Paper 17-25

You CAN Get There from Here (and Back Again): Adding Hot-link Drill-down Capabilities to %DS2HTM and %TAB2HTM Output

Ray Pass
Ray Pass Consulting

ABSTRACT

The SAS Web Publishing Tools (%TAB2HTM, %DS2HTM and %OUT2HTM) can be used to create HTML pages for publishing on the Internet or an Intranet. Right out of the box (and the box is FREE too!), they can be used to create basically one page per PROC TABULATE output, or one page per SAS data set (tabularized display). (%OUT2HTM can be used to add titling and other information to any SAS procedure output, but the basic procedure output is not really enhanced.) By using simple SAS MACRO processing combined with the manipulation of TITLE statements, basic data set variable values and values of variables used as CLASS variables in TABULATE procedures, you can create whole systems of hot-linked pages with almost full drill-down capabilities. The technique is to simply insert (via macros or DATA step coding) the desired navigational HTML tags, in the appropriate places, to the to-be-displayed data (or TITLE) values, before running the data set through %DS2HTM, or through the TABULATE procedure to be used with %TAB2HTM. Techniques and examples are shown in this paper.

INTRODUCTION

The Web Publishing tools provided by SAS Institute have given us a method of preparing HTML formatted versions of output from the TABULATE procedure (%TAB2HTM macro) as well as HTML formatted tabular displays of SAS data sets (%DS2HTM macro). These macros provide for the formatting of all aspects of the tabular display, including row, column and cell elements, as well as titles for the accompanying output. There is a third tool provided, the %OUT2HTM macro, which is applicable to any SAS System procedure output, but it is much more limited than the other two in that although it does provide for title and some cosmetic overall formatting, it really provides no significant enhancements for the actual procedure content output.

The above mentioned web publishing macro tools are applicable to Version 6 of the SAS System as well as Version 7, but the advent of the Output Delivery System (ODS) in Version 7 practically obviates the need for the tools themselves in Version 7 and later. However, the techniques described in this paper are additions to the tools which are equally applicable to ODS HTML output.

The %TAB2HTM and %DS2HTM macros take the output from TABULATE procedures, or the contents of SAS data sets, and allow them to be rendered as HTML formatted documents which can then be read via an HTML browser. The documents consist of titles and HTML tables, and can be afforded the full range of HTML formatting enhancements, including Cascading Style Sheets. This recent addition to the HTML standards is only available in later versions of the major browsers.

Although the transformation from standard SAS output to HTML formatted output is a major step forward in terms of creating high information content reports which can be made available via the internet or any size intranet to large groups of information consumers, it is still rudimentary in nature because each output is static. One of the seminal definitional features of web information presentation is the ability to connect related informational displays via hot links (hypertext), or drill down techniques. The hot link capability is accomplished by enriching an element so it can serve as a clickable launching point to a related collection of information. The main enrichment is in the form of the location, or address, of the target display. This paper presents the basis of a methodology for creating these enriched data elements, as well as for the automatic data-driven generation of all the static outputs necessary for a complete system of related informational displays (static report outputs).

This paper assumes that the reader is already familiar with the use of the web publishing macros. It is not an introduction to these techniques, but is rather an explication of an addition to them. The tools themselves, as well as complete documentation for their use, including extensive examples, are available for downloading at the SAS Institute web site, **www.sas.com**.

SAMPLE DATA

The data to be used throughout his paper consists of daily sales reports of the fictitious RPC Entertainment Enterprises Corporation. There are six geographical regions in the company (NorthEast, NorthCentral, NorthWest, SouthEast. SouthCentral, SouthWest), and each region is further broken down into the states making up the region. There are two divisions in the company (Games, Toys) and each division is broken down into various items produced by the division.

