Paper 10080-2016

Using the REPORT Procedure to Export a Big Data set to an External XML

Guihong Chen, TCF Bank

ABSTRACT

A number of SAS tools can be used to report data such as the PRINT, MEANS, TABULATE, and REPORT procedures. The REPORT procedure is a single tool that can cover features of the other SAS tools. Not only can it produce detail reports like PROC PRINT, but it can summarize and calculate data as in the MEANS and TABULATE procedures. Unfortunately, despite its power, PROC REPORT seems to be used less often than the other tools, possibly due to its seemingly complex coding.

This paper uses PROC REPORT under the ODS environment to export a big data set into a colorful and customized xml file so that a non-SAS user can read the data easily in the xml file for useful information. DEFINE, COMPUTE, and COLUMN statements are used in the paper along with several options to present your data in a more colorful way. It shows how to control the format of the selected columns and rows. The headings of columns can be more meaningful and pretty. It also talks about how to color any selected cells differently to bring attention to the audience.

INTRODUCTION

When asked to send the detail data to business or management for review and investigation, you often need to convert SAS data to other report formats if they do not have the access to SAS. If they have Office 2003 or higher, they should be able to view the data set via Excel when you output the SAS data into Excel files under ODS environment. If the data is sufficiently big, in the sense of number of rows or number of columns, we may want to enhance the Excel report to address some interesting items by changing the column and row formats or adding some summary information. PROC REPORT provides a handful of options and statements users can leverage to control the appearance of the output of the data extensively. Things you can exert control over include, but are not limited to, report type, variable usage, formats, labels, styles and keyword statistics. If you go to the SAS website http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a000146852.htm , you can easily find the syntax of REPORT Procedure as below.

PROC REPORT *<option(s)>*;

  BREAK *location break-variable*</ *option(s)*>;

  BY< DESCENDING> *variable-1*
  < ...<DESCENDING> *variable-n*><NOTSORTED>;

  COLUMN *column-specification(s)*;

    COMPUTE *location <target>*
    < / STYLE=<*style-element-name*>
    < [*style-attribute-specification(s)*]>>;

      LINE *specification(s)*;

      . . . *select SAS language elements . . .*

      ENDCOMP;

    COMPUTE *report-item* </ *type-specification*>;

CALL DEFINE (*column-id*, '*attribute-name*', *value*);

. . . select SAS language elements . . .

ENDCOMP;

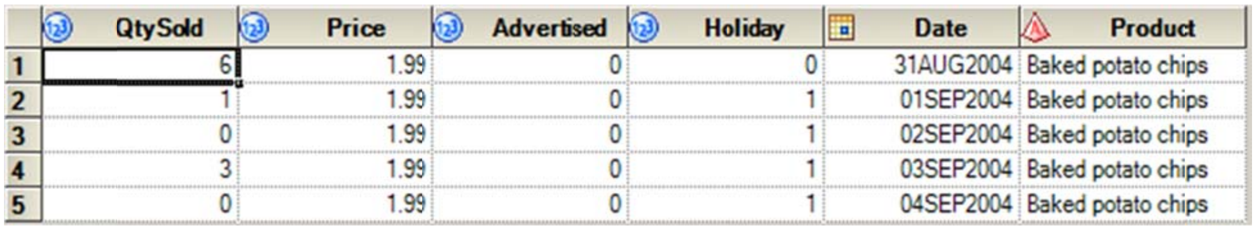
DEFINE *report-item* /< *usage*>
< *attribute(s)*>
< *option(s)*>
< *justification*>
< COLOR=*color*>
< '*column-header-1*' <...'*column-header-n*'>>
< *style*>;

FREQ *variable*;

RBREAK *location*< / *option(s)*>;

WEIGHT *variable*;

However, the complexity of the coding may deter new users from using it without seeing the benefits of it in real examples. This paper uses a data set to show the benefits of using PROC REPORT which can't be achieved by other report procedures like PRINT, MEANS, and TABULATE. The data set used for the whole paper is called "Sashelp.snacks" provided by SAS in the Sashelp library. This data set provides daily snack food sales. A snapshot of the data for the first five observations is below.

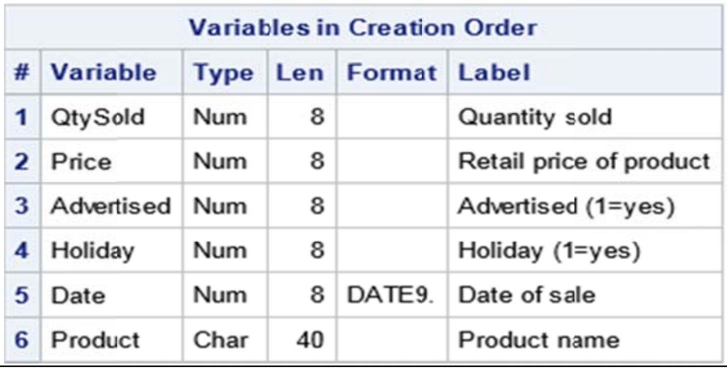


	QtySold	Price	Advertised	Holiday	Date	Product
1	6	1.99	0	0	31AUG2004	Baked potato chips
2	1	1.99	0	1	01SEP2004	Baked potato chips
3	0	1.99	0	1	02SEP2004	Baked potato chips
4	3	1.99	0	1	03SEP2004	Baked potato chips
5	0	1.99	0	1	04SEP2004	Baked potato chips

Display 1 : Data Snapshot for the First Five Observations

The data set contains 35,770 observations with 6 variables as shown below using PROC CONTENTS.

```
proc contents data= sashelp.SNACKS order=VARNUM;run;
```



