

Customize your SAS® Output with the Template Procedure: A Beginning Tutorial

Carol Gosselin, North Carolina State University, Raleigh, NC

Joy Munk Smith, North Carolina State University, Raleigh, NC

Abstract

The Output Delivery System (ODS), introduced in version 7 of the SAS® System, offers many options for formatting SAS output. Default templates can be replaced with custom templates created using the Template Procedure. Output from entire SAS jobs can be modified with style-definitions and single output objects can be formatted with user created table-definitions.

This beginning tutorial will provide several basic PROC TEMPLATE examples. It will cover where templates are stored, how they are accessed, and how they are created. It will cover the concepts of inheritance. PROC TEMPLATE has more options than are reasonable to cover in a beginning tutorial. However, the basic concepts will be demonstrated and references to the most current sources of documentation and examples will be mentioned.

This paper assumes the reader has a basic understanding of ODS. The basics are thoroughly covered in Bryant, Muller, and Pass (1999) and Olinger (2000).

In this paper we will cover:

What are Templates?

Working with Predefined Templates

Creating your own Templates

Key Words: PROC TEMPLATE, Output Delivery System, ODS, SASHELP.TMPLMST, style-definition, table-definition, templates, style element, table element, inheritance.

Introduction

Prior to version 7, all output from SAS procedures was designed for a traditional line-printer. This was limiting because SAS programmers had no control over the output, the output did not transfer well into documents, and some parts of the output were not easily accessed for further analysis and reporting.

SAS version 7 was the first version to put the control for formatting output and creating data sets firmly in the hands of the programmer. SAS procedures no longer format output, instead they create output objects that are sent to the Output Delivery System (ODS) for formatting. ODS formats output objects based on the instructions provided by the destination and template chosen by the programmer. SAS provides default templates so programmers are not required to create templates unless there is a desire to modify the default results.

PROC TEMPLATE can create six types of templates: Table, Column, Header, Footer, Tree, and Style. Table-definitions provide ODS with instructions for displaying

tabular output objects. Table-definitions apply to a specific output object, not the whole SAS job. Header, footer, and column definitions are used with table-definitions to control column order, header text, and footer text. Tree definitions are used for equations and functions. Style-definitions control the presentation aspects of the entire SAS job. Style elements like colors, fonts, and backgrounds are defined with STYLE statements and stored in style-definitions.

The Output Delivery System continues to be enhanced with each subsequent release of SAS. New output destinations are continuously being tested. In version 8, the available destinations include LISTING (for "SAS classic" output), HTML (for HyperText Mark Up), PRINTER (for PS or PCL), and OUTPUT (for creating SAS data sets). In version 8.1, RTF (for Rich Text Format) is production. Other destinations under development include PRINTER (for PDF), XML, LATEX, and DOCUMENT (for binary store).

In this paper, we provide a brief review of ODS and an introduction to the major elements in PROC TEMPLATE. The examples are designed to introduce the concepts involved in creating templates.

Basic ODS Review

ODS output destinations are either open or closed. SAS procedures deliver output objects to all open destinations. No output is sent to a closed destination. Output objects can be sent to multiple destinations. The HTML, PRINTER, and OUTPUT destinations must be closed before the results can be viewed.

Note: It is necessary to submit a RUN statement before closing any destination. If this RUN statement is omitted the output from the last procedure will be absent. Procedures like REG and GLM, that support run-group processing, require a QUIT statement before the OUTPUT destination is closed. There is no harm in including both RUN and QUIT statements and this practice may reduce confusion.

Review of Version 8 ODS Destinations and Options

- ODS LISTING controls the Listing destination. This is the SAS classic output and is open unless specifically closed. When open all output is sent to this destination unless the selection/exclusion list is modified.
- ODS HTML creates HTML files. OPTIONS: STYLESHEET=, BODY=, FILE=, CONTENTS=, FRAME=, PATH=, CLOSE. When open all output is sent to this destination.

