Paper 108-25

Make Room for Me

Gregg Taylor, Navistar International Transportation Corporation, Westchester, IL

Abstract

This paper is most beneficial to mainframe users who frequently use the FSVIEW and FSEDIT procedures to visualize data under a MVS/TSO environment. Working with tables in FSVIEW in a TSO mainframe environment can at times be tedious because of the limited screen width. Formatting columns to smaller widths can increase the number of columns that may be viewed at one time.

Sometimes, just a brief glance at the table will allow an idea of what might be an acceptable format for better visualization. But using this approach allows for a great possibility of truncating the view of the data. A better approach is to find the length of the longest value in a column, then reset the length and formatted length to the longest trimmed value for each a column. By using the SQL procedure and the SAS® System Macro Language, you can automate this process.

Introduction

This paper presents 3 macros that may be used to shrink the lengths of character value columns in a table. This saves storage space, decreases CPU processing time and possibly increases the number of columns that can be displayed with FSVIEW or FSEDIT. This macro resets the length and formatted length of each character type column to the maximum trimmed length of the longest value in the column. Three variations of the macro are included. %CHARVAR1 will operate on the most-recently-created data set. %CHARVAR2 has call parameters for a libref and a table name. %CHARLIB1 has a call parameter for a libref only and will operate on each table in the library. Only %CHARLIB1 is explained, but the other two macros are included in this paper.

If the table(s) being “shrunk” are used in merge processing steps, then it is important to remember to reset the column lengths in each table to the same length. Otherwise, merging on these character columns may not produce the desired results. If trimming functions and SQL are used to match character values, then having columns of the same lengths is not quite so critical. As always, know your data and how it will be used.

The Code

The first step would be to assign &LIBREF the libname you intend to process, which is accomplished by the %LET statement.

```
%LET LIBREF= libname;
%MACRO CHARLIB1;
```

/* &FILEOBS is declared global so that it may be utilized throughout all of the macro. */

```
%GLOBAL FILEOBS;
```

/* Next, the table MEMBERS is created. This table is used to select members one at a time for processing. It is created by selecting distinct members from DICTIONARY.COLUMNS having TYPE=’CHAR’ and from DICTIONARY.TABLES having rows or observations greater than 0. */

```
PROC SQL NOPRINT;
   CREATE TABLE MEMBERS AS
   SELECT DISTINCT
      C.MEMNAME
   FROM
      DICTIONARY.COLUMNS C,
      DICTIONARY.TABLES T
   WHERE
      (T.NOBS-T.DELOBS) > 0
      AND
      C.MEMNAME=T.MEMNAME
      AND
      UPCASE(C.TYPE)='CHAR'
      AND
      LIBNAME=T.LIBNAME="&LIBREF";
```

/* &FILEOBS is assigned the value of %SQLOBS, which is the numbers of rows that exist in TABLE MEMBERS. FILEOBS is used as the maximum boundary in the next do loop processing. */

```
%LET FILEOBS=&SQLOBS;

  %DO I=1 %TO &FILEOBS;
```

/* &DSNAME is assigned the first member name from the Table MEMBERS */

```
    SELECT
      MEMNAME INTO :DSNAME
    FROM
      MEMBERS(OBS=1);
```

/*Table CVARLIST is created, consisting of all column names with TYPE=’CHAR’ */

```
   CREATE
     TABLE CVARLIST AS
   SELECT
     NAME
   FROM
     DICTIONARY.COLUMNS
   WHERE
     LIBNAME="&LIBREF"
     AND
```

```
        MEMNAME="&DSNAME"
        AND
        UPCASE(TYPE)='CHAR';
```

/* &CHAROBS is the number of rows that exist in the Table CVARLIST and is used as the maximum boundary in subsequent do loop processing */

```
    %LET CHAROBS=&SQLOBS;
```

/* &NAME1- &NAME&CHAROBS are created from the list of names in CVARLIST */

```
      SELECT
        NAME INTO :NAME1-:NAME&CHAROBS
      FROM
        CVARLIST;
```

/* &LENGTH1- &LENGTH&CHAROBS are created by selecting the maximum trimmed length of each &NAME1- &NAME&CHAROBS.
&FORM1- &FORM&CHARBOS are created by concatenating '$' with the maximum trimmed length and then with '.', creating a macro variable similar to '$8.'. *?

```
      %DO J=1 %TO &CHAROBS;
        SELECT
          MAX(LENGTH(TRIM(&&NAME&J))),
          COMPRESS('$' !!
PUT(MAX(LENGTH(TRIM(&&NAME&J))),3.) !! '.')
            INTO :LENG&J,:FORM&J
        FROM
          &LIBREF..&DSNAME;
      %END;
```

