Paper 283-2012

An Automatic Approach to Create Data Dictionary in SharePoint Using SAS®

Kevin Chung, Fannie Mae, Washington DC

ABSTRACT

You might have several data sets created by SAS applications and want to share the information about these data to users. This paper illustrates how to use SAS to create a data dictionary for any existing SAS data sets and store the result in a form of html file in the Microsoft SharePoint. A data dictionary is a collection of information about data such as name, attribute, definition, source, data type, and length. SharePoint is a web application platform developed by Microsoft. It's widely used by divisions/departments in an enterprise as a portal to communicate with internal and/or external users. By creating a data dictionary and making it available on SharePoint, new and old users will have a single location to pull descriptive attributes about fields for data sets quickly and efficiently.

INTRODUCTION

This is a revised and concise version based on the major output SAS data sets of a financial model in Fannie Mae. The inputs used in our application are the mortgage data and they should not be disclosed. To demonstrate the entire process, an input SAS data set was created based on the SAS-provided data set SASHELP.prdsale. All programs used in this paper have been thoroughly tested using SAS 9.2 on IBM Unix AIX 6.1 platform. In addition to the metadata information for each variable, the data dictionary created by this process also includes frequency tables for categorical fields, character variables, discrete numeric, and continuous numeric variables with the user-defined format, and the outputs of proc UNIVARIATE for continuous numeric variables. It also contains the descriptions of the fields, table name, logics used to derive the field, data type, and length. Users can easily browse the frequency table and/or UNIVARIATE statistics output and attributes of data field by clicking the "Name" and/or "Description" columns. Please refer to Appendix A for more information. This paper only focuses on the SAS process to create the data dictionary web page and does not cover any configuration setup for creating document library and web part in SharePoint.

Assumptions and applications used in this paper are as follows.

- SAS 9.2 and Microsoft Excel 2003 are used.
- The SharePoint 2007 is available and the document library and web part has been created to store the files.
- The input SAS data set is created on a monthly basis.

The data dictionary can be located by the following link

http://www.kevin-chung.com/SGF2012/Product_Sale_201203.html

All input data, source codes, and presentation materials can be downloaded from www.kevin-chung.com

TEST DATA & ENVIRONMENT

The SAS-supplied data set SASHELP.prdsale is used as the input data to create a mocked test data set. **Table 1a** below contains the partial listing of the original SAS data set SASHELP.prdsale.

COUNTRY	REGION	DIVISION	PRODTYPE	PRODUCT	QUARTER	YEAR	MONTH
CANADA	EAST	EDUCATION	FURNITURE	SOFA	1	1993	Jan
CANADA	EAST	EDUCATION	FURNITURE	SOFA	1	1993	Feb
CANADA	EAST	EDUCATION	FURNITURE	SOFA	1	1993	Mar
CANADA	EAST	EDUCATION	FURNITURE	SOFA	2	1993	Apr
CANADA	EAST	EDUCATION	FURNITURE	SOFA	2	1993	May

Table 1a

To demonstrate the features of the SAS process, some mocked fields are derived. Several assumptions are made below for the input SAS data set.

1. A mocked field, **actg_dt**, stands for accounting date, is created. It's always the first day of the month.
2. The data set contains the rolling 18-month data based on the current accounting date.
3. The label contents have been removed from the input SAS data set.

Table 1b below shows the partial listing of the input data set after adding some mocked fields.

COUNTRY	REGION	DIVISION	PRODTYPE	Product_ Category_ Code	QUARTER	YEAR	MONTH	actg_dt	Diff	Reach_ Goal
CANADA	EAST	EDUCATION	FURNITURE	4	1	2011	Jan	201101	75	Y
CANADA	EAST	EDUCATION	FURNITURE	4	1	2011	Feb	201102	702	Y
CANADA	EAST	EDUCATION	FURNITURE	4	1	2011	Mar	201103	-238	N
CANADA	EAST	EDUCATION	FURNITURE	4	2	2011	Apr	201104	109	Y
CANADA	EAST	EDUCATION	FURNITURE	4	2	2011	May	201105	10	Y

Table 1b

The libname reference **SGF2012** refers to a Unix directory `/home/sgf2012` and the input data set **prdsale** is created in this directory. When this data set is created, the purpose of the ATTRIB statement below is to remove the label contents to simplify the parse of the output of FREQ procedure. This is the item 3 at the top of this page.

```
proc datasets lib=SGF2012 nolist;
  modify prdsale;
  attrib _all_ label=' ';
quit;
```

The code for creating the test data, **crt_test_data.sas**, is included in the download package.

DEFINE METADATA IN AN EXCEL FILE

The metadata should be defined in an Excel file so that the SAS process reads the information from the Excel file and produce appropriate outputs in html file for each data element.

	A	B	C	D	E	F	G	H	I
1	Name	Type	LENGTH	FLAG	Format	Table	Attribute	Definition	Logic
2	ACTUAL	NUM	8	U		tbl_SALE	Actual Sale Amount	Actual sale amount	
3	COUNTRY	CHAR	10	F		tbl_GEO	Country	Country name	
4	DIVISION	CHAR	10	F		tbl_GEO	Division	Division name	
5	Diff	NUM	8	U		Derived	Difference	sale amount and PREDICT amount	Diff = Actual - Predict<EOL>
6	MONTH	NUM	8	N			Month		
7	PREDICT	NUM	8	U		tbl_GOAL	Predict Sale Amount	The predict sale amount	
8	PRODTYPE	CHAR	10	F		tbl_PRODUCT	Product Type	Product type	
9	Product_Category_Code	NUM	8	F	prdctfmt	tbl_PRODUCT	Product Category Code	Product Category Code	
10	QUARTER	NUM	8	N			Quarter		
11	REGION	CHAR	10	F		tbl_GEO	Region	Region	
12	Reach_Goal	CHAR	1	F		Derived	Reach the Goal	PREDICT sale amount from ACTUAL	Reach_Goal='Y'<EOL>
13	YEAR	NUM	8	N			Year		
14	actg_dt	NUM	8	F	dtfmt	tbl_SALE	Accounting Period	Accounting period	

Table 2

- **Name** – SAS variables
- **Type** – NUM or CHAR
- **LENGTH** – SAS variable length
- **FLAG** – F: Frequency Count U: Univariate Statistics N: N/A, can be ignored or reserved for future use

- **Format** – if the variable needs a format in FREQ procedure
- **Table** – a mock-up table name indicating the source of the field
- **Attribute** – The attribute or characteristic of the field
- **Definition** – The definition of the field
- **Logic** – The logic used to derive the field

NOTE:

- The Attribute and Definition fields are not necessary to be presented in the same Description section.
- When enter the logics to the cells in column I, remember to add the <EOL>, stands for end of line, to the end of each row to ensure the contents can be printed in the Logic section of html file correctly.
- The Excel file should be transferred to Unix as the input data.

