



CC07 PhUSE 2011

Seven Sharp tips for Clinical Programmers

David Garbutt
Rohit Banga
BIOP AG

Agenda

- **Innovative way of Checking Log**
- **Unexpected Merge behaviour**
- **Use Formats instead of joins**
- **Use Hash Objects for table lookup**
- **PROC FCMP to replace macros**
- **Using Picture Formats to display dates**

Innovative of checking Log

- Use DM Statement to copy log to catalog
- Input the catalog in a dataset and search for errors
- Display errors in a separate window
- Make a toolbar

Unexpected Merge Behaviour

LAB

SUBJECT	VISIT	VISITNAME
SUB_1	1	OldVisit1
SUB_1	1	OldVisit1
SUB_1	1	OldVisit1
SUB_1	1	OldVisit1
SUB_1	2	OldVisit2
SUB_1	2	OldVisit2
SUB_1	2	OldVisit2
SUB_1	2	OldVisit2

VIS

SUBJECT	VISIT	VISITNAME
SUB_1	1	NewVisit1
SUB_1	2	NewVisit1
SUB_2	1	NewVisit1
SUB_2	2	NewVisit1



MERGE

Expected
LAB

SUBJECT	VISIT	VISITNAME
SUB_1	1	NewVisit1
SUB_1	1	NewVisit1
SUB_1	1	NewVisit1
SUB_1	1	NewVisit1
SUB_1	2	NewVisit2
SUB_1	2	NewVisit2
SUB_1	2	NewVisit2
SUB_1	2	NewVisit2

Formats Instead of Joins

Problem 1 –

The subject's last dosing date is in CMP dataset. I want to check whether dosing date (in DAR dataset) is before subject's last dosing date

Using hash Objects for joining

Not as hard as it sounds

```
Disvalue[labparm] ++ 1 ;
```

SAS program for to replace format in merge

With hash object

```
data houtput ;
  set dar ;
  if _n_ = 1 then do;
    rc = 0 ;
    length Subject $ 8 LastTRT $9;
    *-- declare the hash object. Ours is called endtrt
    *-- load the hash obj by specifying the dataset name ;
    DECLARE HASH endtrt(dataset: "work.cmp",HASHEXP:16);
    rc + endtrt.DEFINEKEY ('SUBJECT');
    rc + endtrt.DEFINEDATA('SUBJECT', 'LastTRT');
    rc + endtrt.DEFINEDONE();
  end ;
  rc = endtrt.find() ;
  If input(DOSDT,??Date9.) GT input(LastTRT,??Date9.)
    and rc = 0 ;
  drop rc ;
run;
```

SAS program for merge

With hash object

```
*** merge using a hash table instead of a data step
    model the dataset exactly *** ;

data hResult ;
    set lab ;
    if _n_ = 1 then do;
        rc = 0 ;

        length Subject $ 8  visitname $10;
        *-- declare the hash object. Ours is called endtrt
        *-- load the hash obj by specifying the datasetname ;

        DECLARE HASH visnam(dataset: work.vis",HASHEXP:16);
        rc + visnam.DEFINEKEY ('SUBJECT','VISITNO');
        rc + visnam.DEFINEDATA('SUBJECT', 'VISITNO','VISITNAME');
        rc + visnam.DEFINEDONE();

        if 0 then visitname = ' ' ;
    end ;

    *--- load the value of visit name in the from the hash that
    *--- is stored with current value of subject and visit;
    rc = visnam.find() ;

    drop rc ;

run;
```

Proc FCMP

- Compiled functions
 - Using with data step code

Why Functions?

```
data _null_;  
infile "dataset_list.txt" lrecl=10000 dsd dlm='|'  
    end=lastrow length=linelen column=currcol;  
input dsname $ descrip : $32. @; ...
```

```
call symput ('descrip' || left (_n_),  
    tranwrd((tranwrd ((tranwrd ((tranwrd ((tranwrd  
    ((tranwrd ((tranwrd ((tranwrd ((tranwrd  
    ((tranwrd  
        (trim (descrip), '&', '\&'))),  
    '{', '\{''))), '}', '\}''))), '_ ', '\_{}''))), '%', '\%''))),  
    '~', '\~{}''))), '$', '\$''))), '<', '$<$''))), '>', '$>$''))),  
    '^', '\^{}''))), '#', '\#'')));
```

