

Paper 149-25

ODS, YES! Odious, NO! – An Introduction to the SAS Output Delivery System

Lara Bryant, University of North Carolina at Chapel Hill, Chapel Hill, NC
 Sally Muller, University of North Carolina at Chapel Hill, Chapel Hill, NC
 Ray Pass, Ray Pass Consulting, Hartsdale, NY

ABSTRACT

ODS (originally pronounced ‘odious’, but now pronounced ‘ya gotta love it’) is the new SAS system facility, starting with Version 7, that you can use to format your PROC and DATA output in ways just recently only dreamed about. ODS offers greatly enhanced flexibility and ease of use for both new SAS users and experienced SAS users new to ODS. This paper will discuss the basics of ODS, emphasizing methods of converting standard PROC output to the following destinations:

- **Listing** - default (the only way to get PROC output up to now)
- **HTML** - HyperText Markup Language (probably the best tool available for information exchange today)
- **Output** - SAS data sets (no more PROC PRINTTO!)

And, although not covered in this paper, these destinations are also available as experimental:

- **RTF** - Rich Text Format (for importing into MS Word)
- **PS/PDF/PCL** - Post Script, Portable Document Format, Printer Control Language (for high fidelity printers)
- **LaTeX** - text formatter
- **XML**

In addition, this presentation will demonstrate how you select the results to send to each destination, and the syntax for each destination. We will touch on customizing the default ODS procedure output templates and creating user-defined ones. By the end of the presentation you will have a working knowledge of ODS and feel comfortable enough to easily create at least two new kinds of output in SAS!

INTRODUCTION

Creating output objects that can be sent to the above new “destinations” (e.g. HTML) is often just a matter of running procedures in your existing SAS program with just a few extra lines of code (sometimes only one line). When you run a PROC or DATA step, ODS combines the resulting data with a template (or table definition) to create an output object, or a series of output objects coming from various parts of the procedure’s output. ODS allows you to choose specific output objects created from a PROC or DATA step to send to an output destination. ODS provides default table definitions specific to most procedures and for the DATA step. You can also create or modify your own table definition with the PROC TEMPLATE command. The output object is formatted according to its content and the destination you send it to. You can even send your output to more than one destination. For example, you can create an output data set from a PROC MEANS that is also displayed on an HTML page.

OPENING AND CLOSING DESTINATIONS

The Listing, HTML, and Output destinations can be open or closed. By default, the Listing destination is open, and the HTML and Output destinations are closed. The syntax for opening the Listing destination is:

```
ods listing;
```

The commands for opening the HTML and Output destinations are more detailed, and therefore are presented in the overview of each destination. To close a destination, the syntax is

```
ods <destination> close;
```

You may want to close the Listing destination to free up resources that would otherwise be used to send output objects to this destination.

SELECTION AND EXCLUSION LISTS

The SAS system maintains a list of the objects that can be sent to each destination. If you are selecting objects to send to a specific destination, SAS maintains a SELECTION list for that destination. If you are selecting objects that you do not want sent to a destination, SAS maintains an EXCLUSION list for that destination. Generally you need only select or exclude objects for a particular destination, rather than trying to maintain both a SELECTION and an EXCLUSION list. ODS also maintains an overall SELECTION or EXCLUSION list. In order for an object to be sent to a destination, it must be on both the overall SELECTION list, and the SELECTION list for that particular destination.

There are two ways that these SELECTION and EXCLUSION lists can be modified:

- explicit modification from a command by the user
- automatic modification by ODS at certain points (step boundaries) in the SAS program

Explicit Modification

To explicitly modify the SELECTION and EXCLUSION lists, you may use the following syntax:

```
ods <options>;
ods listing <options>;
ods html <options>;
```

where the options are

```
select <selections>
select all
select none
exclude <exclusions>
exclude all
exclude none
```

The default values for the destinations are as follows:

```
Overall list - select all
Listing destination - select all
HTML destination - select all
Output destination - exclude all
```

Changing the overall list is helpful if you want to exclude an object from all destinations. For example, rather than typing

```
ods html exclude all;
ods listing exclude all;
```

You could simply type:

```
ods exclude all;
```

Automatic Modification

When you do NOT explicitly modify a SELECTION or EXCLUSION list, ODS automatically sets defaults as noted above at every step boundary. When you do use explicit modification, ODS, by default, maintains the modifications for *one use only*, reverting back to defaults at the boundary. This can be overcome by using the PERSIST option. For example, suppose you had the following code:

```
ods listing select BasicMeasures;
proc univariate data='A:\meddat';
run;
proc univariate data='A:\meddat';
run;
```

