

SAS Tips

- by Philip Mason
- E-mail: PMASON@VCRPMRKT.TELECOM.COM.AU

- New features you may have missed
- Some useful techniques
- Undocumented features
- The key is to go and try them out

Autoscroll 0

- Saves elapsed time when running online
- Can around to 80%
 - Options Autoscroll=0 ;
- Best when you produce lots of output

Catalog Stats

■ How big are my catalog members

- AF program, format, etc.

```
Options Linesize=132 ; * So it fits nicely ;  
Proc Catalog catalog=libref.catname ;  
  contents STAT ;
```

■ Gives member name, # blocks, # pages, etc.

Undocumented

SAS/Connecting mainframes

■ 3 types of Client Servers Services

- Compute Services (eg. RSUBMIT)
- Remote Library Services
eg. Libname libref REMOTE 'library-name' SERVER=rem-session-id ;
- Data Transfer services (eg. PROC UPLOAD/DOWNLOAD)

■ Connect between any machines running SAS

- PC to PC
- Mainframe to PC
- Mainframe to Mainframe

Basic Data Encryption

- Simple method of encrypting numbers
- Encode
 - `secret=BXOR(number,key);`
- Decode
 - `decoded=BXOR(secret,key);`
- Note:
 - truncates to nearest 1
 - so convert money to cents

P-222, p50 (6.07+)

Datastep Views

- Define once, use many
- View is 'run' each time accessed

```
Data Lib.Fred / view=Lib.Fred ;  
infile flat ;  
input @1 name $30. @40 address $30. ;
```

P-222 p149-155 (6.07+)

Data Step Debugger

- Very similar to SCL debugger
- Help and Pmenus are available
- Use to find logic errors in datasteps

```
Data temp / debug ;  
  set freddy ;  
  a=sqrt(_n_);
```

Undocumented

Multiple PROCs per page

- Useful to save printed space
 - Especially if lots of brief output
- Set it for no page breaks
 - Options formdlm=' ';
- Reset it using a null string
 - Options formdlm="";

Nested Formats

- Can refer to other formats within formats
- Can nest formats up to 5 deep

```
Proc format ;  
  value nest  
    1 = 'First'  
    2 = 'Second'  
    3 = 'Third'  
    other = (!comma6.!) ;
```

P-222, p213 (6.07+)

Mixed Numeric informat

- unquoted numerics treated as numbers
 - prior to 6.07 were treated as characters

```
Proc format ;  
  invalue mixed  
    1-10 = 1  
    11-20 = 2  
    'XYZ' = 9  
    other = 999 ;
```

- Note:
 - in 6.06 111 would have been in the 11-20 range

P-222 p210-216 (6.07+)

Modifying standard (in)formats

- Modify behaviour of supplied formats
- eg. Numeric data with some character values
 - using 8. will cause errors

- To handle error data

```
proc format ;  
  invalue myfmt  
    " = 0  
    . = 0  
    'A'-'Z' = 0  
    'a'-'z' = 0  
    other = (18.1) ;
```

P-222 p211 (6.07+)

Using Colon in variable lists

- Colon can be used as wildcard
- Works in most places

```
Proc print data=x ;  
  var name addr ;
```

```
Data keepsome ;  
  set lotsdata(keep=a: var: x z) ;
```

- Note:
 - Colon must be at end of name
 - can't be embedded

Undocumented

Resident SCL methods

- Make SCL methods resident in memory
- Similar to PROGRAM entry resident option

label:

```
method data_x 8  prodcode $ / RESIDENT ;  
  prodcode=getnitemc(data_x,'prodcode') ;  
endmethod ;
```

- Can reduce response time and I/Os

Balancing your quotes

- Sometimes we get the annoying message
CHARACTER STRING WAS MORE THAN 200 CHARACTERS
- Indicates you may have unbalanced quotes
- To close submit the harmless commands

```
*!.*;*/;
```
- If that fails and you are using macros then try

```
*);*/;/**/***/;%mend;
```
- If that fails then hit ATTENTION or BREAK
- If that fails then you have found a bug

Pseudo Random numbers

- UNIFORM function returns "random" numbers
- Numbers are based on a seed
- A seed ≤ 0 uses system time as the seed
- Same seed returns same numeric sequence

My test showed

- 2 datasteps with same seed generated identical random numbers (using Datasteps & PROC COMPARE)
- This seems to apply to other random functions
 - ✦ eg. RANUNI, RANTRI, RANPOI, etc.

