Paper #GR006

Using SAS® ODS to Create Web-Based Neighborhood Profiles
For the District of Columbia
Peter A. Tatian, The Urban Institute, Washington, D.C.

ABSTRACT
This paper describes how the SAS Output Delivery System was used to produce web pages displaying neighborhood data for the District of Columbia. The procedure involved creating basic output using ODS and then modifying it with a post-processor program.

INTRODUCTION
This paper describes how the SAS Output Delivery System (ODS) was used to create a series of web-based neighborhood profiles for the District of Columbia. In collaboration with DC Agenda, the Urban Institute operates the DC Data Warehouse, a repository of data on the District and its neighborhoods. The goal of the Warehouse is to make local data available for community building purposes. One means of doing this is through the “neighborhood profiles,” which summarize a variety of indicators for all neighborhoods throughout the city.

A sample neighborhood profile is provided in Attachment A. The process for creating these profiles consists of three steps: 1) creating the profile data set, 2) generating the HTML output with SAS ODS, and 3) using a post-processor to modify the HTML files.

CREATE DATA SET
The input data set used to generate the profiles consists of 40 observations—one for the entire city and 39 for each of the District’s neighborhood clusters. (“Neighborhood clusters” are groups of three to five neighborhoods defined by the D.C. Office of Planning.) Each observation has a set of indicators defined from the Data Warehouse. Below is an excerpt of a sample file (NBR_DATA):

<table>
<thead>
<tr>
<th><em>TYPE</em></th>
<th>CLUSTER</th>
<th>SEC_POP</th>
<th>AVGPOP8</th>
<th>AVGPOP9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01</td>
<td>.</td>
<td>16016</td>
<td>15169</td>
</tr>
<tr>
<td>1</td>
<td>01</td>
<td>.</td>
<td>18149</td>
<td>17812</td>
</tr>
<tr>
<td>1</td>
<td>02</td>
<td>.</td>
<td>44081</td>
<td>45728</td>
</tr>
<tr>
<td>1</td>
<td>03</td>
<td>.</td>
<td>11335</td>
<td>10925</td>
</tr>
<tr>
<td>1</td>
<td>04</td>
<td>.</td>
<td>16071</td>
<td>17919</td>
</tr>
<tr>
<td>1</td>
<td>05</td>
<td>.</td>
<td>10731</td>
<td>11104</td>
</tr>
<tr>
<td>1</td>
<td>06</td>
<td>.</td>
<td>16621</td>
<td>16216</td>
</tr>
<tr>
<td>1</td>
<td>07</td>
<td>.</td>
<td>18367</td>
<td>19867</td>
</tr>
<tr>
<td>1</td>
<td>08</td>
<td>.</td>
<td>7587</td>
<td>8300</td>
</tr>
<tr>
<td>1</td>
<td>09</td>
<td>.</td>
<td>12332</td>
<td>11592</td>
</tr>
</tbody>
</table>

The _TYPE_ variable distinguishes between the city level (0) and neighborhood level (1) observations. The variable CLUSTER is the identification number for the neighborhood cluster. AVGPOP8 and AVGPOP9 are two of the data fields in the profile, which contain the cluster populations in 1980 and 1990, respectively. (The values for the city total are the average across all clusters.) The variable SEC_POP, which has all missing values, is a placeholder that will be used to create “Population” section heading in the profile. All variables have been labeled with the text that should appear for in the profile. For example, AVGPOP8 is labeled, “Population, 1980.”

Because the list of variables and section headings will need to be used several times in creating the profiles, we first define a macro variable that contains this list:

```
%let prof_vars = 
  Sec.Pop avgpop8 avgpop9 avgpop0
  trpop8_9 trpop9_0 child8 child9
  child0 child8_9 child9_0
  shrfor8 shrfor9 diffhse9
  Sec_Race shrnhb8 shrnhb9 shrnhb0
  shrnhw8 shrnhw9 shrnhw0 etc. ;
```

Note that the variables and section heads must be listed in the exact order that we wish them to appear in the profile.