/* The next do loop uses the macro variables &name*n*,&leng*n* and &form*n* and alters the columns for the table declared by &DSNAME, resetting both the length and format for each character column. */

```
      %DO K=1 %TO &CHAROBS;
        ALTER TABLE &LIBREF..&DSNAME
          MODIFY &&NAME&K CHAR(&&LENG&K)
FORMAT= &&FORM&K;
      %END;
```

/* After the table has been altered, the table name is deleted from the MEMBERS table so that the next table name become the first table name. */

```
        DELETE
        FROM
          MEMBERS
        WHERE
          MEMNAME="&DSNAME";
```

/* Processing then returns to the portion of the do loop below until the table MEMBERS is empty.
SELECT
MEMNAME INTO :DSNAME
FROM
MEMBERS(OBS=1); */

/*When the table Members is empty the macro will end with the following code */

```
%END;
  QUIT;
%MEND CHARLIB1;
```

/* To invoke macro, supply name of fileref to process in the preceding let statement before compiling the macro */

```
%CHARLIB1
```

Code for Macro CHARVAR1

/* This macro will change format length for character vars for last data set created */

```
%MACRO CHARVAR1;
%LET LIBREF=%SCAN(&SYSLAST,1,'.');
%LET DSNAME=%SCAN(&SYSLAST,2,'.');
PROC SQL NOPRINT;
CREATE
    TABLE CVARLIST AS
  SELECT
    NAME
  FROM
    DICTIONARY.COLUMNS
  WHERE
    LIBNAME="&LIBREF"
    AND
    MEMNAME="&DSNAME"
    AND
    UPCASE(TYPE)='CHAR';
%LET CHAROBS=&SQLOBS;
SELECT
    NAME INTO :NAME1-:NAME&CHAROBS
  FROM
    CVARLIST;
%DO I=1 %TO &CHAROBS;
    SELECT
      MAX(LENGTH(TRIM(&&NAME&I))),
      COMPRESS('$' !!
PUT(MAX(LENGTH(TRIM(&&NAME&I))),3.) !! '.')
        INTO :LENG&I,:FORM&I
    FROM
      &LIBREF..&DSNAME;
  %END;
%DO J=1 %TO &CHAROBS;
```

```
    ALTER TABLE &LIBREF..&DSNAME
      MODIFY &&NAME&J CHAR(&&LENG&J)
FORMAT= &&FORM&J;
%END;
QUIT;
%MEND CHARVAR1;

%CHARVAR1
```

Code for Macro CHARVAR2

/* This macro will change format length for character vars when you supply the parameter values for LIBREF and DSNAME */

```
%MACRO CHARVAR2(LIBREF,DSNAME);
PROC SQL NOPRINT;
   CREATE
     TABLE CVARLIST AS
   SELECT
     NAME
   FROM
     DICTIONARY.COLUMNS
   WHERE
     LIBNAME="&LIBREF"
     AND
     MEMNAME="&DSNAME"
     AND
     UPCASE(TYPE)='CHAR';
%LET CHAROBS=&SQLOBS;
SELECT
     NAME INTO :NAME1-:NAME&CHAROBS
   FROM
     CVARLIST;
%DO I=1 %TO &CHAROBS;
     SELECT
       MAX(LENGTH(TRIM(&&NAME&I))),
       COMPRESS('$' !!
PUT(MAX(LENGTH(TRIM(&&NAME&I))),3.) !! '.')
         INTO :LENG&I,:FORM&I
     FROM
       &LIBREF..&DSNAME;
   %END;
%DO J=1 %TO &CHAROBS;
     ALTER TABLE &LIBREF..&DSNAME
       MODIFY &&NAME&J CHAR(&&LENG&J)
FORMAT= &&FORM&J;
   %END;
QUIT;
%MEND CHARVAR2;
```

ACKNOWLEDGEMENTS

Thanks to Derek Morgan for editorial assistance.

CONCLUSIONS

The macros included here may impact three areas of data processing: decreasing storage requirements, decreasing CPU processing time, and improving data visualization through FSVIEW or FSEDIT. An important point to remember is to set corresponding columns in multiple tables to the same length before processing when using SET or MERGE statements.

Gregg Taylor
Fleet Support Manager
Navistar International Transportation Corporation
5 Westbrook Corp Center
Westchester IL 60154
PH 708-947-6737
FX 708-947-6374
gregg.taylor@navistar.com