The chart in **Table 3** below shows the use of each field in the Description section of the html file.

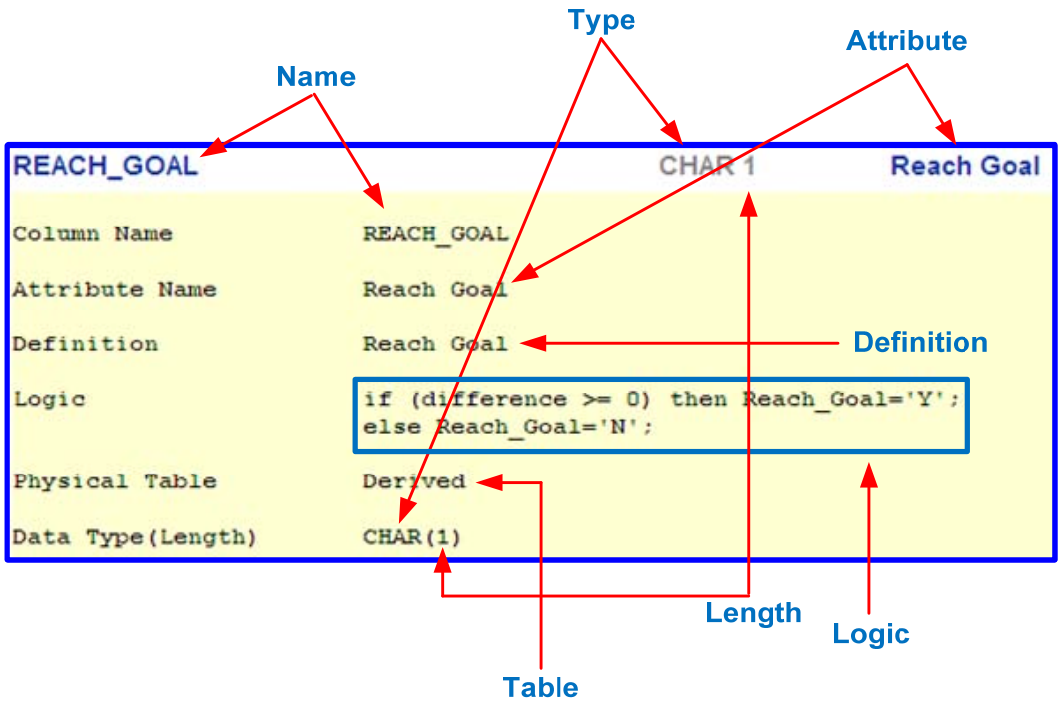


Table 3

CREATE AN HTML TEMPLATE FILE

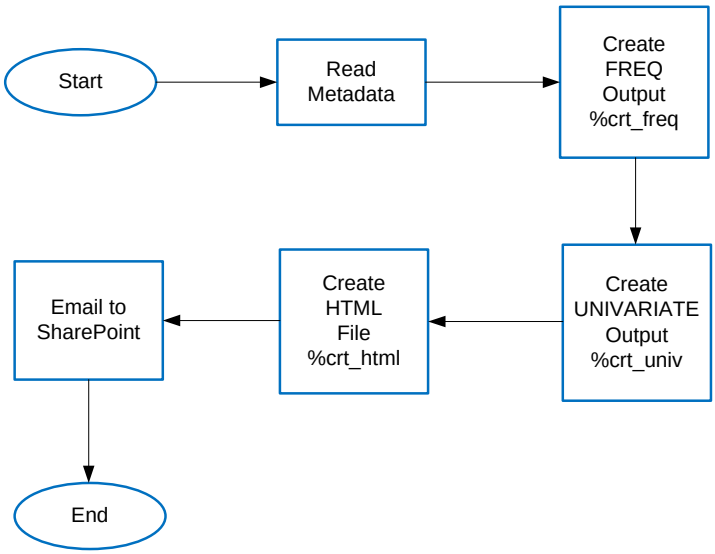
The HTML template, **DD_template.html**, is a pre-defined HTML file with embedded JavaScript functions to expand (display) / collapse (hide) the data dictionary information. This file is provided in the download package.

The following contents are hard-coded as macro variables so that they can be replaced with the values defined in the SAS program. This is to make the SAS program and HTML file more generic for general-purpose use.

```
<H1><FONT face="Arial">Data Dictionary for <FONT color="red">&actg_month</FONT>
Product Sale</FONT></H1>


<I>Questions or comments?   Please contact &contact or send an e-mail to <A
href="mailto:&email_contact">&email_contact</A></I>
```

SAS PROCESS

Several SAS steps are required to automate the entire process. A high level process flow is described in the following flow diagram.



