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 while SYSDBRC and SQLXRC are "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 56.

Output 5.1 SAS Log for an ORACLE Error

```
2? libname mydblib oracle user=pierre pass=paris path='t:blunzer: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/password; logon denied
4? %put &sysdbrc;
-1017
```

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 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 84.

**SQLXRC**

contains DBMS specific error codes. See “SQL Procedure Pass-Through Facility Return Codes” on page 84.

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 REPLACE SAS system option, which is not described here, is not supported and will be ignored by the DBMS engines.
Examples of Dictionary

**SASTRACE**

Generates trace information from a DBMS engine

Default: NONE

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

**Syntax**

SASTRACE=’,,,d’

**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 DBMS engine. SASTRACE output is DBMS specific; however, most 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.

SASTRACE generates this message:

SASTRACE=’,,,d’

- gives information about SAS/ACCESS engine calls to the DBMS.


*Note:* Output from SASTRACE will differ depending on your DBMS.

**Example**

The following example generates several messages from the SASTRACE= system option.

```sas
data mydata.week;
do i=1 to 7;
output;
run;
options sastrace=’,,,d’ sastraceloc=saslog;
libname mydblib oracle user=testuser
   pass=testpass;
data mydblib.newweek;
set mydata.week;
run;
libname mydblib;
```
SASTRACELOC

Prints SASTRACE information to a specified location

Default: stdout

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

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. You can send the output to a SASLOG by specifying SASTRACELOC=SASLOG. See “SASTRACE” on page 57 for an example that specifies SASTRACELOC=.

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

VALIDVARNANE

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

Default: V7

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

**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 uppercased.

**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 11 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.

```sas
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;
options validvarname=v6;
proc contents data=myview;
run;
```

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