# SAS Macro: Beyond the Basics

VINCE DELGOBBO, SAS



## Agenda

- The basics
- Macro with a parameter
- Iterative macro processing
- Conditional macro processing
- Macro timing and resolution
- Interfacing macro and DATA step
- Introduction to macro quoting

# The Basics

3



## Macro Language Basics

- Good replacement for common/repetitive tasks
- Macro **variables** and **macros**
- Can create global variables used anywhere
- Can create variables available only within a macro
- Macro: collection of macro and SAS statements
- String-based language – produces text
- Macro "triggers": & and %
  - **`&VARIABLE-NAME`** resolves a macro variable into text string
  - **`%MACRO-NAME`** executes a macro
- Macro processing happens before anything else
- Need double quotes (") if used within quotes

4

## Macro Language Basics – Examples

- Open code

```
%let CITY=Denver;
*  Lots of SAS code here;
title1 "Report for &CITY";
proc print ... ; run; quit;
```

`CITY` is a **global** macro variable

Need double quotes to resolve

- Convert to a macro

```
%macro do_something(CITY=);
  %*  Lots of SAS code here;
  title1 "Report for &CITY";
  proc print ... ; run; quit;
%mend do_something;

%DO_SOMETHING(city=Denver)
```

`CITY` is a **local** macro variable

5



## Macro with a Parameter

6

## Macro with a Parameter

```
%macro claiminfo(YEAR=);

title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;

title1 "Summary for &YEAR";
proc means data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;

%mend claiminfo;
```

7



## Macro with a Parameter

```
%macro claiminfo(YEAR=);

title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;

title1 "Summary for &YEAR";
proc means data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;

%mend claiminfo;
```

**YEAR** is a local macro variable

Need double quotes to resolve

**SVC_Year** is a character variable

8

## Macro with a Parameter

```
%macro claiminfo(YEAR=);
```

YEAR is 2005

```
title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;

title1 "Summary for &YEAR";
proc means data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;

%mend claiminfo;

options mprint;

%CLAIMINFO(year=2005)
```

9



## Log for a Single Year

```
MPRINT(CLAIMINFO):   title1 "Report for 2005";
MPRINT(CLAIMINFO):   proc print data=health.claims_sample;
MPRINT(CLAIMINFO):   var ... ;
MPRINT(CLAIMINFO):   where (SVC_Year eq "2005");
MPRINT(CLAIMINFO):   run;

NOTE: There were 337 observations read from the data set
      HEALTH.CLAIMS_SAMPLE.
      WHERE SVC_Year="2005";

MPRINT(CLAIMINFO):   title1 "Summary for 2005";
MPRINT(CLAIMINFO):   proc means data=health.claims_sample;
MPRINT(CLAIMINFO):   var ... ;
MPRINT(CLAIMINFO):   where (SVC_Year eq "2005");
MPRINT(CLAIMINFO):   run;

NOTE: There were 337 observations read from the data set
      HEALTH.CLAIMS_SAMPLE.
      WHERE SVC_Year="2005";
```

10

# Output for a Single Year

**Report for 2005**

| Obs | ProviderID | MemberID | CHGAMT | LNDISCAMT | EERESP | PaidAmt |
|---|---|---|---|---|---|---|
| 9 | 303-60-2456 | 012-01-8821 | 110.00 | 16.50 | 93.50 | 0.00 |
| 18 | 770-03-1665 | 012-54-4665 | 11.11 | 0.00 | 0.00 | 11.11 |
| 21 | 380-04-3952 | 012-56-7160 | 95.00 | 14.25 | 12.11 | 68.64 |
| 26 | 325-32-3393 | 013-64-4285 | 44.35 | 4.44 | 5.99 | 33.92 |
| 49 | 406-51-3042 | 020-30-5626 | 65.00 | 0.00 | 0.00 | 6.60 |

