PhUSE 2011

Paper DH05

Comparing dataset metadata

Jim Groeneveld, OCS Consulting, 's Hertogenbosch, Netherlands

SUMMARY

Similarities and differences of structure and data between two datasets at a time can be studied using PROC COMPARE. A SAS® macro CrossRef has been developed that compares the **structure** of more datasets at once. It may be useful with similar datasets, i.e. which have approximately the same structure, variables and variable attributes. The information is output as a metadata dataset, a rectangular table with the original dataset names as variables/columns and meta information as records/rows. The metadata presented are library names, dataset names and labels, creation dates, change dates, number of variables, number of records and all variable attributes, like names and labels and optionally types and lengths, format and informats. In the future the information can be extended to contain descriptive statistics. From the resulting tabular dataset it can easily be seen which dataset structures are similar and which variables they have in common. The user may report the cross reference dataset as desired, e.g. with PROC REPORT.

INTRODUCTION

Suppose one has obtained several SAS datasets that are quite similar, i.e. which have approximately the same structure, variables and variable attributes and whatever data (same or different). If one wants to see the similarities and differences between the various dataset structures one can compare two datasets at a time with PROC COMPARE, but doing that for all dataset pairs is quite cumbersome and not very clear. Here a SAS macro CrossRef is described that shows the dataset's structure, the most important metadata of all datasets and their variables, at once in a convenient way in a cross reference dataset as a rectangular table.

The cross reference dataset described here is different from a straightforward inventory of dataset metadata that can be obtained by concatenating dataset information from PROC CONTENTS or PROC DATASETS (CONTENTS statement), especially from the metadata variables LibName, MemName, Name, Label, Type, Length, Format, FormatL, FormatD, Informat, InformL and InformD, though these variables are used to generate the cross reference as well.

CROSSREF

In that cross reference dataset the original dataset names are the variables (columns) and the meta information is stored in the records (rows). The metadata in the cross reference table consists of library names, dataset names and labels, number of variables, (physical) number of records (observations), creation dates, change dates and all variable attributes, like names and labels and optionally types and lengths, formats and informats. There is a record for each attribute except for the variable labels which are contained in the (common) variable name record. From the information in the resulting cross reference dataset (which actually is a table) it can easily be seen which dataset structures are similar and which variables they have in common. The data in the datasets are not shown, neither are the differences between datasets (like they are with PROC COMPARE).

Partial examples of the resulting cross reference dataset with a rectangular table layout are shown in tables 3 and 4 (below). It can easily be seen where those compared datasets overlap each other and which datasets could be combined into one (or more smaller) datasets by classification variables. Filled cells in variable name rows indicate the variable labels; empty cells in variable rows indicate the absence of the concerning variables; the text "-no label-" indicates the presence of the variable, but the absence of a related variable label. The data are not included in the comparison, that remains something to investigate using e.g. PROC COMPARE (but certain aggregated statistics of the data might be; this may be a future feature as indicated below).

HISTORY

A first attempt to perform an investigation on common variables in different datasets was realised in 2005 with the macro %VarLists. It produced a simple cross tabulation of variable names with various dataset names and called a macro %Crosstab to actually generate and present the cross table from concatenated PROC CONTENTS OUT=dataset output. Partial output was as represented in table 1 (below).

Next, attempts to improve the quality and amount of information were undertaken in 2009. These resulted in a prototype macro %MetaData that actually created a cross reference table by a kind of cross tabulation of extracted dataset attributes (from PROC CONTENTS) and variables with dataset names. Instead of frequencies in the table cells the dataset's attributes were included as well as the labels of the variables, if any. Partial output was as represented in table 2 (below).

USE

Macro %CrossRef has evolved from the code of macro %MetaData. %MetaData only could generate attribute data of one dataset at a time and should be called for every dataset to compare, while the user had to concatenate the results for all datasets and post-process (PROC TRANSPOSE) them before (s)he had the desired cross reference dataset. %CrossRef does all that by itself and can be called with a list of dataset names to be compared as one of its argument values.

Another required argument value for the macro %CrossRef is the name of the reference dataset to be generated. Other optional parameters are those to request optional attribute information on the variable's type, length, format and informat. (There are some further optional parameters that the average user generally does not need to care about; these involve several standard parameters that are supplied with most of my macros and that mainly are used during development.)

