

Funky Formats.

Zeke Torres, 1016inc.com, Chicago, IL

ABSTRACT

I will review a PROC TABULATE or PROC PRINT etc. and get a little fancy with ODS and PROC FORMAT adding HTML to it allow links to the Excel output which is also generated from this code. This gives a drill down effect and a quick way to allow end users to go thru lots of data in an easier way.

INTRODUCTION

In this example I employ a PROC FORMAT concept that incorporates HTML and gives the finished product a very good look and feel of "drill down" ability. You'll be able to take this example further and get some very good reports and HTML output easily with PROC FORMAT from now on. The output generated by ODS is not only in HTML but I create a printable PDF page for my end users. I'll go over a concept I started using to make it easy for the user to not only view the ODS HTML but have choices, ie PDF output, for print. I also set this program up to generate Microsoft Excel workbooks and they are linked to the HTML page via the links in the PROC FORMAT I mentioned earlier.

INTENDED AUDIENCE

This paper is intended for SAS users who have a technical skill level that begins at the higher 'beginner' level thru intermediate. The author has made every attempt to keep the examples of code focused around Base SAS. The use of supplemental software or other languages are general and it is assumed that the audience will obtain sufficient information via this abstract to adequately manage changes required by their own scenario(s).

ABOUT THE AUTHOR

Ezequiel 'Zeke' Torres is the President of 1016 Incorporated located in Chicago Illinois. With 10 years of SAS experience he is not just a SAS junkie but a "sas-aholic". Some of the clients that have benefited from his addiction to SAS are: Allstate Insurance, GE, Transamerica Distribution Finance, AEIG, Trustmark Insurance. Zeke also worked as an employee of Household International. Zeke develops end-to-end solutions using SAS for projects and business information needs clients present. This only feeds his addiction to SAS because creative solutions are not a choice, but a requirement.

OUTLINE OF THE PROCESS

PURPOSE OF THE JOB

Every morning managers need to see what the Delinquency level is for a specific portfolio. produce tabulate output with ODS. This ODS output will be in two types of output HTML and PDF. Our HTML is for 'online' viewing and sharing via the 'browser' method. We will also use the ODS PDF method to give the user the ability to print the same information or share it in hard copy format.

In this example the managers requested the ability to see the detail of what the TOTALS represent. In this code we use PROC FORMAT in the ODS statement and add HTML HREF tags to it. These tags will point to EXCEL files that have the detail. So via the ODS HTML output our user will see the report totals. They can click on the FORMAT Category and go to excel files which have the detail. Or they can use the PDF version to print and share the output.

So we will have one FORMAT to present without HTML code. One with HTML code. We will then use those formats to generate ODS that has a Title Page, with Links to Detail or source data. Lets try to remember that stepped process as we construct this example.

FILE REFERENCES AND PROC FORMAT WITH HTML.

```
%LET FT=XLS;  
%LET FOLD=SUG;
```

The &FT variable will tell my program what kind of file extension I will send data to in this program. It is required in the PROC FORMAT example we are about to see. I want to use the same format statement or code for multiple types of output in the future. Example today for PDF, later for XLS, or CSV.

```
PROC FORMAT;  
VALUE DELKYiia  
OTHER="    <strong>Zero to 29</strong></a>" 2="    <strong>30 to 59</strong></a>" 3="    <strong>60 to 89</strong></a>" 4="    <strong>90 to 119</strong></a>" 5="    <strong>120 to 149</strong></a>" 6="    <strong>150 to 179</strong></a>" 7="    <strong>180 to 109</strong></a>" 8-HIGH="    <strong>210 - Plus</strong></a>"  
;  
RUN;
```

The PROC FORMAT has HTML syntax we have set up for each of the values the FORMAT will encounter. This HTML will allow the user to see a report that gives them the ability to hover their mouse cursor over the 'FORMAT' value and if they click take them to this 'href' file. We have the ability to point them to another Web Page, or as in this case the detail of that FORMAT value. If we look closer we see that with this example we are using the Macro Variable "&FT" to let us use this code for multiple kinds of output. Say PDF, EXCEL (XLS), TEXT, you name it. However in the code that I am presenting there are 3 kinds of output. One for each part of the portfolio. So we need to set up distinct FORMAT outputs so that they point to the correct file(s).