NON-HTML REPORTS

A series of PROC TABULATEs (and accompanying TITLE statements) can be used without HTML enhancement to produce the needed daily sales reports. Although all of the Region by State by Division by Item data could be presented in their most granular form in one TABULATE output, since the end goal is a hypertext system of related reports, a representative sample of these modular reports is presented here. Only one of the six (one for each Region) possible *State by Division* reports, one of the two (one for each Division) possible *Region by Item* reports, and one of the 12 (one for each Region-Division combination) possible *State by Item* reports are shown. These are found in Figs 1-4 ***(figures are placed out of sequential order on this page so as to avoid breaking them across columns.)***

RPC Entertainment Enterprises
State by Division Sales Report: Jun 1, 1999

Region: NorthEast	Division		
	Games	Toys	TOTAL
State			
CT	631	1,075	1,706
DC	1,319	1,763	3,082
DE	207	631	838
MA	1,075	1,319	2,394
MD	1,763	207	1,970
ME	631	1,075	1,706
NH	1,319	1,763	3,082
NJ	207	631	838
NY	1,075	1,319	2,394
PA	1,763	207	1,970
RI	631	1,075	1,706
VT	1,319	1,763	3,082
TOTAL	11,940	12,828	24,768

Fig 2. Non-HTML Report
State by Division (Region: NorthEast)

RPC Entertainment Enterprises
Region by Division Sales Report: Jun 1, 1999

All Regions by All Divisions	Division		
	Games	Toys	TOTAL
Region			
NorthCentral	10,490	9,590	20,080
NorthEast	11,940	12,828	24,768
NorthWest	5,202	5,626	10,828
SouthCentral	6,070	6,314	12,384
SouthEast	9,990	9,990	19,980
SouthWest	7,389	6,277	13,666
TOTAL	51,081	50,625	101706

Fig 1. Non-HTML Report
Region by Division

RPC Entertainment Enterprises
Region by Item Sales Report: Jun 1, 1999

Division: Toys	Item		
	GI Joe	SI Jim	TOTAL
Region			
NorthCentral	4,740	4,850	9,590
NorthEast	5,748	7,080	12,828
NorthWest	2,480	3,146	5,626
SouthCentral	2,824	3,490	6,314
SouthEast	4,440	5,550	9,990
SouthWest	2,750	3,527	6,277
TOTAL	22,982	27,643	50,625

Fig 3. Non-HTML Report
Region by Item (Division: Toys)

RPC Entertainment Enterprises
State by Item Sales Report: Jun 1, 1999

Region: NorthEast Division: Toys	Item		
	GI Joe	SI Jim	TOTAL
State			
CT	482	593	1,075
DC	826	937	1,763
DE	260	371	631
MA	604	715	1,319
MD	48	159	207
ME	482	593	1,075
NH	826	937	1,763
NJ	260	371	631
NY	604	715	1,319
PA	48	159	207
RI	482	593	1,075
VT	826	937	1,763
TOTAL	5,748	7,080	12,828

Fig 4. Non-HTML Report
State by Item (Region: NorthEast, Division: Toys)

SIMPLE %TAB2HTM OUTPUT

To use the web publishing tool to produce HTML output instead of the standard output listings seen above, we would embed the `TABULATE` code between a `%TAB2THM` macro call to turn capture on, and another one after the `TABULATE` code to turn it off. The closing `%TAB2HTM` macro call contains all the optional formatting code. As this paper assumes the mechanics and syntax of the `%TAB2HTM` macro are known to the reader, it will not go into great detail about the code of the macro itself.

Outputs of the preliminary use of the macro as viewed via an HTML browser are shown in Figs 5-8. Full HTML table formatting options are available and the output presented here uses one of a great number of combinations of these options (colors, fonts, sizes, etc.).

The result of one `%TAB2HTM` macro is an HTML document containing the output from one `TABULATE` procedure translated into HTML tagged code. The name of this output file is contained as a parameter in the macro code. As an example, the actual names of the HTML files as displayed via browser in Figs 5-8 are respectively:

simple regxdiv_1999_06_01.htm
simple Toys_regxitm_1999_06_01.htm
simple NE_stxdiv_1999_06_01.htm
simple NE_Toys_stxitm_1999_06_01.htm.

