Advanced Uses of User-defined Formats and Informats

Dr. Ron Cody

Topics to be Covered

- *Using formats with a PUT function to create new variables*
- *Creating user-defined INFORMATS*
- *Enhanced numeric informat – Reading character and numeric data in one step*
- *Using formats (and informats) to perform a table look-up*
- *Using a SAS data set to create a format: CNTLIN*
- *Updating and maintaining your formats: CNTLOUT*

Topics (continued)

- ***Using formats within formats: Nesting formats***
- ***Multilabel formats***
- ***Using the INPUTN function to perform a more complicated table look-up***
- ***FMTSEARCH and FMTERR system options***

Reviewing the PUT Function

Syntax: PUT (*variable*, *format*)

Where *variable* is a character or numeric variable and *format* is a SAS or user-defined format.

The result of a PUT function is always a character value

Examples:
Date = 1, PUT(Date, mmddyy10.) is “01/02/1960”
Cost = 12345, PUT(Cost, dollar8.) is “$12,345.”

Note; The storage length of the resulting variable will be the length of the longest formatted value.

Using formats with a PUT function to create new variables

```
proc format;
   value agefmt  0 - <20  = '< 20'
                20 - <40  = '20 to 39'
                40 - <60  = '40 to 59'
                60 - high = '60+';
run;

data survey;
   set learn.survey;
   AgeGroup = put(Age,agefmt.);
Run;
```

Output

Listing of SURVEY

ID	Gender	Age	Salary	Ques1	Ques2	Ques3	Ques4	Ques5	AgeGroup
001	M	23	28000	1	2	1	2	3	20 to 39
002	F	55	76123	4	5	2	1	1	40 to 59
003	M	38	36500	2	2	2	2	1	20 to 39
004	F	67	128000	5	3	2	2	4	60+
005	M	22	23060	3	3	3	4	2	20 to 39
006	M	63	90000	2	3	5	4	3	60+
007	F	45	76100	5	3	4	3	3	40 to 59

Creating user-defined INFORMATS

```
proc format;
   invalue convert 'A+' = 100 'A'  = 96
                   'A-' = 92  'B+' = 88
                   'B'  = 84  'B-' = 80
                   'C+' = 76  'C'  = 72
                   'F'  = 65;
run;

data grades;
   input @1 ID           $3.
         @4 Grade convert2.;
datalines;
001A-
002B+
003F
004C+
005A
;
```

number of columns to read

Listing of GRADES

ID	Grade
001	92
002	88
003	65
004	76
005	96

INFORMAT options UPCASE and JUST

```
proc format;
      invalue convert(upcase just)
            'A+' = 100   'A'  = 96
            'A-' = 92    'B+' = 88
            'B'  = 84    'B-' = 80
            'C+' = 76    'C'  = 72
            'F'  = 65     other  =  .;
run;
data grades;
   input @1 ID           $3.
         @4 Grade convert2.;
datalines;
001A-
002b+
003F
004c+
005 A
006X
;
```

Listing of GRADES

ID	Grade
001	92
002	88
003	65
004	76
005	96
006	.

Reviewing the INPUT Function

Syntax: INPUT (*char_val, informat*)
Where *char_val* is a SAS character value and *informat* is a SAS or a user-defined informat

Note: the *informat* must be an informat name followed by a period—it cannot be a character constant, variable, or expression

Example: C_Date = “01/02/1960”
INPUT(C_Date, mmddyy10.) is 1

A Traditional Approach to Reading Character and Numeric Values

```
data temperatures;
    input Dummy $ @@;
    if upcase(Dummy) = 'N' then Temp = 98.6;
    else Temp = input(Dummy,8.);
    drop dummy;
datalines;
101 N 97.3 n N 104.5
;
```

```
Listing of
TEMPERATURES

 Temp

101.0
 98.6
 97.3
 98.6
 98.6
104.5
```

Using an Enhanced Numeric Informat

```
proc format;
   invalue readtemp(upcase)
                  96 - 106 = _same_
                  'N'      = 98.6
                  other    = .;
run;
data temperatures;
   input Temp : readtemp. @@;
datalines;
101 N 97.3 n N 67 104.5
;
```

```
Listing of
TEMPERATURES

 Temp

101.0
 98.6
 97.3
 98.6
 98.6
   .
104.5
```

