2 for 1 Advanced Macro & SQL

Nebraska User Group
14 May 2024

Charu Shankar
SAS Institute Inc

Instructor

Charu Shankar, SAS® Institute

With a background in computer systems management. SAS Instructor Charu Shankar engages with logic, visuals, and analogies to spark critical thinking since 2007.

Charu curates and delivers unique content on SAS, SQL, Viya, etc. to support users in the adoption of SAS software.

When not coding, Charu teaches yoga and loves to explore Canadian trails with her husky Miko.

Agenda

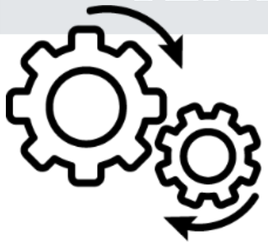
- Why Macro
- Create Macro Variables For Text Substitution
- Using Macro Variables For Text Substitution
- Create Macro Variables with PROC SQL
- Handy Links

Why SAS Macro?

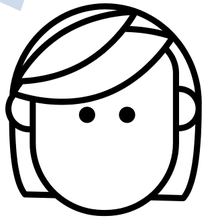
Macro Programming

SAS Macro Facility

TEXT



The SAS macro facility enables you to write code that rewrites itself!

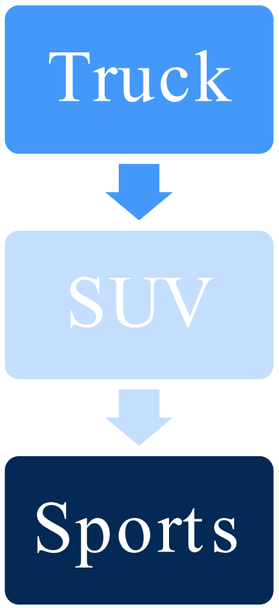


Substituting User-Defined Values

```
title "Trucks by Origin";
proc freq data=sashelp.cars;
     where Type="Truck";
     table Origin;
run;

title "Average Highway MPG for Trucks";
proc means data=sashelp.cars mean maxdec=1;
     where Type="Truck";
     var MPG_Highway;
     class Origin;
run;
```

Easily replace repetitive values.



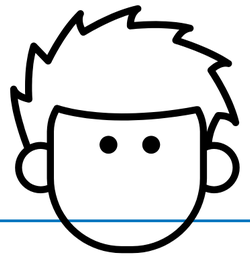
Substituting System Values

Automatically substitute system values into a program.

```
title "Cars List";
footnote "Created at 10:24 AM on May 14, 2024;
title "Trucks by Origin";
proc freq data=sashelp.cars;
     where Type="Truck";
     table Origin;
run;

title "Average Highway MPG for Trucks";
proc means data=sashelp.cars mean maxdec=1;
     where Type="Truck";
     var MPG_Highway;
     class Origin;
run;
```

How can the macro language make your job easier as a SAS programmer?



CarsMacro.sas

Efficiency of Macro-Based Applications

The macro facility processes the text in a program to automate and customize the code.

The macro language won't make your code run faster, but it can reduce your development and maintenance time.

CarsMacro.sas

Creating Macro Variables

SAS Programming Languages

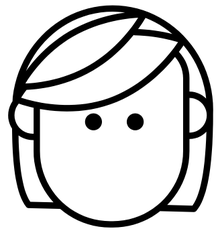
DATA Step	data manipulation
PROCSQL Step	data manipulation and reporting
SAS Procedures	data analysis and reporting
SAS Macro Language	generate SAS program code

Macro Variables

```
title "      s with Horsepower >    ";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="      " and Horsepower>    ;
run;
```

Truck	250
Sedan	150
SUV	200

Macro variables store text that can be used anywhere in our SAS programs.



Macro Variables

Macro variables each have a name and value that are stored in a memory-based symbol table.

