SESUG 2020 Paper 163

Creating Data Listings Utilizing PROC Procedures

Ballari Sen, Agios Pharmaceuticals,INC.

ABSTRACT

This paper describes the implementation of clinical datasets mapping with structured programming design and methodology. PROC procedures is a powerful tool for SAS® programmers.

PROC SQL Procedure combines data step functionality & PROC data steps into a single step. PROC SQL can be used to retrieve, update, sort & summarize & create PROC Report from other SAS Datasets or another database.

The focus of this paper will detail the advantage of using PROC SQL and to help clinical programmers utilize the various SQL style codes of the SELECT statement. PROC SQL is a preferred choice in programming to improve the performance & efficiency.

This paper further discusses the utilization of PROC SQL to create data listings for data review & mapping specifications.

INTRODUCTION

The SQL procedure is a wonderful tool for querying and subsetting data sets; it helps to reconstruct data by using Case expressions and by joining by two or more tables to further explore data relationships (Lafler, Kirk Paul (2013, 2011).

PROC SQL is one of the most useful and handy built-in procedures in SAS and it has multiple functionality as it can sort, merge, concatenate two different table or datasets, create new variables and tables and summarize all data all at once (Lafler, Kirk Paul (2013, 2011).

The Syntax for a basic SQL Query looks like the following :

```
PROC SQL;
CREATE TABLE Table_name AS
FROM Master_table_name
WHERE conditions
GROUP BY column_names
HAVING having_conditions
ORDER BY column_name_list ;
QUIT;
```

The PROC SQL statement, the select statement and FROM Clauses are required in a SQL Query. The select statement can list out the columns to be selected or by PROC SQL utilizes asterisk (*) to select specific columns from the master table (Wang, Jessica (2014).

The FROM clause lists the original table name. CREATE TABLE clause saves the query into a new table. The WHERE clause lists the query conditions for the query output table. The ORDER BY clause sorts the result. GROUP BY is used together with group functions in the SELECT clause to subset the query result or to summarize the result. The HAVING clause, is used together with the GROUP BY clause, to select only the query result that satisfies the condition after the group function (Wang, Jessica (2014).

The order of the SQL clauses should follow the order given in the syntax above.

In PROC SQL Language, according to SAS terminology datasets, observations & variables are referred also as Table, row & column accordingly (Wang, Jessica (2014).

This paper focuses in producing a data listing for having a list of subjects who crossed over from placebo to the active study Drug in order to better understand when the last date of the placebo dose was administered, the start date of the first dose of the active Drug, and also to include whether the adverse events occurred during the administration of placebo or the active dose Drug. The use of the data listing is to provide with small targeted details at data elements as part of the workflow which helps in conducting and reviewing clinical trial data on an ongoing basis.

In this paper, dummy clinical trial data in pharmaceutical industry is used for examples; such as Exposure (EX), Adverse Events (AE), Disposition (DS).

CREATING LIBRARY NAMES

PROC SQL provides several ways to create new tables.

Here we are going to discuss how to create a new table from a query result.

In this paper we need to obtain data variables from multiple tables and combine those tables horizontally by PROC SQL JOIN.

As an example, dummy clinical trials dataset/tables are utilized. Three tables DS_SDTM,SUBVIS & AE_SDTM, contain the Disposition, Subject visit & Adverse events information needed for the subjects. Table DS_SDTM has individual Subject information about the first & last dose of the study Drug or placebo & subjects receiving active Drug or placebo.

Table SUBVIS has subject visit name and corresponding visit date. Table AE_SDTM has Start & End date of Adverse Event & Status Suggesting if the start date of the Adverse Event is the same as the date of first dose of study medication then populate values : Before First Dose/After Last Dose, Serious & Ongoing Adverse Events Variables.

We need to merge DS_SDTM,SUBVIS & AE_SDTM based on subject in order to populate the data listings about Subjects and to figure out which Subjects Crossed Over from placebo to the Active Drug.

The first step is creating library name for the clinical trial study datasets.

```
/**Macro creating Raw Datasets Library name***/

%let drive =A:\;

Libname XXXXR "&drive.\Clinical Trials data\Data Staging\XXXXX\XXXXX-B-
017\Export\raw";

/**Macro creating SDTM Datasets Library name ***/

%let drive =B:\;

Libname XXXF "&drive.\XXXXX\Final SDTM"

outencoding='wlatin1';
```

CREATING TABLES WITH PRE-DEFINED TABLE STRUCTURE

The second step in the programming process is to populate the records for all the subjects from the disposition domain.

The below code defines :

1. DSYN Variable : Did the subject receive active Drug or the placebo (Yes/No)?
2. DSCROSYN Variable : Is the subject continuing to the crossover visit?
3. Date of first dose of study Drug.
4. Date of last dose of study drug or placebo.