- ODS OUTPUT creates SAS data sets. This destination is closed unless specifically opened. This destination requires that desired output objects be requested and the data sets produced must be given a name.
- ODS PRINTER creates PS or PCL files.
OPTIONS: STYLE=, FILE=, PRINTER|PRT=, FONTSCALE=, POSTSCRIPT|PS, HOST | SAS, COLOR, UNIFORM, CLOSE .

Selection and Exclusion Lists

Selection and exclusion lists determine which output objects are sent to open destinations. SAS maintains overall selection and exclusion lists and similar lists for each destination. Requested output objects must be on the overall selection list and the selection list for each desired destination. Selection lists can be viewed with "ODS destination SHOW;". For example: "ODS HTML SHOW;".

ODS SELECT ALL; is the default for the following selection lists: OVERALL, HTML, LISTING, and PRINTER. ODS EXCLUDE ALL; is the default for the OUTPUT destination. The OUTPUT destination produces SAS data sets and requires explicit selection and naming of desired output objects.

Selection lists are automatically reset at step boundaries unless the PERSIST option is specified. This may be confusing to beginning ODS adventurers. For clarification see Bryant, Muller, and Pass (1999).

ODS TRACE statement

The ODS TRACE statement displays information about each of the output objects generated by SAS procedures in the LOG window. Displayed are the name, path, and label of each output object produced. In addition, the template being used is reported. The LISTING option on the ODS trace statement causes this information to appear in the list output instead of in the log. This information is necessary for identifying objects on SELECT, EXCLUDE, and ODS OUTPUT statements.

Sample code to write to the HTML output destination.

```
FILENAME htmldoc "e:\student\ ";
ODS HTML PATH=htmldoc
  FILE='table1.htm'
  FRAME='table1f.htm'
  CONTENTS='table1c.htm'
  STYLE=styles.mystyle
  STYLESHEET= style.css;
```

- FILENAME: (Optional) Defines a FILEREF
- ODS HTML : (Required) Open the HTML destination
- PATH = : (Optional) Define storage location for all HTML files. Can be either a FILEREF or a fully qualified filename.
- FILE= : (Required) BODY= can be substituted. Identifies the HTML output file.

- FRAME = : (Optional) Provides the name for the frame file that integrates the contents and body files when opened in a browser.
- CONTENTS= : (Optional) File containing the table of contents, links to each of the objects in the body file
- STYLE= : (Optional) Indicates the chosen STYLE template. The default style STYLES.DEFAULT is used if no style is specified.
- STYLESHEET= style.css: (Optional) indicates a cascading style sheet to use.

Note: ODS support for cascading style sheets is experimental in version 8 and production in version 8.1. Style settings in the cascading style sheet override style settings in SAS style templates.

Template Store

Templates provided by SAS are located in a "template store" named SASHELP.TMPLMST. A template store is a special item file that allows multiple levels of subdirectories to be stored within. This template store is located in the SASHELP subdirectory of the CORE SAS subdirectory. TMPLMST.SAS7BITM is the actual file name.

User created templates are stored in the template store named SASUSER.TEMPLAT. Other template stores can be specified on the ODS PATH statement.

When searching for a template to use, SAS searches the template stores listed on the ODS PATH statement in the order they are listed, and uses the first matching template it finds. When storing a template SAS uses the first updateable template store it finds. Search order can be checked using the ODS PATH SHOW statement.

Instructions for viewing the contents of a template store are provided below. The SOURCE and LIST statements can also be used. More information on these statements is available in [The Complete Guide to the SAS Output Delivery System](#).

To View The Contents Of A Template Store:

- In the Results Window: Right click on the Results folder
- select Template.

The template stores are listed in the left frame. Within these template stores the templates are organized in folders identified by product and procedure name. For example, the SAS supplied templates for the GLM procedure are stored in the folder SASHELP.TMPLMST.STAT.GLM. The template code can be viewed by clicking on the folder and then on the file to be viewed. Scanning these files is a great way to become familiar with PROC TEMPLATE code, and a good way to learn the names of elements that can be changed.

