Paper TT15

How To Get Ten Pounds Of Data Into A Five Pound Can – Using Flyover Technique In HTML Output

Daniel K. Downing, CaremarkPCS, Scottsdale, AZ

ABSTRACT

When delivering results today, it is more and more common that delivering the output to the screen will be a requirement. Without getting into issues of font size or page orientation, there is generally about one third less horizontal space to work with between a screen and a hardcopy document. However, there is an easy to use technique available that allows for the presentation of more data to the user without requiring them to scroll back and forth. This paper is a guide through the processes necessary to convert static hardcopy output from PROC PRINT to the more flexible PROC REPORT, use flyovers to present descriptive information, thereby reducing the screen real estate consumed, and package it all in an HTML wrapper. In other words, a way of getting ten pounds of data into a five pound can.

INTRODUCTION

Once upon a time, a *long* time ago, ancient programmers actually had to design reports. Metal rulers that contained 132 spaces were used to lay out where each and every character of the output would land. There were no fonts to control. There was no PROC PRINT to perform it automatically. That was all the space there was to work with, 132 characters, and you liked it. Then the PC changed that world. The ability to deliver results to a screen based presentation dramatically reduced the need for the traditional hardcopy report. While making output available to the screen certainly conserves resources and increases accessibility, how does one get all that information in a limited amount of space? An easy way to maximize the available space without requiring the user to scroll left and right is to use the **style=flyover** method in PROC REPORT.

THE NEED

First, the particular hardcopy report should be evaluated to see if it is an appropriate candidate for screen deployment with flyovers. While the chances are good that someday ALL reports will have hardcopy/softcopy options, the information contained in the flyover does not materialize when printed from the screen, so if it is imperative that all the variables be available to print, hold on to the original program. In the real world case that led to the exploration of this technique, the user was looking to visual identify patterns on the screen to assist in the detection of potential fraud and possible prescription drug abuse. The trigger to include a member was based on data provided from the pharmacy at the time the prescription was filled. The data is flagged if it shows that the member had a date of birth different than what was on file. Showing data on the screen allows the fraud investigator to see if the anomaly is explainable, or is a situation requiring further research. An explainable event might be a transposition of numbers, such as Year of Birth = 1956 vs. 1965. A case requiring further investigation might be an 18 year age difference with four prescriptions for a class three narcotics from two different pharmacies in a one month period. The user does not need to see all the details on the screen, all the time. It is only necessary to see the high level identifiers and have the ability to get to additional information if any points of interest are uncovered.

THE DATA

The data involved consists of unique identifiers, the corresponding descriptive values, and associated transaction details. In the prescription data case, those variables consist of, who, from where, of what, how many, for how long and for how much. Since this application was originally developed to help detect potential member level fraud, there was of course, patient level information presented. However, in this age of HIPAA compliancy, not even imaginary patient data will be included, as none of us are probably authorized to see it! The portion of code shown below defines the variables that will be used to meet **THE NEED**. A complete version of the program solution necessary to show an example of this technique, including the creation of the demo data, can be found at the end of this paper.

```
data work.scripts ;
     length DR_NAME      PHARM_NAME   DRUG_NAME      $ 35
            MD_NB        PHARM_NB     RX_NB          $ 9
            DRUG_QTY     DAY_SPLY     REFILL_NB  CNT   3   AMOUNT 5.2 ;
```

THE CAN

Obviously, the only way to get ten pounds of data into a five pound can is to possess a magical can! For this processing, that magic manifests itself in the interconnected interface between the features of PROC REPORT and the Output Delivery System (ODS). The technique used in this transformation from hardcopy to “screen ready” (it is the HTML wrapper where the actual display happens) is done in a three step process. Those steps are, converting the procedure, defining to PROC REPORT the variables in **THE DATA** that will be used as the flyover values, and performing “the squeeze” to associate the displayed values with their flyover components.

CHANGE THE PROCEDURE

The first part of the solution involves transforming a PROC PRINT to a PROC REPORT. In the simplest form, change the procedure name from PRINT to REPORT and the variable selection from VAR to COLUMN. Done.

