code is the latest of the ongoing versions and hopefully has enough documentation to allow anyone to try it or to customize it. If you wish a copy of the code emerald to you just send a request.

John Charles Gober
Beekeeper Enterprises
8105 Carlyle Place
Alexandria, Virginia 22308-1402
703-768-1319
j.gober@worldnet.att.net

[a] SAS Institute Inc. (1990), *SAS Guide to Macro processing, Version 6, Second Edition*, Cary, NC: SAS Institute Inc.
[b] SAS Institute Inc. (1987), *SAS Applications Guide, 1987 Edition*, Cary, NC: SAS Institute Inc.
[c] SAS Institute Inc. (1990), *SAS Language: Reference, Version 6*, First Edition, Cary, NC: SAS Institute Inc.
[d] SAS Institute Inc. (1989), *SAS Language and Procedures: Usage, Version 6, First Edition*, Cary, NC: SAS Institute Inc.
[e] Ibid a.

```
OPTIONS MPRINT;

 %MACRO SAS2TEXT
  (SASFILE=,        /* SAS DATAFILE                 */
   OUTPGM=,         /* FLATFILE PROGRAM             */
   OUTFILE=,        /* FLATFILE DATA                */
   FILETYPE=,       /* ASCII,LOTUS                  */
   POSITION=ALPHA,  /* 'ALPHA','TYPE','VARNUM'      */
   TEMPSET=TEMP,    /* PREFIX FOR TEMP DATASETS     */
   FRSTVARS=,       /* VARIABLE TO BE PLACE FIRST   */
   DROP=,           /* VARIABLES TO DROP            */
   KEEP=,           /* VARIABLES TO KEEP            */
   FORMATS=,        /* SPECIAL OUTPUT FORMATS       */
   DUMPDATA=NO);    /* 'YES', 'NO'                  */
 /***********************************************/
 /*                                             */
 /* MACRO NAME:  SAS2TEXT                       */
 /* PURPOSE:THIS MACRO CREATES A SAS PROGRAM    */
 /*         THAT CAN BE USED TO CONVERT A SAS   */
 /*         DATASET TO AN SAS2TEXT FILE.        */
 /*                                             */
 /*         THIS PROGRAM ANALYSES EVERY ELEMENT */
 /*         OF EACH NUMERIC VARIABLE.  THEREFORE,*/
 /*         IT MAY BE EXTREMELY TIME            */
 /*         CONSUMING.                          */
 /* PARAMETERS:                                 */
 /*    SASFILE  - COMPLETE NAME OF THE SAS      */
 /*               DATASET TO BE CONVERTED       */
 /*    OUTPGM   - THE FULLY QUALIFIED NAME      */
 /*               OF THE OUTPUT FILE WHERE      */
 /*               THE SAS PROGRAM WILL BE       */
 /*               LOCATED.                      */
 /*    OUTFILE  - THE FULLY QUALIFIED NAME      */
 /*               OF THE FLATFILE CONTAINING    */
 /*               THE CONVERTED DATA            */
 /*    POSITION - EITHER 'ALPHA'                */
 /*                      'TYPE'                 */
 /*                      'VARNUM'               */
 /*               THIS ALLOWS THE FLAT FILE     */
 /*               TO HAVE THE VARIABLE NAMES    */
 /*               PLACED IN ALPHABETICAL        */
 /*               ORDER, GROUP BY VARIABLE      */
 /*               TYPE (ALPHA OR NUMERIC),      */
 /*               OR BY THE SAS VARIABLE        */
 /*               NUMBER BY THE SAS VARIABLE    */
 /*    TEMPSET  - ALLOWS THE USER TO CHOOSE     */
 /*               THE NAME OF THE TEMPORARY     */
 /*               DATASETS USED BY THIS         */
 /*               MACRO. NAME OF TEMPSET IS     */
 /*               RESTRICTED TO SEVEN           */
 /*               CHARACTERS.                   */
