PROC FORMAT

The FORMAT Procedure enables you to define your own informats and formats for variables. In addition, you can print the contents of a catalog that contains informats or formats and store description of informats or formats in a SAS data set.

Proc format is useful to create user defined Informats and Formats.

Syntax:-

```
Proc format <options>;
Invalue <$>informat 'old value1'='new value1'
                    'old value2'='new value2'
                    'old valueN'='new valueN';
value <$>format 'old value1'='new value1'
                'old value2'='new value2'
                'old valueN'='new valueN';
Picture format low-high='value';
Run;
```

Invalue statement creates informats.
Value and **Picture** statements create formats.

Difference between Informats and Formats

Informats are useful to read data.
Formats are useful to write data.

Informats we can use when we are reading data from External file (Notepads) or Datalines.
Formats we can use when we are reading data from existing datasets.

Informats works at Input buffer.
Formats works at PDV.

Examples:-

Creating User defined Informats.

Informats works at Input buffer. So we can apply while reading the data from external file (Notepad) or while reading data from Datalines.

```
PROC FORMAT;
INVALUE $GEN 1='FEMALE' 2='MALE';
RUN;
```

```
DATA DS1;
INFILE DATALINES;
INPUT IDNO NAME&$18. TEAM $ SEX$ STRTWGHT ENDWGHT;
INFORMAT SEX$ GEN.;
DATALINES;
1331 JASON SCHOCK LONG  BLUE 1 187 172
1067 KANOKO NAGASAKA  GREEN 1 135 122
1251 RICHARD ROSE  BLUE 2 181 166
1192 CHARLENE ARMSTRONG  YELLOW 2 152 139
1352 BETTE LONG SCHOCK  GREEN 1 156 137
1262 YAO CHEN GARG  BLUE 2 196 180
1124 ADRIENNE FINK  GREEN 2 156 142
;
RUN;
```

Creating User defined Formats.

```
PROC FORMAT;
VALUE $GEND  'F'='FEMALE'
             'M'='MALE';
RUN;
```

Formats we can apply while writing the data into dataset. It means it works in pdv


```
DATA DS1;
INFILE DATALINES;
INPUT IDNO NAME&$18. TEAM $ SEX$ STRTWGHT ENDWGHT ;
/*FORMAT SEX$ GEND.;*/ - APPLYING USER DEFINED FORMAT
DATALINES;
1331 JASON SCHOCK LONG  BLUE F 187 172
1067 KANOKO NAGASAKA  GREEN F 135 122
1251 RICHARD ROSE  BLUE M 181 166
1192 CHARLENE ARMSTRONG  YELLOW M 152 139
1352 BETTE LONG SCHOCK  GREEN F 156 137
1262 YAO CHEN GARG  BLUE M 196 180
1124 ADRIENNE FINK  GREEN M 156 142
;
RUN;
```

Applying Formats in existing Dataset

```
DATA DS2;
SET DS1;
FORMAT SEX $GEND.;
RUN;
```

Applying Format in Report

```
PROC REPORT DATA=DS1 NOWD;
COLUMN  IDNO NAME TEAM SEX STRTWGHT ENDWGHT;
DEFINE SEX/DISPLAY FORMAT=$GEND.;
RUN;
```

Applying Format in Report with styles

```
PROC FORMAT;
VALUE $COL 'F'='RED' 'M'='BLUE';
RUN;
PROC SORT DATA=DS1;
BY SEX;
RUN;
PROC PRINT DATA=DS1 NOOBS;
VAR IDNO NAME TEAM STRTWGHT ENDWGHT SEX/STYLE=[BACKGROUND=$COL.] ;
RUN;
PROC PRINT DATA=DS1 NOOBS;
VAR IDNO NAME TEAM STRTWGHT ENDWGHT SEX/STYLE=[BACKGROUND=$COL.
FOREGROUND=GREEN FONT_WEIGHT=BOLD FONT_SIZE=5 FONT_FACE='COMIC SANS MS'] ;
RUN;
```


