

Programming with the Output Delivery System in SAS® Software

Prepared by



*International SAS® Training and
Consulting*

Destiny Corporation • 100 Great Meadow Rd
Suite 601 • Wethersfield, CT 06109-2355

Phone: (860) 721-1684 • 1-800-7TRAINING

Fax: (860) 721-9784

Web: WWW.DESTINYCORP.COM

Email: info@destinycorp.com

1999-05-17

Programming with the Output Delivery System

All of the items seen in the Results window can be coded in SAS. The ODS statement and Proc Template are key items to understand. The general ODS syntax can look like the following.

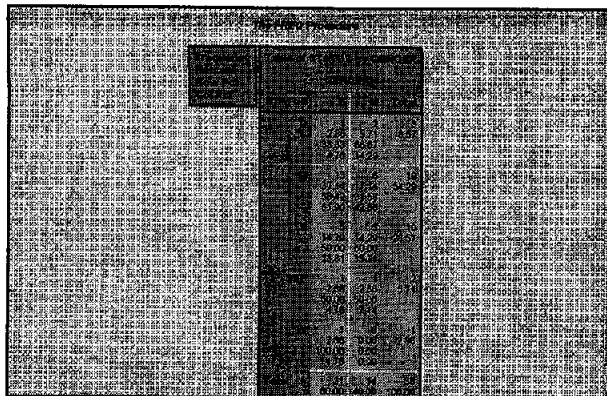
ODS HTML

```

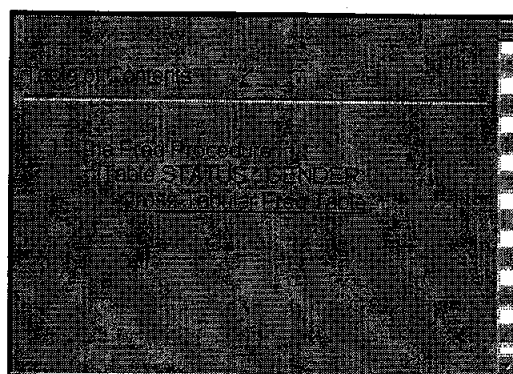
Program Editor - (Untitled)
Command --->
00001 ods html body='c:\body.html'
00002         contents='c:\contents.html'
00003         page='c:\page.html'
00004         frame='c:\frame.html';
00005
00006 proc freq data=saved.demograf;
00007     table status*gender;
00008 run;
00009 ods html close;
00010
00011
    
```

This syntax routes HTML output anywhere you desire. It creates

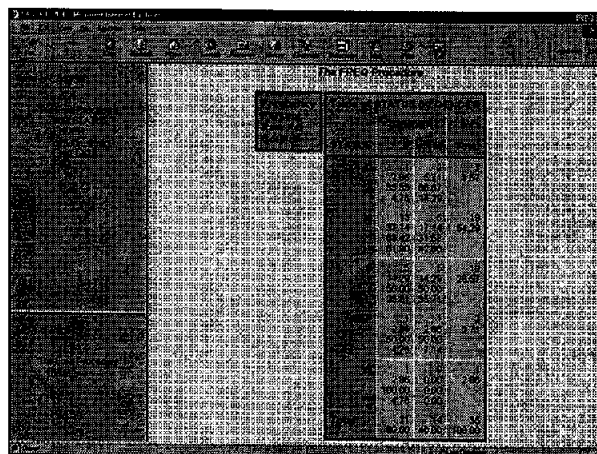
BODY.HTML



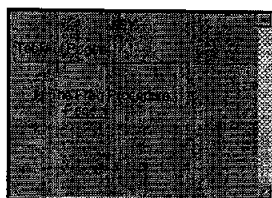
CONTENTS.HTML



FRAME.HTML



PAGE.HTML



There are many types of ODS statements available. Listed below is SAS's Online Help

Overview of ODS Statements	
The ODS statements control all the features of the Output Delivery System. Each ODS statement controls a different feature. You can use these statements anywhere in your SAS program. Some of them remain in effect until you explicitly change them. Others are automatically cleared at particular times (see the documentation for individual statements).	
What Does Each ODS Statement Do?	
Specify output objects to exclude from ODS destinations.	ODS EXCLUDE
Open, manage, or close the HTML destination. If the destination is open, you can cruise HTML procedure output.	ODS HTML
Open, manage, or close the Listing destination.	ODS LISTING
Create a SAS data set from an output object.	ODS OUTPUT
Specify which locations to search for ODS templates and the order in which to search for them.	ODS PATH
Specify output objects to send to ODS destinations.	ODS SELECT
Write to the SAS log the specified selection or exclusion list.	ODS SHOW
Write to the SAS log a record of each output object that is created, or suppress the writing of this record.	ODS TRACE
Print or suppress a warning that the ODS template that is used is not supplied by SAS Institute.	ODS VERIFY

Browse the on-line help to understand the primary aspects of these statements.

