

Letter From the President



Dear SAS User:

SUGI was held in April in Philadelphia and several of us attended and presented, as we always do. SUGI of course is a great place to find out the latest and greatest, as well as meet SAS developers and support people from SAS. SUGI is always in a nice place and a great conference.

Being in Philadelphia, we had to try one of the city's famous cheesesteaks. We learned that the proper way to order was never to say "Philly Cheesesteak", but rather say something like, "Cheesesteak with or without (onions), wiz or no wiz (Cheese whiz)". Of course they were great however they were ordered.

This got us talking about all the great food we have enjoyed at SUGI over the years and then some of the great (or not so great) places we've eaten while teaching or doing a SAS project. The net result is the quiz below showing some of our favorite restaurants in the country. Some of them are truly treasures, and all are unique in some way according to our staff.

Match the 16 restaurants and cities with the correct corresponding hint or specialty.

For example 1. Pizzeria Due = P. Best deep dish pizza.

Restaurant Name	City, State	Hint or specialty
1 Pizzeria Due	Chicago, IL	A Doesn't matter if you order mild, medium, hot, you get what he feels like
2 Winterborne	Portland, OR	B Third largest grossing restaurant in US
3 Chinn's Crab House	Wheeling, IL	C Line stretches to sidewalk waiting for one of 10 stools
4 John Hardies Barbeque	Rochester, MN	D Fabulous Seafood, 6 tables
5 Coney Island	La Crosse, WI	E Students order 6 with onions
6 Galitoire's	New Orleans, LA	F Southern Greasy Spoon, conserves beer money
7 Figlio's	Minneapolis, MN	G No reservations, even President Clinton waited in line
8 Lutece	New York, NY	H Famous pies and meatball dinners
9 Den Chili	Springfield, IL	I Northern greasy spoon
10 Mickey's Diner	St. Paul, MN	J Hopple Popple goes with Beer
11 Al's Breakfast	Dinkytown (MPLS), MN	K Finish the Hot, write your name on the wall
12 Norske Nook	Osseo, WI	L Head uptown for nouveau Italian
13 Nick's No Name	Boston, MA	M Beef on Weck
14 Benji's	Milwaukee, WI	N Mismatched tableware and fabulous cheap lobster
15 Dunk and Dine	Atlanta, GA	O Middle of city townhouse
16 Every Restaurant in Town	Buffalo, NY	P Best deep dish pizza

We will be sending a flash drive along with a small food prize to the first 3 correct entries. We will publish the answers and winners names in our next newsletter. Please email your entries to jfirst@sys-seminar.com. Even if you don't enter, you may want to check out these places if they are near you.

As always thanks for your support and good luck!

Steven First,
President



SYSTEMS SEMINAR CONSULTANTS, INC.

Copyright © 2005 Systems Seminar Consultants, Inc. Madison, WI. All rights reserved. Printed in USA.
The Missing Semicolon is a trademark of Systems Seminar Consultants, Inc. SAS, SAS/IntrNet, and SAS/ACCESS are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.

Thou Shall

CONTINUED FROM PAGE 1

4) Use ATTRIB or LENGTH statements.

Explicitly defining variable lengths to the smallest possible value has the potential for saving large amounts of space. Numeric variables default to a length of 8 bytes. SAS dates are stored as numerics but can always be stored in as few as 4 bytes, and for dates close to January 1, 1960 in as few as 2 bytes. In a data set with 1000 observations, changing the length of a single date variable to 4 bytes will save 4,000 bytes of space.

5) Minimize numeric type variables.

Reading and writing numeric types to and from flat files is one of the slowest operations in SAS. When reading the value \$19,862.34 from a flat file using the INFORMAT COMMA10., SAS must first strip out the dollar sign and comma. Once the raw number is available it then must translate this value into the 8 byte floating point number used within the SAS data set for storing the value. Numeric type variables may be minimized by:

- reading only the numeric fields necessary for your processing
- reading the value as character if the variable is not required to be used in computations.

6) Use DROP or KEEP statement.