RPC Entertainment Enterprises
Region by Division Sales Report: Jun 1, 1999

All Regions by All Divisions	Division		TOTAL
	Games	Toys	
Region			
NorthCentral	10,490	9,590	20,080
NorthEast	11,940	12,828	24,768
NorthWest	5,202	5,626	10,828
SouthCentral	6,070	6,314	12,384
SouthEast	9,990	9,990	19,980
SouthWest	7,389	6,277	13,666
TOTAL	51,081	50,625	101,706

Fig 5. Simple HTML Report
Region by Division

RPC Entertainment Enterprises
State by Division Sales Report: Jun 1, 1999

Region: NorthEast	Division		TOTAL
	Games	Toys	
State			
CT	631	1,075	1,706
DC	1,319	1,763	3,082
DE	207	631	838
MA	1,075	1,319	2,394
MD	1,763	207	1,970
ME	631	1,075	1,706
NH	1,319	1,763	3,082
NJ	207	631	838
NY	1,075	1,319	2,394
PA	1,763	207	1,970
RI	631	1,075	1,706
VT	1,319	1,763	3,082
TOTAL	11,940	12,828	24,768

Fig 6. Simple HTML Report
State by Divison (Region: NorthEast)

RPC Entertainment Enterprises
Region by Item Sales Report: Jun 1, 1999

Division: Toys	Item		TOTAL
	GI Joe	SI Jim	
Region			
NorthCentral	4,740	4,850	9,590
NorthEast	5,748	7,080	12,828
NorthWest	2,480	3,146	5,626
SouthCentral	2,824	3,490	6,314
SouthEast	4,440	5,550	9,990
SouthWest	2,750	3,527	6,277
TOTAL	22,982	27,643	50,625

Fig 7. Simple HTML Report
Region by Item (Division: Toys)

RPC Entertainment Enterprises
State by Item Sales Report: Jun 1, 1999

Region: NorthEast Division: Toys	Item		TOTAL
	GI Joe	SI Jim	
State			
CT	482	593	1,075
DC	826	937	1,763
DE	260	371	631
MA	604	715	1,319
MD	48	159	207
ME	482	593	1,075
NH	826	937	1,763
NJ	260	371	631
NY	604	715	1,319
PA	48	159	207
RI	482	593	1,075
VT	826	937	1,763
TOTAL	5,748	7,080	12,828

Fig 8. Simple HTML Report
State by Item (Region: NE
Division: Toys)

ENHANCED %TAB2HTM OUTPUT

Now that we have the individual reports rendered as HTML documents, the next step is to create the navigational tools to be able to go from one report to another by clicking on a hot link. This part is the crux of the whole method presented here, and is actually really very simple. All that needs to be done is to create a set of alternate variables from which the tables are constructed. The change that is necessary is to enhance each data item (that is to be displayed as a hot link) with additional location information contained in HTML tags, specifically HREF tags. As an example, each region is actually coded as a two-byte character variable called `REG` with values of: NC, NE, NW, SE, SC, or SW. The non-HTML output uses a user-defined format, `$REGFMT.`, to display expanded names via a `FORMAT` statement in the `TABULATE` code. If a separate *State by Division* HTML table was created for each region, they could have file names of:

enhanced NC_stxdiv_1999_06_01.htm
enhanced NE_stxdiv_1999_06_01.htm
enhanced NW_stxdiv_1999_06_01.htm
enhanced SC_stxdiv_1999_06_01.htm
enhanced SE_stxdiv_1999_06_01.htm
enhanced SW_stxdiv_1999_06_01.htm.

If we then created an alternate variable `REG2` by surrounding the old values of `REG` with the above path names along with the appropriate HTML code, we could then produce HTML clickable links in our `TABULATE` output by using `REG2` in the `TABULATE` code instead of `REG`. The code to create `REG2` would look something like:

```
reg2 = "<A HREF='enhanced "
       || trim(reg)
       || "_stxdiv_1999_06_01.htm'>"
       || trim(put(reg,$regfmt.))
       || '</A>';
```

This would produce a converted value for NE of:

<A HREF='enhanced NE_stxdiv_1999_06_01.htm'>NE</A> .

Actually, there would have to be two alternate versions of `REG`, a `REG2` and a `REG3`, because clicking on a region name in the *Region by Division* table would have to link to a *State by Division* table for the clicked region, whereas clicking on the same region name in a *Region by Item* table would link to a *State by Item* table for the clicked region. Since the names of the two target files would be different, the values of `REG2` and `REG3`, both based on `REG`, would be different. Two alternate versions of `DIV` would be needed as well.

If the new versions of `REG` and `DIV` were used in the same `TABULATE` code that was used to produce the tables shown in Figs 5-8, the `%TAB2HTM` macro would cause the region and division names to be displayed as clickable HTML hot links. The resulting HTML output files as seen via an HTML browser are shown in Figs. 9-12.

