Paper SAS2032-2018

Management and Usage of User-Defined Formats in the SAS® Cloud Analytic Services Server

Denise Poll, SAS Institute Inc.

ABSTRACT

Since the “dawn of SAS®,” user-defined formats have been created and applied to data values. Formats are used to control the written appearance of data values, or, in some cases, to group data values together for analysis. SAS® Cloud Analytic Services (CAS) server actions support the creation and management of user-defined formats. Other CAS server actions use the formats to manipulate data and apply formats to generate tables or reports. Learn the basic concepts to create and manage format libraries. Get a deeper understanding of how a library can be global and local to a CAS session. Recognize the role of the format search list when applying formats.

INTRODUCTION

SAS provides formats for controlling how variables are displayed or printed. A format is a set of instructions that SAS uses to display a data value. Many formats, such as PERCENT and MONNAME, are shipped with SAS. User-defined format support allows customers to create customized formats. User-defined formats are created, stored, and applied using SAS language and CAS action syntax.

The examples in this paper use PROC FORMAT, the CAS statement, and PROC CAS to demonstrate support for user-defined formats. SAS language procedures and statements, such as PROC FORMAT and the CAS statement, use standard SAS language syntax to generate CAS actions and manage responses to actions. Familiar syntax allows experienced SAS programmers to immediately use PROC FORMAT and the CAS statement. The PROC CAS examples demonstrate the ability to interact with a CAS server from the SAS client using a programming environment based on the CASL language specification. The programming environment enables you to run CAS actions and use the results to prepare the parameters for another action. Use the LISTHISTORY keyword with the CAS statement (such as “Cas mysess listhistory” ;) to list actions generated by PROC FORMAT and the CAS statement.

WHY USER-DEFINED FORMATS

Numeric values are efficient when storing and analyzing data. When displaying data, a character representation often provides a more understandable or interpreted representation.

For example, the jobclass format below associates a numeric value representing job classification to a character string representing job title.

```
proc format casfmtlib=myfmts;
  value jobclass
  100='butcher'
  110='baker'
  111='candlestick maker'
  200='other';
```

Character values often represent an abbreviated form of a value. For example, restaurant type can be represented by an abbreviation, which then maps to a value (such as “IT” to Italian food).

```
proc format casfmtlib=myfmts;
  value restype
  IT='Italian'
  MD='Mediterranean'
  KR='Korean';
```

CREATING USER-DEFINED FORMATS

PROC FORMAT and PROC CAS support the creation of a user-defined format in a CAS session. Traditionally user-defined formats created with PROC FORMAT are retained in a SAS catalog. A SAS catalog provides a way to logically group formats. When using a CAS server session, user-defined formats are retained in a format library.

A format library is like a catalog in that a format library contains one or more user-defined formats. The format library is referenced with a logical name. Format libraries can be local to a CAS session or promoted and available to all sessions.

Each format in the format library has a unique name. Promoted and session local format libraries are kept in memory. Format libraries can be dropped from memory, saved to a caslib location, and added from the caslib location.

Create a Format Library and Format Using PROC FORMAT – Generates Actions

This code creates and populates a CAS session format library named MYFMTS with a format named Publication.

```
proc format casfmtlib='MYFMTS';
value Publication
   1='NYTimes'
   2='Insider'
   3='Vogue’
   4='Villager'
   5='NightRider';
```

```
73         proc format casfmtlib='MYFMTS';
NOTE: Both CAS based formats and catalog-based formats will be written.
The CAS based formats will be written to the session MYSESS.
 74         value Publication
 75            1='NYTimes'
 76            2='Insider'
 77            3='Vogue'
 78            4='Villager'
 79            5='NightRider’;

NOTE: Format library MYFMTS added. Format search update using parameter
APPEND completed.
NOTE: Format PUBLICATION has been output.
```

Output 1. PROC FORMAT: Add Format Library and Format

Notice that the log indicates that the format library MYFMTS and the format PUBLICATION were added. The format search append note indicates that the session format search list was appended with MYFMTS. Appending to the format search list is the default behavior when a format library is added. The format search list is searched to find a format when a format is applied to a variable value.