Global Symbol Table

Name	Value

Creating Macro Variables with %LET

%LET *name=value*;

Macro variable names:
- follow SAS naming rules
- are stored as uppercase
- are not case sensitive

```
%let type=Truck;
%let hp=250;
```

Global Symbol Table

Name	Value
TYPE	Truck
HP	250

Creating Macro Variables with %LET

%LET *name=value*;

- Case is preserved.
- Leading and trailing blanks are removed.
- It stores 0 to 65,534 (64K) characters.
- The length is dynamically set each time a value is assigned.

```
%let type=Truck;
%let hp=250;
```

Global Symbol Table

Name	Value
TYPE	Truck
HP	250

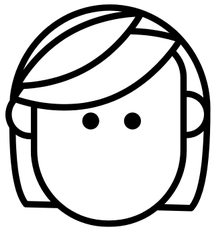
Creating Macro Variables with %LET

```
%let type=Truck;
%let hp=250;
%let type= Sports ;
%let origin=" Europe ";
%let value=;
%let sum=3+4;
%let varlist=Make Model Type;
```

Global Symbol Table

Name	Value
TYPE	Truck
HP	250

Macro variables don't have a type of character or numeric. All values are stored as text.



Creating Macro Variables with %LET

```
%let type=Truck;
%let hp=250;
%let type= Sports ;
%let origin=" Europe ";
%let value=;
%let sum=3+4;
%let varlist=Make Model Type;
```

Global Symbol Table

Name	Value
TYPE	Sports
HP	250

Leading and trailing spaces are removed. The value of an existing macro variable is replaced.

Creating Macro Variables with %LET

```
%let type=Truck;
%let hp=250;
%let type= Sports ;
%let origin=" Europe ";
%let value=;
%let sum=3+4;
%let varlist=Make Model Type;
```

Quotation marks are stored as part of the value.

Global Symbol Table

Name	Value
TYPE	Sports
HP	250
ORIGIN	" Europe "

Creating Macro Variables with %LET

```
%let type=Truck;
%let hp=250;
%let type= Sports ;
%let origin=" Europe ";
%let value=;
%let sum=3+4;
%let varlist=Make Model Type;
```

A null value is stored.

Global Symbol Table

Name	Value
TYPE	Sports
HP	250
ORIGIN	" Europe "
VALUE	

Creating Macro Variables with %LET

```
%let type=Truck;
%let hp=250;
%let type= Sports ;
%let origin=" Europe ";
%let value=;
%let sum=3+4;
%let varlist=Make Model Type;
```

Global Symbol Table

Name	Value
TYPE	Sports
HP	250
ORIGIN	" Europe "
VALUE	
SUM	3+4

Mathematical expressions are not evaluated.

Creating Macro Variables with %LET

```
%let type=Truck;
%let hp=250;
%let type= Sports ;
%let origin=" Europe ";
%let value=;
%let sum=3+4;
%let varlist=Make Model Type;
```

The variable list is stored as a text string.

Global Symbol Table

Name	Value
TYPE	Sports
HP	250
ORIGIN	" Europe "
VALUE	
SUM	3+4
VARLIST	Make Model Type

Quiz

What would be stored as the value of **Mylib**?

```
%let mylib=libname mc1 "s:/workshop";
```

Global Symbol Table

Name	Value
MYLIB	

Quiz – Correct Answer

What would be stored as the value of **Mylib**?

```
%let mylib=libname mc1 "s:/workshop";
```

Global Symbol Table

Name	Value
MYLIB	libname mc1 "s:/workshop"

The semicolon is treated as the conclusion of the %LET statement and is not stored in the macro variable value.

Using Macro Variables

Resolving Macro Variables

&*name*

substitutes the macro variable value into the program

```
%let type=Truck;
%let hp=250;

proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and Horsepower>&hp;
run;
```

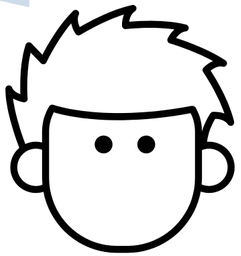
Global Symbol Table

Name	Value
TYPE	Truck
HP	250

Resolving Macro Variables

```
%let type=Truck;
%let hp=250;
title1 "Car Type: &type";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and Horsepower>&hp;
run;
```

Why is **&type** in quotation marks not **&hp**?