RPC Entertainment Enterprises
Region by Division Sales Report: Jun 1, 1999

All Regions by All Divisions	Division		TOTAL
	Games	Toys	
Region			
NorthCentral	10,490	9,590	20,080
NorthEast	11,940	12,828	24,768
NorthWest	5,202	5,626	10,828
SouthCentral	6,070	6,314	12,384
SouthEast	9,990	9,990	19,980
SouthWest	7,389	6,277	13,666
TOTAL	51,081	50,625	101,706

Fig 9. Enhanced HTML Report Region by Division

RPC Entertainment Enterprises
State by Division Sales Report: Jun 1, 1999

Region: NorthEast	Division		TOTAL
	Games	Toys	
State			
CT	631	1,075	1,706
DC	1,319	1,763	3,082
DE	207	631	838
MA	1,075	1,319	2,394
MD	1,763	207	1,970
ME	631	1,075	1,706
NH	1,319	1,763	3,082
NJ	207	631	838
NY	1,075	1,319	2,394
PA	1,763	207	1,970
RI	631	1,075	1,706
VT	1,319	1,763	3,082
TOTAL	11,940	12,828	24,768

Fig 10. Enhanced HTML Report State by Division (Region: NE)

That's basically all there is to it. Just substitute enhanced data values for those values that you want to act as hot links and let `%TAB2HTM` do the rest. Of course there is more to the overall method to deal with, but let's take care of some glossed over details first. Notice that underscores are used throughout in file names instead of dashes, including in the date portions of the names. The reason

RPC Entertainment Enterprises
Region by Item Sales Report: Jun 1, 1999

Division: Toys	Item		TOTAL
	GI Joe	SI Jim	
Region			
NorthCentral	4,740	4,850	9,590
NorthEast	5,748	7,080	12,828
NorthWest	2,480	3,146	5,626
SouthCentral	2,824	3,490	6,314
SouthEast	4,440	5,550	9,990
SouthWest	2,750	3,527	6,277
TOTAL	22,982	27,643	50,625

Fig 11. Enhanced HTML Report Region by Item (Division: Toys)

RPC Entertainment Enterprises
State by Item Sales Report: Jun 1, 1999

Region: NorthEast Division: Toys	Item		TOTAL
	GI Joe	SI Jim	
State			
CT	482	593	1,075
DC	826	937	1,763
DE	260	371	631
MA	604	715	1,319
MD	48	159	207
ME	482	593	1,075
NH	826	937	1,763
NJ	260	371	631
NY	604	715	1,319
PA	48	159	207
RI	482	593	1,075
VT	826	937	1,763
TOTAL	5,748	7,080	12,828

Fig 12. Enhanced HTML Report State by Item (Region: NE Division: Toys)

for this is that `TABULATE` will always insert a line break at a dash in a value in a column or row header if it sees it. This wreaks havoc with the constructed HTML code so it's safer to just not use dashes anywhere. When this method is actually used on a daily basis to create daily reports, the SAS `TODAY()` function is used for daily dates, although its form is modified with the following statement:

```
%let today1=%sysfunc(translate(%sysfunc
          (today(),yymmdd10.),'_','-'));
```

This creates date values of the form 1999_06_01. The actual code for the REG2 assignment statement is then:

```
reg2 = "<A HREF='enhanced "
       || trim(reg)
       || "_stxdiv_&today1..htm'>"
       || trim(put(reg,$regfmt.))
       || '</A>';
```

Those pesky line breaks (and other similar problems) tend to crop up elsewhere as well and are dealt with in a wholesale fashion by passing the result of each %TAB2HTM operation though a text translation process found in macro %TABOUT defined later in this paper.

