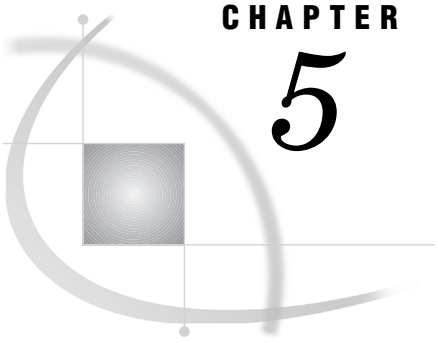

CHAPTER 5

Macro Variables and System Options

Introduction **59**
About the Macro Facility **59**
Automatic Macro Variables **59**
System Options **60**

Introduction

This topic describes the macro variables and system options that are available to use with SAS/ACCESS software.

About the Macro Facility

Most features of the SAS macro facility are portable. This section describes only those components of the macro facility that depend on the SAS/ACCESS engine. For more information, refer to the *SAS Macro Language: Reference* and the online help for the macro facility.

Automatic Macro Variables

The following automatic macro variables are portable, but their values are determined by the SAS/ACCESS engine and your DBMS. Initially, the macro variables SYSDBMSG and SQLXMSG are blank, whereas SYSDBRC and SQLXRC are set to **0**. Two of the macro variables that can be used anywhere while accessing DBMS data are SYSDBMSG and SYSDBRC.

SYSDBMSG
contains DBMS-specific error messages that are generated when you use SAS/ACCESS to access your DBMS data.

SYSDBRC
contains DBMS-specific error codes that are generated when you use SAS/ACCESS to access your DBMS data. Error codes that are returned are text, not numbers.

Because only one set of macro variables is provided, it is possible that, if tables from two different DBMSs are accessed, it might not be clear from which DBMS the error message originated. To address this problem, the name of the DBMS is inserted into the value of the SYSDBMSG macro variable.

For example, if you try to connect to ORACLE and use the incorrect password, you would receive the messages shown in Output 5.1 on page 60.

Output 5.1 SAS Log for an ORACLE Error

```
2? libname mydblib oracle user=pierre pass=paris path="orav7";

ERROR: ORACLE error trying to establish connection.  ORACLE error is
       ORA-01017: invalid username/password; logon denied
ERROR: Error in the LIBNAME or FILENAME statement.
 3? %put &sysdbmsg;

ORACLE: ORA-01017: invalid username/passsword; logon denied
 4? %put &sysdbrc;

-1017
 5?
```

The contents of the SYSDBMSG and SYSDBRC macro variables can be printed in the SAS log by using the %PUT macro. The automatic macro variables SYSDBMSG and SYSDBRC are reset after each SAS/ACCESS LIBNAME statement, DATA step, or procedure has been executed.

The SQL Procedure Pass-Through Facility generates return codes and messages that are available to you through the following two SAS macro variables:

SQLXMSG
: contains DBMS specific error messages. See “SQL Procedure Pass-Through Facility Return Codes” on page 83.

SQLXRC
: contains DBMS specific error codes. See “SQL Procedure Pass-Through Facility Return Codes” on page 83.

SQLXMSG and SQLXRC can be used only with the SQL Procedure Pass-Through Facility.

The contents of the SQLXMSG and SQLXRC macro variables can be printed in the SAS log by using the %PUT macro. SQLXMSG is reset to a blank string and SQLXRC is reset to a “0” when any SQL Procedure Pass-Through statement is executed.

System Options

SASTRACE and VALIDVARNAME are SAS system options that have SAS/ACCESS-specific applications.

Note: The SAS system option, REPLACE, which is not described here, is not supported and will be ignored by the SAS/ACCESS engines. △

SASTRACE

Generates trace information from a DBMS engine

Default: NONE

Valid in: Wherever SAS sytem options are valid: OPTIONS statement, configuration file, SAS invocation.

Syntax

SASTRACE= ',,,d'

',,,d'
gives information about SAS/ACCESS engine calls to a relational DBMS.