The driving force behind the evolution of PROC TEMPLATE was the development of the SAS supplied default templates for the numerous output objects created by SAS procedures. When SAS product developers expressed the need for additional options they were added to PROC TEMPLATE. The options are so

numerous that it is highly unlikely anyone remembers them all. Thus, it is often simpler to modify existing templates than it is to begin from scratch and specify every option.

Note: When creating templates, it is best not to use the same names as are used in SASHELP.TMPLMST. SAS looks first in SASUSER.TEMPLAT and then in SASHELP.TMPLMST for requested templates. If your template in SASUSER.TEMPLAT has the same name as one in SASHELP.TMPLMST, it will be used in place of the Institute supplied template with the same name. If your template has a unique name, it will be used only when you request it.

Inheritance

The easiest way to create a template is often to modify an existing template. The new template will inherit settings from the parent template. There are two types of inheritance: style-definition inheritance and style-element inheritance.

Style-definition inheritance is controlled with the PARENT= option on the DEFINE STYLE statement. If the PARENT= option is used, all of the parent's stylistic elements, and presentation aspects are inherited by the new template unless they are redefined within the definition.

Using the STYLE or REPLACE statements within a style-definition controls style-element inheritance. Use of the REPLACE is more advanced and will not be covered in this tutorial.

Examples 3,5,6 demonstrate inheritance. Additional examples and explanation can be found in [The Complete Guide to the SAS Output Delivery System](#).

PROC TEMPLATE Procedure

This beginning tutorial will cover only a few of the features of the Template Procedure. Further information can be found in the documentation and on the SAS Web site at <http://www.sas.com/rnd/base/>. There have also been many good SUGI papers some of which we reference.

Table templates provide the instructions for displaying a table. Column, header, and footer templates are used to enhance table templates. Column, header and footer templates can be created within a table-definition or independent of a table-definition. If they are created independent of a table-definition they can be referenced by other table-definitions. SAS provides several column and header definitions that are used and stored independently. They are stored in SASHELP.TMPLMST.COMMON.COLUMN and can be viewed using the methods described in the template store section of this paper.

Style templates are used to control the look of the output from an entire SAS job. Several of the SAS supplied style-definitions can be viewed at SASHELP.TMPLMST.STYLES. The default style can be viewed at SASHELP.TMPLMST.STYLES.DEFAULT. STYLES.DEFAULT includes a large number of style

elements. It is a good example to study and a good template to use as a parent when creating your own templates. More than 50 style attributes are listed in the STYLE documentation. They do not all affect every style element, but they provide a lot of options for customizing output.

PROC TEMPLATE Syntax

1. PROC TEMPLATE Statement
2. DEFINE Statement
3. DEFINE COLUMN statement
4. DEFINE FOOTER statement
5. DEFINE HEADER statement
6. DEFINE STYLE statement
7. DEFINE TABLE statement
8. DELETE statement
9. EDIT statement
10. END statement
11. LINK statement
12. PATH statement
13. SOURCE statement
14. TEST statement

Note: All DEFINE statements require a corresponding END statement.

Modifying the Table of Contents

To change the label for each piece of output listed in the table of contents you can use the CONTENTS_LABEL option in PROC TEMPLATE. You can use the CONTENT= option in the PRINT, FREQ, and TABULATE procedures. With SAS/GRAPH® use the DESCRIPTION= option. For examples, see Example 6 or Olinger (2000).

Data Used in Examples:

The data used for all but the last example are fictional student data. The variables used are:

- IOF – Residency (A=In-State, B=Out-of-State, C=International)
- SENRSTAT – Enrollment Status (1=New, 2=Transfer, 3=Continue, 4=Reenter)

Tabulate Code Used in Examples:

The following tabulate code will be included in the examples as follows:

```
%INCLUDE "D:\SESUG\TABULATE1.SAS";
```

```
* tabulate1.sas ;  
PROC TABULATE DATA = b FORMAT = 8.0;  
  FORMAT senrstat $statfmt iof $ioffmt.;  
  CLASS senrstat iof;  
  TABLE iof ALL, SENRSTAT ALL;  
  TITLE 'Enrollment by Residency';  
RUN;
```

Note: The RUN statement is required to ensure that the PROC TABULATE runs before the ODS destination is closed. With some procedures the QUIT statement is required. Use of both can avoid confusion.