ALGORITHM

The macro %CrossRef is based on the output created by PROC CONTENTS for each specified dataset. It gathers information on dataset attributes, like the dataset label (if any), the number of variables, the number of physical observations (including possible deleted observations still in there), and the date and time of creation and last modification. For every variable it gathers its name, label (if any), type, length, format and informat. These metadata for all specified datasets are combined in one rectangular table (dataset) with dataset names as variables (columns) and attributes as observations (records, rows). Variable names and their optional labels in the datasets are in one record, while the other attributes take separate subsequent records each (though Type and Length are combined into one cell of information in the same record). All fields of the reference dataset are of character type (length $256, except for the leftmost attribute specification column, which has a length of $36), even if containing numeric information; that is because other records may have explicit character information for the concerning variable (= dataset name).

Technically, dataset and variable attribute information is obtained from the PROC CONTENTS output per dataset. The attribute information is rotated (transposed) and processed per dataset attribute and per variable per attribute, appending all attribute information to the right of each other. Subsequently all attribute information from all datasets is appended below each other in subsequent records and transposed back with same named variables in same records and their attributes right below them. In more detail variable attributes initially get names that are deduced from the variable names.

Internally, the macro sets OPTIONS VALIDVARNAME=ANY to allow for SAS name literal variable names that it explicitly uses. For that reason it is assumed that the user's datasets do NOT contain variables with SAS name literal names that start with an asterisk (*) or that end with an exclamation character (!) followed by one or more digits. (These are replaced by other text values later in the macro when transposed to variable values.) User's variable names also are assumed not to be longer than 30 characters; this restriction will be relieved in a future version. The resulting cross reference dataset, though, does not contain SAS name literal variable names, unless such DBMS table names are specified by the user. Furthermore the macro internally uses specific WORK datasets which names start with two subsequent underscore characters. It is assumed that these do not interfere with user's WORK datasets. The macro code shows which (thus reserved) names are being applied.

FUTURE

Future features of the macro CrossRef are:

- comparing all datasets in one or more libraries using a wildcard (LibName.*) as the DataList specification. On the other hand the resulting cross reference dataset / table will become rather wide. The macro will generally be used to compare a limited number of datasets;
- adding optional univariate descriptive statistics per variable, mainly for numerical variables only, like
  - (non-deleted) logical number of observations (in contrast to the number of physical observations in the dataset). This would be a dataset attribute, not one per variable;
  - number of non-missing values (character variables as well);
  - number of missing values (character variables as well);
  - minimum values (character variables as well: first non-missing, sorted value, limited in length);
  - maximum values (character variables as well: last non-missing, sorted value, limited in length);
  - mean values;
  - median values (character variables as well: (approximately) middle, non-missing, sorted value, limited in length);
  - standard deviations;
  - percentiles;
  - and possibly more (e.g. distribution information).

  These would be included below the variable names/labels, just like the variable attributes, yet presented as character values too.

REFERENCES

SAS Institute Inc. 2009.Base SAS® 9.2 Procedures Guide. Cary, NC: SAS Institute Inc.
SAS Institute Inc. 2010. SAS ® 9.2 Language Reference: Dictionary, Third Edition . Cary, NC: SAS Institute Inc.

CONTACT INFORMATION

Author Name :	Y. (Jim) Groeneveld	Work Phone :	+31 (0)73 523 6000
Company :	OCS Consulting	Fax :	+31 (0)73 523 6600
Address :	P.O. Box 3434	Email :	SASquestions@ocs-consulting.com
	5203 DK 's-Hertogenbosch	Web :	http://www.ocs-consulting.com/nl
	The Netherlands	Author's website :	http://jim.groeneveld.eu.tf

TABLES

Table 1. Partial cross tabulation from macro %VarLists (vs. 0.0.0)

	Library Member Name			
	CUMULAT IVE_AC	INTRIAL _IMAGES	RECON_2 0100128	TOTAL
Variable Name	Present number	Present number	Present number	Present number
ROOM_TEMPERATURE	1		1	2
SCANTIME	1			1
SCAN_DATE	1	1	1	3
SCAN_TIME	1	1	1	3
SCAN_TYPE	1		1	2
SCAN_VISIT		1		1
.				
.				
TOTAL	48	18	43	109