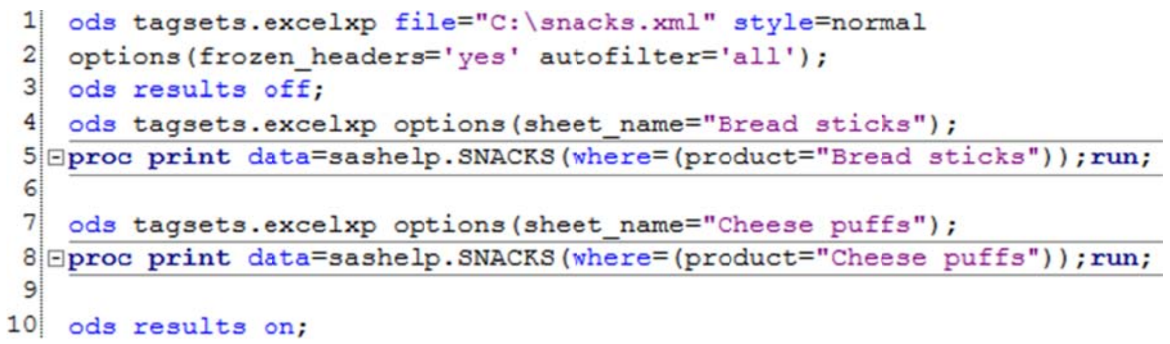
Variables in Creation Order					
#	Variable	Type	Len	Format	Label
1	QtySold	Num	8		Quantity sold
2	Price	Num	8		Retail price of product
3	Advertised	Num	8		Advertised (1=yes)
4	Holiday	Num	8		Holiday (1=yes)
5	Date	Num	8	DATE9.	Date of sale
6	Product	Char	40		Product name

Output 1 : Output from PROC CONTENTS Procedures

PROC REPORT can create both detail and summary reports. This paper will introduce and focus on how to create customized detail reports. Summary reports are not covered in this paper due to size and time limitation.

EXPORTING DETAIL REPORT TO EXCEL

Not everybody in your company may have access to the SAS data set. Management and Business people sometimes need to do some investigation of the subset of the data set. However if they have Office 2003 or higher, they should be able to view the data set via Excel when you output the SAS data into Excel files under ODS environment. The ODS TAGSETS.EXCELXP destination produces a Spreadsheet Markup Language file that can be opened with Office 2003 or higher. For example, when asked to report all the observations for the products “Bread sticks” and “Cheese puffs”, you can create an Excel file with two different sheets as below:



```
 1  ods tagsets.excelxp file="C:\snacks.xml" style=normal
 2  options(frozen_headers='yes' autofilter='all');
 3  ods results off;
 4  ods tagsets.excelxp options(sheet_name="Bread sticks");
 5  proc print data=sashelp.SNACKS(where=(product="Bread sticks"));run;
 6
 7  ods tagsets.excelxp options(sheet_name="Cheese puffs");
 8  proc print data=sashelp.SNACKS(where=(product="Cheese puffs"));run;
 9
10  ods results on;
```

Display 2 : Codes Snapshot to export reports to excel

In Line 1, the STYLE option specifies the “normal” style template to use. SAS provides different style templates to display the presentation aspects of the output. To choose your favorite one, submit the following code to review all the possible names of the built-in style templates:

```
proc template;list styles/store=sashelp.tmplmst;run;
```

In Line 2, two options are added to control the view of the data especially for a big data set that can’t fit into one page of sheet. FROZEN_HEADERS=’yes’ is to freeze the header row when the table data scrolls. The default is ‘no’. AUTOFILTER=’all’ is to add auto filters for all columns. The default is ‘none’.

In Line 3, ODS RESLTS OFF is to turn off the output of data display in the SAS output window. The SAS® Output windows can display approximately 99,999 lines while an excel sheet can display 1,048,576 rows. If you are about to create a detail report with at least 100,000 observations, this could save your running time significantly and avoid the popup window default while SAS sits and waits for user approval.

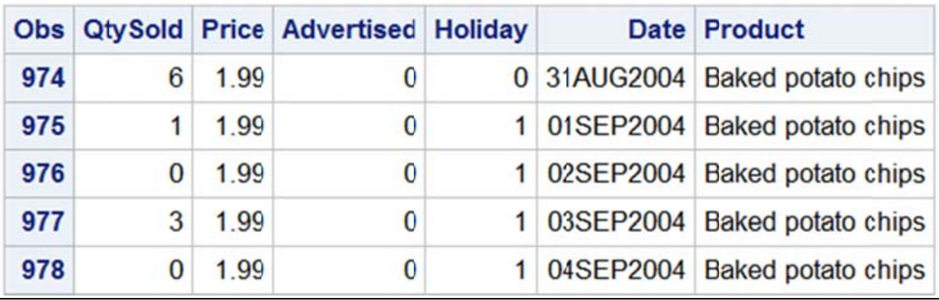
In Line 10, ODS RESULTS ON is to turn on the SAS output window.

In Line 4 and Line 7, the SHEET_NAME option specifies which sheet in the excel file to store the output.

DETAIL REPORT WITH CUSTOMIZED COLUMN AND ROW FORMATS

When asked to output a detail report, the first and easiest procedure we can think of is PROC PRINT. The code and output are shown as below:

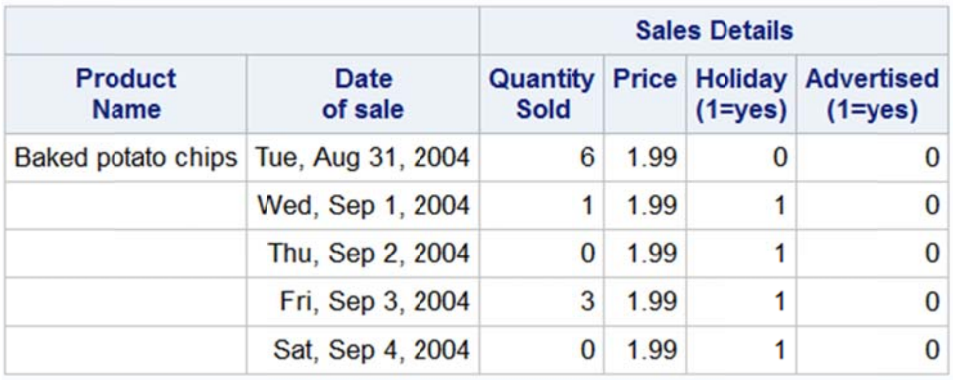
```
proc print data=sashelp.SNACKS(where=(date>"30Aug2004"d) obs=5);run;
```



Obs	QtySold	Price	Advertised	Holiday	Date	Product
974	6	1.99	0	0	31AUG2004	Baked potato chips
975	1	1.99	0	1	01SEP2004	Baked potato chips
976	0	1.99	0	1	02SEP2004	Baked potato chips
977	3	1.99	0	1	03SEP2004	Baked potato chips
978	0	1.99	0	1	04SEP2004	Baked potato chips

Output 2 : Output of PROC PRINT statements

However sometimes we may want to enhance the report to change the column formats for the convenience of management. For example, your manager would like to see the desired report as shown below.