Read Metadata

A SAS product, **SAS/ACCESS® Interface to PC Files**, is required to read the true Excel file on Unix platform using IMPORT procedure. If this license is not available, save the Excel file as a csv file format and use data step to create the same output data set meta_DD.

```
proc import datafile="&home/meta_DD.xls" out=meta_DD dbms=xls replace;
  datarow=2; getnames=Yes;
run;

data &meta;
  length Name $32 fmt_name $12;
  set meta_DD;
  where Flag in ('F','U');

  n+1;
  k=put(n,3. -L);
  name=upcase(name);
  if (Flag='F') and (format ne ' ') then fmt_name=compress(ifc(type=:'N',' ',
    '$')||format||".");
  drop n k;
run;
proc sort data=&meta; by Name; run;
```

Once the metadata SAS data set is created, identify the variable list that will be used by FREQ and UNIVARIATE procedures. This is based on the FLAG field in column D of **Table 2**.

```
proc sql noprint;
  /* Frequency */
  select name into :freq_list separated by ' '
  from &meta
  where Flag='F'
  order by name;

  /* format */
  select trim(name)||' '||trim(fmt_name) into :fmt_list separated by ' '
  from &meta
  where fmt_name is not NULL
  order by name;
```

```
  /* Univariate */
  select name into :univ_list separated by ' '
  from &meta
  where Flag='U'
  order by name;
quit;
```

The variable lists for FREQ and UNIVARIATE procedures are printed on the log as follows:

```
=========
Frequency
=========
freq_list=ACTG_DT COUNTRY DIVISION PRODTYPE PRODUCT_CATEGORY_CODE REACH_GOAL REGION
fmt_list=ACTG_DT dtfmt. PRODUCT_CATEGORY_CODE prdctfmt.

==========
Univariate
==========
univ_list=ACTUAL DIFF PREDICT
```

Apply FREQ Procedure

Some long variable name can not be displayed in one line. An example is shown in **Table 4a** below.

Product_ Category_ Code	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1 - BED	216	20.00	216	20.00
2 - CHAIR	216	20.00	432	40.00
3 - DESK	216	20.00	648	60.00
4 - SOFA	216	20.00	864	80.00
5 - TABLE	216	20.00	1080	100.00

Table 4a

To overcome this issue and make the SAS process easier, the modification of the **Base.Freq.OneWayList** template is necessary.

```
proc template;
  edit Base.Freq.OneWayList;
    edit FVariable;
      width=30; ❶
      just=left;
      parent = Base.Freq.FVariable;
    end;
  end;
run;
```

The purpose of ❶ is to increase the width of FREQ output. The &freq_list and &fmt_list are passed to FREQ procedure below. The PRINTTO procedure only saves the output from FREQ procedure.

```
proc printto print="&home/crt_freq.lst" new; run;
title;
ods noproctitle;
proc freq data=SGF2012.prdsale;
  table &freq_list/missing;
  format &fmt_list;
quit;
```

Table 4b shows one of the output from the FREQ procedure after modifying the Base.Freq.OneWayList template.

Product_Category_Code	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1 – BED	216	20.00	216	20.00
2 - CHAIR	216	20.00	432	40.00
3 - DESK	216	20.00	648	60.00
4 - SOFA	216	20.00	864	80.00
5 - TABLE	216	20.00	1080	100.00

Table 4b

Based on the output file crt_freq.lst, create another output file crt_freq2.lst by removing all blank lines. During the data step process, create macro variables to store each row and identify the line number for the first row of each variable.

```
data _null_;
  infile "&home/crt_freq.lst" end=eof;
  input;
  if (_infile_=' ') then delete;
  else do;
    file "&home/crt_freq2.lst";
    put _infile_;

    n+1;
    k=put(n,4. -L);
    call symput('line'||k, _infile_);  /* store each row including leading space */

    if index(_infile_,'Cumulative') & (scan(_infile_,-1,' ')='Cumulative') then do;
      x+1;
      call symputx('cum'||put(x,3. -L),k,'G'); /* identify the line number for   */
    end;                                       /* the first row of each variable */
  end;

  if eof then do;
    call symputx('n_of_line',k);
    call symputx('n_of_var',x);
  end;
run;
```

Partial Listing of crt_freq2.lst

line1

			Cumulative	Cumulative
actg_dt	Frequency	Percent	Frequency	Percent
2010	180	16.67	180	16.67
2011	540	50.00	720	66.67
201110	60	5.56	780	72.22
201111	60	5.56	840	77.78
201112	60	5.56	900	83.33
201201	60	5.56	960	88.89
201202	60	5.56	1020	94.44
201203	60	5.56	1080	100.00
			Cumulative	Cumulative
COUNTRY	Frequency	Percent	Frequency	Percent
CANADA	360	33.33	360	33.33
GERMANY	360	33.33	720	66.67
U.S.A.	360	33.33	1080	100.00
			Cumulative	Cumulative
DIVISION	Frequency	Percent	Frequency	Percent
CONSUMER	540	50.00	540	50.00
EDUCATION	540	50.00	1080	100.00