The functionless skeleton

```
proc fcmp outlib= sasuser.MySubs.davefunc ;  
  
function Latexencode(var $ ) $ 1024;  
    /* see next slide for code*/  
endsub ;  
  
quit;  
  
options CMPLIB = sasuser.Mysubs;  
/* run some tests */  
...
```

```

do i = 1 to length (name);
  c = upcase(substr (name,i,1));
  j = 0;
  select (c);
    when ('\') put '$\backslash$'@;
    when ('{') put '\{'@;
    when ('}') put '\}'@;
    when (%) put '\%'@;
    when ('&') put '\&'@;
    when ('~') put '\~{}'@;
    when ('$') put '\$'@;
    when ('^') put '\^{}'@;
    when ('_') put '\_{}'@;
    when ('#') put '\#'@;
    otherwise put c@; j = -1;
  end;
  put +(j)@;
end;

```

Loses flexibility due to put

```

function Latexencode(var $ ) $ 1024;
length result $ 1024 c tab sp $ 1 ;
result = '' ; c='' ; sp = ' ' ; tab = byte(5);
do i= 1 to length(var) ;
  c = substr(var,i,1) ;
  select (c);
    when (' ') result = catt(result, TAB) ;
    when ('\\') result = catt(result, '\\backslash$');
    when ('{') result = catt(result, '\\{');
    when ('}') result = catt(result, '\\}');
    when ('%') result = catt(result, '\\%' );
    when ('&') result = catt(result, '\\&' );
    when ('~') result = catt(result, '\\~{ }');
    when ('$') result = catt(result, '\\$');
    when ('^') result = catt(result, '\\^{ }');
    when ('_') result = catt(result, '\\_{ }');
    when ('#') result = catt(result, '\\#');
    when ('<') result = catt(result, '$<$');
    when ('>') result = catt(result, '$>$');
    otherwise result = catt(result, c );
  end;
end;
result = translate(compress(result), sp,tab) ;
return(result);
endsub ;

```

The code- working on variables

```

do i = 1 to length(name);
    c = substr(name,i,1);
    j = 0;
    select (c);
        when (c = '\') put '\backslashslash$';
        when (c = '\b') put '\backslashb$';
        when (c = '\f') put '\f$';
        when (c = '\n') put '\n$';
        when (c = '\r') put '\r$';
        when (c = '\t') put '\t$';
        when (c = '\') put '\$';
        when (c = '\a') put '\a$';
        when (c = '\e') put '\e$';
        otherwise do put c$; j = -1; end;
    end;
    put c$();
end;
put ' ';
do i = 1 to length(label);
    c = substr(label,i,1);
    j = 0;
    select (c);
        when (c = '\') put '\backslashslash$';
        when (c = '\b') put '\backslashb$';
        when (c = '\f') put '\f$';
        when (c = '\n') put '\n$';
        when (c = '\r') put '\r$';
        when (c = '\t') put '\t$';
        when (c = '\') put '\$';
        when (c = '\a') put '\a$';
        when (c = '\e') put '\e$';
        otherwise do put c$; j = -1; end;
    end;
    put c$();
end;
if type = 1 then
    put ' & Num &';
else
    put ' & Char &';
/* Put the variable's length in the report.
do i = 1 to length(length);
    c = substr(length,i,1);
    j = 0;
    select (c);
        when (c = '\') put '\backslashslash$';
        when (c = '\b') put '\backslashb$';
        when (c = '\f') put '\f$';
        when (c = '\n') put '\n$';
        when (c = '\r') put '\r$';
        when (c = '\t') put '\t$';
        when (c = '\') put '\$';
        when (c = '\a') put '\a$';
        when (c = '\e') put '\e$';
        otherwise do put c$; j = -1; end;
    end;
    put c$();
end;
put ' &';
do;
    if (format NE ' ')          AND
       (format NE 'DATE')      AND
       (format NE 'TIME')      AND
       (format NE 'DATETIME')  AND
       (format NE 'REST')      AND
       (format NE 'Y')          AND
       (format NE 'BINARY')    AND
       (format NE 'SCHAR')     AND
       (format NE 'YMWDM')     AND
       (format NE 'VQ')        AND
       (format NE 'VQ64')       AND
       (format NE 'VQ8')        AND
       (format NE 'ZD') then
do;
    put 'Vperror$';
    do i = 1 to length(format);
        j = 0;
        select (c);
            when (c = '\')
                otherwise do put c$; j = -1; end;
        end;
        put c$();
    end;
    put '()';
    do i = 1 to length(format);
        c = substr(format,i,1);
        j = 0;
        select (c);
            when (c = '\') put '\backslashslash$';
            when (c = '\b') put '\backslashb$';
            when (c = '\f') put '\f$';
            when (c = '\n') put '\n$';
            when (c = '\r') put '\r$';
            when (c = '\t') put '\t$';
            when (c = '\') put '\$';
            when (c = '\a') put '\a$';
            when (c = '\e') put '\e$';
            otherwise do put c$; j = -1; end;
        end;
        put c$();
    end;
    put '()';
    end; /* Tell LaTeX to finish off the
put ' \';
put ' \line;