**Summary for 2005**

The MEANS Procedure

| Variable | Label | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|
| CHGAMT | Charged Amount | 337 | 133.8451335 | 288.5492179 | -188.0000000 | 2451.25 |
| LNDISCAMT | Line Item Discount Amount | 337 | 20.0098220 | 65.4137756 | -46.3500000 | 593.2500000 |
| EERESP | Employee Responsibility | 337 | 29.5309792 | 142.9098598 | -188.0000000 | 2451.25 |
| PaidAmt | Paid Amount | 337 | 81.6817507 | 185.1763225 | -97.3400000 | 1779.75 |

11



# Iterative Macro Processing

12

## "Poor Man's" Iterative Processing

```
%macro claiminfo(YEAR=);
...
%mend claiminfo;

%CLAIMINFO(year=2005)
%CLAIMINFO(year=2006)
%CLAIMINFO(year=2007)
...
```

What's the problem with this approach?

13



## Iterative %DO – General Syntax

```
%macro mymacro;

%do macro-variable = start %to end <%by increment> ;
   %*  Text and macro language statements here;
%end;

%mend mymacro;
```

- NOT valid outside of a macro ("open code")
- Note use of % in macro statements

14

## Start with Macro with a Parameter

```
%macro claiminfo(YEAR=);



  title1 "Report for &YEAR";
  proc print data=health.claims_sample;
    var ... ;
    where (SVC_Year eq "&YEAR");
  run; quit;

  title1 "Summary for &YEAR";
  proc means data=health.claims_sample;
    var ... ;
    where (SVC_Year eq "&YEAR");
  run; quit;


%mend claiminfo;
```

15



## Create Start/End Year Parameters

```
%macro claiminfo(STARTYEAR=, ENDYEAR=);



  title1 "Report for &YEAR";
  proc print data=health.claims_sample;
    var ... ;
    where (SVC_Year eq "&YEAR");
  run; quit;

  title "Summary for &YEAR";
  proc means data=health.claims_sample;
    var ... ;
    where (SVC_Year eq "&YEAR");
  run; quit;


%mend claiminfo;
```

16

## Define Scope of YEAR and add %DO

```
%macro claiminfo(STARTYEAR=, ENDYEAR=);
%local YEAR;

%do YEAR = &STARTYEAR %to &ENDYEAR;
  title1 "Report for &YEAR";
  proc print data=health.claims_sample;
    var ... ;
    where (SVC_Year eq "&YEAR");
  run; quit;

  title1 "Summary for &YEAR";
  proc means data=health.claims_sample;
    var ... ;
    where (SVC_Year eq "&YEAR");
  run; quit;
%end;

%mend claiminfo;
```

17



## Better Iterative Processing

Single call instead of many:

```
%CLAIMINFO(startyear=2005, endyear=2007);
```

18

# Output for Multiple Years

**Report for 2005**

| Obs | ProviderID | MemberID | CHGAMT | LNDISCAMT | EERESP | PaidAmt |
|---|---|---|---|---|---|---|
| 9 | 303-60-2456 | 012-01-8821 | 110.00 | 16.50 | 93.50 | 0.00 |
| 18 | 770-03-1665 | 012-54-4665 | 11.11 | 0.00 | 0.00 | 11.11 |

**Summary for 2005**

The MEANS Procedure

| Variable | Label | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|
| CHGAMT | Charged Amount | 337 | 133.8451335 | 288.5492179 | -188.0000000 | 2451.25 |
| LNDISCAMT | Line Item Discount Amount | 337 | 20.0008220 | 35.4137756 | -46.3500000 | 593.2500000 |

**Report for 2006**

| Obs | ProviderID | MemberID | CHGAMT | LNDISCAMT | EERESP | PaidAmt |
|---|---|---|---|---|---|---|
| 8 | 308-13-7454 | 011-78-4712 | 35.00 | 21.11 | 13.89 | 0.00 |
| 13 | 335-82-9189 | 012-12-1836 | 12.00 | 1.80 | 1.53 | 8.67 |

**Summary for 2006**

The MEANS Procedure

| Variable | Label | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|
| CHGAMT | Charged Amount | 411 | 148.3832847 | 453.4626598 | -33.0000000 | 6385.30 |
| LNDISCAMT | Line Item Discount Amount | 411 | 26.7820438 | 119.2000381 | -4.3000000 | 1595.33 |

Etc.

