The Shape of SAS® Code
Hands on Workshop

Western Users of SAS Software WUSS
September 2018

Charu Shankar
SAS Institute Canada

The Shape of SAS® Code Course Notes was developed by Charu Shankar.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.

The Shape of SAS® Code Hands-On Workshop

Copyright © 2018 SAS Institute Inc. Cary, NC, USA. All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

Book code E71348, course code UGSSC, prepared date 20Aug2018. UGSSC_001

Table of Contents

Chapter 1 The Shape of SAS® Code 1-1

1.1 SAS DATA Step – The Manipulator 1-4

1.2 PERL – Finding Your Pattern 1-10

1.3 SAS Macro Language – Reduce, Reuse, Recycle 1-14

1.4 PROC SQL – Speak the Elegant Boolean 1-19

To learn more…

For information about other courses in the curriculum, contact the SAS Education Division at 1-800-333-7660, or send e-mail to training@sas.com. You can also find this information on the web at http://support.sas.com/training/ as well as in the Training Course Catalog.

For a list of other SAS books that relate to the topics covered in this course notes, USA customers can contact the SAS Publishing Department at 1-800-727-3228 or send e-mail to sasbook@sas.com. Customers outside the USA, please contact your local SAS office.

Also, see the SAS Bookstore on the web at http://support.sas.com/publishing/ for a complete list of books and a convenient order form.

Chapter 1 The Shape of SAS® Code

1.1 SAS DATA Step – The Manipulator 1-4

1.2 PERL – Finding Your Pattern 1-10

1.3 SAS Macro Language – Reduce, Reuse, Recycle 1-14

1.4 PROC SQL – Speak the Elegant Boolean.....1-19

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Situation

Crime is down overall in Henrico County, but the two districts in the county's east end continue to experience the highest amount of violent crime by a large margin, according to new crime statistics released by the Henrico Police Department.

Eastern districts Varina and Fairfield hold just 40% of the county's population, yet account for 66% of violent crimes in the county. Those are numbers the county's police chief is working to change.

Crime numbers actually increased in Varina in 2013 compared to 2012.

Burglary, motor vehicle thefts, aggravated assault, and rape offenses are all up from 2012.

3

What Shape of SAS Will Help You Investigate Data?

4

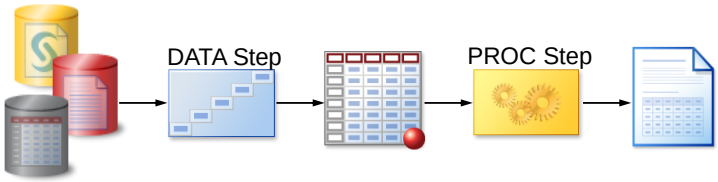
Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

1.1 SAS DATA Step – The Manipulator

SAS Programs

SAS Programs

A *SAS program* is a sequence of one or more steps.



- *DATA steps* typically build and manipulate SAS data sets.
- *PROC steps* typically process SAS data sets to generate reports and graphs, and to manage data.

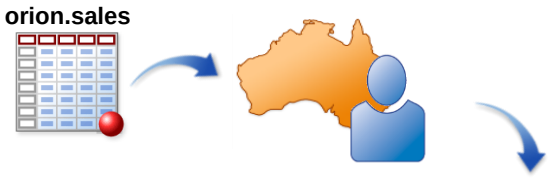
6

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Subsetting Data

Business Scenario

Orion Star management wants a report that lists only the Australian sales representatives.



Last_Name	First_ Name	Country	Job_Title
xxxxxxxxxx	xxxxxxx	xx	xxxxxxxxxxx
xxxxxxxxxx	xxxxxxx	xx	xxxxxxxxxxx
xxxxxxxxxx	xxxxxxx	xx	xxxxxxxxxxx
xxxxxxxxxx	xxxxxxx	xx	xxxxxxxxxxx

7

Subsetting In the Data Step

Include a WHERE statement to subset by **Country** and **Job_Title**.

```
Data AUReps;
Set wuss.sales;
where Country='AU' and
      Job_Title contains 'Rep';
run;
```

CONTAINS is a special WHERE operator.

