

# Tips & Tricks

With lots of help from other SUG  
and SUGI presenters

A decorative graphic on the left side of the slide. It features a series of vertical stripes in shades of brown, tan, and grey. Overlaid on these stripes are several orange circles of varying sizes. One large circle is positioned near the top, with several smaller circles scattered below it, some overlapping the stripes.

1

SAS HUG Meeting,  
November 18, 2010

I DON'T THINK  
I HAVE YOUR FULL  
ATTENTION.



www.dilbert.com  
scottadams@aol.com

IT'S ASOK'S TURN TO  
LISTEN. IF YOU SAY  
ANYTHING USEFUL, HE'LL  
SEND US AN INSTANT  
MESSAGE.

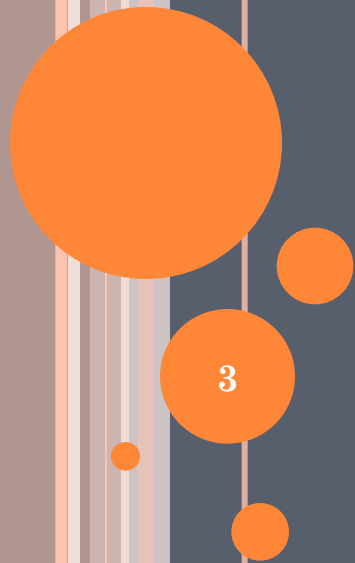


11-8-08 ©2008 Scott Adams, Inc./Dist. by UFS, Inc.

HE'S  
ASLEEP.  
HE'S  
EMPLOYING  
HEURISTICS.



# Sorting



# Threads

- Multi-threading available if your computer has more than one processor (CPU)
- Divide and conquer: calculations are split up and run in parallel on the processors

# Sorting

- In memory or using a utility file
  - `details` option
- Three copies of the data
  - Original data set
  - Temporary sorted files
  - Sorted version of the data set
  - Original file not overwritten until sort is complete ...
    - .... unless you specify the 'overwrite' option

# Sorting - tagsort

- Useful when there is not enough disk space to sort a large data set.
- Only sort keys (listed on the by statement) and observation number (= tags) are sorted
- Tags are used to retrieve record from input dataset in sorted order.
- If sort keys small relative to total record length, temporary disk use is reduced.
- Requires reading each observation twice
- Single-threaded.
- ∴ Considerably slower

# Speaking of disk space

- Where are the “utility” files (sas7butl) stored
  - No “out” file specified – files are stored in same location as the original file
  - “Out” option specified – files are stored in location of the output file

`proc sort data = mydata out = smalldisk.mydata;`

may cause problems.

# The 'noequals' option and SQL

- Default sorting behavior is to retain the order of records within each 'by' group
- 'noequals' does not retain the order
- Faster .....
- ....unless you are relying on retaining the order
- SQL does the equivalent of 'noequals'



# Compression and sorting

- SAS compress removes repeating blanks, characters, and numbers from each observation
- Adds a tag, containing the information needed to uncompress the observation.
- Sorting may be faster on an compressed data set (less I/O)
- !!Compress can result in a data set that is *larger* than the original (even in v9)
  - – size of tag may be larger than saved space.

# Saving Disk Space



10

# Compress (again)

- options compress = yes;
  - Requires CPU to uncompress each observation prior to use
  - Data sets may grow rather than shrink
  - Probably not a good idea to apply compression automatically
- data mydata (compress = yes);
  - Check log file to see if file shrunk

# Drop variables and observations

- Drop unnecessary variables ASAP (use the keep and drop options on the data step)
- Get rid of unnecessary observations ASAP (use the “where” statement on input to avoid wasting CPU time processing observations you don’t need to process)

data females; set all (where = (sex = ‘F’));

- Murphy’s law of SAS datasets
  - Any variable that is dropped from a dataset will be required two procs later.

# Length statement

- Each character (including spaces!) requires 1 byte.

```
data bigword;  
    length word $10;  
    length flag 3;
```

To change the length of a variable after the fact

```
data smallword;  
    length word $4;  
    set mydata;
```

# Numbers

Length in bytes	Largest integer represented exactly	Exponential notation
3	8,192	$2^{13}$
4	2,097,152	$2^{21}$
5	536,870,912	$2^{29}$
6	137,438,953,472	$2^{37}$
7	35,184,372,088,832	$2^{45}$
8	9,007,119,254,740,992	$2^{53}$

# Character strings

- To find the length of a character string (ignoring trailing blanks): `word_length = length(word);`

# The %squeeze macro

## Numbers

- Repeatedly remove 1 byte from each numeric variable until value stored in (n-1) bytes  $\neq$  value stored in (n) bytes.

```
data test ;
```

```
    a = 2001 ;
```

```
    if trunc( a, 7 ) ne a then length_a = 8 ;
```

```
    else if trunc( a, 6 ) ne a then length_a = 7 ;
```

```
    else if trunc( a, 5 ) ne a then length_a = 6 ;
```

```
    else if trunc( a, 4 ) ne a then length_a = 5 ;
```

```
    else if trunc( a, 3 ) ne a then length_a = 4 ;
```

```
    else length_a = 3 ;
```

```
run ;
```



# The %squeeze macro

<http://support.sas.com/kb/24/804.html>

# Other fun tips

# Faster Interactive Processing

- Autoscroll 0 (enter the command in the log window)
  - Suppresses scrolling of windows
  - SAS doesn't use resources to update the display of the LOG window during processing
  - For the output window, autoscroll is set to 0 by default

# Mixed numeric informats

```
proc format;
    invalue mixed 'LOW' = -99
                1-10 = 1
                11-20 = 2
                'BIG' = 99
                other = 0;

run;
data sample;
    informat value mixed.;
    input value;
    datalines;
LOW
1
5
11
50
BIG
;
```

LOW

1

5

11

50

BIG

;

proc print;

-99

1

1

2

0

99

# How many variables are in a data set

```
data _null_;  
    set sashelp.vtable (where = (libname = 'WORK'  
    and memname = 'TEST'));  
    call symput ('nvar', nvar);  
run;  
  
%put &nvar;
```

- Use with caution if you have a lot of libnames specified in your autoexec.sas file.
- Note capital letters for the libname and dataset name.

