Paper PH-11

Generating SUPPQUAL Domains from SDTM-Plus Domains

John R Gerlach, CSG, Inc.
Glenn O'Brien, Allergan, Inc.

ABSTRACT

Generating Supplemental (SUPPQUAL) domains is a staple component of any CDISC conversion project. In fact, many of the SDTM domains often have a respective SUPPQUAL domain, which can be just as challenging to produce, as well as being considerably larger in size. Even if there is an SDTM-Plus domain that contains the variables intended for a respective SUPPQUAL domain, the task of creating the respective SUPPQUAL domain can still be quite tedious, as well as error prone. Obviously, it would be better to automate this process in a way that would guarantee compliance and accuracy. This paper explains a SAS® utility for generating SUPPQUAL domains from SDTM-Plus domains.

INTRODUCTION

Consider a SAS data library that contains SDTM Plus domains; that is, several of the domains contain other variables that are intended to be stored in their respective SUPPQUAL domains. For example, the AE and MH domains might contain variables that originate from the MedDRA dictionary (e.g. SOCCD, HLT, LLT), used for coding adverse events. Or, perhaps the EG domain contains the variable ECCLSIG denoting the clinical significance of a finding. Except for these Plus variables, these domains would be compliant to the CDISC standard. Thus, it seems plausible to generate the Parent domain by keeping those variables defined by that domain and storing the other variables in their respective SUPPQUAL. Obviously, creating the SUPPQUAL domain requires more effort.

A REVIEW of SUPPQUAL

Because the SDTM standard does not allow the addition of new variables, a SUPPQUAL domain affords a means to store additional information on a subject or event. This information can be either objective or subjective values; that is, collected / derived values versus attributions whose values are assigned by a person or committee. For attributions that have been assigned (i.e. QORIG='ASSIGNED'), the QEVAL indicates who assigned the value. Thus, for objective data (e.g. QORIG='CRF'), the QEVAL should be null.

Every SUPPQUAL contains ten variables because variables in a SUPPQUAL domain are either Required or Expected: five Key variables that reference a specific record in its Parent domain and five Q-variables that contain the actual supplemental data, as shown below.

Keys Variables:

• STUDYID	Study Identifier	*JRG-19540907*
• RDOMAIN	Related Domain Abbreviation	*AE*
• USUBJID	Unique Subject Identifier	*54-01-101*
• IDVAR	Identifies the Related Records	*AESEQ*
• IDVARVAL	Value of IDVAR	*3*
•		

Supplemental Data:

• QNAM	Variable Name	*AETRTEM*
• QLABEL	Variable Label	*Treatment Emergent Flag*
• QVAL	Data Value	*Y*
• QORIG	Origin	*Derived*
• QEVAL	Evaluator	*Sponsor*

The IDVAR is usually the Sequence variable (e.g. AESEQ); however, notice that the IDVAR and IDVARVAL variables are null for the SUPPDM (Demography) domain, since the USUBJID is sufficient to reference the records in the DM domain. Although a single SUPPQUAL domain may represent all supplemental information for a study, usually, SUPPQUAL domains are created each representing its parent domain and following the naming convention SUPPxx, where *xx* denotes the parent domain (e.g. SUPPAE).

Compare Parent domains and SUPPQUAL domains. For example, the Demography (DM) domain contains one record per subject and the Adverse Events (AE) domain contains one adverse event (on a given day) per subject. Now, notice the primary key of *any* SUPPQUAL domain: One record per supplemental variable that references a specific record in the parent domain for a subject or, as the CDISC Implementation Guide states, "One record per IDVAR, IDVARVAL, QNAM value per subject." The variables USUBJID, IDVAR and IDVARVAL reference the record in the parent domain for a specific subject. Thus, the structure of the SUPPQUAL is constant; whereas, the Parent domain has two possible structures: Subject and Subject Event.

THE METHOD

Note: For this discussion, it is assumed that the IDVAR / IDVARVAL will be either null (for Demography) or use the Sequence variable for all other Observation class domains. In other words, the –GRPID variable is not considered.