Create a Format Library and Format Using PROC CAS – Explicit Action

```
proc cas;
session mysess;
sessionProp.addFmtLib/
         fmtLibName='MYFMTS'
         fmtSearch='APPEND'
         promote=false
         replace=false;

sessionProp.addFormat/
         fmtLibName='MYFMTS',
         fmtName='Publication',
         ranges={
             "1='NYTimes'",
             "2='Insider'",
             "3='Vogue'",
             "4='Villager'",
             "5='NightRider'"
         },
         replace=true;
```

```
          proc cas;
             session mysess;
             sessionProp.addFmtLib/
                      fmtLibName='MYFMTS'
                      fmtSearch="APPEND"
                      promote=false
                      replace=false;


             sessionProp.addFormat/
                      fmtLibName='MYFMTS',
                      fmtName='Publication',
                      ranges={
                          "1='NYTimes'",
                          "2='Insider'",
                          "3='Vogue'",
                          "4='Villager'",
                          "5='NightRider'"
                      },
                     replace=true;
 NOTE: Active Session now mysess.
 NOTE: Format library MYFMTS added. Format search update using parameter
APPEND completed.
```

Output 2. PROC CAS: Add Format Library and Format

LIST MEMBERS IN A FORMAT LIBRARY

When both LISTFORMATS and MEMBERS are specified on the CAS statement, all format libraries known to the session and all members in each format library are listed. This enables you to see which formats are available to your program.

Use the CAS statement to list members of a format library.

```
cas mysess listformats fmtlibname=myfmts members;
```

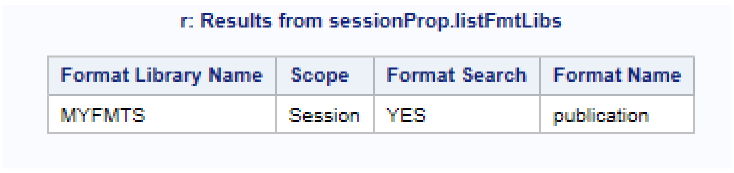
```
73    cas mysess listformats fmtlibname=myfmts members ;
NOTE: Fmtlib = MYFMTS
         Scope = Session
         Fmtsearch = YES
         Format = publication
```

Output 3. CAS Statement: List Format Library Members

Use PROC CAS to list members of a format library.

```
proc cas ;
  sessionProp.listFmtLibs result=r /
  fmtlibname='myfmts'
  showMemNames=true ;
  print r
```

```
73          proc cas ;
74               sessionProp.listFmtLibs result=r /
75               fmtlibname='myfmts'
76               showMemNames=true ;
77               print r ;
```


r: Results from sessionProp.listFmtLibs

Format Library Name	Scope	Format Search	Format Name
MYFMTS	Session	YES	publication

Output 4. PROC CAS: List Format Library Members

Notice RESULT=R in the listFmtLibs action syntax. R represents the response returned from the listFmtLibs action request. For listFmtLibs, the response is a table with four columns and one row for each format library and format. As shown, CASL can be used to print the table or can be used to read or analyze the result by row or column.

LIST THE RANGES OF A FORMAT

Format ranges reflect the association of variable values to character values that are defined when the format is created. After a format is created, the ability to list ranges can be used to determine if a format is appropriate for use with a variable, or if a format range needs to be updated.

Use the CAS statement to list the ranges for a format.

```
cas mysess listfmtranges
    fmtname=publication;
```

Use the CAS statement to list ranges for format Publication.

