Paper 072-2011

Special Missing Values for Character Fields

John Ladds, Statistics Canada, Ottawa, Ontario, Canada

ABSTRACT

In SAS® everyone knows about missing values for character and numeric variables and some people know about special missing values that allow you to have more than one meaning for a missing value on numeric variables. What about character variables? Can we mimic the same behaviour? This paper will look at special missing values and a technique to achieve the same results for character variables using non-printable hex codes.

INTRODUCTION

If there is no data stored in a variable, SAS defines that as a missing value. There are three types of missing values: numeric, special numeric and character.

Numeric missing values are represented as a single period (.).

Character missing values are represented as a blank ( ).

Numeric special missing values allow for different types of missing data, represented by a single period plus a letter or underscore (.a - .z or ._).

Before we begin to talk about special character missing values, we need to know about special numeric missing values and how you might use them.

In our work, we use four types of missing values:

Valid Skip, Don't know, Refused, Not stated.

For numeric missing values, we use .v .d .r and .n

The code to define the special missing values in our case is (Ref 01):

```
missing v d r n;
```

The missing statement tells SAS to treat the listed letters (upper or lower case) as missing values in numeric fields.

To make the missing values easier to use we can define them with variable names.

(Ref 02)

```
** Numeric missing values;
length VS DK NS RF 3;
VS    = .v;
DK    = .d;
RF    = .r;
NS    = .n;
```

We can use PROC FORMAT so that reports and tables are easier to read as well.

(Ref 03)

```
proc format;
   value missFMT
    .v = '*VS*'      /* Valid skip  */
    .d = '*DK*'      /* Don't know  */
    .r = '*RF*'      /* Refused     */
    .n = '*NS*'      /* Not stated  */
    .  = '*UN*';     /* Unspecified */
run;
```

Using asterisks * only helps to identify missing values in reports and tables.

What about character data? SAS provides only one type of missing value, a blank. If we want to represent different types of missing values, we run the risk of someone using the phrase or code we have chosen. All is not lost; character data is stored as ASCII code in SAS. Each character in a field is represented by special codes and not the letters and numbers that we see. On modern PCs, we use the two digit ASCII code. Numbers, special characters, upper and lower case letters use only a few of the 127 possible codes. The non-printable codes give us an opportunity to mimic "numeric special missing" values.

Non-Printable ASCII Codes 0-1F Hex

Dec	Hex	Oct	Char	
0	0	000	NUL	Null
1	1	001	SOH	start of heading
2	2	002	STX	start of text
3	3	003	ETX	end of text
4	4	004	EOT	end of transmission
5	5	005	ENQ	Enquiry
6	6	006	ACK	Acknowledge
7	7	007	BEL	Bell
8	8	010	BS	Backspace
9	9	011	TAB	Horizontal tab
10	A	012	LF	NL line feed, new line
11	B	013	VT	vertical tab
12	C	014	FF	NP form feed, new page
13	D	015	CR	carriage return
14	E	016	SO	shift out
15	F	017	SI	shift in
16	10	020	DLE	data link escape
17	11	021	DC1	device control 1
18	12	022	DC2	device control 2
19	13	023	DC3	device control 3
20	14	024	DC4	device control 4
21	15	025	NAK	negative acknowledge
22	16	026	SYN	synchronous idle
23	17	027	ETB	end of transmission block
24	18	030	CAN	Cancel
25	19	031	EM	end of medium
26	1A	032	SUB	Substitute
27	1B	033	ESC	Escape
28	1C	034	FS	file separator
29	1D	035	GS	group separator
30	1E	036	RS	record separator
31	1F	037	US	unit separator

For character missing values, we use the ACSII codes '1C'x, '1D'x, '1E'x, '1F'x. Any non-printable ASCII code will

work or you can use the codes that are not normally found on a keyboard but produce different symbols. There are 31 codes from 00 hex to 1F hex which are available for use. The only concern is that the codes you select do not cause problems with other routines or programs that your data will encounter. In some programs, a special symbol may indicate that there is a non-printable character in the field.