* The example of this output is on page 8 of this presentation.

```
PROC FORMAT;  
VALUE DELKYiib  
OTHER="    <strong>Zero to 29</strong></a>" 2="    <strong>30 to 59</strong></a>" 3="    <strong>60 to 89</strong></a>" 4="    <strong>90 to 119</strong></a>" 5="    <strong>120 to 149</strong></a>" 6="    <strong>150 to 179</strong></a>" 7="    <strong>180 to 109</strong></a>" 8-HIGH="    <strong>210 - Plus</strong></a>" ;  
RUN;
```

```
PROC FORMAT;  
VALUE DELKYiic  
OTHER="    <strong>Zero to 29</strong></a>" 2="    <strong>30 to 59</strong></a>" 3="    <strong>60 to 89</strong></a>" 4="    <strong>90 to 119</strong></a>" 5="    <strong>120 to 149</strong></a>" 6="    <strong>150 to 179</strong></a>" 7="    <strong>180 to 109</strong></a>" 8-HIGH="    <strong>210 - Plus</strong></a>"  
;  
RUN;
```

We set up three distinct FORMAT statements. Each of which will provide 3 distinct types of HREF file names (a, b, c). They will be used in our ODS statement and in the specific step they will resolve to the values the code needs at that time. We could reduce this example to 1 FORMAT statement and use additional 'macro variables' to assist in the process of differentiating the output file HREF statements. But you would need to initiate the PROC FORMAT statement at key parts of the program.

The PROC FORMAT statement and the MACRO variable used here make this code flexible for adjusting output to go to various file types, Excel, PDF. However since it uses a MACRO variable the process does become STEP dependant or JOB specific. In a larger job with many more repetitive steps or outputs I would modify this concept to work in that fashion. With only 3 kinds of output I wanted to keep the statements and macro variables easy to manage as I made revisions for the users.

One note to the more MACRO oriented. Yes this code may have been generated by a macro or part of a macro scheme. I have opted to simply copy the code since this presentation is to benefit a wider audience. Additionally the original code had 'custom' ranges in delinquency levels that are not documented here. There are lots of places in this code that can benefit from 'macro' code.

We need one more FORMAT to present the information in our PDF output. This final FORMAT will not have HTML or HREF. It's the FORMAT we're generally familiar with.

This is essentially the same FORMAT we used in the earlier 3 examples with the only difference being the HTML is not used. In our PDF output we want to see these labels and not the HTML.

*Note. This paper is a subset of an original paper written in 2002. At that time ODS and output to PDF had not been developed fully within the SAS System. I had continually tried to get the hyperlink support to be part of the PDF output in ODS via SAS. I am not aware of any new developments at this time that facilitate that. However there may very well be some enhancements in SAS 9 that have solved this problem. That advancement would mean this step is optional. It would mean that your use of HTML PROC FORMAT would work in PDF and you would not need to use this PROC FORMAT.

```
PROC FORMAT;
VALUE DELKYYXI
  OTHER="Zero to 29"
    2=" 30 to 59"
    3=" 60 to 89"
    4=" 90 to 119"
    5="120 to 149"
    6="150 to 179"
    7="180 to 109"
    8-HIGH="210 - Plus"
;
RIIN:
```

The PDF output of this presentation is on page 11.

Ok lets get data to output to excel.

This can be any type of SAS code you need.

We are trying to populate the files (like excel) with data using SAS and a Macro.

This step will split the data into the parts that the PROC FORMAT represents.

Its very important to keep them in sync.

I have omitted lots of the DATA step code simply because it is arbitrary to the project you might have.

```
DATA
  ACCT1
  ACCT2
  ACCT3;
SET ACCT;
IF PRODUCT_TYPE IN ('202','201') THEN OUTPUT
ACCT1;
IF PRODUCT_TYPE GE 203 AND
  PRODUCT_TYPE LE 301 THEN OUTPUT ACCT2;
IF PRODUCT_TYPE GE 302 THEN OUTPUT ACCT3;
RUN;
```

This is my SAS code example. But here you would have your own code.

ON TO ODS HTML

```
ODS LISTING CLOSE;
ODS HTML BODY='XBODY'
  STYLE=SASWEB
  TRANTAB=ASCII
  FRAME='STARTPAGE.HTML'
  CONTENTS='DAILYYY.HTML'
  FILE='DAILYX.HTML'
;
OPTIONS MISSING=0 NOCENTER PAGENO=1
  LINESIZE=255;
```

Here we are simply specifying the output file names but note that we are not assigning a 'drive' or path. We want to make sure it's ambiguous so that no matter where these pages are viewed so long as they are in the same folder they will function as a unit. In this way we can distribute it among users and have little or no problems caused by different folder names or drive letter assignments. The other options shown here are my own preference.

Your Drive or Path assignment is up to your location and set up.

Lets take a look at the PROC TABULATE and implement the FORMAT

```
PROC TABULATE DATA= WORK.ACCT1 MISSING;
TITLE1 "1016INC.COM SAS EXAMPLE";
TITLE2 "OUTPUT ON PORTFOLIO - A -";
TITLE3 "PRODUCT TYPE REPORT";
FOOTNOTE "<A HREF='DAILY_A_AP.PDF'><STRONG>PDF -PRINTABLE- VERSION!</STRONG></A>";
KEYLABEL SUM ='TOTAL'; KEYLABEL N   ='# OF ACCTS'; KEYLABEL ALL ='TOTAL';
CLASS PYPSTDU /STYLE=[FONT_SIZE=2 PROTECTSPECIALCHARS=OFF];
FORMAT PYPSTDU DELKYIIA.;
VAR  BALANCE /STYLE=[FONT_SIZE=1 ];
LABEL BALANCE = 'ACCOUNT BALANCE'; LABEL PYPSTDU = 'PAST DUE RANGE';
TABLE (PYPSTDU ALL), N*(F=COMMA8.0) BALANCE *(SUM*F=DOLLAR15.2) /RTS=75 INDENT=3; KEYWORD ALL /
STYLE=[FONT_SIZE=2 FOREGROUND=BLACK];
RUN;
```

Ok the TABULATE procedure. Our TITLE statements need no explanation.