ODS would select only the “BasicMeasures” statistics for the first PROC UNIVARIATE. The RUN statement ends the procedure (this is a step boundary, but even if you did not specify the RUN statement, the beginning of the second PROC UNIVARIATE would end the first PROC UNIVARIATE). Either way, you would only have “BasicMeasures” printed for the first PROC. After the first PROC, the list is automatically set to its default value, which is SELECT ALL for the default Listing destination. The second PROC would therefore include all statistics generated by the PROC UNIVARIATE. Obviously, if you only wanted “BasicMeasures” throughout, it would be tedious to have to specify the desired list after every procedure. ODS provides a way around this. By adding the PERSIST option to the SELECT (EXCLUDE) statement, you only have to specify the SELECTION (EXCLUSION) list once for it to be maintained throughout the program (or at least until the next encountered SELECT or EXCLUDE command). For example, if we now if we run the following program

```
ods listing select BasicMeasures (persist);

proc univariate data='A:\meddat';
run;
proc univariate data='A:\meddat';
run;
```

the BasicMeasures statistics will be selected for both PROC UNIVARIATES.

The PERSIST option can also be used for an HTML list as in:

```
ods html select BasicMeasures (persist);
```

The syntax for the Output destination is more involved, and is explained in that section.

In the previous examples, a RUN statement would end the PROC and reset the SELECTION (EXCLUSION) list to the default value if the PERSIST option was not specified. However, there are several procedures that are not terminated by a RUN statement (RUN processing), such as DATASETS, PROC GLM, and PROC REG. In these cases, unless a QUIT statement is encountered, the PROC will continue to run. This may produce some unexpected results on your SELECTION (EXCLUSION) list. For example, consider the following (the FILE= option is discussed below in “File types”):

```
ods html file='A:\new.htm';
ods html select Anova;

proc reg data='A:\aggrec';
  model inpdol=age;
run;

ods html select FitStatistics;

proc reg data='A:\aggrec';
  model outpdol=age;
run;

ods html close;
```

In the program above ODS would create “Anova” statistics for the first PROC REG. This would remain intact through the RUN statement because a RUN statement does not end a running PROC REG. When ODS reaches the second PROC REG, it would end the first PROC and set the SELECTION list to its default value of SELECT ALL. Therefore, rather than having the desired “FitStatistics” for the last PROC REG, ODS would create ALL the statistics. The simple solution is to specifically terminate the first PROC REG with a QUIT statement as follows (the SHOW statement is discussed below):

```
ods html file='A:\new.htm';
ods html select Anova;

proc reg data='A:\aggrec';
  model inpdol=age;
run;
```

```
ods html show;
quit;

ods html show;
ods html select FitStatistics;

proc reg data='A:\aggrec';
  model outpdol=age;
run;

ods html show;
quit;

ods html close;
```

This program produces the desired results: “Anova” statistics for the first PROC REG and “FitStatistics” for the second PROC REG

ODS SHOW STATEMENT

The ODS SHOW statement is a way to see what ODS has on the SELECTION (EXCLUSION) list for specified destinations at any point in your program. The following statement,

```
ods <destination> show;
```

requests that the SELECTION (EXCLUSION) list for a particular destination appear in the log. If no destination is specified, the OVERALL list is displayed.

In the example immediately above, the log would contain

```
Current HTML select list is:
1. Anova
```

after the first SHOW statement, and

```
Current HTML select list is:
1. FitStatistics
```

after the last one.

ODS TRACE STATEMENT

As you have seen, ODS allows you to choose specific output objects created from a PROC or DATA step to send to an output destination. The example below of PROC UNIVARIATE illustrates that ODS will create an output object for each statistical procedure for each variable (only one variable is specified in this example).