Difference between Proc format & If

Examples:-

Approach #1 (Proc format)

```
PROC FORMAT;
INVALUE $X '1'='F'  '2'='M';
VALUE $Y 'F'='FEMALE'  'M'='MALE';
RUN;

DATA DS1;
INPUT IDNO NAME&$18. SEX$ INCOME ;
INFORMAT SEX$ X. ;
FORMAT SEX$ Y. ;
DATALINES;
1331 JASON SCHOCK LONG  1 500000
1067 KANOKO NAGASAKA  1 450000
1251 RICHARD ROSE  2 670000
1192 CHARLENE ARMSTRONG  2 367800
1352 BETTE LONG SCHOCK  1 305000
1262 YAO CHEN GARG  2 500000
1124 ADRIENNE FINK  2 600000
;
RUN;
```

In above example sex values are Female instead of 1 and Male instead of 2.

The same we can apply using IF conditions like below.

Approach #2 (If condition)

```
DATA DS2;
SET DS1;
IF SEX=1 THEN SEX='FEMALE';
ELSE IF SEX=2 THEN SEX='MALE';
RUN;
```

There is nothing inherently wrong with any approach. Existing data set called ds1 has a column called sex which has values of **1** or **2** only, this data step will change the values in Column sex as 'Female' or 'Male' with any approach but one problem with approach#2(If condition) because this approach might be a waste of computer resources, Because a programmer need to write this if condition many times where he required but programmer creates a format this and he store it permanent that format he can call when ever its required. That to if we have many records if condition is difficult to write each and every time so better to go for approach#1(Proc format) approach.

Some more Examples:-

```
PROC FORMAT;
VALUE $GENDER 1='FEMALE'  2='MALE';
PICTURE VALUE LOW-HIGH="00,00,000";
RUN;
DATA DS1;
INFILE DATALINES;
INPUT IDNO NAME&$18. SEX$ INCOME ;
/*FORMAT SEX$ GENDER. INCOME VALUE.;*/
DATALINES;
1331 JASON SCHOCK LONG  1 500000
1067 KANOKO NAGASAKA  1 450000
1251 RICHARD ROSE  2 670000
1192 CHARLENE ARMSTRONG  2 367800
1352 BETTE LONG SCHOCK  1 305000
1262 YAO CHEN GARG  2 500000
1124 ADRIENNE FINK  2 600000
;
RUN;
DATA DS2;
SET DS1;
FORMAT SEX $GENDER.;
RUN;
```

```
DATA DS3;
SET DS1;
FORMAT SEX $GENDER.;
FORMAT INCOME VALUE.;
RUN;
```

Example:-

```
PROC FORMAT;
PICTURE GRADE 60-100='FIRST CLASS'
              50-<60='SECOND CLASS'
              35-<50='THIRD CLASS'
              0-<35='FAIL' ;
RUN;
```

We can use PICTURE to create formats for range values like above**.**

```
DATA SCHOOL;
INPUT STD_ID  MARKS;
/*FORMAT MARKS GRADE.;*/
CARDS;
101 78
102 67
103 56
104 35
105 23
;
RUN;
PROC PRINT DATA=SCHOOL;
FORMAT MARKS GRADE.;
RUN;
```


EXAMPLE:-

```
PROC FORMAT;
PICTURE GRD 60-100='99 - FIRST CLASS'
            50-<60='99 (SECOND CLASS)'
            35-<50='99 , THIRD CLASS'
            0-<35='99 *** FAIL' ;
RUN;
DATA SCHOOL;
INPUT STD_ID  MARKS;
/*FORMAT MARKS GRD.;*/
CARDS;
101 78
102 67
103 56
```

```
104 35
105 23
;
RUN;
PROC PRINT DATA=SCHOOL;
FORMAT MARKS GRD.;
RUN;
DATA SCHOOOL;
SET SCHOOL;
STATUS=PUT(MARKS,GRD.);
RUN;
PROC PRINT DATA=SCHOOL;
RUN;
```

Options:-

CNTLIN=input-control-SAS-data-set

Specifies a SAS data set from which PROC FORMAT builds informats and formats. CNTLIN= builds formats and informats without using a VALUE, PICTURE, or INVALUE statement. If you specify a one-level name, then the procedure searches only the default library (either the WORK library or USER library) for the data set, regardless of whether you specify the LIBRARY= option.