19



# Conditional Macro Processing

20

## %IF-%THEN-%ELSE – General Syntax

```
%if expression %then action;
```

- NOT valid outside of a macro ("open code")
- Note use of % in macro statements
- %THEN required (no subsetting %IF)
- Common use:

```
%macro mymacro;

%if expression %then %do;
  ...
%end;
%else %do;
  ...
%end;

%mend mymacro;
```

21



## Conditional Macro Processing

Example:

Print the claims report ***OR*** summary for a given year

22

## Start with Macro with a Parameter

```
%macro claiminfo(YEAR=);


title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;


title1 "Summary for &YEAR";
proc means data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;


%mend claiminfo;
```

23



## Add a Parameter for Output Type

```
%macro claiminfo(YEAR=, OUTPUT_TYPE=);


title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;


title1 "Summary for &YEAR";
proc means data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;


%mend claiminfo;
```

24

## Add Conditional Logic

```
%macro claiminfo(YEAR=, OUTPUT_TYPE=);

%if (&OUTPUT_TYPE eq REPORT) %then %do;
title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;
%end;
%else %do;
title1 "Summary for &YEAR";
proc means data=health.claims_sample;
  var ... ;
  where (SVC_Year eq "&YEAR");
run; quit;
%end;

%mend claiminfo;
```

25



## %CLAIMINFO(year=2005, output_type=REPORT)

```
MPRINT(CLAIMINFO):   title1 "Report for 2005";
MPRINT(CLAIMINFO):   proc print data=health.claims_sample;
MPRINT(CLAIMINFO):   var ... ;
MPRINT(CLAIMINFO):   where (SVC_Year eq "2005");
MPRINT(CLAIMINFO):   run;

NOTE: There were 337 observations read from the data set
      HEALTH.CLAIMS_SAMPLE.
      WHERE SVC_Year='2005';
```

Report for 2005

| Obs | ProviderID | MemberID | CHGAMT | LNDISCAMT | EERESP | PaidAmt |
|---|---|---|---|---|---|---|
| 9 | 303-60-2456 | 012-01-8821 | 110.00 | 16.50 | 93.50 | 0.00 |
| 18 | 770-03-1665 | 012-54-4665 | 11.11 | 0.00 | 0.00 | 11.11 |
| 21 | 380-04-3952 | 012-56-7160 | 95.00 | 14.25 | 12.11 | 68.64 |
| 26 | 325-32-3393 | 013-64-4285 | 44.35 | 4.44 | 5.99 | 33.92 |
| 29 | 406-51-3042 | 020-30-5626 | 65.00 | 0.00 | 0.00 | 6.60 |

26

## %CLAIMINFO(year=2005)

```
MPRINT(CLAIMINFO):   title1 "Summary for 2005";
MPRINT(CLAIMINFO):   proc means data=health.claims_sample;
MPRINT(CLAIMINFO):   var ... ;
MPRINT(CLAIMINFO):   where (SVC_Year eq "2005");
MPRINT(CLAIMINFO):   run;

NOTE: There were 337 observations read from the data set
      HEALTH.CLAIMS_SAMPLE.
      WHERE SVC_Year='2005';
```

**Summary for 2005**

The MEANS Procedure

| Variable | Label | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|
| CHGAMT | Charged Amount | 337 | 133.8451335 | 288.5492179 | -188.0000000 | 2451.25 |
| LNDISCAMT | Line Item Discount Amount | 337 | 20.0098220 | 65.4137756 | -46.3500000 | 593.2500000 |
| EERESP | Employee Responsibility | 337 | 29.5309792 | 142.9098598 | -188.0000000 | 2451.25 |
| PaidAmt | Paid Amount | 337 | 81.6817507 | 185.1763225 | -97.3400000 | 1779.75 |

27



# Macro Timing and Resolution

28

## Timing & Resolution of Macro Variables

Compiler

Word Scanner

Input Stack

```
%let YEAR=2006;
title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

Macro Processor

Macro Symbol Table

29



## Word Scanner Detects Macro Trigger

Compiler

Word Scanner

