

Overview Of Common SAS Macro Functions

Table of Contents

Overview Of Common SAS Macro Functions1

Common Functions 3

Basics	
store values for macro variables	%LET <variable> = <value>;
print text to log	%PUT <text>
print macro variable	%PUT &<macro variable> ;
print all system macro variables	%PUT _AUTOMATIC_ ;
print all user macro variables	%PUT _USER_ ;
print all available macro variables	%PUT _ALL_ ;
Macro without parameters	%MACRO <name>; %MEND [<name>;
Macro with positional parameters	%MACRO <name> (<param1>; %MEND [<name>;
Macro with defaults (other than missing possible)	%MACRO <name> (<param1>=<value1>; %MEND [<name>;
Working with Macro Variables	
Loop through macro variable list	%LET list = green pink blue; %DO i = 1 %TO %SYSFUNC(COUNTW(&list)); %PUT %SCAN(list,&i); %END;
Access variables through variable contents	%LET var1 = green; %LET var2 = pink; %LET var3 = blue; %DO i=1 %TO 3; %PUT &&var&i; %END;
Work with Macro variables within data step, e.g. to avoid quoting issues	%LET var = value;

	DATA test; LENGTH var \$200.; Var = SYMGET(var); Var = TRIM(var) "2"; CALL SYMPUT('var',value2); RUN;
--	---

Common Functions

String Functions	
%INDEX(source, string)	Search source for string, returns position
%SCAN(source, position, delimiter)	Search for a word that is specified by its position in a string.
%PUT %SCAN(green pink blue, -1);	Example prints blue.
%LENGTH(<variable>)	Returns length of variable – also “0” for empty variables
%SUBSTR(source, position [, length])	Returns string parts.
%CMPRES(<text>)	Compress spaces to one space (COMPBL)
%UPCASE / %LOWCASE	Upper / lower case
Arithmetic Functions	
%EVAL(...)	Performs integer calculations
%SYSEVALF(...)	Performs floating-point arithmetic

Other Functions	
%DATATYP(...)	Returns “CHAR” or “NUMERIC”
%DATATYP(15.8) will return “NUMERIC”	
%VERIFY	Returns first char unique to an expression
%VERIFY(0&number,0123456789) will return > 0 if &number is not an integer	
Specific Functions	
BEEP tone , e.g. to recognize when a long run SAS program finishes	DATA _null_; CALL SOUND (100,200); CALL SOUND (200,200); RUN;

<p>Use of PARMBUFF, e.g. to avoid SAS errors when parameters has been mistyped or flexible parameter names should be used.</p> <p>Example for initializing unknown macro systems via parameters:</p> <p>CSS Paper 2009 – User friendly management of continuously improving standard macro systems</p>	<pre>%macro test /parmbuff; %PUT syspbuff=&syspbuff; %mend; %sales (parm1 = 1, parm2 = 0); Log output: syspbuff=(parm1 = 1, parm2 = 0)</pre>
<p>Sometimes FILES needs to be modified. Positioning according leading spaces can be used.</p>	<pre>DATA _NULL_; INFILE "c:\test1.txt" lrecl=100 END=eof TRUNCOVER; FILE "c:\test2.txt" lrecl=100; INPUT; text = _INFILE_; i=LENGTH(text)-LENGTH(LEFT(text)); PUT @i text; RUN;</pre>
<p>When the work area should be cleaned completely, then also global macro variables can be deleted with the SYMDEL function.</p>	<pre>Data _temp; Set sashelp.vmacro; Where scope='GLOBAL'; RUN; DATA _null_; Set _temp; Call symdel(name); Run;</pre>