

Paper SIB-106

Automation of an Audit Waterfall Using SAS® Macros and ODS Tagsets.ExcelXP

Denise A. Kruse, Independent Contractor, Atlanta, GA

ABSTRACT

Programmers often need to communicate data results with analysts that do not have access to SAS but have many questions about the data. When forwarding the program log is simply not an option, a waterfall can fill the void and provide useful information. For the purposes of this paper, the term “waterfall” will be used to show how records in a targeted population fall out of that population as a result of exclusion or business rules. In other words, this waterfall will track the reduction of volume from the starting universe of records to the ending universe of records. An automated audit waterfall is one way to provide an analyst with precise data details without additional time spent cutting and pasting record counts from the SAS log to an excel spreadsheet. The automation of this code is made possible by using SAS Macros and Output Delivery System (ODS) Tagsets.ExcelXP.

Additional usage for this logic concept includes: quality assurance checks, quick reporting techniques and production monitoring.

INTRODUCTION

The primary purpose of this logic is to enable the programmer to communicate that the data has been manipulated correctly and accurately according to established business rules. This communication should be available instantly after successful program completion.

The scope of this paper will cover the concept of capturing record counts by use of a custom macro call and organizing that data into a reader friendly format in an excel spreadsheet with custom descriptions. This logic can be used for several different types of data validation.

The working example that will be used throughout this paper will focus on this logic as it applies to an audit waterfall provided to a non-technical business analyst. This data was extracted for the purpose of a marketing campaign list. The exclusion criteria rules follow both national contact standards and internal business standards.

The audience for this paper should be familiar with the concept of Macro programming.

First, this paper will present an example of the end result for the audit waterfall and explain how the data in the waterfall is derived. Then it will guide the reader through the steps necessary to build this data from the main program. Finally, the paper will guide the reader through automating this process.

COLLECTING THE DATA ELEMENTS

The focus of this waterfall report is the Amount column (see Figure A below). This column is the audit of how the data volume in the starting population dwindled down to the ending population. This data is derived from data set record counts that are captured and loaded into a table called WRPT with the variable name of “X” (see Figure B below). The logic to capture this data in the table called WRPT is accessed by including a program at the beginning of the main program:

```
%include '/myUNIX/mydir/waterfall_macro.sas';
```

(Note that the macro programs mentioned in this paper can be made available in four different ways: code written and compiled within the program, with an INCLUDE statement, through the autocall facility, or as stored compiled macros.)

The first step in capturing the excluded volumes for the report is creating data sets with the rejected data. This might be an adjustment for some programmers that are used to eliminating data with a where clause or some similar technique and moving on to process with the desired data.