cum1=1

cum2=12

cum3=18

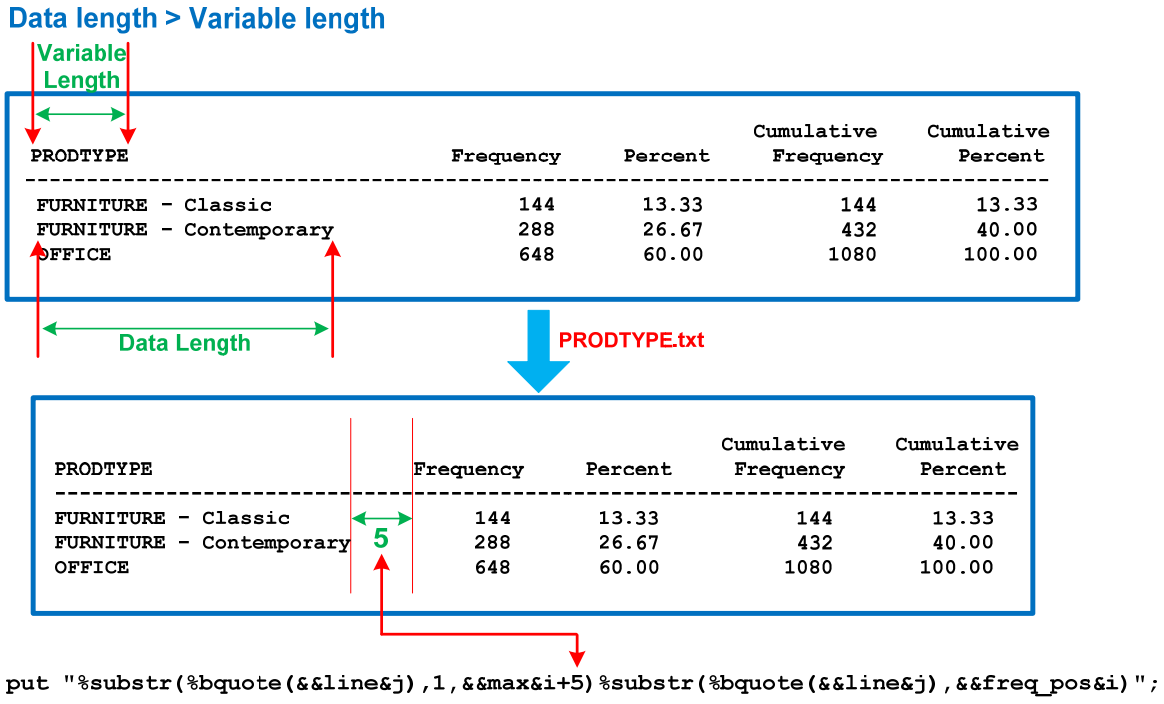
line22

Table 5

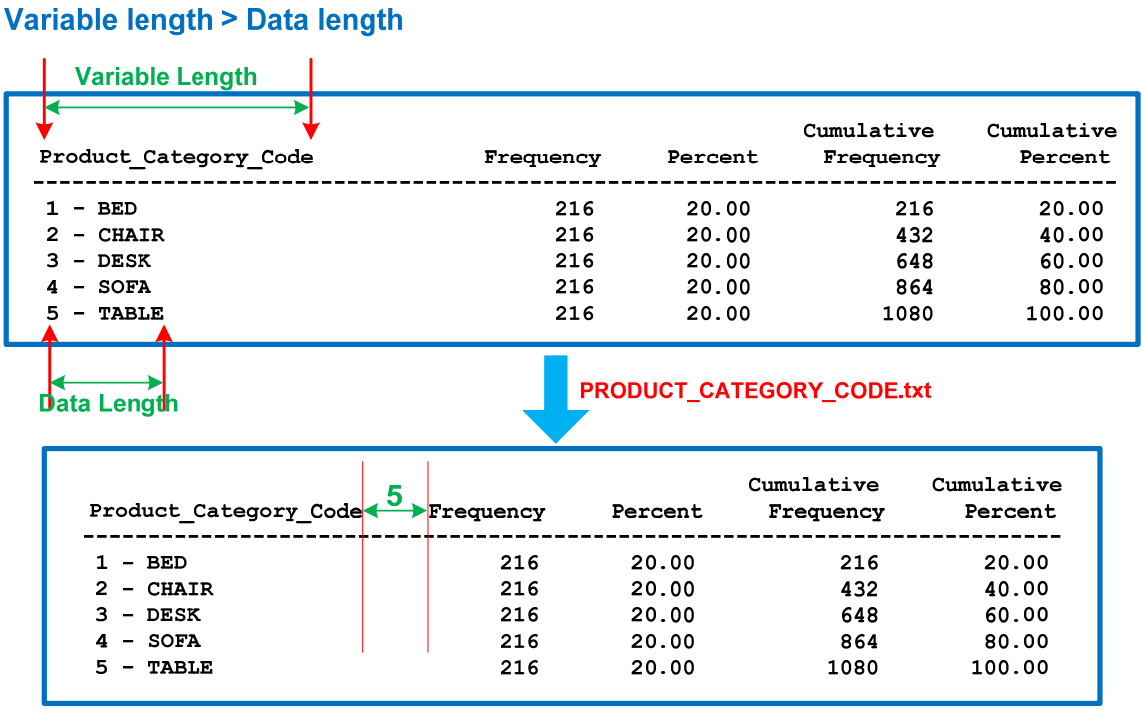
Then the next step is to parse the contents of FREQ output and create a more readable format for each variable. Two cases have to be considered based on the **data length** and **variable length**. The macro variables max&i are created to store the maximum length for each variable. The following codes inside the %DO loop are the excerpts from %prcs_freq.

```
%do i=1 %to &n_of_var;
  %global end&i f_line_count&i f_var&i;

  %let max&i=1;
  %let k=%eval(&i+1);
  %if (&i < &n_of_var) %then %let end&i=%eval(&&cum&k-1);
  %else %let end&i=&n_of_line;

  %let f=%eval(&&cum&i+1);  /* line number for "Frequency" */
  %let freq_pos&i=%index(%bquote(&&line&f),Frequency);  /* column position of "Frequency" */
  %let f_var&i=%upcase(%scan(%bquote(&&line&f),1,%str( )));
  %let var_len=%length(&&f_var&i);
  %let max&i=&var_len;

  %global &&f_var&i;
  %let &&f_var&i=&i; /* for Frequency */
  %do j=&f+2 %to &&end&i;
    %let var_len=%length(%substr(%bquote(&&line&j),1,&&freq_pos&i-1));
    %if (&var_len > &&max&i) %then %let max&i=&var_len;
  %end; /* %do j= */
  %let f_line_count&i=%eval(&&end&i-&&cum&i+1);
%end; /* %do i= */

data _null_;
  %do i=1 %to &n_of_var;
    file "&home/txt/&&f_var&i...txt";  /* create frequency output for each variable */
    %do j=&&cum&i %to &&end&i;
      put "%substr(%bquote(&&line&j),1,&&max&i+5)%substr(%bquote(&&line&j),&&freq_pos&i)";
    %end;
  %end;
run;
```

1) Data length > Variable length



2) Variable length > Data length



After %prcs_freq is executed, the following txt files are created under **&home/txt** directory.