The FOOTNOTE does. We are using HREF or HTML to make our FOOTNOTE stand out. The user will be directed to the PDF version of this output that we will create in a few minutes via a "click-able" link on the same output we are generating. Our user will now have a quick and easy way to obtain the same results in our HTML tabulate but via PDF for printing.

Now take note of the CLASS statement for the variable PYPSTDU.

We are adding the syntax "PROTECTSPECIALCHARS= OFF". This will enable the use of HTML in the FORMAT we will invoke. It will allow the HTML syntax to be moved without being PROTECTED by the quotes which otherwise would simply show HTML code not the HTML results.

The FORMAT statement for PYPSTDU is referencing the code we set up earlier. Remember this FORMAT has HTML in it. Not only that but the HTML references FILES. That means that each category the FORMAT will create will be a click-able link. We are letting SAS generate a pseudo-drill-down report or web page. By using PROC FORMAT and HTML in this fashion you have just enabled the reporting capabilities of your existing reports to be further enhanced. Now you can connect the TABULATE or PRINT or other ODS output with FORMAT to something like an EXCEL sheet with the detail behind these totals. The next few TABULATES are the same but let's remember they are set up for different Data Sets and output.

```
PROC TABULATE DATA= WORK.ACCT2 MISSING;
TITLE1 "1016INC.COM SAS EXAMPLE";
TITLE2 "OUTPUT ON PORTFOLIO - B -";
TITLE3 "PRODUCT TYPE REPORT";
FOOTNOTE "<A HREF='DAILY_A_BP.PDF'><STRONG>PDF -PRINTABLE- VERSION!</STRONG></A>";
KEYLABEL SUM ='TOTAL'; KEYLABEL N   ='# OF ACCTS'; KEYLABEL ALL ='TOTAL';
CLASS PYPSTDU /STYLE=[FONT_SIZE=2 PROTECTSPECIALCHARS=OFF];
FORMAT PYPSTDU DELKYIIB.;
VAR  BALANCE /STYLE=[FONT_SIZE=1 ];
LABEL BALANCE = 'ACCOUNT BALANCE'; LABEL PYPSTDU = 'PAST DUE RANGE';
TABLE (PYPSTDU ALL), N*(F=COMMA8.0) BALANCE *(SUM*F=DOLLAR15.2) /RTS=75 INDENT=3; KEYWORD ALL /
STYLE=[FONT_SIZE=2 FOREGROUND=BLACK];
RUN;
```

```
PROC TABULATE DATA= WORK.ACCT3 MISSING;
TITLE1 "1016INC.COM SAS EXAMPLE";
TITLE2 "OUTPUT ON PORTFOLIO - C -";
TITLE3 "PRODUCT TYPE REPORT";
FOOTNOTE "<A HREF='DAILY_A_CP.PDF'><STRONG>PDF -PRINTABLE- VERSION!</STRONG></A>";
KEYLABEL SUM ='TOTAL'; KEYLABEL N   ='# OF ACCTS'; KEYLABEL ALL ='TOTAL';
CLASS PYPSTDU /STYLE=[FONT_SIZE=2 PROTECTSPECIALCHARS=OFF];
FORMAT PYPSTDU DELKYIIC.;
VAR  BALANCE /STYLE=[FONT_SIZE=1 ];
LABEL BALANCE = 'ACCOUNT BALANCE';
LABEL PYPSTDU = 'PAST DUE RANGE';
TABLE (PYPSTDU ALL), N*(F=COMMA8.0) BALANCE *(SUM*F=DOLLAR15.2) /RTS=75 INDENT=3; KEYWORD ALL /
STYLE=[FONT_SIZE=2 FOREGROUND=BLACK];
RUN;
ODS HTML CLOSE;
ODS LISTING;
```

The output for this part of the program should result in 3 html files. Which have links in what is the FORMAT categories. These links should point to the EXCEL and from the FOOTNOTES the PDF files we will create next
Examples of this output are on page 8 of this presentation.

Lets create the PDF files and then the EXCEL.