Table 2. Partial cross reference from the prototype macro %MetaData (vs. 0.5)

_Dataset	TrialDCM1	Blinded2	Icon_Data	Intrial_Images
_Dataset	TrialDCM1	Blinded2	Icon_Data	Intrial_Images
_DsLabel	-no label-	-no label-	variable labels added	-no label-
_Columns	72	9	4	30
_Records	1987	8996	476	1586
_Created	08JAN2010:11:39:22.78	07JAN2010:16:54:01.86	01SEP2009:07:16:30	09SEP2009:10:22:17.98
_Changed	08JAN2010:11:39:22.78	07JAN2010:16:54:01.86	01SEP2009:07:16:30	09SEP2009:10:22:17.98
batch	-no label-	-no label-		
Blinded	Blinding date and time	Blinding date and time		
BlindID2	2nd Random 4 digit code	2nd Random 4 digit code		
Blind_ID	Random 4 digit code as	Random 4 digit code as		
Blind_PT	3rd Random 4 digit code	3rd Random 4 digit code		
dob	-no label-		subject date of birth	
ICDdate	Instance Creation Date			
ICDTIME	Instance Creation Time			
patid	patid	patid	patid	patid
reader	-no label-			
site	site			site

Table 3. Partial cross reference with dataset attributes from macro %CrossRef (vs. 0.6.2)

columns=datasets, rows=variables and attributes (Structure)	cumulative_ac (cumulative_ac)	recon_20100128 (recon_20100128)	Intrial_Images (Intrial_Images)
*LibName	Datalib	Datalib	Datalib
*Dataset	cumulative_ac	recon_20100128	Intrial_Images
*DsLabel	-no label-	-no label-	-no label-
*NofVars	48	43	18
*NofObs	1706	1978	46
*Created	11JAN2010:13:41:59.78	28JAN2010:10:53:05.08	22SEP2008:14:15:54.36
*Changed	11JAN2010:13:41:59.78	28JAN2010:10:53:05.08	22SEP2008:14:15:54.36

Table 4. Partial cross reference with variable attributes from macro %CrossRef (vs. 0.6.2)

columns=datasets, rows=variables and attributes (Structure)	cumulative_ac (cumulative_ac)	recon_20100128 (recon_20100128)	Intrial_Images (Intrial_Images)
ROOM_TEMPERATURE	Room_Temperature	Room_Temperature	
>Type/Length	Numeric/8	Numeric/8	
>Informat			
>Format			
SCANTIME	-no label-		
>Type/Length	Numeric/8		
>Informat			
>Format			
SCAN_DATE	scan_date	scan_date	scan_date
>Type/Length	Numeric/8	Numeric/8	Numeric/8
>Informat	DATE9.0	DATE9.0	DATE9.0
>Format	DATE9.0	DATE9.0	DATE9.0
SCAN_TIME	scan_time	scan_time	scan_time
>Type/Length	Char/5	Char/5	Numeric/8
>Informat	$5.0	$5.0	TIME8.0
>Format	$5.0	$5.0	TIME8.0
SCAN_TYPE	-no label-	-no label-	
>Type/Length	Char/11	Char/11	
>Informat			
>Format			
SCAN_VISIT			scan_visit
>Type/Length			Char/2
>Informat			$2.0
>Format			$2.0