ACTG_DT.txt
COUNTRY.txt
DIVISION.txt
PRODTYPE.txt
PRODUCT_CATEGORY_CODE.txt
REACH_GOAL.txt
REGION.txt

Apply UNIVARIATE Procedure

```
proc printto print="&home/crt_univ.lst" new; run;
title;
ods select Moments BasicMeasures ExtremeObs;
proc univariate data=SGF2012.prdsale;
  var &univ_list;
run;
proc printto; run;
```

Since only three ODS output objects, **Moments**, **BasicMeasures**, and **ExtremeObs** are selected in UNIVARIATE procedure, the output for variable DIFF is created at the upper part next page (see **Table 6a**). The macro %prcs_univ then parses the contents in crt_univ.lst and creates the txt files under &home/txt directory. Each txt file contains the variable name centered at the first line (see **Table 6b**).

Table 6a

Variable: Diff

Moments

N	1080	Sum Weights	1080
Mean	11.7490741	Sum Observations	12689
Std Deviation	403.190098	Variance	162562.255
Skewness	-0.0328736	Kurtosis	-0.4944583
Uncorrected SS	175553757	Corrected SS	175404673
Coeff Variation	3431.67551	Std Error Mean	12.268684

Basic Statistical Measures

Location		Variability	
Mean	11.74907	Std Deviation	403.19010
Median	13.50000	Variance	162562
Mode	72.00000	Range	1924
		Interquartile Range	563.50000

Extreme Observations

----Lowest----		----Highest---	
Value	Obs	Value	Obs
-978	470	894	169
-971	721	898	658
-970	205	919	805
-951	902	933	954
-949	97	946	440

Table 6b

===== DIFF =====

Moments

N	1080	Sum Weights	1080
Mean	11.7490741	Sum Observations	12689
Std Deviation	403.190098	Variance	162562.255
Skewness	-0.0328736	Kurtosis	-0.4944583
Uncorrected SS	175553757	Corrected SS	175404673
Coeff Variation	3431.67551	Std Error Mean	12.268684

Basic Statistical Measures

Location		Variability	
Mean	11.74907	Std Deviation	403.19010
Median	13.50000	Variance	162562
Mode	72.00000	Range	1924
		Interquartile Range	563.50000

Extreme Observations

----Lowest----		----Highest---	
Value	Obs	Value	Obs
-978	470	894	169
-971	721	898	658
-970	205	919	805
-951	902	933	954
-949	97	946	440

After %prcs_univ is executed, the following txt files are created under **&home/txt** directory.

ACTG_DT.txt
ACTUAL.txt
COUNTRY.txt
DIFF.txt
DIVISION.txt
PREDICT.txt
PRODTYPE.txt
PRODUCT_CATEGORY_CODE.txt
REACH_GOAL.txt
REGION.txt

We have all input data so far. The next step is to integrate metadata information, FREQ outputs, and UNIVARIATE outputs into the html web page.

Create HTML file

Now we have all major outputs from FREQ and UNIVARIATE procedures. Let's see how the outputs and metadata information are added to the html template file. Let's open the html template file, **DD_template.html**, in Internet Expoorer (see **Table 7a**) and take a look at the contents. Three macro variables, **&actg_month**, **&contact**, and **&email_contact**, are embedded in the contents below.

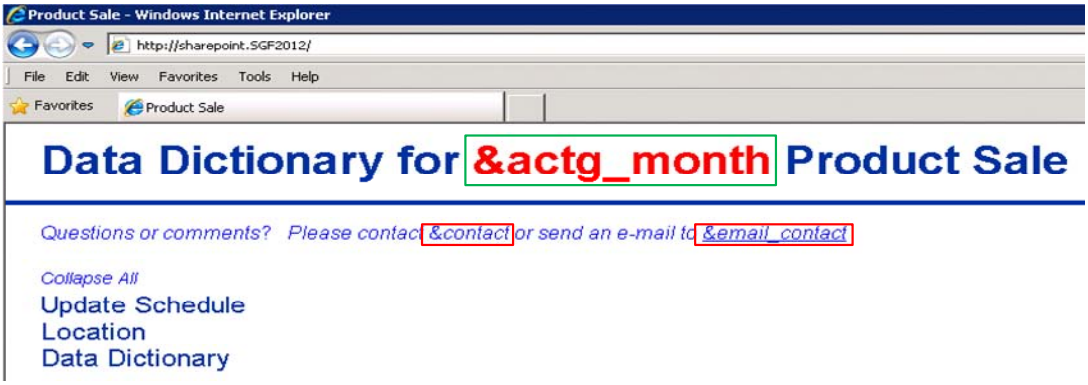


Table 7a

The data step below is to replace three macro variables with the values defined in the SAS program. We assume the SAS job is automatically triggered by job scheduling software at the 5th calendar day every month to process the past 18 month sales data. So, the actg_month is hard-coded as 201203 in this paper and the job is assumed to be executed on April 5, 2012. If this is the case, then the macro variable actg_month can be derived by the following and there is no need to change the yyyymm value every month.

```
%let actg_month=%sysfunc(intnx(month,%sysfunc(date()),-1),yymmn6.);
```

```
01  data _null_;
02    infile "&home/DD_template.html";
03    input;
04    file "&html_file";  /* &home/Product_Sale_&actg_month..html */
05
06    if index(_infile_,'&actg_month') then
07      _infile_=tranwrd(_infile_,'&actg_month',"&actg_month");
08    else if index(_infile_,'&email_contact') then do;
09      _infile_=tranwrd(_infile_,'&email_contact',"&email_contact");
10      _infile_=tranwrd(_infile_,'&contact',"&contact");
11    end;
12
13    put _infile_;
14  run;
```

Actually, the statements between 06 and 11 can be replaced with the one-line statement below.