Part of the power of ODS is that you can indicate which output objects to create, and even tell ODS to send the output objects created by the same procedure to different destinations. For example, rather than all of the statistics, you may want only the mean, standard deviation, and median generated by the PROC UNIVARIATE. However, in order to specify which output objects to select, you must know the name of the object produced by your SAS program. ODS provides a method of viewing the name of each output object created. The command

```
ods trace on </ label> ;
```

displays a record of the path name of each output object produced by the program in the SAS log. The LABEL option instructs ODS to include the label path in the log. For example, suppose you would like to know the name of each output object created by a PROC UNIVARIATE. The program below will write the TRACE records to the log.

```
ods trace on / label;
proc univariate;
  var meddol;
run;
ods trace off;
```

The SAS Log that results from this program would look like:

Output Added:

```
-----
Name:           Moments
Label:          Moments
Template:       base.univariate.Moments
Path:           Univariate.meddol.Moments
Label Path:     "The Univariate Procedure"."meddol".
               "Moments"
-----
```

Output Added:

```
-----
Name:           BasicMeasures
Label:          Basic Measures of Location and
               Variability
Template:       base.univariate.Measures
Path:           Univariate.meddol.BasicMeasures
Label Path:     "The Univariate Procedure"."meddol".
               "Basic Measures of Location and
               Variability"
-----
```

Output Added:

```
-----
Name:           TestsForLocation
Label:          Tests For Location
Template:       base.univariate.Location
Path:           Univariate.meddol.TestsForLocation
Label Path:     "The Univariate Procedure"."meddol".
               "Tests For Location"
-----
```

Output Added:

```
-----
Name:           Quantiles
Label:          Quantiles
Template:       base.univariate.Quantiles
Path:           Univariate.meddol.Quantiles
Label Path:     "The Univariate Procedure"."meddol".
               "Quantiles"
-----
```

Output Added:

```
-----
Name:           ExtremeObs
Label:          Extreme Observations
Template:       base.univariate.ExtObs
Path:           Univariate.meddol.ExtremeObs
Label Path:     "The Univariate Procedure"."meddol".
               "Extreme Observations"
-----
```

You can then specify the output object by using the full path name, or by using any part of the path that begins immediately after a period and continuing to the end. For example, if you want to send the Quantiles and Moments for all variables to a web page, you would type

```
ods html select quantiles moments;
```

or if you just want the Quantiles and Moments for the MEDDOL variable only:

```
ods html select meddol.quantiles
               meddol.moments;
```

The label path can be used in the same way. You can also specify an output object with a mixture of labels and paths, such as

```
ods html select meddol."quantiles";
```

Often it is easier to select the variables in the PROC step, and the desired statistics in the ODS step. For example, rather than typing

```
ods html select meddol.quantiles
               inpdol.quantiles
               hosp.quantiles
               ambul.quantiles;
```

```
proc univariate;
run;
```

an easier method that gives the same results would be

```
ods html select quantiles;

proc univariate;
   var meddol inpdol hosp ambul;
run;
```

The ODS TRACE ON command will continue to write records of the path names to the log until you type the command ODS TRACE OFF; even if you start new PROCs. If you include both the ODS TRACE ON and ODS TRACE OFF commands in the same program, you must have a RUN statement between them for the records to be sent to the log.

ODS HTML DESTINATION

File types

The HTML destination can produce four kinds of files (web pages):

1) **BODY** file: This is a required file that displays the output object(s) generated from the PROC or DATA steps. Basically, this is where you store the results that you will ultimately display on your HTML report or web site. If your SAS job creates an output object that is not a graphic object, but is routed to an HTML destination, ODS places the results within HTML <TABLE> tags, where they are stored as one or more HTML tables. The BODY file has an (image) tag that references graphic output objects. It is important to note that the BODY file can be specified with either the BODY= or the FILE= option.

2) **CONTENTS** file: The BODY file contains output objects created by ODS from the PROC or DATA steps in your program. The CONTENTS file contains a link to each of the output objects that are stored in the BODY file, and is specified by the CONTENTS= option.

3) **PAGE** file: You can use the NEWFILE option (explained below) to tell ODS to create separate BODY files for your output, based on a certain specification. This is useful if you have a lot of output, and you do not want it to all be stored on one long page. For example, you can request that ODS create a new BODY file for each output object. The PAGE file contains a link to each separate page (of the BODY file) of HTML output that ODS creates from a PROC or DATA step. The PAGE file is similar to the CONTENTS file, except that the CONTENTS file has a link to each output object, whereas the PAGE file has a link to each page of output that is created. The CONTENTS and PAGE files will be identical if you specify in the NEWFILE option that you would like each output object placed on a separate BODY file. You specify the PAGE file with the PAGE= option.