BEFORE:

```
  PROC PRINT data=scripts  ;

      VAR  DR_NAME   PHARM_NAME  DRUG_QTY   DAY_SPLY  REFILL_NB    CNT
            FILL_DT   DRUG_NAME   PHARM_NB   MD_NB     RX_NB        AMOUNT  ;
```

AFTER:

```
  PROC REPORT data=scripts  ;

      COLUMN  DR_NAME  PHARM_NAME   DRUG_QTY    DAY_SPLY   REFILL_NB   CNT
               FILL_DT  DRUG_NAME    PHARM_NB    MD_NB      RX_NB       AMOUNT  ;
```

DEFINE THE VARIABLES THAT WILL COMPRISE FLYOVER RESULTS

The second part of the transformation is to determine what values reside on the screen and what values contribute to the flyover. One way to do this is an examination of which variables are the unique identifiers and which variables are the corresponding descriptive values. Once it is determined what will be shown and what will be flown, those decisions need to be coded into the program. The variables that will be used to construct a flyover will never be seen (only the values will), so the define statement utilizing the NOPRINT option is used to keep them from appearing in the PROC REPORT output.

```
   define  DR_NAME    / display noprint ;
   define  PHARM_NAME / display noprint ;
   define  DRUG_QTY   / display noprint ;
   define  DAY_SPLY   / display noprint ;
   define  REFILL_NB  / display noprint ;
   define  CNT        / display noprint sum  ;
```

The variables that will be displayed do not have to be included in the define section, unless there is a need to affect the style or identify them for other purposes, such as breaks or totals.

APPLYING DIRECT PRESSURE TO SQUEEZE THE DATA

The third part of the process is where all the magic work is done. As mentioned above, the actual flyover variable is not displayed to the screen. What is shown on the screen is a macro variable containing the value from the variable. The processing is done inside a compute block for each unique identifier variable that corresponds to the descriptive value. It occurs in two steps. First, the descriptive value is placed into a macro variable with the call symput function. Next, call define is used to assign a style to the displayed value. That is where the flyover style is declared, the value is constructed and attribute assignments take place. In this example, both a text string and the macro variable are being placed into the flyover. Additionally, the font is being set to bold and the color changed to visually inform the user that there is additional information available. The final attribute is the assignment of a cursor value, again used to capture the eye of the user and draw their attention to what is available if they place the cursor over the information displayed on the screen.

```
   compute MD_NB   ;
      call symput('DR_NAME',trim(left((DR_NAME)))) ;
      call define(_col_, 'style',
         'style={flyover="Provider Name - &DR_NAME"
         font_weight=bold foreground=cx639ACE htmlstyle="cursor:hand"}');
   endcomp;
```

If there is a need to display several descriptive values, multiple macro variables can be placed into the flyover.

```
   compute DRUG_NAME  ;
      call symput('QTY',trim(left((put(DRUG_QTY,comma12.))))) ;
      call symput('DAY_SUP',trim(left((put(DAY_SPLY,comma12.)))));
      call define(_col_, 'style',
         'style={flyover="Quantity - &QTY     Days Supply - &DAY_SUP"
         font_weight=bold foreground=cx639ACE htmlstyle="cursor:hand"}');
   endcomp;
```

THE WRAPPER

Every can, even magical ones, can benefit from a pretty label. Just ask marketing! The final process in this transformation is to bring the results to the screen. After all, that is the only place a flyover provides any value. The ODS option used in conjunction with the HTML destination allows this to happen. Inside the ODS there is a rich portfolio of techniques that will make **THE CAN**, now stuffed full of information, look even more attractive. At the fundamental level, the ODS must be turned on, parameters set, procedures executed, and then turned off. The source code at the end of this paper shows just a few of the many ODS features available. When the code is executed, the HTML output is written to the default path in the "Start In" or "Current" folder of the session, which is usually displayed on the bottom, right side of the session. To view the results generated by running the code, open the "`Scripts frame.html`" file.

BUT WAIT… THERE'S MORE!

After a solution is implemented, there are always opportunities for enhancements. This development effort was no exception. Here are some possibilities.

CURSOR SELECTION

Depending on the browser that is being used to view the output, there are several different cursors you can select to use. To do this the "hand" part of the text `htmlstyle="cursor:`**`hand`**`"` attribute assignment can be changed to any of the following values: crosshair, help, move, hand, arrow (default).

WHAT IF THERE IS FIFTEEN POUNDS OF DATA?

