

Define Your Own SAS Functions for Date and Numeric Validation Using PROC FCMP

Jason A Smith
Argo Analytics Ltd

Contents

- Introduction to SAS Function Compiler (FCMP)
- NUMVAL function: simple example
- DATEVAL function: complex example
- Nesting functions
- Conclusion

SAS Function Compiler (FCMP)

- use to create and store SAS functions and CALL routines

```
proc fcmp outlib=<libname>.<dataset>.<package>;  
  function <funcname> (<argument>);  
  
    ...your code here
```

```
  endsub;  
run;
```

- specify location(s) of functions

```
options cmplib = <libname>.<dataset>;
```

NUMVAL function: simple example

numval (var \$)

- check if character variable holds a valid numeric value
- function returns 1 (valid) or 0 (invalid)
- allows any valid number except for the scientific E notation
- avoids this kind of message:

NOTE: Invalid argument to function INPUT at line 47 column 14.

RULE: ----+----1----+----2----+----3----+----4----+----5----+----6----+---

50 <5

lbscresc=<5 lbscresn=, _ERROR_=1 _N_=2

NOTE: Mathematical operations could not be performed at the following places.

The results of the operations have been set to missing values.

Each place is given by: (Number of times) at (Line):(Column).

1 at 47:14



NUMVAL function definition

- define NUMVAL function

```
proc fcmp outlib=work.funcs.myfuncs;  
  function numval (var $);  
  
    return (not (notdigit (strip (translate (var,'000','.+-'))) or  
              count (var,'.')>1 or countc (var,'+-')>1 or  
              indexc (left (var), '+-')>1 or  
              left (var) in ('+', '+.', '-', '-.')));  
  
  endsub;  
run;
```

- specify location(s) of functions

```
options cmplib = work.funcs;
```

NUMVAL function definition

- SAS log confirms that the function has been saved:

```
43      proc fcmp outlib=work.funcs.myfuncs;  
44          function numval (var $);  
45  
46              return (not (notdigit (strip (translate (var,'000','.-')))) or  
47                      count (var, '.')>1 or countc (var, '+-')>1 or  
48                      indexc (left (var), '+-')>1 or  
49                      left (var) in ('+', '+.', '-', '-.')));  
50  
51      endsub;  
52      run;
```

NOTE: Function numval saved to work.funcs.myfuncs.

NOTE: PROCEDURE FCMP used (Total process time):

real time	0.02 seconds
cpu time	0.02 seconds

```
53  
54      options cmplib = work.funcs;
```

NUMVAL function usage

- check if variable contains a numeric value
- if not a valid number then output a warning

```
data one;  
  input lbstresc $;  
  if numval (lbstresc) then lbstresn = input (lbstresc, best.);  
  else put "USER WAR" "NING: invalid value " lbstresc=;  
  
datalines;  
1.1  
<5  
-3  
;  
run;
```

NUMVAL function usage

- no untidy “Invalid argument to function INPUT” messages
- log outputs the specified User Warning for any invalid value

USER WARNING: invalid value lbstresc=<5

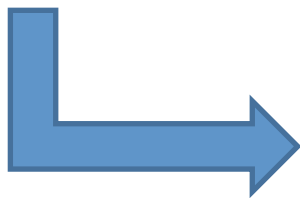
NOTE: The data set WORK.ONE has 3 observations and 2 variables.

NOTE: DATA statement used (Total process time):

real time 0.02 seconds

cpu time 0.04 seconds

- valid numeric values have been input as expected



WORK.ONE

x

ames

Total rows: 3 Total columns: 2

	lbstresc	lbstresn
1	1.1	1.1
2	<5	.
3	-3	-3

DATEVAL function: complex example

dateval (dvar \$, dfmt \$)

- will validate a date in any of the formats:

DATEx – 14 OCT 2014

DDMMYYx – 14/10/2014

MMDDYYx – 10/14/2014

YYMMDDx – 2014/14/10

- second argument specifies the format, eg:

dateval (<varname>, 'yymmdd10')

- function returns 1 (valid) or 0 (invalid)
- mimics the rules used by SAS when using an input statement with the corresponding format



DATEVAL function usage

- check if date is in YYMMDD10. format
- next check if date is in DATE11. format
- if neither format is valid then output a warning

```
data two;
  length visitdte $20;
  format visitdte date9.;
  input visitdte $;

  if      dateval (visitdte,'yymmdd10') then visitdte = input (visitdte,yymmdd10.);
  else if dateval (visitdte,'date11')   then visitdte = input (visitdte,date11.);
  else if visitdte ne '' then put "USER WAR" "NING: invalid date " visitdte=;

datalines;
12-OCT-2014
2014/13/14
2014/10/14
;
run;
```