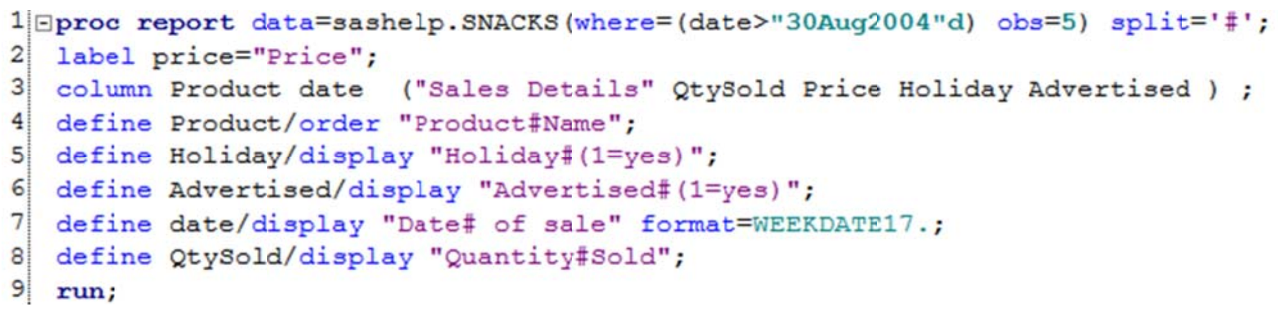
		Sales Details			
Product Name	Date of sale	Quantity Sold	Price	Holiday (1=yes)	Advertised (1=yes)
Baked potato chips	Tue, Aug 31, 2004	6	1.99	0	0
	Wed, Sep 1, 2004	1	1.99	1	0
	Thu, Sep 2, 2004	0	1.99	1	0
	Fri, Sep 3, 2004	3	1.99	1	0
	Sat, Sep 4, 2004	0	1.99	1	0

Output 3 : Output of PROC REPORT statements with customized column formats

There are several changes you need to make to the output.

1. Changing the order of the columns
2. Suppressing repetitious printing of the product name
3. Changing the format of the sale date
4. Specifying a text string as a header to span multiple columns
5. Change the column labels
6. Split the column headers

PROC REPORT can build reports from data set variables and make all the above changes in one procedure as below:



```
1 proc report data=sashelp.SNACKS(where=(date>"30Aug2004"d) obs=5) split='#';
2 label price="Price";
3 column Product date  ("Sales Details" QtySold Price Holiday Advertised ) ;
4 define Product/order "Product#Name";
5 define Holiday/display "Holiday#(1=yes)";
6 define Advertised/display "Advertised#(1=yes)";
7 define date/display "Date# of sale" format=WEEKDATE17.;
8 define QtySold/display "Quantity#Sold";
9 run;
```

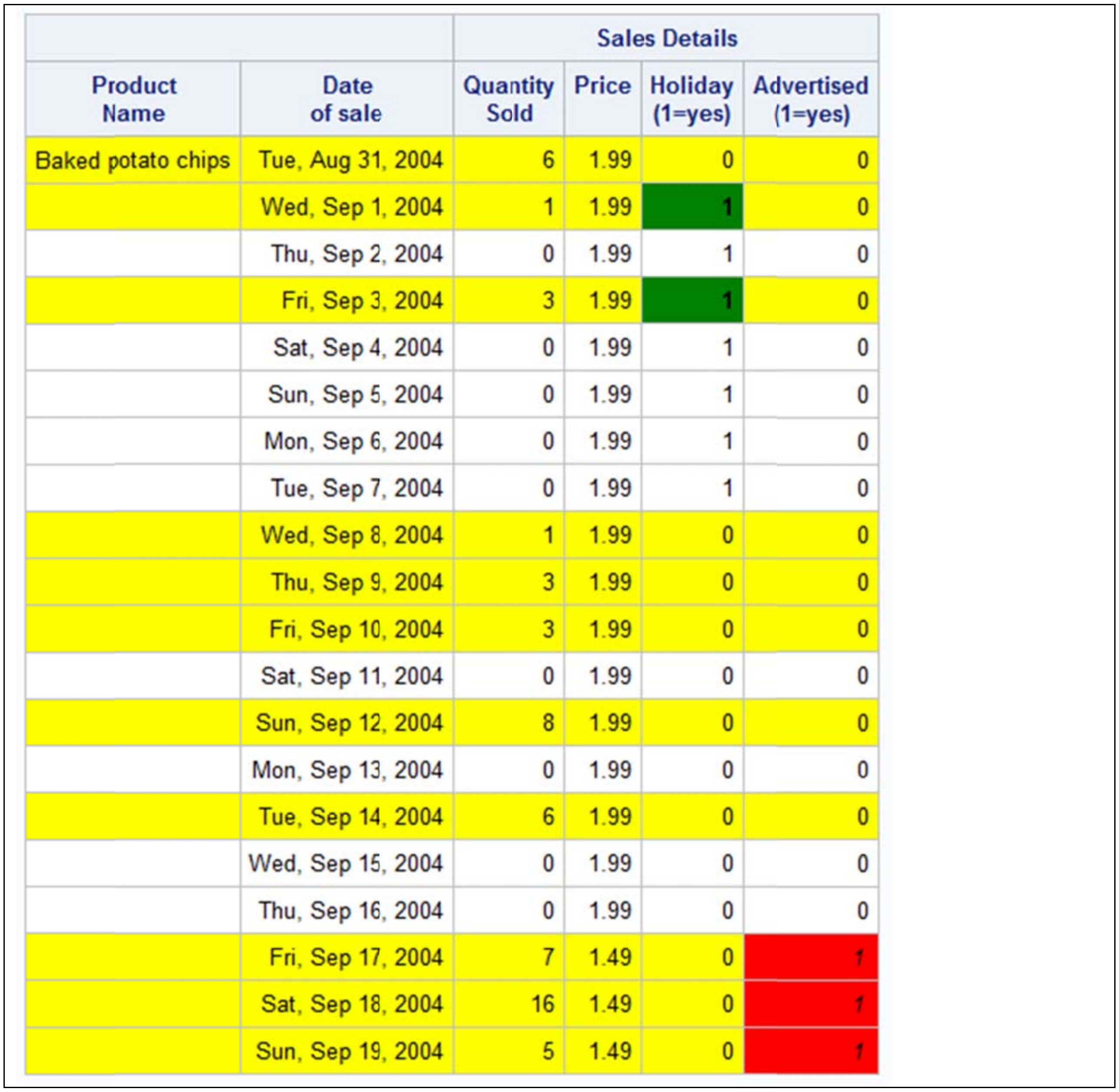
Display 3 : Code Snapshot to use PROC REPORT to change the column formats