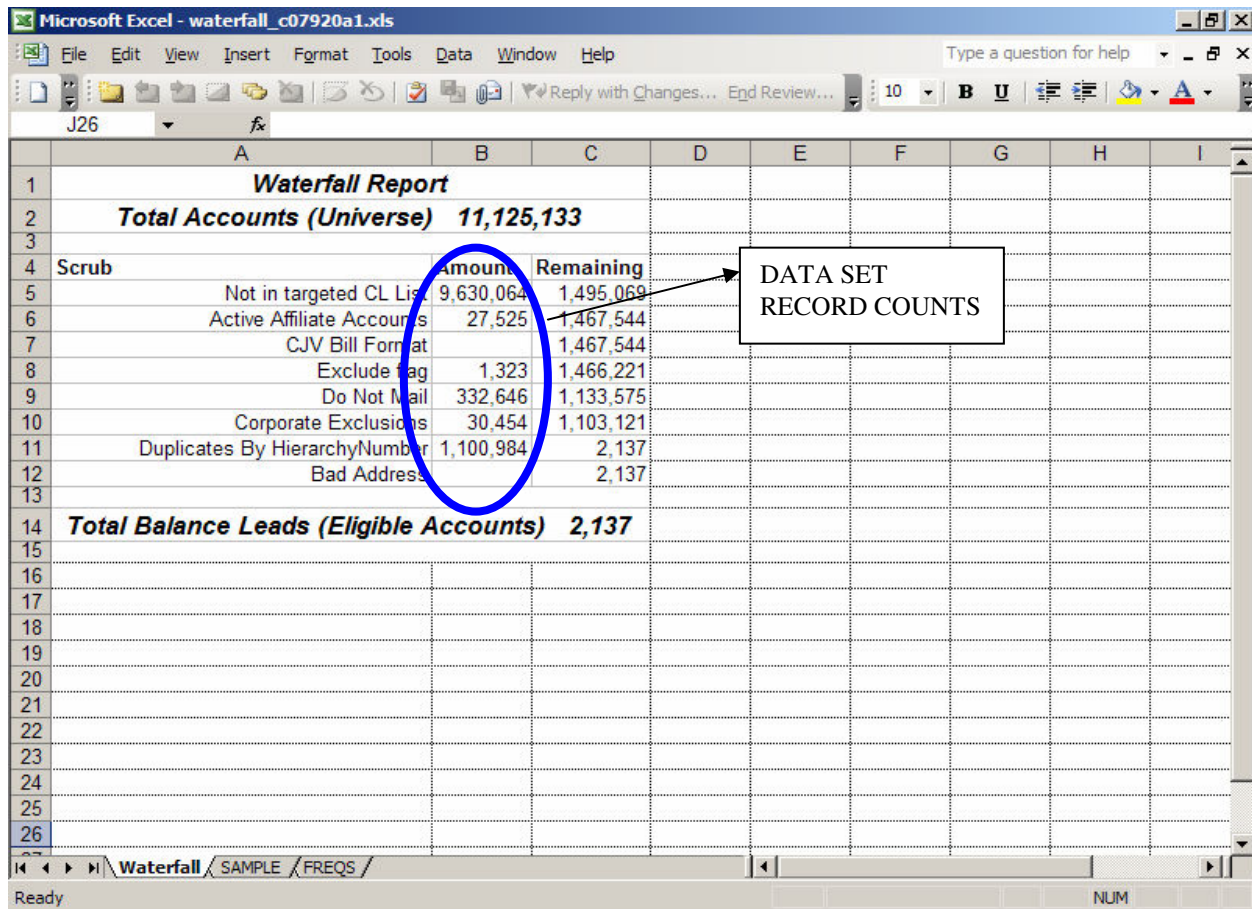


Figure A: Waterfall report showing the column that contains records counts of the undesired data.

In this first example, the programmer wants to select only the records that are eligible to receive a mailing solicitation. Using the Do Not Mail flag (1=yes, 0=no), the programmer executes the following code:

```
DATA desired_data ;
SET main (where=(dnm_flg=0)) ;
RUN ;
```

In order to use the waterfall logic as designed, the data step will need to be recoded to capture the undesired data:

```
DATA desired_data no_mail ;
SET main ;
IF dnm_flag=1 THEN OUTPUT no_mail ;
ELSE OUTPUT desired_data ;
RUN ;
```

After the data is captured in a data set, a macro call must be added (to the main program) to load the entry into the WRPT table. The syntax for the macro call is: %macro_name(library,dataset_name) ;

```
%ds_vol(work,no_mail);
```

The process for capturing undesired data by placing it in a data set and calling the DS_VOL macro should continue throughout the program until the desired ending population has been reached.

x
332,646

Figure B: Illustration of the table WRPT after the macro call to DS_VOL.