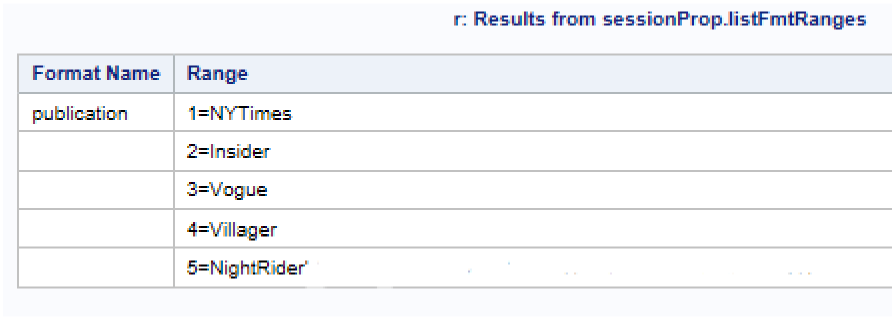
```
            cas mysess listfmtranges
                fmtname=publication;
Format Name         Range
PUBLICATION         1=NYTimes
                    2=Insider
                    3=Vogue
                    4=Villager
                    5=NightRider';
NOTE: Request to LISTFMTRANGES PUBLICATION completed for session
MYSESS.
```

Output 5. CAS Statement: List Format Ranges

Use PROC CAS to list the ranges for format Publication in a specific format library MYFMTS.

```
proc cas ;
  sessionProp.listFmtRanges result=r /
  fmtlibname='MYFMTS'
  fmtname='publication';
```

```
            proc cas ;
                sessionProp.listFmtRanges result=r /
                fmtlibname='MYFMTS'
                fmtname='publication';
```



r: Results from sessionProp.listFmtRanges

Format Name	Range
publication	1=NYTimes
	2=Insider
	3=Vogue
	4=Villager
	5=NightRider'

Output 6. PROC CAS: List Format Ranges for a Specific Format Library and Format

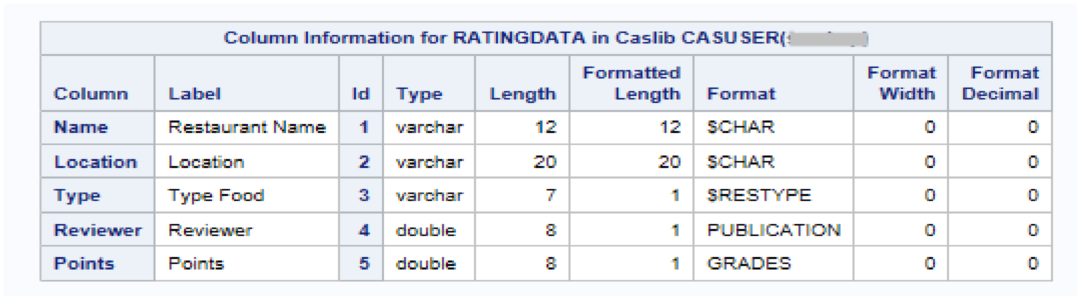
ASSOCIATE A TABLE VARIABLE WITH A FORMAT

The following table, RATINGDATA, identifies restaurant rating information. The column Reviewer contains numeric data with an associated numeric format of PUBLICATION. The column type contains a two-character abbreviation for a type of restaurant with an associated format of $ResType.

This example uses PROC CAS to load a comma-separated value (CSV) file of data and to associate formats with the variables.

```
proc cas ;
 /* Load csv data from a file that is available to the CAS server */
 loadTable /
    caslib='casuser'
    path='sgfratings.csv'
    casOut={caslib='casuser'
            replace='true'
            name='ratingdata' }

    importOptions={
       fileType='CSV'
       encoding='latin1'
       varChars='true'
       delimiter=','}

    /* Associate formats with the vars */
    vars={
         {name="Name",     label="Restaurant Name",  format="$CHAR"}
         {name="Location", label="Location",         format="$CHAR"}
         {name="Type",     label="Type Food",        format="$ResType"}
         {name="Reviewer", label="Reviewer",         format="Publication"}
         {name="Points",   label="Points",           format="Grades"}
   } ;

 /* Display the column attributes */
 columnInfo  result=r /
    table={caslib='casuser' name='ratingsdata'};
 run;
 print r ; run ;
```

```
13? proc cas ;
    loadTable /
       caslib='casuser'
       path='sgfratings.csv'
       casOut={caslib='casuser'
               replace='true'
               name='ratingdata' }
       importOptions={
          fileType='CSV'
          encoding='latin1'
          varChars='true'
          delimiter=','}
 28? /* Associate formats with the vars */
 29? vars={
 30?   {name="Name",     label="Restaurant Name",  format="$CHAR"}
 31?   {name="Location", label="Location",         format="$CHAR"}
 32?   {name="Type",     label="Type Food",        format="$ResType"}
 33?   {name="Reviewer", label="Reviewer",         format="Publication"}
 34?   {name="Points",   label="Points",           format="Grades"} };


NOTE: Active Session now MYSESS.

NOTE: Cloud Analytic Services made the file sgfratings.csv available as
table RATINGDATA in caslib CASUSER.


 38?          columnInfo result=r /
 39?              table={caslib='casuser' name='ratingdata'};
 40?          run;
 41?   print r ; run ;
```