 /*    FRSTVARS - THE VARIABLES TO APPEAR       */
 /*               FIRST IN THE TEXT FILE.       */
 /*    DROP/KEEP- NAMES OF THE SAS VARIABLES TO */
 /*                         TO DROP/KEEP ON THE */
 /*                         TEXT FILE           */
 /*                         PARAMETER).         */
 /*    FORMATS  - OVERRIDES INFORMATS           */
 /*               NORMALLY ASSIGNED TO          */
 /*               VARIABLES.                    */
 /*    DUMPDATA - 'YES' CREATE PROGRAM AND      */
 /*               EXECUTE 'NO'CREATE PROGRAM ONL */
 /* PLATFORM:    DOS,VMS,OS/2,MAC,MVS           */
 /* WRITTEN BY:  JOHN GOBER                     */
 /*              BEEKEEPER ENTERPRISES          */
 /*              8105 CARLYLE PLACE             */
 /*              ALEXANDRIA, VA  22308-1402     */
 /* WRITTEN:     JULY 1992                      */
 /***********************************************/

 %LOCAL __ROOT __STRING __COUNT __WORD __I __LRECL;
 %LOCAL __VAR DKOPT DIMABC DIMDEF LENDEF LENABC;

/****************************************************/
/* CHECK FOR DATASET WITH 0 OBS         SECTION ONE */
/****************************************************/

DATA _NULL_;
    IF numobs = 0 then
    DO;
       PUT "ERROR:   EMPTY DATASET...NO OBSERVATIONS";
       PUT "ERROR:   EMPTY DATASET...NO OBSERVATIONS";
       PUT "ERROR:   EMPTY DATASET...NO OBSERVATIONS";
       abort 9999;
    END;
    if 0 then SET &SASFILE nobs=numobs;
    STOP;
 RUN;

/****************************************************/
```

```
/* CONVERT PARAMETERS TO UPPERCASE      SECTION TWO */
/****************************************************/

 %LET SASFILE=%UPCASE(&SASFILE);
 %LET OUTPGM=%UPCASE(&OUTPGM);
 %LET OUTFILE=%UPCASE(&OUTFILE);
 %LET POSITION=%UPCASE(&POSITION);
 %LET TEMPSET=%UPCASE(&TEMPSET);
 %LET FRSTVARS=%UPCASE(&FRSTVARS);
 %LET DROP=%UPCASE(&DROP);
 %LET KEEP=%UPCASE(&KEEP);
 %LET FORMATS=%UPCASE(&FORMATS);
 %LET DUMPDATA=%UPCASE(&DUMPDATA);
 %LET FILETYPE=%UPCASE(&FILETYPE);

/****************************************************/
/* CREATE MACRO VARIABLES FOR THE     SECTION THREE */
/* FIRSTVARS                                        */
/****************************************************/

 %IF %STR(&FRSTVARS) ^= %STR( ) %THEN
 %DO;
    %LET __COUNT = 1;
    %LET __ROOT=W;
    %LET __STRING=&FRSTVARS;
    %LET __WORD=%NRBQUOTE
               (%SCAN(&__STRING,&__COUNT,%STR( )));
    %DO %WHILE(&__WORD^=);
       %LOCAL &__ROOT&__COUNT;
       %LET &__ROOT&__COUNT=&__WORD;
       %LET __COUNT=%EVAL(&__COUNT+1);
       %LET __WORD=%NRBQUOTE
                  (%SCAN(&__STRING,&__COUNT,%STR( )));
    %END;
    %LET __COUNT=%EVAL(&__COUNT-1);
 %END;

/****************************************************/
/* CREATE THE DROP AND KEEP STATEMENTS  SECTION FOUR */
/****************************************************/

 %IF %STR(&DROP) ^= %STR( ) AND
     %STR(&KEEP) ^= %STR( )
 %THEN %DO;
    DATA _NULL_;
      PUT "ERROR:   CANNOT USE DROP AND KEEP TOGETHER";
      PUT "ERROR:   CANNOT USE DROP AND KEEP TOGETHER";
      PUT "ERROR:   CANNOT USE DROP AND KEEP TOGETHER";
      ABORT 9999;
      STOP;
    RUN;
 %END;
 %ELSE
 %IF %STR(&KEEP) ^= %STR( )
    %THEN %LET DKOPT=(KEEP=&KEEP);
 %ELSE
 %IF %STR(&DROP) ^= %STR( )
    %THEN %LET DKOPT=(DROP=&DROP);