This waterfall_macro.sas program contains the macro that is used throughout the main program:

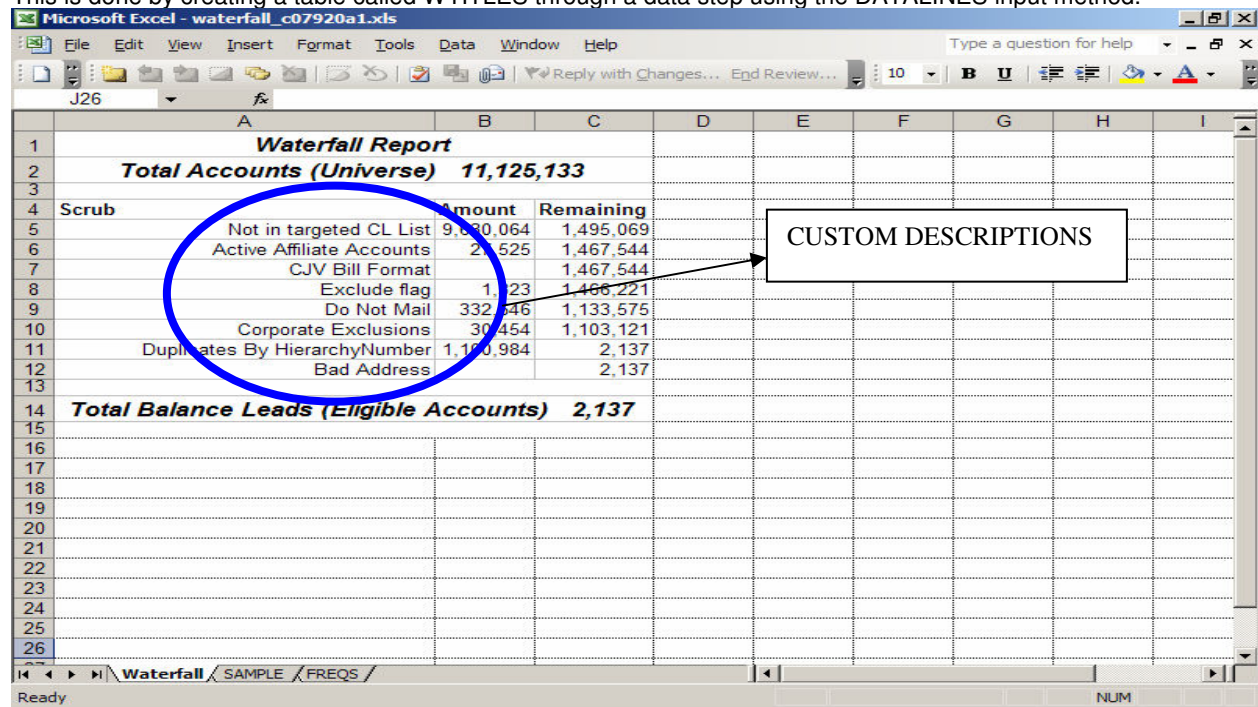
```
%MACRO ds_vol(lib, dsn) ;
    DATA _null_ ;
    IF 0 THEN SET &lib..&dsn. NOBS=NOBS ;
    PUT nob= ;
    CALL symputx('cnt', nob);
    STOP ;
    RUN ;

    DATA wrpt ;
    FORMAT x 8. ;
    LENGTH x 8. ;
    x=&cnt. ;
    RUN ;

    %IF %SYSFUNC(EXIST(wrpt_final))=1 %THEN %DO ;
        PROC APPEND BASE=wrpt_final
                DATA=wrpt FORCE ;
        RUN ;
    %END ;
    %ELSE %DO ;
        DATA wrpt_final ;
        FORMAT x 8. ;
        LENGTH x 8. ;
        SET wrpt ;
        RUN ;
    %END ;
%MEND ds_vol ;
```

CUSTOMIZING THE DESCRIPTIONS

Now that the data rows have been placed into a table, it is time to add custom descriptions (see Figure C below). This is done by creating a table called WTITLES through a data step using the DATALINES input method.