```
%let
```

Input Stack

```
YEAR=2006;
title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

Macro Processor

Macro Symbol Table

30

## %LET Sent to Macro Processor

Compiler

Word Scanner

Input Stack

```
title1 "Report for &YEAR";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

Macro Processor

```
%let YEAR=2006;
```

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

31



## Word Scanner Detects Macro Trigger

Compiler

Word Scanner

```
title1 "Report for
```

Input Stack

```
&YEAR";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

Macro Processor

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

32

## Word Scanner Detects Macro Trigger

**Compiler**

```
title1 "Report for
```

**Word Scanner**

```
&YEAR
```

**Input Stack**

```
";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

**Macro Processor**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

33



## Macro Variable Lookup in Symbol Table

**Compiler**

```
title1 "Report for
```

**Word Scanner**

**Input Stack**

```
";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

**Macro Processor**

```
&YEAR
```

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

34

## Macro Variable Resolved

**Compiler**

```
title1 "Report for
```

**Word Scanner**

**Input Stack**

```
2006";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

**Macro Processor**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

35



## Macro Variable Resolved

**Compiler**

```
title1 "Report for
```

**Word Scanner**

```
2006";
```

**Input Stack**

```
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

**Macro Processor**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

36

## Resolved Macro Variable in TITLE

**Compiler**

```
title1 "Report for 2006";
```

**Word Scanner**

**Input Stack**

```
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "&YEAR");
run; quit;
```

**Macro Processor**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

37



## Word Scanner Detects Macro Trigger

**Compiler**

```
title1 "Report for 2006";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "
```

**Word Scanner**

```
&YEAR
```

**Input Stack**

```
");
run; quit;
```

**Macro Processor**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

38

## Macro Variable Lookup in Symbol Table

**Compiler**

```
title1 "Report for 2006";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "
```

**Word Scanner**

**Input Stack**

```
");
run; quit;
```

**Macro Processor**

```
&YEAR
```

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

39



## Macro Variable Resolved

**Compiler**

```
title1 "Report for 2006";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "
```

**Word Scanner**

**Input Stack**

```
2006");
run; quit;
```

**Macro Processor**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

40

## Resolved Macro Variable in WHERE

**Compiler**

```
title1 "Report for 2006";
proc print data=health.claims_sample;
  var ...;
  where (SVC_Year eq "2006");
run; quit;
```

**Execution**

**Macro Processor**

**Word Scanner**

**Input Stack**

| Macro Symbol Table | |
|---|---|
| YEAR | 2006 |

41



# Interfacing Macro and DATA Step

42

## Timing & Resolution of Macro Variables

- Macro Processing > Code Compilation > Code Execution
- Macro processing happens **before** execution phase
- DATA step variables available **during** execution phase
- Cannot use %LET with DATA step variables

✗ `%let macro-variable-name=value;`

- Use CALL SYMPUTX instead

✓ `call symputx(macro-variable-name, value);`

43



## Total Claim Amount Paid for a Patient

```
data _null_;

set health.claims_sample end=eof;

where (SVC_Year eq '2005') and
      (MemberID eq '317-62-7676');

TotalPaid + PaidAmt;

if (eof) then do;
  put 'In DATA step, ' TotalPaid=;
  %let TOTAL_PAID=TotalPaid;
end;
run;

%put The total paid was &TOTAL_PAID;
```

**PaidAmt** is a data set variable
**TotalPaid** is a computed variable

Quotes not needed

44

## Total Claim Amount Paid for a Patient

```
data _null_;

set health.claims_sample end=eof;

where (SVC_Year eq '2005') and
      (MemberID eq '317-62-7676');

TotalPaid + PaidAmt;

if (eof) then do;
  put 'In DATA step, ' TotalPaid=;
  %let TOTAL_PAID=TotalPaid;
end;
run;

%put The total paid was &TOTAL_PAID;
```

Sent to compiler

Sent to macro processor

45



## Total Claim Amount Paid for a Patient

```
❷ data _null_;

  set health.claims_sample end=eof;

  where (SVC_Year eq '2005') and
        (MemberID eq '317-62-7676');

  TotalPaid + PaidAmt;

  if (eof) then do;
    put 'In DATA step, ' TotalPaid=;
❶   %let TOTAL_PAID=TotalPaid;
  end;
  run;

❸ %put The total paid was &TOTAL_PAID;
```

