Paper 959-2017

Formats are your Friends

Anita Measey, Bank of Montreal Canada; Lei Sun; Bank of Montreal, USA

ABSTRACT

SAS® formats can be used for more than just making your data look nice. Formats can be used as in memory look up tables as well as help you to create data driven code. This paper will show you, with code examples, how to build a format from a dataset; write a format out as a dataset; and how to use formats to help to make programs data driven.

INTRODUCTION

Formats (and informats) are stored in SAS catalogs – a SAS catalog is a special SAS file that contains catalog entries, while formats are a type of catalog entry. Formats can be stored in a permanent library, if for example they are being shared across an organization or in the work library if they are only needed during the execution of a single process or job. A process or job can point to multiple libraries containing formats and the user can specify the order of libraries that SAS should search for the formats using the FMTSEARCH = System Option. Informats are typically used as a way to instruct SAS on how to read data into SAS variables. Formats are typically used as instruction for outputting data. This paper demonstrates how formats can be created from datasets; how they can be viewed as dataset; rules around creating formats; using formats as lookup tables and how formats can help you build data driven processes.

CREATING FORMATS FROM DATASETS

A format dataset must contain, at a minimum, format name, start value, label and format type:

- Format name is the name of the format
- Start is the value in your data
- Label is the value you want to retrieve to assign to your data
- Type determines format/informat, character/numeric

In the example below the dataset contains gender and gender description. To create a format that converts the gender mnemonic into a description, the data needs to be formatted in a way that SAS can recognize and write to the format catalog.

1. Create some data:

```
DATA indata;
  FORMAT Gender $4. Gender_DSC $30.;
  INFILE DATALINES delimiter=',';
  INPUT Gender $ Gender_DSC $;
  DATALINES;
  F,Female
  M,Male
  ;
RUN;
```

	Gender	Gender_DSC
1	F	Female
2	M	Male

2. **Format the data to be uploaded to the catalog:**

```
DATA fmtdata(keep=fmtname start label type);
  SET indata;
      fmtname = 'Gender';
      type = 'C';
      start=Gender;
      label=Gender_DSC;
RUN;
```

	fmtname	type	start	label
1	Gender	C	F	Female
2	Gender	C	M	Male

3. **Write the format to the catalog:**

```
PROC FORMAT LIB=work CNTLIN=fmtdata;
RUN;
```

Multiple formats can be created in a single dataset; in this case the dataset must be sorted by the fmtname value prior to writing to the catalog.

This simple format example converts a single value to a description, formats containing ranges, where multiple values resolve to a single value, can be controlled with the additional variables:

- END (used with start when a range is used)
- SEXCL (Start Exclude – useful when setting ranges)
- EEXCL (End Exclude – useful when setting ranges)
- HLO (High Low Other – useful when setting extremes and defaults)
  – see Table 1 in Appendix for full list of output control data set variables.

WRITE A FORMAT OUT TO A DATASET

Format or formats can be selected from a SAS catalogue and written out to a dataset called a control dataset.

1. **Select a character format called “gender” and write it out to a dataset called fmtout:**

```
PROC FORMAT LIB=work CNTLOUT=fmtout;
  SELECT $Gender;
RUN;
```

	FMTNAME	START	END	LABEL	TYPE	SEXCL	EEXCL	HLO	MIN	MAX
1	GENDER	F	F	Female	C	N	N		1	
2	GENDER	M	M	Male	C	N	N		1	

```
22  NOTE: The data set WORK.FMTOUT has 2 observations and 21 variables.
23  NOTE: Compressing data set WORK.FMTOUT increased size by 100.00 percent.
24        Compressed is 2 pages; un-compressed would require 1 pages.
```

RULES ABOUT FORMATS

- Formats cannot start or end with a number
- Format names can be up to 32 characters long.
- Character formats must start with a dollar sign ($) - this counts towards the 32 characters.
- You cannot create a format name with the same name as a SAS supplied format

USING FORMATS AS LOOKUP TABLES

Scenario: Our data includes cumulative scores for a class of students; we want to weight the scores by a factor that depends on age.

