%EVAL and %SYSEVALF Functions

%EVAL takes ‘numerical expression’ and returns ‘booleans’ (0 or 1) or ‘integers’, e.g.
  %LET i = %EVAL(3+2);
  %LET j = %EVAL(5/2)
  %LET a = 3;
  %LET b = %EVAL(a>0);
  %LET c = %EVAL(a=);

%SYSEVALF takes ‘numerical expression’ and returns ‘floating point numerical value’, e.g.
  %LET r = 2;
  %LET pi = 3.14159;
  %LET area = %SYSEVALF(&pi * &r**2);
Selected Character Functions

%func or %Qfunc

- **%LENGTH (character-string | text-expression):** string length. E.g., %LENGTH(&a); %LENGTH(&a and &b);
- **%INDEX (source, string):** searches source for the first occurrence of string and returns the position of its first character. A 0 is returned if string is not found. E.g., %INDEX(&a, jo);
- **%UPCASE and %QUPCASE:** convert values to uppercase
- **%SCAN and %QSCAN:** search for a word that is specified by its position in a string
- **%SUBSTR and %QSUBSTR:** produce a substring of a character string

Selected Quoting Functions

- **%STR and %NRSTR:** mask special characters and mnemonic operators in constant text at macro compilation. Unmatched quotation marks (" ") and parentheses ( () ) must be marked with a preceding %.
- **%BQUOTE and %NRBQUOTE:** mask special characters and mnemonic operators in a resolved value at macro execution. E.g., %IF %BQUOTE(&statename) NE %THEN ... They do not require that unmatched quotation marks and parentheses be marked.
- **%SUPERQ:** masks all special characters and mnemonic operators at macro execution but prevents resolution of the value.
- **%UNQUOTE:** unMASKs all special characters and mnemonic operators for a value.
%SYSFUNC and %QSYSFUNC Functions

Execute SAS functions or user-written functions (created with SAS/TOOLKIT)

%SYSFUNC(function(argument(s))<, format>)

where

- function:
  - cannot be a macro function
  - cannot be any of the following: INPUT/PUT, DIF/LAG, DIM/LBOUND/HBOUND, IORCMMSG, MISSING, RESOLVE, SYMGET, and all variable information functions.
  - You cannot nest functions to be used with a single %SYSFUNC. However, you can nest %SYSFUNC calls such as
    %LET x=%SYSFUNC(TRIM(%SYSFUNC(LEFT(&ind))));

- arguments: function arguments

- format: optional format to apply to the result of function

Example 1

Checking Existence of SAS Data Set

within a macro definition, assuming that data is a parameter of the macro

%IF %SYSFUNC(exist(&data)) %THEN %DO;
...
Example 2
Determining Rows and Columns in a SAS Data Set

```sas
%macro obsnvars(data);
%global dset nvars nobs;
%let dset=&data;
%let dsid = %sysfunc(open(&dset));
%if &dsid %then
  %do;
    %let nobs =%sysfunc(attrn(&dsid,NOBS));
    %let nvars=%sysfunc(attrn(&dsid,NVARS));
    %let rc = %sysfunc(close(&dsid));
  %end;
%else
  %put Open for data set &dset failed - %sysfunc(sysmsg());
%mend obsnvars;

%obsnvars(work.mydata)
```

Using SAS Autocall Macros

The system option **MAUTOSOURCE** must be in effect before any of these SAS autocall macros can be used. To turn off SAS macro autocall facility, issue the system option **NOMAUTOSOURCE**.
DATATYP Autocall Macro
Returns the data type of a value

\%DATATYP (text | text-expression)

\%MACRO sq (x);
   \%IF \%DATATYP (&x) = NUMERIC \%THEN &x ** 2;
   \%ELSE ;
\%MEND sq;

\%LET a = \%SYSEVALF (\%sq (2.5));
\%LET b = \%sq (abc);
\%PUT b = &b;

Selected Character Autocall Macros

Except for \%VERIFY, all others have the form
\%func (text | text-expression)

- \%VERIFY (source, excerpt): returns the position of the first character in source which does not exist in excerpt.
- \%LEFT & \%QLEFT Left-align an argument by removing leading blanks
- \%TRIM & \%QTRIM Trim trailing blanks
- \%LOWCASE & \%QLOWCASE change uppercase characters to lowercase
- \%CMPRESS & \%QCMPRESS compress multiple blanks and remove leading and trailing blanks
RESOLVE Function
resolves the value of a text expression during DATA step execution

**RESOLVE**(argument)
where argument can be:

- a text expression enclosed in single quotation marks
- the name of a DATA step variable whose value is a text expression
- a character expression that produces a text expression for resolution by the macro facility

**Examples**

- position = RESOLVE(’%look()’);
- x = RESOLVE(’&base’);
- pos = ’&origin’;
  - x = RESOLVE(pos);
- x = RESOLVE(’&dept’ || LEFT(d));
%SYSCALL Statement

invokes all SAS call routines, except LABEL, VNAME, SYMPUT, and EXECUTE. It also allows user-written call routines created with SAS/TOOLKIT. Example:

`%let seed = 0;
%let n = 5;
%LET p = 0.6;
%LET successes = 0;
%SYSCALL RANBIN(seed, n, p, successes);
%PUT seed=&seed, number of successes=&successes;`