8 p104d04

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

WHERE Statement

The WHERE expression defines the condition (or conditions) for selecting observations.

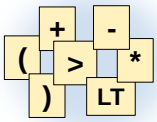
WHERE *WHERE-expression*;

Operands

- character constants
- numeric constants
- date constants
- character variables
- numeric variables

Operators

- symbols that represent a comparison, calculation, or logical operation



- SAS functions
- special WHERE operators

9

CONTAINS Operator

The *CONTAINS operator* selects observations that include the specified substring.

Equivalent Statements
`where Job_Title contains 'Rep';`
`where Job_Title ? 'Rep';`

- ? can be used instead of the mnemonic.
- The position of the substring within the variable's values is not important.
- Comparisons made with the CONTAINS operator are case sensitive.

10

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Viewing the Output

Partial PROC PRINT Output

Last_Name	First_ Name	Country	Job_Title
Elvish	Irenie	AU	Sales Rep. II
Ngan	Christina	AU	Sales Rep. II
Hotstone	Kimiko	AU	Sales Rep. I
Daymond	Lucian	AU	Sales Rep. I
Hofmeister	Fong	AU	Sales Rep. IV

11

Phone Tip

An anonymous caller left a phone tip saying they heard one suspect call the other one Alfred.

But the police officer says it was a heavily accented voice, so they are not sure that they got the name right.

12

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Looking For Alfred ??

The sounds-like operator =* is very useful when fuzzy matching of character values is needed.

It matches character strings based on their phonetic values.

The sounds-like operator is based on the SOUNDEX algorithm for identifying words that sound alike. The SOUNDEX algorithm is English-biased and is less useful for languages other than English. For more information, see *SAS® Functions and CALL Routines: Reference*.

13

Let's Give a Much Misspelled City a Try With the Sounds-Like Operator

```
data soundslike;
length studentId $4 Name $7 city $20;
input studentid $ name $ city $;
datalines;
1111 aaaa mississauga
2222 bbbb misisuga
3333 cccc misisag
4444 dddd misssisaga
5555 eeee missisuga
6666 ffff misisssaga
7777 gggg toronto
;
run;
data mississauga;
set soundslike;
where city=*'Mississauga';
run;
```

MISSISSAUGA

Filter and Sort | Query Builder | Where | Dat

	studentId	Name	city
1	1111	aaaa	mississauga
2	2222	bbbb	misisuga
3	3333	cccc	misisag
4	4444	dddd	misssisaga
5	5555	eeee	missisuga
6	6666	ffff	misisssaga

14

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

How Does the SOUNDEX Algorithm Work?

The steps used by SOUNDEX to derive the phonetic equivalent of a character string are as follows:

a) Retain the first letter of the character string.

b) Discard the letters A E H I O U W Y.

c) Assign a numeric value to the following consonants:

1.B F P V
2.C G J K Q S Z
3.D T
4.L
5.M N
6.R

d) Discard all duplicate classification values if they are adjacent. That is, DT results in a single value of 3, and NN results in a single value of 5).

15

Does the SOUNDEX Work in Every Possible Situation?

Why does our search for *alfred* return *albert*?

```
%let sonic=alfred;
data &sonic;
set wuss.criminal;
where name_age_of_person_charged =*
"&sonic";
run;

proc print data=&sonic;
var name_age_of_person_charged;
run;
```

16

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Use the Contains Operator Instead

SOUNDEX looks for the first word in your string and matches that with what you asked for.

```
%let sonic=Alfred;
data &sonic;
set wuss.criminal;
where name_age_of_person_charged
contains "&sonic";
run;

proc print data=&sonic;
var name_age_of_person_charged;
run;
```

17

1.2 PERL – Finding Your Pattern

Henrico County Cops Are Not Impressed. They Don't Want to Look Just at Specific Text, Rather at Patterns

20

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

That's When I Remembered a Customer Data Challenge

"Help! How can I find data that matches a specific pattern? My **HS10**_ column has a series of any 10- or 6-digit numbers. An additional challenge is that this series never appears in the same position."

