# Controlling your formats

By [Jerry Leonard](https://blogs.sas.com/content/author/jerryleonard/) on [SAS Users](https://blogs.sas.com/content/sgf/) December 4, 2017[Learn SAS](https://blogs.sas.com/content/topic/learn-sas/) | [Programming Tips](https://blogs.sas.com/content/topic/programming-tips/)

During my 35 years of using SAS® software, I have found the CNTLIN and CNTLOUT options in the FORMAT procedure to be among the most useful features that I routinely suggest to other SAS users. The CNTLIN option enables you to create user-defined formats from a SAS data set (input control data set). The CNTLOUT option enables you to create a SAS data set (output control data set) containing format details from an entry in a SAS format catalog.

In this blog post, I provide a few examples demonstrating how to use the CNTLIN option. I also mention how to use the CNTLOUT option to store your format information in case you need to move to a new operating environment.

You can store all the format details from a SAS format catalog in a CNTLOUT data set and later restore them in a format catalog in your new operating environment using the CNTLIN option. For details, see [SAS Usage Note 22194](http://support.sas.com/kb/22/194.html): “How to use the CNTLOUT= and CNTLIN= options in PROC FORMAT to move formats from one platform to another.”

A data set for the CNTLIN option contains variables that give specific information about ranges and values. At a minimum, the data set must contain the following variables:

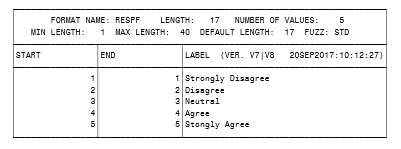
FMTNAME specifies a character variable whose value is the format or informat name.  
START specifies a variable that gives the range's starting value.  
LABEL specifies a variable whose value is associated with a format or an informat.

For details about input and output control data sets, see the “[FORMAT Procedure](http://support.sas.com/documentation/cdl/en/proc/70377/HTML/default/viewer.htm#p0owa4ftikc2ekn1q0rmpulg86cx.htm)” section of *Base SAS® 9.4 Procedures Guide, Seventh Edition.*

### Create a Numeric Format

The following simple example using the CNTLIN option creates a numeric format named respf:

|  |
| --- |
| **data** test;  input response desc $20.;  datalines;  **1** Strongly Disagree  **2** Disagree  **3** Neutral  **4** Agree  **5** Stongly Agree  ;  **run**;    **data** crfmt;  set test;  start=response;  label=desc;  fmtname='respf';  **run**;    proc format library=work cntlin=crfmt fmtlib;  select respf;  **run**; |



### Reveal Data Set Variables

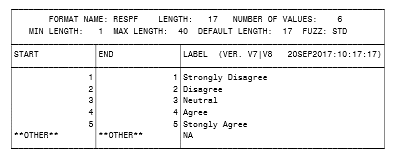
To see the other variables that are included in data sets created by the CNTLIN and CNTLOUT options, use CNTLOUT to create a data set for the respf format created above:

|  |
| --- |
| proc format library=work cntlout=outfmt;  select respf;  **run**;  **proc print** **data**=outfmt;  **run**; |

### Add Additional Ranges

To add another range to the respf format, you can use DATA step processing with the data set created by the CNTLOUT option. Then, re-create the format using the CNTLIN option:

|  |
| --- |
| **data** infmt;  set outfmt end=last;  output;  if last then do;  HLO='O'; */\* indicates a special other range \*/*  label='NA';  output;  end;  **run**;    proc format library=work cntlin=infmt fmtlib;  select respf;  **run**; |

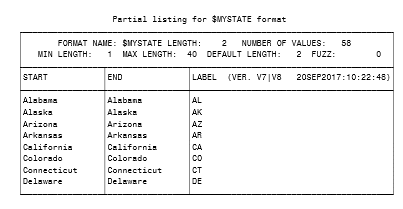


### Convert a State Name to Its Postal Abbreviation

One use for the CNTLIN option is to create a format that converts a state name to its 2-letter postal abbreviation. For example, this option can convert 'North Carolina' to 'NC'.  Because SAS does not have a function or format to convert state names to postal abbreviations, this is an excellent use of the CNTLIN option.

We can use data from the SASHELP.ZIPCODE data set to create a user-defined format using the CNTLIN option, as shown below:

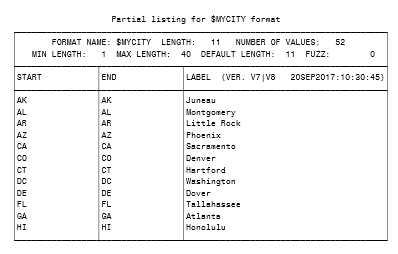
|  |
| --- |
| **proc sql** noprint;  create table crfmt as  select distinct statename as start,  statecode as label,  '$mystate' as fmtname  from sashelp.zipcode;  **quit**;    proc format library=work cntlin=crfmt fmtlib;  select $mystate;  **run**; |



### Identify State Capitals

In a similar manner, we can use the MAPS.USCITY data set to create a user-defined format that identifies state capitals from the 2-letter state abbreviation. See the sample code and partial results below:

|  |
| --- |
| **proc sql** noprint;  create table crfmt as  select distinct statecode as start,  city as label,  '$mycity' as fmtname  from maps.uscity  where capital='Y';  **quit**;    proc format library=work cntlin=crfmt fmtlib;  select $mycity;  **run**; |

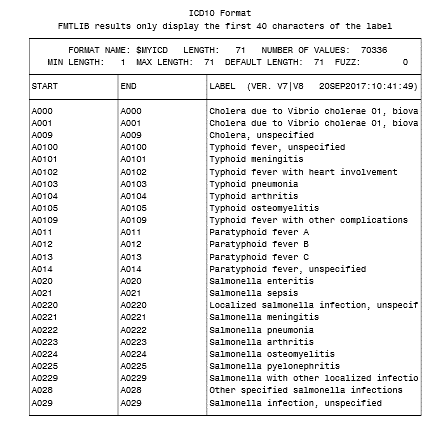


### Use External Data Sources

You can gather information from external data sources and read that information into a data set created by the CNTLIN option to create user-defined formats.

The following example uses ICD10 medical diagnosis codes. I downloaded a list of ICD10 codes and their descriptions into a Microsoft Excel file from the [Center for Medicare & Medicaid Services](https://www.cob.cms.hhs.gov/Section111/assets/section111/icd10.dx.codes.htm) website. Then, I created a user-defined format from the first 25 records. You can also download the codes as a text file.

|  |
| --- |
| */\* This code reads in the Excel file. \*/*  **proc import** out==myicd10  datafile= "C:**\S**ection111ValidICD10-2017.xlsx"  dbms=excelcs replace;  range="'Valid ICD10 2017 &amp; NF Exclude$'";  scantext=yes;  usedate=yes;  scantime=yes;  **run**;      **data** crfmt;  set myicd10 (obs=**25**);  fmtname='$myicd';  start=code;  label=short\_description;  **run**;    title1 'ICD10 Format';  title3 'FMTLIB results only display the first 40 characters of the label';  proc format library=work cntlin=crfmt fmtlib;  select $myicd;  **run**; |



A more complicated example that uses other data set variables created by the CNTLIN option is included in the linked sample program in [Sample 47312](http://support.sas.com/kb/47/312.html): “Create a user-defined format containing decile ranges from PROC UNIVARIATE results.”