Given a Plus domain, each supplemental variable contributes a record to its respective SUPPQUAL domain. But, how do we determine which SUPPQUAL domains are required and which variables are supplemental? Moreover, what about the QORIG and QEVAL variables, which are Required and Expected variables, respectively? The answer to both questions – Metadata, of course.

Below is an excerpt of an Excel spreadsheet that contains the values for the variables QORIG and QEVAL. This external file does more than allow a mechanism for assigning QORIG and QEVAL. This file actually identifies those variables in the SDTM Plus domains that contribute to the respective SUPPQUAL domains. Thus, the SDTM Plus domains could have variables that are not used for creating SUPPQUALs, as well as the variables belonging to the Parent domain. But, that may be a compliance issue.

Domain	Variable	QORIG	QEVAL
AE	ACTTAKN	CRF	NULL
AE	AESOSP	CRF	NULL
AE	AECLSIG	DERIVED	SPONSOR
AE	AETRTEM	DERIVED	SPONSOR
AE	HLT	CRF	NULL
AE	SOCCD	ASSIGNED	NULL
AE	PTCD	ASSIGNED	NULL
DM	COMPLT	DERIVED	SPONSOR
DM	FULLSET	DERIVED	SPONSOR
DM	ITT	DERIVED	SPONSOR
DM	PPROT	DERIVED	SPONSOR
DM	SAFETY	DERIVED	SPONSOR
DM	RACEOTH	CRF	NULL
EG	EGCLSIG	DERIVED	SPONSOR
MH	SOCCD	ASSIGNED	NULL
MH	PTCD	ASSIGNED	NULL
MH	REAS	CRF	NULL

Table 1. Partial List of SUPPQUAL variables.

Admittedly, the utility could be designed to identify a SUPPQUAL variable as those outside the collection of variables defined for a given domain. However, such a method does not afford the opportunity to assign the variables QORIG and QEVAL, and thereby explicitly identify the variables to be transferred. Thus, the utility assumes that the external file identifies all supplemental variables and creates a data set called SUPPVARS. Thus, there are two input data sources for this utility: the SDTM Plus data library and the metadata. The SDTM Plus data library has the library reference SDTMPM; whereas, the metadata resides in the WORK data library.

DRIVER MACRO

Conceptually, the driver macro contains two steps:

- Identify the target SUPPQUAL domains
- Create each SUPPQUAL domain.

The first SQL step creates the macro variable NSUPPS denoting the number of SUPPQUAL domains which is known from the SUPPVARS data set. The second SQL step creates macro variables denoting each SUPPQUAL domain. The %DO loop iteratively invokes the macro %Gen_Suppqual that creates each SUPPQUAL domain. Notice the single parameter &&domain&i.. that denotes the *ith* domain.

```
%macro Gen_SuppQuals;
   proc sql noprint;
      select count(distinct memnam) into :nsupps
         from suppvars;
   quit;
   %let nsupps = nsupps.;
   proc sql noprint;
      select distinct memname into :domain1 - :domain&nsupps.
         from suppvars;
   quit;
   %do i = 1 %to &nsupps.;
      %Gen_SuppQual(&&domain&i..) ;
      %end;
   proc datasets library=work nolist kill;
   quit;
%mend Gen_SuppQuals;
```

GENERATING EACH SUPPQUAL DOMAIN

Keep in mind that the premise is a Plus domain that contains one or more supplemental variables that are used to create its respective SUPPQUAL domain. Consider the Demography (DM) domain that includes many of its defined variables along with the variables denoting study population flags: FULLSET, ITT, SAFETY, PPROT, and COMPLT. Thus, an abstract of the augmented DM Plus domain would appear as follows:

```
 #     Variable     Type     Len     Label

 1     STUDYID      Char      10     Study Identifier
 2     DOMAIN       Char       2     Domain Abbreviation
 3     USUBJID      Char      20     Unique Subject Identifier
 4     SUBJID       Char       9     Subject Identifier
 5     RFSTDTC      Char      19     Subject Reference Start Date/Time

       :            :            :            :            :

20     FULLSET      Char       1     Full Analysis Set Flag
21     ITT          Char       1     Intent-to-Treat Population Flag
22     SAFETY       Char       1     Safety Population Flag
23     PPROT        Char       1     Per Protocol Set Flag
24     COMPLT       Char       1     Completers Population Flag
```

The supplemental variables in the DM domain must be transferred to its supplemental domain SUPPDM. In fact, each variable becomes a record for each subject in the DM domain. Thus, if there are five variables denoting study population flags, then there will be five records for each subject, assuming each variable is not null. Thus, if there are 100 subjects in the DM domain, there should be 500 records in the SUPPDM domain. Obviously, it is very important to be able to estimate the number of observations in each SUPPQUAL domain.

