AGENDA

- Role of macro processing
- Macro variable assignments
- Substitution of user-defined values
- Macro functions
- DATA Step functions for Macro processing
- Macro definition and execution
- Macro looping and concatenation
- Illustrations
Purpose of the Macro Facility

OVERVIEW

- Symbolic substitution within SAS code
- Permits passing parameters across step-boundaries
- Automate generation of SAS code
- Dynamic generation of SAS code
- Permits conditional branching of SAS program logic
Use-Case 1: Macro Variable Assignments

Three methods to insert macro variable values into the “symbol table”

1. %let assignment
   %let city = Seattle;

2. Data Step CALL SYMPUT function
   call symput (‘city’, ‘Fargo’);

3. INTO statement of PROC SQL
   select location into :city ....
Use-Case 1: %LET Assignment Behaviors

- Always results in a character string
- Maximum length is 65,534 characters
- Minimum length is 0
- Mathematical expressions are **not** evaluated
- Values are case-sensitive
- Quotes bounding literals are stored as part of the SAS Macro value
- Leading and trailing blanks **are removed** from value before assignment is made
Use Case 1: %LET Assignment Behaviors

```
options symbolgen;
%let city1 = Fargo;
%let city2 = 'Fargo';
%put Macro Variable city1 Resolves to &city1;
%put Macro Variable city2 Resolves to &city2;
```

```
9  options symbolgen;
10  %let city1 = Fargo;
11  %let city2 = 'Fargo';
12  %put Macro Variable city1 Resolves to &city1;
SYMBOLGEN: Macro variable CITY1 resolves to Fargo
Macro Variable city1 Resolves to Fargo
13  %put Macro Variable city2 Resolves to &city2;
SYMBOLGEN: Macro variable CITY2 resolves to 'Fargo'
Macro Variable city2 Resolves to 'Fargo'
```
Use Case 1: CALL SYMPUT Routine

Assigns a value created in the DATA step to a Macro variable

Syntax:
```
CALL SYMPUT(macro-variable, value);
```

Where `macro-variable` can be:
- Character string in quotes to create a new Macro variable
- Name of a SAS Character variable
- Character expression

Where `value` can be:
- Character string in quotes
- SAS numeric or character variable
- DATA step expression
Use Case 1: CALL SYMPUT Routine

data _null_; call symput('venue','American Crystal Sugar Building');
%put We are in the &venue;
WARNING: Apparent symbolic reference VENUE not resolved.
We are in the &venue
run;

NOTE: DATA statement used (Total process time):
   real time        0.28 seconds
   cpu time         0.01 seconds

data _null_; %put Now, We are in the &venue;
Now, We are in the American Crystal Sugar Building
**Use Case 1: CALL SYMPUT Routine**

```sas
/* data teachers; 
input topic $10. name $10.; 
call symput(topic,name); 
data _null_; 
11 data _null_; 
12 %put Astronomy is &astronomy; 
13 %put Biology is &biology; 
14 %put Agronomy is &agronomy; 
Astronomy is Rahim 
Biology is Terry 
Agronomy is Joseph 
;;;; 
run; 
*/
data _null_; 
%put Astronomy is &astronomy; 
%put Biology is &biology; 
%put Agronomy is &agronomy; 
```
Use Case 1: SYMGET Function

```
filename filelst pipe 'dir "C:\users\sastrb\" /ad /b';

data folders;
  length name $256 ;
  infile filelst length=reclen;
    input name $varying256. reclen ;
    call symput('dir'|left(_n_),name);
run ;

data _null_
  set folders;
    folder_name = symget('dir'||left(_n_));
    put folder_name;
run;
```
Use Case 1: SQL SELECT INTO

```sql
options ls=80;
%global table_name;
proc sql noprofit;
    select memname into :dsname separated by "+" 
    from dictionary.tables
    where libname="SASUSER" and memtype="DATA" ;
quit;
```

