Paper 1744-2014

VFORMAT Lets SAS® Do the Format Searching

Peter Crawford, Crawford Software Consultancy, UK

ABSTRACT

When reading data files or writing SAS® programs, we are often hunting for the right format or informat. There are so many to choose from! Does it seem like too many to search the manual? Let SAS help find the right one! We use the SAS dictionary table VFORMAT and a very small SAS program. This presentation demonstrates how two simple functions unlock the potential of this great resource: SASHELP.VFORMAT.

INTRODUCTION

The programming language of the base SAS System has many capabilities that are like hidden gems, waiting to be discovered. Making the SAS System extremely flexible, it offers a great many ways for formatting values for display, and for parsing text into values. These are the formats and informats of the SAS System.

This paper introduces a small SAS program to help choose the format we need.

The paper is intended for all expertise levels of SAS Programmer and SAS Developer.

THE PROBLEM DESCRIBED

There are so many formats and informats it has become a challenge to remember all of these and how they work to help us.

In my base SAS 9.4 install there are 1,020 formats and informats. Adding “user-defined” formats created by the FORMAT Procedure will increase this count.

Finding the most suitable format and informat is not always easy. Although we can create and share our own formats with the FORMAT Procedure, we save ourselves the fuss of supporting User Format Libraries when we can choose the built-in formats provided with the SAS System. It might take quite a time to pick out the most suitable format if you do not know how each behaves.

In SAS manuals, the documentation of formats and informats is also split. There are 284 pages documenting formats and informats in the primary formats manual: “SAS® 9.4 Formats and Informats: Reference”[1]. Further format documentation is separated into manuals for:

- NLS[2] (National Language Support)
- DS2[3] (“Data Step 2” provides SAS language handling for DBMS data types not normally in SAS Data)
- FedSQL[4] (Federation SQL® is SAS proprietary implementation of ANSI SQL:1999 core standard)
- Additional SAS products (like SAS/ETS®) add additional formats, documented within those products.

Unfortunately Google does not (yet?) offer a search facility that associates a SAS format name with the output style we desire.

One of the old Technical Support documents TS:486 listed SAS Functions and Formats with sample inputs and expected outputs. Some efforts have been made to bring that document up-to-date in www.sascommunity.org

THE SOLUTIONS PROPOSED

Although the documentation on 1000+ formats and informats are very difficult to consider a realistic place to search for the format we need, as programmers we have a very effective alternative: write a program:

> Program Spec:
>
> Test each format with a sample value. Test each informat with a sample string. When the informat produces no _ERROR_ the informat might be suitable. For both format and informat searches, we can test if the result is “as expected”.

OK – but I’m not suggesting a thousand lines of code!

SOLUTION PART 1 THE FORMATS LIST

The SAS System includes many metadata tables one of which lists all the formats installed in that system. This metadata can be accessed in a procedure or data step through view

- SASHELP.VFORMAT

SOLUTION PART 2 FORMAT NAME AS A VARIABLE

To engineer a solution, the next part must be:

- Some way to treat each of those format names as a “variable”.

This is unusual because nearly always, we have to explicitly specify the format name – even when it is provided by a macro variable.

The format still has to be explicitly defined:

- in the statements, PUT,INPUT, FORMAT, ATTRIB;
- in the FORMAT attribute of a column in an SQL query;
- in the functions INPUT and PUT;
- even when the second parameter of macro function %SYSFUNC is used, the format must be named.

There is only one place where a format name can be “dynamic”, i.e. provided in the value of a variable (the most easily understood example of this is having the format named in the value of a DATA Step variable).

Similarly, there is only the one context where the INFORMAT name can be “dynamic” in this way.

This context is the PUTN function (for formats), and (for informats) the INPUTN function.

The second parameter for these functions is either a variable which contains the format name or an expression which evaluates at run-time (not compile-time), to a format name.

SOLUTION PART 3 SIMPLE EXAMPLE WITHOUT A LIST OF FORMATS

Although these functions can be used in many environments, the easiest to describe is a DATA Step. To demonstrate the PUTN function, here is some source code (DEMO1):

```
DATA one;
 value = '9:9:9'T ;
 format= 'time' ;
 string= PUTN( value, format ) ;
RUN ;
```

Clearly this demonstrates only PUTN() handling a variable containing a format name. It contains nothing to loop over all the formats.

SOLUTION PART 4 – USING A VARIABLE FORMAT FROM THE LIST

The normal DATA Step provides looping, so it could loop over all formats.

A recent (typical) request for help on SAS Communities provides an example of the “need”.

> With a user-defined format as the “last resort”, what format will provide leading zeroes on the time parts?
>
> (I have paraphrased the request in the original posting at https://communities.sas.com/message/191798)

The requested format is found with code like:

DEMO 2

```
DATA two;
  RETAIN wanted '090909'             /* result wanted       */
         value   '9:9:9'T            /* time to be tested   */
        ;
  LENGTH string $40 ;                 * container for result ;
  SET    sashelp.vformat ;            * list of all formats  ;
  WHERE  fmtType = 'F'   ;            * ignore INFORMAT rows ;
  WHERE  ALSO fmtName ne: '$' ;       * and character formats;

  String = PUTN( value, fmtName ); * APPLY THE FORMAT   ;

  if LEFT( string ) = wanted ;        * check result         ;
  PUT value= time. fmtName=  string ; * report               ;
RUN ;
```

Clearly there are limits in this demonstration, but on this occasion the “format hunter” has found the answer with these results in the SAS log.