1. **Create some test data that contains score:**

```
DATA Indata(DROP=I seed Height Weight);
  SET sashelp.class;
      FORMAT score 8.2;
      RETAIN seed 1298573062;
    DO I = 1 TO 100;
      SCORE=100.00*I* RANUNI(seed);
      OUTPUT;
    END;
  RUN;
```

INDATA

Filter and Sort | Query Builder | Where | Data | Describe

	Name	Sex	Age	score
1	Alfred	M	14	23.70
2	Alfred	M	14	55.44
3	Alfred	M	14	64.92
4	Alfred	M	14	74.02
5	Alfred	M	14	82.09
6	Alfred	M	14	126.14
7	Alfred	M	14	134.08
8	Alfred	M	14	164.27
9	Alfred	M	14	178.94
10	Alfred	M	14	186.61
11	Alfred	M	14	243.19
12	Alfred	M	14	269.76

2. **Create the factor table that maps the age range to the weight:**

```
DATA Factor;
   FORMAT START 8. END 8. FACTOR 8.2;
   INFILE DATALINES delimiter =',';
   INPUT start end factor;
    DATALINES;
     , 11, 1.5
   11, 12, 1.4
   12, 13, 1.3
   13, 14, 1.2
   14, 15, 1.1
   15,   , 1.0
   ;
RUN;
```

FACTOR

Filter and Sort | Query Builder | Where

	START	END	FACTOR
1	.	11	1.50
2	11	12	1.40
3	12	13	1.30
4	13	14	1.20
5	14	15	1.10
6	15	.	1.00

3. Create a format from the factor table:

```
DATA fmtchar(KEEP=fmtname start end label hlo type);
  LENGTH fmtname $30 start end label 8;
 SET Factor;
 RETAIN type 'I';
 fmtname = 'FACTOR';
   start = start;
     end = end;
   label = FACTOR;
 IF start = . THEN hlo = 'L';
 ELSE hlo = '';
 IF end = . THEN hlo = 'H';
RUN;
```

FMTCHAR

Filter and Sort | Query Builder | Where | Data | Describe | Graph | Analyze | Export

	fmtname	start	end	label	type	hlo
1	FACTOR	.	11	1.5	i	L
2	FACTOR	11	12	1.4	i	
3	FACTOR	12	13	1.3	i	
4	FACTOR	13	14	1.2	i	
5	FACTOR	14	15	1.1	i	
6	FACTOR	15	.	1	i	H

```
PROC FORMAT LIB=WORK CNTLIN=fmtchar;
RUN;
```

4. Apply the format to the data:

```
DATA new_score;
  SET INDATA;
   ** apply the format to the age to determine the FACTOR **;
   FACTOR        = INPUT(age,FACTOR.);
   ** apply the FACTOR to the score **;
   new_SCORE     = SUM(score*FACTOR);
RUN;
```

OR

```
DATA new_score_alt;
  SET INDATA;
   ** apply the format to the age to determine the FACTOR then **;
   ** apply the FACTOR to the score **;
   new_SCORE     = SUM(score*INPUT(age,FACTOR.));
RUN;
```

NEW_SCORE

Filter and Sort | Query Builder | Where | Data | Describe | Graph | Analyze | Export | Se

	Name	Sex	Age	score	FACTOR	new_SCORE
1	Alfred	M	14	23.70	1.2	28.438832997
2	Alfred	M	14	55.44	1.2	66.527602704
3	Alfred	M	14	64.92	1.2	77.908759489
4	Alfred	M	14	74.02	1.2	88.829244137
5	Alfred	M	14	82.09	1.2	98.506689192
6	Alfred	M	14	126.14	1.2	151.36973274
7	Alfred	M	14	134.08	1.2	160.89211829

USING YOUR DATA TO DRIVE THE LOOKUP

Scenario: Our data includes cumulative scores for a class of students; we want to weight the scores by a factor that depends on age and gender.