4) **FRAME** file: Provides a simultaneous view of all files included in the ODS HTML statement. You specify the FRAME file with the FRAME= option.

The syntax for creating the above files is

```
ods html <file-type> = 'file-specification'
               <(options)>;
```

Here is an example of ODS HTML statements which generate a BODY file and a CONTENTS file:

```
ods html body      = 'c:\temp\body.htm'
               contents = 'c:\temp\contents.htm';
```

In the above code, the BODY file could also have been specified with a FILE= option.. It is important to note that the BODY file, and only the BODY file, is *required* as an HTML output destination.

HTML Options

1) **PATH=** option: You use the PATH= option to tell ODS where to store all the HTML files that you create. The PATH= option may refer to an external (quoted) file specification, a SAS fileref or a SAS libname.catalog. For example,

```
ods html path = 'C:\MyDocuments'
      body = 'body.htm'
      contents = 'contents.htm';
```

2) **URL=** sub-option: You can improve on the PATH= option by including a Uniform Resource Locator (URL) sub-option that will use the given URL instead of the file name for all the links and references that it creates to the file. This is very helpful if you want to create a FRAME file, and/or will be moving the files around. For example:

```
ods html path = 'C:\MyDocuments'
      (url = 'http://www.unc.edu/~jismith')
      body = 'body.htm'
      contents = 'contents.htm';
```

Note that the URL= sub-option of the PATH= option is enclosed in parentheses.

You can also specify the URL= sub-option in the option for the BODY file, as in the following:

```
ods html path = 'C:\MyDocuments '
      body = 'body.htm'
      (url = 'http://www.unc.edu/~jismith');
```

The results will be identical.

3) **ANCHOR=** option: Each output object in the BODY file is identified by an HTML <ANCHOR> tag. These anchor tags allow the CONTENTS, PAGE and FRAME files to link to or reference the output objects in the BODY file. You can change the base name for the HTML anchor tags with the ANCHOR= option. The syntax for this option is

```
anchor = 'anchor-name'
```

Since each anchor name in a file must be unique, ODS will automatically “increment” the name that you specify. For example, if you specify

```
Anchor = 'tabulate'
```

ODS names the first anchor TABULATE. The second anchor is named TABULATE1; the third is named TABULATE2, and so on. The anchor names are only of interest to you if you need to write to the HTML page; otherwise you need not concern yourself with them. However, you do need to remember to *always* specify a new anchor name each time you open the BODY file so that the same anchor tags are not written to the file again.

4) **NO_TOP_MATTER** and **NO_BOTTOM_MATTER** options: These options circumvent the default action of writing some HTML code to the top and bottom of the file that is open for HTML output. The benefit of these options is that the HTML BODY page is “cleaner” when viewed by the browser.

5) **Descriptive text** options: These options allow you to include comments in between the output of your PROCs. You specify these options inside parentheses next to the BODY=, CONTENTS=, PAGE=, or FRAME= options. Adding comments to your HTML page is helpful for many reasons, as for example you might like to point out some of the interesting results you obtained.

EXAMPLE 1. Putting it all together. The following code places output from several procedures on the same HTML page and uses many of the HTML options discussed above - including putting descriptive text in between the output objects. The SAS statements are numbered for comments following the code:

```
1) libname health 'C:\Data';
2) filename web 'C:\Data\body.htm';
3) ods listing close;
4) ods html path = 'C:\Data'
      (url = 'http://www.unc.edu/~jismith/')
      body = web (no_bottom_matter);
5) proc univariate data=health.meddat;
      var inpdol outpdol;
6) run;
7) ods html close;
8) filename web 'C:\Data\body.htm' mod;
9) data _null_;
10) file web;
11) put '<h3> We want to put comments in
      after the first procedure. </h3>';
12) run;
13) ods html body = web (no_top_matter
      no_bottom_matter)
      anchor = 'univ';
14) proc freq data=health.meddat;
      table site;
15) run;
16) ods html close;
17) data _null_;
      file web;
      put '<h3> We also want comments after
      the second procedure is run. </h3>';
18) run;
19) ods html body = web (no_top_matter)
      anchor = 'freq';
20) ods html close;
```

And now the comments:

(1) Identifies the location of the SAS catalog (C:\Data) containing the SAS data set used for the PROCs.