1. Changing the order of the columns can be done by putting the desired order of variables in the COLUMN statement (Line 3).
2. Suppress repetitious printing of the product name can be done by adding an ORDER option in the DEFINE statement in Line 4.
3. Changing the format of the sale date can be done by adding FORMAT option in the DEFINE statement in Line 7 or by adding a FORMAT statement.
4. Specify a text string as a header to span multiple columns can be done by putting the spanned variable names in parentheses in the COLUMN statement preceded by a quoted text string used as a header as you can see in Line 3.
5. Changing the column labels can be done by adding a LABEL statements in Line 2 for "price" or adding an option in a quoted string in the DEFINE statement for the other variables, which would override a label for the same variable specified in the LABEL statement.
6. Splitting the column headers to multiple lines can be done with the SPLIT option in the REPORT statement (Line 1) and also use '#' in the label string specified in the DEFINE statement to separate words.

PROC PRINT can achieve most of the format changes above except for the fourth one: spanning headers. An example of code is shown below:

```
proc print data=sashelp.SNACKS(obs=5) split='#'
style(header)={TEXTALIGN=center};
var date  QtySold Price Holiday Advertised;
label product="Product#Name" QtySold="Quantity#Sold" price="Price"
date="Date# of sale";
format date  WEEKDATE17.;
id product;by product ;
run;
```

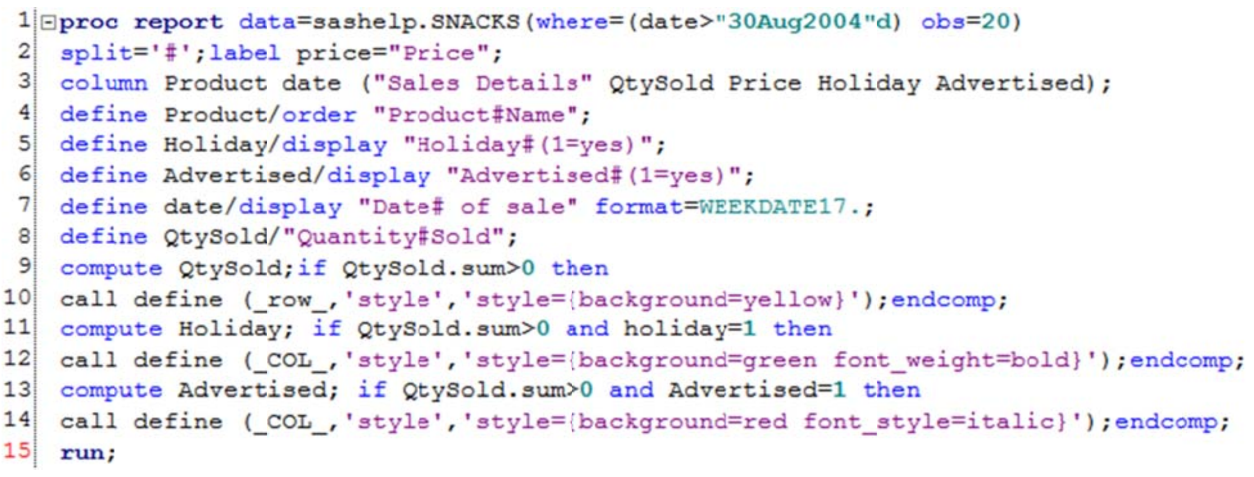
Although PROC PRINT gives sufficient tools to control column formats, it may not be powerful to change the row formats to the output below, which has traffic lighting for positive sales.



		Sales Details			
Product Name	Date of sale	Quantity Sold	Price	Holiday (1=yes)	Advertised (1=yes)
Baked potato chips	Tue, Aug 31, 2004	6	1.99	0	0
	Wed, Sep 1, 2004	1	1.99	1	0
	Thu, Sep 2, 2004	0	1.99	1	0
	Fri, Sep 3, 2004	3	1.99	1	0
	Sat, Sep 4, 2004	0	1.99	1	0
	Sun, Sep 5, 2004	0	1.99	1	0
	Mon, Sep 6, 2004	0	1.99	1	0
	Tue, Sep 7, 2004	0	1.99	1	0
	Wed, Sep 8, 2004	1	1.99	0	0
	Thu, Sep 9, 2004	3	1.99	0	0
	Fri, Sep 10, 2004	3	1.99	0	0
	Sat, Sep 11, 2004	0	1.99	0	0
	Sun, Sep 12, 2004	8	1.99	0	0
	Mon, Sep 13, 2004	0	1.99	0	0
	Tue, Sep 14, 2004	6	1.99	0	0
	Wed, Sep 15, 2004	0	1.99	0	0
	Thu, Sep 16, 2004	0	1.99	0	0
	Fri, Sep 17, 2004	7	1.49	0	1
	Sat, Sep 18, 2004	16	1.49	0	1
	Sun, Sep 19, 2004	5	1.49	0	1