The character missing value codes are less user-friendly than the numeric special values but by using hard coded variable names just as we did with numeric special missing values that can be easily solved. (Ref 04)

```
** Character values;
length VSc DKc NSc RFc $1;
VSc = '1C'x;
DKc = '1D'x;
RFc = '1E'x;
NSc = '1F'x;
drop VSc DKc NSc RFc;
```

Using the variable is cleaner and easier to read.

```
select(Q01);
   when(VS) Q04=VSc;
   when(DK) Q04=DKc;
   when(RF) Q04=RFc;
   when(NS) Q04=NSc;
   otherwise; /* no change */
end;
```

Again, just as we did for the numeric special missing values, we can use PROC FORMAT to assign value labels for reports and tables. (Ref 03)

```
proc format;
   ** Character values;
   value $missFMT (default=32767)
    '1C'x = '*VS*'  /* Valid skip  */
    '1D'x = '*DK*'  /* Don't know  */
    '1E'x = '*RF*'  /* Refused     */
    '1F'x = '*NS*'  /* Not stated  */
     ' '  = '*UN*'; /* Unspecified */
run;
```

The value statement takes the length of the longest formatted value. This causes the formatted value of undefined strings to be truncated. The option “(default=32767)” resets the default length to the longest possible string length for a SAS variable. The formats can also be use write text output to be used where non-printable hex codes are not appropriate. (Ref 05)

```
** Ref 05 Convert non-printable hex to text;
Q04_txt = put(Q04, $nullFmt.);
```

Value labels can be assigned to non-printable characters making a powerful tool for special missing values.

With Formats Off

Numeric Q01	Numeric Q02	Numeric Q03	Character Q04	Character Q04 text	Final Weight
V	2	2	_	*VS*	0.23
1	5	6	help	help	0.34
4	R	4	Good data	Good data	0.45
2	5	7	OK Data	OK Data	0.10
R	2	2	_	*RF*	0.29
N	5	5	_	*VS*	0.38
N	2	2	_	*VS*	0.47
4	5	9		*UN*	0.56
D	2	2	_	*DK*	0.65
4	5	9		*UN*	0.74
1	2	3		*UN*	0.83
D	5	5	_	*DK*	0.92
R	2	2	_	*RF*	0.01
4	5	9	Not Blank	Not Blank	0.12
0	0	0		*UN*	0.21

This is the same report using formats for the missing values.

With Formats On

Numeric Q01	Numeric Q02	Numeric Q03	Character Q04	Character Q04 text	Final Weight
VS	2	2	*VS*	*VS*	0.23
1	5	6	help	help	0.34
4	*RF*	4	Good data	Good data	0.45
2	5	7	OK Data	OK Data	0.10
RF	2	2	*RF*	*RF*	0.29
NS	5	5	*VS*	*VS*	0.38
NS	2	2	*VS*	*VS*	0.47
4	5	9	*UN*	*UN*	0.56
DK	2	2	*DK*	*DK*	0.65
4	5	9	*UN*	*UN*	0.74
1	2	3	*UN*	*UN*	0.83
DK	5	5	*DK*	*DK*	0.92
RF	2	2	*RF*	*RF*	0.01
4	5	9	Not Blank	Not Blank	0.12
0	0	0	*UN*	*UN*	0.21

CONCLUSION

As with any work around, there are always some caveats but this technique is a good method to add more meaning to ‘missing values’ especially character data that is, easier to code, harder to confuse with valid data and easier to visualize in reports and tables.

REFERENCES

SAS OnlineDoc 9.2
– Base SAS
– SAS 9.2 language Reference: Concepts
– Missing Values
SAS Publishing
SAS Campus Drive
Cary, NC 27513 USA
support.sas.com/bookstore
sasbook@sas.com
+1 800 727 3228

http://en.wikipedia.org/wiki/ASCII

IBM System / 370 Reference Summary GX20-1850-4
Fifth Edition (October 1981)
IBM Corporation, Technical Publications, Dept 824,
1133 Westchester Avenue, White Plains, N.Y. 10604

ACKNOWLEDGMENTS

Special thanks to Social Surveys Processing Environment IT team especially DJ Pieper for brain storming the problems around character missing values.

RECOMMENDED READING

SUGI 31 Paper 025-31
MISSING! – Understanding and Making the Most of Missing Data
Suzanne M. Humphreys, PRA International, Victoria, BC (Canada)

CONTACT INFORMATION

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

Name: John Ladds
Enterprise: Statistics Canada – SED SAS Technology Centre
Address: 14-R-90 R.H. Coats Building,
100 Tunney's Pasture Driveway
Ottawa, Ontario, Canada
K1A 0T6
Phone: 613 951-1767
E-mail: John.Ladds@statcan.gc.ca
Web: www.statcan.ca