(2) The FILENAME statement creates a fileref (WEB) to the file where all the BODY output will be stored. Recall the default list for HTML is SELECT ALL. Since no selection commands are specified, everything included in the program will be sent to the HTML file at 'C:\Data\body.htm'

(3) The Listing destination is closed to free up resources.

(4) The NO_BOTTOM_MATTER option suppresses any default HTML at the bottom of 'C:\Data\body.htm'

(5) All statistics created by PROC UNIVARIATE will be generated for the variables INPDOL and OUTPDOL.

(6) Remember that a RUN statement goes after the PROC, and before closing the HTML destination.

(7) The HTML destination must be closed to append to it later.

(8) This references the BODY file used above, and MOD indicates that we want to add to the file.

(9) This DATA _NULL_ step writes some descriptive HTML code to the BODY file via the PUT and FILE statements.

(13) This opens the HTML destination 'C:\Data\body.htm' as identified by the fileref WEB, and suppresses any default HTML code on the top and bottom of the file. The ANCHOR= option creates a base name for the HTML anchor tags. You should *always* specify a new anchor name each time you open the BODY location so that the same anchor tags are not written to the file again.

(19) Open the HTML destination again in order for the new output to be written to the HTML file. The ANCHOR statement provides a new base name.

The resulting HTML page is shown below. The name of the HTML file that is created is 'body.htm' and it is stored at 'C:\Data'

The SAS System

The UNIVARIATE Procedure
Variable: INPDOL

Moments			
N	1000	Sum Weights	1000
Mean	247.50903	Sum Observations	247509.03
Std Deviation	1354.43565	Variance	1834495.93
Skewness	9.68499218	Kurtosis	119.342445
Uncorrected SS	1893922159	Corrected SS	1832661439
Coeff Variation	547.226762	Std Error Mean	42.831016

Basic Statistical Measures			
Location		Variability	
Mean	247.5090	Std Deviation	1354
Median	0.0000	Variance	1834496
Mode	0.0000	Range	23018
		Interquartile Range	0

Tests for Location: Mu0=0				
Test	Statistic		p Value	
Student's t	T	5.778734	Pr > t	<.0001
Sign	M	43.5	Pr >= M	<.0001
Signed Rank	S	1914	Pr >= S	<.0001

Quantiles (Definition 5)	
Quantile	Estimate
100% Max	23018.39
99%	5618.30
95%	1538.71
90%	0.00
75% Q3	0.00
50% Median	0.00
25% Q1	0.00
10%	0.00
5%	0.00
1%	0.00
0% Min	0.00

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
0	1000	11367.8	722
0	999	12175.4	983
0	998	13119.0	162

The SAS System

The UNIVARIATE Procedure
Variable: OUTPDOL

Moments			
N	1000	Sum Weights	1000
Mean	111.39919	Sum Observations	111399.19
Std Deviation	248.217815	Variance	61612.0838
Skewness	9.53237587	Kurtosis	139.188553
Uncorrected SS	73960251.2	Corrected SS	61550471.7
Coeff Variation	222.81833	Std Error Mean	7.84933652

Basic Statistical Measures			
Location		Variability	
Mean	111.3992	Std Deviation	248.21782
Median	44.0000	Variance	61612
Mode	0.0000	Range	4523
		Interquartile Range	107.15000

Tests for Location: Mu0=0				
Test	Statistic		p Value	
Student's t	t	14.19218	Pr > t	<.0001
Sign	M	390.5	Pr >= M	<.0001
Signed Rank	S	152685.5	Pr >= S	<.0001

Quantiles (Definition 5)	
Quantile	Estimate
100% Max	4522.680
99%	970.105
95%	412.975
90%	267.525
75% Q3	117.150
50% Median	44.000
25% Q1	10.000
10%	0.000
5%	0.000
1%	0.000
0% Min	0.000

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
0	998	1338.90	221
0	997	1415.49	58
0	989	1627.62	147
0	978	3535.44	414
0	975	4522.68	415

We want to put comments in after the first procedure.

The SAS System

The FREQ Procedure

SITE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	1000	100.00	1000	100.00

We also want comments after the second procedure is run.

6) **NEWFILE=** option: In the HTML output shown above, you might have wanted a separate page (file) for each table, rather than having all tables on the same page. For this purpose, the **NEWFILE=** option could have been used to specify the starting point for each new **BODY** file. The syntax for this option is

```
newfile = <starting point>;
```

where a starting point can be:

NONE - write all output to the **BODY** file that is currently open
 OUTPUT - start a new **BODY** file for each output object
 PAGE - start a new **BODY** file for each page of output
 PROC - start a new **BODY** file each time that a new procedure is started.