Column Information for RATINGDATA in Caslib CASUSER([redacted])

Column	Label	Id	Type	Length	Formatted Length	Format	Format Width	Format Decimal
Name	Restaurant Name	1	varchar	12	12	$CHAR	0	0
Location	Location	2	varchar	20	20	$CHAR	0	0
Type	Type Food	3	varchar	7	1	$RESTYPE	0	0
Reviewer	Reviewer	4	double	8	1	PUBLICATION	0	0
Points	Points	5	double	8	1	GRADES	0	0

Output 7. Load a Table with Associated Formats

The in-memory table named RATINGDATA is local to the session in the caslib CASUSER. The columnInfo action returns a table result identified using the syntax result=r.

HOW A FORMAT IS FOUND WHEN NEEDED

During table processing, when a user-defined format is associated with a variable, format libraries are searched to find the format. A format search list for the CAS session is used to list format libraries to be searched and the order in which the search occurs. The first occurrence of the format name in the search list is used.

The format search list is established using the CAS statement or PROC CAS with the setFmtSearch action.

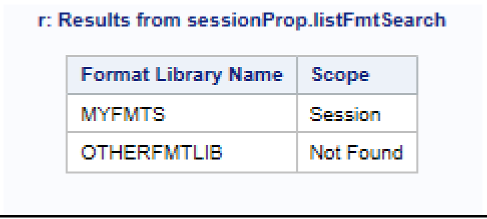
```
cas mysess fmtsearch=(myfmts otherfmlib);

cas mysess listfmtsearch ; run;
```

```
99    cas mysess fmtsearch=(myfmts otherfmlib) ;
NOTE: Request to FMTSEARCH completed for session MYSESS.

100   cas mysess listfmtsearch ;
NOTE: FmtLibName = MYFMTS
         Scope = Session
NOTE: FmtLibName = OTHERFMLIB
         Scope = Not Found
NOTE: Request to LISTFMTSEARCH completed for session MYSESS.
```

Output 8. CAS Statement: Establish and List Format Search

```
proc cas ;
 sessionProp.setFmtSearch / fmtLibnames={'MYFMTS','OTHERFMTLIB'};

 listFmtSearch result=r ;
 print r ; run;
```

```
101  proc cas ;
102   sessionprop.setfmtsearch / fmtLibNames={'MYFMTS', 'OTHERFMTLIB'}

103   listFmtSearch result=r ;
104   print r ; run;
NOTE: Active Session now MYSESS.
```


r: Results from sessionProp.listFmtSearch

Format Library Name	Scope
MYFMTS	Session
OTHERFMTLIB	Not Found

Output 9. Proc CAS: Establish and List Format Search

Notice that information about the scope of the format library is returned as scope local or scope global. "Not Found" indicates that the format library is in the search list but is not available local to the session or global to the server. When performing a search for a format, a "not found" format library is ignored in the search.

FORMAT LIBRARY - SESSION LOCAL VERSUS GLOBAL

A format library can be promoted, which makes the format library global to all sessions. Usually, an administrator determines if a format library needs to be global. Use the listFmtSearch action to identify format libraries that are local to the session and global to all sessions.