Example 1: ODS code to create an HTML file using PROC TABULATE and the default SAS style template.

```
FILENAME htmldoc "d:\ncair";
ODS HTML PATH=htmldoc
FILE='table1.htm'
STYLE=styles.default;
```

```
%INCLUDE "d:\sesug\tabulate1.sas";
ODS HTML CLOSE;
```

Note: Since FRAME= and CONTENTS= are not specified, those files will not be created.

Note: The ODS HTML CLOSE statement is required to allow the output to be written to the file specified in the FILE= statement. The output is not available for browsing until after the ODS CLOSE statement is executed.

Running this ODS and PROC TABULATE code yields the output shown in **Figure 1**.

Rerunning the same code, with style set to STYLES.MINIMAL gives a completely different look for the output, as shown in **Figure 2**. This style works well in Microsoft Word® documents.

```
FILENAME htmldoc "d:\ncair";
ODS HTML PATH=htmldoc
FILE='table1.htm'
STYLE=styles.minimal;
RUN;
```

```
%INCLUDE "d:\sesug\tabulate1.sas";
ODS HTML CLOSE;
```

Example 2: Create STYLES.MYSTYLE using PROC TEMPLATE. Start with a DEFINE STYLE statement.

Specify the options for the header, using STYLE HEADER / followed by style elements to be set, a white background, red foreground (text), bold Arial font. The

END statement is required to close the DEFINE statement on PROC TEMPLATE. For any options not set, ODS will use the system defaults.

```
PROC TEMPLATE;
  DEFINE STYLE styles.mystyle;
  STYLE HEADER /
    BACKGROUND= white
    FOREGROUND = red
    FONT_FACE= "arial"
    FONT_WEIGHT= bold;
  END;
RUN;
```

Now, we need another ODS statement to use this style.

```
ODS HTML PATH=htmldoc
FILE='table2.htm'
STYLE=styles.mystyle;

%INCLUDE "d:\sesug\tabulate1.sas";
ODS HTML CLOSE;
```

Running the same PROC TABULATE with this style creates the output shown in **Figure 3**.

Note: Messages in the SAS Log indicate that default style elements will be used for any not defined in this template.

Note: STYLES.MYSTYLE will be stored in SASUSER.TEMPLAT.

Next modify STYLES.MYSTYLE by adding some options for the table. Using STYLE TABLE statement, add a black border around cells of width 2, and spacing of 3 in each cell. Add this code to the above PROC TEMPLATE example and rerun.

```
STYLE TABLE /
  CELLSPACING = 3
  BORDERWIDTH = 2
  BORDERCOLORDARK = black;
```

The addition of this code creates a much more readable table. See **Figure 4**.

Example 3: Create STYLES.FORMSWORD by modifying STYLES.MINIMAL. The PARENT=STYLES.MINIMAL statement allows inheritance of all unspecified style elements from STYLES.MINIMAL. Change the header and system title to use a Bold, Italic, Arial font, and the table so that the width of the borders around the cells is changed to 5. Make the background of the header section gray.

```
PROC TEMPLATE;
  DEFINE STYLE styles.formsword;
  PARENT= styles.minimal;
  STYLE header FROM header/
    BACKGROUND= gray
    FONT_STYLE=italic
    FONT_FACE= "arial"
    FONT_WEIGHT= bold;
  STYLE systemtitle FROM systemtitle /
    FONT_STYLE=italic
    FONT_FACE= "arial"
    FONT_WEIGHT= bold;
  STYLE table FROM table /
    BORDERWIDTH=5;
  END;
RUN;

ODS HTML PATH=htmldoc
FILE='table3.htm'
STYLE=styles.formsword;

%INCLUDE "d:\sesug\tabulate1.sas";
ODS HTML CLOSE;
```

When this code is run the output in **Figure 5** is produced.