The input data set has the different indicators as columns (variables) and the city and neighborhood clusters as rows (observations). The profile format, however, has the indicators as the rows and the geographic areas as columns. To put the data in the proper format for creating the profile, we use the TRANSPOSE procedure to flip the columns and rows. The first PROC TRANSPOSE flips the cluster-level observations:

```
proc sort data=Nbr_Data;
  by cluster;
proc transpose data=Nbr_Data out=Tr_Data;
  by cluster;
where _type_ > 0;
var &prof_vars;
```

This results in a file (Tr_data) that resembles the following:

```
_01       Sec.Pop Population             .
_01       avgpop8 Population, 1980  18149.00
_01       avgpop9 Population, 1990  17812.00
_01       avgpop0 Population, 2000  18167.00
```

The variable COL1 contains the values for the individual indicator for the specified cluster. Each observation is now an individual indicator for a cluster, as identified by the input data set variable name (_NAME_) and its SAS label (_LABEL_). Note the missing value in COL1 for the section header (_NAME_=Sec_pop). The effect of the BY CLUSTER statement in PROC TRANSPOSE is to repeat the complete list of section headers and variables for each of the 39 neighborhood clusters.

This data set Tr_data will be used to create the first column in the profile—the indicator values for each individual cluster. To create the last two columns, the minimum and maximum values across all clusters will be used. The SUMMARY procedure will be used to summarize the data. The MIN function will be used to find the minimum value, and the MAX function will be used to find the maximum value. The following code will create the minimum and maximum columns:

```
proc summary data=Tr_data;
  by cluster;
  var avgpop8-avgpop9;
  output out=MinMax_results min=min avgpop8 max=max avgpop9;
run;
```

The resulting data set MinMax_results will be used to create the last two columns in the profile. The final step is to modify the HTML output with SAS ODS. This paper describes how the SAS Output Delivery System was used to produce web pages displaying neighborhood data for the District of Columbia. The procedure involved creating basic output using ODS and then modifying it with a post-processor program.
The new data set `Tr_Data_2` has a variable `N` with the original observation order. We can now compute the minimum and maximum values for each indicator:

```sas
proc sort data=Tr_Data_2;
by _name_;
proc summary data=Tr_Data_2;
var col1;
by _name_; output out=Tr_Data_Min_Max min= max=/autoname;
```

The data set `Tr_Data_Min_Max` will have one observation for each indicator/section head with variable COL1_MIN and COL1_MAX containing the minimum and maximum values, respectively.

Finally, we create the `city` column for the profile by transposing the city-level observation in the original `Tr_data` data set:

```sas
proc transpose data=Nbr_Data out=Tr_Data_Cty; where _type_ = 0; var &prof_vars; proc sort data=Tr_Data_Cty; by _name_;
```

The three transposed data sets can then be merged together and sorted by `N` to place them back in the proper profile order:

```sas
data Tr_Data_Mrgd;
merge
    Tr_Data_2
    Tr_Data_Cty (rename=(col1=col1_mean))
    Tr_Data_Min_Max;
by _name_; proc sort data=Tr_Data_Mrgd out=Nbr_Profile_Web; by n;
```

The final output data set, `Nbr_Profile_Web`, looks something like figure 1.

### Figure 1. Excerpt of Data Set Nbr_Profile_Web

<table>
<thead>
<tr>
<th>CLUSTER</th>
<th><em>LABEL</em></th>
<th>COL1</th>
<th>COL1_MEAN</th>
<th>COL1_MIN</th>
<th>COL1_MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Population</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>01</td>
<td>Population, 1980</td>
<td>18149.00</td>
<td>16016.05</td>
<td>3506.00</td>
<td>44081.00</td>
</tr>
<tr>
<td>01</td>
<td>Population, 1990</td>
<td>17812.00</td>
<td>15169.26</td>
<td>1341.00</td>
<td>45728.00</td>
</tr>
<tr>
<td>01</td>
<td>Population, 2000</td>
<td>18167.00</td>
<td>14455.64</td>
<td>2374.00</td>
<td>46779.00</td>
</tr>
<tr>
<td>01</td>
<td>% Change Population, 1980-</td>
<td>-1.86</td>
<td>-5.29</td>
<td>-61.75</td>
<td>22.62</td>
</tr>
<tr>
<td>01</td>
<td>% Change Population, 1990-</td>
<td>1.99</td>
<td>-4.70</td>
<td>-20.11</td>
<td>77.03</td>
</tr>
<tr>
<td>01</td>
<td>Children, 1980</td>
<td>13.36</td>
<td>22.47</td>
<td>1.77</td>
<td>42.67</td>
</tr>
<tr>
<td>01</td>
<td>Children, 1990</td>
<td>10.82</td>
<td>19.16</td>
<td>1.41</td>
<td>37.50</td>
</tr>
<tr>
<td>01</td>
<td>Children, 2000</td>
<td>9.04</td>
<td>19.90</td>
<td>1.22</td>
<td>42.58</td>
</tr>
</tbody>
</table>