788.65

TotalPaid

46

## Total Claim Amount Paid for a Patient

```
data _null_;

set health.claims_sample end=eof;

where (SVC_Year eq '2005') and
      (MemberID eq '317-62-7676');

TotalPaid + PaidAmt;

if (eof) then do;
  put 'In DATA step, ' TotalPaid=;
  %let TOTAL_PAID=TotalPaid;
end;
run;

%put The total paid was &TOTAL_PAID;
```

47



## Total Claim Amount Paid for a Patient

```
data _null_;

set health.claims_sample end=eof;

where (SVC_Year eq '2005') and
      (MemberID eq '317-62-7676');

TotalPaid + PaidAmt;

if (eof) then do;
  put 'In DATA step, ' TotalPaid=;
  call symputx('TOTAL_PAID', TotalPaid);
end;
run;

%put The total paid was &TOTAL_PAID;
```

Sent to compiler

48

## Total Claim Amount Paid for a Patient

```
❶ data _null_;

  set health.claims_sample end=eof;

  where (SVC_Year eq '2005') and
        (MemberID eq '317-62-7676');

  TotalPaid + PaidAmt;

  if (eof) then do;
    put 'In DATA step, ' TotalPaid=;
    call symputx('TOTAL_PAID', TotalPaid);
  end;
  run;

❷ %put The total paid was &TOTAL_PAID;
```

788.65

788.65

49



# Introduction to Macro Quoting

50

## Macro Quoting (Masking) Functions

- Needed to eliminate ambiguity

  Examples: Is OR a mnemonic or a state abbreviation?
  Is %Revenue text or a macro invocation?
- Quoting masks mnemonics and special characters
- Mnemonics & special characters treated as text

  Examples: OR is treated as a state abbreviation
  %Revenue treated as text
- Most common functions:
  - %STR and %NRSTR
  - %BQUOTE and %NRBQUOTE
  - %SUPERQ

Hint: Think of "NR" as "Not Resolved"

51



## %STR and %NRSTR

- Mnemonics & special characters masked by %STR:

| | | | |
|---|---|---|---|
| blank | ) | = | NE |
| ; | ( | \| | LE |
| ¬ | + | # | LT |
| ^ | — | AND | GE |
| ~ | * | OR | GT |
| " | / | NOT | IN |
| ' | < | EQ | |
| , (comma) | > | | |

- Parentheses and quotes must be matched
- %NRSTR masks & and %

52

## %STR and %NRSTR

- Use when you can see the characters that need masking – compilation time
- Use % to mask unmatched quotes or parentheses
- Use %NRSTR to prevent resolution of & and %
- Example:

```
%let TITLE=%nrstr(Vince%'s Dog&Pony Show);
```

53



## %STR and %NRSTR

Terminates %LET; statement is incomplete

```
%let MYSTMT=proc print; run; ;
```

```
%let MYSTMT=%str(proc print; run;);
```

Interpreted as macro variable D

```
%let MYTEXT=R&D Staff;
WARNING: Apparent symbolic reference D not resolved.
```

```
%let MYTEXT=%str(R&D Staff);
WARNING: Apparent symbolic reference D not resolved.
```

```
%let MYTEXT=%nrstr(R&D Staff);
```

54

## %BQUOTE and %NBQUOTE

- %BQUOTE masks all the same as %STR:

| | | | |
|---|---|---|---|
| blank | ) | = | NE |
| ; | ( | \| | LE |
| ¬ | + | # | LT |
| ^ | — | AND | GE |
| ~ | * | OR | GT |
| " | / | NOT | IN |
| ' | < | EQ | |
| , (comma) | > | | |

- But ... parentheses and quotes can be unmatched
- %NRBQUOTE masks & and %

55



## %BQUOTE and %NBQUOTE

- Use when resolved values need masking
- Use %NRBQUOTE to mask & and % at execution time
- Attempt to resolve macro references and masks result
- Masked result does not resolve when used