The screenshot shows a Microsoft Excel spreadsheet titled 'waterfall_c07920a1.xls'. The spreadsheet displays a 'Waterfall Report' with the following data:

	Amount	Remaining
Total Accounts (Universe) 11,125,133		
Scrub		
Not in targeted CL List	9,630,064	1,495,069
Active Affiliate Accounts	2,525	1,467,544
CJV Bill Format		1,467,544
Exclude flag	1,223	1,466,321
Do Not Mail	332,546	1,133,775
Corporate Exclusions	30,454	1,103,321
Duplicates By HierarchyNumber	1,160,984	2,137
Bad Address		2,137
Total Balance Leads (Eligible Accounts) 2,137		

Figure C: Illustrates the custom descriptions for the listed line items (record counts).

```

DATA wtitles ;
    FORMAT Y $100. ;
    INFILE datalines ;
    INPUT @ 1 Y & ;

DATALINES ;
Do Not Mail
RUN ;

```

The ampersand shown on the INPUT line allows for spaces between words in the custom description. Omitting this ampersand will result in truncation of the full description. In this case, it would read “Do” instead of the intended heading “Do Not Mail”.

This data step will need to appear in the main program. The order of the entries for the descriptions must align with the order of the macro calls throughout the program. This is necessary for the step where the two tables (WTITLES and WRPT) are merged together as the merge relies on positional alignment.

Example with multiple rows in the waterfall:

```

DATA affil_zip    cjbv_flag    exclude    no_mail        desired_data ;
SET main ;
IF affil_zip_flag = 1      THEN OUTPUT affil_zip ;
ELSE IF cjbv_flag = 1     THEN OUTPUT cjbv_flag ;
ELSE IF exclude_flag=1   THEN OUTPUT exclude ;
ELSE IF dnm_flag=1       THEN OUTPUT no_mail ;
ELSE OUTPUT desired_data ;
RUN ;

****WATERFALL - ROW 2;
%ds_vol(work,affil_zip);
****WATERFALL - ROW 3;
%ds_vol(work,cjbv_flag);
****WATERFALL - ROW 4;
%ds_vol(work,exclude);
****WATERFALL - ROW 5;
%ds_vol(work,no_mail);

DATA wtitles ;
    FORMAT Y $100. ;
    INFILE datalines ;
    INPUT @ 1 Y & ;

DATALINES ;
Active Affiliate Accounts
CJB Bill Format
Exclude flag
Do Not Mail
RUN ;

```

CODING THE HEADER AND FOOTER VOLUME

The volume that appears formatted with commas in the header and footer (see Figure D below) of the report is derived in two separate DATA steps that need to be added to the main program. The DATA step for the header is created by using the NOBS option of a SET statement to retrieve the record count. The SAS language routine SYMPUTX is then used to make the variable a macro variable which will be used in the title and footnote statements of the final report. A second call to SYMPUTX is necessary to create a macro variable without commas. This variable is used in calculating the values found in the Remaining column of the report (see Figure E below).

```

DATA _NULL_;
IF 0 THEN SET main nobobs=nobs;
CALL SYMPUTX('title1_amt2',put(nobs,comma11.));
CALL SYMPUTX('title1_amt',nobs);
STOP ;
RUN ;

DATA _NULL_;
IF 0 THEN SET eligible nobobs=nobs;
CALL SYMPUTX('eligible',put(nobs,comma11.));
STOP ;
RUN ;

```

Waterfall Report		
Total Accounts (Universe) 11,125,133		
Scrub	Amount	Remaining
Not in targeted CL List	9,630,064	1,495,069
Active Affiliate Accounts	27,525	1,467,544
CJV Bill Format		1,467,544
Exclude flag	1,323	1,466,221
Do Not Mail	332,646	1,133,575
Corporate Exclusions	30,454	1,103,121
Duplicates By HierarchyNumber	1,100,984	2,137
Bad Address		2,137
Total Balance Leads (Eligible Accounts) 2,137		

Figure D: Illustrates the formatted header and footer counts.