etc.

---

### GENERATE HTML PROFILES

Once the underlying data has been properly formatted, the second step is to create the basic HTML output using SAS ODS. We use the `REPORT` procedure to create the basic profile tables. Before calling `PROC REPORT`, we need to create a few formats that will be used to display information in the profile:

```sas
proc format;
   picture profnum (default=14 round)
        low - -10  = '0,000,000,009' (prefix='-')
        -10 <-< 0  = '009.9' (prefix='-')
        0  -< 10  = '009.9'
        10  - high = '00,000,000,009';
   value $cluster
        "01" = "Cluster 1 - Ward 1 / Kalorama Heights, Adams Morgan, Lanier Heights"
        "02" = "Cluster 2 - Ward 1 / Mt. Pleasant, Columbia Heights, Park View"
        "03" = "Cluster 3 - Ward 1 / Howard University, Ledroit Park, Cardozo/Shaw"
    etc.;
```

The picture format `PROFNUM` formats all of the numbers appearing in the profile so that numbers that round to 10 or higher or to –10 or lower display no digits after the decimal point, and numbers rounding between –10 and 10 display one digit after the decimal point. The `$CLUSTER` value format is used to display the neighborhood cluster descriptions that coincide with each of the cluster numbers.

We can then open the ODS HTML destination and run `PROC REPORT`:

```sas
options missing=" " nobyline;
ods html body="d:\profiles\Nbr_prof_clus1.html" title="DCNIS - Neighborhood Cluster Profile" style=styles.sasweb newfile=page;
proc report list data=Nbr_Profile_Web nowd split="*";
   by cluster;
   column _label_ col1;
       format col1 col1_mean col1_min col1_max profnum.
       cluster $cluster;
   label _label_ = "";
```

The picture format `PROFNUM` formats all of the numbers appearing in the profile so that numbers that round to 10 or higher or to –10 or lower display no digits after the decimal point, and numbers rounding between –10 and 10 display one digit after the decimal point. The `$CLUSTER` value format is used to display the neighborhood cluster descriptions that coincide with each of the cluster numbers.
The two system options, MISSING=" " and NOBYLINE, suppress the display of the period (".") as the numeric missing value and the default "by line" that is displayed when using a BY statement, respectively. The ODS HTML statement opens the HTML destination and designates the file location for the output. It also tells SAS we are using the SASWEB style to format the output and that we want to create a new HTML file for each new page of SAS output. In conjunction with the BY CLUSTER statement in PROC REPORT, this will generate a separate HTML page for each neighborhood cluster. The HTML file name will be "Nbr_prof_clus1.html" for the first cluster, "Nbr_prof_clus2.html" for the second, and so on. The TITLE= option after the body file name changes the default title for the HTML web pages from "SAS Output" to "DCNIS - Neighborhood Cluster Profile.".

In the REPORT statement, we have specified the LIST option to create a listing report (basically a printout of the data), the NOWD option to suppress the PROC REPORT interactive window interface (since we are running this as a batch operation), and the SPLIT="*" option to designate the asterisk as a special character to split labels. As mentioned above, the BY statement produces a separate table for each neighborhood cluster. The COLUMN statement lists the five variables that define the columns in the report. The "All Clusters" label enclosed in parentheses with the variables COL1_MEAN, COL1_MIN, and COL1_MAX creates a spanning header across those three columns. The FORMAT statement associates the PROFNUM and $CLUSTER formats with the appropriate variables, while the LABEL statement defines the labels for the column headings.