In this ODS PDF example we generate only one TABULATE. One for each of the portfolio segments we are reviewing. Notice how the FOOTNOTE statement is intentionally blank. We want to remove the values that would have had a residual HTML syntax from our last procedure. The CLASS statement for PYPMPSTDU has no syntax for the special character settings. And the FORMAT statement is for our usual FORMAT statement that has no HTML. Remember this is PDF. Our user(s) will not have HTML on that format. The TABULATE is essentially the same. We'll see 3 individual ODS PDF blocks for each of our examples.

```
ODS LISTING CLOSE;
ODS PDF FILE='DAILY_A_AP.PDF' STYLE=SASWEB NOTOC;
OPTIONS MISSING=0 NOCENTER PAGENO=1;

PROC TABULATE DATA= WORK.ACCT1 MISSING;
  TITLE1 "1016INC.COM SAS EXAMPLE";
  TITLE2 "OUTPUT ON PORTFOLIO - A -";
  TITLE3 "PRODUCT TYPE REPORT"; FOOTNOTE " ";
  KEYLABEL SUM ='TOTAL'; KEYLABEL N ='# OF ACCTS'; KEYLABEL ALL ='TOTAL' ;
  CLASS PYPMPSTDU /STYLE=[FONT_SIZE=2 ];
  FORMAT PYPMPSTDU DELKYXI.;
  VAR BALANCE /STYLE=[FONT_SIZE=1 ];
  LABEL BALANCE = 'ACCOUNT BALANCE'; LABEL PYPMPSTDU = 'PAST DUE RANGE';
  TABLE (PYPMPSTDU ALL), N*(F=COMMA8.0) BALANCE *(SUM*F=DOLLAR15.2) /RTS=75 INDENT=3; KEYWORD ALL /
    STYLE=[FONT_SIZE=2 FOREGROUND=BLACK];
RUN;
ODS PDF CLOSE;
ODS LISTING;
```

```
ODS LISTING CLOSE;
ODS PDF FILE='DAILY_A_BP.PDF' STYLE=SASWEB NOTOC;
OPTIONS MISSING=0 NOCENTER PAGENO=1;

PROC TABULATE DATA= WORK.ACCT1 MISSING;
  TITLE1 "1016INC.COM SAS EXAMPLE";
  TITLE2 "OUTPUT ON PORTFOLIO - B -";
  TITLE3 "PRODUCT TYPE REPORT"; FOOTNOTE " ";
  KEYLABEL SUM ='TOTAL'; KEYLABEL N ='# OF ACCTS'; KEYLABEL ALL ='TOTAL' ;
  CLASS PYPMPSTDU /STYLE=[FONT_SIZE=2 ];
  FORMAT PYPMPSTDU DELKYXI.;
  VAR BALANCE /STYLE=[FONT_SIZE=1 ];
  LABEL BALANCE = 'ACCOUNT BALANCE'; LABEL PYPMPSTDU = 'PAST DUE RANGE';
  TABLE (PYPMPSTDU ALL), N*(F=COMMA8.0) BALANCE *(SUM*F=DOLLAR15.2) /RTS=75 INDENT=3; KEYWORD ALL /
    STYLE=[FONT_SIZE=2 FOREGROUND=BLACK];
RUN;
ODS PDF CLOSE;
ODS LISTING;
```

```
ODS LISTING CLOSE;
ODS PDF FILE='DAILY_A_CP.PDF' STYLE=SASWEB NOTOC;
OPTIONS MISSING=0 NOCENTER PAGENO=1;

PROC TABULATE DATA= WORK.ACCT1 MISSING;
  TITLE1 "1016INC.COM SAS EXAMPLE";
  TITLE2 "OUTPUT ON PORTFOLIO - C -";
  TITLE3 "PRODUCT TYPE REPORT"; FOOTNOTE " ";
  KEYLABEL SUM ='TOTAL'; KEYLABEL N ='# OF ACCTS'; KEYLABEL ALL ='TOTAL' ;
  CLASS PYPMPSTDU /STYLE=[FONT_SIZE=2 ];
  FORMAT PYPMPSTDU DELKYXI.;
  VAR BALANCE /STYLE=[FONT_SIZE=1 ];
  LABEL BALANCE = 'ACCOUNT BALANCE'; LABEL PYPMPSTDU = 'PAST DUE RANGE';
  TABLE (PYPMPSTDU ALL), N*(F=COMMA8.0) BALANCE *(SUM*F=DOLLAR15.2) /RTS=75 INDENT=3; KEYWORD ALL /
    STYLE=[FONT_SIZE=2 FOREGROUND=BLACK];
RUN;
ODS PDF CLOSE;
ODS LISTING;
```

The example of this output is on page 11 of this presentation.

MS EXCEL FOR THE HTML

There are different approaches to getting SAS Data to EXCEL. I must stress that for this example I needed to have separate EXCEL workbooks for each of the categories. The users would only be interested in specific workbooks according to their department etc. This example also uses a specific SAS Dataset, which has the variables, labels, and formats properly set up for facilitation of output. Your results on your own data or site would need to take this into consideration when you begin to get different results in EXCEL after performing this same kind of export. The macro I used was adjusted to work for this code. You're welcome to use the macro I've mentioned but please configure it to work for your scenario.