CODE

The **partial** %CrossRef version 0.6.2 macro code is included below. **Full** and **future** versions can be downloaded from: http://jim.groeneveld.eu.tf/software/SASmacro/CrossRef.zip or http://jim.groeneveld.eu.tf/crossref

```
*************************************************************************;
** SAS macro Crossref vs. 0.6.2 ,  9 Aug 2011, by Jim Groeneveld, Netherlands *;
*************************************************************************;
** STILL UNDER CONSTRUCTION, NOT FOR VALIDATION, USE WITH CAUTION AT OWN RISK *;
**                                                                       *;
*; %MACRO   Crossref   /* cross reference of multiple datasets           */
/* ------------------------------------------------------------------- */
/* ---------- A l w a y s   s p e c i f i e d   a r g u m e n t s ---------- */
/* Argument Default       Description and remarks     Required/Optional: R/O */
/* -------- --------      ------------------------------------------------ */
(  DataList= _LAST_    /* input dataset or list of library.dataset names    R */
,  Crossref=           /* output dataset, the resulting cross reference     R */
,  OverWrit= No        /* Yes: overwrite existing CrossRef output dataset   R */
/*                                                                         */
/* ------ F r e q u e n t l y   s p e c i f i e d   a r g u m e n t s ------- */
/* Argument Default       Description and remarks     Required/Optional: R/O */
/* -------- --------      ------------------------------------------------ */
,  TypeLen =           /* not empty: include Type and Length attribute info O */
,  Informat=           /* not empty: include Informat atribute information  O */
,  Format  =           /* not empty: include Format   atribute information  O */
/*                                                                         */
/* ---------- R a r e l y   s p e c i f i e d   a r g u m e n t s ---------- */
/* Argument Default       Description and remarks     Required/Optional: R/O */
/* -------- --------      ------------------------------------------------ */
,  Version = 0.6.2     /* forced version specification for version control  O */
)/ DES     =             'cross reference of multiple datasets' /*Store*/     ;
%*                                                                         *;
%* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*;
%*                                                                         *;
%* Purpose    : cross reference of data structure of multiple datasets     *;
%*              by common variable name of existing variable in datasets   *;
%*              involving various dataset attributes and variable labels.  *;
%*              Optionally also variable type, length, informat and format.*;
%*                                                                         *;
%* Programmer : Jim Groeneveld                                             *;
%* Date       :  9 Aug 2011                                                *;
%* Version    : 0.6.2 ½ THIS VERSION IN THE PAPER IS RATHER MUCH TRUNCATED ╗  *;
%*                                                                         *;
%* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*;
%*                                                                         *;
%* Used auxiliary macros:                                                  *;
%* ----------------------                                                  *;
%* = internal: -                                                           *;
%* = attached: -                                                           *;
%* = external: %UstrMcmp, %ERR0R, %_Delete_                                *;
%*                                                                         *;