**NOTE:** PROCEDURE SQL used (Total process time):
- real time 0.01 seconds
- cpu time 0.01 seconds

```sql
%put &dsname;
ADVISE+ADVERTISE+BLOCKED+BOSTONMARATHON+BOSTONNEWYORK+BOSTONTOP100+CANCER+CARS1993+CATALOGSALES+CATALOGSALESINCLEVEL+CEMENT+CEREALS+DRUG+EDU_SALES+EXACT+FINAL+FINAL2+FINAL2_BACKUP+FINAL_BACKUP+FITNESS+FIXWIDTH+FREQCHARFREQUENCYFORADHOC+FRMUNIX+FUNCS+GRADES+HOSPICE+IMPW+LIBREFS+MARKET+MEAN+MEANSUMMARY+STATS+MYSUBS+NEED
_PREDICTION+PAINTS+PAINTSBLOCKEDROAD+PARSELLOG+PREDLINREGPREDICTION
SFITNESS+RISECEREAL+SASMBC+SASPARM+STABSUMMARY+TABLES+IMPW+STATE_STATS+STUDENTS+THANKSGIVING_WEEK+TSDSTIMESERIESDATAQUERY_FOR_0001+TSDSTIMESERIESDATAQUERY_FOR_R_MC+TSDSTIMESERIESDATAQUERY_FOR_UNIB+TSDSTIMESERIESDATAUNI.BLOCK2+ULCER+UNIVCHARUNITYVARIATEFORADHOC+USCOLLEGES1991+VEHICLESAFETY+VEENEER+ASTDCOL
```
Use Case 1: SQL SELECT INTO

```sas
proc sql noprint;
   select memname into :memname1 - :memname100
      from dictionary.tables
      where libname='SASUSER' and memtype='DATA';
quit;
NOTE: PROCEDURE SQL used (Total process time):
    real time 0.21 seconds
    cpu time 0.04 seconds

WARNING: Apparent symbolic reference MEMNAME100 not resolved.
```

```sas
%put &memname1 &memname2 &memname3 &memname100;
advertise advertiseblocked bostonmarathon &memname100
```
Use Case 2: Macro Substitution

options symbolgen;
%let make = Audi;
proc print
   data=sashelp.cars(where=(make=&make));
SYMBOLGEN: Macro variable MAKE resolves to Audi
ERROR: Variable Audi is not on file SASHELP.CARS.
   id make;
   var msrp;
run;

NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE PRINT used (Total process time):
   real time 0.00 seconds
   cpu time 0.00 seconds

proc print
   data=sashelp.cars(where=(make="&make"));
SYMBOLGEN: Macro variable MAKE resolves to Audi
   id make;
   var msrp;
run;

NOTE: There were 19 observations read from the data set SASHELP.CARS.
   WHERE make='Audi';
NOTE: PROCEDURE PRINT used (Total process time):
   real time 0.01 seconds
   cpu time 0.01 seconds
Use Case 2:  %PUT_ALL_

53  %let A_var = Value Assigned to Macro Variable A_var;
54  %put _all_;
GLOBAL SQLOBS 54
GLOBAL MEMNAME3 BOSTONMARATHON
GLOBAL MEMNAME30 MEANSUMMARYSTATS
GLOBAL MEMNAME31 MYSUBS
GLOBAL MEMNAME32 NEED_PREDICTION
GLOBAL MEMNAME33 PAINTS
GLOBAL MEMNAME34 PAINTSBLOCKEDROAD
GLOBAL MEMNAME35 PARSELOG
GLOBAL MEMNAME36 PREDICTED_OUT
GLOBAL MEMNAME37 PREDLINREGPREDICTIONSFITNESS
GLOBAL MEMNAME38 RISECEREAL
GLOBAL MEMNAME39 SASMBC
GLOBAL SQLOOPS 125
GLOBAL MEMNAME1 ADVERTISE
GLOBAL MEMNAME10 CEMENT
GLOBAL MEMNAME11 CEREALS
GLOBAL MEMNAME12 DRUG
GLOBAL MEMNAME13 EDU_SALES
GLOBAL MEMNAME14 EXACT
GLOBAL MEMNAME15 FINAL
GLOBAL MEMNAME16 FINAL2
GLOBAL MEMNAME17 FINAL2_BACKUP
GLOBAL MEMNAME18 FINAL_BACKUP
GLOBAL MEMNAME19 FITNESS
GLOBAL A_VAR Value Assigned to Macro Variable A_var
Use Case 2: %PUT_ALL_

AUTOMATIC AFDSID 0
AUTOMATIC AFDSNAME
AUTOMATIC AFLIB
AUTOMATIC AFSTR1
AUTOMATIC AFSTR2
AUTOMATIC FSPBDV
AUTOMATIC SYSADDBITS 64
AUTOMATIC SYSBUFFR
AUTOMATIC SYSCC 3000
AUTOMATIC SYSCHARDATA 1
AUTOMATIC SYSCMD
AUTOMATIC SYSDATE 27MAY13
AUTOMATIC SYSDATE9 27MAY2013
AUTOMATIC SYSDAY Monday
AUTOMATIC SYSENV
AUTOMATIC SYSDM 0
AUTOMATIC SYSDSN _NULL_
AUTOMATIC SYSENCODING wlatin1
AUTOMATIC SYSENDCASE LITTLE
AUTOMATIC SYSENV FORE
AUTOMATIC SYSER 0
AUTOMATIC SYSSRORTEXT Variable Audi is not on file SASHELP.CARS.
Use Case 2: Macro Substitution for SAS Date Values