First, let's examine TRACE. This is probably one of the most important utilities. It tells you what each step will potentially create.

Program Editor - (Untitled)	
Command ==>	
00001 ods trace on;	
00002 proc tabulate data=saved.demograf;	
00003 class status;	
00004 var salary;	
00005 table status*salary;	
00006 run;	
00007 ods trace off;	
00008	
00009	

Yields this report in the LOG.

Log - (Untitled)	
Command ==>	
205 ods trace on;	
206 proc tabulate data=saved.demograf;	
207 class status;	
208 var salary;	
209 table status*salary;	
210 run;	
Output Added:	
Name: Report	
Label: Cross-tabular summary report	
Data Name: Summary	
Data Label: Summary statistics	
Path: Tabulate.Report	
NOTE: PROCEDURE TABULATE used:	
real time 0.10 seconds	
211 ods trace off;	

The object name is REPORT. (Be careful, they all have different names)

To create a SAS data set from this object, use the following code.

Program Editor - (Untitled)	
Command ==>	
00001 ods output report=work.test;	
00002 proc tabulate data=saved.demograf;	
00003 class status;	
00004 var salary;	
00005 table status*salary;	
00006 run;	
00007 ods output close;	
00008	
00009	

This code creates a SAS data set called TEST in the WORK library.

Log - (Untitled)	
Command ==>	
220 ods output report=work.test;	
221 proc tabulate data=saved.demograf;	
222 class status;	
223 var salary;	
224 table status*salary;	
225 run;	
NOTE: The data set WORK.TEST has 5 observations and 5 variables.	
NOTE: PROCEDURE TABULATE used:	
real time 0.15 seconds	
226 ods output close;	

It looks like this.

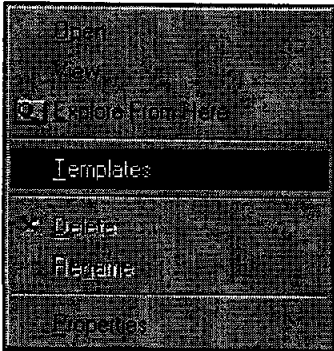
VIEWTABLE work test					
	STATUS	Obs #	Obs #	Obs #	SALARY \$
1	D	1	1	1	28000
2	M	1	1	1	249500
3	S	1	1	1	60800
4	SEP	1	1	1	30000
5	W	1	1	1	30000

Styles in HTML

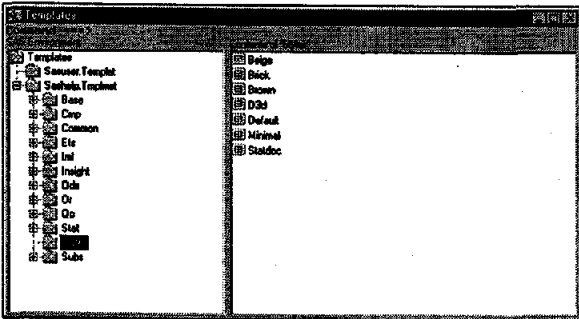
There are several HTML styles that come with SAS. Any of the default ones can be used, or they can be modified to suite your needs. To see the existing styles, select.



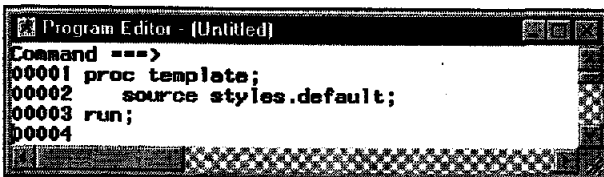
Right click and select Templates.



Select SASHELP.TMPLMST, Styles to see the available styles.