DATEVAL function usage

- no untidy “Invalid argument to function INPUT” messages
- log outputs the specified User Warning for any invalid value

```
USER WARNING: invalid date visitdtc=2014/13/14
```

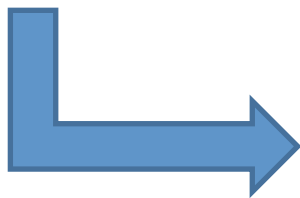
```
NOTE: The data set WORK.TWO has 3 observations and 2 variables.
```

```
NOTE: DATA statement used (Total process time):
```

```
    real time           0.06 seconds
```

```
    cpu time            0.08 seconds
```

- valid date values have been input as expected



 WORK.TWO x

names



Total rows: 3 Total columns: 2

	visitdtc	visitdt
1	12-OCT-2014	12OCT2014
2	2014/13/14	.
3	2014/10/14	14OCT2014

DATEVAL function definition

```
*-----*
* DATEVAL Function Definition
*
* Parameters: DVAR   - Input Date Variable in character format           (required)
*              DFMT   - Date Format - must be DATEx, DDMMYYx, MMDDYYx or YYMMDDx (required)
*-----*
proc fcmp outlib=work.funcs.myfuncs;
  function dateval (dvar $, dfmt $);
    n      = 0;
    fmtc   = upcase (translate (dfmt, ' ', '.'));
    alpha  = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
    num    = '0123456789';
    sep    = ' #!$%&()*+-. /: ;<=>?[\\]^_{|}~) ' ;
    leapyr = '04,08,12,16,20,24,28,32,36,40,44,48,52,56,60,64,68,72,76,80,84,88,92,96' ;

*-----*
* validate input parameters
*-----*

    if fmtc in : ('DDMMYY', 'MMDDYY', 'YYMMDD') then do;
      fmt  = substr(fmtc,1,6);
      lenc = substr(fmtc,7);
    end;
  else if fmtc =: 'DATE' then do;
    fmt  = substr(fmtc,1,4);
    lenc = substr(fmtc,5);
  end;
```

Define Your Own SAS Functions for Date and Numeric Validation Using PROC FCMP

DATEVAL function definition

```
*-----*
* validate input parameters
*-----*

if notdigit(trim(lenc))=0 then len = input(lenc,best.);
else if lenc = ''          then len = 32;

if fmt='' or len=. then do;
  put "%str(ERR)OR:[DATEVAL] date format must be DATEx, DMMYYx, MMDDYYx or YYMMDDx";
  goto uexit;
end;
else if fmt='DATE' & (len < 9 or len > 32) then do;
  put "%str(ERR)OR:[DATEVAL] format length must be 9-32";
  goto uexit;
end;
else if len < 8 or len > 32 then do;
  put "%str(ERR)OR:[DATEVAL] format length must be 8-32";
  goto uexit;
end;
else if length(dvar) > len then goto uexit;
```

DATEVAL function definition

```
*-----*
* parse character date into YY, MM and DD macro variables
*-----*

if fmt='DATE' then do;
    dd = put(scan(upcase(dvar),1,sep||alpha), $2.);
    mm = put(scan(upcase(dvar),1,sep||num), $3.);
    yy = put(scan(upcase(dvar),2,sep||alpha), $4.);

    if compress(upcase(dvar), sep) ne cats(dd, mm, yy) then goto uexit;

end;
else if fmt='DDMMYY' then do;
    dd = strip(put(left(put(scan(right(put(strip(dvar), $11.)), 1, num) || scan(dvar, 1, sep),
        $5.)), $2.));
    mm = strip(substr(put(scan(dvar, 1, sep) || scan(dvar, 2, sep), $4.), length(dd)+1, 2));
    yy = substr(right(put(strip(dvar), $10.)), 7, 4);
    ss = put(scan(strip(dvar), 1, num), $1.);

    if strip(dvar) ne cat(strip(mm), ss, strip(dd), ss, strip(yy)) and
        strip(dvar) ne cats(dd, mm, yy) then goto uexit;

end;
```

