Building a Simple SAS Macro to Generate SQL Instructions in Frequently Used DB2 Tables

Monique Bryher / MRI Consulting, Inc., Los Angeles, CA

ABSTRACT

Writing SQL instructions to extract data stored in DB2 can be made easier through the use of simple coding techniques that utilize SAS macros and the SAS Autocall Facility. Symbolic variables are replaced with values for common SQL expressions in frequently-accessed tables; this has the advantage of minimizing coding time, reducing syntactical errors, affording multiple individuals access to code stored in a “tool kit” of macros that have been tested, debugged, and which can be documented. As will be shown in this paper, these techniques can also be employed to generate large lists in a WHERE predicate that would otherwise be impractical to enter manually.

INTRODUCTION

The topics of interest can best be discussed by referring to a common situation. Figure 1 shows code, developed in an MVS operating environment, which will extract selected rows (observations) and columns (variables) from a DB2 table called MEMBERS:

```
PROC SQL;
  CONNECT TO DB2(SSID=DSNB);
  CREATE TABLE MBRLIST AS
  SELECT
    MBR, SEX, DOB, SUB_ACCT, GROUP_NO
    FROM CONNECTION TO DB2(
      SELECT DISTINCT
        MBR, SEX, DOB, SUB_ACCT, GROUP_NO
        FROM APPLICN.MEMBERS
        WHERE MBR IN
          (‘MBR1’, ‘MBR2’, . . . ‘MBRn’)
          AND
           MBR <> '00000000')
        ORDER BY MBR;
%PUT &SQLXRC;
%PUT &SQLXMSG;
```

Figure 1

The data extracted from the above table needs to be linked to a second commonly-used table, POLICY:

```
PROC SQL;
  CONNECT TO DB2(SSID=DSNB);
  CREATE TABLE POLICY AS
  SELECT
   *
  FROM CONNECTION TO DB2(
   SELECT
    P.MBR, P.SUB_ACCT, P.GROUP_NO,
    SEX, DOB, BEGIN_DT, END_DT
    FROM
      C000007.MBRLIST M,
      APPLICN.POLICY P
    WHERE M.MBR = P.MBR)
    ORDER BY MBR
    ;
```

Figure 2

What if the selection list in Figure 1 contains more than a few entries, say a thousand or more? There has to be a better way to enter a list of member numbers than by typing them in! Further, the number of columns that need to be extracted changes often; it would be convenient to be able to make those changes easily.

In this situation, writing a macro that will instruct SAS to build its own list of members to populate the WHERE clause and replacing key elements, such as file names, selection parameters, and other conditions with macro variables, will introduce flexibility to the code. Also, placing the macro in an Autocall library will make it accessible to other programmers, saving them effort and time.

STEP 1 - Define Macro Variables

Which variables in our example would be candidates for conversion to macro variables? The table name MBRLIST is one, and so is the ORDER BY variable. Also, we might want to select two columns from APPLICN.MEMBERS today, but tomorrow want four or five columns. Figures 3 and 4 highlight the variables we might wish to convert to macro variables.

```
PROC SQL;
    CONNECT TO DB2(SSID=DSNB);
    CREATE TABLE MBRLIST AS
    SELECT
      MBR, SEX, DOB, SUB_ACCT,
      GROUP_NO,
    FROM CONNECTION TO DB2(
      SELECT DISTINCT
        MBR, SEX, DOB, SUB_ACCT,
       GROUP_NO
      FROM APPLICN.MEMBERS
      WHERE MBR IN
        (‘MBR1’, ‘MBR2’, . . . ‘MBRn’)
        AND
        MBR <> '00000000')
      ORDER BY MBR;
```

Figure 3

```
PROC SQL;
  CONNECT TO DB2(SSID=DSNB);
  CREATE TABLE POLICY AS
  SELECT
   *
  FROM CONNECTION TO DB2(
   SELECT
    P.MBR, P.SUB_ACCT, P.GROUP_NO,
    SEX, DOB, BEGIN_DT, END_DT
    FROM
     C000007.MBRLIST M,
     APPLICN.POLICY  P
    WHERE M.MBR = P.MBR)
    ORDER BY MBR ;
```

Figure 4

STEP 2: Automating the WHERE Clause Selection

Getting SAS to construct a list of members to extract from a DB2 table that can change from run to run is easy:

❶ Create an external file that consists of the code from Figure 3 arranged in a series of PUT statements. When _N_ = 1, the main SQL code is written; on successive reads of the incoming data, each member number is written to the file. When end-of-file is reached, the SQL instructions are closed.