SAS/ACCESS Specific Details

SASTRACE is a SAS system option that has SAS/ACCESS-specific behavior. SASTRACE is a very powerful tool to use when you want to see the commands sent to your DBMS by the SAS/ACCESS engine. SASTRACE output is DBMS-specific; however, most SAS/ACCESS engines will show you statements like SELECT or COMMIT as the DBMS processes them for the SAS application. It replaces the DBDEBUG option from Version 6; this option name might be different for your DBMS or operating environment.

See Chapter 7, "Advanced Topics in SAS/ACCESS," on page 85 for more information.

Note: Output from SASTRACE differ depending on your DBMS. △

Example

The following example generates several messages from the SASTRACE= system option. Output 5.2 on page 61 was written to the SAS log, as specified by the SASTRACELOC=SASLOG option.

```
data work.winter_birthdays;
   input empid birthdat date9. lastname $18.;
   format birthdat date9.;
datalines;
678999 28DEC1966 PAVEO             JULIANA        3451
456788 12JAN1977 SHIPTON           TIFFANY        3468
890123 20FEB1973 THORSTAD          EDVARD         3329
;
run;

libname mydblib oracle user=dmitry password=elvis schema=bday_data;

options sastrace=',,,d' sastraceloc=saslog;
data mydblib.snow_birthdays;
   set work.winter_birthdays;
run;

libname mydblib clear;
```

Output 5.2 SAS Log Output from the SASTRACE= and SASTRACELOC= System Options

```
50   data work.winter_birthdays;
51      input empid birthdat date9. lastname $18.;
52      format birthdat date9.;
53   datalines;

NOTE: The data set WORK.WINTER_BIRTHDAYS has 3 observations and 3 variables.
NOTE: DATA statement used:
      real time           0.08 seconds
      cpu time            0.03 seconds

...
57   ;
58   run;
59
60   libname mydblib oracle user=dmitry password=XXXXX schema=bday_dat;
NOTE: Libref MYDBLIB was successfully assigned as follows:
      Engine:        ORACLE
      Physical Name:
61
62   options sastrace=',,,d' sastraceloc=saslog;
63   data mydblib.snow_birthdays;
64      set work.winter_birthdays;
65   run;

DEBUG: Open Cursor - CDA=2060054152 49 923702189 orusti 296 DATASTEP
DEBUG: PREPARE SQL statement:  50 923702189 orprep 63 DATASTEP
 SELECT * FROM SNOW_BIRTHDAYS 51 923702189 orprep 64 DATASTEP
DEBUG: Close Cursor - CDA=2060054152 52 923702189 orustt 365 DATASTEP
DEBUG: PHYSICAL connect. 53 923702190 orcon 372 DATASTEP
DEBUG: USER=scott 54 923702190 orcon 373 DATASTEP
DEBUG: Open Cursor - CDA=2057325192 55 923702190 orusti 296 DATASTEP
DEBUG: Open Cursor - CDA=2057332360 56 923702190 orusti 296 DATASTEP
NOTE: SAS variable labels, formats, and lengths are not written to DBMS
      tables.
DEBUG: EXECUTE SQL statement:  57 923702190 orexec 75 DATASTEP
CREATE TABLE SNOW_BIRTHDAYS(empid NUMBER ,birthdat DATE,lastname VARCHAR2
(18)) 58 923702190 orexec 76 DATASTEP
DEBUG: PREPARE SQL statement:  59 923702190 orins 330 DATASTEP
INSERT INTO SNOW_BIRTHDAYS (empid,birthdat,lastname) VALUES
(:empid,TO_DATE(:birthdat,'DDMONYYYY','NLS_DATE_LANGUAGE=American'),:lastnam
e) 60 923702190 orins 331 DATASTEP
NOTE: There were 3 observations read from the dataset WORK.WINTER_BIRTHDAYS.
DEBUG: *-*-*-*-*-*-* COMMIT *-*-*-*-*-*-* 61 923702190 orforc 102 DATASTEP
NOTE: The data set MYDBLIB.SNOW_BIRTHDAYS has 3 observations and 3
      variables.
DEBUG: *-*-*-*-*-*-* COMMIT *-*-*-*-*-*-* 62 923702190 orforc 102 DATASTEP
DEBUG: Close Cursor - CDA=2057325192 63 923702190 orustt 365 DATASTEP
DEBUG: Close Cursor - CDA=2057332360 64 923702190 orustt 365 DATASTEP
DEBUG: PHYSICAL disconnect. 65 923702190 ordcon 445 DATASTEP
DEBUG: USER=scott 66 923702191 ordcon 446 DATASTEP
NOTE: DATA statement used:
      real time           1.30 seconds
      cpu time            0.18 seconds
```