Prepare for the 1st Excel output.

```
DATA REVW029A REVW030A REVW060A REVW090A REVW120A REVW150A REVW180A REVW210A;
SET WORK.ACCT1;
FORMAT PYMPSTDU DELKYXI.;
IF PYMPSTDU LT 2 THEN OUTPUT REVW029A;
IF PYMPSTDU EQ 2 THEN OUTPUT REVW030A;
IF PYMPSTDU EQ 3 THEN OUTPUT REVW060A;
IF PYMPSTDU EQ 4 THEN OUTPUT REVW090A;
IF PYMPSTDU EQ 5 THEN OUTPUT REVW120A;
IF PYMPSTDU EQ 6 THEN OUTPUT REVW150A;
IF PYMPSTDU EQ 7 THEN OUTPUT REVW180A;
IF PYMPSTDU GE 8 THEN OUTPUT REVW210A;
RUN;
```

Since my example calls for separate tabs for each of the portfolios according to the FORMAT Categories we need to output EXCEL workbooks for each and so we want to have separate SAS Datasets. We'll do the same for each portfolio. Notice the DATASET names.

```
DATA REVW029B REVW030B REVW060B REVW090B REVW120B REVW150B REVW180B REVW210B;
SET WORK.ACCT2;
FORMAT PYMPSTDU DELKYXI.;
IF PYMPSTDU LT 2 THEN OUTPUT REVW029B;
IF PYMPSTDU EQ 2 THEN OUTPUT REVW030B;
```

```
DATA REVW029C REVW030C REVW060C REVW090C REVW120C REVW150C REVW180C REVW210C;
SET WORK.ACCT3;
FORMAT PYMPSTDU DELKYXI.;
IF PYMPSTDU LT 2 THEN OUTPUT REVW029C;
IF PYMPSTDU EQ 2 THEN OUTPUT REVW030C;
```

Now we have distinct SAS Datasets for each of the portfolios. One additional item is that on each we have used the FORMAT that does not use HTML.

```
OPTIONS MPRINT MACROGEN NOSYMBOLGEN;
OPTIONS NOXWAIT NOXSYNC;

X "DEL C:\&FOLD\ALL_D*.&FT";
```

Now lets prepare for our new output. This macro uses the DDE method. You need to remember that without a NOXWAIT your session will sit there and wait for whatever response the prompts called by X or other routines will wait for. I use an 'X' command to clean out old excel files from the folder we are working in.

```
data setup; dsname1='work.REVW029A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a029.xls,yes); run;
data setup; dsname1='work.REVW030A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a030.xls,yes); run;
data setup; dsname1='work.REVW060A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a060.xls,yes); run;
data setup; dsname1='work.REVW090A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a090.xls,yes); run;
data setup; dsname1='work.REVW120A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a120.xls,yes); run;
data setup; dsname1='work.REVW150A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a150.xls,yes); run;
data setup; dsname1='work.REVW180A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a180.xls,yes); run;
data setup; dsname1='work.REVW210A'; dsname2=scan(dsname1,2); %exloop(c:\&fold\all_dq_a210.xls,yes); run;
```

This set of macro steps will export the first segment or portfolio portion of our data to EXCEL. Please note that the remainder of the code for this step in this specific part of the code is not presented in this abstract, as it is repetitive. However the working example does have the entire code.

An example of how this Excel output is viewed is on page 10 of this presentation.

CONCLUSION

Certainly the techniques I used are a variety of basic thru intermediate level concepts. The goal was to allow the information in the report to be 'drillable' to the users and employ some of the HTML ODS features PC SAS has to offer. Combined with some 3rd party software like EXCEL you end up getting a very flexible concept for producing functional reports that users will find very easy to navigate and work with.

There are more exciting features and uses just waiting for you to discover.

ADDITIONAL CODE OR DATA USED

C:\SUG\MACROEXPORTERNEW.SAS

macroexporternew.sas was obtained from:

<http://www.sas.com/service/techsup/unotes/SN/006/006695.html>

Modifications were made to suffice the desired results for this code. Changes or modifications may be required for your site, pc, station.

This code is available with sufficient documentation and as such this abstract will not cover it.

**end macroexporternew.sas;*

SAS DATASET WORK.ACCT (SASLIB.DAT)

This is a SAS dataset on the host that we pick up as a SAS Transport file. However this can be almost any file you can transfer via FTP to your PC and then read with PC SAS. Say a flat file. Our example works because on the host we have already set up the data for this transport in batch. You can also have a dataset or file available for pick up and still use this method.

FINAL CONSIDERATIONS

Please note that the code supplied in this paper is designed only to illustrate the concepts being discussed and may need to be modified to work in other applications and/or environments. The author of this paper does not support modified code.

FINAL COMMENTS

Please all efforts have been made to provide working examples of code for the benefit of those reading this presentation or viewing it. If you would like a functional example please contact Zeke Torres.