❷ Immediately following that code, insert a line to %INCLUDE the external file.

❸ After the DB2 column names are replaced with macro variables, wrap the code in a macro, and you're ready to go.

```
*********************************************;
* macro to build and extract member selection list  ;
*********************************************;
%macro sqllist(dataset,sortordr,filename,listname,
     ❸              finalist,cond,varlist1,varlist2);
 * ;
 DATA _NULL_;
  SET &dataset END=EOF;
  BY &sortordr;
  FILE &filename;
  IF _N_ = 1 THEN DO;
     PUT @004 "PROC SQL; CONNECT TO
                   DB2(SSID=DSNB);";
 ❶    PUT @006 "CREATE TABLE &listname
                   AS";
     PUT @006 "SELECT MBR, &varlist1, "
     PUT @006 "FROM CONNECTION TO DB2(";
     PUT @008 "SELECT DISTINCT MBR,
                    &varlist2, "
     PUT @008 "FROM APPLICN.MEMBERS";
     PUT @008 "WHERE &sortordr IN (" @;
     END;
  PUT @022 "'" MBR $8. "'" @;   🡰  Writes the
  IF EOF THEN DO;                list of members
     PUT +000 ") AND";
     PUT +000 "MBR <> '00000000')";
     PUT @008 "ORDER BY &sortordr;";
     END;
  ELSE PUT +000 ", ";
* ;
 %INCLUDE &filename;    ❷
 %PUT &sqlxrc;
 %PUT &sqlxmsg;
 OPTIONS OBS=MAX FIRSTOBS=1;
 * ;
 PROC DBLOAD DBMS=DB2
   DATA=&listname;
   IN XXXX01DB.XXXX01TS;
   LIMIT=0;
   SSID=DSNB;
   TABLE=C000007.&listname;
   LOAD;
 %PUT &sqlxrc;
```

```
%PUT &sqlxmsg;
* ;
PROC SQL;
 CONNECT TO DB2(SSID=DSNB);
 CREATE TABLE &finalist AS
 SELECT
  *
 FROM CONNECTION TO DB2(
  SELECT
   P.MBR, &varlist2,
   BEGIN_DT,
   END_DT,
  FROM
     C000007.&listname M,
     APPLICN.POLICY P
  WHERE &COND)
   ORDER BY &sortordr;
%put &sqlxrc;
%put &sqlxmsg;

execute (drop table c000007.&listname) by db2;
%mend sqllist;  ❸
```

STEP 3: Using the Autocall Library

After debugging the macro, use the following simple steps to access it using the Autocall Facility:

1) Copy the macro to a PDS that contains only macros. An example of a valid name in TSO would be "USER.SASAUTOS.LIB".

2) The name of the member you have just created in the macro library must be the same as that of the macro, e.g., "sqllist".

3) In programs calling the macro, add a DD statement in your JCL pointing to the PDS named in 1) above, e.g.,

```
//MACLIB DD DSN=USER.SASAUTOS.LIB,
//                    DISP=OLD
```

4) Add the following OPTIONS statement at the beginning of your program:

```
        MAUTOSOURCE and
        SASAUTOS=MACLIB.
```

STEP 4: Running the Job

The macro, when submitted, will create an external file called MBRFIND, which is then included and executed. Portions of the SAS log and the statements produced by the macro are re-printed at the end of this paper.