SASTRACELOC

Prints SASTRACE information to a specified location

Default: stdout

Valid in: OPTIONS statement, configuration file, SAS invocation.

Syntax

SASTRACELOC=stdout | SASLOG

Details

SASTRACELOC is a SAS system option that enables you to specify where to put the trace messages that are generated by SASTRACE. By default, the output goes to stdout or the default for your operating environment. You can send the output to a SASLOG by specifying SASTRACELOC=SASLOG. See "SASTRACE" on page 60 for an example of SASTRACELOC=.

Note: This option and its values may differ for each host. △

VALIDVARNAME

Controls the type of SAS variable names that can be used and/or created during a SAS session

Default: V7 (for Version 7 and later)

Valid in: configuration file, SAS invocation, OPTIONS statement, OPTIONS window.

Syntax

VALIDVARNAME=V7 | V6 | UPCASE | ANY

SAS/ACCESS Specific Details

VALIDVARNAME is a SAS system option that interacts with SAS/ACCESS applications. It enables you to control which rules apply for SAS variable names. For more information about the VALIDVARNAME= system option, see the *SAS Language Reference: Dictionary*. The settings are as follows:

VALIDVARNAME=V7
indicates that a DBMS column name will be mapped to a valid SAS name by using the following rules:

- Up to 32 mixed case alphanumeric characters are allowed.
- Names must begin with alphabetic characters or an underscore.
- Non SAS characters are mapped to underscores.
- Any column name that is not unique when normalized is made unique by appending a counter (0,1,2,...) to the name.

V7 is the default value for Version 7 and later of SAS software.

VALIDVARNAME=V6
indicates that only those variable names considered valid SAS variable names in Version 6 are considered valid. When V6 is specified in SQL Pass-Through code,

the DBMS engine truncates column names to 8 characters as it did in Version 6. If required, numbers are appended to the end of the truncated name to make it unique.

VALIDVARNAME=UPCASE
indicates that a DBMS column name will be mapped to a valid SAS name as described in VALIDVARNAME=V7 except that variable names are in uppercase.

VALIDVARNAME=ANY
allows any characters in DBMS column names to appear as valid characters in SAS variable names. Symbols, such as "=" and "*", must be contained in a *'varname'*n construct. ANY is required whenever you want to read DBMS column names that don't follow the SAS naming conventions.

For more information on SAS naming conventions, see Chapter 2, "SAS Names and Support for DBMS Names," on page 9 and system options in the *SAS Language Reference: Dictionary*.

Example

The following example shows how the PROC SQL Pass-Through Facility works with VALIDVARNAME=V6.

```
options validvarname=v6;
proc sql;
   connect to oracle (user=testuser pass=testpass);
   create view myview as
     select amount_b, amount_s
     from connection to oracle
       (select "Amount Budgeted$", "Amount Spent$"
         from mytable);
quit;

proc contents data=myview;
run;
```

The output from this example would show that "Amount Budgeted$" becomes AMOUNT_B and "Amount Spent$" becomes AMOUNT_S.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/ACCESS® Software for Relational Databases: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/ACCESS® Software for Relational Databases: Reference, Version 8

ISBN 1–58025–558–2

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999