CAS server configuration files can contain actions statements to execute during server startup prior to any sessions starting. Specify addFmtLib actions to make one or more format libraries global to all sessions and available when the first session starts. Specify the configuration option FMTSEARCH= to identify the format library search order list to set during session startup. Use both addFmtLib actions and FMTSEARCH= to establish a user-defined format environment that is immediately available when the first session starts. The server startup support eliminates the need for each session to run actions to add format libraries and set a format search list.

SAVING A FORMAT LIBRARY

Cas format libraries can be saved to a caslib with the saveFmtLib action in Proc CAS or with the SAVEFMTLIB option on the CAS statement.

A best practice is for a site to define a caslib named Formats and use this caslib as a centralized place to save format libraries. The name that you use when saving a format library does not have to match the logical name of the format library. In other words, myfmts can be saved to table=savedmyfmts. The extension of the saved format library is ".sashdat". A saved format library is always added using the addfmtlib action. Saved format libraries are a special type of table. Therefore, the addFmtLib action is called to add the format library to a session instead of the loadTable action.

```
cas mysess savefmtlib fmtlibname=myfmts
        caslib=formats table=myfmts replacetable;
cas mysess addfmtlib fmtlibname=myfmts
        caslib=formats table=myfmts ;
```

```
 73          cas mysess savefmtlib fmtlibname=myfmts
 74              caslib=formats table=myfmts replacetable;
 NOTE: Cloud Analytic Services saved the file myfmts.sashdat in caslib
FORMATS
 NOTE: The format library MYFMTS save to myfmts completed successfully.
 NOTE: Request to SAVEFMTLIB MYFMTS completed for session MYSESS.


 75          cas mysess addfmtlib fmtlibname=myfmts
 76              caslib=formats table=myfmts replacefmtlib;
 NOTE: Format library MYFMTS added. Format search update using parameter
APPEND completed.
 NOTE: Request to ADDFMTLIB MYFMTS completed for session MYSESS.
```

Output 10. CAS Statement: Save and Add a Format Library

```
proc cas ;
    saveFmtLib / fmtLibName='myfmts' caslib='formats'
                 name='myfmts' replace=true;
    addFmtLib / fmtLibName='myfmts' caslib='formats'
                name='myfmts' replace=true;
```

```
76?  proc cas ;
77?    saveFmtLib / fmtLibName='myfmts' caslib='formats'
                    name='myfmts' replace=true;
NOTE: Active Session now ONE.
NOTE: Cloud Analytic Services saved the file myfmts.sashdat in caslib
Formats.
NOTE: The format library MYFMTS save to myfmts completed successfully.

 78?  addFmtLib / fmtLibName='myfmts' caslib='formats'
                  name='myfmts' replace=true;
NOTE: The format library MYFMTS was added. The format search list was not
      updated. The format library already exists in the search list.
```

Output 11. PROC CAS: Save and Add a Format Library

Notice that the addFmtLib action did not update the session format search list because the format library was already in the format search list.

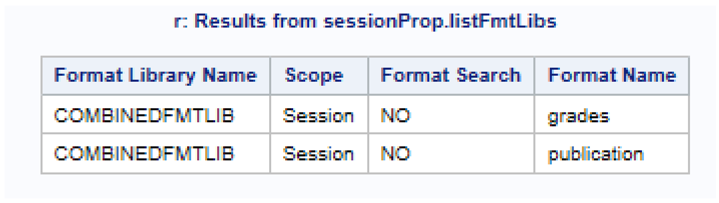
COMBINE FORMAT LIBRARIES

The ability to combine whole format libraries or a select list of formats from one or more format libraries is supported. Use the combineFmtLibs action to identify the format libraries and, if needed, specific formats to combine. This action provides "organizational therapy" for format libraries. Use this action to consolidate several format libraries into a single format library or to select formats specific to an application and create a single format library.