Use the DROP or KEEP statements and data set options as early as possible. DROP and KEEP limit the variables included on the data set. By limiting variables as early in the processing as possible the movement of data is minimized. In addition, the amount of storage space required to hold the values is reduced.

7) Read only required fields.

Reading variables which are not required uses I/O resources as well as work space when creating a new dataset. Reading only required fields minimizes the usage of I/O resources and reduces later processing to remove the unnecessary fields. Reading only required fields may be accomplished by:

- removing code for non-required fields
- commenting out non-required fields
- use of the trailing @

8) Eliminate Extra Passes of Data.

In a single pass of data, you should execute as much logic as you can against the data.

Example: A typical program is written in the “one thought per step” style:

```
data sales;
  infile rawin;
  input @01 name $10.
        @12 div $1.
        @19 sales 7.
        @27 expense 7.2;
run;
data sales2;
  set sales;
```

For More On SAS Efficiencies...

Sign up for our SAS Efficiencies course!

This course will help you learn advanced methods and save system resources.

Call 1-800-997-7081 to schedule onsite training today, or

Sign up for our next public class on September 16 at <http://www.sys-seminar.com/training.php>.

Visit www.sys-seminar.com/pdfs/class_efficiencies.pdf for a full course description.

```
net = sales - expense;
run;
```

This program requires the sales data to be read twice, one time to create the data set sales and a second time to calculate the variable net. A more efficient program would be to calculate the variable net within the same step in which the data is read in:

```
data sales;
  infile rawin;
  input @01 name $10.
        @12 div $1.
        @19 sales 7.
        @27 expense 7.2;
  net = sales - expense;
run;
```

9) Use RETAIN statement.

When a constant value is set using the assignment statement the statement will be executed every iteration of the data step and I/O resources will be used to write out the same value to the PDV each time.

```
data midwest;
  set salesdata;
  desc = 'MIDWEST';
  ...
run;
```

Constant values may be set using the RETAIN statement. When the RETAIN statement is used, the initial value will be set at compile time. Because a flag is also set to hold the value through each iteration of the step, it is not necessary to reset the value unless it is necessary to change it. The RETAIN statement is a compile time statement, so no action is taken during the execution phase.

```
data midwest;
  set salesdata;
  retain desc 'MIDWEST';
  ...
run;
```

Continued on next page.....

10) Avoid unnecessary SORT.

While sorting of data is very common the SORT procedure requires a lot of computer resources:

- a) SORT almost always requires three or more passes of the data
- b) SORT may momentarily require two copies of the data
- c) Space is required for the SORT work area
- d) SORT is I/O, CPU, and clock time intensive

While it may not be possible to completely eliminate sorting of the data, it is possible to minimize the number of SORT procedures required or minimize the resources required to complete the sort. SORT procedures can be eliminated or minimized with the following steps:

- a) When multiple steps require data to be in the same sort order, group the steps together and sort the data one time prior to the group of steps.
- b) Use the CLASS statement whenever possible
- c) Limit observations
- d) Limit variables

These are only a few possibilities for more efficient SAS code. Watch for more efficiencies in future editions of The Missing Semicolon.



This article is a continuation from our Winter 2005 issue. For the first half, visit www.sys-seminar.com/pdfs/tms2005winter.pdf.

REPLACING MISSING VALUES

PROC SQL can replace a missing value with another value by using the SQL COALESCE function. COALESCE can return a default value if the field is missing, or choose the order of looking for a value from tables that have same name variables.

The COALESCE function scans the list of values from left to right and returns the first non-null value. COALESCE is often used in SQL joins to select the first non-missing value when a variable is on more than one table. The COALESCE function has been available in PROC SQL, but starting with SAS 9 two functions, COALESCE for numerics, and COALESCEC character values will now be available in the DATA step. This example will return the hours1, hours2, hours3 values, and if any are missing, a zero will be substituted.

```
proc sql;                                /* select payments */
  create table test1 as                  /* create test1 dsn */
  select name,                            /* get customerid */
         bill_amt,                        /* bill_amt $ */
         coalesce(hours1,0)              /* get hours1 */
         as hours1,                       /* never missing */
         coalesce(hours2,0)              /* get hours2 */
         as hours2,                       /* never missing */
         coalesce(hours3,0)              /* get hours2 */
         as hours3                        /* never missing */
  from hourrate;                          /* from hourrate */
quit;                                     /* end sql step */
proc print data=test1;
  title 'Test SQL Coalesce ';
run;
```