SCL in Batch

- Useful for accessing functionality of SCL
 - Some SCL functions not available elsewhere in SAS
 - eg. CEXIST function
- Wont work if entry contains a display

Proc display catalog=libref.catalog.entry.type BATCH ;

Reading CSV data

■ CSV data best read using DSD INFILE option

- quotes are removed from quoted strings
- quotes can be kept by specifying ~ format modifier
- default delimiter set to comma
- can override default delimiter (DELIM= option)
- consecutive delimiters assign missing values

Example

```
data x ;  
  infile flatfile DSD missover ;  
  input name $10. address $20. phone $15. ;  
run ;
```

P-222 p30 (6.07+)

Call Execute

- Queues SAS code to be run next
- Similar to writing code to a file and then
 %INCLUDEing the code so it will run

Example

```
data _null_ ;  
  call execute("data y;set sasuser.crime;run;");  
run ;
```

Rearranging variables

- RETAIN statement can be used

```
Data details ;  
  Retain name address phone ;  
set details ;
```

- LENGTH statement can be used

```
data details ;  
  length name $ 20 address $ 30 phone $ 15 ;  
set details ;
```

- SQL can be used

```
Proc SQL ;  
  create table details as  
  select name, address, phone from details ;
```

Wildcard file reading

- Useful to allocate many similarly named and identically laid out files

- Works in most operating systems
but not MVS (yet)

- Example

```
filename in 'C:\tab*.dat' ;  
data report ;  
infile in ;  
input a b c ;
```

Multiple format libraries

- FMTSEARCH lets SAS search multiple format libraries
- Order specified is order searched
- SAS searches WORK.FORMATS and LIBRARY.FORMATS first
 - unless explicitly specified in FMTSEARCH option

options fmtsearch=(mylib yourlib.special) ;

- searches work.formats, library.formats, mylib.format, yourlib.special

options fmtsearch=(mylib library work) ;

- searches mylib.formats, library.formats, work.formats

- P-222 p91 (6.07+)

Testing SQL

- Check syntax using VALIDATE

```
proc sql ;  
  VALIDATE select * from lib.user ;
```

- Test run with limited number of observations using INOBS

- This example would run code with maximum of 2 obs from each input dataset

```
proc sql INOBS=2 ;  
  select * from lib.user, lib2.other ;
```

Long labels on FREQ

- PROC FREQ truncates variables
- Variables truncated to 16 characters
- This may give wrong results
 - Sample data
 - this is a very very very long string
 - this is a very very very long carrot
 - Frequency table would show
 - 2 of 'this is a very v'
- Full variable names can be shown
 - PROC SUMMARY/MEANS
 - PROC SQL

Saving Space (1 of 4)

- Delete temporary datasets using PROC DATASETS
- Limit variables with KEEP=, DROP=
- Use WHERE= or WHERE to reduce obs processed
- Use _NULL_ datasteps when you don't need to keep them
- Length to limit bytes for each variable
- Remote Library Services to only keep one copy of data on network
- Temp. datasets on ramdisk, VIO or Hiperspace
- Dataset compression (COMPRESS=YES) - but it can increase space

Saving Space (2 of 4)

- Use views rather than temporary datasets
- Use pipes to access compressed data
- SQL to do merge, summary, sort & print avoiding temporary datasets
- Keep temporary files on tape, cart, optical disk, etc.
- SQL passthru to use temp. space on server machine
- Format to enable coded values to be stored in dataset and later decoded
- Data into 3rd normal form - beware performance costs
- Store data in order it's usually required in - avoiding sorts

Saving Space (3 of 4)

- Delete obs. when reading in data as soon as they are not needed
- Length of time or date values to 4 bytes
- Minimise data kept in SAS libraries
 - Do I need this data?
 - Can data be derived from information stored elsewhere?
 - Is there another copy elsewhere? (eg. DB/2, ORACLE)
 - Is the cost of reproducing the data high?
 - Am I likely to need this data before it is outdated?
 - How much does it cost to keep this data?
 - Can I tell the system to delete my data when it is no longer of any use?

Saving Space (4 of 4)

- Numeric data can be saved in character form
- And visa-versa
- Save numbers with lots of digits as numeric
 - eg. Account #, employee #, student #

MVS example

- 2 byte number can represent a 5 digit number (<65,536)
 - 6 byte number can represent a 13 digit number
- eg. Don't use LENGTH acct \$ 12 ;
But use LENGTH acct 6 ;

There's more ...

- If you want more tips then ...
- Buy the SAS Tips manuals
- Subscribe to SAS-L on the Internet
- Read your Technical Reports
- Attend SMUG and the SAS User Day
- Share your tips with others and they will too