%*************************************************************************;

%* OPTIONS MPRINT MERROR SERROR MLOGIC SYMBOLGEN MACROGEN;

 %LOCAL    /* Define internal macro variables explicitely local */
  MacName  /* This macros external name: Crossref */
  MacVs    /* Version number major.minor.patchSub of macro: 0.6.2 */
  MacDate  /* Date of this version of macro:  9 Aug 2011 */
  ErrCount /* number of erroneously specified arguments */
  ValidVar /* stored VALIDVARNAME option */
  Contents /* output dataset of PROC CONTENTS */
  AttrData /* dataset with variable attributes: Type, Length, Informat, Format */
  NameData /* transposed attributes dataset */
  TypeLen1 /* transposed dataset with Type and Length attributes */
  Inform2  /* transposed dataset with Informat attributes */
  Format3  /* transposed dataset with Format attributes */
  Appended /* (horizontally structured) dataset to append metadata to */
  Sorted   /* sorted (partial) results to force desired order */
```

```
  I        /* incremental counter for various purposes */
  DataFile /* Dataset element of DataList argument: libname.dataset */
  ;

  %LET MacName  = Crossref;
  %LET MacVs    = 0.6.2 ;
  %LET MacDate  =  9 Aug 2011;
  %LET ValidVar = %SYSFUNC(GETOPTION(VALIDVARNAME));
  %LET Contents = __ContDt;
  %LET AttrData = __AttrDt;
  %LET NameData = __NameDt;
  %LET TypeLen1 = __TypLen;
  %LET Inform2  = __Inform;
  %LET Format3  = __Format;
  %LET Appended = __Append;
  %LET Sorted   = __Sorted; %* Might also be one of the other names;

  %*~~~~~~~~~~~~~~~~~~~~~~~~~;
%* Macro header in LOG file;
%*_________________________;

  %PUT;
  %PUT |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~|;
  %PUT |   Macro &MacName, vs. &MacVs, by Jim Groeneveld, &MacDate  |;
  %PUT |    Cross reference of data structure of multiple datasets    |;
  %PUT |______________________________________________________________|;
  %PUT;

%IF (&Version NE AND &Version NE &MacVs) %THEN
  %DO;
    %PUT *** &MacName *** %ERR0R-: Specified version &Version does not match &MacVs,;
    %PUT .                       macro &MacName will abort;
    %GOTO Exit;
  %END;

%* Argument checks;

%* DataList argument;
  %IF (NOT %LENGTH(&DataList)) %THEN %DO;
    %PUT *** &MacName *** %ERR0R-: DataList not specified, macro &MacName will abort;
    %LET ErrCount = %EVAL ( &ErrCount + 1);
  %END;

  %IF ( NOT %UstrMcmp(&Overwrit,Yes) AND NOT %UstrMcmp(&Overwrit,No) ) %THEN %DO;
    %PUT *** &MacName *** %ERR0R-: Overwrit argument not specified or illegal: &Overwrit,;
    %PUT .                      macro &MacName will abort;
    %LET ErrCount = %EVAL ( &ErrCount + 1);
  %END;

  %IF ( %SYSFUNC(EXIST(&CrossRef)) AND NOT %UstrMcmp(&OverWrit,Yes) ) %THEN %DO;
    %PUT *** &MacName *** %ERR0R-: Dataset CrossRef=&CrossRef exists and OverWrit is not 
Yes,;
    %PUT .                     macro &MacName will abort;
    %LET ErrCount = %EVAL ( &ErrCount + 1);
  %END;

  %IF ( &ErrCount GT 0 ) %THEN %GOTO Aborting;

%* - - - - - - - - - - - - ;
%* Core code starts here ;
%* - - - - - - - - - - - - ;

  OPTIONS VALIDVARNAME=ANY;
```

```
/*PROC DATASETS LIBRARY=WORK; DELETE &Appended; RUN; * delete existing, old result;*/
  DATA &Appended; STOP; RUN; * empty intermediate result dataset as a start ;

  %LET I = 1;
  %LET DataFile = %SCAN ( &DataList, &I, %STR( ) );
  %DO %WHILE ( %LENGTH ( &DataFile ) );

%* Process keyword _LAST_ with DataFile;
    %IF (%UPCASE(&DataFile) EQ _LAST_) %THEN %LET Data = &SYSLAST;

    %IF (NOT %SYSFUNC(EXIST(&DataFile))) %THEN %DO;
      %PUT *** &MacName *** %ERR0R-: Dataset &DataFile does not exist,;
      %PUT .                       macro &MacName will abort;
      %GOTO Aborting;
    %END;

    %LOCAL Dataset;
    %LET Dataset = %SCAN ( &DataFile, -1, .);

    PROC CONTENTS DATA=&DataFile NOPRINT