	A	B	C	D	E	F	G	H	I
1	Waterfall Report								
2	Total Accounts (Universe) 11,125,133								
3									
4	Scrub	Amount	Remaining						
5	Not in targeted CL List	9,630,064	1,495,069						
6	Active Affiliate Accounts	27,521	1,467,544						
7	CJV Bill Format		1,467,544						
8	Exclude flag	1,323	1,466,221						
9	Do Not Mail	332,646	1,133,575						
10	Corporate Exclusions	30,451	1,103,121						
11	Duplicates By HierarchyNumber	1,100,984	2,137						
12	Bad Address		2,137						
13									
14	Total Balance Leads (Eligible Accounts) 2,137								
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									

Figure E: Illustrates the calculated values derived in the custom macro program.

THE MAGIC OF MULTI_WATERFALL.SAS

This is the program that will calculate the Remaining column of the waterfall, create a custom template (using PROC TEMPLATE), execute PROC REPORT and finally, use ODS Tagsets.ExcelXP to create the excel spreadsheet. The time savings for the programmer comes from using this customizable program. The waterfall becomes available immediately after successful program execution and allows the programmer to move onto the next project while the business analyst examines and approves the data.

The benefit of using this method includes increased productivity and built in quality assurance checks. As a contractor, this method assists in working quickly and efficiently and as a manager; this tool will enable easy spot checks of an employee's work.

The magic is that it only takes one line of code in the main program to include all of that logic in the existing program:

```
%include '/myUNIX/mydir/multi_waterfall.sas';
```

INSIDE MULTI_WATERFALL.SAS

As mentioned in the previous section, this program is where most of the modifications will take place when adapting this logic for other needs. This existing program can be broken out into several sections:

- Defining the aesthetics of the report using PROC TEMPLATE
- Determine the number of rows in the waterfall (as this is a dynamic number)
- Calculate the value in each of the rows of the Remaining column
- Merge the data sets WTITLES and WRPT together
- Invoke ODS Tagsets.ExcelXP and define the destination filename
- Customize the first tab of the excel worksheet using ODS Tagsets.ExcelXP
- Customize and execute a PROC REPORT that will appear on the first tab of the excel worksheet
- Close ODS Tagsets. ExcelXP and turn on the default listing

DEFINING THE AESTHETICS OF THE REPORT USING PROC TEMPLATE

When designing the waterfall used in this paper, the only requirements were that the column headers be bolded and the numbers be formatted with commas. Therefore, this example is a very basic design that was slightly modified from the SANSPRINTER style offered by SAS.

The ODS PATH statement provides instruction for ODS:

- WORK.TEMPLAT provides space for temporary templates created and used during the current SAS session.
- SASUSER.TEMPLAT is the location of any previously created templates.
- SASHELP.TMPLMST is the location of SAS standard templates.

```
ODS PATH work.templat(update) sasuser.templat(read)
sashelp.tmplmst(read);

PROC TEMPLATE ;
DEFINE style styles.waterfall ;
    parent = styles.sansprinter;
style numeric_data from data /
    tagattr='format:#,##0;;'
    just = right
;
END ;
RUN ;
```

Keep in mind that this template can be replaced with a custom template that is already in use for a certain company. It can also be modified to include more colors or details if desired.

DETERMINE THE NUMBER OF ROWS IN THE WATERFALL

This piece of code needs to be dynamic so that several programs may use this logic and have the appropriate number of waterfall rows (in accordance with the exclusions for that program) appear on the spreadsheet. Therefore, the data set that determines the number of rows in the waterfall is WTITLES. This is where the custom descriptions were designated. The automation will use the number of rows in this table as a gauge in determining the number of waterfall rows necessary.

```
DATA _NULL_;
IF 0 THEN SET wtitles nobs=nobs;
CALL symputx('cnt', (nobs));
STOP ;
RUN ;
```