Test Sql Coalesce					
Obs	NAME	BILL_AMT	HOURS1	HOURS2	HOURS3
1	ROBERT	1422	15	15	10
2	BEATRICE	771	10	10	20
3	RICHARD	2241	40	2	0
4	ELAINE	1987	20	20	1

CHANGING ZERO TO MISSING

Sometimes systems have an actual value like zero, that would be better if considered to be a missing value. Other languages don't often have a missing value and so a common thing to do is substitute zero when all else fails. There is of course a difference between zero which is a valid number, and a missing value which is an unknown. In such a case, invalid values might need to be changed to missing so they can be disregarded in calculations. We worked recently on a banking application that stored a balance as zero even though it truly should have been missing. In this case the zero balances were paid off loans, but in any case calculating the average loan balance for all loans skewed results to a lower amount. We have a few options available to us. First we could

Continued on next page.....



SYSTEMS SEMINAR CONSULTANTS, INC.
1-800-997-7081 ♦ www.sys-seminar.com

We're Hiring!

Fill out our online application:
<http://www.sys-seminar.com/onlineapp.php>.

Staff Placement

Call us to discuss your need for full-time and contract employees.

♦ **The Missing Semicolon**

Tips from the experts!
Sign up for our complimentary professional newsletter.

♦ **SAS Training**

Free follow up help desk
Customized courses available
Public and Onsite Training



Accessing Databases

Application Development

Data Cleaning

Data & System Validation

Data Conversion

Data Warehousing

Data Extraction

Efficiencies

Project Management

Process Automation

Analysis

Reporting

Is Something Missing? Part 2

CONTINUED FROM PAGE 4

change the zero balances to missing then calculate. Second we could subset the data to calculate the average on only accounts with a balance.

First we have a report with a skewed statistic:

```
proc means data=custbal mean min;
  title 'Invalid Averages';
  var balance;
run;
```

Invalid Averages	
The MEANS Procedure	
Analysis Variable : BALANCE	
Mean	Minimum
479.4670000	0

To change the invalid values on the dataset to missing:

```
data fixcust;
  set custbal;
  if balance=0 then balance=.;
run;
proc means data=fixcust mean min;
  title 'Fixed Averages';
  var balance;
run;
```

FIXED AVERAGES	
The MEANS Procedure	
Analysis Variable : BALANCE	
Mean	Minimum
532.7411111	21.0900000

QUICK TIP

Use the informat \$UPCASEw. to read in character data which may be in upper, lower, or mixed case and store the value in the SAS data set as an upper case value.

```
Input file:
William   h  WI

input @01 Name      $upcase10.
      @12 Division  $upcase1.
      @14 State     $upcase2.;
```

```
SAS data set:
Name      Division  State
WILLIAM   H           WI
```



Another way to handle the above would be to subset the dataset.

```
proc means data=custbal mean min;
  title 'Subset For Averages';
  var balance;
  where balance not in (0);
run;
```

Subset For Averages	
The MEANS Procedure	
Analysis Variable : BALANCE	
Mean	Minimum
532.7411111	21.0900000

Printing Substitutes For Missing Values

Sometimes missing values are acceptable, but we would like to display these values as something else. The SAS option MISSING defines a single character that will be printed for missing numeric values. To print a character other than the default period for missing, code an options statement. The MISSING option does not change the value that is stored, only the way it prints.

```
options missing=X;
proc print data=hourrate;
  title 'Missing Printed As Zero';
run;
```

Missing Printed As Zero					
Obs	NAME	BILL_AMT	HOURS1	HOURS2	HOURS3
1	ROBERT	1422	15	15	10
2	BEATRICE	771	10	10	20
3	RICHARD	2241	40	2	X
4	ELAINE	1987	20	20	1

Continued on next page.....