To examine a style, execute Proc Template with the Style name. Default is the default style used.



```
Command ===>
254 proc template;
255   source styles.default;
256   define style Styles.Default;
257     style fonts /
258       'TitleFont2' = ('Arial, Helvetica, Helv',4,Bold Italic)
259       'TitleFont' = ('Arial, Helvetica, Helv',5,Bold Italic)
260       'StrongFont' = ('Arial, Helvetica, Helv',4,Bold)
261       'EmphasisFont' = ('Arial, Helvetica, Helv',3,Italic)
262       'FixedEmphasisFont' = ('Courier',3,Italic)
263       'FixedStrongFont' = ('Courier',4,Bold)
264       'FixedHeadingFont' = ('Courier',4)
265       'FixedFont' = ('Courier',4)
266       'HeadingEmphasisFont' = ('Arial, Helvetica, Helv',4,Bold Italic)
267       'HeadingFont' = ('Arial, Helvetica, Helv',4,Bold)
268       'DocFont' = ('Arial, Helvetica, Helv',3);
269     style color_list /
270       'fg2' = cxx0055aa
271       'fg3' = cxx004488
272       'fg4' = cxx00ff77
273       'bg4' = cxx999999
274       'bg2' = cxx003300
275       'fg2' = cxx0033aa
276       'bg2' = cxx000000
277       'fg1' = cxx000000
278       'bg1' = cxxff0000
279       'fg' = cxx002288
280       'bg' = cxx000000;
281     style colors /
282       'headerfgemph' = color_list('fg2')
283       'headerbgemph' = color_list('bg2')
284       'headerfgstrong' = color_list('fg2')
285       'headerbgstrong' = color_list('bg2')
286       'headerfg' = color_list('fg2')
287       'headerbg' = color_list('bg2')
288       'datafgemph' = color_list('fg2')
289       'databgemph' = color_list('bg2')
290       'datafgstrong' = color_list('fg2')
291       'databgstrong' = color_list('bg2')
292       'datafg' = color_list('fg1')
293       'databg' = color_list('bg1')
294       'batchbg' = color_list('bg2')
295       'batchfg' = color_list('fg2')
296       'tableborder' = color_list('fg1')
297       'tablebg' = color_list('bg1')
298       'notefg' = color_list('fg')
299       'notebg' = color_list('bg');
300   end;
301 Command ===>
302 bylinebg = color_list('bg2')
303 captionfg = color_list('fg1')
304 captionbg = color_list('bg1')
305 proctitlefg = color_list('fg1')
306 proctitlebg = color_list('bg1')
307 titlefg = color_list('fg1')
308 titlebg = color_list('bg1')
309 mytitlefg = color_list('fg1')
310 mytitlebg = color_list('bg1')
311 Conentryfg = color_list('fg1')
312 Conentrybg = color_list('fg1')
313 Contitlefg = color_list('fg1')
314 link2 = color_list('fg2')
315 link1 = color_list('fg1')
316 contentfg = color_list('fg2')
317 contentbg = color_list('bg2')
318 docfg = color_list('fg1')
319 docbg = color_list('bg1');
320 style list /
321   'expandAll' = '<SPAN onClick="if(asis4==1)expandAll(1)">'
322   'posthtml flyover line' = '</SPAN><HR size=3>'
323   'prehtml flyover line' = '<SPAN><HR size=3>'
324   'prehtml flyover bullet' = '<SPAN><b>&#183;</b>'
325   'posthtml flyover' = '</SPAN>'
326   'prehtml flyover' = '<SPAN>'
327   'break' = '<br>'
328   'line' = '<HR size=3>'
329   'fake bullet' = '<b>&#183;</b>';
330 style text /
331   Fatal Banner = 'FATAL:'
332   Error Banner = 'ERROR:'
333   Warn Banner = 'WARNING:'
334   Note Banner = 'NOTE:'
335   Pages Title = 'Table of Pages'
336   Content Title = 'Table of Contents'
337   suffix1 = 'Procedure'
338   prefix1 = 'The ';
339 style Container
340   'Abstract style for basic colors and font.' /
341   font = FontList('DocFont')
342   foreground = colors('docfg')
343   background = colors('docbg');
344 style Index from Container
345   'Abstract style for Contents and pages.' /
346   foreground = colors('contentfg')
347   background = colors('contentbg');
```

```

Command ===
htmldoctype = '<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN'
htmlcontenttype = 'text/html'
prolapsedspecialchars = auto
linkcolor = colors('link2')
visitedlinkcolor = colors('link1');
style Body from Document /
leftmargin = 8
pagebreakital = html('Line');
style Frame from Document /
contentposition = L
bodyacrolbar = auto
bodysize = 4
contentacrolbar = auto
contentsize = 25%
framespacing = 1
frameborderwidth = 4
frameborder = on;
style Contents from Document /
bullet = 'upper-roman'
pagebreakital = html('break')
foreground = colors('contentfg')
background = colors('contentbg')
leftmargin = 8;
style Pages from Document /
bullet = 'upper-roman'
pagebreakital = html('break')
foreground = colors('contentfg')
background = colors('contentbg')
leftmargin = 8;
style Flyover /
bordercolor = cccccc00
borderwidth = 1px
foreground = colors('contentfg')
background = colors('contentbg');
style Date from Container /
outputwidth = 100%
background = colors('contentbg');
foreground = colors('contentfg');
style BodyDate from Date /
background = colors('docbg')
foreground = colors('docfg');
style ContentsDate from Date;
style PagesDate from Date;
style Indexites from Container /
prehtml = html('fake bullet')
listentryanchor = on
ulclass = 'INDEX';
Command ===
foreground = colors('contentfg');
style ContentFolder from Indexites /
posthtml = html('posthtml flyover')
prehtml = html('prehtml flyover bullet')
listentryanchor = off
foreground = colors('contentfg');
style ContentItem from Indexites;
style PagesItem from Indexites;
style IndexProcName from Index /
listentryanchor = off
bullet = 'upper-roman'
posthtml = html('posthtml flyover')
prehtml = html('prehtml flyover')
posttext = text('suffix')
pretext = text('prefix')
foreground = colors('contentfg');
style ContentProcName from IndexProcName;
style PagesProcName from IndexProcName;
style IndexAction from Index;
style FolderAction from IndexAction;
style ItemAction from IndexAction;
style ProcNameAction from IndexAction;
style TitleAction from IndexAction;
style IndexTitle from Index /
posthtml = html('posthtml flyover line')
prehtml = html('expandall')
font = fonts('EmphasisFont')
foreground = colors('contentfg');
style ContentTitle from IndexTitle /
pretext = text('content title');
style PagesTitle from IndexTitle /
pretext = text('pages title');
style SysTitleAndFooterContainer from Container /
rules = NONE
frame = VOID
outputwidth = 100%
cellpadding = 1
cellspacing = 1
borderwidth = 0;
style TitleAndNoteContainer from Container /
rules = NONE
frame = VOID
outputwidth = 100%
cellpadding = 1
cellspacing = 1
borderwidth = 0;
style Table from Container /