CALCULATE THE VALUES IN THE REMAINING COLUMN

The starting volume for this calculation is the record count from the starting data set. In this case it was passed in the macro variable title1_amt. The logic will start with this value and subtract each exclusion row found in table WRPT_FINAL.

```
DATA wrpt2;
    FORMAT remain1-remain&cnt. num prev 8. ;
    LENGTH remain1-remain&cnt. num prev 8. ;
SET wrpt_final;
    RETAIN remain1-remain&cnt. prev ;
    IF _n_ =1 THEN DO;
        *initialize counter and array;
        num=0;
        ARRAY remain (1:&cnt.) 8.;
        remain(1) = &title1_amt. - x;
        TOT_REMAIN=remain(1) ;
        OUTPUT ;
        num + 1 ;
    END ;
    ELSE DO ;
        num + 1 ;
        prev=num-1 ;
    END ;
```

```

        remain(num)= remain(prev)  - x ;
        TOT_REMAIN=remain(num) ;
        OUTPUT ;
    END ;
    RUN;

```

JOINING THE DATA

The final step in the data manipulation piece of this logic is to merge the tables together so PROC REPORT has all the information in one data set. At this point in the process, one data set holds all of the calculated values and exclusion volume (WRPT2) and one holds the customized descriptions (WTITLES).

Y	X	TOT_REMAIN
Not in targeted CL List	9,630,064	1,495,069
Active Affiliate Accounts	27,525	1,467,544
CJV Bill Format		1,467,544
Exclude flag	1,323	1,466,221
Do Not Mail	332,646	1,133,575
Corporate Exclusions	30,454	1,103,121
Duplicates By Hierarchy Number	1,100,984	2,137
Bad Address		2,137

Figure F: Illustrates the last two tables to be merged WRPT2 and WTITLES.

```

DATA skippy ;
FORMAT y $100. ;
MERGE wtitles wrpt2 ;
RUN ;

```

INVOKE ODS TAGSETS.EXCELXP AND DEFINE THE DESTINATION FILENAME

The data is ready to be displayed in an excel spreadsheet for the analyst. The flexible and incredible SAS tool that will make this happen is ODS Tagsets.ExcelXP. This tool warrants detailed exploration as it can open many new and creative ways to deliver data to customers.

The first step in using this tool is closing the default listing and defining the filename destination:

```

ODS LISTING CLOSE ;
RUN ;
ODS Tagsets.ExcelXP FILE="/myUNIX/mydir/waterfall/my_waterfall.xls"
                    STYLE=waterfall;
                    *options(doc='help' );

```

Notice this is where the waterfall style defined at the beginning of this program is used. Also, a very helpful option that can be invoked is doc='help'. This will display various options for Tagsets.ExcelXP that can be used in designing the aesthetics of the final spreadsheet. It does make the log file much longer which is why it is commented out in the example code above.

CUSTOMIZE AND EXECUTE A PROC REPORT THAT WILL APPEAR ON THE FIRST TAB OF THE EXCEL WORKSHEET

Each tab in the excel spreadsheet needs to be defined with an ODS Tagsets.ExcelXP options statement:

```
ODS Tagsets.ExcelXP options(sheet_name='Waterfall'
                             embedded_titles='Yes'
                             embedded_footnotes='Yes');
```

The design in this example requires a title and footnote. This is also where the sheet name is customized.

Now it is time to tell ODS how to display the data that was collected. In this example, a PROC REPORT is used to display the information on the first tab of the excel spreadsheet.

```
PROC REPORT DATA=skippy SPLIT='*' LS=256 HEADLINE nowd
STYLE(header)={font_weight=bold just=CENTER} ;
TITLE1 "Waterfall Report" ;
TITLE2 "Total Accounts (Universe) " &title1_amt2. ;
FOOTNOTE "Total Balance Leads (Eligible Accounts) " &eligible. ;
COLUMN y
       x
       TOT_REMAIN;
DEFINE y / ORDER=data 'Scrub' ;
DEFINE x / ORDER=data STYLE=numeric_data 'Amount' ;
DEFINE TOT_REMAIN / ORDER=data STYLE=numeric_data 'Remaining' ;
RUN ;
```

Note that any PROC or DATA step can appear after the ODS Tagsets.ExcelXP options statement.