Sometimes there are multiple variables containing descriptive or supporting detail information that need to be displayed in a horizontal fashion. This can be achieved by forcing the flyover to wrap. An example of this might be the Date of Fill for the last ten prescriptions. It is necessary to "tune" the flyover with spaces to make the overflow occur at the appropriate break point. **CAUTION -** This technique is sensitive to space, both between the text constant and the macro variables, as well as the resolved length of the values contained in the macro variables. When assembling the flyover value from multiple components, construct it as one long contiguous line of code. This will help control the spaces. The decision of when to wrap takes place internal to PROC REPORT and occurs somewhere between sixty and one hundred twenty five characters. It is influenced by factors such as the number of spaces embedded within the data, font definitions, and any spaces that present an opportunity to break once the overflow threshold has been reached. Be careful and check the results, as the use of varying length data may deliver a different parsing for each observation.

```
   compute PHARM_NB ;
      call sympput('PHARM_NAME',trim(left((PHARM_NAME)))) ;
      call sympput('DOF_1',trim(left((DOF_1)))) ;
      call sympput('DOF_2',trim(left((DOF_2)))) ;  . . .
      call define(_col_, 'style',
         'style={flyover="Date Of Fill - &DOF1                Date Of Fill - &DOF2 ..."
         font_weight=bold foreground=cx639ACE just=c htmlstyle="cursor:hand"}');
   endcomp;
```

CONCLUSION

This technique provides a simple solution for making more data available to the user without requiring drill downs or scrolling from left to right. It may not be appropriate if the user prints the HTML and needs the information contained in the flyover available on the hardcopy.

ACKNOWLEDGMENTS

Thanks to all the users at CaremarkPCS who allow me the privilege of using SAS® to meet their informational needs.

CONTACT INFORMATION

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

Daniel K. Downing
CaremarkPCS
9501 E. Shea Blvd.
Scottsdale AZ, 85260
Work Phone: 480.391.4282
Email: daniel.downing@caremark.com