%put "01%substr(&sysdate9,3,5)10"d;
"01MAY2010"d

proc print data=sasuser.edu_sales(where = (
    begin_date between
    "01%substr(&sysdate9,3,5)10"d and
    "&sysdate9"d));

title "All Courses Held Between 01%substr(&sysdate9,3,5)10 and &sysdate9";

run;

NOTE: There were 88 observations read from the data set SASUSER.EDU_SALES.
    WHERE (begin_date='01MAY2010'D and begin_date<='27MAY2013'D);
NOTE: PROCEDURE PRINT used (Total process time):
    real time 0.07 seconds
cpu time 0.06 seconds
Use Case 3: MACRO FUNCTIONS

- Selected character string manipulation functions:

  %UPCASE translates letters from lowercase to uppercase
  %SUBSTR extracts a substring from a character string
  %SCAN extracts a word from a character string
  %INDEX searches a character string for specified text
  %LENGTH returns character string length or text expression

Other functions:
%QSYSFUNC executes SAS functions
%EVAL performs arithmetic and logical operations
%BQUOTE protects blanks and other special characters
Use Case 3:  `%QSYSFUNC` FUNCTION

```
data _null_;  
%let when = %left(%qsysfunc(date(),worddate14.));  
%let ttime = %left(%qsysfunc(time(),time14.));  

%put &ttime on &when;  
23:14:46 on May 27, 2013
```
Use Case 3:  &QSYSFUNC FUNCTION WITH EXIST FUNCTION

144  data _null_;  
145  %let dataset_name=sashelp.cars;  
146  %let exist=%qsysfunc(exist(%upcase(&dataset_name)));  
147  if &exist = 1 then put "Data Set &dataset_name Found";  
148  else put "Data Set &dataset_name Not Found";  
149  run;  

Data Set sashelp.cars Found
NOTE: DATA statement used (Total process time):
    real time           0.00 seconds
    cpu time           0.00 seconds
Use Case 3: RESOLVE FUNCTION WITH CALL SYMPUT

```sas
165 data _null_;
166 call symput('venue', 'American Crystal Sugar Building');
167 location=resolve("&venue");
168 put "The location is " location;
169 run;
```

The location is American Crystal Sugar Building

NOTE: DATA statement used (Total process time):
  real time 0.00 seconds
  cpu time 0.00 seconds
Use Case 4: MACRO VARIABLE CONCATENATION

Use a dot ‘.’ to Append When Text to the Right of the Macro Variable

229  %let libref = sastrb;
230  %let dsname = &userid.test;
231  %put &dsname;
sastrbtest

Use 2 dots to Insert a dot in the Results

233  %let libref = sastrb;
234  %let dsname = &userid..test;
235  %put &dsname;
sastrb.test
Use Case 4: MACRO VARIABLE CONCATENATION

221 %let table = my_table;
222 %put permlib.&table&sysdate;
permlib.my_table27MAY13
Use Case 4: MACRO VARIABLE CONCATENATION

208  %* Build a Table name with the Form;
209  %* LIBREF.table_DDMMMYYYY
210  %let lib1 = saslib;
211  %let table1 = example1;
212  %put &lib1.&table1_&sysdate;
** WARNING: Apparent symbolic reference TABLE1_ not resolved.**
saslib&table1_27MAY13

214  %let lib2 = saslib;
215  %let table2 = example1;
216  %put &lib2..&table2%str(_)&sysdate9;
      saslib.example1_27MAY2013
The general form for defining a SAS Macro:

```
%MACRO macro-name;
   macro-text
%MEND <macro-name>;
```

Macro-name follows SAS naming conventions and Macro-text can include:
- Any text
- SAS Statement or steps
- Macro variables, functions, statements, or calls
- Any combination of the above
Use Case 5: MACRO DEFINITION AND EXECUTION

190 options mcompilenote=all;
191 %macro time;
192   %put The current time is %sysfunc(time(),timeampm11.2);.
193 %mend time;
NOTE: The macro TIME completed compilation without errors.
      5 instructions 104 bytes.