Another date transformation is used to get the formatted date in the titles (each TITLE statement has &TODAY2 in it):

```
%let today2=%sysfunc(today(),worddate12.);
```

ENHANCED %DS2HTM OUTPUT

OK, now we know how to create hot links in the row and column headers in the %TAB2HTM output table. But, there's more to it than that. We can use similar techniques with %DS2HTM, the web publishing tool that creates an HTML table from a SAS data set. The following code, when run through %DS2HTM, creates the HTML report *Table of Contents* page as shown in Fig. 13 (as viewed through an HTML browser.) The three string values of REP are broken here to fit into the column requirements of this paper; in actual code they are each one continuous string.

```
data reps;
   length rep $ 77;
   label  rep="Reports for &today2";
   rep="<A HREF='regxdiv_&today1..htm'>
   Region by Division Sales Report</A>";
   output;
   rep="<U>Report 2 (<I>non-operational
   </I>)</U>";
   output;
   rep="<U>Report 3 (<I>non-operational
   </I>)</U>";
   output;
run;
```

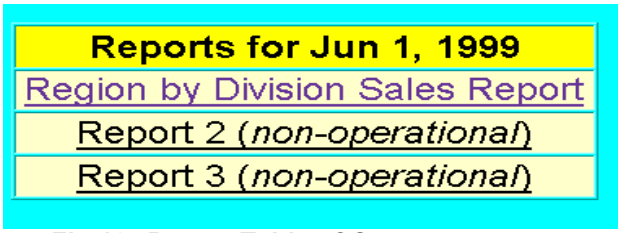

Fig 13. Report Table of Contents

We can also use %DS2HTM to create a *Calendar* page (as viewed via an HTML browser in Fig 14). New reports are created every day with the creation date used as part of the file name for each table (HTML file). The code used to create the calendar is presented below. Dates for which reports are available are visually highlighted in addition to the HTML underlining (note for example that there are no reports for Mon, June 7). This is accomplished through the use of Cascading Style Sheet code in the %OUT2DS macro. The method also includes a start date for the first month to be displayed.

```
%macro calendar;
  data day;
     length date $ 38;
     date="<A HREF=reps_toc_&today1..htm>"
          ||put(today(),day2.)||"</A>";
     datex = today();                                    run;
  *-------------------------------------------------;
  proc datasets;
     append base=hot._days new=day;                      run;
  *-------------------------------------------------;
  proc sort data=hot._days nodupkey;  by datex;    run;
  *-------------------------------------------------;
  proc format;
     value $miss (default=38) ' '=' ';
     value $hil  ' ' - '31' = 'calendarVariable'
                      other = 'calendarHighlight'; run;
  *-------------------------------------------------;
  data alldays(drop=start d);
    length date $ 34;
    start = '01may1999'd;
    d     = 0;
    do until (datex=intnx('month',today(),1)-1);
       datex = start + d;
       date  = put(datex,day2.);
       year  = year(datex);
       month = month(datex);
       output;
       d + 1;
    end;                                                 run;
  *-------------------------------------------------;
  proc sort data=alldays;       by datex;          run;
  *-------------------------------------------------;
  data alldays;
     update alldays hot._days;  by datex;          run;
  *-------------------------------------------------;
  proc sort data=alldays;  by year month datex;    run;
  *-------------------------------------------------;
  data cal(keep=mon_yr sun mon tue wed thu fri sat);
    set alldays end=lastrec;
    by year month;
    length          sun mon tue wed thu fri sat $ 34;
    array days{7} $ sun  mon  tue  wed  thu  fri  sat;
    array daysx{7}  sunx monx tuex wedx thux frix satx;
    format          sun mon tue wed thu fri sat $miss.;
    retain days;
    mon_yr = intnx('month',datex, 0);
    label mon_yr = "Date";

    if      weekday(datex)=1 then do;
            sunx=datex;  sun=date;  end;
    else if weekday(datex)=2 then do;
            monx=datex;  mon=date;  end;
    else if weekday(datex)=3 then do;
            tuex=datex;  tue=date;  end;
    else if weekday(datex)=4 then do;
            wedx=datex;  wed=date;  end;
    else if weekday(datex)=5 then do;
            thux=datex;  thu=date;  end;
    else if weekday(datex)=6 then do;
            frix=datex;  fri=date;  end;
    else if weekday(datex)=7 then do;
            satx=datex;  sat=date;  end;

    if last.month or lastrec then
    do i=(weekday(datex)+1) to 7;
       if daysx[i] lt datex then days[i] = ' ';
    end;
```

```
    if first.month or _n_=1 then
    do i=1 to (weekday(datex)-1);
       days[i] = ' ';
    end;

    if weekday(datex) = 7 or last.month or lastrec then
    output;                                              run;
*-----------------------------------------------------;
  title1 "Calendar";                                     run;
*-----------------------------------------------------;
  %ds2htm(runmode=b,   openmode=replace,    encode  =n,
          center =y,   bwrap   =n,          formats =y,
          septype=none,data    =cal, htmlfref=calendar,
          by =mon_yr(order=descending, format=monyy7.,
          class=calendarByline),       labels=y,
          var  =sun(classfmt=$hil.) mon(classfmt=$hil.)
                tue(classfmt=$hil.) wed(classfmt=$hil.)
                thu(classfmt=$hil.) fri(classfmt=$hil.)
                sat(classfmt=$hil.),
          sshref1=_hotlinks.css, sstype1=text/css,
          ssrel1=stylesheet,
          bclass=calendarByline,
          clclass=calendarColLabel);
   *-----------------------------------------------------;
   title1 "<A HREF='_cal.htm'>Calendar</A>";
   run;
%mend calendar;
```