25 % Training Discounts!

Any onsite training scheduled for July 2005 will receive a **25% discount** off of the list price. Please call 1-800-997-7081 ext. 306 for details.

Is Something Missing? Part 2

CONTINUED FROM PAGE 5

Some procs, such as PROC TABULATE, can use an option such as MISSTEXT TABLE statement, to allow the printing of more than one character.

```
proc tabulate data=hourrate;
  class name;
  var hours1 hours2 hours3;
  table name, hours1 hours2 hours3/misstext='NONE';
run;
```

	HOURS1	HOURS2	HOURS3
	Sum	Sum	Sum
NAME			
BEATRICE	10.00	10.00	20.00
ELAINE	20.00	20.00	1.00
RICHARD	40.00	2.00	NONE
ROBERT	15.00	15.00	10.00

In summary, our data may have invalid values, or missing responses, but we have ways to deal with these situations. There are many ways to handle missing values. We have looked at a just few methods in SAS and PROC SQL. The question is, did we miss any?

SAS® Consulting Services

Export IT Consultants

- ◆ We specialize in reporting and data manipulation.
- ◆ Each member of our team has specialties of their own, which include mainframe applications, SQL and database programming, web development, graphs, and maps.

Extensive SAS® Experience

- ◆ We work with a variety of industries, such as healthcare, marketing, manufacturing, finance, insurance, academic, government, and telecommunications.
- ◆ Our specialty areas include Data Systems Development, Decision Support and Business Consultation, Report Design, Development, and Web-enablement.
- ◆ As a SAS Affiliate, we are Consulting Partners of the SAS® Institute.
- ◆ All members of our team are SAS certified professionals.

QUICK TIP

To change variable values, use the tranwrd function instead of an if, then statement.

```
For example, instead of:
if state = 'California'
  then state = 'CA';
```

```
Make the change with tranwrd:
state = tranwrd(state, "California", "CA");
```

Remember to use caution with this function so only the strings you intend to change are altered.

run:

TECHNICAL CREDIT AND RECOGNITION



Steve First
President
Letter from the President, Page 2

Teresa Schudrowitz
Trainer/Consultant
Thou Shall...Program Efficiencies, Page 1
Quick Tip 1, Page 5



Gerry Frey
Trainer/Consultant
Is Something Missing? Part Two, Page 4

Jennifer First
Office Manager
Quick Tip, Page 6



Editors.....Jennifer First, Katie Ronk, Susan Bakken, Ann Vinge

Onsite SAS Training Solutions

Systems Seminar Consultants Offers:

- ◆ Onsite Training **Nationwide**
- ◆ Quality Training from Experienced Consultants
- ◆ Competitive Prices
- ◆ Customized Materials and Courses
- ◆ Experience with a Variety of Industries
- ◆ Complimentary Follow-up Help Desk Support



New Course Announcements!

Tips, Tricks, and Techniques

An eclectic mix of SAS tips and techniques that our consultants have used in contract programming.

Advanced Macros

A course that teaches the advanced features of the SAS macro language.

Public SAS® Training Schedule 2005 Madison, Wisconsin

Introduction to SAS®

- ◆ September 12-14

The SAS® SQL Procedure

- ◆ September 15

SAS® Efficiencies

- ◆ September 16

SAS® Report Writing

- ◆ September 26-27

Introduction to Proc Report

- ◆ September 28

SAS® Macros

- ◆ September 29-30

Advanced SAS®

- ◆ October 3-5

What's New in SAS® 9

- ◆ October 6

Tips, Tricks, and Techniques

- ◆ October 7

Exploiting ODS

- ◆ December 8-9

Mainframe Made Easy

- ◆ To Be Scheduled

SAS/ACCESS® to Relational Databases

- ◆ To Be Scheduled

Syncsort®

- ◆ To Be Scheduled

To register for a class or for further information, please call: 1-800-997-7081 or visit www.sys-seminar.com

Can't make it to a scheduled public class? Can't get the class you need onsite?

Get it with **Public On Demand!** Call for details!

Call 1-800-997-7081 ext. 306 to discover your SAS training solution.