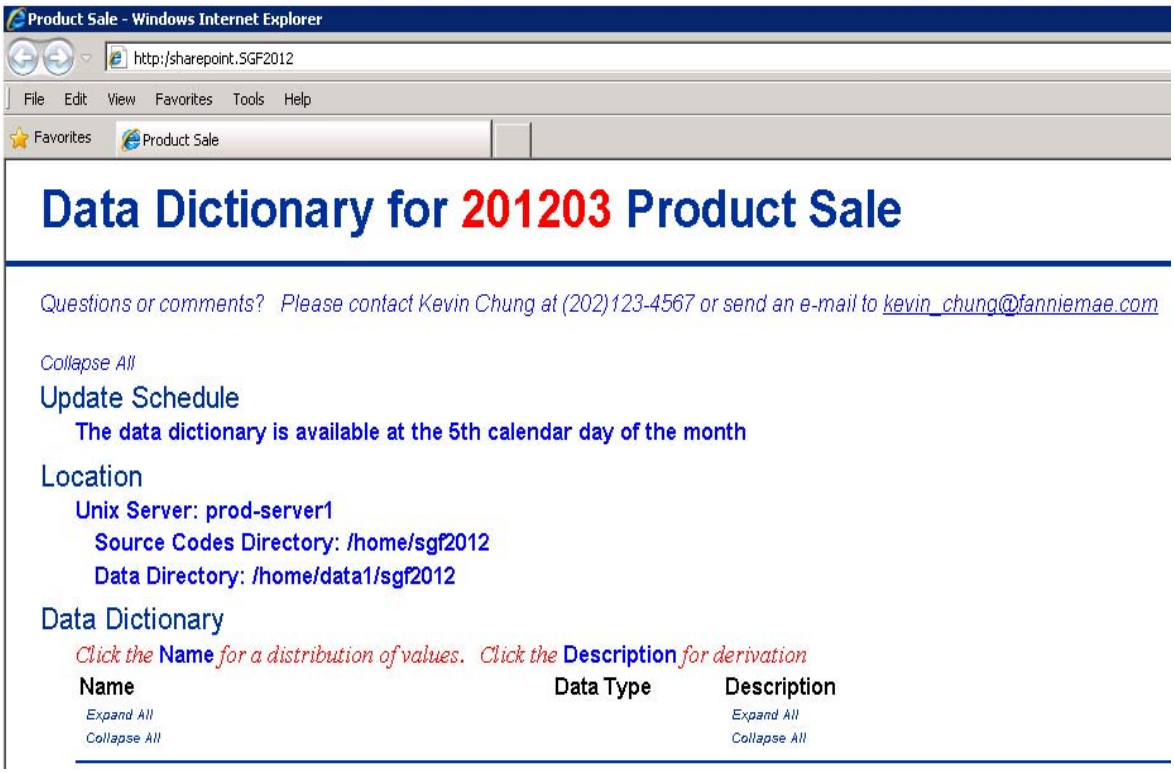
```
_infile_=RESOLVE(_infile_);
```

Why don't we use it? The purpose of RESOLVE function is to resolve the value of a text expression during DATA step execution. The reason it's not used in the code is to prevent the WARNING message from being printed in the log file as follow.

WARNING: Apparent symbolic reference NBSP not resolved.

What does NBSP mean? Why does it happen? To improve the readability, a non-breaking space, ** **, is usually used in the html file. Since   can not be resolved by RESOLVE function, the argument, ** **, is returned and the macro processor issues a warning message. But, the html file is still generated correctly. We have the following new look after the data step is executed.

Let's click on three links, **Update Schedule**, **Location**, and **Data Dictionary**, below to expand (display) the contents under each link.



The contents under **Update Schedule** and **Location** are hard-coded in the HTML template file since they are static data and look less likely to be changed every month. The only thing the SAS process needs to handle is to add the metadata information from Excel file and the frequency count / UNIVARIATE statistics from the SAS procedures. Our goal is to integrate all information for each variable and put them into the html template file at the appropriate location in an organized manner. The final appearance should look like the one in **Appendix A**.

The macro %crt_html is used to add the information and statistics mentioned above to the html template file. Refer to **Table 3** for more detailed information about the use of each field in Excel file. The following contents are the example of the field PRODUCT_CATEGORY_CODE created by %crt_html and added to the html template file.

```
    <!--                          -->   ❶
    <!--  PRODUCT_CATEGORY_CODE   -->
    <!--                          -->
    <TABLE class=folder width="100%">
      <TBODY>
      <TR>
        <TD width="28%"><A onclick="DropDown('PRODUCT_CATEGORY_CODE
summary')">PRODUCT_CATEGORY_CODE</A>  ❷
        <TD width="10%">NUMBER 8
        <TD><A onclick="DropDown('PRODUCT_CATEGORY_CODE derivation')">Product Category
Code</A></TD></TR></TBODY></TABLE>  ❸
    <TABLE style="DISPLAY: none" id="PRODUCT_CATEGORY_CODE derivation" width="100%">
❹
      <TBODY>
      <TR>
        <TD><PRE><font face="Courier New"><div style="background:#FFFFCC">  ❺
Column Name             PRODUCT_CATEGORY_CODE

Attribute Name          Product Category Code

Definition              Product Category Code

Physical Table          tbl_PRODUCT

Data Type(Length)       NUMBER(8)
         </div></font></PRE></TD>
```

```
    </TR>
    </TBODY>
  </TABLE>

  <TABLE style="DISPLAY: none" id="PRODUCT_CATEGORY_CODE summary" width="100%">  ❹
    <TBODY>
    <TR>
      <TD><PRE><font face="Courier New"><div style="background:#CCFFFF">    ❺
                                                  Cumulative    Cumulative
Product_Category_Code     Frequency     Percent     Frequency      Percent
-------------------------------------------------------------------------
1 - BED                        216       20.00           216        20.00
2 - CHAIR                      216       20.00           432        40.00
3 - DESK                       216       20.00           648        60.00
4 - SOFA                       216       20.00           864        80.00
5 - TABLE                      216       20.00          1080       100.00
          </div></font></PRE>
      </TD>
    </TR>
    </TBODY>
  </TABLE>
  <HR align=left color=#003399 SIZE=1 width="90%" noShade>
```

❶ The <!-- is the HTML comment tag.