```

```

Command ===
font = Fonts('TitleFont2')
background = colors('systitlebg')
foreground = colors('systitlefg');
style BylineContainer from Container /
background = colors('docbg')
rules = NONE
frame = VOID
outputwidth = 100%
cellpadding = 1
cellspacing = 1
borderwidth = 0;
style SysTitle from TitleAndFooters /
font = Fonts('TitleFont');
style SystemFooter from TitleAndFooters /
font = Fonts('TitleFont');
style Byline from TitleAndFooters /
font = fonts('headingFont');
background = colors('bylinebg')
foreground = colors('bylinefg');
style ProcTitle from TitleAndFooters /
background = colors('proctitlebg')
foreground = colors('proctitlefg');
style ProcTitleFixed from ProcTitle /
font = fonts('FixedStrongFont');
style Output from Container /
background = colors('tablebg')
rules = GROUPS
frame = BOX
cellpadding = 7
cellspacing = 1
bordercolor = colors('tableborder')
borderwidth = 1;
style Table from Output
Data Table style.;
style Batch from Output
Batch (capture) Output style. /
font = fonts('FixedFont')
foreground = colors('batchfg')
background = colors('batchbg');
style Graph from Output
Graph style.;
cellpadding = 0
cellspacing = 0
background = colors('docbg');
style Note from Container /
background = colors('notdbg');
Command ===
style NoteBanner from Note /
pretext = text('Note Banner');
style NoteContent from Note;
style NoteContentFixed from NoteContent /
font = fonts('FixedFont');
style WarnBanner from Note /
pretext = text('Warn Banner');
style WarnContent from Note;
style WarnContentFixed from Note /
font = fonts('FixedFont');
style ErrorBanner from Note /
pretext = text('Error Banner');
style ErrorContent from Note;
style ErrorContentFixed from Note /
font = fonts('FixedFont');
style FatalBanner from Note /
pretext = text('Fatal Banner');
style FatalContent from Note;
style FatalContentFixed from Note /
font = fonts('FixedFont');
style Cell from Container;
style Data from Cell /
foreground = colors('datafg')
background = colors('databg');
style DataFixed from Data /
font = fonts('FixedFont');
style DataEmpty from Data;
style DataEmphasis from Data /
foreground = colors('datafgeph')
background = colors('databgeph')
font = fonts('EmphasisFont');
style DataEmphasisFixed from DataEmphasis /
font = fonts('FixedEmphasisFont');
style DataStrong from Data /
foreground = colors('datafgstrong')
background = colors('databgstrong')
font = fonts('StrongFont');
style DataStrongFixed from DataStrong /
font = fonts('FixedStrongFont');
style HeadersAndFooters from Cell /
font = fonts('HeadingFont')
foreground = colors('headerfg')
background = colors('headerbg');
style Caption from HeadersAndFooters /
foreground = colors('captionfg')
background = colors('captionbg');