DATEVAL function definition

```
*-----*
* parse character date into YY, MM and DD macro variables
*-----*

else if fmt='MMDDYY' then do;
    mm = strip(put(left(put(scan(right(put(strip(dvar), $11.)), 1, num) || scan(dvar, 1, sep),
        $5.)), $2.));
    dd = strip(substr(put(scan(dvar, 1, sep) || scan(dvar, 2, sep), $4.), length(mm)+1, 2));
    yy = substr(right(put(strip(dvar), $10.)), 7, 4);
    ss = put(scan(strip(dvar), 1, num), $1.);

    if strip(dvar) ne cat(strip(mm), ss, strip(dd), ss, strip(yy)) and
        strip(dvar) ne cats(mm, dd, yy) then goto uexit;

end;

else if fmt='YYMMDD' then do;
    yy = put(strip(dvar), $4.);
    mm = strip(substr(put(scan(strip(dvar), 1, sep) || scan(strip(dvar), 2, sep), $6.), 5));
    dd = strip(substr(right(put(strip(dvar), $10.)), 9));
    ss = put(scan(strip(dvar), 1, num), $1.);

    if strip(dvar) ne cat(strip(yy), ss, strip(mm), ss, strip(dd)) and
        strip(dvar) ne cats(yy, mm, dd) then goto uexit;

end;
```

Define Your Own SAS Functions for Date and Numeric Validation Using PROC FCMP

DATEVAL function definition

```
*-----*
* check that DD, MM and YY is a valid date
*-----*

if notdigit(trim(dd))=0 then ddn = input(dd,2.);
if notdigit(trim(mm))=0 then mmn = input(mm,2.);
if notdigit(trim(yy))=0 then yyn = input(yy,4.);
if ddn=. or (mmn=. & fmt ne 'DATE') or yyn < 1582 or yyn > 2999 then goto uexit;

if fmt='DATE' then
  n = ((mm in ('JAN','MAR','MAY','JUL','AUG','OCT','DEC') and 1 <= ddn <= 31) or
        (mm in ('APR','JUN','SEP','NOV') and 1 <= ddn <= 30) or
        (mm = 'FEB' and 1 <= ddn <= 28) or
        (mm = 'FEB' and ddn = 29 and (index (leapyr, substr (yy, 3, 2)) or
          yyn in (1600,2000,2400,2800)))));

else n = ((mmn in (1,3,5,7,8,10,12) and 1 <= ddn <= 31) or
          (mmn in (4,6,9,11) and 1 <= ddn <= 30) or
          (mmn = 2 and 1 <= ddn <= 28) or
          (mmn = 2 and ddn = 29 and (index (leapyr, substr (yy, 3, 2)) or
            yyn in (1600,2000,2400,2800)))));

uexit:

return (n);
endsub;
run;
```

Define Your Own SAS Functions for Date and Numeric Validation Using PROC FCMP

Nesting functions

- a saved function can be used in further function definitions
- the DATEIMP function uses the DATEVAL function to:
 - check if a full or partial date is valid (YYYY-MM-DD)
 - input any valid complete date
 - impute the date to the start of the month for a missing day
 - impute the date to the start of the year for a missing month/day



DATEIMP function definition

- define DATEIMP function

```
proc fcmp outlib=work.funcs.myfuncs;  
  function dateimp (dvar $);  
  
  /* Complete Date */  
    if dateval (dvar,'yymmdd10') then impdt = input (dvar,yymmdd10.);  
  
  /* Missing Day */  
    else if dateval (cats(dvar,'-01'),'yymmdd10') then  
      impdt = input (cats(dvar,'-01'),yymmdd10.);  
  
  /* Missing Month */  
    else if dateval (cats(dvar,'-01-01'),'yymmdd10') then  
      impdt = input (cats(dvar,'-01-01'),yymmdd10.);  
  
    else put "%str(USER WAR)NING: invalid date" dvar;  
  
    return (impdt);  
  
  endsub;  
run;
```

Define Your Own SAS Functions for Date and Numeric Validation Using PROC FCMP

DATEIMP function usage

- use DATEVAL to check if date is valid and complete
- use DATEIMP to impute any valid partial date

```
data three;  
  length visitdte $20;  
  format visitdt visitdte date11.;  
  input visitdte $;  
  
  if dateval (visitdte,'yymmdd10') then visitdt = input(visitdte,yymmdd10.);  
  
  visitdte = dateimp (visitdte);  
  
datalines;  
2014-10-14  
2014-10  
2014-101  
2014  
;  
run;
```

DATEIMP function usage

- no untidy “Invalid argument to function INPUT” messages
- log outputs the specified User Warning for any invalid value

USER WARNING: invalid date 2014-101

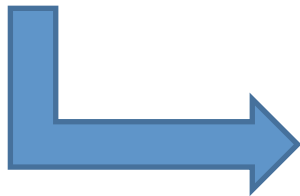
NOTE: The data set WORK.THREE has 4 observations and 3 variables.