56

## %BQUOTE and %NRBQUOTE

```
%macro claiminfo(YEAR=, OUTPUT_TYPE=);

data _null_;
  *  Some other code here;
  title_text = "Vince's Dog & Pony Show";
  call symputx('TITLE', title_text, 'local');
run;

title1 "Report for &YEAR";

%if (&TITLE ne ) %then %do;
  title2 "&TITLE";
%end;

%* Remainder of macro code;

%mend claiminfo;
```

- ✗ %STR
- ✗ %NRSTR
- ✗ %BQUOTE
- ✓ %NRBQUOTE

57



## %BQUOTE and %NRBQUOTE

```
%macro claiminfo(YEAR=, OUTPUT_TYPE=);

data _null_;
  *  Some other code here;
  title_text = "Vince's Dog & Pony Show";
  call symputx('TITLE', title_text, 'local');
run;

title1 "Report for &YEAR";

%let TITLE=%nrbquote(&TITLE);

%if (&TITLE ne ) %then %do;
  title2 "&TITLE";
%end;

%* Remainder of macro code;

%mend claiminfo;
```

58

## %SUPERQ

- Sledge hammer
- Masks everything
- Prevents ALL resolution of macro elements
- Argument is the NAME of an existing variable (no &)
- Not often used because usually need some resolution

59



## Previous Code with a Different Title

```
%macro claiminfo(YEAR=, OUTPUT_TYPE=);

data _null_;
  *  Some other code here;
  title_text = "Vince's Dog" || '&Pony Show';
  call symputx('TITLE', title_text, 'local');
run;

title1 "Report for &YEAR";

%let TITLE=%nrbquote(&TITLE);

%if (&TITLE ne ) %then %do;
  title2 "&TITLE";
%end;

%* Remainder of macro code;

%mend claiminfo;
```

Vince's Dog&Pony Show

60

```
%CLAIMINFO(year=2005)

NOTE: DATA statement used (Total process time):
      real time           1.22 seconds
      cpu time            0.29 seconds

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

Report for 2005
Vince's Dog&Pony Show

- Appears to have worked
- 1 message instead of 3 ???

61



```
%let PONY=Cat; %CLAIMINFO(year=2005)

NOTE: DATA statement used (Total process time):
      real time           1.22 seconds
      cpu time            0.29 seconds
```

Report for 2005
Vince's DogCat Show

62

## Use %SUPERQ Instead of %NRBQUOTE

```
%macro claiminfo(YEAR=, OUTPUT_TYPE=);

data _null_;
  *  Some other code here;
  title_text = "Vince's Dog" || '&Pony Show';
  call symputx('TITLE', title_text, 'local');
run;

title1 "Report for &YEAR";

✓ %let TITLE=%superq(TITLE);

%if (&TITLE ne ) %then %do;
  title2 "&TITLE";
%end;

%* Remainder of macro code;

%mend claiminfo;
```

Vince's Dog&Pony Show

No & before variable name

63



```
%let PONY=Cat; %CLAIMINFO(year=2005)

NOTE: DATA statement used (Total process time):
      real time           1.22 seconds
      cpu time            0.29 seconds
```

Report for 2005
Vince's Dog&Pony Show

- It really did work
- No warning messages

64

## Conclusion

- Macro variables local or global scope
- Timing: Macro processing happens first
- Use macros to replace common/repetitive tasks
- Iterative and conditional processing
- Macro quoting functions mask special characters
- Quoting: test, test, test your code!

65



## Resources

- SAS 9.4 Macro Language Reference
  http://bit.ly/1rzqhVN
- SAS 9.3 Macro Language Reference
  http://bit.ly/1rPj3fw
- SAS paper search @ lexjansen.com
  http://bit.ly/1tk7OZ0

66

## Contact Information

Please send questions, comments and feedback to:

Vince DelGobbo
sasvcd@SAS.com

If your registered in-house or local SAS users group would like to request this presentation as your annual SAS presentation (as a seminar, talk or workshop) at an upcoming meeting, please submit an online User Group Request Form (support.sas.com/usergroups/namerica/lug-form.html) at least eight weeks in advance.

67