❷ DropDown is a JavaScript function used to expand (display) or collapse (hide) the information. If the page is loaded and the link PRODUCT_CATEGORY_CODE is clicked, the frequency count is displayed. If the link for variable DIFF is clicked, then the UNIVARIATE statistics is displayed. Click the same link again; the information is collapsed (hidden).

PRODUCT_CATEGORY_CODE NUMBER 8 Product Category Code

Product_Category_Code	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1 - BED	216	20.00	216	20.00
2 - CHAIR	216	20.00	432	40.00
3 - DESK	216	20.00	648	60.00
4 - SOFA	216	20.00	864	80.00
5 - TABLE	216	20.00	1080	100.00

❸ If the link Product Category Code is clicked, the metadata information from the Excel file is displayed. The metadata is always displayed above the frequency count or UNIVARIATE statistics if they are both clicked.

PRODUCT_CATEGORY_CODE NUMBER 8 Product Category Code

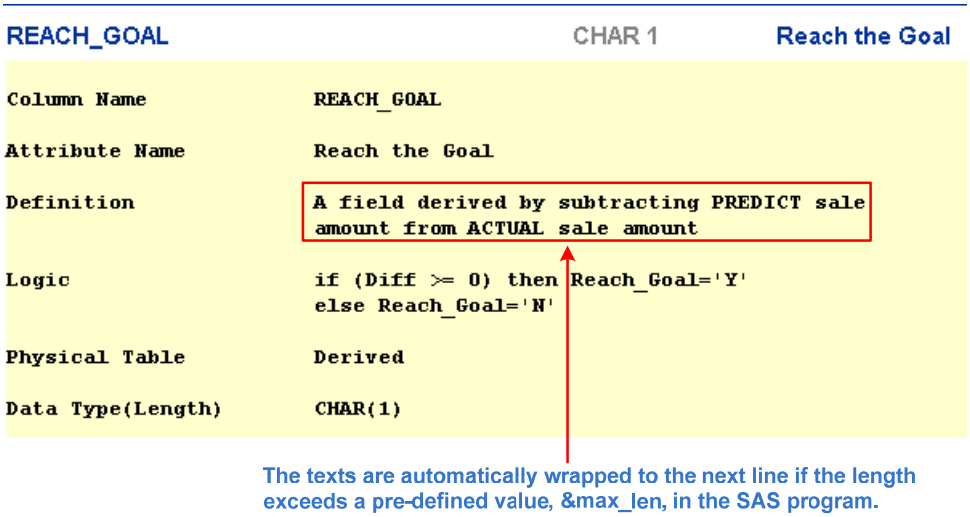
Column Name	PRODUCT_CATEGORY_CODE
Attribute Name	Product Category Code
Definition	Product Category Code
Physical Table	tbl_PRODUCT
Data Type(Length)	NUMBER(8)

❹ The attribute id in <TABLE> tag provides the unique id for the JavaScript function DropDown to perform the “expand” or “collapse” operation.

❺ The <PRE> tag defines preformatted text. Text in a <PRE> element is displayed in a fixed-width font (usually Courier), and it preserves both spaces and line breaks. The contents inside this section are either the frequency count or the UNIVARIATE statistics created in &home/txt directory. A %DO loop in macro %crt_html identifies the variable name, locates the correct file in txt directory, and writes the contents to this section.

The #FFFFCC and #CCFFFF are the color codes used to display the background color for metadata information and FREQ/UNIVARIATE outputs, respectively.

The SAS codes also provide a text wrapping feature to print the contents at the next line if the entire length exceeds a pre-defined value in the SAS program. To demonstrate this feature, the macro variable **&max_len** is defined as 45 in the SAS program to force the texts in Definition section wrapped to the next line.



After the web page is created, the last step is to email the html file, Product_Sale_201203.html, to the SharePoint.