                  OUT=&Contents /*(KEEP=MemLabel Name Label CrDate MoDate)*/;
    RUN;

    DATA &AttrData (KEEP=Name Label Name1 TypeLen Name2 Infmt Name3 Fmt);
      SET &Contents (KEEP=Name Label Type Length
                          Informat InformL InformD Format FormatL FormatD);
      LENGTH Name1 $32 TypeLen Infmt Fmt $16;
      Name = UPCASE(Name);
* Label;
      IF (MISSING(Label)) THEN Label = '-no label-';
* Type and Length;
      IF (Type EQ 1) THEN TypeLen = 'Numeric/8';
      ELSE TypeLen = 'Char/' || LEFT(PUT (Length, BEST.));
      Name1 = TRIM(Name) || '!1';
* Informat and Format;
      IF (InformL IN (0 .)) THEN DO; Infmt = ''; Fmt = ''; END;
      ELSE DO;
        InFmt = TRIM(InFormat) || TRIM(LEFT(PUT(InFormL,BEST.))) || '.' ||
TRIM(LEFT(PUT(InFormD,BEST.)));
         Fmt = TRIM(  Format) || TRIM(LEFT(PUT(FormatL,BEST.))) || '.' ||
TRIM(LEFT(PUT(FormatD,BEST.)));
      END;
      Name2 = TRIM(Name) || '!2'; * Informat;
      Name3 = TRIM(Name) || '!3'; * Format;
    RUN;

* Variable Label;
    PROC TRANSPOSE DATA=&AttrData OUT=&NameData;
      ID Name; VAR Label;
    RUN;

* Type and Length;
    PROC TRANSPOSE DATA=&AttrData OUT=&TypeLen1;
      ID Name1; VAR TypeLen;
    RUN;

* Informat;
    PROC TRANSPOSE DATA=&AttrData OUT=&Inform2;
      ID Name2; VAR Infmt;
    RUN;

* Format;
    PROC TRANSPOSE DATA=&AttrData OUT=&Format3;
      ID Name3; VAR Fmt;
```

```
    RUN;

    DATA &NameData (DROP=MemLabel CrDate MoDate);
      LENGTH '*LibName'n $8 '*Dataset'n $ 32 '*DsLabel'n $ 256 '*NofVars'n '*NofObs'n $ 16
                        '*Created'n '*Changed'n $ 40;
      IF 0 THEN SET &DataFile (DROP=_ALL_) NOBS=Nobs; * to obtain number of observations;
* To obtain Date & Time of last modification and number of variables in dataset;
      SET &Contents (KEEP=MemLabel CrDate MoDate) NOBS=Nvars;
      SET &NameData (DROP=_NAME_);
      %IF (%LENGTH(&TypeLen )) %THEN SET &TypeLen1 (DROP=_NAME_);;
      %IF (%LENGTH(&Informat)) %THEN SET &Inform2 (DROP=_NAME_);;
      %IF (%LENGTH(&Format  )) %THEN SET &Format3 (DROP=_NAME_);;
      %IF (%LENGTH ( %SCAN ( &DataFile, 2, . ))) %THEN %DO;
            '*LibName'n = "%SCAN ( &DataFile, 1, . )";
      %END;
      %ELSE '*LibName'n = "WORK";;
      '*Dataset'n = "&Dataset";
      '*DsLabel'n = MemLabel; IF (MISSING('*DsLabel'n)) THEN '*DsLabel'n = '-no label-';
      '*Created'n = PUT (CrDate, DATETIME40.2);
      '*Changed'n = PUT (MoDate, DATETIME40.2);
      '*NofVars'n = PUT (Nvars, F16.);
      '*NofObs'n  = PUT (NOBS, F16.);
      OUTPUT; STOP; * Just one record! ;
    RUN;

    DATA &Appended; SET &Appended &NameData; RUN; * append, check equal vars+lengths;

    %_Delete_ (&Contents &AttrData &NameData &TypeLen1 &Inform2 &Format3);

    %LET I = %EVAL ( &I + 1 );
    %LET DataFile = %SCAN ( &DataList, &I, %STR( ) );
  %END; %* macro DO WHILE;

  PROC TRANSPOSE DATA=&Appended (DROP=_LABEL_) NAME=Attribute OUT=&Appended;
    ID '*Dataset'n; VAR _ALL_;
  RUN;

  PROC SORT DATA=&Appended OUT=&Sorted; BY Attribute; RUN;

  DATA &Crossref;
    SET &Appended (WHERE=(Attribute EQ: '*'))
        &Sorted (WHERE=(Attribute NE: '*'));
    IF      ( SCAN ( Attribute, 2, '!' ) EQ '1' ) THEN Attribute = '>Type/Length';
    ELSE IF ( SCAN ( Attribute, 2, '!' ) EQ '2' ) THEN Attribute = '>Informat';
    ELSE IF ( SCAN ( Attribute, 2, '!' ) EQ '3' ) THEN Attribute = '>Format';
    LABEL Attribute = 'columns=datasets, rows=variables and attributes';
  RUN;

%Aborting:
%Exit:
%Finish:

%* Cleaning up ;

%* Restore modified option setting of VALIDVARNAME to previous one;
  OPTIONS VALIDVARNAME = &ValidVar;

%* Delete work datasets;
  %_Delete_ (&Contents &AttrData &NameData &TypeLen1 &Inform2 &Format3 &Appended &Sorted);

%MEND Crossref;
```

In the code of version 0.6.2 presented here all debugging code and some documentation has been removed.