

Don't be a slave to your SAS programs

Marje Fecht
Senior Partner
Prowerk Consulting

Larry Stewart
Sr. Director, Education
SAS Institute

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

Copyright © 2007, Prowerk Consulting LLC. All rights reserved.

Overview

- Do you spend a lot of time updating your programs to adjust inclusion criteria?
- Do your programs run forever while you patiently wait?



This tutorial focuses on maintenance – free, efficient coding techniques so that you can spend your work time being more productive.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Topics to be Covered

- Introduction
- Remove Inefficient and “Wordy” Coding
- Reduce Programmer Intervention
- Testing Tips
- Questions & Answers

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Introduction

How do we waste our time?

- Changing dates and timeframes
- Changing titles and inclusion criteria to reflect the current run
- Writing multiple steps when only one step is needed
- Waiting for programs that take forever to run
- Writing multiple programs when only one is needed

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Introduction

- Strike a balance !
- Minimize your intervention with production jobs
- Minimize runtime
- Focus your programming time on reusable code (not one-time queries)

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Business Task

You need to generate a report that requires

- subsetting a SAS data set
- sorting the resulting data set.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Subset the data prior to sorting

```
data compare;
  set report;
  if ProductCode in ('XER', 'REF', 'CRS')
  and Date gt '01MAY2007'd;
  keep ProductCode Usage Date Location;
run;
proc sort data=compare;
  by ProductCode Date;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Remove Inefficient Coding

Problems

- All observations in the REPORT data set are read but only specific ProductCode and Date values are required
 - Use WHERE instead
- Data set is read and written at least twice
 - Use one step instead of two

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Remove Inefficient Coding

Subset and sort the data in one step

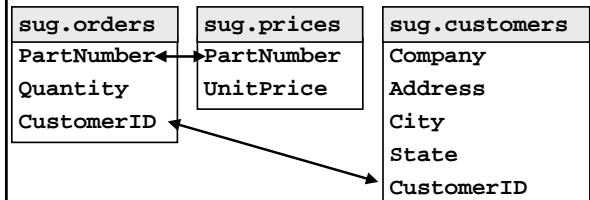
```
proc sort data=report
  (keep=ProductCode Usage Date Location)
  out=Compare;
  where ProductCode in ('XER','REF','CRS')
  and Date gt '01MAY2007'd ;
  by ProductCode Date;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Business Task

You need to

- merge 3 SAS data sets that do not have a common BY variable.



Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

```
proc sort data=sug.Orders out=Orders
  by PartNumber;
data step1;
  merge Orders(in=inOrders) sug.Pri
  by PartNumber;
  if inOrders;
proc sort data=step1;
  by CustomerID;
proc sort data=sug.Customers out=Customers;
  by CustomerID;
data step2;
  merge step1(in=inStep1) Customers;
  by CustomerID;
  if inStep1;
  InvoiceAmt=Quantity*UnitPrice;
proc sort data=step2;
  by Company;
proc print data=step2;
  var CustomerID Company Address City State
  PartNumber Quantity InvoiceAmt;
run;
```

Complicated Merge – no common BY value.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Step 1: Merge the ORDERS and PRICES data sets by PartNumber and create the STEP1 data set.

```
proc sort data=sug.orders out=orders;
  by PartNumber;
data step1;
  merge orders(in=inOrders) sug.prices;
  by PartNumber;
  if inOrders;
```

Complicated Merge – no common BY value.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Step 2: Merge the STEP1 data set with the CUSTOMERS data set by CustomerID.

Complicated Merge – no common BY value.

```
proc sort data=step1;
  by CustomerID;
proc sort data=sug.Customers out=Customers;
  by CustomerID;
data step2;
  merge step1(in=inStep1) Customers;
  by CustomerID;
  if inStep1;
  InvoiceAmt=Quantity*UnitPrice;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Step 3: Sort the final data set by Company and print the data set.

Complicated Merge – no common BY value.

```
proc sort data=step2;
  by Company;
proc print data=step2;
  var CustomerID Company Address City State
  PartNumber Quantity InvoiceAmt;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Complex Coding

Problems

- Multiple steps to read and combine data
- Multiple SORTS of the same data
- Use SQL instead for fewer steps and less complex coding

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Complex Coding

Solution: use SQL instead

```
proc sql;
  select a.CustomerID, Company, Address,
  State, b.PartNumber, Quantity,
  Quantity*UnitPrice as InvoiceAmt
  from sug.Customers as a,
  sug.Prices as b, sug.Orders as c
  where a.CustomerID=c.CustomerID and
  b.PartNumber=c.PartNumber
  order by Company ;
quit;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Business Task

You need to

- compute and compare revenue figures by day of the week.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Compute total revenue by Day of Week

```
data work.revenue;
  set sug.sales;
  DayOfWeek=weekday(date);
run;
proc format;
  value DayWk 1='Sunday'
  2='Monday'
  3='Tuesday'
  4='Wednesday'
  5='Thursday'
  6='Friday'
  7='Saturday';
run;
proc means data=work.revenue n sum;
  class DayOfWeek;
  var Revenue;
  format DayOfWeek DayWk.;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Step 1: Create a variable that contains the day of the week.

```
data work.revenue;
  set sug.sales;
  DayOfWeek=weekday(Date);
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Step 2: Create a format that substitutes the name of the day for the day number.