Example 4: PROC TEMPLATE can be used to modify the system title and add a graphic image. Returning to the style created in Example 2, add a STYLE SYSTEMTITLE definition to control the look of the text in the TITLE statement. This will produce a large, bold, black title. The PREIMAGE= option will insert the North Carolina State University graphic image before the title. POSTIMAGE= would place the graphic image after the

title. PREIMAGE= and POSTIMAGE= may be used in several of the STYLE statements to place graphics.

```
PROC TEMPLATE;
  DEFINE STYLE styles.mystyle1;
  STYLE header /
    BACKGROUND= white
    FOREGROUND = red
    FONT_FACE= "arial"
    FONT_WEIGHT= bold;

  STYLE table /
    CELLSPACING = 3
    BORDERWIDTH = 2
    BORDERCOLORDARK = black;

  STYLE systemtitle /
    FONT_FACE= "arial"
    FONT_WEIGHT=bold
    FONT_SIZE=4
    PREIMAGE="d:\sesug\ban_logo.gif";

  END;
  RUN;

  ODS HTML PATH=htmldoc
    FILE='table4.htm'
    STYLE=styles.mystyle1;

  %INCLUDE "D:\SESUG\TABULATE1.SAS";
  ODS HTML CLOSE;
```

This code produces the output shown in **Figure 6**.

Example 5: PROC TEMPLATE and ODS can be used to control the frame containing the generated table of contents and body files. Here is an example using the STYLES.MINIMAL. This PROC TEMPLATE creates STYLES.FrameChanges where the table of contents has been moved to the right side of the screen and increased to 40% of the screen width. FRAMEBORDER=on ensures that there will be a divider between the contents and body displays.

```
PROC TEMPLATE;
  DEFINE STYLE styles.FrameChanges;
  PARENT=styles.minimal;
  STYLE frame FROM document /
    CONTENTSIZE= 40%
    CONTENTPOSITION=r
    FRAMEBORDER=on ;
  END;
  RUN;

  ODS HTML PATH=htmldoc
    FILE='table5.htm'
    CONTENTS='content5.htm'
    FRAME='frame5.htm'
    STYLE=styles.FrameChanges;

  %INCLUDE "D:\SESUG\TABULATE1.SAS";
  ODS HTML CLOSE;
```

Running this code produces the output in **Figure 7**.

Example 6:

This example combines the use of many of the style elements mentioned in previous examples along with creating a customized ANOVA table. Color is used heavily to stress the topics being demonstrated. Hopefully the use of color will aid understanding of this example. The colors will not show up well in print, but they will be clear when the program is submitted and in the paper presentation. Part of the output from this example can be seen in **Figure 8**.

Create the style definition: STYLES.MYDEFAULT

```
PROC TEMPLATE;
  DEFINE STYLE styles.mydefault;
  PARENT= styles.default;
  NOTES 'Notes are stored with the template,
  comments are not';
  **USER DEFINED ATTRIBUTES;
  STYLE PROC TEMPLATE;
  fonts 'User defined attributes' /
    'myfont'=("ARIAL, Helvetica, Helv",2)
    'myboldfont'=("ARIAL, Helvetica, Helv",2,Bold);
  STYLE html 'user defined html' /
    'fake BULLET'= 'upper-roman'
    'fake2 BULLET'='upper-alpha';
  **ABSTRACTS;
  STYLE mydoc /
    FONT=fonts('myfont')
    FOREGROUND=black
    BACKGROUND=white;
  STYLE mydocbold FROM mydoc /
    FONT_WEIGHT=BOLD;
  **MODIFY FRAME FILE;
  STYLE frame FROM mydoc /
    BACKGROUND=white
    CONTENTSIZE=25%;
  **MODIFY CONTENTS FILE;
  STYLE contents FROM mydocbold /
    BULLET=html('fake BULLET')
    LEFTMARGIN=5;
  STYLE indexitem FROM mydocbold /
    /*BULLET=html('fake2 BULLET')*/
    BULLET=decimal
    LISTENTRYANCHOR=on
    FOREGROUND=red;
  STYLE contentprocname FROM indexitem /
    PRETEXT=""
    POSTTEXT=""
    FONT_SIZE=4.2;
  STYLE contentfolder FROM indexitem /
    FONT_WEIGHT=light
    /*FOREGROUND=white*/
    BULLET=NONE
    FONT_SIZE=3.2;
  STYLE contentitem FROM indexitem /
    FONT_STYLE=italic
    FONT_SIZE=2.2;
  **MODIFY BODY FILE;
  STYLE body FROM mydoc /
    BACKGROUND=blue
    LEFTMARGIN=5;
```

```

STYLE header FROM mydocbold/
  BACKGROUND=red;
STYLE footer FROM mydocbold/
  BACKGROUND=yellow;
STYLE data FROM mydoc/
  BACKGROUND=pink;
STYLE titleandnotecontainer/
  BORDERWIDTH=1;
STYLE systitleandfootercontainer/
  BORDERWIDTH=1;
STYLE systitle FROM systitle/
  BACKGROUND=lightblue;
END;RUN;

