Additional Metadata for Common Catalog Entry Types
Kenneth W. Borowiak, PPD, Inc., Morrisville, NC

ABSTRACT

Anyone who has worked with SAS® has probably added descriptive attributes to entities such as variables (labels and formats), data sets (labels), reports (titles and footnotes), and programs (comments). Though not as well known, one can add descriptive labels to commonly used catalog entry types, namely formats and macros. This paper will demonstrate how to add, modify and retrieve this metadata.

Keywords: format, macros, PROC CATALOG, metadata

INTRODUCTION

Anyone who has worked with SAS has probably added descriptive attributes to entities such as variables (labels and formats), data sets (labels), reports (titles and footnotes), and programs (comments). The goal of augmenting these entities with more information includes reducing ambiguity, increasing usability, and creating an audit trail. Information about information is referred to as metadata. Creating user defined metadata comes at the expense of time, which itself may be scarce, and requires diligence of the programmer.

This paper explores adding metadata to and retrieving it from two commonly used catalog entry types, namely formats and macros. While in many cases one can look at the source code for these entities, it is possible that you may need to use a compiled format or macro of unknown heritage. Or you might be a developer of a macro who wishes to provide some information on the entry to the end user while purposely shielding them from the source code.

MACROS

A macro is compiled code which when invoked generates code to be executed. Consider the very simple macro below in Figure 1 that writes to the log the date of the previous day.

Figure 1
Adding a Descriptive Label to a Macro with the DES Option

```
option sasmstore=nesug mstrored ;

@macro yesterday / store des="Display yesterday's date in log" ;
  option nomprint nomlogic nosymbolgen ;
  %let today=%sysfunc( today() ) ;
  %let yesterday=%sysevalf( &today.-1 ) ;
  %put WARNING-Yesterday was %sysfunc( putn(&yesterday.,date9.) ) ;
%mend ;
```

The macro %yesterday is compiled in a macro catalog in the library associated with the libref NESUG. Notice the use of the DES option on the macro statement to add a descriptive label to %yesterday. Any user pointing to the library associated with the libref

\[1\] The assumption is that the information provided is accurate.
NESUG with the appropriate system options set can call %yesterday. However, they cannot see the code that generated the output because the debugging options have been explicitly turned off within the macro and %copy is not helpful since the SOURCE option was omitted from the macro statement. That is unless they have access to the program that created %yesterday. By adding a description to the macro, anyone with access to the macro can retrieve any user-defined information about it. Such information may include the author, purpose, version, caveats, where to find detailed documentation about it or location of the source code. The DES option allows up to 40 characters for the description.

In addition to adding a label to a macro at the time it is compiled with the DES option, one can also augment the catalog entry with a description by using PROC CATALOG, as shown below in Figure 2.

**Figure 2**  
Adding a Descriptive Label to a Macro with PROC CATALOG

```sas
proc catalog catalog=nesug.sasmacr et=macro;
    modify yesterday(desc='See K. Borowiak for source code') ;
    run;
    quit;
SAS log:
NOTE: Description changed for entry YESTERDAY.MACRO in catalog NESUG.SASMACR.
```

The syntax for PROC CATALOG is similar to that of PROC DATASETS, but operates on catalogs rather than data sets. On the PROC CATALOG statement, you specify the catalog you want to operate on and the entry type (i.e. `et=`), which is `macro` in this case. To add or modify a description for a macro entry in the catalog, use the MODIFY statement and the DESCRIPTION option after the entry. An alias for the DESCRIPTION option is DESC and up to 256 characters can be used for the label.

There are a few ways to retrieve the description associated with a macro. One is to use PROC CATALOG, which is analogous to using PROC CONTENTS to retrieve information on data sets.

**Figure 3**  
Retrieving the Label of a Macro with PROC CATALOG

```sas
proc catalog catalog=nesug.sasmacr ;
    contents out=kman(where=(name='YESTERDAY'));
    run ;
    quit ;
SAS log:
NOTE: The data set WORK.FIG3 has 1 observations and 8 variables.
```

The CONTENTS keyword will display various attributes about all the entries in the specified catalog to whatever ODS destination is open. You can also send the results to a data set with the OUT= option, where data set options can be used to subset records and variables.
Metadata on macros can also be retrieved by querying the DICTIONARY table CATALOGS or the SASHELP view VCATALG, as demonstrated below in Figure 4.

**Figure 4**
Retrieving the Label of a Macro from Dictionary.Catalogs

```sql
proc sql;
select memname,
     objname,
     objdesc
from Dictionary.Catalogs
where libname='NESUG' and
     objtype='MACRO' and
     objname='YESTERDAY';
quit;
```

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Object Name</th>
<th>Object Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SASMACR</td>
<td>YESTERDAY</td>
<td>See K. Borowiak for source code</td>
</tr>
</tbody>
</table>

If you are working in interactive mode, you can also view the description of catalog entries through the Explorer window with the Details option from the View menu. Figure 5 below is a depiction of SAS in the Windows environment, but can also be an analogous view is available in a Unix environment.

**Figure 5**
Viewing the Label of a Macro from the Explorer Window

---

**FORMATS/ INFORMATS**

Formats are applied to variables, which affect how they are displayed. User-defined formats are stored in catalogs and are available as long as the appropriate librefs are
specified in the FMTSEARCH system option. Unlike macros, the values and ranges and their associated labels for formats can always be retrieved by using the CNTLOUT or FMTLIB options on the PROC FORMAT statement. However, you may not be privy to the actual program that generated the format, which may contain useful information such as the author, source of the codes, or why it is needed. This useful metadata can be added as part of the description for format catalog entries.

Unlike macros, descriptions cannot be added to user defined formats at their time of creation. They can be added or modified with PROC CATALOG by specifying the appropriate entry type, as show below in Figure 6.

![Figure 6](image)

*Figure 6*
Adding a Descriptive Label to a Format with PROC CATALOG

```sas
proc catalog catalog=nesugformats et=format;
    modify FNesug(desc='Site of future NESUGs: Source: R. Burgundy');
run;
quit;
```

SAS log:

```
NOTE: Description changed for entry FNESUG.FORMAT in catalog NESUG.FORMATS.
```

Retrieving the descriptions for formats can be accomplished in similar ways already discussed for macros. Figure 7 below shows a query against the VCATALG view to extract the description for the format catalog entry FNESUG.

![Figure 7](image)

*Figure 7*
Retrieving a Label of Format from Sashelp.Vcatalg

```sas
proc sql;
select memname, objname, objdesc
from Sashelp.VCatalg
where libname='WORK' and objtype='FORMAT' and objname like 'FNES%';
quit;
```

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Object Name</th>
<th>Object Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMATS</td>
<td>FNESUG</td>
<td>Site of Future NESUGs: Source: R. Burgundy</td>
</tr>
</tbody>
</table>

The discussion in this section has focused on formats, but it also applies to its cousin the informat, which defines how a variable is read.
CONCLUSION

Having descriptive information on entities in the SAS system is useful and is generally considered a good idea. While many users are familiar with adding descriptive information to variables, data sets, reports and programs, this paper addresses how formats, informats, and macros can be augmented with user provided metadata. This can be particularly helpful for those who need to use a compiled version of the catalog entries when they are not privy to the code that produced them.

REFERENCES


SAS OnLineDoc®

ACKNOWLEDGEMENTS

The author would like to thank Toby Dunn and Mike Molter for their insightful comments in reviewing this paper.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Ken Borowiak
PPD, Inc.
3900 Paramount Parkway
Morrisville NC 27560

Ken.Borowiak@rtp.ppdi.com
EvilPettingZoo97@aol.com