- Dictionary and SAShelp Views
  - contain information about the current session
  - can be used like any read-only SAS dataset
  - VTABLE summarizes data sets
- Dictionary views require SQL,
- SAShelpviews can be used with a data step

```
proc contents data = sashelp.vtable;
```

## How many variables are in a data set (2)

```
proc sql noprint;  
    select nvar  
    into :nvar2  
    from dictionary.tables  
    where libname = 'WORK'  
    and memname = 'TEST';  
quit;  
  
%put &nvar2;
```



# How many variables are in a data set (3)

```
%let  
  nvar3=%sysfunc(attrn(%sysfunc(pen(work.test,i))  
    ,nvars));  
  
%put &nvar3;
```

# Dates

## The INTNX function

- Advances a date by a given interval
- Returns the SAS date that is the given number of increments of a time interval from the starting date
- Syntax: INTNX('interval', start-from-date, increment)
  - The first parameter is an interval key word enclosed in single quotes. This can be YEAR, QTR, MONTH, DAY, etc.
  - The second parameter is a SAS date.
  - The third parameter is a number, how many increments to move.

# The INTNX function (cont'd)

- The name of the interval has the syntax:  
Name<multiple><.starting-point>.
- Name is the name of the interval (week, year, etc.)
- Multiple creates a multiple of the interval (default = 1).
  - For example, YEAR2 indicates a two-year interval.
- .starting-point is the starting point of the interval (default = 1).
  - A value greater than 1 shifts the start to a later point within the interval.
  - The unit for shifting depends on the interval.
  - YEAR.3 specifies a yearly period from the first of March through the end of February of the following year.
  - WEEK.4 specifies a weekly period, starting on Wednesday
- <http://www2.sas.com/proceedings/sugi31/015-31.pdf>

```
data test (drop = i);  
  do i = 1 to 12;  
    date = mdy(i, 01, 2003);  
    if weekday(date) = 4 then thirdwed =  
      intnx ('week.4', date, 2);  
    else thirdwed = intnx('week.4', date, 3);  
    output;  
  end;  
  format date thirdwed weekdate.;  
run;
```

# Spell Checker

```
filename temp temp;  
data _null_;  
    input word: $12. @@;  
    list;  
    file temp;  
    put word;  
    datalines;
```

lets see if the sas spell checker procedure  
can be used to verify whether the  
separate words in this file are valid  
against a standard internal dictionary  
;

```
proc spell wordlist = temp verify suggest;  
run;
```

<http://analytics.ncsu.edu/sesug/2007/SD06.pdf>

# Options

- To see what options are available:
  - Proc options; run;
  - Proc options internal; run;
- The most important option in SAS
  - option mergenoby = error [warn nowarn];
- The second most important option in SAS
  - options nofmterr;
- The most environmentally important option
  - options formdlm = '-';

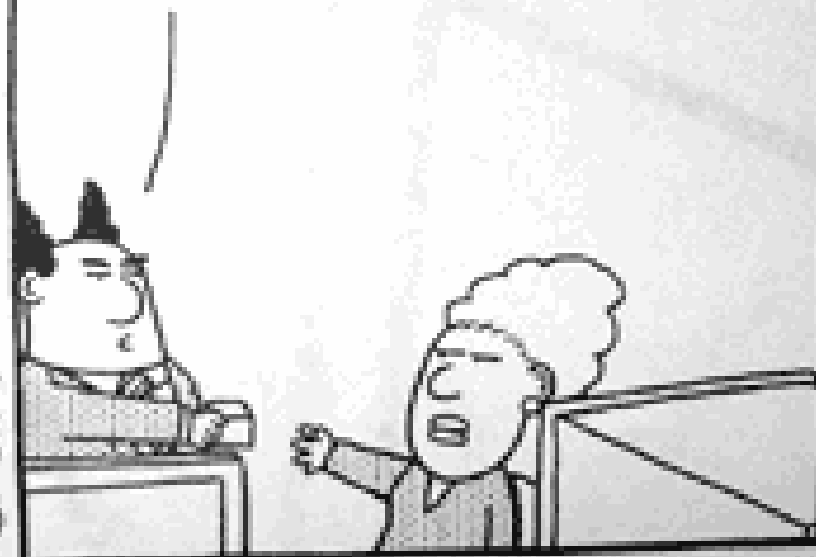


CAROL, THE NEXT TIME YOU  
ORDER MY BUSINESS CARDS,  
SPELL OUT MY FULL TITLE:  
"DIRECTOR OF PRODUCT  
ENHANCEMENTS."



S. ALBARRAS

DON'T USE THE  
ACRONYM "DOPE."



© 1996 United Feature Syndicate, Inc. (NYC)