```

```
col1 = latexencode(uppercase(varname) ) ;
col2 = latexencode(varlabel);
```

```
col3 = ' & Num & ' ;
```

```
col3 = ' & Char & ' ;
```

```
col5 = latexencode(formatname) ;
```

```
put ' } \ \ ';
```

```
put' \hline' ;
```

225 lines → **11 lines**

Simplifying put:

Put does not allow function calls, the new cats function help...

```
Thisline = cats(  
  latexencode( uppercase(varname) ) ,  
  '&',  
  latexencode(varlabel),  
  ifc( type , 'Num (' || strip(put(length, 10.0)) || ')',  
        'Char (' || strip(put(length, 10.0)) || ,  
        '???No Type defined!') ,  
        ' & \hyperref(' , latexencode(formatname),  
        '{}{' , latexencode(formatname),  
        '} \\ hline' );  
Put Thisline ;
```

225 lines → 2 (logical) lines

Results

→ In file

```
Age\_s & Age at study start & Num (8) & \hyperref{age\_st.}{age  
\_st.} \\ \hline
```

→ appears in PDF as:

<i>Variable</i>	<i>Label</i>	<i>Type (Length)</i>	<i>Format</i>
Age_s	Age at study start	Num (8)	age-st

Picture Formats

Problem 1 –

I want to Display Today's date as –

- 16/11/2010
- Nov.16.2010
- Tuesday, November 16, 2010
- Maybe -> 16+November+10

Many Thanks for your attention

Also to
Christoph Baumer – BIOP AG
Armin Gemperli – (Not) BIOP AG (anymore)







1. Distrust
Can I do it?



2. Excitement
I can do it!!!



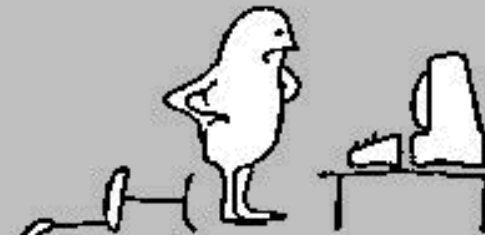
3. Astonishment
How will I do it?



4. Enthusiasm
I got hold of the flow!!!



5. Love
I am an excellent programmer!



6. Disillusionment
Code is not functioning properly



7. Fright
Will this logic work?



8. Horror
Another A level bug!!!



9. Fury
Damn with computers
#@#\$\$@^



10. Frustration
It is not working in expected manner



11. The End
Project Appraisal