Output 4 : Output of PROC REPORT statements with customized row formats

Under PROC REPORT, You can use a CALL DEFINE statement with conditional logic to customize a particular row in a COMPUTE block as below:



```
 1 proc report data=sashelp.SNACKS(where=(date>"30Aug2004"d) obs=20)
 2  split='#';label price="Price";
 3  column Product date ("Sales Details" QtySold Price Holiday Advertised);
 4  define Product/order "Product#Name";
 5  define Holiday/display "Holiday#(1=yes)";
 6  define Advertised/display "Advertised#(1=yes)";
 7  define date/display "Date# of sale" format=WEEKDATE17.;
 8  define QtySold/"Quantity#Sold";
 9  compute QtySold;if QtySold.sum>0 then
10  call define (_row_,'style','style={background=yellow}');endcomp;
11  compute Holiday; if QtySold.sum>0 and holiday=1 then
12  call define (_COL_,'style','style={background=green font_weight=bold}');endcomp;
13  compute Advertised; if QtySold.sum>0 and Advertised=1 then
14  call define (_COL_,'style','style={background=red font_style=italic}');endcomp;
15  run;
```

Display 4 : Code Snapshot to use PROC REPORT to change the row formats

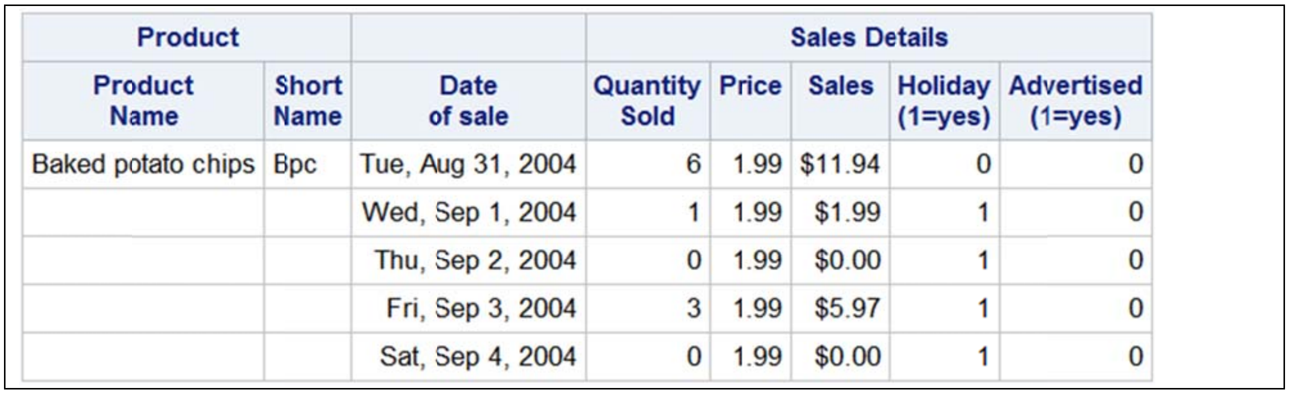
Note that inside the COMPUTE block, any variable with an ANALYSIS usage must be used with a compound name with the form *variable-name.statistic-name* to tell SAS how to use the variable in the COMPUTE blocks. The usage of variable can be specified after the slash in the DEFINE statement. Otherwise, the default usage for a numerical variable is ANALYSIS and for a character variable is DISPLAY. The variables “Holiday” and “Advertised” have been redefined with the DISPLAY usage in Line 5 and Line 6. Hence, compound names are not needed as there is no ambiguity about the usage in the COMPUTE block. However “QtySold” is an ANALYSIS variable with the default statistics SUM so we must use a compound name “QtySold.sum” to be referred inside any COMPUTE block.

The CALL DEFINE statement has three arguments which are not case sensitive.

1) The first argument specifies the area affected. In Line 10, we want to change the formats for all the variables in the same row so we use “_ROW_” for the first argument. In Line 12 and 14, you can also use “_COL_” here if you only want to change the format of the variable for which the COMPUTE block is attached.
2) The second argument is usually quoted and can be ‘FORMAT’, ‘STYLE’ or ‘URL’ which specifies whether to change the format, style or to add a link to the affected area.
3) The third argument may need quotes and provide details for the second argument.

DETAIL REPORT WITH NEW VARIABLES

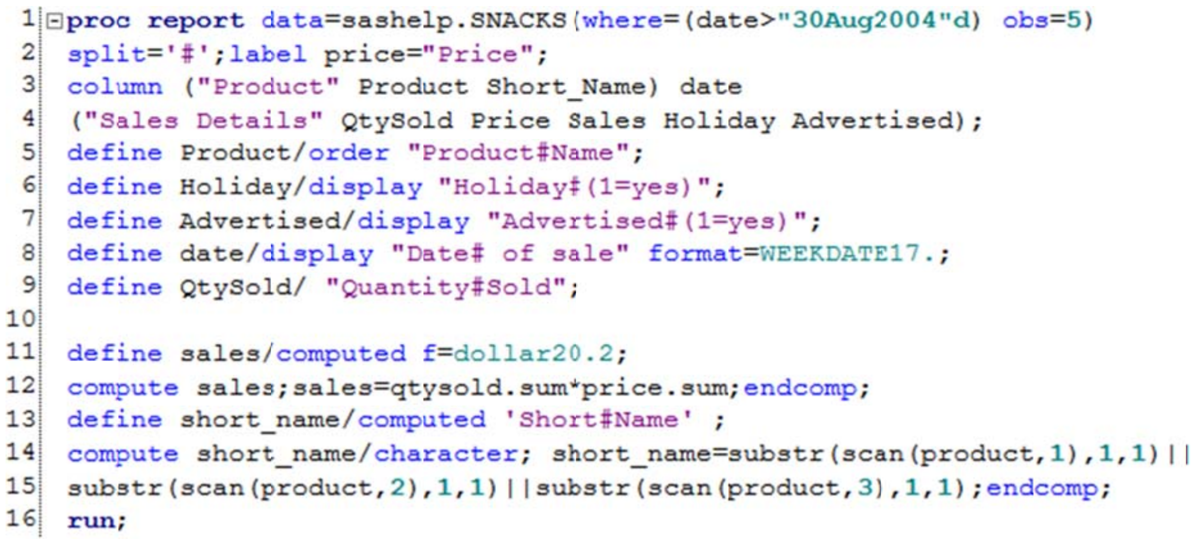
Sometimes you want to add new variables to the report. In Output 5, we add two new variables: “Sales” and “Short_Name” to the report. “Sales” is a numerical variable while “Short_Name” is a character variable.