CLOSE ODS TAGSETS. EXCELXP AND TURN ON THE DEFAULT LISTING

Similar to any other code that is written with ODS, Tagsets.ExcelXP needs to be closed and the default listing should be re-invoked.

```
ODS tagsets.ExcelXP CLOSE ;
ODS LISTING ;
RUN ;
```

ENHANCING THE CUSTOM REPORT

That completes the process to automate and streamline the waterfall logic for the first tab of the report. There is so much more that a programmer can do to provide more proof of data validation. For the project that this automated waterfall was designed, part of the mandatory quality assurance check was to provide a sample of the data to the business analyst that would be sent to the vendor or loaded into production. In short, a data dump of 50 records was done using standard PROC PRINT (see Figure G below). As noted in the previous section, to define each tab in the excel spreadsheet, a Tagsets.ExcelXP option statement must precede the PROC or DATA step. Therefore, the logic to add a second tab with a sample of the final data looks like this:

```
ODS Tagsets.ExcelXP options(sheet_name='SAMPLE'
                             embedded_titles='No'
                             embedded_footnotes='No');

PROC PRINT DATA = &sample. (OBS=&s_cnt.) ;
RUN ;
```

While the ODS Tagsets.ExcelXP logic belongs in multi_waterfall.sas, the macro variables used in this example should be passed from the main program:

```
DATA _NULL_ ;
*** insert the name of the final data set to sample records from and run
the frequencies from;
smpl='work.eligible' ;
scnt=50 ;
CALL symputx('sample',smpl) ;
CALL symputx('s_cnt',scnt) ;
RUN ;
```

Obs	lit_code	val_prop	ACCT_LOF_MONTHS	ACCT_START_DT	ACCT_STATUS_CODE	ACCT_TYPE_CODE	AC
1	FE8006-3140	MLGRTB011A	24	09JAN2006	A	B	
2	FE8006-3140	MLGRTB011A	12	23JAN2007	A	P	
3	FE8006-3140	MLGRTB011A	84	18JAN2001	A	M	
4	FE8006-3140	MLGRTB011A	4	27SEP2007	A	B	
5	FE8006-3140	MLGRTB011A	9	03APR2007	A	B	
6	FE8006-3140	MLGRTB011A	56	19MAY2003	A	I	
7	FE8006-3152	MLGRTB051A	4	20SEP2007	A	M	
8	FE8006-3140	MLGRTB011A	92	18MAY2000	A	B	
9	FE8006-3140	MLGRTB011A	24	19JAN2006	A	M	
10	FE8006-3140	MLGRTB011A	14	30OCT2006	A	B	
11	FE8006-3140	MLGRTB011A	11	31JAN2007	A	B	
12	FE8006-3140	MLGRTB011A	38	17NOV2004	A	N	
13	FE8006-3140	MLGRTB011A	94	17MAR2000	A	N	
14	FE8006-3140	MLGRTB011A	23	14FEB2006	A	N	
15	FE8006-3140	MLGRTB011A	25	23DEC2005	A	N	
16	FE8006-3152	MLGRTB051A	2	21NOV2007	A	B	
17	FE8006-3140	MLGRTB011A	125	31JUL1997	A	M	
18	FE8006-3140	MLGRTB011A	22	06MAR2006	A	N	
19	FE8006-3149	MLGRTB041A	53	14AUG2003	A	B	
20	FE8006-3140	MLGRTB011A	94	02MAR2000	A	M	
21	FE8006-3140	MLGRTB011A	108	29JAN1999	A	M	
22	FE8006-3140	MLGRTB011A	8	26MAY2007	A	B	
23	FE8006-3140	MLGRTB011A	94	22MAR2000	A	B	
24	FE8006-3149	MLGRTB041A	1	16DEC2007	A	B	
25	FE8006-3140	MLGRTB011A	53	14AUG2003	A	N	
26	FE8006-3140	MLGRTB011A	102	29JUL1999	A	M	
27	FE8006-3140	MLGRTB011A	93	18APR2000	A	B	
28	FE8006-3140	MLGRTB011A	93	29APR2000	A	N	
29	FE8006-3140	MLGRTB011A	38	11NOV2004	A	M	

Figure G: Data dump of sample records as part of quality assurance checks.