Example:-

```
DATA PATDATA;
INPUT SUBJECT$ TRT_CODE$;
DATALINES;
124263 A
124264 A
124265 B
124266 B
;
RUN;
DATA ADVERSE;
INPUT ADVERSE$ EVENT$;
DATALINES;
124263 HEADACHE
124266 FEVER
124266 NAUSEA
124267 FRACTURE
;
RUN;
```

```
DATA NEW;
RETAIN FMTNAME 'TRT_FMT' TYPE 'C';
SET PATDATA;
RENAME SUBJECT=START TRT_CODE=LABEL;
RUN;
PROC FORMAT CNTLIN=NEW;
RUN;
```

CNTLOUT=output-control-SAS-data-set

Creates a SAS data set that stores information about informats and formats that are contained in the catalog specified in the LIBRARY= option.

Example:-

```
LIBNAME MYSAS "D:\STANSYS\HYDERABAD\SAS\NOTES";
PROC FORMAT LIBRARY=MYSAS.FMT cntlout=mysas.ds2;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
```

FMTLIB

Prints information about all the informats and formats in the catalog that is specified in the LIBRARY= option. To get information only about specific informats or formats, subset the catalog using the SELECT or EXCLUDE statement.

Example:-

```
LIBNAME MYSAS "D:\STANSYS\HYDERABAD\SAS\NOTES";
PROC FORMAT LIBRARY=MYSAS.FMT fmtlib;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
```

LIBRARY=libref<.catalog>

Specifies a catalog to contain informats or formats that you are creating in the current PROC FORMAT step. The procedure stores these informats and formats in the catalog that you specify so that you can use them in subsequent SAS sessions or jobs.

Note: LIBRARY= can point to either a library or a catalog. If only a libref is specified, then a catalog name of FORMATS is assumed.

Example:-

```
LIBNAME MYSAS "D:\STANSYS\HYDERABAD\SAS\NOTES";
PROC FORMAT library=mysas;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
PROC FORMAT library=mysas.ctlg1;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
```

NOREPLACE

Prevents a new informat or format that you are creating from replacing an existing informat or format of the same name. If you omit NOREPLACE, then the procedure warns you that the informat or format already exists and replaces it.

Example:-

```
PROC FORMAT LIBRARY=MYSAS.CTLG1 fmtlib;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
```

```
PROC FORMAT LIBRARY=MYSAS.CTLG1 fmtlib;
VALUE $GENDER 'F'='X' 'M'='Y';
RUN;
```

Both formats are same ($GENDER) so last format overwrites first format
When you don't want to overwrite use NOREPLACE option.

```
PROC FORMAT LIBRARY=MYSAS.CTLG1 fmtlib noreplace;
VALUE $GENDER 'F'='X' 'M'='Y';
RUN;
```

PAGE

Prints information about each format and informat in the catalog on a separate page.

```
PROC FORMAT LIBRARY=MYSAS.CTLG1 ;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
VALUE $COL 'F'='RED' 'M'='BLUE';
PICTURE VALUE LOW-HIGH="00,00,000";
RUN;
PROC FORMAT LIBRARY=MYSAS.CTLG1 fmtlib page;
RUN;
```

Proc format creates Catalog

```
LIBNAME MYSAS "D:\STANSYS\HYDERABAD\SAS\NOTES";
PROC FORMAT LIBRARY=MYSAS;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
```

Default proc format creates catalog with the name of Formats.
But if you want to create with different name write code like below.

```
LIBNAME MYSAS "D:\STANSYS\HYDERABAD\SAS\NOTES";
PROC FORMAT LIBRARY=MYSAS.CATALOG;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;
```

Proc format prints content of catalog

```
LIBNAME MYSAS "D:\STANSYS\HYDERABAD\SAS\NOTES";
PROC FORMAT LIBRARY=MYSAS FMTLIB;
VALUE $GENDE'F'='FEMALE' 'M'='MALE';
PICTURE AGE 0-16='LOW-AGE' 16-20='TEEN-AGE' 20-40='MIDDLE-AGE' 40-100='OLD-AGE';
RUN;
```