Product			Sales Details				
Product Name	Short Name	Date of sale	Quantity Sold	Price	Sales	Holiday (1=yes)	Advertised (1=yes)
Baked potato chips	Bpc	Tue, Aug 31, 2004	6	1.99	$11.94	0	0
		Wed, Sep 1, 2004	1	1.99	$1.99	1	0
		Thu, Sep 2, 2004	0	1.99	$0.00	1	0
		Fri, Sep 3, 2004	3	1.99	$5.97	1	0
		Sat, Sep 4, 2004	0	1.99	$0.00	1	0

Output 5 : Output of PROC REPORT statements with new variables added

You cannot accomplish this job with PROC PRINT without changing the data set. However, you can easily add both numeric and character variables in PROC REPORT without changing the data by using COMPUTE blocks as in Line 11 to Line 15 below:



```
 1 proc report data=sashelp.SNACKS(where=(date>"30Aug2004"d) obs=5)
 2   split='#';label price="Price";
 3   column ("Product" Product Short_Name) date
 4   ("Sales Details" QtySold Price Sales Holiday Advertised);
 5   define Product/order "Product#Name";
 6   define Holiday/display "Holiday#(1=yes)";
 7   define Advertised/display "Advertised#(1=yes)";
 8   define date/display "Date# of sale" format=WEEKDATE17.;
 9   define QtySold/ "Quantity#Sold";
10
11   define sales/computed f=dollar20.2;
12   compute sales;sales=qtysold.sum*price.sum;endcomp;
13   define short_name/computed 'Short#Name' ;
14   compute short_name/character; short_name=substr(scan(product,1),1,1)||
15   substr(scan(product,2),1,1)||substr(scan(product,3),1,1);endcomp;
16   run;
```

Display 5 : Code Snapshot to use PROC REPORT with compute blocks

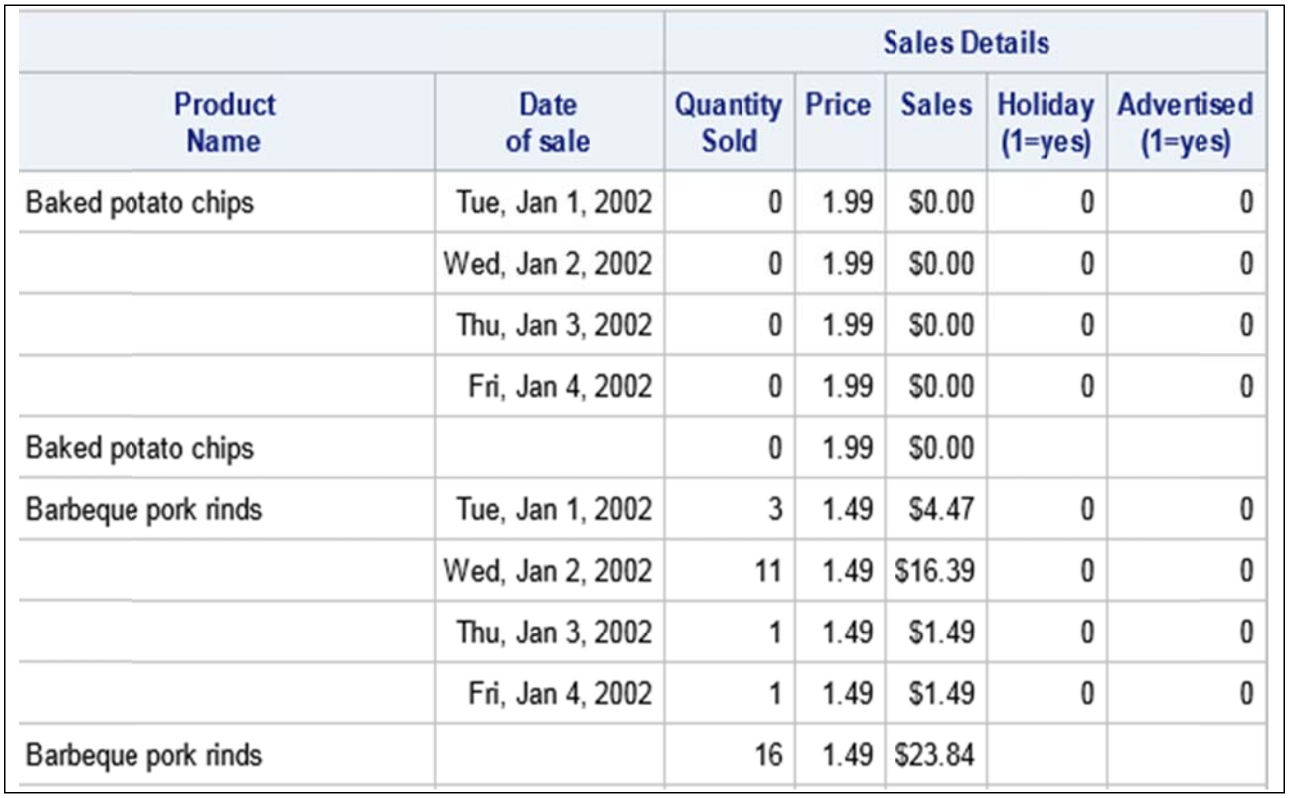
A few things need to be addressed using COMPUTE blocks.

1) In Line 2, the new report variables ("Short_Name" and "Sales") must be put to the right of any variables used to create these new variables in the COLUMN statement. For example, "Sales" is the new variable we create for this report using "QtySold" and "Price" so we put "Sales" right after these two variables in the COLUMN statement. However, it doesn't matter where you put DEFINE statement for the input variables before or after the COMPUTE blocks. Here Line 11 can be put after Line 12 but still generates the same results.

2) The COMPUTE blocks for numerical variables and character variables are different. To calculate numerical variables, we must use compound names for input variables with ANALYSIS usage, which is the default usage for numeric variables in the data, unless they are calculated in this PROC REPORT procedure. We won't get expected results if we compute "sales" as "*sales=qtysold*price*" inside the COMPUTE block as "qtrsold" and "price" have ANALYSIS usage. To calculate character variables, we must add CHARACTER or LENGTH option after the new character name as in Line 14.