1. Create a factor table based on age, a different factor for each gender:

```
DATA Factor2;
  FORMAT start 8. end 8. factor_F 8.2 factor_M 8.2;
  INFILE DATALINES delimiter =',';
INPUT start end factor_F factor_M;
DATALINES;
  , 11, 1.5, 1.6
11, 12, 1.4, 1.5
12, 13, 1.3, 1.4
13, 14, 1.2, 1.3
14, 15, 1.1, 1.2
15,   , 1.0, 1.1
;
RUN;
```

FACTOR2

Filter and Sort | Query Builder | Where | Data | Describe

	START	END	FACTOR_F	FACTOR_M
1	.	11	1.50	1.60
2	11	12	1.40	1.50
3	12	13	1.30	1.40
4	13	14	1.20	1.30
5	14	15	1.10	1.20
6	15	.	1.00	1.10

2. Create formats from the factor table for each gender

```
DATA fmtchar(KEEP=fmtname start end label hlo type);
  LENGTH fmtname $30 start end label 8;
 SET Factor2;
 RETAIN type 'i';
 fmtname = FACTOR_typ;
   start = start;
     end = end;
   label = FACTOR;
 IF start = . THEN hlo = 'L';
 ELSE hlo = '';
 IF end = . THEN hlo = 'H';
RUN;
```

FMTCHAR

Filter and Sort | Query Builder | Where | Data | Describe | Graph | Analyze | Export

	fmtname	start	end	label	type	hlo
1	FACTOR_F	.	11	1.5	i	L
2	FACTOR_F	11	12	1.4	i	
3	FACTOR_F	12	13	1.3	i	
4	FACTOR_F	13	14	1.2	i	
5	FACTOR_F	14	15	1.1	i	
6	FACTOR_F	15	.	1	i	H
7	FACTOR_M	.	11	1.6	i	L
8	FACTOR_M	11	12	1.5	i	
9	FACTOR_M	12	13	1.4	i	
10	FACTOR_M	13	14	1.3	i	
11	FACTOR_M	14	15	1.2	i	
12	FACTOR_M	15	.	1.1	i	H

```
PROC FORMAT LIB=WORK CNTLIN=fmtchar;
RUN;
```

3. Create some test data:

```
DATA indata(DROP=I seed Height Weight);
SET sashelp.class;
      FORMAT score 8.2;
      RETAIN seed 1298573062;
DO I = 1 TO 100;
  SCORE=100.00*I* RANUNI(seed);
  OUTPUT;
END;
RUN;
```

INDATA

Filter and Sort | Query Builder | Where | Data | Describe

	Name	Sex	Age	score
94	Alfred	M	14	7189.08
95	Alfred	M	14	7243.59
96	Alfred	M	14	7688.02
97	Alfred	M	14	8096.89
98	Alfred	M	14	8187.96
99	Alfred	M	14	8426.79
100	Alfred	M	14	8815.65
101	Alice	F	13	37.55
102	Alice	F	13	44.35
103	Alice	F	13	52.33
104	Alice	F	13	75.65
105	Alice	F	13	83.18

4. Apply the format to the data:

```
DATA NEW_SCORE2;
 SET indata;
   ** figure out the format we need to use to resolve the SCORE **;
   FACTOR_TYPE    = UPCASE(compress('FACTOR_'||sex));
   ** apply the format to the SCORE to determine the FACTOR **;
   FACTOR         = INPUTN(age,FACTOR_type);
   ** apply the FACTOR to the SCORE **;
   new_bal      = SUM(SCORE*FACTOR);
RUN;
```

NEW_SCORE2

Filter and Sort | Query Builder | Where | Data | Describe | Graph | Analyze | Export | Send To

	Name	Sex	Age	score	FACTOR_T...	FACTOR	new_bal
94	Alfred	M	14	7189.08	FACTOR_M	1.3	9345.8005728
95	Alfred	M	14	7243.59	FACTOR_M	1.3	9416.6612717
96	Alfred	M	14	7688.02	FACTOR_M	1.3	9994.4237248
97	Alfred	M	14	8096.89	FACTOR_M	1.3	10525.961157
98	Alfred	M	14	8187.96	FACTOR_M	1.3	10644.352121
99	Alfred	M	14	8426.79	FACTOR_M	1.3	10954.825431
100	Alfred	M	14	8815.65	FACTOR_M	1.3	11460.341581
101	Alice	F	13	37.55	FACTOR_F	1.3	48.811667771
102	Alice	F	13	44.35	FACTOR_F	1.3	57.658174661
103	Alice	F	13	52.33	FACTOR_F	1.3	68.034539254
104	Alice	F	13	75.65	FACTOR_F	1.3	98.350349748
105	Alice	F	13	83.18	FACTOR_F	1.3	108.13021955

OR

```
DATA NEW_SCORE_alt;
 SET indata;
   ** figure out the format we need to use to resolve the SCORE **;
   ** apply the format to the SCORE to determine the FACTOR **;
   ** apply the FACTOR to the SCORE **;
   new_bal = SUM(SCORE*INPUTN(age,UPCASE(compress('FACTOR_'||sex))));
RUN;
```

CONCLUSION

Formats can be user defined and stored for permanent or temporary use; they can be used in the traditional way to tell SAS how to read in data or to what the data will look like when written out; or can be used as look up tables to help make programs more data driven.