The results of the format search:

```
value=9:09:09 fmtName=B8601TM 090909
NOTE: There were 620 observations read from the data set SASHELP.VFORMAT.
      WHERE (fmtType='F') and (fmtName not =: '$');
```

Output 1. Log from Format Hunter

MORE SOLUTION DEMONSTRATIONS

DEMO 3

Show any formats which provide the day of the week as well as time, for a date-and-time value.

```
DATA THREE ;
  RETAIN value  '26Mar2014:10:11:23'dt  /* value to be tested */
         Want1  'Wednesday'             /* result parts wanted*/
         Want2  '26'                    /* result parts wanted*/
         Want3  '11'                    /* result parts wanted*/
        ;
  LENGTH string $40                     /* container for result*/
        ;
  SET   sashelp.vformat ;               * list of all formats  ;
  WHERE fmtType = 'F'   ;               * ignore INFORMAT rows ;
  WHERE ALSO fmtName ne: '$' ;          * and character formats;

  string= PUTN( value, fmtName ) ;    * perform test         ;
  if find( string , want1, 'it' ) and
     find( string , want2, 'it' ) and
     find( string , want3, 'it' ) ;  * check result         ;
  PUT value= time. fmtname=  string ;* report               ;
RUN ;
```

results of search for day and time.

```
value=475402 fmtname=NLDATMW Wednesday, 26 March 2014 10:11:23
value=475402 fmtname=NLDATMWZ Wednesday, 26 March 2014 10:11:23 +0000
value=475402 fmtname=TWMDY 10:11 Wednesday, March 26, 2014
NOTE: There were 621 observations read from the data set SASHELP.VFORMAT.
      WHERE (fmtType='F') and (fmtName not =: '$');
NOTE: The data set WORK.THREE has 3 observations and 18 variables
```

Output 2. Log from Format Hunter

Nice to see that format **TWMDY** makes the style provided by the DATE option on a TITLE line, available anywhere.

DEMO 4

To show an INFORMAT search

```
option ls= 160 ;
%LET string = 26/03/2014 10.11.23 ; * value to be tested *;
%LET require= "26Mar2014:10:11:23"dt ; * result wanted ;
DATA FOUR_dt  ;
  RETAIN string "&string"          /* value to be tested */
         Want1   &require          /* result wanted      */
        ;
  SET    sashelp.vformat ;          * list of all formats  ;
  WHERE  fmtType = 'I'   ;          * only   INFORMAT rows ;
  WHERE  ALSO fmtName ne: '$' ;     * not character formats;
  where  also fmtName NE 'PCPIB' ;  * exclude defect !     ;
```

```
    inValue = INPUTN( string, fmtName, length(string) ) ;    * perform test ;
    if _error_ then do;
        _error_ = 0 ; * avoiding variable dump in the SAS log ;
        Delete ;
     End ;
    if invalue = want1  ;        * now check value returned ;
    PUT "&string " inValue= invalue dateTime. +1 fmtname=  ;
RUN ;
```

```
……99 LOG lines suppressed……..
NOTE: Argument 3 to function INPUTN('26/03/2014 1'[12 of 19 characters shown], 'MSEC      '[12 of 32 characters shown],19) at line 4514 column 13 is invalid.
NOTE: Over 100 NOTES, additional NOTES suppressed.
NOTE: Argument 3 to function INPUTN('26/03/2014 1'[12 of 19 characters shown],'ND8601DA   '[12 of 32 characters shown],19) at line 4514 column 13 is invalid.
26/03/2014 10.11.23 inValue=1711447883 26MAR14:10:11:23 fmtname=NLDATM
26/03/2014 10.11.23 inValue=1711447883 26MAR14:10:11:23 fmtname=NLDATMAP
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).
      181 at 4514:13
NOTE: There were 284 observations read from the data set SASHELP.VFORMAT.
      WHERE (fmtType='I') and (fmtName not =: '$') and (fmtName not = 'PCPIB');
NOTE: The data set WORK.FOUR_DT has 4 observations
```

Output 3. Log from Format Hunter seeking INFORMAT

Since the results in the log are overcrowded by messages for the testing of unsuccessful informats (that I cannot find a way to suppress), here is a view of the table of effective informats:

VIEWTABLE: WORK.FOUR_DT

	string	Want1	fmtname	fmttype	source	minw	mind	maxw	maxd	defw	defd	inValue
1	26/03/2014 10.11.23	1711447883	ANYDTDTM	I	U	1	0	60	0	19	0	1711447883
2	26/03/2014 10.11.23	1711447883	FINDFDT	I	U	13	0	40	39	18	0	1711447883
3	26/03/2014 10.11.23	1711447883	NLDATM	I	U	19	0	200	0	19	0	1711447883
4	26/03/2014 10.11.23	1711447883	NLDATMAP	I	U	17	0	200	0	32	0	1711447883

Output 4. Table of results from Format Hunter seeking DateTime INFORMAT

DEMO 5

To show PROC SQL techniques with VFORMAT.