How does a *variable* in the parent domain become a *record* in its SUPPQUAL domain? Also, how do you generate each SUPPQUAL such that the process becomes transparent, that is, depending on the contents of the SDTM Plus data library? First of all, the supplemental variables must be identified. The following SQL step creates the macro variable SUPPVARS that contains the enumerated list of supplemental variables specific to a Parent domain.

```
proc sql noprint;
   select distinct qnam into :suppvars separated by ' '
      from suppvars
      where upcase(domain) eq "&domain.";
quit;
```

The following Data step obtains a copy of the Parent domain keeping only the variables of interest, that is, those needed to populate the Key variables and the Supplemental Q-variables, except for QORIG and QEVAL. Although the existence of a SUPPCO domain seems unlikely, the Data step drops the RDOMAIN variable for the Comments (CO) domain, since it exists already, albeit for a different purpose. That is, the RDOMAIN in the CO domain references the origin of the comment such that for general comments, its value is null. In contrast, the RDOMAIN variable in SUPPQUAL domains references the Parent domain and is constant, in this case having the value "CO". Also, notice the case when formulating the KEEP statement when processing the Demography (DM) domain. There is no Sequence variable in the DM domain, hence, the condition. Similarly, the BY statement of the SORT procedure is formulated accordingly with respect to the Sequence variable. It is arguable whether the data set should be sorted, since the SUPPQUAL domain will be sorted later.

```
data &domain.;
   set sdtm_pm.domain.
      %if &domain. Eq CO
         %then %do; (drop=rdomain) &end; ;
   keep STUDYID USUBJID
      %if &domain. Ne DM
         %then %do; &domain.SEQ %end;  &suppvars.;
run;
proc sort data=&domain.;
   by studyid usubjid
      %if &domain. ne DM
         %then %do; &domain.SEQ %end;
run;
```

In order to generate the SUPPQUAL domain, the utility processes the Parent domain for each variable. The %DO %WHILE statement performs the iterative process by parsing the SUPPVARS macro variable via the %SCAN macro function.

```
%let v = 1;
%do %while(%scan(&suppvars.,&v.,' ') ne  );
   %let suppvar = %scan(&suppvars.,&v.,' ');

   < Process the Parent domain for the ith variable >

   %end;
```

Using the COLUMNS Dictionary table the SQL procedure creates two macro variables denoting the data type and label for the variable. The following SQL step creates two more macro variables denoting the QORIG and QEVAL, which is found in the SUPPVARS data set.

```
proc sql noprint;
   select upcase(substr(type,1,1)), label into :type, :label
      from dictionary.columns
      where libname eq 'SDTM_PM' and memname eq "&domain."
         and name eq "&suppvar.";
quit;
proc sql noprint;
   select qorig, qeval, into :qorig, :qeval
      from suppvars
      where domain eq "&domain." and qnam eq "&suppvar.";
quit;
%let qorig = &qorig.;
%let qeval = &qeval.;
```

The Parent domain and the several macro variables represent all the information needed to create the SUPPQUAL domain for one variable. The variables STUDYID and USUBJID need no special treatment, since they exist in the