Resolving Macro Variables

```
%let type=Truck;
%let hp=250;
title1 "Car Type: &type";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and Horsepower>&hp;
run;
```

Formulate the Where statement correctly

```
where Type="Truck" and Horsepower>250;
```

character expression

numeric expression

Resolving Macro Variables

```
%let type=Truck;
%let hp=250;
title "Car Type: &type";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and Horsepower>&hp;
run;
```

Typically, macro variable values don't include quotation marks.

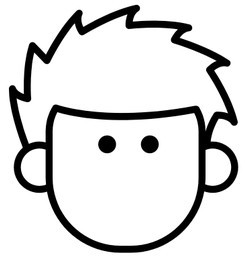
Use double quotation marks where necessary when resolving macro variables.

Troubleshooting

OPTIONS SYMBOLGEN|NOSYMBOLGEN;

```
options symbolgen;
%let type=Truck;
%let hp=250;
title1 "Car Type: &type";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and Horsepower>&hp;
run;
```

The SYMBOLGEN option writes information to the log when macro variable references resolve.



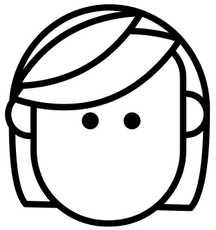
```
80 where Type="&type" and Horsepower>&hp;
SYMBOLGEN: Macro variable TYPE resolves to Truck
SYMBOLGEN: Macro variable HP resolves to 250
```

Quotation Marks

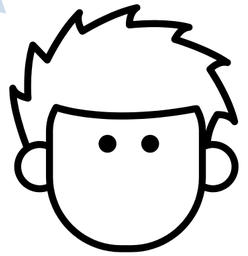
```
title1 "Car Type: &type";
title2 'Car&Power Report';
```

Car Type: Truck
Car&Power Report

Macro triggers in double quotation marks are sent to the macro processor.



Macro triggers in single quotation marks are treated as regular text and are not resolved.



Delimiting Macro Variable References

```
...
%let type=Truck;
title "&types with Horsepower > &hp";
...
```

Trucks with Horsepower > 250

Obs	Make	Model	MSRP	Horsepower
63	Cadillac	Escalade EXT	$52,975	345
85	Chevrolet	Avalanche 1500	$36,100	295
88	Chevrolet	Silverado SS	$40,340	300

desired results

What happens if a macro variable reference is concatenated with trailing text?

Delimiting Macro Variable References

```
...
%let type=Truck;
title "&types with Horsepower > &hp";
...
```

TYPESis not found.

Global Symbol Table

Name	Value
TYPE	Truck
HP	250

```
74 %let type=Truck;
75 %let hp=250;
WARNING: Apparent symbolic reference TYPES not resolved.
SYMBOLGEN: Macro variable HP resolves to 250
76 title "&types with Horsepower > &hp";
```

Delimiting Macro Variable References

```
...
%let type=Truck;
title "&type.s with Horsepower > &hp";
...
```

Use a period to delimit the macro variable name from the text.

```
title "Trucks with Horsepower > 250";
```

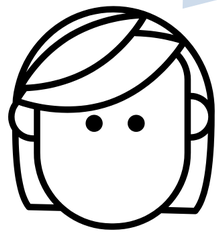
The period does not appear in the resolved text.

Delimiting Macro Variable References

```
footnote "Data Source: &lib..CARS";
proc print data=&lib..cars;
```

Use two periods between the macro variable and table name.

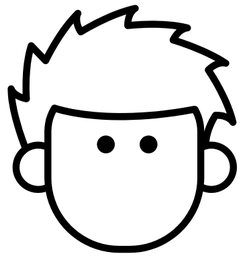
The first period is a delimiter and is removed when **&lib** resolves. The second period remains as text.