EXAMPLE CODE

```
** Reserved Codes Using Special Missing Values ;

** Ref 01 Set the MISSING values codes ;
missing v d r n;

** Ref 03 Use  value labels to mask the printed output in reports;
proc format;
   value nullFMT
    .v = '*VS*'      /* 6 Valid skip  */
    .d = '*DK*'      /* 7 Don't know  */
    .r = '*RF*'      /* 8 Refused     */
    .n = '*NS*'      /* 9 Not stated  */
    .  = '*UN*'      /* . Unspecified */
    ;
   value $nullFMT (default=32767)
    '1c'x = '*VS*'  /* Valid skip  */
    '1d'x = '*DK*'  /* Don't know  */
    '1e'x = '*RF*'  /* Refused     */
    '1f'x = '*NS*'  /* Not stated  */
     ' '  = '*UN*'; /* Unspecified */
run;

** Example of one macro to check multiple lengths and types variables ;
%macro checkVar(Q);

   select(vtype(&Q.));
      ** Numeric data;
      when('N')
         select(&Q.);
            when(VS)     put 'WARNING: nN/A or Valid skip' id= &Q.=;
            when(DK)     put "WARNING: nDon't know "       id= &Q.=;
            when(RF)     put 'WARNING: nRefused '          id= &Q.=;
            when(NS)     put 'WARNING: nNot stated '       id= &Q.=;
            otherwise; /* Good Variable */
         end;
      ** Character data;
      when('C')
         select(&Q.);
            when(VSc)     put 'WARNING: cN/A or Valid skip' id= &Q.=;
            when(DKc)     put "WARNING: cDon't know "       id= &Q.=;
            when(RFc)     put 'WARNING: cRefused '          id= &Q.=;
            when(NSc)     put 'WARNING: cNot stated '       id= &Q.=;
            when(' ')     put 'WARNING: cBlank '            id= &Q.=;
            otherwise; /* Good Variable */
         end;
     otherwise put 'ERROR: Variable type problem ' &Q.;
  end;

   if missing(&Q.) then put;
%mend checkVar;

** Read test data with the special missing code already added          ;
** In real life the program would recode the raw codes to the correct ;
** missing value codes when the data was first read                    ;
data theData;
```

```
** Ref 02 Numeric values;
length VS DK NS RF 3;
VS     = .v;
DK     = .d;
RF     = .r;
NS     = .n;
drop VS DK NS RF;

** Ref 04 Character values;
length VSc DKc NSc RFc $1;
VSc = '1c'x;
DKc = '1d'x;
RFc = '1e'x;
NSc = '1f'x;
drop VSc DKc NSc RFc;

input @01 id   $02.
      @03 Q01    1.
      @04 Q02    2.
      @06 Q03    3.
      @09 Q04  $11.
      @20 WTS    4.2;

length Q04_txt $ 11;

** Add variable and value labels;
label ID      ='Unique Identifier'
      Q01     ='Numeric Q01'
      Q02     ='Numeric Q02'
      Q03     ='Numeric Q03'
      Q04     ='Character Q04'
      Q04_txt ='Character Q04 text'
      wts     ='Final Weight';

format Q01-Q03 nullFMT. Q04 $nullFMT.;

select(Q01);
   when(VS) Q04=VSc;
   when(DK) Q04=DKc;
   when(RF) Q04=RFc;
   when(NS) Q04=NSc;
   otherwise; /* no change */
end;

** Generic macro for numeric or character checking;
%checkVar(Q01);
%checkVar(Q02);
%checkVar(Q03);
%checkVar(Q04);

** Use mnemonics to check variable contents and set values;
if Q01 = NS
   then do;
      Q03 = VS;
      Q04 = VSc;
   end;
```

```
   if missing(Q01) or missing(Q02)
      then Q03 = VS;
      else Q03 = Q01 * Q02;

   ** Summing variables with missing;
   Q03 = Q01 + Q02;
   Q03 = sum(Q01,Q02);

   ** Ref 05 Convert non-printable hex to text;
   Q04_txt = put(Q04, $nullFmt.);

datalines4;
01v 2  3           0.23
021 5  6help       0.34
034 r  3Good data  0.45
042 5  nOK Data    0.10
05r 2  3           0.29
06n 5  6           0.38
07n 2  3Other data 0.47
084 5  6           0.56
09d 2  r           0.65
104 5  6           0.74
111 2  3           0.83
12d 5  6           0.92
13r 2  v           0.01
144 5  6Not Blank  0.12
150 0  0           0.21
;;;;
run;

title1 'Without Special Missing Values';

data theData2;
   set theData;
   if missing(q01) then Q01 = .;
   if missing(q02) then Q02 = .;
   if missing(q03) then Q03 = .;
   if q04 in ('1c'x,'1d'x,'1e'x,'1f'x) then Q04 = ' ';
   drop Q04_txt;
run;

proc print data=thedata2;
   var Q01-Q04 wts;
   format _all_;
run;

title1 'With Formats Off';
proc print data=thedata label noobs;
   var Q01-Q04 Q04_txt wts;
   format _all_;
run;

title1 'With Formats On';
proc print data=thedata label noobs;
   var Q01-Q04 Q04_txt wts;
run;
```