Paper RF13-2015

Building PROC FORMAT Code from Data Dictionary Automatically

Ran Gu, Nebraska DHHS /UNL, Lincoln, NE
Ashley Newmyer, Nebraska Department of Health and Human Service, Lincoln, NE

ABSTRACT

In Public Health, software data collection systems gather information according to a data dictionary or data coding manual. A data dictionary provides tables for mapping a valid numeric input to a corresponding descriptive syntax, but typically the software will only record the numeric value into the dataset, not the descriptive syntax. In this case, the format procedure was used to convert the numbers back into words during data analysis. Specifically PROC Format was used to map the numerical input back to the descriptive syntax for each data element. The PROC Format needs to comply according to the same data dictionary. However writing PROC FORMAT code itself would be a tedious task, especially when the dataset contains many numeric fields to convert back to words. We present a method in which SAS ® generates PROC Format code automatically for all of these variables at once based on the data dictionary table. The advantage of this methods is not only in reducing the time and effort of editing the PROC FORMAT code for many variables, but also avoiding human error. Applying this method facilitates data quality checking, segmentation analysis and data integration of data in different formats.

INTRODUCTION

According to Wikipedia, a data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them. Due to the complexity and consistency requirement of public health datasets, data dictionaries are widely used in medical or health care records. In addition, public health data are large in both dimension and number of records. Therefore, electronic records in public health usually convert a word description of patient disposition or treatment procedure into a numeric number via a data dictionary for the purpose of storage. To avoid reading errors, each variable have different ranges of numeric values. For the purpose of checking data quality and segmentation analysis, we need to find out the invalid/missing input. A combination of PROC FORMAT and PUT statement can help us with this task. Furthermore, this combination can also facilitate further data analysis.

PREPARATION OF DATA DICTIONARY

Data dictionaries can be viewed as a table which contains at least 3 columns: a list of all variable names in the dictionary, word descriptions of each the variable, and corresponding numeric coded for the word value description. If the data is a combination of multiple sources, it will also show the original data source. Table 1 shows part of the data dictionary required to generate the PROC FORMAT code. Then we can read in the table into SAS® as a data set.

Table 1. Part of Sample Data Dictionary and 3 Key Elements in Generating PROC FORMAT Code

Variable	Code	Format
Gender	650	Male
Gender	655	Female
Gender	-20	Not Recorded
Ethnicity	695	Not Hispanic or Latino
Ethnicity	690	Hispanic or Latino
Ethnicity	-20	Not Recorded
Possible Injury	0	No
Possible Injury	1	Yes
Transport Mode From Scene	4955	Initial Lights and Sirens, Downgraded to No Lights or Sirens
Transport Mode From Scene	4960	Initial No Lights or Sirens, Upgraded to Lights and Sirens
Transport Mode From Scene	4965	Lights and Sirens
Transport Mode From Scene	808001	Lights Only - No Sirens
Transport Mode From Scene	4970	No Lights or Sirens
Transport Mode From Scene	-20	Not Recorded

PROCESS DATA DICTONARY INTO A NEW DATASET

The basic idea is to generate a new dataset from the dictionary dataset via .FIRST and .LAST statement in DATA Step.

1. Sort the data dictionary by variable name.
2. Combine the code and the format of each entry with an equal sign within the new dataset as one data element.

   For example, the 2nd observation of the dataset is "'Gender' '655' 'Female' ". The new dataset the second observation turns into " 655='Female' " as one data element.
3. Append "VALUE" and variable name before every first observation of each variable.

   For example, the first observation of Gender in original dataset is "'Gender' '650' 'Male' ", and in the new dataset, this entry turns into "value Gender 650='Male' ".

   It is the same with first observation of other variables, eg "'Ethnicity' '695' 'Not Hispanic or Latino'" turns into "value Ethnicity 695='NotHispanicorLatino' "
4. Append "other = 'Invalid' " after the last observation of each variable.

   For example, the last observation of Gender in original dataset is "'Gender' '-20' 'Not Recorded' ", and in the new dataset, this entry turns into " -20='NotRecorded' other='Invalid'; " .The methods can be used to label all data input out of the allowed range as "Invalid".
5. Using ODS to output the SAS dataset into an EXCEL file.

   The final EXCEL file will only have one column but contains all the information needed to generate PROC FORMAT code. Table 2 provides anexample of the final EXCEL file which corresponds to Table 1 (Excluding the column name).

Table 2. Final Excel Table Generated from Data Dictionary

```
DATA import;
set dictionary;
by variable;
variable=compress(strip(variable));
format=compress(format);
input=strip(code);
if first.variable=1 then prefix="value "||variable;
if last.variable=1 then suffix="  other='Invalid';";
content=right(strip(prefix)||"
"||code||"='"||compress(format)||"'"||suffix);
keep content;
 run;

  ods csv file='C:\Users\rgu\Desktop\dictionary2.csv';
 proc print data=import2 noobs;
var content;
run;
ods csv close;
```

value Gender 650='Male'
655='Female'
-20='NotRecorded' other='Invalid';
value Ethnicity 695='NotHispanicorLatino'
690='HispanicorLatino'
-20='NotRecorded' other='Invalid';
value PossibleInjury 0='No'
1='Yes' other='Invalid';
value TransportModeFromScene 4955='InitialLightsandSirens,DowngradedtoNoLightsorSirens'
4960='InitialNoLightsorSirens,UpgradedtoLightsandSirens'
4965='LightsandSirens'
808001='LightsOnly-NoSirens'
4970='NoLightsorSirens'
-20='NotRecorded' other='Invalid';

COPY AND PASTE THE CONTENT OF EXCEL FILE INTO SAS PROGRAM EDITOR

In SAS Editor, simply copy and paste the content of the EXCEL file between the head and end of PROC FORMAT in SAS Editor. It will return the PROC FORMAT code automatically.

Figure 1. Final SAS PROC FORMAT

```
proc format;
*copy and past from dictionary2.csv;
value Gender          650='Male'
655='Female'
-20='NotRecorded'  other='Invalid';
value Ethnicity          695='NotHispanicorLatino'
690='HispanicorLatino'
-20='NotRecorded'  other='Invalid';
value PossibleInjury            0='No'
1='Yes'  other='Invalid';
value TransportModeFromScene         4955='InitialLightsandSirens,DowngradedtoNoLightsorSirens'
4960='InitialNoLightsorSirens,UpgradedtoLightsandSirens'
4965='LightsandSirens'
808001='LightsOnly-NoSirens'
4970='NoLightsorSirens'
-20='NotRecorded'  other='Invalid';
run;
```

CONCLUSION

Having the computer generate an executable code by itself is a brand new idea in information technology. In public health, the data dictionary usually contains hundreds of variables and each variable has several classes. PROC format combined with a PUT statement can be used to 1) filter all invalid input from other input, and 2) turn numeric code into its word description. However, it is a tedious job for the SAS programmer to input all of the necessary variables into PROC FORMAT. This method will save time in editing PROC FORMAT code and avoid any human error.

CONTACT INFORMATION

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

Name: Ran Gu
Enterprise: Nebraska Department of Health and Human Service/ University of Nebraska-Lincoln
Address: 3rd Floor, 301 Centennial Mall,
City, State ZIP: Lincoln, NE 68508
Work Phone: (402) 471-4613
E-mail: ran.gu@nebraska.gov