TRADEMARK CITATION

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

AUTHORS CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the author at:



Ezequiel 'Zeke' Torres
1016Inc.com
Chicago, Illinois 60630
Work Phone: (773)467-8492
Fax: (773)467-8492
Email: info@1016inc.com
Web: www.1016inc.com

WARNING

This code and the concept and/or your ability to execute it does not indicate you are complying with the security protocols or standards your site, office, place of work have established. The author makes no warranty, assurances implied or otherwise that this method is in anyway 'secure' or complies with your site standards or protocols. The code, information and concept is presented 'as is'. The use and/or manipulation of USER ID's, Passwords and similar confidential user information can be considered ILLEGAL and against policy or standards at your location and in some cases by public agencies, communications companies, telephone, internet service providers and/or law enforcement agencies. You are responsible for taking that into consideration before applying this model in any way.

Use the information, concept presented here at your own risk.

Example of Main ODS PAGE produced.

This is the main page we would get from the ODS Code used.

Note how the links are available to drill down and get to the source data.

Additionally note the PDF Link that is part of our FOOTNOTE in the code but fits nicely into the presentation of the page.

Table of Contents

1. The Tabulate Procedure
·Cross-tabular
summary report
·[Table 1](#)
2. The Tabulate Procedure
·Cross-tabular
summary report
·[Table 1](#)
3. The Tabulate Procedure
·Cross-tabular
summary report
·[Table 1](#)

1016INC.COM SAS EXAMPLE OUTPUT ON PORTFOLIO - A - PRODUCT TYPE REPORT

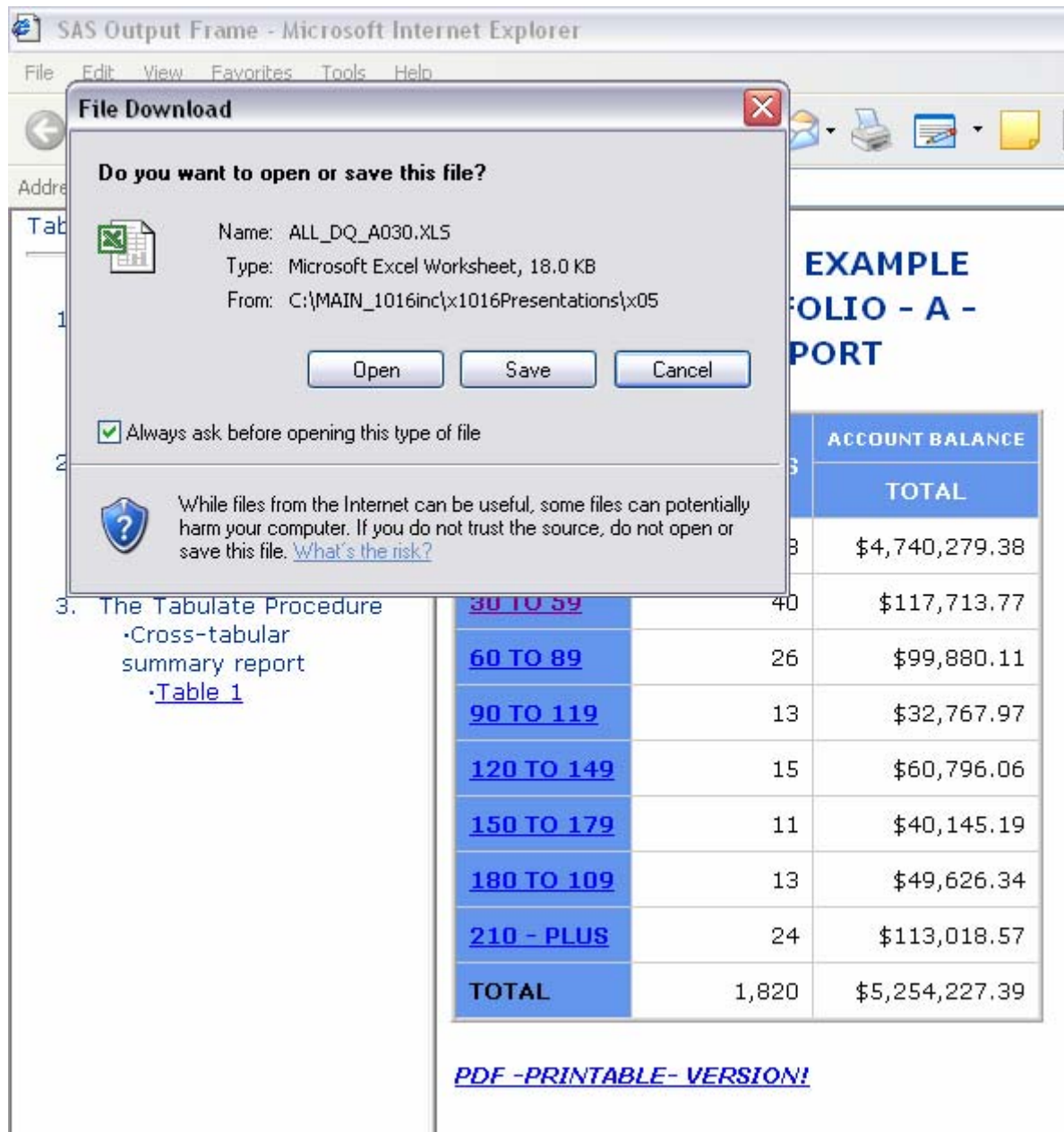
	# OF ACCTS	ACCOUNT BALANCE
		TOTAL
ZERO TO 29	1,678	\$4,740,279.38
30 TO 59	40	\$117,713.77
60 TO 89	26	\$99,880.11
90 TO 119	13	\$32,767.97
120 TO 149	15	\$60,796.06
150 TO 179	11	\$40,145.19
180 TO 109	13	\$49,626.34
210 - PLUS	24	\$113,018.57
TOTAL	1,820	\$5,254,227.39

