

Formats as Date Range Lookup Functions

Brief Overview

- You can build your own format based on the contents of a dataset
- Your own format can be used (with the put function) in the data step to operate like a lookup function
- Formats can map ranges of values to a single result – this feature can be used to implement date ranges

Types of SAS Formats

- Built-in SAS formats
- User-created
 - Picture type – indicates pattern
 - Value type – converts value or range to an output value

Sample Proc Formats (Value type)

```
proc format;  
  value $pie  
    'A' = 'Apple'  
    'B' = 'Berry'  
    'C' = 'Cherry';  
run;
```

```
proc format;  
  value square  
    2 = '04'  
    4 = '16'  
    7 = '49';  
run;
```

Sample Proc Format with Ranges

```
proc format;  
  value $agerng  
    0      = 'Newborn'  
    1-4    = 'One to Four'  
    5-9    = 'Five to Nine'  
    10-14  = 'Ten to Fourteen';  
run;
```

Using a Format

- **When printing**

```
proc print data=my_pies;  
  format piecode $pie.;  
run;
```

- **In a data step using the put function**

```
data my_squares;  
  length square 8;  
  set my_numbers;  
  square = put(num, square.);  
run;
```

Saving Formats More Permanently

- **Libname statement**

```
libname mylib "c:\spot\run"; or  
libname mylib "/unix/dir/name";
```

- **Options fmtsearch**

```
options fmtsearch=(work mylib library);
```

CNTLIN and CNTLOUT

```
proc format lib=work cntlout=my_pie;  
  select $pie;  
run;
```

<u>FMTNAME</u>	<u>TYPE</u>	<u>START</u>	<u>END</u>	<u>LABEL</u>	<u>HLO</u>
PIE	C	A	A	Apple	
PIE	C	B	B	Berry	
PIE	C	C	C	Cherry	

```
proc format lib=work cntlin=my_pie;  
run;
```

Other values & HLO

HLO Variable:

- H – range includes highest value
- L – range includes lowest value
- O – Other range, for values that do not fit into any other range

Generating CNTLIN (dataset) from data in a Dataset

```
data pracnm_fmt;  
  set sasdata.pracds end=eof;  
  fmtname = 'pracnm';  
  type = 'N';  
  start = pracnum;  
  end = pracnum;  
  label = pracnm;  
  hlo = ' ';  
  output;  
  if (eof) then do;  
    start = .;  
    end = .;  
    label = '** Not Found**';  
    hlo = '0';  
    output;  
  end;  
run;
```

Handling SAS Dates

- Two methods – number or date

```
data date_examples;  
  format date_2 yymmdd10.;  
  length date_3 8;  
  date_1 = 20050823;  
  date_2 = input("20050823", yymmdd8.);  
  date_3 = put(date_2, yymmddn8.);  
run;
```

Date Range Format Data

```
data fitmhs;
  length label $ 11;
  set sasdata.fitmhs end=eof;
  type = 'C';
  fmtname = 'fitmhs';
  label = put(fitmamt, 11.2);
  start = put(fitm, z5.) || put(efctdate, z8.);
  end    = put(fitm, z5.) || put(cncldate, z8.);
  output;
  if (eof) then do;
    hlo = 'O';
    start = ' '; end = ' ';
    label = ' ';
    output;
  end;
run;

proc format cntlin=fitmhs;
run;
```

Excerpt of Data from fitmhs dataset

<u>FMTNAME</u>	<u>TYPE</u>	<u>START</u>	<u>END</u>	<u>LABEL</u>	<u>HLO</u>
fitmhs	C	0010019961001	0010019961015	25.10	
fitmhs	C	0010019961016	0010019970331	26.10	
fitmhs	C	0010019970401	0010019980331	25.83	
fitmhs	C	0010019980401	0010019981031	25.96	
fitmhs	C	0010019981101	0010020000331	26.53	
fitmhs	C	0010020000401	0010020000831	26.36	
fitmhs	C	0010020000901	0010020020331	26.53	
fitmhs	C	0010020020401	0010020021214	29.47	
fitmhs	C	0010020021215	0010099991231	27.90	

Using the Date-Range Format

```
options fmtsearch=(work library);

data office_visit;
  length fitmamt 8;
  infile cards4;
  input fitm servdt;
  fitmamt = put( put(fitm, z5.) || put(servdt, z8.),
               fitmhs.);

cards4;
100 20020425
100 20030913
100 19970202
100 20000223
;;;;
run;
```

Comparison to Interleave – 1

```
data office_visit;
  infile cards4;
  input fitm servdt;
  sort_ord = 2;
cards4;
100 20020425
100 20030913
100 19970202
100 20000223
iiii
run;

proc sort data=office_visit;
  by fitm servdt;
run;
```

Comparison to Interleave – 2

```
data my_fitmhs;  
  set sasdata.fitmhs (keep=fitm efctdate cncldate  
                      fitmamt);  
  sort_ord = 1;  
run;  
  
proc sort data=my_fitmhs;  
  by fitm efctdate;  
run;
```

Comparison to Interleave – 3

```
data office_visit_2;  
  retain fitmamt cncldate;  
  set office_visit (in=a)  
    my_fitmhs (in=b rename=(efctdate=servdt  
                      fitmamt=fitmamt_0  
                      cncldate=cncldate_0));  
  by fitm servdt sort_ord;  
  if (b) then do;  
    fitmamt = fitmamt_0;  
    cncldate = cncldate_0;  
  end;  
  if (a);  
  if (servdt > cncldate) then fitmamt = .;  
run;
```

Comparison to Proc SQL – 1

```
data office_visit;  
  infile cards4;  
  input fitm servdt;  
cards4;  
100 20020425  
100 20030913  
100 19970202  
100 20000223  
iiii  
run;
```

Comparison to Proc SQL – 2

```
proc sql;
  create table office_visit_2 as
  select a.*,
         b.fitmamt
  from office_visit    a    left join
       sasdata.fitmhs  b
  on a.fitm = b.fitm
  and a.servdt between b.efctdate and b.cncldate;
quit;
```

Some Observations

- Do not have to sort the data prior to performing a look up with the format
- Can build the format once and save it permanently for many subsequent uses
- Can control execution with data step IF statements, and call different lookup format/functions within a data step
- If there are overlapping ranges in the data, it will cause an error

Formats in Primary Health Care

- Suite of formats (including many date-range formats) are re-built regularly
- Formats can easily be made available for use via `options fmtsearch` for adhoc use
- Formats are used as lookup functions in data steps throughout production code