Creating Macro Variables

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>%GLOBAL macro-variable-1 ...macro-variable-n;</code></td>
<td>Creates a macro variable that is available during the execution of the entire SAS session.</td>
</tr>
<tr>
<td><code>%LET variable=value;</code></td>
<td>Creates a macro variable and assigns it a value.</td>
</tr>
<tr>
<td><code>%LOCAL macro-variable-1 ...macro-variable-n;</code></td>
<td>Creates a macro variable that is available only during the execution of the macro where it is defined.</td>
</tr>
</tbody>
</table>

Defining a Macro

%MACRO macro-name<(parameter-list)>;
  macro-text
%MEND <macro-name>;

The parameter-list can be:
<positional-parameter-1, ...positional-parameter-n> or <keyword-1=value-1, ....keyword-n=value-n>

Calling a Macro

%macro-name
%macro-name(positional-parameter-1, ...positional-parameter-n)
%macro-name(keyword-1=value-1, ....keyword-n=value-n)

Referencing a Macro Variable

Use the name of the macro variable with an ampersand.
&macro-variable;

Macro Character Functions

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>%INDEX (source, string)</code></td>
<td>Determines the position of the first character of a string within another string.</td>
</tr>
<tr>
<td><code>%SCAN (argument, n, &lt;charlist &lt;,modifiers&gt;&gt;&gt;)</code></td>
<td>Searches the argument and returns the nth word.</td>
</tr>
<tr>
<td><code>%SUBSTR (argument, position, &lt;,length&gt;)</code></td>
<td>Produces a substring of character string (argument) by extracting the specified number of characters (length) beginning at the specified starting position.</td>
</tr>
<tr>
<td>`%UPCASE (character-string</td>
<td>text-expression)`</td>
</tr>
</tbody>
</table>

SAS Functions with Macro Variables

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>%EVAL (arithmetic or logical expression)</code></td>
<td>Evaluates arithmetic and logical expressions using integer arithmetic.</td>
</tr>
<tr>
<td><code>%SYSEVALF (expression &lt;,conversion-type&gt;)</code></td>
<td>Evaluates arithmetic and logical expressions using floating-point arithmetic.</td>
</tr>
<tr>
<td><code>%SYSFUNC (function (argument-1 &lt;...argument-n&gt;) &lt;,format&gt;)</code></td>
<td>Executes SAS functions or user-written functions in the macro facility.</td>
</tr>
</tbody>
</table>

Troubleshooting Macro Variable References

Enables you to write your own messages to the SAS log.
%PUT text;

Deletes the specified variables from the macro global symbol table.
%SYMDEL macro-variable-1 <...macro-variable-n><option>;

Copyright © 2019 SAS Institute Inc. Cary, NC, USA. All rights reserved.
### Masking Special Characters

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>%STR(argument)</code></td>
<td>Hides the usual meaning of a semicolon(;) so it appears as constant text.</td>
</tr>
<tr>
<td><code>%NRSTR (character-string)</code></td>
<td>Hides the usual meaning of an ampersand (&amp;) or a percent sign (%) so they appear as constant text.</td>
</tr>
<tr>
<td><code>%SUPERQ (argument)</code></td>
<td>Masks all special characters and mnemonic operators at macro execution but prevents further resolution of the value.</td>
</tr>
<tr>
<td>`%BQUOTE(character-string</td>
<td>text-expression)`</td>
</tr>
<tr>
<td>`%QUPCASE(character-string</td>
<td>text-expression)`</td>
</tr>
<tr>
<td><code>%SUBSTR(argument, position &lt;,length&gt;)</code></td>
<td>Produces a substring of a character string.</td>
</tr>
<tr>
<td><code>%SCAN(argument, n &lt;,charlist&lt;, modifiers&gt;&gt;)</code></td>
<td>Searches for a word and masks special characters and mnemonic operators.</td>
</tr>
<tr>
<td><code>%SYSFUNC(function( arguments) &lt;,format&gt;)</code></td>
<td>Executes functions and masks special characters and mnemonic operators.</td>
</tr>
</tbody>
</table>

### Options

- `OPTIONS MCOMPILENOTE=NONE | NOAUTOCALL | ALL;`  
- `OPTIONS MPRINT | NOMPRINT;`  
- `OPTIONS MLOGIC | NOMLOGIC;`  
- `OPTIONS MAUTOSOURCE | NOAUTOSOURCE;`  

### Creating Macros in SQL

```
PROC SQL NOPRINT;  
SELECT column1<,column2,...>  
INTO :macro-variable-1<,:macro-variable-2,...>  
<TRIMMED>  
FROM table-1 | view-1  
<WHERE expression>  
<other clauses>;  
QUIT;
```

### DATA Step Interface

```
CALL SYMPUTX(macro-variable-name, value <,symbol-table>);  
PUT(source,format.);  
```

### Advanced Macro Techniques

```
%INCLUDE file-specification </SOURCE2>;  
DOSUBL(text-string);
```

### Conditional Processing

```
%IF expression %THEN text;  
<%ELSE text>;  
%IF expression %THEN %DO;  
text and/or macro language statements;  
%END;  
%ELSE %DO;  
text and/or macro language statements;  
%END;  
%DO index-variable=start %TO stop <%BY increment>;  
text  
%END;
```