Calendar

Date=JUN1999

SUN	MON	TUE	WED	THU	FRI	SAT
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Date=MAY1999

SUN	MON	TUE	WED	THU	FRI	SAT
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Fig 14. Calendar

HTML ENHANCED TITLES

The next parts of the process needed to create the fully navigational system are hot-link enhanced titles. These are easy to do because they come along for a free ride when the TABULATE output or the SAS data set is transformed into HTML code. The trick is to make sure that the correct titles appear at the top of each page. Parts of the titles can be generally modularized as follows:

```
%let mcal = <A HREF=_cal.htm>Calendar</A>;
%let mrep = <A HREF=reps_toc_&today1..htm>
               Reports</A>;
```

These would be used to produce the hot links to go to the *Calendar* page and a day-specific *Table of Contents* page, as in:

```
title1 "<H3>RPC Entertainment Enterprises
        <BR>Region by Division Sales
            Report: &today2</H3>";
title2 "<H4>&mcal &mrep</H4>";
```

Other titles would be constructed using the same general format.

DATA-DRIVEN MACRO AUTOMATION

To make the whole system automatic and data driven, it should all be contained in a system of macros. In this case it works as follows. %DS2HTM is used to recreate the *Calendar* page each day, and to create a new *Table of Contents* page each day. Each day the source data set is re-created as a SAS data set called HLDATA. This data set contains variables REG, DIV and SALES. An alternate data set, HLDATA2 is created with the converted variables REG2, REG3, DIV2 and DIV3, along with SALES. Next, a macro called %TABOUT is compiled which contains the second necessary %TAB2HTM macro call for each TABULATE (capture off, and HTML formatting) along with the input/output text transformation code (to handle line breaks, etc). Note that Cascading Style Sheet information is used to format various parts of the tabular output.

```
%macro tabout(rowxcol=);
   %tab2htm(capture  = off,
            runmode  = b,
            htmlfref = &rowxcol,
            twidth   = 100,
            encode   = n,
            openmode = replace,
            center   = y,
            septype  = none,
            ttag     = no formatting,
            sshref1  =_hotlinks.css,
            sstype1  = text/css,
            ssrel1   = stylesheet,
            bxhalign = left,
            bxvalign = middle,
            bxclass  = reportBoxCell,
            cclass   = reportCaption,
            clclass  = reportColLabel,
            rlclass  = reportRowLabel,
            dclass   = reportDataCell
           );
   *-----------------------------------;
```

```
data _null_;
      infile &rowxcol missover
             length=linelen;
      input line $varying200. linelen;
      if  (index(line,'<TH')
      and  index(line,'HREF=')
      and  index(line,'<BR>'))
      then do;
         line=tranwrd(line,'-<BR>','?');
         line=tranwrd(line,'<BR>','?');
         line=compress(line,'?');
         line=tranwrd(line,'AHREF',
                           'A?HREF');
         line=translate(line,' ','?');
         line=translate(line,'_','-');
      end;

      file &rowxcol;
      put line $varying200. linelen;
   run;
%mend tabout;
```