194
195 %time;
The current time is 12:03:41 AM.
Use Case 5: MACRO DEFINITION AND EXECUTION

294  %macro obs_count(libname=, memname=);
295  %global nobs;
296  proc sql noprint;
297      select nobs into :nobs
298            from dictionary.tables
299            where libname = %upcase("&libname")
300                and memname = %upcase("&memname");
301  quit;
302  quit;
303  %put The SAS Data Set &libname..&memname has %left(&nobs) Records;
304  %mend;

NOTE: The macro OBS_COUNT completed compilation without errors.
       13 instructions 468 bytes.

305
306  %obs_count(libname=sasuser, memname=cars1993);
NOTE: PROCEDURE SQL used (Total process time):
       real time          0.01 seconds
       cpu time           0.01 seconds

The SAS Data Set sasuser.cars1993 has 92 Records
Use Case 6: MACRO LOOPING

Requirement: From EDU_SALES, for each teacher, create a table containing records of the courses taught:

Uses PROC FREQ to create the Teacher table with one row per teacher

Use a DATA Step to loop though the Teacher table to:

- Create Macro Variables dataset1—dataset<n> to name new tables
- Create Macro Variables teacher1—teacher<n> to name columns
- Create Macro Variable &n_rec to store record count

Define the %split_table macro with %do/%end looping

- Use the teacher table as an index to search the EDU_SALES table
- Loop executes 1 to &n_rec times

Call the Macro %split_table macro to generate the individual tables
### Use Case 6: MACRO LOOPING

<table>
<thead>
<tr>
<th>Location</th>
<th>Date</th>
<th>Teacher</th>
<th>Course_Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dallas</td>
<td>01JAN2010</td>
<td>Tally, Ms. Julia</td>
<td>Database Design</td>
</tr>
<tr>
<td>Boston</td>
<td>01JAN2010</td>
<td>Berthan, Ms. Judy</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>Seattle</td>
<td>01JAN2010</td>
<td>Wickam, Dr. Alice</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>Seattle</td>
<td>01JAN2009</td>
<td>Hallis, Dr. George</td>
<td>Basic Telecommunications</td>
</tr>
<tr>
<td>Dallas</td>
<td>01JAN2010</td>
<td>Hallis, Dr. George</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>Boston</td>
<td>01JAN2010</td>
<td>Hallis, Dr. George</td>
<td>Basic Telecommunications</td>
</tr>
</tbody>
</table>
Use Case 6: MACRO LOOPING

12 proc freq data = sasuser.edu_sales noprint;
13   tables teacher/ out=teachers(keep=teacher);
14
NOTE: There were 434 observations read from the data set SASUSER.EDU_SALES.
NOTE: The data set WORK.TEACHERS has 5 observations and 1 variables.
NOTE: PROCEDURE FREQ used (Total process time):
   real time           0.07 seconds
   cpu time            0.01 seconds

15 data _null_;
16   set teachers;
17   call symput('dataset'||left(_n_),compress(scan(teacher,1)));  
18   call symput('teacher'||left(_n_),teacher);
19   call symput('n_rec',trim(left(put(_n_,8.))));
20
21 run;
22
23 %put Data Set is &dataset1, Teacher is &teacher1 %left(%str(&) Obs=&n_rec);
   Data Set is Berthan, Teacher is Berthan, Ms. Judy   & Obs=5
Use Case 6: MACRO LOOPING

```
24  %macro split_table;
25    %do i = 1 %to &n_rec;
26        data &dataset&i;
27        set sasuser.edu_sales(where=(teacher="&teacher&i");
28    %end;
29  %end;
30 %mend;
31 %split_table;
```

NOTE: There were 66 observations read from the data set SASUSER.EDU_SALES.
WHERE teacher='Berthan, Ms. Judy';

NOTE: The data set WORK.BERTHAN has 66 observations and 12 variables.
NOTE: DATA statement used (Total process time):
    real time          0.60 seconds
    cpu time           0.00 seconds

NOTE: There were 74 observations read from the data set SASUSER.EDU_SALES.
WHERE teacher='Forest, Mr. Peter';

NOTE: The data set WORK.FOREST has 74 observations and 12 variables.
NOTE: DATA statement used (Total process time):
    real time          0.04 seconds
    cpu time           0.01 seconds
References

An Introduction to SAS Character Functions (Including New SAS Functions), by Ron Cody

SAS Functions Old and New, by Ron Coleman, Internal SAS Presentation

SAS Date/Time Processing, by Cynthia Johnson, Internal SAS Presentation

SAS 9.3 Functions and CALL Routines: Reference at: support.sas.com