/****************************************************/
/* CREATE MACRO VARIABLE FOR THE       SECTION FIVE */
/* ORDER OF VARIABLES                               */
/****************************************************/

 %IF &POSITION=ALPHA %THEN %LET __VAR=NAME;
 %ELSE
 %IF &POSITION=TYPE %THEN %LET __VAR=TYPE NAME;
 %ELSE
 %IF &POSITION=VARNUM %THEN %LET __VAR=VARNUM;
 %ELSE
 %DO;
  %PUT NOTE:  NO ACCEPTABLE SORT ORDER DETECTED;
  %PUT NOTE:  NO ACCEPTABLE SORT ORDER DETECTED;
  %PUT NOTE:  NO ACCEPTABLE SORT ORDER DETECTED;
  %PUT NOTE:  VARIABLES WILL BE SORTED ALPHABETICALLY;
  %LET __VAR=NAME;
 %END;

/****************************************************/
/* CHANGE FORMATS OF VARIABLES          SECTION SIX */
/****************************************************/

    DATA &TEMPSET.0;
       SET &SASFILE(OBS=0);
       %IF %STR(&FORMATS) ^= %STR( ) %THEN
       %DO;
           FORMAT &FORMATS;
       %END;
    RUN;
```

```
/****************************************************/
/* CREATE DATASET CONTAINING          SECTION SEVEN */
/* VARIABLE ATTRIBUTES                              */
/****************************************************/

 PROC CONTENTS DATA=&TEMPSET.0&DKOPT
 OUT=&TEMPSET.0(KEEP=NAME TYPE LABEL LENGTH
                    VARNUM FORMAT FORMATD FORMATL)
              NOPRINT;
 RUN;

/****************************************************/
/* DETERMINE IF ONLY CHARACTER        SECTION EIGHT */
/* OR ONLY NUMERIC VARIABLES EXIST                  */
/****************************************************/

 PROC FREQ DATA=&TEMPSET.0;
    TABLE TYPE / OUT=&TEMPSET.5 NOPRINT;
 RUN;

 %LET DIMDEF=0;
 %LET DIMABC=0;
 DATA _NULL_;
    SET &TEMPSET.5;
    IF TYPE = 1 THEN
       CALL SYMPUT('DIMDEF',PUT(COUNT,5.));
    IF TYPE = 2 THEN
       CALL SYMPUT('DIMABC',PUT(COUNT,5.));
 RUN;

/****************************************************/
/* DETERMINE MAXIMUM LENGTHS FOR       SECTION NINE */
/* VARIABLES                                        */
/****************************************************/

 DATA &TEMPSET.1(DROP=__I);
    SET &SASFILE&DKOPT END=LAST;
    %IF &DIMDEF > 0 %THEN
    %DO;
       ARRAY  DEF{&DIMDEF} _NUMERIC_;
       ARRAY  LENDEF{&DIMDEF} _TEMPORARY_;
       DO __I = 1 TO DIM(DEF);
          LENDEF{__I} =
          MAX
           (LENGTH
            (LEFT
             (PUT
              (DEF{__I},BEST30.))),LENDEF{__I});
       END;
       IF LAST THEN
       DO;
          DO __I = 1 TO DIM(DEF);
             DEF{__I} = LENDEF{__I};
          END;
       END;
    %END;
    %IF &DIMABC > 0 %THEN
    %DO;
       ARRAY  ABC{&DIMABC} _CHARACTER_;
       ARRAY  LENABC{&DIMABC} _TEMPORARY_;
       DO __I = 1 TO DIM(ABC);
          LENABC{__I} =
           MAX(LENGTH(ABC{__I}),LENABC{__I});
       END;
       IF LAST THEN
       DO;
          DO __I = 1 TO DIM(ABC);
             ABC{__I} = LEFT(PUT(LENABC{__I},3.));
          END;
       END;
    %END;
    IF LAST THEN OUTPUT;
 RUN;

 PROC TRANSPOSE DATA=&TEMPSET.1
                OUT=&TEMPSET.2 NAME=NAME;
    VAR
    %IF &DIMABC > 0 %THEN _CHARACTER_;
    %IF &DIMDEF > 0 %THEN _NUMERIC_;
    %STR(;);
 RUN;

%IF %upcase(&FILETYPE) = ASCII %THEN
%DO;

/****************************************************/
/* DETERMINE LRECL OF OUTPUT RECORD     SECTION TEN */
