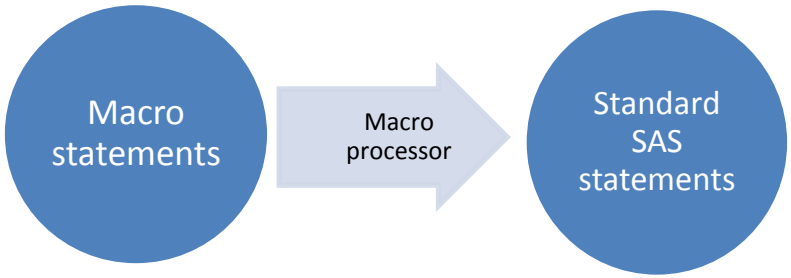
Chapter 9. Macros

9.1. Macros

- A macro is a way to *automate* a task you perform repeatedly or on a regular basis.
- A series of commands and cations can be stored and run whenever needed.
- Macros can make the development and maintenance of production programs much easier.
  - Avoid repetitious SAS code.
  - Create generalizable and flexible SAS code.
  - Conditionally execute DATA steps and PROC steps.
  - Pass information from one part of a SAS job to another.
  - Make one small change and have SAS echo that change throughout the program.
  - Store macros in a central location and share them between programs and between programmers.

- Macro processor



  - Standard SAS program: SAS complies and immediately executes it.
  - Macro: SAS must pass the macro statements to the macro processor that resolves them, generating standard SAS code.
  - 'meta-programming': Write a program that writes a program.
- SAS macro code consists of two basic parts: macros and macro variables

9.2. Key Symbols

- *&name* (Macro variable reference)
  - Name of macro variables are prefixed with an ampersand (&).
  - It does not belong to a dataset, and its value is always character.
  - This value could be a variable name, a numeral, or any text that you want to substituted into your program.

- *%name* (Macro call)
  - Name of macros are prefixed with a percent sign (%).
  - Larger piece of a program that may contain complex logic including complete DATA steps and PROC steps and macro statements.
  - e.g. %DO and %END, %IF-%THEN/%ELSE.

9.3. Macro Variables

- Efficient way of replacing text strings in SAS code
- Can be defined within a macro definition (local) or within a statement that is outside a macro definition (global).
- Macro variables defined by SAS: When you invoke SAS, the macro processor creates automatic macro variable that supply information related to the SAS session.
- %LET: Assign a value to a macro variable.

Example

Before

```
%let iterations=10;
%let country = Canada;

do i=1 to &iterations;
title "Addresses in &country";[2]
```

After
(Resolved by macro processor)

```
do i=1 to 10;
title "Addresses in Canada";
```

[2] In open code (anywhere outside a macro definition), the macro variables should be referenced only within double quotation marks. Macro processor does not look for macros inside single quotation mark.

- Some automatic SAS macro variables

Variable	Description
SYSDATE	Current date
SYSDAY	Current day of the week
SYSTIME	Starting time of job
SYSDSN	Last SAS dataset built
SYSINFO	System information given by some PROCs
SYSSCP	Operating system where SAS is running
SYSVER	SAS version

9.4. Macro functions

- Process one or more arguments and produce a result.
- Used in both macro definitions and open code. (i.e. inside or outside the macro)
- Example: %LENGTH, %EVAL, %UPCASE, %PUT

Example

Raw Data

Obs	obs	Gender	Type	Agegroup	wbc	rbc	chol
1	1	Female	AB	Young	7710	7.40	258
2	2	Male	AB	Old	6560	4.70	.
3	3	Male	A	Young	5690	7.53	184
4	4	Male	B	Old	6680	6.85	.
5	5	Male	A	Young	.	7.72	187

SAS Code

```
%let bc= blood cell;

data blood;
     set blood;
     label wbc = "White &bc"
           rbc = "Red &bc"
           chol = "Cholesterol";
run;
```

Output

Obs	obs	Gender	Type	Agegroup	White blood cell	Red blood cell	Cholesterol
1	1	Female	AB	Young	7710	7.40	258
2	2	Male	AB	Old	6560	4.70	.
3	3	Male	A	Young	5690	7.53	184
4	4	Male	B	Old	6680	6.85	.
5	5	Male	A	Young	.	7.72	187

```
%let var_list = rbc wbc chol;
proc means data=blood n mean min max
maxdec=1;
   var &var_list;
run;
```

Variable	Label	N	Mean	Minimum	Maximum
rbc	Red blood cell	916	5.5	1.7	8.8
wbc	White blood cell	908	7043.0	4070.0	10550.0
chol	Cholesterol	795	201.4	17.0	331.0

```
title "It is &systime on &sysday,
&sysdate.";
proc print data=blood (obs=5) noobs;
run;
```

It is 00:23 on Wednesday, 14FEB18

obs	Gender	Type	Agegroup	wbc	rbc	chol
1	Female	AB	Young	7710	7.40	258
2	Male	AB	Old	6560	4.70	.
3	Male	A	Young	5690	7.53	184
4	Male	B	Old	6680	6.85	.
5	Male	A	Young	.	7.72	187

9.5. Macro Programs

General Syntax

```
* Macro programs;
%macro macro-name(list-of-parameters);
Macro-text (Macro definition)
%mend macro-name;

* Invoke the macro;
%macro-name(list-of-parameters);
```

- %MACRO: Tell SAS that this is the beginning of a macro.
- %MEND: Mark the end.
- The macro-name in %MEND statement is optional, but recommended for easier debugging.
- Macro-text: A set of statements
- Invoking a macro: Add the percent sign prefix to its name.
- Macros with conditional logic: Combine macros and macro variables.

9.6. Tips: How to Avoid Macro Errors

- Develop your program in a *piecewise* fashion.
- Write your code in standard SAS code and make sure that it is bug-free. Then, convert it to macro logic by adding one feature at a time.

Example

```
* MACRO with PROC FREQ;
%macro freqmac(datain, var1, var2);
proc freq data=&datain;
     table &var1*&var2;
run;
%mend freqmac;

%freqmac(blood, gender, type);
```

```
* Macros with conditional logic;
%macro report;
%if &sysday = Friday %then %do;
proc print data=blood (obs=10);
     where gender = "Female";
run;
%end;
%else %do;
proc print data=blood (obs=10);
     where gender ne "Female";
run;
%end;
%mend report;

%report;
```