Just as ODS “increments” the name of the anchor, ODS will also automatically increment the names of the new files. For example, if the original **BODY** file is named **RESULTS**, each new **BODY** file that is created based on the **NEWFILE** option will be called **RESULTS1**, **RESULTS2**, etc.

7) **PAGE=** option: If the **NEWFILE=** option is specified, you may also want to include the **PAGE=** option in your HTML statements:

```
page=<file-specification>
```

The file specified will contain a description of each page of the **BODY** file as well as links to the **BODY** files. It is worth mentioning that the SAS system option **PAGESIZE=** has no effect on pages of HTML output.

EXAMPLE 2. Putting it all together (again). The following program illustrates the **NEWFILE=** option and the **PAGE=** option, as well as some HTML options already discussed.

```
libname health 'C:\Data';

1) ods listing close;

2) ods html path      = 'C:\Data'
   (url   ='http://www.unc.edu/~jismith')
   file   = 'file.htm'
   contents = 'contents.htm'
   frame  = 'frame.htm'
   page   = 'page.htm' (no_top_matter)
   newfile = page;

3) ods html select Moments;

proc univariate data=health.meddat;
  var dentdol drugdol ;
run;

proc print data=health.meddat;
  var site person contyr;
run;

4) ods html close;
```

The four ODS statements in the above code require some discussion.

(1) This ODS statement closes **LISTING** as a destination for output. This is done to conserve resources.

(2) In this ODS statement, the following things happen:

- the **PATH=** option specifies where to store your HTML files;
- the **URL=** sub-option tells ODS to use this URL for links and references;
- the **FILE=** option tells ODS the location of the **BODY** file;
- the **CONTENTS=** option tells ODS to use this file for links to the **BODY** file for every HTML table that is created in a **PROC** or **DATA** step.
- the **FRAME=** option provides a simultaneous view of all files included in the ODS HTML statement.
- the **PAGE=** option tells ODS to use this file for a link to the **BODY** file for every page of HTML that ODS creates from a **PROC** or **DATA** step.
- the **NEWFILE=** option tells ODS to create a new **BODY** file for each new page of output. In this case this would mean a new file of output for each variable in the **UNIVARIATE** procedure and a new file for the **PROC PRINT** output. The name of each new file is based on the name specified by **FILE=** option. The **BODY** files that are created in this example are **FILE.HTM**, **FILE1.HTM**, and **FILE2.HTM**.

(3) This ODS statement instructs ODS to send only the ‘Moments’ statistics from the **PROC UNIVARIATE** to the HTML output destination.

(4) This ODS statement closes the HTML destination for output to be sent there.

The **CONTENTS** page created from this program by the **CONTENTS= 'CONTENTS.HTM'** option is shown below. Each of the references under the procedure titles is a hyper-text link to the location of the respective table in the **BODY** file.

The **PAGE** file created from the **PAGE= 'PAGE.HTM'** (**NO_TOP_MATTER**) option is also shown below. Each page reference (**PAGE 1**, **PAGE 2**, **PAGE 3**) is a hyper-text link to that page in the **BODY** file.

Each of the **BODY** files that are created is shown below after the **CONTENTS** and **PAGE** files that are created. The first **BODY** file that is created is called **FILE.HTM**, and it contains the Moments data for the variable **DENTDOL**. This page is created first because **DENTDOL** is the first variable mentioned in the **PROC UNIVARIATE**, and **PROC UNIVARIATE** is the first **PROC** in the program. The second **BODY** file created is called **FILE2.HTM** and it contains the Moments data for **DRUGDOL**. The last page, **FILE3.HTM**, is from the **PROC PRINT**. The middle observations are excluded in this paper in order to save space.

By clicking on a reference in either the **CONTENTS** FILE display or the **PAGE** file display, we can link to each of the **BODY** files. The **FRAME** FILE, **FRAME.HTM**, would combine the **PAGE** file, **CONTENTS** file, and whichever **BODY** file was selected, onto one page.