The final requirement for this project was to provide frequencies of certain variables (see Figure H below). The variables to be displayed were defined in the main program and passed with a macro to multi_waterfall.sas:

```
DATA _NULL_ ;
freq_vars='
    dnc_flag          dnm_flag
    network_flag      cl_il_flag
    exclude_flag      affil_zip_flag
    cjb_flag          bad_debt_flag
    active_flag       acct_status_code
    sub_status_code   con_mnth_to_end_qty
    hier_corp_id      hier_corp_nm
';
CALL symputx('freq_vars',freq_vars) ;
RUN ;
```

The third tab was then defined in multi_waterfall.sas:

```
ODS Tagsets.ExcelXP options(sheet_name='FREQS'
                             sheet_interval='None'
                             embedded_titles='No'
                             embedded_footnotes='No');
```

```
PROC FREQ DATA = &sample.;
TABLE &freq_vars. / MISSING ;
RUN ;
```

Microsoft Excel - waterfall_example.xls

File Edit View Insert Format Tools Data Window Help

Type a question for help

10 B I U

L4

1	DNC_FLAG										
2	DNC_FLAG	Frequency	Percent	Cumulative Frequency	Cumulative Percent						
3	0	1666	77.78	1666	77.78						
4	1	476	22.22	2142	100						
5											
6	DNM_FLAG										
7	DNM_FLAG	Frequency	Percent	Cumulative Frequency	Cumulative Percent						
8	0	2142	100	2142	100						
9											
10	NETWORK_FLAG										
11	NETWORK_FLAG	Frequency	Percent	Cumulative Frequency	Cumulative Percent						
12		5	0.23	5	0.23						
13	C	2137	99.77	2142	100						
14											
15	CL_IL_FLAG										
16	CL_IL_FLAG	Frequency	Percent	Cumulative Frequency	Cumulative Percent						
17		5	0.23	5	0.23						
18	CL	2137	99.77	2142	100						
19											
20	EXCLUDE_FLAG										
21	EXCLUDE_FLAG	Frequency	Percent	Cumulative Frequency	Cumulative Percent						
22	0	2142	100	2142	100						
23											
24	AFFIL_ZIP_FLAG										
25	AFFIL_ZIP_FLAG	Frequency	Percent	Cumulative Frequency	Cumulative Percent						
26	0	2142	100	2142	100						
27											

Waterfall SAMPLE FREQS

NUM

Figure H: Frequencies of key variables displayed as part of quality assurance checks.

CONCLUSION

The goal of this paper is to encourage automation of processes that are manual and use up valuable time. This paper detailed one business requirement that was solved with automation and improved the level of communication between programmer and business analyst. It brought more independence to the business analyst and made more time available in the programmer's schedule to increase productivity.

There are many different ways to conquer the communication barrier. Using SAS Macros and ODS Tagsets.ExcelXP was the best solution available for the specifications of this project. Be creative and spend time developing methods to increase efficiency and accuracy in any industry or department.

REFERENCES

- <http://support.sas.com/rnd/base/ods/odsmarkup/p236-31.pdf>
Author: Eric Gebhart, SAS Institute Inc.

ACKNOWLEDGMENTS

THANKS TO:

- Bruce Johnson from SAS-L for help with PROC template
- David Steves from SunTrust Bank for always encouraging me to explore the SAS magic

CONTACT INFORMATION

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

Denise A. Kruse
Independent Contractor
Atlanta, Georgia
Work Phone: 678-457-3677
E-mail: deniseakruse@gmail.com
Web: www.gasug.org

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 trademarks of their respective companies.