/* DETERMINE VARIABLE TYPES AND VARIABLES LENGTHS   */
/****************************************************/
```

```
 PROC SORT DATA=&TEMPSET.0;
    BY NAME;
 RUN;
 PROC SORT DATA=&TEMPSET.2;
    BY NAME;
 RUN;

 DATA &TEMPSET.3(DROP=POINTER);
    LENGTH FORMAT $12;
    MERGE &TEMPSET.2(IN=IN1)
          &TEMPSET.0(IN=IN2) END=LAST;
    BY NAME;
    RETAIN POINTER 1;
    IF TYPE = 1 THEN
       DO;
          IF FORMATL = 0 THEN
          DO;
             LEN = INPUT(COMPRESS(COL1),3.);
             FORMAT=COMPRESS('BEST'||PUT(LEN,3.)||'.');
             IF FORMAT = "BEST1." THEN FORMAT = "1.";
          END;
          ELSE
          DO;
             LEN=FORMATL;
             IF FORMATD^=0 THEN
             FORMAT=COMPRESS(FORMAT||
                     PUT(FORMATL,2.)||
                     '.'||PUT(FORMATD,2.));
             ELSE
             FORMAT=COMPRESS
                     (FORMAT||PUT(FORMATL,2.)||'.');
          END;
       END;
    IF TYPE = 2 THEN
    DO;
       IF FORMATL = 0 THEN
       DO;
          LEN = INPUT(COMPRESS(COL1),3.);
          FORMAT=COMPRESS('$CHAR'||PUT(LEN,3.)||'.');
       END;
       ELSE
       DO;
          LEN=FORMATL;
        FORMAT=COMPRESS(FORMAT||PUT(FORMATL,3.)||'.');
       END;
    END;
    POINTER = POINTER + LEN;
    IF LAST THEN
    DO;
       POINTER = POINTER - 1;
       CALL SYMPUT('__LRECL',PUT(POINTER,5.));
    END;
 RUN;

/****************************************************/
/* DETERMINE POSITIONS OF            SECTION ELEVEN */
/* VARIABLES IN OUTPUT FILE                         */
/****************************************************/

%IF %STR(&FRSTVARS)^= %STR( ) %THEN
 %DO;
    %IF %SUBSTR(&FRSTVARS,1,1) ^= %STR( ) %THEN
    %DO;
       DATA &TEMPSET.3;
          SET &TEMPSET.3;
          %IF &__VAR = NAME %THEN
          %DO;
             TYPE = 1000;
          %END;
          %ELSE
          %DO;
             __ORDER = TYPE*1000;
          %END;
          %DO __I = 1 %TO &__COUNT;
             IF NAME="&&&__ROOT&__I" THEN
               __ORDER = &__I;
          %END;
       RUN;
       PROC SORT DATA=&TEMPSET.3;
          BY __ORDER &__VAR;
       RUN;
    %END;
 %END;
 %ELSE %DO;
    PROC SORT DATA=&TEMPSET.3;
       BY &__VAR;
    RUN;
 %END;
```

```
/*******************************************/
/* CREATE SAS TO SAS2TEXT  SECTION TWELVE */
/*PROGRAM                                 */
/*******************************************/

 DATA &TEMPSET.4;
    RETAIN POINTER 1;
    FILE &OUTPGM NOPRINT;
    SET &TEMPSET.3 END=LAST;
    IF _N_ = 1 THEN
    DO;
       PUT @001 "DATA _NULL_;";
       PUT @004 "FILE &OUTFILE NOPRINT
                 LRECL=&__LRECL;";
       PUT @004 "SET &SASFILE;";
       PUT @008 "PUT";
    END;
    PUT @008 "@"  POINTER
        @015 NAME
        @025 FORMAT
        @040 "/*" LABEL "*/";
    IF LAST THEN
    DO;
       PUT @008 ";";
      PUT @001 "RUN;";
    END;
    POINTER = POINTER + LEN;
 RUN;

%END;
%ELSE
%IF &FILETYPE = LOTUS %THEN
%DO;
****************THIS IS THE DELIMITED VERSION*********;

/****************************************************/
/* DETERMINE LOGICAL RECORD LENGTH OF    SECTION TEN */
/* OUTPUT FILE   SECTION TEN                         */
/* DETERMINE VARIABLE TYPES AND VARIABLES LENGTHS    */
/****************************************************/