Each different type of TABULATE (*Region by Division, State by Division, Region by Item, State by Item*) is contained in a table macro (%REGXDIV, %STXDIV, %REGXITM, %STXITM) along with its initial %TAB2HTM call (capture on) and its %TABOUT call. The code for one of these table macros (%STXDIV) is as follows (the rest are similar):

```
%macro stxdiv(mreg=);
   %let mregbox = %qsysfunc(
                  putc(&mreg,$regfmt.));
   %let allregs = <A
               HREF=regxdiv_&today1..htm>
               All Regions</A>;

   title1 "<H3>RPC Entertainment
            Enterprises<BR>
            State by Division Sales Report:
             &today2</H3>";
   title2 "<H4>&mcal &mrep &allregs</H4>";
   run;
   *-------------------------------------;
   filename stxdiv
            "&mreg._stxdiv_&today1..htm";
   *-------------------------------------;
   %tab2htm(capture=on,
            runmode=b);
   *-------------------------------------;
   proc tabulate missing
        data=hldata2(where=(reg="&mreg"));
      class  state div3;
      var    sales;
      table  state='State' all='TOTAL',
             (div3='Division' all='TOTAL')
             *sales=''*sum=''*f=comma32.
             / box="Region:  &mregbox";
   run;
   *-------------------------------------;
   %tabout(rowxcol=stxdiv)
   *-------------------------------------;
%mend stxdiv;
```

These table macros are called as needed by using a data-driven macro driver. The code for this driver is in two parts. The first part (%MVARS) creates a set of driver macro variables needed to run the appropriate table macros. It is totally data-driven and contains information about the values of REG and DIV present in the source data.

```
%macro mvars;
   %global mregct mdivct mregs mdivs;
   proc sql noprint;
      create table regdivs as
         select distinct reg, div
         from hldata;

         select count(*)
         into   :mregct
         from   (select distinct reg
                 from   regdivs);

         select count(*)
         into   :mdivct
         from   (select distinct div
                 from   regdivs);

         select distinct reg
         into   :mregs SEPARATED BY ' '
         from   regdivs;

         select distinct div
         into   :mdivs SEPARATED BY ' '
         from   regdivs;
   quit;
%mend mvars;
```

The results of the above macro execution are four macro variables with the following values:

```
mregct - 6
mdivct - 2
mregs  - NC NE NW SC SE SW
mdivs  - Games Toys
```

The second part of the macro driver (%RUNALL) calls the needed table macros, along with all the other needed macros. It is self contained and is made up of modules which are turned on or off by macro switches at the top of the program.

```
%macro runall;
  %if %upcase(&calendar)=Y %then %calendar;
  %if %upcase(&repstoc) =Y %then %repstoc;
  %if %upcase(&hldata2) =Y %then %hldata2;
  %if %upcase(&mvars)   =Y %then %mvars;
  %if %upcase(&regxdiv) =Y %then %regxdiv;
  %if %upcase(&stxdiv)  =Y %then
  %do r=1 %to &mregct;
     %stxdiv(mreg=%qscan(&mregs,&r));
  %end;
  %if %upcase(&regxitm) =Y %then
  %do d = 1 %to &mdivct;
     %regxitm(mdiv=%qscan(&mdivs,&d));
  %end;
  %if %upcase(&stxitm)  =Y %then
```

```
  %do r = 1 %to &mregct;
     %do d = 1 %to &mdivct;
        %stxitm(mreg=%qscan(&mregs,&r),
                mdiv=%qscan(&mdivs,&d));
     %end;
  %end;
%mend;
```

The entire process is executed by running `%RUNALL`. The final, fully HTML navigational system, comprised of the calendar, report table of contents and all component table reports, is shown in Figs 15-20, as viewed through an HTML browser.

HIGHLIGHTING

A companion paper to the present effort entitled *Let There Be Highlights: Data-driven Cell and Row Highlights in %TAB2HTM and %DS2HTM Output*, by Matt Flynn and Ray Pass, presents methods for creating data-driven conditional highlights of various elements of the HTML output from the Web Publishing macros. When used in tandem, the complementary techniques significantly enhance the potential productivity of these powerful tools.