```

*Read in the data.

```

*SPLIT PLOT DATA;
DATA CULT;
  INPUT rep cultivar $ inoc $ DRYWT @@;
  CARDS;
  1 A CONT 27.4 1 A DEAD 29.7
  1 A LIVE 34.5 1 B CONT 29.4
  1 B DEAD 32.5 1 B LIVE 34.4
  2 A CONT 28.9 2 A DEAD 28.7
  2 A LIVE 33.4 2 B CONT 28.7
  2 B DEAD 32.4 2 B LIVE 36.4
  3 A CONT 28.6 3 A DEAD 29.7
  3 A LIVE 32.9 3 B CONT 27.2
  3 B DEAD 29.1 3 B LIVE 32.6
  4 A CONT 26.7 4 A DEAD 28.9
  4 A LIVE 31.8 4 B CONT 26.8
  4 B DEAD 28.6 4 B LIVE 30.7
  ;
RUN;

```

Submit the ODS statements.

```

*ODS TRACE on /listing;
ODS TRACE off;
ODS LISTING close;

FILENAME myhtml 'c:\joysas\ods';
ODS HTML PATH=myhtml
  FILE='anova1.html'
  CONTENTS='anova1c.html'
  FRAME='anova1f.html'
  STYLE=styles.mydefault;
*ODS HTML SELECT modelanova;
ODS HTML SHOW;
*ODS PROCLABEL = 'Anova1.sas';

```

Run the ANOVA.

```

TITLE 'DEFAULT ANOVA TABLE';
PROC ANOVA; CLASS rep cultivar inoc;
  MODEL drywt= rep cultivar rep*cultivar inoc
  inoc*cultivar;
  *MEANS cultivar|inoc;
  *MEANS cultivar / WALLER E=rep*cultivar;
  *TEST H=cultivar E=rep*cultivar;
RUN;

```

Edit STAT.GLM.TESTS.

```

PROC TEMPLATE; **Edit the default table definition
for the anova table;
EDIT stat.glm.tests as stat.glm.tests;
HEADER "Anova with
template='sasuser.stat.anova.mytest' ";
MVAR cont sysdate9;
CONTENTS_LABEL=cont;

COLUMN Source DF SS MS FVALUE PROBF STAR;
DEFINE SS;
  PARENT=Stat.glm.ss;
  FORMAT = 10.3; END;
DEFINE MS;
  PARENT=stat.glm.ms;
  FORMAT= 10.3; END;
DEFINE FVALUE;
  PARENT=stat.glm.fvalue; END;
DEFINE PROBF;
  PARENT= stat.glm.probf; END;
DEFINE Star;
  PRE_SPACE=1 WIDTH=3 JUST=I;
COMPUTE as ProbF;
TRANSLATE
  _val_ >0.05 into "",
  _val_ >0.01 into "**",
  _val_ >0.001 into "***",
  _val_ <=0.001 and _val_ >0 into "****"; END;
DEFINE footer table_footer;
  TEXT 'Example 6 -- Report Prepared on ' sysdate9;
END;
END;RUN;