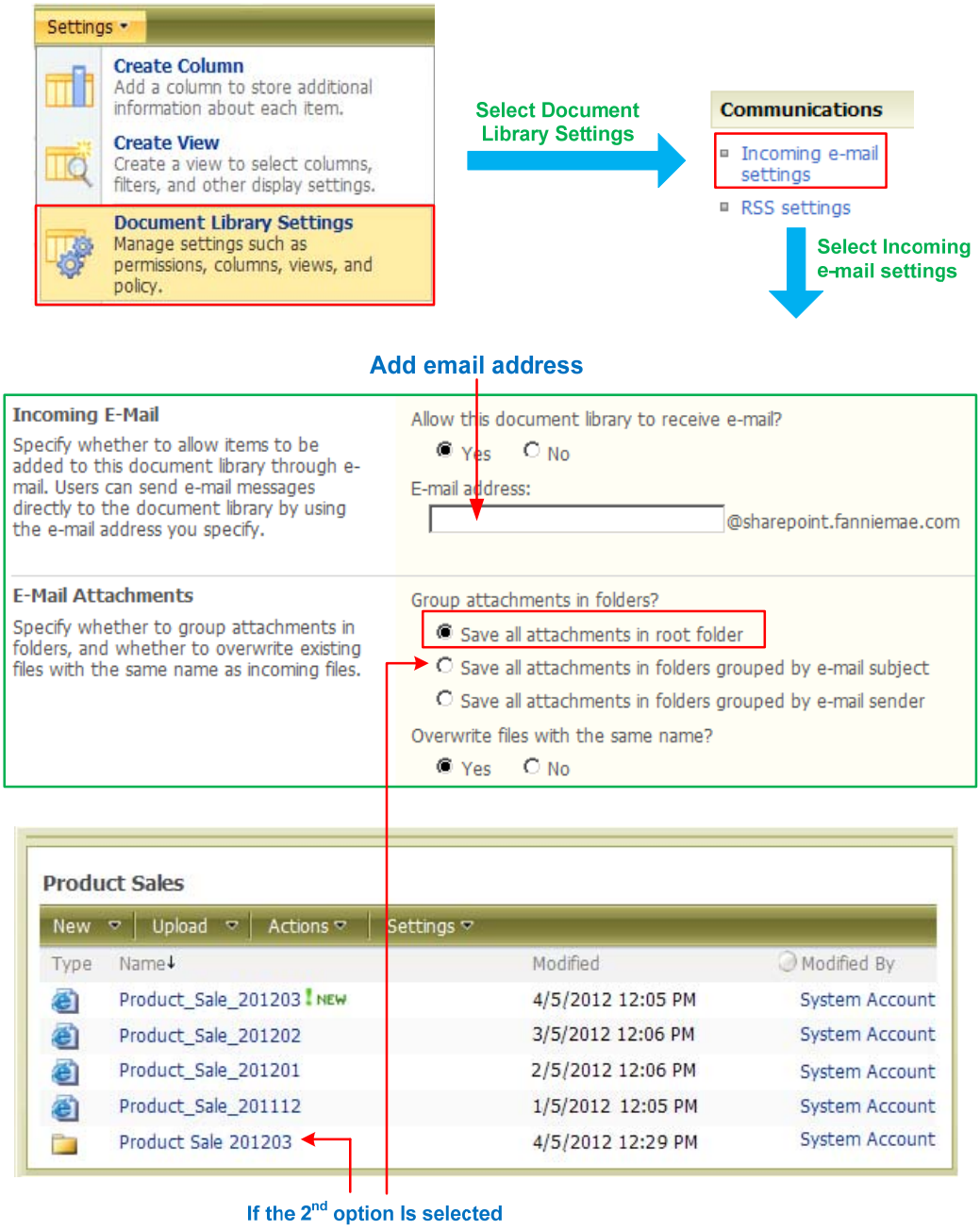
Email HTML File to Sharepoint

The following attributes should be configured correctly in SharePoint site to ensure the html file is sent to right place. Assume the document library is available with name Product. Follow the steps below to configure the email setting if it's not done yet.

- Click on **Settings** in Product document library
- Select "**Document Library Settings**"
- Select "**Incoming e-mail settings**" from **Communications** section.
- Add the email address to the box in section "**Incoming E-Mail**".
- Select the first option, **Save all attachments in root folder**, in "**E-Mail Attachments**" section.

The html file will be created in Product document library via filename with email method in SAS process.

An Automatic Approach to Create Data Dictionary in SharePoint Using SAS® continued



If the second option in "**E-Mail Attachments**" is selected, the html file is created under a directory and the directory name is the same as &subject specified in the filename statement as follow.

```
filename out email to="&addr" subject="&subject" attach="&html_file";
data _null_;
  file out;
run;
```

Appendix A

When html page is loaded

Data Dictionary for 201203 Product Sale

Questions or comments? Please contact Kevin Chung at (202)123-4567 or send an e-mail to kevin_chung@fanniemae.com.

Collapse All

Update Schedule
Location
Data Dictionary

When each section header is clicked

Data Dictionary for 201203 Product Sale

Questions or comments? Please contact Kevin Chung at (202)123-4567 or send an e-mail to kevin_chung@fanniemae.com.

Collapse All

Update Schedule

The data dictionary is available at the 5th calendar day of the month

Location

Unix Server: prod-server1

Source Codes Directory: /home/sgf2012

Data Directory: /home/data1/sgf2012

Data Dictionary

Click the **Name** *for a distribution of values. Click the* **Description** *for derivation rules and other documentation*

Name *Expand All* *Collapse All*	Data Type	Description *Expand All* *Collapse All*
ACTG_DT	NUMBER 8	Accounting Period
ACTUAL	NUMBER 8	Actual Sale Amount
COUNTRY	CHAR 10	Country
DIFF	NUMBER 8	Difference
DIVISION	CHAR 10	Division
PREDICT	NUMBER 8	Predict Sale Amount
PRODTYPE	CHAR 10	Product Type
PRODUCT_CATEGORY_CODE	NUMBER 8	Product Category Code
REACH_GOAL	CHAR 1	Reach the Goal
REGION	CHAR 10	Region

Appendix B

When Name and Descrption are clicked on fields PRODUCT_CATEGORY_CODE and REACH_GOAL

PRODUCT_CATEGORY_CODE NUMBER 8 Product Category Code

Column Name	PRODUCT_CATEGORY_CODE
Attribute Name	Product Category Code
Definition	Product Category Code
Physical Table	tbl_PRODUCT
Data Type(Length)	NUMBER(8)

Product_Category_Code	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1 - BED	216	20.00	216	20.00
2 - CHAIR	216	20.00	432	40.00
3 - DESK	216	20.00	648	60.00
4 - SOFA	216	20.00	864	80.00
5 - TABLE	216	20.00	1080	100.00

REACH_GOAL CHAR 1 Reach the Goal

Column Name	REACH_GOAL
Attribute Name	Reach the Goal
Definition	A field derived by subtracting PREDICT sale amount from ACTUAL sale amount
Logic	if (Diff >= 0) then Reach_Goal='Y' else Reach_Goal='N'
Physical Table	Derived
Data Type(Length)	CHAR(1)

Reach_Goal	Frequency	Percent	Cumulative Frequency	Cumulative Percent
N	531	49.17	531	49.17
Y	549	50.83	1080	100.00

CONCLUSION

We can take the advantage of the powerful SAS data processing capability as well as the advantage of the easy-to-use SharePoint web feature to share information with users. If you want to convert the process to fit your need, remember to update the Excel file, change the macro variable values at the top of the SAS program prdsale.sas. Give it a try and enjoy the elegant and nice-looking web page you create.

REFERENCES

[1] SAS OnlineDoc® 9.2, SAS Institute Inc. Cary, NC.
http://support.sas.com/documentation/cdl/en/lrdict/64316/PDF/default/lrdict.pdf

[2] The HTML 4 Reference
http://www.w3schools.com/tags/

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Feel free to contact the author at:

Kevin Chung
Fannie Mae
4000 Wisconsin Ave., NW
Mail Stop: 2H-4S/07
Washington, DC 20016
Work Phone: 202-752-1568
E-mail: kevin_chung@fanniemae.com
kchung01@hotmail.com