Calendar

Date=JUN1999

SUN	MON	TUE	WED	THU	FRI	SAT
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Date=MAY1999

SUN	MON	TUE	WED	THU	FRI	SAT
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Fig 15. Final Calendar

Calendar

Reports for Jun 1, 1999

Region by Division Sales Report

Report 2 (*non-operational*)

Report 3 (*non-operational*)

Fig 16. Final Report Table of Contents

RPC Entertainment Enterprises
Region by Division Sales Report: Jun 1, 1999

Calendar Reports

All Regions by All Divisions	Division		TOTAL
	Games	Toys	
Region			
NorthCentral	10,490	9,590	20,080
NorthEast	11,940	12,828	24,768
NorthWest	5,202	5,626	10,828
SouthCentral	6,070	6,314	12,384
SouthEast	9,990	9,990	19,980
SouthWest	7,389	6,277	13,666
TOTAL	51,081	50,625	101,706

Fig 17. Final HTML Report Region by Division

RPC Entertainment Enterprises
Region by Item Sales Report: Jun 1, 1999

Calendar Reports All Divisions

Division: Toys	Item		TOTAL
	GI Joe	SI Jim	
Region			
NorthCentral	4,740	4,850	9,590
NorthEast	5,748	7,080	12,828
NorthWest	2,480	3,146	5,626
SouthCentral	2,824	3,490	6,314
SouthEast	4,440	5,550	9,990
SouthWest	2,750	3,527	6,277
TOTAL	22,982	27,643	50,625

Fig 18. Final HTML Report Region by Item (Division: Toys

RPC Entertainment Enterprises
State by Division Sales Report: Jun 1, 1999

Calendar Reports All Regions

Region: NorthEast	Division		TOTAL
	Games	Toys	
State			
CT	631	1,075	1,706
DC	1,319	1,763	3,082
DE	207	631	838
MA	1,075	1,319	2,394
MD	1,763	207	1,970
ME	631	1,075	1,706
NH	1,319	1,763	3,082
NJ	207	631	838
NY	1,075	1,319	2,394
PA	1,763	207	1,970
RI	631	1,075	1,706
VT	1,319	1,763	3,082
TOTAL	11,940	12,828	24,768

Fig 19. Final HTML Report
State by Division (Region: NE)

RPC Entertainment Enterprises
State by Item Sales Report: Jun 1, 1999

Calendar Reports All Regions All Divisions

Region: NorthEast Division: Toys	Item		TOTAL
	GI Joe	SI Jim	
State			
CT	482	593	1,075
DC	826	937	1,763
DE	260	371	631
MA	604	715	1,319
MD	48	159	207
ME	482	593	1,075
NH	826	937	1,763
NJ	260	371	631
NY	604	715	1,319
PA	48	159	207
RI	482	593	1,075
VT	826	937	1,763
TOTAL	5,748	7,080	12,828

Fig 20. Enhanced HTML Report
State by Item (Region: NE
Division: Toys)

CONCLUSION

This paper has given an introduction to an optional method of turning the static HTML documents created by the SAS Web Publishing Tool macros into a fully functional, HTML hot-link navigational, data-driven system of information display that can be implemented on a routine basis in an internet or intranet environment. Although the system as explicated here is based on the pre-Version 7 Web Publishing Tools, it is also applicable to the new Output Delivery System implemented in Version 7 of the SAS System. The basic paradigm is to enhance the data being used as input to the tools or ODS, so the values displayed can be rendered by an HTML browser as fully functional HTML hot links.

The method deals with outputs from runs of the `TABULATE` procedure which are processed with `%TAB2HTM`, as well as SAS data sets which are displayed via `%DS2HTM` and HTML enhanced `TITLE` statements. In addition, a method is displayed in which data-driven executions of the necessary procedures and data set creations are run automatically.

The author has used these techniques successfully on a large nationwide intranet system with extremely productive results. In fact, the initial implementation has spawned numerous other similar systems on the same intranet. These techniques have also been used with these sample data in a Version 7 environment using ODS for the HTML rendering. Stay tuned for that write-up, or better yet, go out and create your own systems in Version 6 or 7. The tools are there.

APPENDICES

A complete working program with associated sample data sets and adjunct files can be found on the CD version of the NESUG '99 Proceedings, and on the NESUG Website. The files are contained in the '168' series (168a.txt, 168b.txt, etc.) Instructions can be found in 168a.txt.

REFERENCES

CONTACT INFORMATION

The author can be contacted at:

Ray Pass
Ray Pass Consulting
5 Sinclair Place
Hartsdale, NY 10530

(914) 693-5553
raypass@att.net