We have reviewed a few of the options used with the HTML destination. Others can be found in the “The Complete Guide to the SAS® Output Delivery System, Version 8.”

contents.htm:

Table of Contents

- The Univariate Procedure
 - DENTDOL
 - [Moments](#)
 - DRUGDOL
 - [Moments](#)
- The Print Procedure
 - [Data Set IN.AGGREC](#)

page.htm:

Table of Pages

- The Univariate Procedure
 - [Page 1](#)
 - [Page 2](#)
- The Print Procedure
 - [Page 3](#)

file.htm:

The SAS System

*The UNIVARIATE Procedure
Variable: DENTDOL*

Moments			
N	100	Sum Weights	100
Mean	52.38	Sum Observations	5238
Std Deviation	226.742535	Variance	51412.1774
Skewness	8.17753631	Kurtosis	73.6960078
Uncorrected SS	5364172	Corrected SS	5089805.56
Coeff Variation	432.879984	Std Error Mean	22.6742535

file2.htm:

The SAS System

*The UNIVARIATE Procedure
Variable: DRUGDOL*

Moments			
N	100	Sum Weights	100
Mean	14.8721	Sum Observations	1487.21
Std Deviation	34.3546782	Variance	1180.24391
Skewness	3.32771071	Kurtosis	13.1047585
Uncorrected SS	138962.083	Corrected SS	116844.147
Coeff Variation	231.000855	Std Error Mean	3.43546782

file3.htm:

The SAS System

Obs	SITE	PERSON	CONTYR
1	1	MA250247	01
2	1	MA250247	02
3	1	MA250247	03
4	1	MA250247	04
5	1	MA250247	05
6	1	MA250255	01
7	1	MA250255	02
8	1	MA250255	03
9	1	MA250255	04
10	1	MA250255	05
11	1	MA250263	01
.	.	.	.
.	.	.	.
.	.	.	.
93	1	MA25162A	05
94	1	MA251638	01
95	1	MA251638	02
96	1	MA251638	03
97	1	MA251638	04
98	1	MA251638	05
99	1	MA251646	01
100	1	MA251646	02

ODS OUTPUT DESTINATION

When you first start SAS Version 7 or Version 8, the Output destination is closed and the exclusion list is set to EXCLUDE ALL. You can change these default actions with the ODS OUTPUT statement. The ODS OUTPUT statement is used to specify an action, or to create one or more data sets.

Specifying an action

When used to specify an action, the ODS OUTPUT statement syntax is:

```
ods output <action>;
```

where the action choices are:

CLEAR - set the list for the OUTPUT destination to EXCLUDE ALL
SHOW - display the selection or exclusion list that apply at this point in the program in the SAS log
CLOSE - close the OUTPUT destination - once the destination is closed, you cannot send output to this destination.

Creating output data sets

You open the Output destination by specifying the data set(s) that you would like created. To create a single output data set, the syntax is

```
ods output <output-object> = <sas data set>;
```

where the output-object can be identified with the use of the ODS TRACE statement. For example,

```
ods listing;
ods trace on / label;
ods output BasicMeasures = measures;
```

```
proc univariate data = meddat;
    var meddol suppdol;
run;
ods trace off;
```

Since we have both the Listing destination and the Output destination open, after running the program above with the ODS TRACE statement, all the statistics created by the PROC UNIVARIATE will appear in the LOG. Although all the PROC UNIVARIATE statistics will be sent to the Listing destination (whose default value is SELECT ALL), only the BasicMeasures statistics will be sent to the Output destination. You can look in the log or in the SAS Explorer window (in this case in the WORK library), to see that ODS has created the MEASURES data set. This newly created SAS data set will have the BasicMeasures statistics for both the MEDDOL and SUPPDOL variables. By looking in the Results window and clicking on BasicMeasures, you will see in the Output window the BasicMeasures statistics. Unlike the HTML destination, you do not have to close the Output destination to have objects sent there.