CONCLUSION

The techniques described in this brief discussion are relatively simple to implement and carry a number of advantages worth considering when accessing large DB2 tables:

Development and debugging time, while initially requiring a longer effort, will, over the long run, save significant time if the macro is stored in an Autocall library because other programmers will have access to the code.

Macros stored in an Autocall library will have an additional benefit of encouraging standardization in accessing the tables, as opposed to programmers writing their own unique (and sometimes imperfect) code.

Once the selection criteria, access method and linkage (if more than one table) are established, programmers are free to work on any unique requirements their task might have.

Last, using the flexibility of the SAS System to generate the selection list in the WHERE clause is efficient in that only the observation (rows) desired are extracted; obviously, run time can be reduced signficantly as well.

References

SAS® Guide to Macro Processing,
 Version 6, 2nd Edition

SAS®Guide to the SQL Procedure: Usage and
 Reference, Version 6, 1st Edition

"Developing a SAS System Autocall Macro Library
 As an Effective Toolkit", by Steven A. Wilson,
 WUSS Proceedings, 1994

Acknowledgements

Thanks to Steve Wilson for his own illuminating discussion of the Autocall Facility (see References).

Biographical Notes

Monique Bryher is president of MRI Consulting, Inc., which has specialized in developing applications and database management using SAS for over 6 years. She has an undergraduate degree in economics and a master’s degree in public health, both from UCLA. Her areas of product knowledge include hospital and health care information systems, materials management, finance and banking. She has been a Quality Partner of SAS Institute, Inc. since August 1995.

Author’s Address
Monique Bryher, MSPH
MRI Consulting, Inc.
6043 Shirley Avenue
Tarzana, CA 91356
(818) 774-0043
INTERNET: rebwest@aol.com