 PROC SORT DATA=&TEMPSET.0;
    BY NAME;
 RUN;
 PROC SORT DATA=&TEMPSET.2;
    BY NAME;
 RUN;
 DATA &TEMPSET.3(DROP=POINTER);
    LENGTH FORMAT $12;
    MERGE &TEMPSET.2(IN=IN1)
          &TEMPSET.0(IN=IN2) END=LAST;
    BY NAME;
    RETAIN POINTER 1;
    IF TYPE = 1 THEN
       DO;
          IF FORMATL = 0 THEN
          DO;
             LEN = INPUT(COMPRESS(COL1),3.) + 1;
             FORMAT=COMPRESS('BEST'||PUT(LEN,3.)||'.');
             IF FORMAT = "BEST1." THEN FORMAT = "1.";
          END;
          ELSE
          DO;
             LEN=FORMATL + 1;
             IF FORMATD^=0 THEN
             FORMAT=COMPRESS(FORMAT||
                     PUT(FORMATL,2.)||
                     '.'||PUT(FORMATD,2.));
             ELSE
             FORMAT=COMPRESS
                     (FORMAT||PUT(FORMATL,2.)||'.');
          END;
       END;
    IF TYPE = 2 THEN
    DO;
       IF FORMATL = 0 THEN
       DO;
          LEN = INPUT(COMPRESS(COL1),3.) +3;
          FORMAT=COMPRESS('$CHAR'||PUT(LEN,3.)||'.');
       END;
       ELSE
       DO;
          LEN=FORMATL + 3;

FORMAT=COMPRESS(FORMAT||PUT(FORMATL,3.)||'.');
       END;
    END;
    POINTER = POINTER + LEN;
    IF LAST THEN
    DO;
       POINTER = POINTER - 1;
       CALL SYMPUT('__LRECL',PUT(POINTER,5.));
    END;
 RUN;

/*******************************************/
/* CREATE SAS TO SAS2TEXT  SECTION TWELVE */
/*PROGRAM                                 */
/*******************************************/

 DATA &TEMPSET.4;
    COMMA  = ',';
    SQUOTE = "'";
    DQUOTE = '"';
    RETAIN POINTER 1;
    FILE &OUTPGM NOPRINT;
    SET &TEMPSET.3 END=LAST;
    IF _N_ = 1 THEN
    DO;
       PUT @001 "DATA _NULL_;";
       PUT @004 "FILE &OUTFILE NOPRINT
LRECL=&__LRECL;";
       PUT @004 "SET &SASFILE;";
       PUT @008 "PUT";
    END;

    IF TYPE = 1 THEN
    DO;
       PUT @008 "@"  POINTER
           @015 NAME
           @030 FORMAT
           @045 "/*" LABEL "*/";
    END;
    ELSE IF TYPE = 2 THEN
    DO;
       PUT @008 "@"  POINTER
           @015 SQUOTE  NAME SQUOTE COMMA
           @030 FORMAT
           @045 "/*" LABEL "*/";
    END;

    IF LAST THEN
    DO;
       PUT @008 ";";
      PUT @001 "RUN;";
    END;
    POINTER = POINTER + LEN;
 RUN;

%END;

/****************************************************/
/* DELETE DATASETS                 SECTION THIRTEEN */
/****************************************************/

 PROC DATASETS LIB=WORK;
     DELETE &TEMPSET.0 &TEMPSET.1 &TEMPSET.2
            &TEMPSET.3 &TEMPSET.4 &TEMPSET.5;
 RUN;

/****************************************************/
/* EXECUTE CREATED PROGRAM    SECTION FOURTEEN      */
/****************************************************/

 %IF &DUMPDATA=YES %THEN
    %INCLUDE &OUTPGM;

 %MEND SAS2TEXT;

/****************************************************/
/*  END END END END END END END END END END END END  */
/****************************************************/

 /* SAMPLE INVOCATION */

 LIBNAME TEMP 'D:\SAS\SIGI\';
  %SAS2TEXT(SASFILE=TEMP.TESTFILE,
            OUTPGM='d:\sas\sugi\outpgm.sas',
            OUTFILE='d:\sas\sugi\flatfile.dat',
            FILETYPE=ASCII,
            POSITION=ALPHA,
            TEMPSET=TEMP,
            FRSTVARS=IDNO LNAME FNAME,
            DROP=,
            KEEP=,
            FORMATS=,
            DUMPDATA=NO);
```