Updating Macro Variables

```
%let type=Truck;
%let hp=250;
title "&type.s with Horsepower > &hp";
footnote "Report Created on &sysday, &sysdate";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and
           Horsepower>&hp;
run;
```

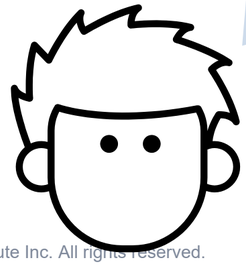
What must be modified in the program to generate a list of SUVs with horsepower greater than 300, and then print the date in the footnote?



Updating Macro Variables

```
%let type=SUV;
%let hp=300;
title "&type.s with Horsepower > &hp";
footnote "Report Created on &sysday, &sysdate";
proc print data=sashelp.cars;
     var Make Model MSRP Horsepower;
     where Type="&type" and
           Horsepower>&hp;
run;
```

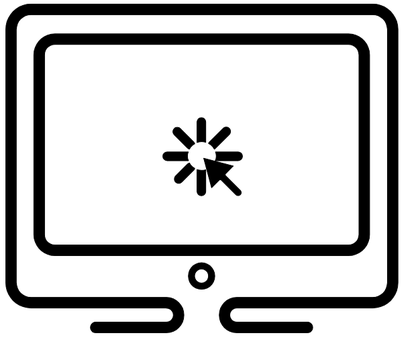
Simply update the %LET statements!



SUVs with Horsepower > 300

Obs	Make	Model	MSRP	Horsepower
28	BMW	X5 4.4i	$52,195	325
57	Cadillac	SRX V8	$46,995	320
119	Ford	Excursion 6.8 XLT	$41,475	310
144	GMC	Yukon XL 2500 SLT	$46,265	325
167	Hummer	H2	$49,995	316
231	Lincoln	Aviator Ultimate	$42,915	302
300	Nissan	Pathfinder Armada SE	$33,840	305
331	Porsche	Cayenne S	$56,665	340
378	Toyota	Land Cruiser	$54,765	325

Report Created on Friday, 01NOV19



Creating and using Macro Variables

This demonstration illustrates creating and using macro variables

Creating Macro Variables with PROC SQL

Creating Macro Variables

%LET statement

PROC SQL

PROC SQL can create and assign macro variables based on your data.

SELECT Statement: Syntax Order Mnemonic

SO FEW WORKERS GO HOME ON TIME	**SELECT** *object-item <, ...object-item>* **FROM** *from-list* <**WHERE** *sql-expression*> <**GROUP BY** *object-item <, … object-item >>* <**HAVING** *sql-expression*> <**ORDER BY** *order-by-item <DESC>* *<, …order-by-item>>*;

- The WHERE clause specifies data that meets certain conditions.
- The GROUP BY clause groups data for processing.
- The HAVING clause specifies groups that meet certain conditions.
- The ORDER BY clause specifies an order for the data.

PROC SQL Query (Review)

```
proc sql;
select Model, MPG_Highway
     from sashelp.cars
     where MPG_Highway>50
     order by MPG_Highway;
quit;
```

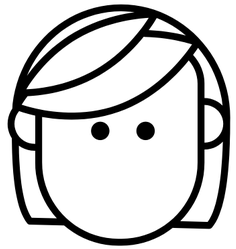
Model	MPG (Highway)
Prius 4dr (gas/electric)	51
Civic Hybrid 4dr manual (gas/electric)	51
Insight 2dr (gas/electric)	66

Creating Macro Variables with PROC SQL

```
PROC SQL;
SELECT <DISTINCT> item-1 < , item-2, …>
    <INTO : macvar-1 < …, : macvar-n>
    FROM clause
         <WHERE clause>
         <ORDER BY clause>;
QUIT;
```

The INTO clause assigns values produced by the query to macro variables.

Be sure to precede each macro variable name with a colon.



Creating Macro Variables with PROC SQL

Syntax 1– Storing Value of First Row in Declared Macro Variables

```
proc sql noprint;
select make, msrp into :expmake, :maxmsrp
from sashelp.cars
order by msrp desc
;
%put &=expmake;
%put &=maxmsrp;
```

Store the first row of the query into 2 macro variables & then request the variable values.