Please feel free to send your comments, suggestions, and experiences.

```
%sqllist(dataset=sellist,sortordr=mbr,filename=mbrfind,listname=mbrlist,finalist=newlist,
          cond=%str(m.mbr=p.mbr and m.sub_acct=p.sub_acct and m.group_no=p.group_no),
          varlist1=%str(sub_acct, group_no),
          varlist2=%str(p.sub_acct, p.group_no,sex, dob));

NOTE: The file MBRFIND is:
Dsname=C000007.MBRFIND.DATA,
     Unit=3390,Volume=SMSL10,Disp=SHR,Blksize=1600,
     Lrecl=80,Recfm=FB

NOTE: 2000 records were written to the file MBRFIND.
NOTE: The DATA statement used 0.08 CPU seconds and 3835K.

PROC SQL;
CONNECT TO DB2(SSID=DSNB);
CREATE TABLE MBRLIST AS
SELECT MBR, SEX, DOB, SUB_ACCT, GROUP_NO FROM CONNECTION TO DB2( SELECT
DISTINCT MBR, SEX, DOB FROM APPLICN.MEMBERS WHERE MBR IN ('29430953', '29556961',
'29827921', '29866793', '27040774', '27074612', '27417935', '27854491', '02667348', '33360922', '33360931',
'33360938', '33361494', '33361497', '33361510', '33361512', '33361574', '33361564', '33361623', '33361640',
'33361651', '33361671', '33361675', '33361690', '33361751', '33361753', '33361760', '33361770', '33361771',
'33361804', '33361972', '33362199', '33362359', '33598722', '33959165', '33959234', '33959444', '33959599',
'33959656', '33959660', '33959677', '33959733', '33959892', '33959925', '33962909', '33962922', '33962925',
'33962928', '33960230', '33960262', '33960412', '33960415', '33960419', '33960446', '33960548', '33960617',
'33960648', '33960660', '33960682', '33960688', '33960894', '33960989', '33961024', '33961077', '33961082',
'33962299', '33961519', '33961522', '33961542', '33961545', '33961558', '33961586', '33961845', '33961858',
'33961862', '33961874', '33961906', '33961936','33962093', '33962238', '33962267', '33962275', '33962307',
'33962317', '22051991', '22051992', '22051993', '22051994', '22051995', '22051996', '22057453', '22055920',
'22074273', '22074330', '22074331', '22074332', '22074041', '22074042', '22074044', '22074046', '22074047',
'22074049', '22074058', '22074059', '22074060', '22074061', '22074065', '22074066', '22074075', '22074076',
'22074077', '22074078', '22074079', '22074080', '22074097', '22074098', '22074102', '22074105', '22074106',
'22074107', '22074227', '22074228', '22074120', '22074121', '22074123', '22074125', '22074135', '22074136',
'22074137', '22074139', '22074142', '22074146') AND MBR <> '00000000' ) ORDER BY MBR;
SELECT DISTINCT MBR, SEX, DOB, SUB_ACCT, GROUP_NO FROM APPLICN.MEMBERS WHERE
MBR IN ('29430953', '29556961', '29827921', '29866793', '27040774', '27074612', '27417935', '27854491',
'02667348', '33360922', '33360931', '33360938', '33361494', '33361497', '33361510', '33361512', '33361574',
'33361564', '33361623', '33361640', '33361651', '33361671', '33361675', '33361690', '33361751', '33361753',
'33361760', '33361770', '33361771', '33361804', '33361972', '33362199', '33362359', '33598722', '33959165',
'33959234', '33959444', '33959599', '33959656', '33959660', '33959677', '33959733', '33959892', '33959925',
'33962909', '33962922', '33962925', '33962928', '33960230', '33960262', '33960412', '33960415', '33960419',
'33960446', '33960548', '33960617', '33960648', '33960660', '33960682', '33960688', '33960894', '33960989',
'33961024', '33961077', '33961082', '33962299', '33961519', '33961522', '33961542', '33961545', '33961558',
'33961586', '33961845', '33961858', '33961862', '33961874', '33961906', '33961936','33962093', '33962238',
'33962267', '33962275', '33962307', '33962317', '22051991', '22051992', '22051993', '22051994', '22051995',
'22051996', '22057453', '22055920', '22074273', '22074330', '22074331', '22074332', '22074041', '22074042',
'22074044', '22074046', '22074047', '22074049', '22074058', '22074059', '22074060', '22074061', '22074065',
'22074066', '22074075', '22074076', '22074077', '22074078', '22074079', '22074080', '22074097', '22074098',
'22074102', '22074105', '22074106', '22074107', '22074227', '22074228', '22074120', '22074121', '22074123',
'22074125', '22074135', '22074136', '22074137', '22074139', '22074142', '22074146') AND MBR <>
'00000000' ) ORDER BY MBR;

NOTE: Table WORK.MBRLIST created, with 2000 rows and 8 columns.
SYMBOLGEN:  Macro variable SQLXRC resolves to 0
SYMBOLGEN:  Macro variable SQLXMSG resolves to 0
NOTE: The PROCEDURE SQL used 2.34 CPU seconds and 5404K.

SYMBOLGEN:  Macro variable LISTNAME resolves to mbrlist
SELECT * FROM "C000007"."MBRLIST"
SYMBOLGEN:  Macro variable SQLXRC resolves to 0
SYMBOLGEN:  Macro variable SQLXMSG resolves to  0

CREATE TABLE "C000007"."MBRLIST" ( "MBR" CHAR(8), "SEX" CHAR(1), "DOB" DATE,
"SUB_ACCT"CHAR(8),"GROUP_NO"CHAR(7)) IN XXXX01DB.XXXX01TS
COMMIT WORK
SELECT * FROM "C000007"."MBRLIST"
INSERT INTO "C000007"."MBRLIST" VALUES (?,?,?,?,?,?,?,?)
COMMIT WORK
INSERT INTO "C000007"."MBRLIST" VALUES (?,?,?,?,?,?,?,?)
COMMIT WORK
NOTE: Load completed. Examine statistics below.
NOTE:  Inserted (2000) rows into table (C000007.MBRLIST)
NOTE:  Rejected (0) insert attempts see the log for details.

NOTE: The PROCEDURE DBLOAD used 0.65 CPU seconds and 5716K.
```