[PDF -PRINTABLE- VERSION!](#)

Example of Drilling into the Data.

The image below illustrates how the Excel application is called up when the links are tied to MS Excel. This is very beneficial to share data with end users and "NON" SAS users.

Also note that we could have used MS Access or some other application. Which is why I chose to make the extension a "%Let" variable in order to switch the application type easier within the code use.



The screenshot shows a web browser window titled "SAS Output Frame - Microsoft Internet Explorer". A "File Download" dialog box is open, asking "Do you want to open or save this file?". The file details are: Name: ALL_DQ_A030.XLS, Type: Microsoft Excel Worksheet, 18.0 KB, From: C:\MAIN_1016inc\%x1016Presentations\%x05. The dialog has "Open", "Save", and "Cancel" buttons. Below the buttons is a checkbox labeled "Always ask before opening this type of file" which is checked. At the bottom of the dialog is a warning icon and text: "While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not open or save this file. [What's the risk?](#)".

In the background, a table titled "EXAMPLE PORTFOLIO - A - REPORT" is visible. The table has two columns: "ACCOUNT BALANCE" and "TOTAL". The rows are as follows:

ACCOUNT BALANCE	TOTAL
30 TO 59	40
60 TO 89	26
90 TO 119	13
120 TO 149	15
150 TO 179	11
180 TO 109	13
210 - PLUS	24
TOTAL	1,820

Below the table, there is a link: [PDF -PRINTABLE- VERSION!](#)

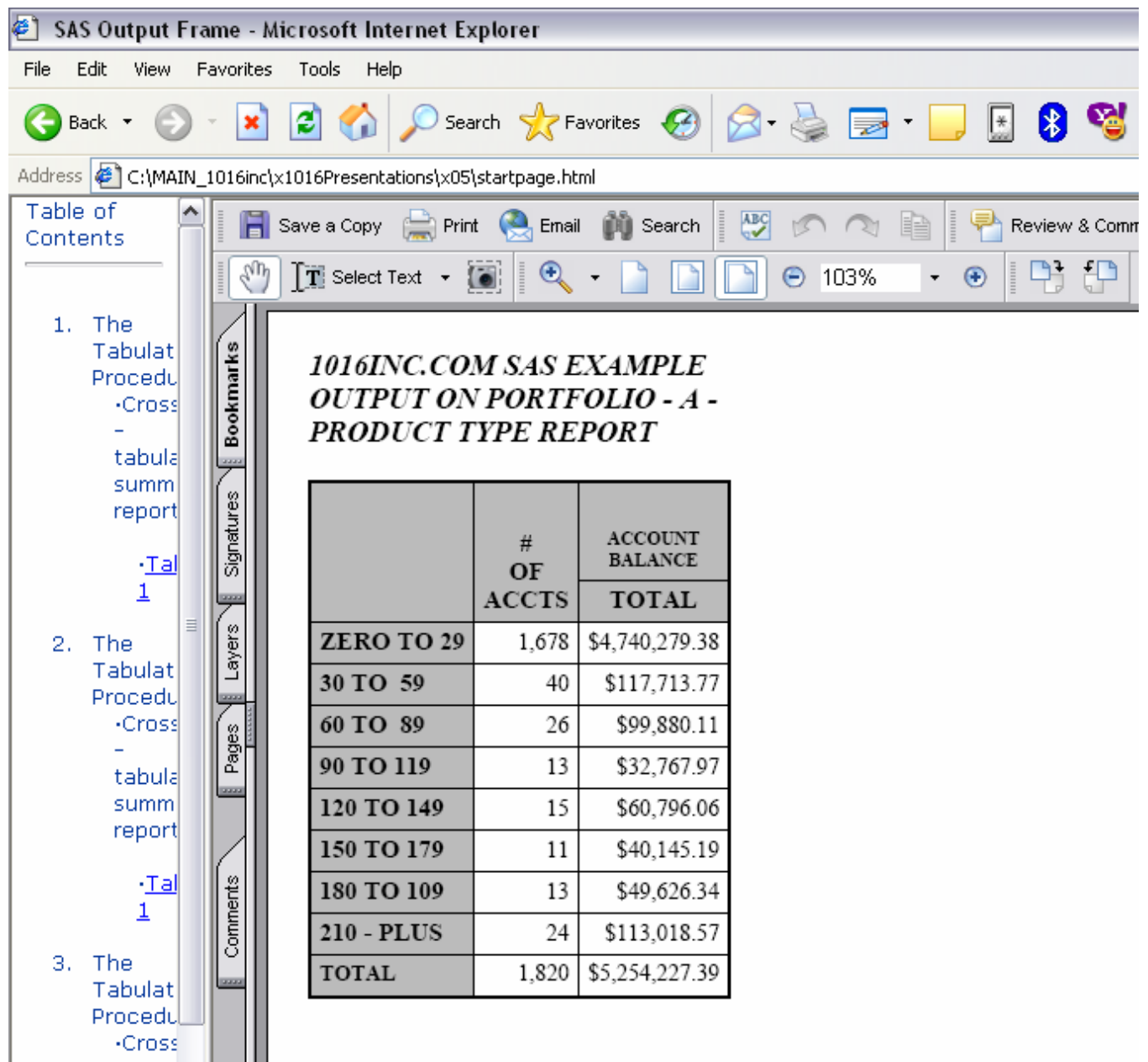
EXAMPLE OF EXCEL WITHIN THE PAGE.