Populating distinct records of date of first dose of study Drug, date of last dose of study Drug or placebo variable from the Disposition dataset.

```
PROC SQL;
create table SUBDET as select distinct
SUBJECT,DSENDATE,DSEXDATE from XXXX.DS_SDTM
where DSYN = 'Yes' AND DSCROSYN = 'Yes' ;
run;
```

```
NOTE : Table Work.SUBDET created, with 3 rows and 3 columns.
```

	Subject name or identifier	Date subject ended treatment	First dose date of Study Drug
1	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000
2	10002	20FEB2019:00:00:00.000	28FEB2019:00:00:00.000
3	10003	12MAR2018:00:00:00.000	15MAR2018:00:00:00.000

Output 1. Output from Disposition Dataset

The third step to populate distinct records from the subject visit dataset, where the visit is "Crossover Cycle 1 Day 1".

Populating distinct records of Visit not done variable is not null, Visit name & Date of visit variable from the Subject Visit dataset.

```
PROC SQL;
create table SUBVISF as select distinct SUBJECT,VISITNDNE,VISIT,SVDATE
from XXXX.SUBVIS where VISIT = 'Crossover Cycle 1 Day 1'
and VISITNDNE ^=1;
RUN;
```

```
NOTE : Table Work.SUBVISF created, with 3 rows and 3 columns.
```

	Subject name or identifier	Date of Visit Not Done	Visit Name	Date of Visit
1	10001	0	Crossover Cycle 1 Day 1	05DEC2017:00:00:00.000
2	10002	0	Crossover Cycle 1 Day 1	28FEB2019:00:00:00.000
3	10003	0	Crossover Cycle 1 Day 1	15MAR2018:00:00:00.000

Output 2. Output from Subject Visit Dataset

A left join is produced between SUBDET dataset & SUBVISF dataset to populate the start date for the subject visit variable from SUBVISF dataset that matches with the(i.e. Subject value) SUBDET dataset.

```
PROC SQL;
CREATE TABLE JOIN_DS_SV AS
SELECT A.*,B.SVDATE from work.SUBDET A LEFT JOIN WORK.SUBVISF B
ON A.SUBJECT = B.SUBJECT;
RUN;
```

```
NOTE : Table Work.JOIN_DS_SV created, with 3 rows and 3 columns.
```

	Subject name or identifier	Date subject ended treatment	First dose date of Study Drug	Date of Visit
1	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000	05DEC2017:00:00:00.000
2	10002	20FEB2019:00:00:00.000	28FEB2019:00:00:00.000	28FEB2019:00:00:00.000
3	10003	12MAR2018:00:00:00.000	15MAR2018:00:00:00.000	15MAR2018:00:00:00.000

Output 3. Output from Left Join between Disposition & Subject Visit Dataset.

The fourth step in the programming process is to populate the records for all the subjects from the Adverse Events domain.

The below code defines :

1. AEYESNO Variable : Were any adverse events experienced? (Yes/No)?
2. Adverse Event
3. Start Date of Adverse Event
4. End Date of Adverse Event
5. Status Suggesting if the start date of the Adverse Event is the same as the date of first dose of study medication then populate values : Before First Dose/After Last Dose
6. Adverse Event Ongoing after the End of Treatment : AE Ongoing
7. Adverse Event Serious or Not : Serious.

```
PROC SQL;
Create table AE_SDTM_F as select
distinct SUBJECT,AEYESNO,AETERM,AESTDATE,AESTAT,AEENDATE,AEONGONG,AESER
from XXXX.AE_SDTM where AEYESNO = 'Yes';
RUN;
```

```
NOTE : Table Work.AE_SDTM_F created, with 2286 rows and 8 columns.
```

	Subject name or identifier	Were any adverse events experienced	Adverse event	AE Start date	Before or After First Dose	AE End Date	AE Ongoing	Serious
1	10001	Yes	Back pain (lower back), intermittent	12NOV2017:00:00:00.000		.	1	No
2	10001	Yes	Blurred vision, intermittent	09JAN2018:00:00:00.000		.	1	No
3	10001	Yes	Bruising	03NOV2017:00:00:00.000		.	1	No

Output 4. Output from Adverse Event Dataset.

The fifth step in the programming process is to program a LEFT JOIN between JOIN_DS_SV dataset & AE_SDTM dataset to include the Adverse Event Start, End date & populate variable for Adverse Event Serious for all the subjects who has crossed over from placebo to the Active Study Drug where the Visit Name is Cycle 1 Day 1.

```
PROC SQL;
CREATE TABLE LIST_JOIN AS SELECT distinct
A.*,B.AETERM,B.AESTDATE,B.AEENDATE,B.AESER from work.JOIN_DS_SV
A LEFT JOIN WORK.AE_SDTM_F B
ON A.SUBJECT = B.SUBJECT ;
RUN;
```

```
NOTE : Table Work.LIST_JOIN created, with 61 rows and 8 columns.
```

	Subject name or identifier	Date subject ended treatment	First dose date of Study Drug	Date of Visit	Adverse event	AE Start date
1	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000	05DEC2017:00:00:00.000	Back pain (lower back), intermittent	12NOV2017:00:00:00.000
2	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000	05DEC2017:00:00:00.000	Blurred vision, intermittent	09JAN2018:00:00:00.000
3	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000	05DEC2017:00:00:00.000	Bruising	03NOV2017:00:00:00.000
4	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000	05DEC2017:00:00:00.000	Diarrhea, intermittent	01NOV2017:00:00:00.000
5	10001	20NOV2017:00:00:00.000	05DEC2017:00:00:00.000	05DEC2017:00:00:00.000	Fatigue, intermittent	05NOV2017:00:00:00.000

Output 5. Output from Left Join between Disposition, Subject Visit & Adverse Dataset.