- Sticks or profile shapes of subheading 3916.10
- Reproduction proofs for the production of printing plates, rolls, of tariff item No. 8442.50.20
- Microcopies of tariff item No. 4903.00.10, 4905.91.00, 4911.10.10 or 4911.10.20

21

PERL Code

```
DATA jan2011;
set wuss.perldata;
where prxmatch(("/\d{4}\./"),HS10_TSCHED_EDESC) > 0;
run;
```

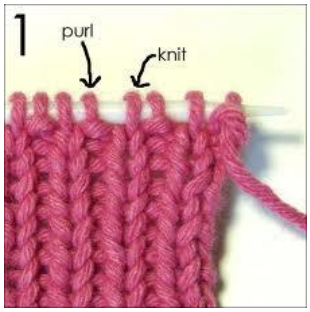
```
NOTE: There were 58 observations read from the data set WUSS.PERLDATA.
      WHERE PRXMATCH('/\d{4}\./', HS10_TSCHED_EDESC)>0;
NOTE: The data set WORK.JAN2011 has 58 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           10.93 seconds
      cpu time            0.14 seconds
```

/ I used forward slashes as default Perl delimiters
\d matches a digit 0 to 9
{n} matches the previous expression n times
\d{4} matches any 4 digits
\. is the pattern to match a period

22

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Language 2 PERL Natural Language Processing



23

A Closer Look

We are now interested in all statutes that have ( ) and can have any type of text values up to three characters. This is leading us closer to the crime!!

	Arrest# - Seq#	ICR #	Specific Crime	Statute Description	Arrest Date/Time
30	14022706402 - C02	140227064	619 - Larceny-Over 200-Other	18.2-95(i) - Grand Larceny: >=$200 Not From A Person	17MAY14:16:32:00
31	14022706402 - C03	140227064	1011 - Forgery-Check	18.2-172 - Other Forgery Writing:Not In 18.2-168 & 18.2-170	17MAY14:16:32:00
32	14022706402 - C04	140227064	1012 - Forgery-Check-Uttering	18.2-172 - Other Forgery Writing:Not In 18.2-168 & 18.2-170	17MAY14:16:32:00
33	14022706402 - C05	140227064	1012 - Forgery-Check-Uttering	18.2-172 - Other Forgery Writing:Not In 18.2-168 & 18.2-170	17MAY14:16:32:00
34	14022706402 - C06	140227064	1011 - Forgery-Check	18.2-172 - Other Forgery Writing:Not In 18.2-168 & 18.2-170	17MAY14:16:32:00
35	14030131701 - C01		1821 - Drug Viol-Marijuana-Possess	18.2-250.1 - Drugs: Possess Marijuana, 1St Off	13MAY14:11:02:00
36	14031133901 - C01		2607 - Other Offenses-Contempt Of Court	16.1-278.16 - Order: Not Comply W/ J&Dr Court Order	21MAY14:19:24:00
37	14031929101 - C01		1850 - Drug Viol - Poss Of Drug Paraphernalia	54.1-3466 - Possess Or Distribute Controlled Paraphernalia	03JUN14:10:02:00
38	14032322201 - C01		1821 - Drug Viol-Marijuana-Possess	18.2-250.1 - Drugs: Possess Marijuana, 1St Off	29MAY14:10:46:00
39	14040205601 - C01	140402056	1200 - Embezzlement	18.2-111 - Embezzlement: >=$200	13MAY14:20:28:00
40	14040323101 - C01		2618 - Other Offenses-Probation Violation	19.2-306 - Good Behavior: Violation On Felony Offense	23MAY14:01:24:00
41	14040416901 - C01	140404169	1152 - Fraud-Credit Card	18.2-192(1)(a) - Credit Card Larceny: Take/Obtain No.	14MAY14:17:32:00
42	14040416901 - C02	140404169	1152 - Fraud-Credit Card	18.2-195(1) - Credit Card Fraud: By Person, <$200 In 6M	14MAY14:17:32:00
43	14040416901 - C03	140404169	1152 - Fraud-Credit Card	18.2-195(1) - Credit Card Fraud: By Person, <$200 In 6M	14MAY14:17:32:00
44	14040416901 - C04	140404169	1152 - Fraud-Credit Card	18.2-195(1) - Credit Card Fraud: By Person, <$200 In 6M	14MAY14:17:32:00
45	14041111501 - C01	140411115	450 - Assault-Non-Aggravated	18.2-57.2(A) - Assault & Battery - Family Member	14MAY14:15:57:00
46	14042013401 - C01	140420134	1821 - Drug Viol-Marijuana-Possess	18.2-250.1 - Drugs: Possess Marijuana, 1St Off	21MAY14:23:44:00