THE SOURCE CODE

```
* CREATE DEMONSTRATION DATA  ;

  data work.scripts ;

    length MD_NB    PHARM_NB    RX_NB          $ 9
           DR_NAME  PHARM_NAME  DRUG_NAME      $ 35
           DRUG_QTY DAY_SPLY    REFILL_NB  CNT    3 AMOUNT 5.2 ;

    DR_NAME = 'Dr. Strangelove' ; PHARM_NAME  = 'Drugs -R- Us #6349' ;
    DRUG_QTY = 60 ; DAY_SPLY = 60 ; REFILL_NB = 12 ; CNT = 1 ;
    FILL_DT = '20OCT2003'd ; DRUG_NAME = 'Love Potion Number 9' ;
    PHARM_NB = '290763822' ; MD_NB = 'BB8475309' ; RX_NB =  '1928374' ;
    AMOUNT      =  13.99 ;
    output ;

    DR_NAME = 'Dr. Spock MD' ; PHARM_NAME  = 'Mom & Pop Pill Shop' ;
    DRUG_QTY = 90 ; DAY_SPLY = 30 ; REFILL_NB = 0 ; CNT = 1 ;
    FILL_DT = '15JAN2004'd ; DRUG_NAME = 'Amoxicylin ' ;
    PHARM_NB = '987654321' ; MD_NB = 'AA1234567' ; RX_NB =  '2747123' ;
    AMOUNT      =  29.99 ;
    output ;

    DR_NAME = 'Dr. Quinn MW' ; PHARM_NAME  = 'Buckboard Apothecary & Dry Goods' ;
    DRUG_QTY = 60 ; DAY_SPLY = 60 ; REFILL_NB = 2 ; CNT = 1 ;
    FILL_DT = '28AUG2004'd ; DRUG_NAME = 'Bite The Bullet Pain Remedy' ;
    PHARM_NB = '281937465' ; MD_NB = 'WW0001849' ; RX_NB =  '8564739' ;
    AMOUNT      =  9.99 ;
    output ;

    label DR_NAME       = 'Doctor Name'
          PHARM_NAME = 'Pharmacy Name'
          DRUG_QTY      = 'Quantity'
          DAY_SPLY        = 'Days Supply'
          REFILL_NB       = 'Refill Number'
          CNT                  = 'Count'
          FILL_DT            = 'Date Of Fill'
          DRUG_NAME    = 'Drug Name'
          PHARM_NB       = 'Pharmacy Number'
          MD_NB              = 'DEA Number'
          RX_NB               = 'Prescription Number'  ;

  run ;

* SET UP ODS DESTINATION ;

  ods listing close ;

  ods html
     body     = "Scripts body.html"
     contents = "Scripts contents.html"
     frame    = "Scripts frame.html"
     style    =  d3d ;

  ods noproctitle ;
```

```
* USE PROC PRINT TO SHOW HTML SCREEN WITH AN OVERFLOW OF DATA    ;

  %let TITL_COL = red ;

  PROC PRINT data=scripts contents="Prescription Data" ;

    var DR_NAME   PHARM_NAME  DRUG_QTY  DAY_SPLY  REFILL_NB  CNT
        FILL_DT   DRUG_NAME   PHARM_NB  MD_NB     RX_NB      AMOUNT ;

            title1 "Prescription Level Detail Claims" ;
    title2 color=&TITL_COL height=2 "Using PROC PRINT To Show All Variables";

  run ;

* USE PROC REPORT TO PRODUCE PRESCRIPTION DATA ODS HTML OUTPUT   ;

  %let TITL_COL = green ;

  PROC REPORT nowd data=scripts headline contents="Prescription Data" ;

    column DR_NAME   PHARM_NAME DRUG_QTY DAY_SPLY  REFILL_NB  CNT
          FILL_DT   DRUG_NAME  PHARM_NB MD_NB RX_NB AMOUNT ;

    define  DR_NAME    / display noprint ;
    define  PHARM_NAME / display noprint ;
    define  DRUG_QTY   / display noprint ;
    define  DAY_SPLY   / display noprint ;
    define  REFILL_NB  / display noprint ;
    define  CNT        / display noprint sum  ;
    define  AMOUNT     / sum  'Amount' ;

    title1 "Prescription Level Detail Claims" ;
    title2 color=&TITL_COL height=2 "Demonstrating the FLYOVER Technique";

    footnote " "  ;

    compute DRUG_NAME  ;
      call symput
       ('QTY',trim(left((put(DRUG_QTY,comma12.))))) ;
      call symput
       ('DAY_SUP',trim(left((put(DAY_SPLY,comma12.)))));
      call define(_col_, 'style',
       'style={flyover="Quantity - &QTY     Days Supply - &DAY_SUP"
        font_weight=bold foreground=cx639ACE htmlstyle="cursor:hand"}');
    endcomp;

    compute PHARM_NB ;
      call symput('PHARM_NAME',trim(left((PHARM_NAME)))) ;
      call define(_col_, 'style',
       'style={flyover="Pharmacy Name - &PHARM_NAME"
        font_weight=bold foreground=cx639ACE just=c htmlstyle="cursor:hand"}');
    endcomp;

    compute MD_NB   ;
      call symput('DR_NAME',trim(left((DR_NAME)))) ;
      call define(_col_, 'style',
        'style={flyover="Provider Name - &DR_NAME"
        font_weight=bold foreground=cx639ACE htmlstyle="cursor:hand"}');
    endcomp;

    compute RX_NB ;
      if REFILL_NB LE 0 then do ;
         call define(_col_, 'style','style={just=c}');
      end ;
      if REFILL_NB > 0 then do ;
         call symput('REFILL_NB',trim(left((REFILL_NB)))) ;
         call define(_col_, 'style',
          'style={flyover="Refill # - &REFILL_NB"
           font_weight=bold foreground=cx639ACE just=c htmlstyle="cursor:hand"}');
      end ;
    endcomp;

    compute AMOUNT ;
      tot_amt = AMOUNT.sum  ;
      tot_cnt = cnt.sum ;
    endcomp;

    compute after ;

      length text1 $ 75 ;
      length text2 $ 75 ;

      text1 =
         "Total amount of claims is "
         || trim(left(put(tot_amt,dollar15.2))) ;

      text2 =
         "Total count of claims is "||
          trim(left(put(tot_cnt,comma15.)))||'.' ;

      line ' '         ;
      line text1 $75.  ;
      line text2 $75.  ;
      line ' '         ;
    endcomp ;

    rbreak after  / style=
     {background=cx639ACE foreground=white
      font_weight=bold font_size=4 }  ;

  run;


ods html close;
ods listing ;
```