To create a separate output data set for each variable used in a PROC or DATA step, use the following syntax:

```
ods output <output-object> (match_all)
    = <sas data set>;
```

For example, the following code will select the OneWayFreqs statistics from the PROC FREQ. The OUTPUT statement creates a different data set for each variable in the procedure by the MATCH_ALL option, and bases the name of these data sets on the name STATS:

```
ods output onewayfreqs (match_all) = stats;
proc freq data='A:\test';
run;
ods output close;
run;
```

When this program is run, the data sets created are STATS, STATS1, STATS2 ... STATSN for however many variables there are in the data set TEST. If you would like to combine all the data sets created for all the variables into one larger data set, you use the following syntax:

```
(match_all=<macro-var-name>)
```

When this program is run, the data sets created are STATS, STATS1, STATS2 ... STATSN for however many variables there are in the data set TEST. If you would like to combine all the data sets created for all the variables into one larger data set, you use the following syntax:

```
(match_all=<macro-var-name>)
```

This creates a macro variable which is used to store a list of the names of the data sets that are created in the ODS OUTPUT statement. In a separate DATA step, you combine the data sets by concatenation. This is illustrated in the example below.

```
ods output OneWayFreqs (match_all=name) = stats;
```

To concatenate the data sets, you specify

```
data all;
  set &name;
run;
```

Be careful using the MATCH_ALL option. In the following program, separate data sets are created for the Moments data, but not for the Basic Measures data. All the variables are included in one data set for the Basic Measures statistics.

```
ods output BasicMeasures Moments
  (match_all)=moments;
```

To create output data sets for both sets of statistics, we would need to specify

```
ods output BasicMeasures (match_all) = measures
  Moments (match_all) = moments;
```

Permanent SAS data sets can be created by using librefs as in:

```
libname in 'A:\';
ods output BasicMeasures = in.measures;
```

OUTPUT Options

PERSIST option: One option that you can specify within the OUTPUT statement is the PERSIST option. This option is useful if you are creating output data sets from more than one PROC and/or DATA step. The syntax is as follows.

```
ods output OneWayFreqs (match_all=name
  persist=<proc | run>)=stats;
```

The PERSIST option specifies when to close any data sets that are being created, and when to remove output objects from the SELECTION list for the OUTPUT destination. The PERSIST option can only be used in conjunction with the MATCH_ALL option.

PROC argument: The PROC argument to the PERSIST option preserves the list of definitions that are specified in the ODS OUTPUT statement across step boundaries. This means that the list of output objects specified in the ODS OUTPUT statement is preserved even after the PROC or DATA steps have completed. You must explicitly modify the list to change the definitions; i.e. with

```
ods output exclude all.
```

RUN argument: The RUN argument to the PERSIST option serves the same function as the PROC statement, and it also keeps the data sets open. The following is an example of a program that implicitly uses the RUN argument but does not specify the PERSIST option (although it was intended to).

```
ods output OneWayFreqs (match_all=name) = stats;

proc freq data=health.test;
run;
proc freq data=health.test2;
run;
```

In this case, the data sets are not created for the second PROC FREQ. Without the PERSIST option, the second PROC statement is treated as a step boundary, and a data set for the variables in HEALTH.TEST2 is not created. This is easily corrected by explicitly specifying:

```
ods output OneWayFreqs (match_all=name
  persist=proc) = stats;
```

CONCLUSION

This purpose of this paper was to help you get started using the Output Delivery System. Although ODS is powerful, it is also quite easy to use. Hopefully our examples have illustrated this. We have tried to show that you can create web pages with the addition of as little as one line of code to your existing SAS program! With the addition of a single OUTPUT statement you can also create one or more SAS data sets. With a few additional words you can send your object objects to more than one destination at a time. The best way to convince yourself, though, that ODS is just this easy, is to visit our website and submit the examples presented in this paper for yourself. The website can be found at:

www.unc.edu/~lkbryant/intro_ods

REFERENCES

SAS® Version 8 Software.
SAS® Institute, The Complete Guide to the SAS® Output Delivery System, Version 8

ACKNOWLEDGEMENTS

We thank Paul Kent, Base SAS R&D Director, and Chris Olinger, Base SAS Software Manager, for teaching classes on these subjects for our site and local SAS users group meetings. We thank SAS Technical Support Division for answering our questions so promptly regarding items in our paper.

CONTACT INFORMATION

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

Lara K. Bryant
Jordan Institute for Families
CB 3550
301 Pittsboro St.
Chapel Hill, NC 27599
Email: lbryant@email.unc.edu

Sally S. Muller
Jordan Institute for Families
CB 3550
301 Pittsboro St.
Chapel Hill, NC 27599
Email: sally@email.unc.edu
Work Phone: 919-843-7798
Fax: 919 967 7015

Ray Pass
Ray Pass Consulting
5 Sinclair Place
Hartsdale, NY 10530
Email: raypass@att.net
Work Phone: 914-693-5553
Fax: 914-206-3780