*Kirk has been working with the SAS System since 1979 and is a SAS Certified Professional®. His company provides custom SAS programming, application design and development, consulting services, and instructor-led and hands-on SAS training to clients around the world. Kirk is the author of four books including PROC SQL: Beyond the Basics Using SAS by SAS Institute, and more than three hundred peer-reviewed articles and papers that have appeared in professional journals and SAS User Group proceedings. He has also been an Invited speaker and instructor at more than three hundred SAS International, regional, local, and special-interest user group conferences and meetings throughout North America.*

Kirk's Korner:

# Simple and Effective Macro Coding Techniques for SAS® Users

***~ Part 2 ~***

The SAS® Macro Language is a powerful tool for extending the capabilities of the SAS System. In this issue of the SESUG Informat, I've identified two more macro-related tips that I use frequently in my own code. This technique illustrates an approach for defining the parameters in a user-defined macro.

## *Defining Positional Parameters*

Macros are frequently designed to allow the passing of one or more parameters. This allows the creation of macro variables so text strings can be passed into the macro. The order of macro variables as positional parameters is specified when the macro is coded. The assignment of values for each positional parameter is supplied at the time the macro is called.

To illustrate the definition of a two positional parameter macro, the following macro was created to display all table names (data sets) that contain the variable TITLE in the user-assigned MYDATA libref as a cross-reference listing. To retrieve the needed type of information, you could execute one or more PROC CONTENTS against selected tables. Or in a more efficient approach, you could retrieve the information directly from the read-only Dictionary table COLUMNS with the selected columns LIBNAME, MEMNAME, NAME, TYPE and LENGTH, as shown. For more information about Dictionary tables, readers may want to view the "free" SAS Press Webinar by Kirk Paul Lafler at http://support.sas.com/publishing/bbu/webinar.html#lafler2 or the published paper by Kirk Paul Lafler, Exploring Dictionary Tables and SASHELP Views.

*Macro Code*

```
%MACRO COLUMNS(LIB, COLNAME);
  PROC SQL;
    SELECT LIBNAME, MEMNAME, NAME, TYPE, LENGTH
      FROM DICTIONARY.COLUMNS
        WHERE UPCASE(LIBNAME)="&LIB" AND
              UPCASE(NAME)="&COLNAME";
  QUIT;
%MEND COLUMNS;

%COLUMNS(MYDATA,TITLE);
```

*(Continued on page 2)*

*(Continued from page 1)*

*After Macro Resolution*

```
PROC SQL;
  SELECT LIBNAME, MEMNAME, NAME, TYPE, LENGTH
    FROM DICTIONARY.COLUMNS
      WHERE UPCASE(LIBNAME)="MYDATA"  AND
            UPCASE(NAME)="TITLE";
QUIT;
```

*Output*

| Library Name | Member Name | Column Name | Column Type | Column Length |
|---|---|---|---|---|
| MYDATA | ACTORS | Title | char | 30 |
| MYDATA | MOVIES | Title | char | 30 |
| MYDATA | PG_MOVIES | Title | char | 30 |

## ~ Part 3 ~

This technique illustrates an approach for defining one or more keyword parameters in a user-defined macro.

## Defining Keyword Parameters

Macros can be designed to allow one or more parameters to be predefined or passed into a macro. As a design objective, this permits the creation of more flexible macros as well as the prospect of passing text string values directly into the macro's symbolic variables. Unlike positional parameter macros (discussed in the previous Newsletter issue), a keyword parameter macro enables the macro variables to be specified in any order when calling the macro. This inherent feature provides a significant advantage for macro developers and users alike since the assignment of values for each keyword parameter is supplied at the time the macro is called.

To illustrate the definition of a keyword parameter macro with two parameters, the following macro is defined to display all table names (data sets) that contain the variable TITLE in the user-assigned MYDATA libref generating a cross-reference listing. To retrieve the needed type of information, you could execute one or more PROC CONTENTS against selected tables. Or using a more efficient approach, you could retrieve the information directly from the read-only Dictionary table COLUMNS by specifying the columns LIBNAME, MEMNAME, NAME, TYPE and LENGTH, as shown. For more information about Dictionary tables, readers may want to view the "free" SAS Press Webinar by Kirk Paul Lafler at http://support.sas.com/publishing/bbu/webinar.html#lafler2 or the published paper by Kirk Paul Lafler, Exploring Dictionary Tables and SASHELP Views.

*(Continued on page 3)*

*(Continued from page 2)*

Macro Code

```
%MACRO COLUMNS(LIB=, COLNAME=);
  PROC SQL;
    SELECT LIBNAME, MEMNAME, NAME, TYPE, LENGTH
      FROM DICTIONARY.COLUMNS
        WHERE UPCASE(LIBNAME)="&LIB" AND
                UPCASE(NAME)="&COLNAME";
  QUIT;
%MEND COLUMNS;

%COLUMNS(LIB=MYDATA,COLNAME=TITLE);
```

After Macro Resolution

```
PROC SQL;
  SELECT LIBNAME, MEMNAME, NAME, TYPE, LENGTH
    FROM DICTIONARY.COLUMNS
      WHERE UPCASE(LIBNAME)="MYDATA"  AND
            UPCASE(NAME)="TITLE";
QUIT;
```

Output:

| Library Name | Member Name | Column Name | Column Type | Column Length |
|---|---|---|---|---|
| MYDATA | ACTORS | Title | char | 30 |
| MYDATA | MOVIES | Title | char | 30 |
| MYDATA | PG_MOVIES | Title | char | 30 |

## Contact Information

If you would like more information or have any questions about this tip, please contact Kirk Paul Lafler, Software Intelligence Corporation at KirkLafler@cs.com.

Kirk Paul Lafler
Software Intelligence Corporation
World Headquarters
P.O. Box 1390
Spring Valley, California 91979-1390