REFERENCES

SAS® 9.4 Language Reference: Concepts, Sixth Edition, http://support.sas.com/documentation/cdl/en/lrcon/69852/HTML/default/viewer.htm#p01gryeviu67qwn1m9wue2t0k8sk.htm

Introduction to SAS Informats and Formats, http://support.sas.com/publishing/pubcat/chaps/59498.pdf

Base SAS® 9.4 Procedure Guide, Sixth Edition, Format Procedure, https://support.sas.com/documentation/cdl/en/proc/69850/HTML/default/viewer.htm#p0owa4ftikc2ekn1q0rmpulg86cx.htm

RECOMMENDED READING

- *Base SAS® Procedures Guide*
- *SAS® For Dummies®*

CONTACT INFORMATION

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

Anita Measey
Bank of Montreal
416-867-6728
Anita.Measey@bmo.com

APPENDIX

Table 1	**OUTPUT CONTROL DATA SET**
Variable	Description
DATATYPE	enables the use of directives in a picture as a template to format date, time, or datetime values
DECAP	specifies the separator character for the fractional part of a number
DEFAULT	specifies a numeric variable that indicates the default length for format or informat.
DIG3SEP	specifies the three-digit separator character for a number.
END	specifies a character variable that gives the range's ending value
EEXCL	specifies a character variable that indicates whether the range's ending value is excluded. Valid values are as follows:
	Y - specifies that the range's ending value is excluded.
	N - specifies that the range's ending value is not excluded.
FILL	for picture formats, specifies a numeric variable whose value is the value of the FILL= option.
FMTNAME	specifies a character variable whose value is the format or informat name.
FUZZ	specifies a numeric variable whose value is the value of the FUZZ= option.
HLO	specifies a character variable that contains range information about the format or informat. The following valid values can appear in any combination
	F - specifies a standard SAS format or informat that is used with a value
	H - specifies that a range's ending value is HIGH
	I - specifies a numeric informat range
	J - specifies justification for an informat
	L - specifies that a range's starting value is LOW
	M - specifies that the MULTILABEL option is in effect
	N - specifies that the format or informat has no ranges, including no OTHER= range
	O - specifies that the range is OTHER.
	R - specifies that the ROUND option is in effect
	S - specifies that the NOTSORTED option is in effect
	U - specifies that the UPCASE option for an informat be used
LABEL	specifies a character variable whose value is associated with a

	format or an informat
LANGUAGE	specifies the language that is used for weekdays and months that you can substitute in a date, time, or datetime picture. If you specify a language that is not supported or is invalid, English is used.
LENGTH	specifies a numeric variable whose value is the value of the LENGTH= option
MAX	specifies a numeric variable whose value is the value of the MAX= option
MIN	specifies a numeric variable whose value is the value of the MIN= option
MULT	specifies a numeric variable whose value is the value of the MULT= option.
NOEDIT	for picture formats, specifies a numeric variable whose value indicates whether the NOEDIT option is in effect. Valid values are as follows
	1 - specifies that the NOEDIT option is in effect.
	0 - specifies that the NOEDIT option is not in effect.
PREFIX	for picture formats, specifies a character variable whose value is the value of the PREFIX= option
SEXCL	specifies a character variable that indicates whether the range's starting value is excluded. Valid values are as follows:
	Y - specifies that the range's starting value is excluded
	N - specifies that the range's starting value is not excluded
START	specifies a character variable that gives the range's starting value
TYPE	specifies a character variable that indicates the type of format. Possible values are as follows
	C - specifies a character format
	I - specifies a numeric informat.
	J - specifies a character informat.
	N - specifies a numeric format (excluding pictures).
	P - specifies a picture format