Another Example of an Enhanced Numeric Informat

```
proc format;
   invalue readgrade(upcase)
      'A' = 95   'B' = 85
      'C' = 75   'F' = 65
       0-100 = _same_
      other = .;
run;

data school;
   input Grade : readgrade. @@;
datalines;
97 99 A C 72 f b 101
;
```

```
Listing of
SCHOOL

Grade

  97
  99
  95
  75
  72
  65
  85
   .
```

Using Formats and Informats to Perform a Table Lookup

```
proc format;
  value namelookup
    122 = 'Salt'
    188 = 'Sugar'
    101 = 'Cereal'
    755 = 'Eggs'
  other = ' ';
  invalue pricelookup
    'Salt'   = 3.76
    'Sugar'  = 4.99
    'Cereal' = 5.97
    'Eggs'   = 2.65
     other   = .;
run;
```

```
data grocery;
  input ItemNumber @@;
  Name = put(ItemNumber,namelookup.);
  Price = input(Name,pricelookup.);
datalines;
101 755 122 188 999 755
;
```

Listing of GROCERY

Item Number	Name	Price
101	Cereal	5.97
755	Eggs	2.65
122	Salt	3.76
188	Sugar	4.99
999		.
755	Eggs	2.65

Creating a Format from a SAS Data Set

Listing of Data Set LEARN.CODES

ICD9	Description
020	Plague
022	Anthrax
390	Rheumatic fever
410	Myocardial infarction
493	Asthma
540	Appendicitis

Variables in a Control Data Set

Variable Name	Description
FMTNAME	Name of the format or informat
TYPE	“C” for character formats, “N” for numeric, "I" for Informat (optional)
START	Starting value for a range or a unique value
END	The ending value (if there is one)
LABEL	The format or informat label
HLO	Provides a High, Low, or Other value (optional)

Creating a Control Data Set

```
data control;
   set learn.codes(rename=
                     (ICD9 = Start
                      Description = Label));
   retain Fmtname '$ICDFMT'
          Type 'C';
run;

title "Demonstrating an Input Control Data Set";
proc format cntlin=control;
   select $ICDFMT;
run;
```

Note: The PROC FORMAT option FMTLIB will display all the formats in the library listed in the LIBRARY= option.

Creating a Control Data Set

Listing of CONTROL

Start	Label	Fmtname	Type
020	Plague	$ICDFMT	C
022	Anthrax	$ICDFMT	C
390	Rheumatic fever	$ICDFMT	C
410	Myocardial infarction	$ICDFMT	C
493	Asthma	$ICDFMT	C
540	Appendicitis	$ICDFMT	C

Result of the SELECT Statement

The $ICD9 Format

FORMAT NAME: $ICDFMT LENGTH: 21 NUMBER OF VALUES: 6
MIN LENGTH: 1 MAX LENGTH: 40 DEFAULT LENGTH 21 FUZZ: 0

START	END	LABEL (VER. V7|V8 12DEC2007:10:31:39)
020	020	Plague
022	022	Anthrax
390	390	Rheumatic fever
410	410	Myocardial infarction
493	493	Asthma
540	540	Appendicitis

Using the Format

```
data disease;
   input ICD9 : $5. @@;
datalines;
020 410 500 493
;
title "Listing of DISEASE";
proc print data=disease noobs;
   format ICD9 $ICDFMT.;
run;
```

```
Listing of DISEASE

ICD9

Plague
Myocardial infarction
500
Asthma
```

Using PROC REPORT to Display Both the Formatted and Unformatted values

```
title "Listing of DISEASE ";
proc report data=disease nowd;
    columns ICD9 ICD9=temp;
    define ICD9 / "ICD Code";
    define temp / "Description"
                  format = $ICDFMT.;
run;
```

Listing of DISEASE

ICD Code	Description
020	Plague
410	Myocardial infarction
500	500
493	Asthma

Adding the OTHER Category

```
data control / view=control;
    set learn.codes(rename=
                    (ICD9 = Start
                    Description = Label))
                    end = last;
    retain Fmtname '$ICDFMT'
           Type 'C';
    output;
    if last then do;
       Start = ' ';
       Hlo = 'o';
       Label = 'Not Found';
       output;
    end;
run;
```

optional

Adding the Other category

Adding the OTHER Category (cont.)

```
proc format cntlin=control;
    select $ICDFMT;
run;
```