24

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Looking at Crime Data Patterns

```
data wuss.perl;
set wuss.criminal;
where prxmatch(("/\(\D\D\)/"), statute_description) > 0 ;
run;
```

in a regular expression, how do you search a string for a \ character or a left or right parenthesis? Do this by preceding any of these special characters with a backslash (called an *escape character* in Perl).To match a \ in a string, you code two back slashes like this: \\. To match an open parenthesis, you use \(.

/ delimiters

\(matches an open parenthesis

\D matches a non-digit

{n,m} matches the previous subexpression n or more times, but no more than m

\) matches a closed parenthesis

25

PERL Advantage – Density

Perl is a very high-level language.

That means that the code is quite dense

A Perl program might be around 30% to 70% as long as the corresponding program in C.

26

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

PERL In SAS

Perl regular expressions were added to SAS in Version 9. SAS regular expressions (similar to Perl regular expressions but using a different syntax to indicate text patterns) have actually been around since version 6.12, but many SAS users are unfamiliar with either SAS or Perl regular expressions.

Because SAS already has such a powerful set of string functions, you might wonder why you need regular expressions.

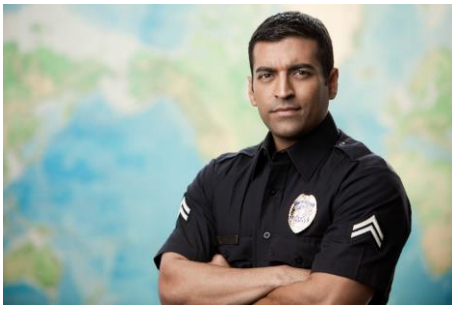
Many of the string processing tasks can be performed either with the traditional character functions or regular expressions. However, regular expressions can sometimes provide a much more compact solution to a complicated string manipulation task.

27

1.3 SAS Macro Language – Reduce, Reuse, Recycle

Now the County Sheriff Is Sitting Up. He's Beginning To See the Light Of Day.

He says, "We want the city in Henrico county with the highest count of top crimes. But we have a memory problem. Can't remember stuff, too much going on."



"Is there a way to store these names so we can reuse over and over again. We want to feed that data to another report."

29

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

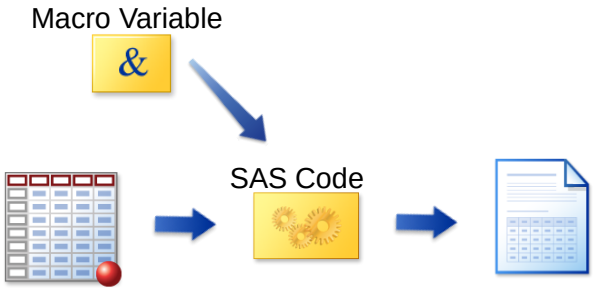
Macro Language

- We want the city in Henrico county with the highest count of top crimes.
- Let's store it in a macro so that we can reuse it in another report
- Reduce, reuse, recycle of course

30

Creating and Referencing Macro Variables

Create macro variables and then submit code to create a report using the defined macro variables.

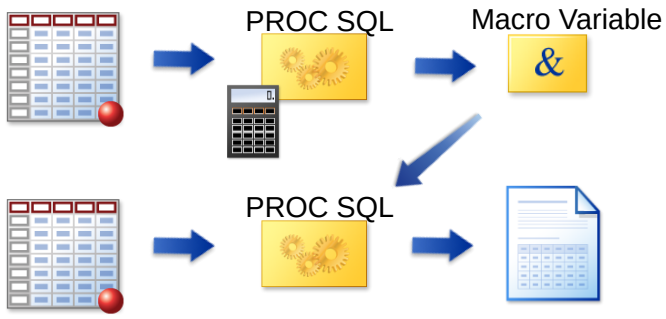


31

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

SQL Procedure and Macro Variables

In PROC SQL, you can use *macro variables* to store values returned by a query. You can then reference macro variables in other PROC SQL statements and steps.