Parent domain. The LENGTH statement ensures that the SUPPQUAL variables will have the appropriate attributes. The RETAIN statement assigns the variables RDOMAIN, QNAM, QLABEL, QORIG, and QEVAL their constant values for that supplemental variable. The variables IDVAR and IDVARVAL variables are assigned either null values, for the DM domain, or the name and value of the Sequence variable. Finally, the QVAL variable is assigned the value of the supplemental variable, albeit on condition. That is, if the supplemental variable is numeric, then, the PUT function is used; otherwise, the UPCASE function is used, since QVAL is always a character variable. The KEEP statement makes sure that the SUPPQUAL data set contains only the ten target variables.

```
data suppqual;
   length RDOMAIN $2 IDVAR $8 IDVARVAL $100 QNAM $8 QLABEL $40
      QVAL $100 QORIG QEVAL $50;
   retain RDOMAIN "&domain." QNAM "&suppvar." QLABEL "&label."
      QORIG "&qorig." QEVAL "&qeval.";
   set &domain.;
   %if %upcase(&domain.) eq DM
      %then %do;
         IDVAR    = " ";
         IDVARVAL = " ";
         %end;
      %else %do;
         IDVAR    = "%upcase(&domain.)SEQ";
         IDVARVAL = left(put(&domain.seq,best.));
         %end;
   %if %upcase(&type.) eq N
      %then %do;
         QVAL = left(put(&suppvar.,best.));
         %end;
   if QVAL ne ''
     then output;
   keep STUDYID RDOMAIN USUBJID IDVAR IDVARVAL QNAM QLABEL QVAL QORIG QEVAL;
run;

proc append base=supp&domain. data=suppqual;
run;
```

The SUPPQUAL data set represents only one supplemental variable, which is appended to a Base data set, such as SUPPDM. Thus, obviously, the Parent data set is processed as many times as there are supplemental variables. It is left to the reader to consider an enhancement to this SAS solution so that the Parent domain is processed once.

Even though the SUPPQUAL domain contains all the appropriate data, it is just as important to ensure CDISC compliance. The following SQL step accomplishes the task of ensuring compliance, including variable labels, arranging the order of the variables, and sorting the domain. Notice that the data set label for the SUPPQUAL domain follows a simple format, "Supplemental Qualifiers for <Domain Abbreviation>." The target SUPPQUAL domain is stored in the SDTM data library.

```
proc sql;
   create table sdtm.SUPP%upcase(&domain.)
      (label="Supplemental Qualifiers for %upcase(&domain.)") as
   select STUDYID   label = 'Study Identifier',
          RDOMAIN   label = 'Related Domain Abbreviation',
          USUBJID   label = 'Unique Subject Identifier',
          IDVAR     label = 'Identifying Variable',
          IDVARVAL  label = 'Identifying Variable Value',
          QNAM      label = 'Qualifier Variable Name',
          QLABEL    label = 'Qualifier Variable Label',
          QVAL      label = 'Data Value',
          QORIG     label = 'Origin',
          QEVAL     label = 'Evaluator'
      from sup&domain.
      order by STUDYID, RDOMAIN, USUBJID, IDVAR, IDVARVAL;
quit;
```

Finally, the utility performs several quality assurance checks, for example, a frequency distribution on QNAM to corroborate expected cardinality (not shown). As discussed earlier, if there are one hundred subjects in a study, then there should be no more than one hundred instances of a given value for QNAM.

CONCLUSION

Creating SUPPQUAL domains has become a common feature of CDISC conversion projects. The process of creating these important domains can be both time-consuming and error prone. However, beginning with Plus domains, there's a way to automate the process of creating SUPPQUAL domains that saves time and ensures compliance. By using metadata that identifies the supplemental variables along with the Parent domain, the proposed utility creates a SUPPQUAL domain straightaway.

REFERENCES

Clinical Data Interchange Standards Consortium, 2008. *CDISC SDTM Implementation Guide, Version 3.1.2.*

SAS (PharmaSUG) Conference Proceedings:

http://www.lexjansen.com/cgi-bin/xsl_transform.php?x=psug11&s=pharmasug&c=pharmasug

ACKNOWLEDGMENTS

The primary author thanks CSG, Inc., for its support of this conference.

CONTACT INFORMATION

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

Name:	John R. Gerlach	Glenn O'Brien
Enterprise:	CSG, Inc.	Allergan, Inc.
City, State:	Raleigh, NC	Irvine, CA
E-mail:	JRGerlach@optonline.net	lobgob@verizon.net