Global Symbol Table

Name	Value
MAKE	Porsche
MAXMSRP	$192,465

MAKE & MAXMSRP are created and stores the make &MSRP of car with highest MSRP

Make	MSRP
Porsche	$192,465
Mercedes-Benz	$128,420
Mercedes-Benz	$126,670
Mercedes-Benz	$121,770
Mercedes-Benz	$94,820
Mercedes-Benz	$90,520
Acura	$89,765
Jaguar	$86,995
Mercedes-Benz	$86,970
Audi	$84,600
Porsche	$84,165
Jaguar	$81,995
Dodge	$81,795
Porsche	$79,165
Mercedes-Benz	$76,870
Porsche	$76,765
Cadillac	$76,200
Volkswagen	$75,000
Jaguar	$74,995
Jaguar	$74,995
Mercedes-Benz	$74,320
BMW	$73,195
Land Rover	$72,250

Creating Macro Variables with PROC SQL

Syntax 1 – Storing Values from Multiple Rows in Declared Macro Variables

```
proc sql noprint;
select distinct Origin
     into :origin1-:origin3
     from sashelp.cars;
quit;
```

suppresses the report

creates a series of macro variables for the three distinct values of **Origin**

Global Symbol Table

Name	Value
ORIGIN1	Asia
ORIGIN2	Europe
ORIGIN3	USA

Creating Macro Variables with PROC SQL

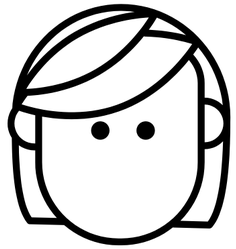
Syntax 2 - Storing Values from Multiple Rows in a List of Macro Variables

```
proc sql noprint;
select distinct Type
     into :type1-
     from sashelp.cars;
quit;
```

Global Symbol Table

Name	Value
TYPE1	Hybrid
TYPE2	SUV
...	
TYPE6	Wagon

If you don't know how many macro variables to create, you can omit the upper bound.



Creating Macro Variables with PROC SQL

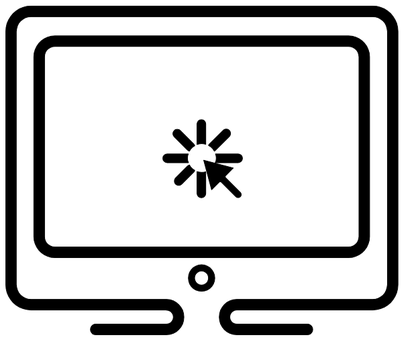
Syntax 3 - Storing Values of All Rows in One Macro Variable

```
proc sql noprint;
select distinct Origin
     into :originlist separated by ", "
     from sashelp.cars;
quit;
```

Use SEPARATED BY to assign multiple values to a single macro variable.

Global Symbol Table

Name	Value
ORIGINLIST	Asia, Europe, USA
SQLOBS	3



Creating Macro Variables with a PROC SQL Query

This demonstration illustrates creating and populating macro variables using a PROC SQL query with the INTO clause.

Handy Links

- PROC SQL INTO Clause
- SAS®Macro Language: Reference
- The Power of SAS SQL – SAS YouTube Video
- Macro Variables Defined by the Macro Processor
- SAS Tutorial | Step-by-Step PROC SQL – SAS YouTube Video
- "Shankar, Charu. "Know Thy Data: Techniques for Data Exploration". Pharmasug 2018,
- Ask the Expert Webinar - Why choose between SAS data Step & PROC SQL When You Can Have Both

Thank You

Charu Shankar
SAS Institute Toronto

EMAIL Charu.shankar@sas.com
BLOG https://blogs.sas.com/content/author/charushankar/
TWITTER CharuYogaCan
LINKEDIN https://www.linkedin.com/in/charushankar/

✓ Did you enjoy this session, Let us know in the **evaluation**