DETAIL REPORT WITH SUMMARY INFORMATION

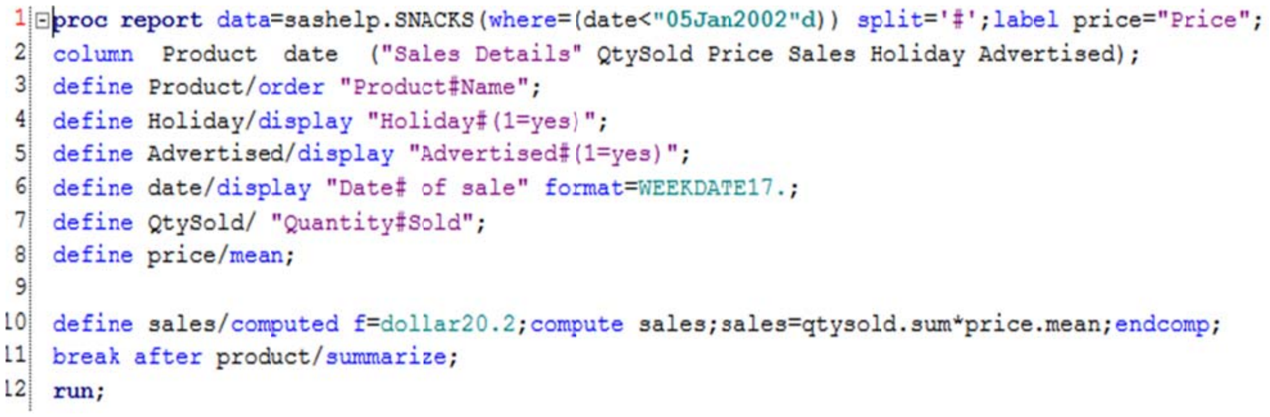
Detail reports can also contain summary information. In Output 6, we can add summary lines at the end of each product to calculate the subtotal statistics of quantity and sales and the mean of price.



		Sales Details				
Product Name	Date of sale	Quantity Sold	Price	Sales	Holiday (1=yes)	Advertised (1=yes)
Baked potato chips	Tue, Jan 1, 2002	0	1.99	$0.00	0	0
	Wed, Jan 2, 2002	0	1.99	$0.00	0	0
	Thu, Jan 3, 2002	0	1.99	$0.00	0	0
	Fri, Jan 4, 2002	0	1.99	$0.00	0	0
Baked potato chips		0	1.99	$0.00		
Barbeque pork rinds	Tue, Jan 1, 2002	3	1.49	$4.47	0	0
	Wed, Jan 2, 2002	11	1.49	$16.39	0	0
	Thu, Jan 3, 2002	1	1.49	$1.49	0	0
	Fri, Jan 4, 2002	1	1.49	$1.49	0	0
Barbeque pork rinds		16	1.49	$23.84		

Output 6 : Output of PROC REPORT statements with summary information

This can easily be done by adding a BREAK statement as below:



```
proc report data=sashelp.SNACKS(where=(date<"05Jan2002"d)) split='#';label price="Price";
column  Product  date  ("Sales Details" QtySold Price Sales Holiday Advertised);
define Product/order "Product#Name";
define Holiday/display "Holiday#(1=yes)";
define Advertised/display "Advertised#(1=yes)";
define date/display "Date# of sale" format=WEEKDATE17.;
define QtySold/ "Quantity#Sold";
define price/mean;

define sales/computed f=dollar20.2;compute sales;sales=qtysold.sum*price.mean;endcomp;
break after product/summarize;
run;
```

Display 6 : Code Snapshot to use PROC REPORT with BREAK STATEMENT

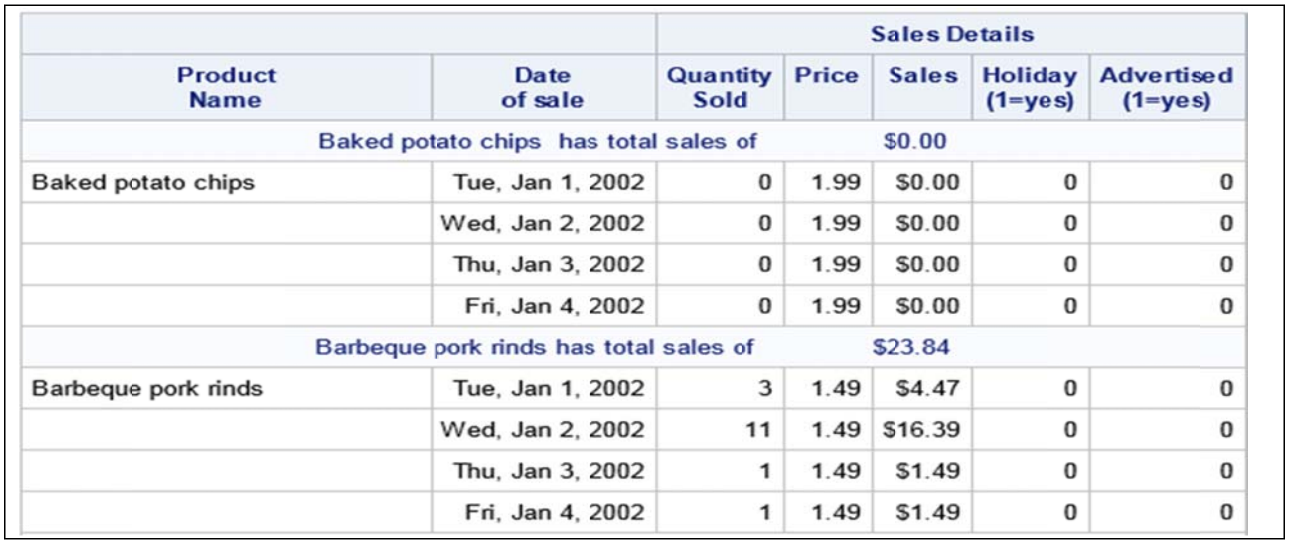
BREAK AFTER is used to perform break processing for each product. When using this statement, we should be mindful of these three things:

1) Changing the usage of group variable "Product" from the default usage DISPLAY to ORDER or GROUP because you can only BREAK on GROUP or ORDER variables.

2) Adding SUMMARIZE option at the end. Otherwise, no summary information is generated anywhere.