From this view the End User can save the excel page to their local drive or folder. More importantly they can investigate the data without the explicit need of SAS on their PC. It becomes very beneficial and productive when sharing data with those who haven't the benefit of the SAS System on their PC.

SAS Output Frame - Microsoft Internet Explorer							
File Edit View Favorites Tools Help							
Back Search Favorites							
Address C:\MAIN_1016\inc\1016Presentations\1016\startpage.html							
Table of Contents							
1. The Tabulate Procedure • Cross-tabular summary report • Table 1							
2. The Tabulate Procedure • Cross-tabular summary report • Table 1							
3. The Tabulate Procedure • Cross-tabular summary report • Table 1							
A1 STATE							
	A	B	C	D	E	F	G
1	STATE	BALANCE	LAST_PA	LOAN_TE	INTERES	CREDIT_S	PROC
2	TX	3820.5	6/4/2002	84	18	597	202
3	NY	2826.29	7/13/2002	48	17.75	630	202
4	OR	3147.8	6/3/2002	91	10	625	202
5	TX	2014.3	7/8/2002	48	18	643	201
6	OR	3097.25	6/4/2002	84	19.9	639	202
7	IN	6913.71	7/15/2002	84	19.9	644	202
8	MI	3742.6	6/10/2002	84	13.5	669	202
9	FL	1.16	7/3/2002	48	19.1	623	202
10	CO	2892.01	5/10/2002	48	19.9	623	202
11	NV	5665	6/9/2002	48	19.9	655	202
12	AZ	3630.31	5/2/2002	48	19.9	623	202
13	WI	1274.09	6/20/2002	84	19.9	625	202
14	TX	1938.49	5/24/2002	48	18	648	202
15	NV	3979.83	5/23/2002	48	19.9	636	202
16	TX	3517.96	6/27/2002	60	18	642	201
17	NE	1303.73	5/20/2002	48	18	645	202
18	MI	92.84	5/4/2002	48	14.25	759	201
19	TX	3904.05	5/16/2002	48	18	647	202
20	AZ	3271.29	6/13/2002	48	19.9	636	201
21	GA	42.8	5/8/2002	48	19.9	736	202
22	NV	3402.35	8/3/2002	48	19.9	658	201
23	TX	3660.25	5/31/2002	48	18	650	202
24	KS	4549.73	7/11/2002	48	19.9	742	202
25	TX	3266.69	5/13/2002	48	18	622	201
26	AZ	4370.8	6/17/2002	48	19.9	655	202
27	FL	2544.43	4/26/2002	48	19.1	659	201
28	NV	4250.44	5/17/2002	48	19.9	670	202
29	OK	2065.84	4/17/2002	48	19.9	622	202

EXAMPLE OF PDF OUTPUT

This is how the PDF output would look from the application or code we have used.



**1016INC.COM SAS EXAMPLE
OUTPUT ON PORTFOLIO - A -
PRODUCT TYPE REPORT**

	# OF ACCTS	ACCOUNT BALANCE TOTAL
ZERO TO 29	1,678	\$4,740,279.38
30 TO 59	40	\$117,713.77
60 TO 89	26	\$99,880.11
90 TO 119	13	\$32,767.97
120 TO 149	15	\$60,796.06
150 TO 179	11	\$40,145.19
180 TO 109	13	\$49,626.34
210 - PLUS	24	\$113,018.57
TOTAL	1,820	\$5,254,227.39