Proc format unload content of catalog

```
PROC FORMAT LIBRARY=MYSAS CNTLOUT=MYSAS.DATASET;
VALUE $GENDE'F'='FEMALE' 'M'='MALE';
PICTURE AGE 0-16='LOW-AGE' 16-20='TEEN-AGE' 20-40='MIDDLE-AGE' 40-100='OLD-AGE';
RUN;
```

Creating Formats in Permanent library and reusing that Format

```
PROC FORMAT LIBRARY=MYSAS.FMT;
VALUE $GENDER 'F'='FEMALE' 'M'='MALE';
RUN;

OPTIONS FMTSEARCH=(MYSAS.FMT) ;
DATA MYSAS.DS;
SET SASHELP.CLASS;S
FORMAT SEX $GENDER.;
RUN;
```

```
FMTSEARCH=(LIBRARY.CATALOG)    /*GLOBAL OPTION*/
List of catalogs to search for formats and informats
```


Required columns in Proc format

If that seems like a lot of columns, it is. Most are there to provide the extra levels of control which are needed in specific circumstances. In fact there are only three required columns: FMTNAME, START, and LABEL. In addition to theses required columns it is good habit to include the TYPE column which explicitly tells PROC FORMAT that you are building a numeric or character format. Of course if your format is to include ranges, you will need to include an END column as well as the START column. Finally, the HIGH, LOW, and OTHER keywords are coded in the HLO column.
In summary, the six commonly useful columns are listed below:

```
VARIABLE   TYPE   LABEL

FMTNAME    Char   Format name
TYPE       Char   Type of format
START      Char   Starting value for format
END        Char   Ending value for format
LABEL      Char   Format value label
HLO        Char   Additional information
```

```
DATA FMT;
RETAIN FMTNAME 'TRT_FMT' TYPE 'C';
SET PATDATA;
RENAME SUBJECT=START TRT_CODE=LABEL;
RUN;
PROC FORMAT CNTLIN=FMT FMTLIB;
RUN;
```

Applying formats using PUT Function

```
DATA ALLDATA;
SET ADVERSE;
ATTRIB TRT_CODE  LABEL='TREATMENT CODE';
TRT_CODE=PUT(SUBJECT,$TRT_FMT.);
RUN;
```

Creating formats using Picture

```
PROC FORMAT;
PICTURE PHONE
LOW - HIGH = '(999)999-9999'( PREFIX = '(' );
RUN;

DATA PHONES;
INFILE CARDS;
INPUT PHONE;
FORMAT PHONE PHONE.;
CARDS;
3363153714
8009595605
3153820
;
RUN;
```

MULTIVALUE LABELS

```
DATA VENDOR;
INFILE CARDS;
INPUT VENDOR $ REGION $ SALESP $;
CARDS;
ABC NE ALICE
DEF MW MOLLY
XYZ SE LINDA
;
RUN;
DATA CNTLIN( KEEP = FMTNAME TYPE START LABEL );
RETAIN TYPE 'C';
SET VENDOR;
START = VENDOR;
FMTNAME = 'REGION'; LABEL = REGION; OUTPUT;
FMTNAME = 'SALESP'; LABEL = SALESP; OUTPUT;
RUN;
PROC SORT DATA = CNTLIN;
BY FMTNAME;
RUN;
PROC FORMAT CNTLIN = CNTLIN;
RUN;
```

HYBRID FORMATS

```
PROC FORMAT;
VALUE OTDATE .Z = 'SOME ZS'
             .N = 'SOME 9S'
             OTHER = [DATE9.];
RUN;

PROC FORMAT;
INVALUE INDATE '00000000' = .Z
               '99999999' = .N
               OTHER = [YYMMDD8.];
RUN;
DATA SUGME;
INPUT ADATE INDATE8.;
CARDS;
00000000
99999999
20000605
;
RUN;
PROC PRINT DATA = SUGME;
FORMAT ADATE OTDATE.;
RUN;
```