32

PROC SQL and Macro Variables

The ANSI specification requires SQL to provide a mechanism for passing data values returned by a query to the host system. In PROC SQL, the host (SAS) receives data from a query as macro variable values.

PROC SQL creates or updates macro variables using an INTO clause.

The INTO clause has three syntaxes, and each produces a different result.

33

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Creating Macro Variables with PROC SQL

In PROC SQL, you use an INTO clause to create macro variables and assign a value to them or to update existing macro variable values. The INTO clause must follow the SELECT clause.

```
SELECT …
      INTO …
      FROM table|view …
     <additional clauses>
```

The INTO clause has three syntaxes, and each produces a different result.

34

Creating Macro Variables: Syntax 1

Syntax 1 places values from the ***first row*** returned by an SQL query into one or more macro variables. Data from additional rows returned by the query is ignored.

```
SELECT column-1 format=format-name.
<, …column-n>
      INTO :macvar_1<, ... :macvar_n>
      FROM table|view …
```

The value from the first column in the SELECT clause is placed in the first macro variable listed in the INTO clause, and so on.

35

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

PROC SQL and Macro Variables

Example: Create a single macro variable containing the average salary for the entire company, and use the INTO clause.

```
proc sql noprint;
   select avg(Salary)
      into :MeanSalary
       from wuss.employee_payroll;
 %put The average salary is &MeanSalary;
```

Macro variable names are preceded by a colon (:).

Partial SAS Log

```
The average salary is 38041.51
```

36 s108d06

PROC SQL into a Macro to Get the City with the Highest Crime Count

```
proc sql noprint;
select city ,count(city)  into : city, : citycoun
from wuss.string
group by city
order by 2 desc
;
%put &city= &citycoun=;
```

```
32         %put city=&city   ;
city= Henrico
33         %put citycount=&citycoun;
citycount=     553
```

37

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

1.4 PROC SQL – Speak the Elegant Boolean

Overview of the SQL Procedure

PROC SQL

"Give me powerful enough SQL and I can analyze the world." – Archimedes, circa 270 BCE (or maybe not)

39

SELECT Statement: Optional Clauses

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.

40

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

PROC SQL and the Boolean

PROC SQL and the Boolean Is…

The efficient way, the fast way, the only way

clean and elegant way

to

getting the haves and the have nots

41

Counting Rows That Meet a Specified Criterion

How do you determine the rows where crimes have been committed by 18- to 21-year-olds versus 22 to 35.

Question : Can you use the WHERE clause? ***No!***

42

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

SQL and the Boolean Code

```
options ls=150;
title "crime data by age group for &city";
proc sql;
select  Specific_Crime, sum(nage < 18) 'Minor Age <18',
sum(nage between 18 and 21) 'Age 18-21',
sum(nage between 22 and 35) 'Age 22-35',
sum(nage between 36 and 50) 'Age 36-50',
sum(nage > 51) 'Age 51 and older'
from wuss.string
where city="&city"
group by 1
;
quit;
```

43

Crime by Age Group for the City with Highest Crime

crime data by age group for Henrico

Specific Crime	Minor Age <18	Age 18-21	Age 22-35	Age 36-50	Age 51 and older
0 - Unknown	0	0	1	0	0
1011 - Forgery-Check	0	0	2	2	0
1012 - Forgery-Check-Uttering	0	0	2	2	0
1021 - Counterfeiting-Bank Note,Etc	0	0	1	0	0
1090 - Conspiracy To Commit Forgery	0	0	1	0	0
1140 - Fraud-Money-False Pretense	0	0	1	0	0
1151 - Fraud-Identity Theft	0	0	1	0	0
1152 - Fraud-Credit Card	0	9	1	4	0
1191 - Conspiracy To Commit Fraud	0	0	0	1	0
1192 - File False Report	0	1	0	0	0
1200 - Embezzlement	0	1	1	0	0
1310 - Stolen Property-Possession	0	2	7	0	0
1320 - Stolen Property-Receiving	0	0	0	0	1

44

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.

Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.