3) SUM is the default statistic for a numerical variable. If you want to change the default, specify a statistic keyword (MIN, MEAN, MAX, etc.) after the slash in the DEFINE statement for the variable. For example, in Line 8 we change the default usage of the variable “price” from SUM to MEAN to display the desired summary information so that “price” has to be referred and used as the MEAN price anywhere in the PROC REPORT procedure including the COMPUTE block in Line 10 and BREAK statement in Line 11.

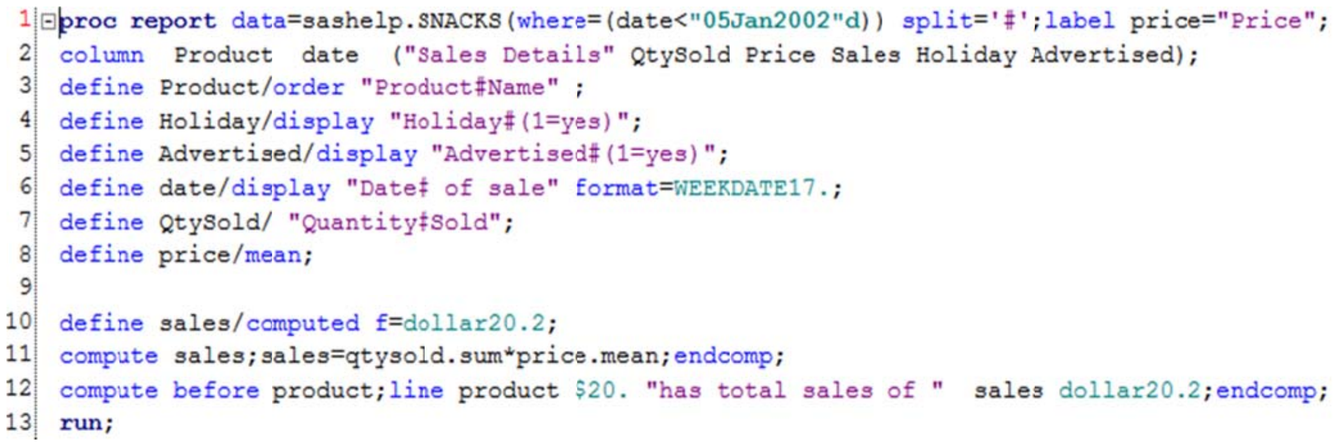
Sometimes, we may want to customize the summary information to display what interests us most. For example, we are only interested in the total sales amount rather than the total sales quantity and mean price. In Output 7, we create a customized break line above each product where we summarize the total sales for each product.



		Sales Details				
Product Name	Date of sale	Quantity Sold	Price	Sales	Holiday (1=yes)	Advertised (1=yes)
Baked potato chips has total sales of $0.00						
Baked potato chips	Tue, Jan 1, 2002	0	1.99	$0.00	0	0
	Wed, Jan 2, 2002	0	1.99	$0.00	0	0
	Thu, Jan 3, 2002	0	1.99	$0.00	0	0
	Fri, Jan 4, 2002	0	1.99	$0.00	0	0
Barbeque pork rinds has total sales of $23.84						
Barbeque pork rinds	Tue, Jan 1, 2002	3	1.49	$4.47	0	0
	Wed, Jan 2, 2002	11	1.49	$16.39	0	0
	Thu, Jan 3, 2002	1	1.49	$1.49	0	0
	Fri, Jan 4, 2002	1	1.49	$1.49	0	0

Output 7 : Output of PROC REPORT statements with summary information

This can easily be done by adding COMPUTE BEFORE block as below:



```
1  proc report data=sashelp.SNACKS(where=(date<"05Jan2002"d)) split='#';label price="Price";
2  column  Product  date  ("Sales Details" QtySold Price Sales Holiday Advertised);
3  define Product/order "Product#Name" ;
4  define Holiday/display "Holiday#(1=yes)";
5  define Advertised/display "Advertised#(1=yes)";
6  define date/display "Date# of sale" format=WEEKDATE17.;
7  define QtySold/ "Quantity#Sold";
8  define price/mean;
9
10 define sales/computed f=dollar20.2;
11 compute sales;sales=qtysold.sum*price.mean;endcomp;
12 compute before product;line product $20. "has total sales of "  sales dollar20.2;endcomp;
13 run;
```

Display 7 : Code Snapshot to use PROC REPORT with COMPUTE BEFORE block

The LINE statement can be used to modify the text that would normally appear on a summary line created by BREAK statement. To use this LINE statement correctly, you should note that:

1) It is valid only in COMPUTE BEFORE or COMPUTE AFTER block.
2) It is executed after all the other statements in the compute block regardless of placement in the block.
3) A character variable name must be followed by a format while a numerical variable must use a compound name to specify which statistics associated except for the newly computed variable like the variable “sales” in Line 12.

CONCLUSION

When asked to send detail data to business or management for review and investigation, you normally need to convert SAS data to other report formats if they do not have the access to SAS. If they have Office 2003 or higher, you can output the SAS data into Excel files under ODS environment. If the data is sufficiently big in the sense of number of rows or number of columns, we may want to enhance the Excel report to address some interesting items by changing the column and row formats or adding some summary information.

To produce a detail report, you don’t have the same flexibility with PROC SQL and PRINT as you do with PROC REPORT which provides a handful of options and statements users can leverage to control the appearance of the output of the data extensively. Unfortunately, despite its powerfulness, PROC REPORT seems to be used less often than the other tools possibly due to its seemingly complex coding.

This paper uses PROC REPORT under the ODS environment to export a detail report into a colorful and customized xml file so that non-SAS user can read the data easily for useful information. DEFINE, COMPUTE, and COLUMN statements are used in the paper along with several options to present your data in a more colorful way. It tells us how to control the format of the selected columns and rows. The headings of columns can be more meaningful and pretty. It also talks about how to color any selected cells differently to bring attention to the audience. Customized summary information can also be attached to the desired areas of the report.

REFERENCES

Zender, Cynthia. 2009. *SAS ® Report Writing 1: Using Procedures and ODS*. Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

I am grateful to Duke Owen and Pete Lund for their encouragement, guidance and always interesting correspondence. My colleagues in TCF bank also helped to produce what you find here.

CONTACT INFORMATION

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

Guihong Chen
TCF Bank
612-202-0550
guihongc@gmail.com