Performing MERGE using PROC FORMAT

If both datasets contains same subjects

```
DATA PATDATA;
INPUT SUBJECT$ TRT_CODE$;
DATALINES;
124263 A
124264 A
124265 B
124266 B
;
RUN;
DATA ADVERSE;
INPUT SUBJECT$ EVENT$;
DATALINES;
124263 HEADACHE
124264 FEVER
124265 NAUSEA
124266 FRACTURE
;
RUN;
```

```
DATA FMT;
RETAIN FMTNAME 'TRT_FMT' TYPE 'C';
SET PATDATA;
RENAME SUBJECT=START TRT_CODE=LABEL;
RUN;

PROC FORMAT CNTLIN=FMT FMTLIB;
RUN;
DATA ALLDATA;
SET ADVERSE;
ATTRIB TRT_CODE  LABEL='TREATMENT CODE';
TRT_CODE=PUT(SUBJECT,$TRT_FMT.);
RUN;
```

If both datasets contains different subjects

```
DATA PATDATA;
INFILE DATALINES;
INPUT SUBJECT$ TRT_CODE$;
DATALINES;
124263 A
124264 A
124265 B
124266 B
;
RUN;
```



```
DATA ADVERSE;
INFILE DATALINES;
INPUT SUBJECT$ EVENT$;
DATALINES;
124263 HEADACHE
124267 FEVER
124268 NAUSEA
124269 FRACTURE
;
RUN;

DATA FMT;
RETAIN FMTNAME 'TRT_FMT' TYPE 'C';
SET PATDATA;
RENAME SUBJECT=START TRT_CODE=LABEL;
RUN;

PROC FORMAT CNTLIN=FMT FMTLIB;
RUN;
```

```
DATA ALLDATA;
SET ADVERSE;
ATTRIB TRT_CODE  LABEL='TREATMENT CODE';
TRT_CODE=PUT(SUBJECT,$TRT_FMT.);
RUN;
```

Important documents which helps to understand the power of Proc format


Using Advanced Features of User-defi


Proc Format - Advanced Techniques


THE POWER OF PROC FORMAT.pdf


Advanced Techniques to Build a


PROC FORMAT in action.pdf


Ten Things You Should Know About P


I Can Do That With PROC FORMAT.pdf


Alternatives to Merging SAS Data Se


Merging Data Eight Different Ways.pdf


Using Advanced Features of User-defi


Proc format options -suji.pdf

Frequently asking interview questions about Proc format

1) What is difference between Informats & Formats?
2) What is system defined Informats & Formats? Explain with example?
3) What is user defined Informats & Formats? Hot you can create?
4) Write syntax of proc format?
5) Can we apply user defined formats in input statement after variable?
6) If you create a format using Proc format where it will be store? How can you store into permanent libraries?
7) How can you call an existing format from your permanent library to your new program?
8) Can we create separate catalog for each format using Proc format? How?
9) Can we create formats for Numeric values? How?
10) What is an FMTSEARCH global option?
11) What is picture in Proc format? Explain?
12) What is CNTLIN=DATASET and CNTLOUT=DATASET in Proc format? Explain?
13) What is FMTLIB in Proc format? Explain?
14) Can we perform Merge using Formats? How?
15) What is hybrid formats? How can you create?
16) Can we perform merge using proc format?
17) How can you apply multi value labels using proc format?
18) What is noreplace and page options in proc format?
19) What is ERROR: The format [NAME] was not found or could not be loaded.
20) Describe 1 way to avoid using a series of "IF" statements such as

```
If branch = 1 then premium = amount;
Else if branch = 2 then premium = amount;
Else if..... ;
Else if branch = 20 then premium = amount;
```

21) How would remove a format that has been permanently associated with a variable?

22) How do I create a data set with a permanent variable value label (permanent format)?

23) I have just received a SAS data file (emp.sas7bdat) with a SAS format file (formats.sas7bcat). How do I use them?

24) What is the result with below code? If anything is wrong correct it?

```
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 ID$3.  GRADES CONVERT.;
        DATALINES;
        001 A-
        002 B+
        003 F
        004 C+
        005 A
        ;
        RUN;
PROC PRINT DATA=GRADES;
RUN;
```