```

```

Command ==>
style AfterCaption from Caption;
style Header from HeadersAndFooters;
style HeaderFixed from Header /
  font = font('FixedStrongFont');
style HeaderEmpty from Header;
style HeaderEmphasis from Header /
  foreground = colors('headerfemph');
  background = colors('headerbemph');
  font = font('EmphasisFont');
style HeaderEmphasisFixed from HeaderEmphasis /
  font = font('FixedEmphasisFont');
style HeaderStrong from Header /
  foreground = colors('headerfstro');
  background = colors('headerbstro');
  font = font('StrongFont');
style HeaderStrongFixed from HeaderStrong /
  font = font('FixedStrongFont');
style RowHeader from Header;
style RowHeaderFixed from RowHeader /
  font = font('FixedFont');
style RowHeaderEmpty from RowHeader;
style RowHeaderEmphasis from RowHeader /
  font = font('EmphasisFont');
style RowHeaderEmphasisFixed from RowHeaderEmphasis /
  font = font('FixedEmphasisFont');
style RowHeaderStrong from RowHeader /
  font = font('StrongFont');
style RowHeaderStrongFixed from RowHeaderStrong /
  font = font('FixedStrongFont');
style Footer from HeadersAndFooters;
style FooterFixed from Footer /
  font = font('FixedFont');
style FooterEmpty from Footer;
style FooterEmphasis from Footer /
  font = font('EmphasisFont');
style FooterEmphasisFixed from FooterEmphasis /
  font = font('FixedEmphasisFont');
style FooterStrong from Footer /
  font = font('StrongFont');
style FooterStrongFixed from FooterStrong /
  font = font('FixedStrongFont');
style RowFooter from Footer;
style RowFooterFixed from RowFooter /
  font = font('FixedFont');
style RowFooterEmpty from RowFooter;
style RowFooterEmphasis from RowFooter /
  font = font('EmphasisFont');
style RowFooterEmphasisFixed from RowFooterEmphasis /
  font = font('FixedEmphasisFont');
style RowFooterStrong from RowFooter /
  font = font('StrongFont');
style RowFooterStrongFixed from RowFooterStrong /
  font = font('FixedStrongFont');
end;
256 run;
NOTE: PROCEDURE TEMPLATE used:
      real time      0.44 seconds

```

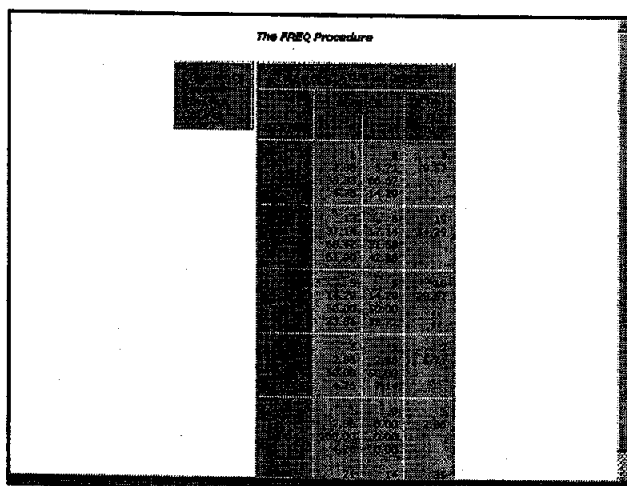
To change a style, use the ODS statement

```

Program Editor - (Untitled)
Command ==>
00001 ods html body='c:\body.html' style=styles.statdoc;
00002 proc freq data=saved.demograf;
00003   table status*gender;
00004 run;

```

It produces the following HTML. (Looks different than styles.default)



See the online help to learn more about changing styles and what the values mean (See Output Delivery System, Customizing the Styles Used in HTML Output triE373.HTM)

No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Destiny Corporation. ©1999 Destiny Corporation. All rights reserved.