Within PROC SQL the metadata of formats can be addressed directly as the dictionary table DICTIONARY.FORMATS

```
%let fmt_test = '9:9:9't ;        /* value to be formatted */
%let wanted   = 090909   ;        /* result desired        */

PROC SQL ;
CREATE VIEW selected_formats as
 SELECT f.*
      , PUTN( &fmt_test, fmtname !! '-L' ) AS formatted
   FROM dictionary.formats          f
  WHERE fmtType        = 'F'
    AND fmtName NOT LIKE '$%'
/*  and calculated formatted =  "&wanted"*/
      ;
QUIT ;

PROC PRINT DATA= selected_formats ;
WHERE  formatted =  "&wanted"
;*     putn( datetime(), fmtname ) like '%Wed%'
   and putn( datetime(), fmtname ) like '%26%'
   and putn( datetime(), fmtname ) like '%11%'
   ;
RUN ;
```

The results window shows

Obs	libname	memname	path	objname	fmtname	fmttype	source	minw	mind	maxw	maxd	defw	defd	formatted
17			C:\Program Files\SASHome2\SASFoundation\9.4\core\sasexe	UWFB8601	B8601TM	F	U	6	0	15	6	8	0	090909

Output 4. Results from SQL Format Hunter

ISSUES DISCOVERED

A defect discovered in early use of this technique when seeking an informat, is (at Feb2014) still to be corrected - so exclude informat PCPIB (see examples above).

The default length for an informat might not be wide enough to use all characters of the input string.

Varying the length for a format might alter its appearance. For example the DATE format with default length presents differently when wider lengths are specified, like DATE9. and DATE11.

There are too many things to consider so I don't enclose it all in a macro or one complete SQL view..

Constructing an SQL view (like DEMO5 above) might provide a re-useable shortcut for trialing a subset of formats.

CONCLUSION

As these examples demonstrate SASHELP.VFORMAT provides the list of formats which SAS programmers can use to test each format to find the one most suitable for our needs, with the functions PUTN and INPUTN.

APPENDIX 1

ACKNOWLEDGMENTS

Phil Holland for first publishing this idea in the Views Newsletter [5]

_DATA_NULL_; (John King) for modestly indicating on SAS programmers at even the highest skill-levels do not command full knowledge of all formats (See the link above at SOLUTION PART 4 – using a variable format from the list)

RECOMMENDED READING

Documentation listed in References

- *TS-486: Quick Reference Sheet for SAS Functions, Formats, and Informats*
  http://support.sas.com/techsup/technote/ts486.pdf is no longer maintained by SAS Institute
- the collaborative effort to maintain a current version of TS486 *at*
  http://www.sascommunity.org/wiki/TS_486_Functions,_Informats,_and_Formats

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Peter Crawford
Organization:	Crawford Software Consultancy Limited
City, State ZIP:	London, UK, CR0 7HS
Work Phone:	+44(0)7802732254
Email:	Peter.Crawford@blueyonder.co.uk
sascommunity.org:	*http://www.sascommunity.org/wiki/User:Peter.Crawford*

REFERENCES

A list of documentation describing formats:

1. SAS Institute Inc. 2013. SAS® 9.4 Formats and Informats: Reference. Cary, NC: SAS Institute Inc.
2. SAS Institute Inc. 2013. SAS® 9.4 National Language Support (NLS): Reference Guide, Second Edition. Cary, NC: SAS Institute Inc.
3. SAS Institute Inc. 2013. SAS® 9.4 DS2 Language Reference, Second Edition. Cary, NC: SAS Institute Inc.
4. SAS Institute Inc. 2013. SAS® 9.4 FedSQL Language Reference, Second Edition. Cary, NC: SAS Institute Inc.

Copyright 2013, SAS Institute Inc., Cary, NC, USA. All Rights Reserved. Reproduced with permission of SAS Institute Inc., Cary, NC.

Other references to the technique

5. Crawford, Peter,2006,"Formats, Options and Functions", "Views Newsletter 35, Hints and Tips for SAS® Users", page 4, available at http://www.sascommunity.org/mwiki/images/6/69/VIEWS_News_Issue35.pdf

References to SASHELP.VFORMAT on SAS-L

- http://listserv.uga.edu/cgi-bin/wa?A2=ind0705B&L=sas-l&P=R36535
- http://listserv.uga.edu/cgi-bin/wa?A2=ind0903B&L=sas-l&P=R22347

Reference to TS:486 – ongoing support in www.sascommunity.org

- http://www.sascommunity.org/wiki/TS_486_Functions,_Informats,_and_Formats

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.