```
proc format;
  value DayWk 1='Sunday'
              2='Monday'
              3='Tuesday'
              4='Wednesday'
              5='Thursday'
              6='Friday'
              7='Saturday';
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

Step 3: Use the format when computing the totals.

```
proc means data=work.Revenue n sum;
  class DayOfWeek;
  var Revenue;
  format DayOfWeek DayWk.;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Inefficient Coding

Problems

- Multiple steps when one is enough
- Creating a format when one already exists
 - Use WEEKDATE9. format

```
Original Date      : 14MAY2007
SAS Date           : 17300
WEEKDATE Format    : Monday, May 14, 2007
WEEKDATE9 Format   : Monday
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Inefficient Coding

Solution: Use WEEKDATE9. format

```
proc means data=sug.sales n sum;
  class Date;
  var Revenue;
  format Date weekdate9.;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Business Task

You need to

- run a daily report that compares current month to date revenue with prior month revenue.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

```
data currentmonth;
  set sug.sales;
  if month(Date) = 5 and year(Date) = 2007;
  MonthYear = " 5/2007";
data lastmonth;
  set sug.sales;
  if month(Date) = 4 and year(Date) = 2007;
  MonthYear = " 4/2007";
data comparemonths;
  set lastmonth currentmonth;
run;
proc means data=comparemonths;
  class MonthYear;
  var Revenue;
  title "MTD Revenue vs Last Month";
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention and Remove Inefficient Coding

Problems

- Dataset is read multiple times when only one read is required
 - Use one step instead of two (or three)
- Monthly intervention required to change values
 - Use macro variables or SAS functions instead

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Remove Inefficient Coding

Solution 1: Reduce 3 DATA steps to just 1

```
data comparemonths;
  set sug.sales;
  if month(Date)=5 and year(Date)=2007 then do;
    MonthYear = " 5/2007";
    output;
  end;
  else if month(Date)=4 and year(Date)=2007 then do;
    MonthYear = " 4/2007";
    output;
  end;
run;
*** proc means step unchanged;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention

Solution 2: Reduce intervention

```
%let currmon = 5;          %let lastmon = 4;
%let curryear=2007;       %let lastyear=2007;
data comparemonths;
  set sug.sales;
  if month(Date)=&currmon and year(Date)=&curryear
  then do;
    MonthYear = " &currmon/&curryear";
    output;
  end;
  else if month(Date) = &lastmon and year(Date) =
  &lastyear then do;
    MonthYear = " &lastmon/&lastyear";
    output;
  end;
run;
*** proc means step unchanged;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention

Solution 3: Remove Intervention

```
data comparemonths;
  set sug.sales;
  if month(date) = month(today()) and
  year(date) = year(today()) then do;
    MonthYear = put(today() , mmyyS7.);
    output;
  end;
  else do;
    lastmonth=intnx('MONTH' , today() , -1);
    if month(date) = month(lastmonth) and
    year(date) = year(lastmonth) then do;
      MonthYear = put(lastmonth , mmyyS7.);
      output;
    end;
  end;
run; *** proc means step unchanged;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Business Task

You need to

- use the same query logic and reporting formats on a regular basis for each new product / campaign / region.

The program is always the same, except for changes to

- date ranges
- inclusion codes
- titles and footnotes
- search logic.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?

- You make a copy of the most recent version you can find of the query / reporting program
- You step through the program and make all the changes that apply for your current use
- You run the program only to find that you overtyped some quotes and semicolons
- You apply corrections, and try again

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention

Problems

- It is difficult to locate the most current version of the query/reporting program
 - Store a single dated copy of the main program
- Updating values in the program often leads to errors, or missed parameters
 - Use a driver program with macro variables

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention

Solution: Use a driver program to specify parameters and run standard program.

```
*** Driver Program - specify correct parameters;
%let prestart = 01OCT2006;
%let prestop = 31DEC2006;
%let poststart = 01JAN2007;
%let poststop = 30JUN2007;
%let title = "January 2007 Introduction of Low
Interest Rates";
%let products = ('VRS', 'GQL', 'BGO', 'DLA');
%let codes = ('015', '119', '214');

*** run standard reporting program;
%include '2007_01_10CampaignReporting.sas';
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention – Success!

The Old Way – Hours to locate / QA / etc

The New Way – Campaign reporting is ready with less than ½ hour of effort

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Business Task

You need to

- run a daily revenue report that includes a monthly recap on the 1st of each month

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Does this look familiar?



✓ You have a reminder on the 1st of each month to submit your monthly reporting programs

✓ You manually combine the reports from the daily and monthly output, and provide them to the viewers

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention

Problems

- you must remember to submit the monthly program, unless you use a batch scheduler
 - you are producing two separate reports, and the viewers may prefer a single report
- Use a single program with macro code to decide when to run the monthly logic and report

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Reduce Intervention

Solution: Reduce intervention

```
*** Run monthly report on first of each month;

%macro dayone;
  %if %sysfunc(day( %sysfunc(today()) )) = 1
    %then %do;
      %include 'monthly_report.sas';
    %end;
%mend;

%dayone

*** Run daily report every day;
%include 'daily_report.sas';
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Testing Tips

Remember: COMPARE the *before and after* data sets to confirm comparability of solutions!

```
*** Useful to compare 2 solutions;
proc compare data=sales compare=sales2;
run;
```

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Conclusion

As you learn more about SAS, you will find that there are numerous features that enable you to accomplish tasks easily and efficiently.

In this presentation, you have seen

- tips to reduce how many times data is processed
- code reduction techniques
- suggestions to reduce or remove manual intervention to enable "maintenance-free" jobs.

Copyright © 2006, Prowerk Consulting LLC. All rights reserved.

Thank You!

Marje Fecht

Senior Partner
Prowerk Consulting

marje.fecht@prowerk.com



Copyright © 2006, Prowerk Consulting LLC. All rights reserved.