Using Proc Template

Templates come with SAS and can be built for a specific appearance using HTML. A simple template example would be SASHELP.TMPLMST.BASE.OneWayFreqs.

OneWayFreqs

```

proc template;
  link Base.Freq.OneWayFreqs to
  Base.Freq.OneWayList / notes =
    'One-Way Frequency table';
  define table Base.Freq.OneWayList;
    notes 'Parent for One-Way Frequency table and
    LIST table';
    dynamic page needlines plabel varlabel lw
    varjust varname gluef gluep;

```

```

    column Line Variable ListVariable Frequency
  TestFrequency Percent

```

```

    TestPercent CumFrequency CumPercent;

```

```

header h1;

```

```

translate _val_=_ into '';

```

```

define h1;

```

```

  text varlabel;

```

```

  split = '';

```

```

  space = 1;

```

```

  spill_margin;

```

```

  highlight;

```

```

end;

```

```

define Line;

```

```

  header = 'Line';

```

```

  just = c;

```

```

  format_width = lw;

```

```

  format_ndec = 0;

```

```

  id;

```

```

end;

```

```

define Variable;

```

```

  header = varname;

```

```

  just = varjust;

```

```

  id;

```

```

end;

```

```

define ListVariable;

```

```

  just = varjust;

```

```

  generic;

```

```

end;

```

```

define Frequency;

```

```

  header = 'Frequency';

```

```

  just = c;

```

```

  format = BEST8.;

```

```

  glue = gluef;

```

```

end;

```

```

define TestFrequency;

```

```

  header = '\ Test \Frequency\';

```

```

  just = c;

```

```

  format = BEST8.;

```

```

  glue = 4;

```

```

end;

```

```

define Percent;

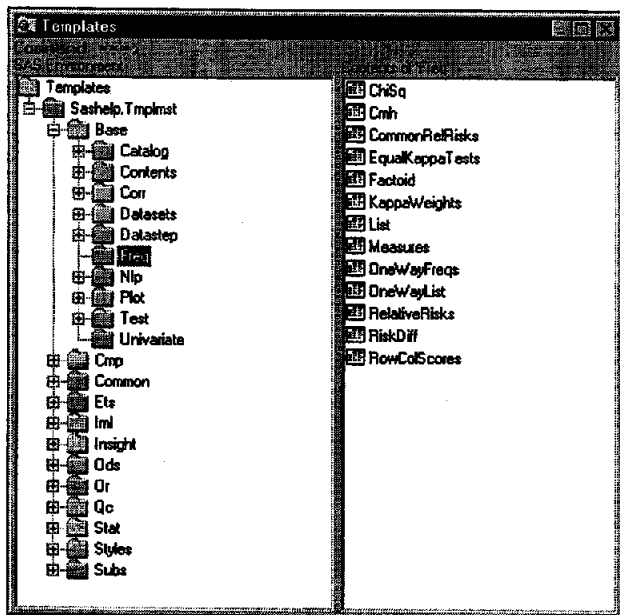
```

```

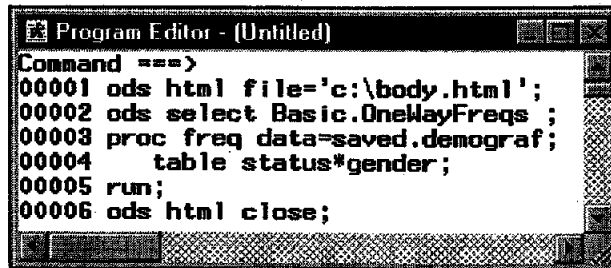
header = '\ Percent\';
just = c;
format = 6.2;
glue = gluep;
end;
define TestPercent;
header = '\ Test\ Percent\';
just = c;
format = 6.2;
glue = 3;
end;
define CumFrequency;
header = '\Cumulative\ Frequency\';
just = c;
format = BEST8.;
glue = 4;
end;
define CumPercent;
header = '\Cumulative\ Percent\';
just = c;
format = 6.2;
end;
required_space = needlines;
newpage = page;
print_headers = plabel;
underline;
use_name;
end;
run;

```

To see existing templates that come with SAS, in the Results window, click on SASHELP.TMPLMST, Base, and Freq to see the following templates.



The syntax for using a template is documented as being.



For more information, Destiny Corporation offers the following courses:

- *Version 7/8 Changes and Enhancements*
- *Using the Output Delivery System in SAS Software.*