```

Rerun the ANOVA.

```

*RERUN ANOVA WITH NEW TEMPLATE;
TITLE 'CUSTOMIZED ANOVA TABLE';
%LET reportdate=&sysdate9.;
%let cont=CUSTOMIZED ANOVA;
ODS HTML SELECT modelanova;
PROC ANOVA DATA=cult; CLASS rep cultivar inoc;
  MODEL drywt= rep cultivar rep*cultivar inoc
  inoc*cultivar;
  *MEANS cultivar|inoc;
  *MEANS cultivar / WALLER E=rep*cultivar;
  *TEST H=cultivar E=rep*cultivar;
RUN;QUIT;
ODS HTML close;
PROC TEMPLATE;
  DELETE stat.glm.tests ;
run;
DM "wbrowse 'c:\joysas\ods\anova1f.html' ";

```

Conclusion

"If you master the syntax, then it is safe to say you are the undisputed output adventurer at your site." Olinger (1999).

PROC TEMPLATE is a powerful tool for modifying SAS output. It takes a significant amount of time to master

PROC TEMPLATE, but creating simple templates is not hard.

Contact Information

Carol Gosselin -- Carol_gosselin@ncsu.edu
Joy Munk Smith -- Joy_smith@ncsu.edu

Examples are available at

<http://www.stat.ncsu.edu/sas/ods/templates>

References

Bryant, Muller, and Pass (1999), "ODS, YES! Odious, NO! – An Introduction to the SAS Output Delivery System.", in the Proceedings for the Twenty-Fourth Annual SAS Users Group International Conference, Cary, NC: SAS Institute Inc.

Olinger and Tobias (1998), "ODS for Data Analysis: Output As-You-Like-It in Version 7", in the Proceedings

for the Twenty-Third Annual SAS Users Group International Conference, Cary, NC: SAS Institute, Inc.

Olinger (2000), "ODS for Dummies", in the Proceedings for the Twenty-Fifth Annual SAS Users Group International Conference, Cary, NC: SAS Institute Inc.

Olinger (1999), "Twisty Little Passages, All Alike – ODS Templates Exposed", in the Proceedings for the SAS® Institute, The Complete Guide to SAS® Output Delivery System, Version 8, Cary, NC: SAS Institute, Inc.

SAS® (1999), The Complete Guide to the Output Delivery System, Version 8.

Acknowledgements

SAS is a registered trademark of SAS Institute, Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are registered trademarks or trademarks or their respective companies.

Figure 1 (STYLE=Default):

Enrollment by Residency

	SENSTAT				All
	New	Transfer	Continue	Reenter	
	N	N	N	N	
IOF					
In-State	988	448	4917	60	6413
Out-of-State	180	39	285	5	509
International	77	1	312	.	390
All	1245	488	5514	65	7312

Figure 2 (STYLE=Minimal):

Enrollment by Residency

	SENSTAT				All
	New	Transfer	Continue	Reenter	
	N	N	N	N	
IOF					
In-State	988	448	4917	60	6413
Out-of-State	180	39	285	5	509
International	77	1	312	.	390
All	1245	488	5514	65	7312

Figure 3 (STYLE=Mystyle):

Enrollment by Residency

	SENRSTAT				All	
	New	Transfer	Continue	Reenter		
	N	N	N	N	N	
IOF						
In-State	988	448	4917	60	6413	
Out-of-State	180	39	285	5	509	
International	77	1	312	.	390	
All	1245	488	5514	65	7312	

Figure 4 (STYLE=Mystyle):

Enrollment by Residency

	SENRSTAT				All	
	New	Transfer	Continue	Reenter		
	N	N	N	N	N	
IOF						
In-State	988	448	4917	60	6413	
Out-of-State	180	39	285	5	509	
International	77	1	312	.	390	
All	1245	488	5514	65	7312	