NOTE: DATA statement used (Total process time):

real time 0.06 seconds

cpu time 0.07 seconds

- valid partial dates have been imputed as expected

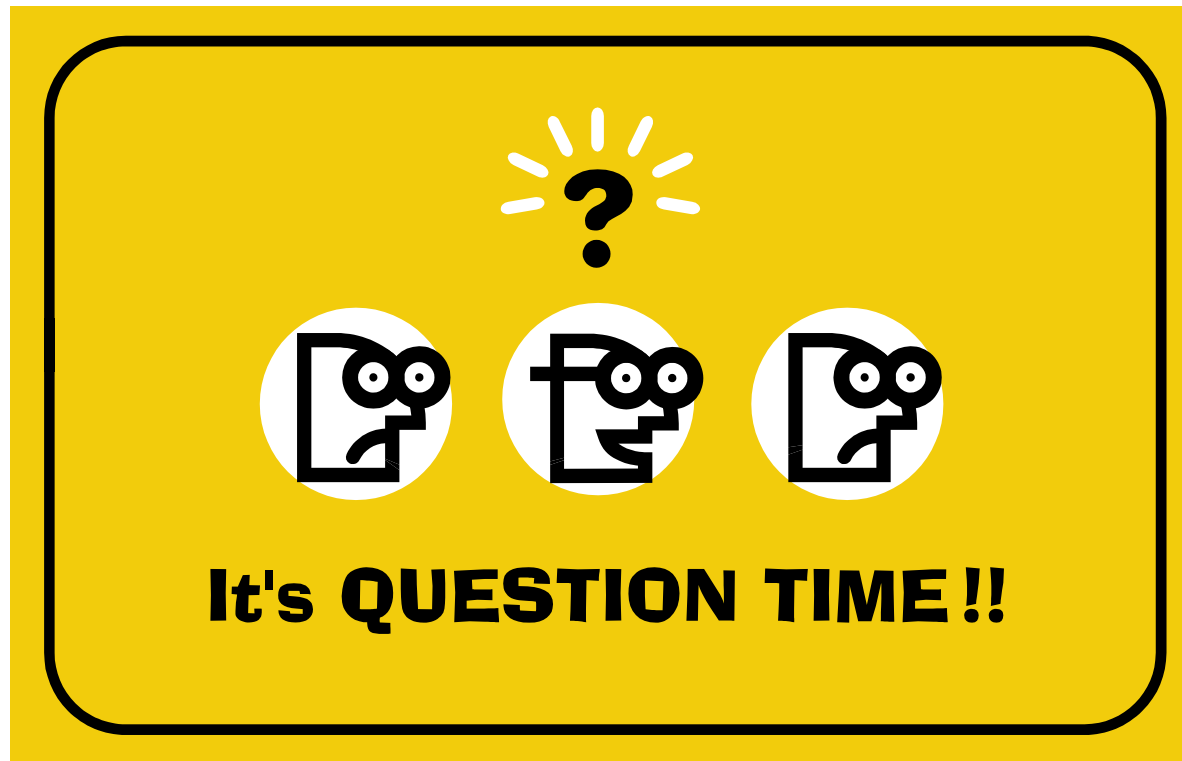


WORK.THREE x			
names			
Total rows: 4 Total columns: 3			
	visitdtc	visitdt	visitdti
1	2014-10-11	11-OCT-2014	11-OCT-2014
2	2014-10	.	01-OCT-2014
3	2014-101	.	.
4	2014	.	01-JAN-2014

Conclusion

- The SAS Function Compiler is a powerful and flexible tool that can simplify programming and data checking
- Next time that you find yourself wishing for a function that does not exist, say “I can define this function for myself!”

Questions?



Contact

Jason A Smith

Argo Analytics Ltd

32 Woodlark Road, Cambridge CB3 0HS, UK

Phone: +44-7792-046599

Email: jason@argoanalytics.co.uk

LinkedIn: <http://uk.linkedin.com/pub/jason-smith/46/baa/61>

Web: www.argoanalytics.co.uk