This code combines format libraries myfmts1 and myfmts2 to create the format library combineFmtlibs, which contains only two members, publication and grades.

```
proc cas ;
   combineFmtlibs /
      fmtLibsIn={'myfmts1' 'myfmts2}
      formatNames={'publication' 'grades'}
      fmtLibOut='combinedfmtlib'
      ignoreNameNotFound=true;

   listFmtLibs result=r /
      fmtLibName='combinedfmtlib'
      showMemNames=true;
   print r;
```

```
        proc cas ;

          combinefmtlibs /
             fmtlibsin={'myfmts1' 'myfmts2'}
             formatNames={'publication' 'grades'}
             fmtlibout='combinedfmtlib'
             ignorenamenotfound=true;

          listfmtlibs result=r /
             fmtlibname='combinedfmtlib' showmembernames=true;
          print r;
.
```



r: Results from sessionProp.listFmtLibs

Format Library Name	Scope	Format Search	Format Name
COMBINEDFMTLIB	Session	NO	grades
COMBINEDFMTLIB	Session	NO	publication

Output 12. PROC CAS: Combine Format Libraries

CONCLUSION

As demonstrated, PROC FORMAT, the CAS statement and PROC CAS support adding, listing and saving user-defined format libraries.

The following reference table provides a summary of supported keywords and parameters. Note that the actions and CASL language that are shown with PROC CAS can also be used with other clients, such as Python, Java, Rest, and Lua.

REFERENCE TABLE

Language	Purpose
PROC FORMAT	Creates a format
	Moves client-side catalog or catalogs to a format library
CAS statement	ADDFMTLIB PROMOTEFMTLIB SAVEFMTLIB DROPFMTLIB FMTSEARCH LISTFORMATS LISTFMTRANGES LISTFMTSEARCH
PROC CAS	ADDFMTLIB ADDFORMAT DELETEFORMAT PROMOTEFMTLIB SAVEFMTLIB DROPFMTLIB SETFMTSEARCH LISTFMTLIBS

Language	**Purpose**
	LISTFMTRANGES LISTFMTVALUES LISTFMTSEARCH COMBINEFMTLIBS
%UDFSEL	Scans data sets in a SAS library (libref) and generates SELECT statements for the user-defined formats that are referenced. Points to a catalog or catalogs of formats and creates a CAS session format library.
PROC FMTC2ITM	Migrate catalog or catalogs containing user-defined formats to a CAS session.

ACKNOWLEDGMENTS

Software development is a collaborative effort. I want to acknowledge development, testing, and documenting partners who contribute in the creation and delivery of SAS System and CAS server support.

RECOMMENDED READING

SAS Institute Inc. 2018. *SAS® Cloud Analytic Services 3.3: Fundamentals*. Cary, NC.: SAS Institute Inc. Available at http://go.documentation.sas.com/?docsetId=casfun&docsetTarget=titlepage.htm&docsetVersion=3.3&locale=en.

SAS Institute Inc. 2018. "FORMAT Procedure." In *Base SAS® Procedures Guide*. Cary, NC.: SAS Institute Inc. Available at http://go.documentation.sas.com/?docsetId=proc&docsetTarget=n1c16dxnndwfzyn14o1kb8a4312m.htm&docsetVersion=9.4&locale=en.

SAS Institute Inc. 2018. "CAS Statement" In *SAS® Cloud Analytic Services 3.3: User's Guide*. Cary, NC.: SAS Institute Inc. Available at http://go.documentation.sas.com/?docsetId=casref&docsetTarget=p13nxu1c3v5fghn18pyakf52of5e.htm&docsetVersion=3.3&locale=en.

SAS Institute Inc. 2018. "User-Defined Formats." In *SAS® Cloud Analytic Services 3.3: User-Defined Formats*. Cary, NC.: SAS Institute Inc. Available at http://go.documentation.sas.com/?docsetId=casformats&docsetTarget=titlepage.htm&docsetVersion=3.3&locale=en.

CONTACT INFORMATION

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

Denise Poll
100 SAS Campus Drive
Cary, NC 27513
SAS Institute, Inc.
Denise.Poll@sas.com
http:www.sas.com