Adding OTHER Category

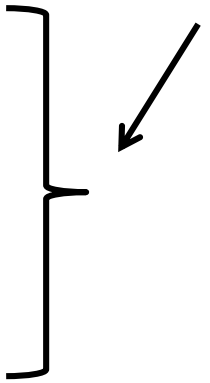
FORMAT NAME: $ICDFMT LENGTH: 21 NUMBER OF VALUES: 7
MIN LENGTH: 1 MAX LENGTH: 40 DEFAULT LENGTH 21 FUZZ: 0

START	END	LABEL (VER. V7|V8 24MAR2008:15:33:55)
020	020	Plague
022	022	Anthrax
390	390	Rheumatic fever
410	410	Myocardial infarction
493	493	Asthma
540	540	Appendicitis
OTHER	**OTHER**	Not Found

Modifying an Existing Format

```
proc format cntlout=control_out;
    select $ICDFMT;
Run;
data new_control;
    length Label $ 21;
    set control_out end=Last;
    output;
    if Last then do;
        Hlo = ' ';
        Start = '427.5';
        End = Start;
        Label = 'Cardiac Arrest';
        output;
                (cont)
```

Adding 427.5 As Cardiac Arrest



Modifying an Existing Format (cont)

```
Start = '466';
        End = Start;
        Label = 'Bronchitis';
        output;
    end;
Run;

proc format cntlin=new_control;
    select $ICDFMT;
Run;
```

Adding 466
As Bronchitis

Creating the
New Format

Demonstrating Nested Formats

```
proc format;
  value registration low - <'15Jul2005'd = 'Not Open'
          '15Jul2005'd - '31Dec2006'd = [mmddyy10.]
          '01Jan2007'd - high         = 'Too Late';
run;

data conference;
   input @1  Name $10.
         @11 Date mmddyy10.;
   format Date registration.;
datalines;
Smith      10/21/2005
Jones      06/13/2005
Harris     01/03/2007
Arnold     09/12/2005
;
```

Listing of CONFERENCE

Name	Date
Smith	10/21/2005
Jones	Not Open
Harris	Too Late
Arnold	09/12/2005

Another Example of Nested Formats

```
proc format;
    value $icdfmt_temp
      '401' = 'Hypertension'
      '480' = 'Viral Pneumonia'
      other = [$icdfmt.];
run;
```

Creating a Multi-label Format

```
proc format;
    value agegroup (multilabel)
     0 - <20    = '0 to <20'
     20 - <40   = '20 to <40'
     40 - <60   = '40 to <60'
     60 - <80   = '60 to <80'
     80 - high = '80 +'

     0 - <50    = 'Less than 50'
     50 - high = '> or = to 50';
run;
```

Using Multi-label Format

```
title "Demonstrating a Multilabel Format";
title2 "PROC MEANS Example";
proc means data=learn.survey noobs mean;
    class Age / MLF;
    var Salary;
    format Age agegroup.;
run;
```

Demonstrating a Multilabel Format
PROC MEANS Example

The MEANS Procedure

Analysis Variable : Salary

Age	Mean
20 to <40	29186.67
40 to <60	76111.50
60 to <80	109000.00
> or = to 50	98041.00
Less than 50	40915.00

Using the PRINTMISS, and MISSTEXT options with PROC TABULATE

```
title "Multilabel Format - Tabulate";
proc tabulate data=learn.survey format=5.;
   class Age Gender / MLF;
   table Age ,
         Gender*N=' ' / printmiss  misstext=' ';
   format Age agegroup.;
run;
quit;
```

Using the PRINTMISS, and MISSTEXT options with PROC TABULATE

Multilabel Format - Tabulate

	Gender	
	F	M
Age		
20 to <40		3
40 to <60	2	
60 to <80	1	1
> or = to 50	2	1
Less than 50	1	3

Using the PRELOADFMT, PRINTMISS, and MISSTEXT options with PROC TABULATE

```
title "Demonstrating a Multilabel Format";
title2 "PROC TABULATE Example";
proc tabulate data=learn.survey format=5.;
   class Age Gender / MLF preloadfmt;
   table Age ,
      gender*n=' ' / printmiss  misstext=' ';
   format Age agegroup.;
run;
quit;
```