The TITLE and FOOTNOTE statements specify the title and footnote text for each cluster profile. In the TITLE1 statement (the first title line), we are using the ITALIC option to display this line in italics. The TITLE3 statement uses the HEIGHT= option to enlarge the font for this line. The code "#byval( cluster )" causes the formatted value of the BY statement variable, CLUSTER, to be inserted in this title line. Finally, the FOOTNOTE statement has the code "%sysfunc(date(),worddate.)", which inserts the current date written out with the WORDDATE format, as a date stamp on each file.

The last two statements run the report and then close the HTML destination. Please note that the RUN statement is necessary before the ODS CLOSE statement; if it is omitted, none of the output will be directed to the HTML file.

HTML-SPECIFIC EXTENSIONS

All of the code used thus far will work with any ODS output destination. To further customize the appearance of the PROC REPORT output, however, we will need to introduce some extensions that will only work with the HTML destination. Specifically, we want to use HTML tags to control how different elements of the profile appear. For example, HTML uses the tags "&lt;b&gt;...&lt;/b&gt;" to designate text that should appear in bold.

Before we can use HTML tags, however, we need to change the settings in the ODS template that cause SAS to "protect" special characters in the output destination. For the HTML destination, the characters "<" and ">" are special because they are used to designate HTML tags. If you actually wanted to display these symbols in your web page, you would need to use the HTML symbols "&lt;" and "&gt;", respectively. With special character protection turned on, SAS assumes that if you type "<" in a title it is not a tag and you really want to display the "less than" symbol in the output. Therefore, if we put "&lt;b&gt;Bold text&lt;/b&gt;" in a SAS TITLE statement, the output would appear as:

   &lt;b&gt;Bold text&lt;/b&gt;

rather than:

   Bold text

as we would like.

To turn off the special character protection, we can change the settings in the ODS template using the TEMPLATE procedure:

   proc template;
      /** Turn off special char protection **/
      style systemtitle from systemtitle /
         protectspecialchars=off;
      style systemfooter from systemfooter /
         protectspecialchars=off;
      style data from data /
         protectspecialchars=off;
      run;
   
ods html close;

   *** Create new style PROFILE_WEB based on style SASWEB ***
   define style profiles.profile_web;
      parent=styles.sasweb;
   
   *** Next stmt same as original style ***
   style html
      "Common HTML text used in the default style" /
      'expandAll' = "&lt;SPAN onclick="if(msie4==1)expandAll()"">"
      'posthtml flyover line' = "&lt;HR size=3>"
      'prehtml flyover line' = "&lt;HR size=3>"
      'fake bullet' = %nrstr("\&#183;");
      'Line' = "&lt;HR size=3>"
      'PageBreakLine' = %nrstr("\&lt;p style="page-break-after: always;"&gt;\&lt;HR size=3>\&lt;br\&gt;\&lt;span style="\&lt;b\&gt;fake bullet\&lt;/b\&gt;"&gt;\&lt;b\&lt;br\&gt;</span>\&lt;/b\&gt;\&lt;br\&gt;\&lt;span style="\&lt;b\&lt;br\&gt;
   
   /* Turn off special char protection **/
   style systemtitle from systemtitle /
      protectspecialchars=off;
   style systemfooter from systemfooter /
      protectspecialchars=off;
   style data from data /
      protectspecialchars=off;

   run;
This code takes the SAS-supplied SASWEB template and modifies it by setting the PROTECTSPECIALCHARS options to "OFF" for the title, footer, and data objects in the output. The modified style is saved as a new style called PROFILE_WEB in the PROFILES library. The code in the STYLE HTML statement is actually just cut and pasted from the original SASWEB style, which can be displayed in the SAS LOG by running the following command:

```sas
proc template;
source styles.sasweb;

To apply the new template, we simply change the ODS statement that opens the HTML destination:

```sas
ods html
body="d:\profiles\Nbr_prof_clus1.html"  
(title="DCNIS - Neighborhood Cluster Profile")  
style=profiles.profile_web
newfile=page;
```

With this done, we can now incorporate HTML tags directly into the variable labels, titles, and footnotes in the output. For example, we can change the $CLUSTER format to display the cluster number in bold, and the rest in smaller, plain typeface on a new line by inserting appropriate tags:

```sas
proc format;
value $cluster
"01" = "Cluster 1 / Kalorama Heights, Adams Morgan, Lanier Heights"<BR>
"02" = "Cluster 2 / Ward 1 / Mt. Pleasant, Columbia Heights, Park View"<BR>
</B>  
etc.
```

Note the use of the </B> and </SMALL> tags. These are necessary to "turn off" tags that SAS is already inserting in the output. If we were to examine the default HTML code produced by SAS for the title line (before we introduced HTML tags in to the $CLUSTER format), it would look like this:

```html
<b>Cluster 1 - Ward 1 / Kalorama Heights, Adams Morgan, Lanier Heights</b>
```

Therefore, to display the second line with a plain typeface we need to include the code to turn off the bold typeface (<B>). With the modified $CLUSTER format, the resulting HTML looks like this (SAS-generated tags are shown in lowercase to distinguish them from the tags we are inserting):

```html
<b>Cluster 1</b>:<BR><b>Cluster 1 - Ward 1 / Kalorama Heights, Adams Morgan, Lanier Heights</b>
```

Note the use of the final <B> tag to balance the SAS-generated </B> tag.

Similarly, we can use HTML tags in the LABEL statement with PROC REPORT. In this case, we are using the HTML non-breaking space symbol ("&nbsp;") to pad the column headings and make the columns more uniform in width:

```sas
label
cluster = ""
_label_ = ""
```

MODIFY HTML FILES

The final step is to modify the SAS created HTML files to include additional formatting that is not easy to add with ODS. Specifically, we wish to add the table of contents at the top of the page that links to bookmarks to each of the section headers in the main profile table and a link to the sources and notes and the DCNIS logo at the bottom of the page.

To do this, we use a program called a "post-processor" (also often referred to as a "pre-processor"). This is a piece of software that will insert preset code into a file in a specific location (somewhat like the %INCLUDE directive in SAS). We can use a post-processor to overcome certain limitations of SAS ODS. In particular, in version 8.2 of SAS titles and footnotes longer than 108 characters are broken automatically with an HTML break <BR>, disrupting other HTML tags. Furthermore, use of a post-processor allows greater control over the placement of HTML elements, like images.

Several post-processor programs exist that can be used in this situation. We chose a program called "Orb," which is a free, DOS-based utility created by Craig Berry. To incorporate Orb into the process of creating the web-based profiles, we need to make certain changes to the SAS code. First, we must change the names of the files created by ODS to have a .ORB extension, instead of .HTML:

```sas
ods html
body="d:\profiles\Nbr_prof_clus1.orb" etc.;
```

Second, we need to insert Orb commands to insert the header and footer code into the title and footnote statements:
We must then create the files nbr_prof_web_header.orh and nbr_prof_web_footer.orh that are to be inserted into the ODS output files. When ODS adds titles and footnotes to the HTML destination, it does this by creating separate HTML tables for these elements. As before, we first need to turn off the SAS created HTML tags before we can insert those of our own. So, for the profile header (nbr_prof_web_header.orh):

```sas
<!--Turn off SAS codes-->
</B></FONT></TD></TR></TABLE>
<!--Begin table of contents-->
<HR WIDTH="85%" ALIGN=LEFT>
<TABLE ALIGN="left" BORDER="0"
CELLSPACING="5" CELLPADDING="0" WIDTH="75%"
bgcolor="#E0E0E0">
<TR ALIGN="LEFT">
<TD><SMALL><A HREF="#Sec_pop">Population</A></SMALL></TD>
<TD><SMALL><A HREF="#Sec_Child_Ind">Child Well-Being Indicators</A></SMALL></TD>
</TR>
</TABLE>
<!--End of table of contents-->
<!--Turn SAS codes back on-->
```

Similarly, for the page footer (nbr_prof_web_header.orh):

```sas
<!--Turn off SAS codes-->
</I></B></FONT></TD></TR></TABLE>
<!--End of footer-->
```

Finally, to make the bookmark links in the table of contents work, we need to add the bookmarks to the profile section headers by changing the variable labels in the data set Nbr_dat:

```sas
label
Sec_pop = "<A NAME=Sec_pop><B>Population</B></A>"
Sec_race = "<A NAME=Sec_race><B>Population by Race/Ethnicity</B></A>"
Sec_fam_risk = "<A NAME=Sec_fam_risk><B>Family Risk Factors</B></A>"
```

CONCLUSION

As we can see, with a little bit of effort we can achieve quite a high degree of customizability in SAS-generated HTML output. We have used the same basic code, converted into a SAS macro, to create parallel sets of profiles for a variety of other geographic areas relevant to people in the District of Columbia—wards, census tracts, police service areas, etc. We hope that what we have learned in producing these profiles can be useful to others who may wish to produce similar reports for other needs.

REFERENCES

For additional information on ODS, please see the following:

- ODS Support page: http://support.sas.com/md/base/index-ods-resources.html
- Proc Template FAQ: http://support.sas.com/md/base/topics/templateFAQ/Template.html

NOTES

1. The complete set of neighborhood profiles may be viewed on the DC Agenda web site at http://www.dcagenda.org/nis.
2. As anyone who has worked with SAS ODS knows, you often have to search around to find documentation on particular features. The only place we found the information on how to change the title of ODS-produced web pages was on the site of PW Consulting, a
Netherlands-based consulting firm (http://tips.pwcons.com/modules.php?name=News&file=article&sid=70). To them we would like to say, “Dank u wel!”

ACKNOWLEDGMENTS
The author would like to acknowledge the Annie E. Casey Foundation for their support of the DC Data Warehouse project.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. The author may be contacted at:

   Peter A. Tatian  
   The Urban Institute  
   2100 M Street NW  
   Washington, DC 20037  
   Work Phone: 202-261-5588  
   Fax: 202-872-9322  
   Email: ptatian@ui.urban.org  
   Web: http://www.urban.org/dcdata

3 See SAS note SN-005118 (http://support.sas.com/techsup/unotes/SN/005/005118.html).

4 To download a copy of Orb go to http://www.cinenet.net/~cberry/orbinfo.html.
Attachment A. Sample Neighborhood Profile (First and Last Pages)

**DCNIS - Neighborhood Cluster Profile**

<table>
<thead>
<tr>
<th>Population</th>
<th>Child Well-Being Indicators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population by Race/Ethnicity</td>
<td>Income and Housing Conditions</td>
</tr>
<tr>
<td>Family Risk Factors</td>
<td>Reported Crimes (per 1,000 pop.)</td>
</tr>
<tr>
<td>Isolation Indicators</td>
<td></td>
</tr>
</tbody>
</table>

**Cluster 1**  
Ward 1 / Kalorama Heights, Adams Morgan, Lanier Heights

<table>
<thead>
<tr>
<th></th>
<th>This Cluster</th>
<th>Average</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population, 1980</td>
<td>18,149</td>
<td>16,015</td>
<td>3,505</td>
<td>44,081</td>
</tr>
<tr>
<td>Population, 1990</td>
<td>17,812</td>
<td>15,169</td>
<td>1,341</td>
<td>45,728</td>
</tr>
<tr>
<td>Population, 2000</td>
<td>18,157</td>
<td>14,455</td>
<td>2,374</td>
<td>46,775</td>
</tr>
<tr>
<td>Violent Crimes, 1998</td>
<td>12</td>
<td>15</td>
<td>0.7</td>
<td>51</td>
</tr>
<tr>
<td>Violent Crimes, 1999</td>
<td>10</td>
<td>14</td>
<td>0.9</td>
<td>40</td>
</tr>
<tr>
<td>Violent Crimes, 2000</td>
<td>0.5</td>
<td>15</td>
<td>0.5</td>
<td>51</td>
</tr>
<tr>
<td>Property Crimes, 1998</td>
<td>86</td>
<td>62</td>
<td>14</td>
<td>291</td>
</tr>
<tr>
<td>Property Crimes, 1999</td>
<td>90</td>
<td>53</td>
<td>15</td>
<td>253</td>
</tr>
<tr>
<td>Property Crimes, 2000</td>
<td>51</td>
<td>51</td>
<td>14</td>
<td>289</td>
</tr>
</tbody>
</table>

Revised October 7, 2002.

Prepared by the DC Neighborhood Information Service and the DC Data Warehouse.

Sources and Notes