Figure 5 (STYLE=Formsword):

Enrollment by Residency

	SENRSTAT				All	
	New	Transfer	Continue	Reenter		
	N	N	N	N	N	
IOF						
In-State	988	448	4917	60	6413	
Out-of-State	180	39	285	5	509	
International	77	1	312	.	390	
All	1245	488	5514	65	7312	

NC STATE UNIVERSITY

Figure 6 (STYLE=Mystyle):

Enrollment by Residency

	SENRSTAT				All	
	New	Transfer	Continue	Reenter		
	N	N	N	N	N	
IOF						
In-State	988	448	4917	60	6413	
Out-of-State	180	39	285	5	509	
International	77	1	312	.	390	
All	1245	488	5514	65	7312	

Figure 7 (STYLE=FrameChanges):

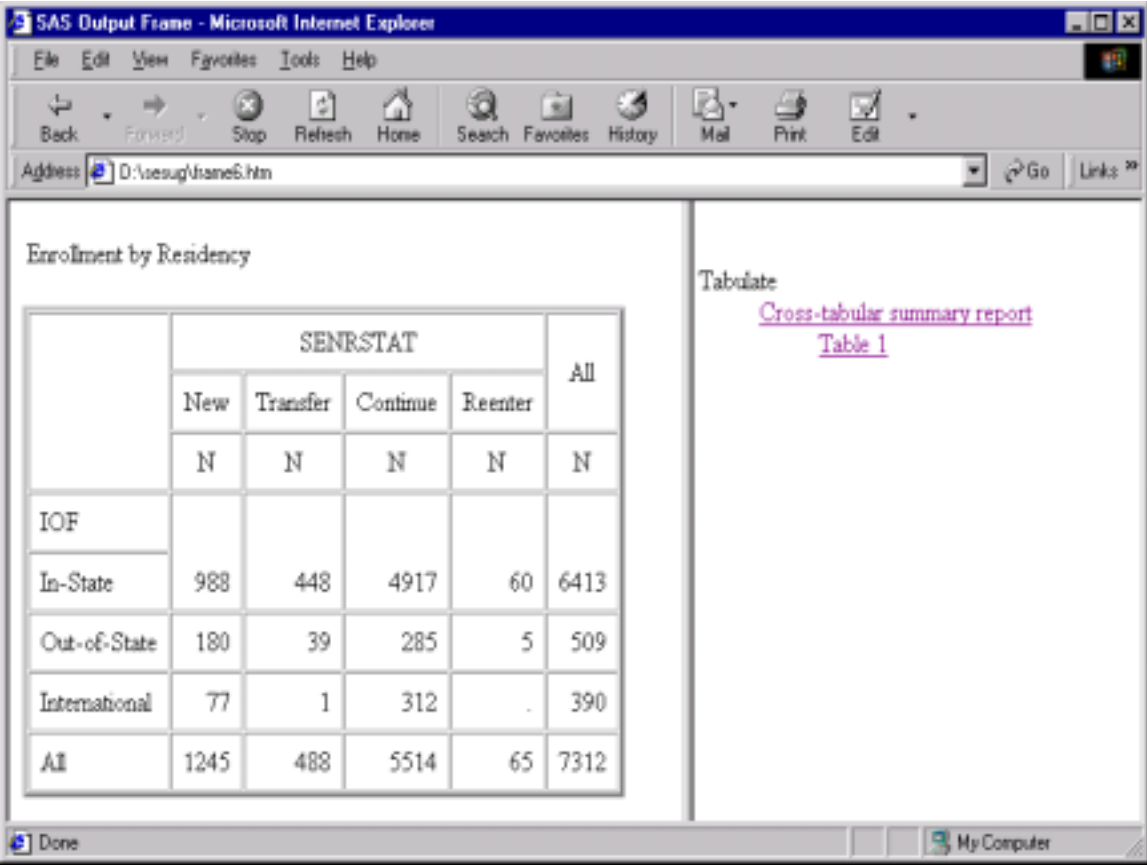


Figure 8.