Note: PRELOAD only works with three Procs: MEANS (SUMMARY), TABULATE, and REPORT

Multilabel Formats with PROC TABULATE

Demonstrating a Multilabel Format
PROC TABULATE Example

	Gender	
	F	M
Age		
0 to <20		
20 to <40		3
40 to <60	2	
60 to <80	1	1
80 +		
> or = to 50	2	1
Less than 50	1	3

Reviewing the INPUTN Function

Syntax: INPUTN (*char_val*, *informat*)
Where *char_val* is a character value and *informat* is either a SAS or user-defined numeric informat. It may be created in a DATA Step.

Example:

```
Data dates;
    input Style $ C_Date : $10.;
    if Style = "Europe" then i = "ddmmyy10.";
    else if style = "American" then i = "mmddyy10.";
    Date = inputn(C_date,i);
    format Date date9.;
    keep Style C_Date Date;
Datalines;
Europe 5/6/2000
American 5/6/2000
;
```

```
Listing of DATES

 Style    C_Date        Date

Europe   5/6/2000 05JUN2000
American 5/6/2000 06MAY2000
```

Creating Several Informats with a Single Control Data Set

```
data exposure;
   retain Type 'I' Hlo 'U';
   do Year = 1944 to 1949;
      Fmtname = cats('exp',Year,'fmt');
      do Start = 'A','B','C','D','E';
         input Label : $3. @;
         output;
      end;
   end;
   drop Year;
datalines;
220   180   210   110   90
202   170   208   100   85
150   110   150    60   50
105    56    88    40   30
 60    30    40    20   10
 45    22    22    10    8
;
```

I specifies an informat

U specifies the UPCASE option

Listing of the Control Data Set

Listing of EXPOSURE Data Set

Type	Hlo	Fmtname	Start	Label
I	U	exp1944fmt	A	220
I	U	exp1944fmt	B	180
I	U	exp1944fmt	C	210
I	U	exp1944fmt	D	110
I	U	exp1944fmt	E	90
I	U	exp1945fmt	A	202
I	U	exp1945fmt	B	170
I	U	exp1945fmt	C	208
I	U	exp1945fmt	D	100
I	U	exp1945fmt	E	85

. . .

Creating the Informats

INFORMAT NAME: @EXP1944FMT LENGTH: 1 MIN LENGTH: 1 MAX LENGTH: 40 DEFAULT LENGTH 1 FUZZ: 0		
START	END	INVALUE(VER. 9.2 09NOV2007:20:19:35)
A	A	220
B	B	180
C	C	210
D	D	110
E	E	90

INFORMAT NAME: @EXP1945FMT LENGTH: 1 MIN LENGTH: 1 MAX LENGTH: 40 DEFAULT LENGTH 1 FUZZ: 0		
START	END	INVALUE(VER. 9.2 09NOV2007:20:19:35)
A	A	202
B	B	170
C	C	208
D	D	100
E	E	85

Using the Informat to Perform a Table Look-up

```
data read_exp;
   input Worker $ Year JobCode $;
   Exposure = inputn(JobCode,cats('exp',Year,'fmt8.'));
datalines;
001 1944 B
002 1948 E
003 1947 C
005 1945 A
006 1948 D
;
```

Listing of READ_EXP

Worker	Year	Job Code	Exposure
001	1944	B	180
002	1948	E	10
003	1947	C	88
005	1945	A	202
006	1948	D	20

1st Obs:
Year = 1944
Exposure = inputn(B,exp1944fmt8.);

Two Useful System Options

```
Libname myfmts 'c:\sasdata\formats';
Options FMTSEARCH = (myfmts);
```

The location of your format library

Search order: Work, Library, myfmts

To change the order:

```
Options FMTSEARCH = (myfmts work library);
```

Search order: myfmts, Work, Library

If you have a SAS data set with permanent formats, but do not have the format library, use:

```
Options NOFMTERR;
```

Contact Information

Name: Ron Cody
email: ron.cody@gmail.com

More information on advanced uses of formats and informats can be found in:

Learning SAS by Example: A Programmer's Guide, published by SAS Press