DEFINE TABLE TEMPLATE

1. The step creates the table template for the data listings which represents the subject information and their treatment groups with their Start & End date.
2. LABELS will give the ability to define longer column headings.
3. Each column relates to a specific variable in the data and will have a column heading predefined in the template.

```
PROC SQL;
title  "XXXX-X-XXX";
      CREATE TABLE WORK.LISTING AS
            SELECT t1.Subject LABEL = "Subject",
              (Put(DATEPART(t1.DSENDATE),date11.))AS DSENDATE LABEL = "Date
of Last placebo Dose Administered",
              (Put(DATEPART(t1.DSEXDATE),date11.))AS DSEXDATE LABEL = "Date
of first Dose of Active Drug",
```

```
            (Put(DATEPART(t1.SVDATE),date11.)) AS SVDATE LABEL = "Date of
crossover C1D1",
                  t1.AETERM LABEL = "Adverse Event",
               (Put(DATEPART(t1.AESTDATE),date11.)) AS AESTDATE LABEL = "AE
Start date",
                  (Put(DATEPART(t1.AEENDATE),date11.)) AS AEENDATE LABEL =
"AE End date",
                  t1.AESER LABEL = "Adverse Event Serious"
     FROM work.LIST_JOIN t1
      ORDER BY t1.Subject;
QUIT;
```

```
NOTE : Table Work. Listing created, with 61 rows and 8 columns.
```

	Subject	Date of Last Placebo Dose Administered	Date of first Dose of Active Drug XXXX	Date of crossover C1D1	Adverse Event	AE Start date	AE End date	Adverse Event Serious
1	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Back pain (lower back), intermittent	12-NOV-2017	.	No
2	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Blurred vision, intermittent	09-JAN-2018	.	No
3	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Bruising	03-NOV-2017	.	No
4	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Diarrhea, intermittent	01-NOV-2017	.	No
5	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Fatigue, intermittent	05-NOV-2017	.	No
6	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Fever	12-NOV-2017	19-NOV-2017	No
7	10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Injury, poisoning, and procedural complications-Other, specify: Back injury	23-DEC-2017	.	No

Output 6. Output from the Report.

DEFINE THE PROGRAM STYLE TEMPLATE

The below code defines the PROC REPORT Style :

1. The protocol Name
2. The Company Name
3. Creation Date of the SDTM Compliant datasets
4. where the profile template will be stored (Out-path location)
5. Creation Date of the Data Listing was implemented & programmed.

```
/******** Variables for storing dataset header footer info **********/
%Let subjid='';
%Let protocol=%str(XXXXX-X-XXX);
%let company = %str(Company Name);
/***********Get Dataset Date**************/
proc contents data=XXXX.DM noprint out=crdate (keep= crdate);
run;
proc SQL;
      CREATE TABLE WORK.CDATE AS
SELECT put(datepart(CDATE),date11.) || ' ' || put(timepart(CDATE),hhmm.) AS
CRDATE
FROM
```

```
(SELECT MAX(CRDATE) AS CDATE FROM CRDATE);
QUIT;
data _null_;
set CDATE;
call symput ("CRDATE", CRDATE);
run;
/********Basic Header***********/
            title1 height=1 j=L "&company.";
            title2 height=1 j=L "&Protocol." j=r "For Company Review";
            title3 height=4 bold j=L "Listing of subjects who crossed over
from placebo to Active Drug" j=r;
ods excel file='A:\patientlisting.xlsx' options (embedded_titles='yes'
tab_color='purple' autofilter='1-8');
Title 'Listing of subjects who crossed over from placebo to Active Drug' ;
/**Programing PROC REPORT for Data Listing      **/

proc report data=work.listing center wrap nowd headline headskip
spacing=1 NOFS split="|" missing style(header)= [font_size=1]
style(header)=[background=pink];
column ('Patient Listing' SUBJECT DSENDATE DSEXDATE SVSTDTC AETERM AESTDTC
AEENDTC AESER);
define DSENDAT                / style=[just=L cellwidth=200];
define DSEXDAT            / style=[just=L cellwidth=200];
define SVSTDTC                 / style=[just=L cellwidth=200];
define AETERM               / style=[just=L cellwidth=100];
define AESTDTC              / style=[just=L cellwidth=100];
define AEENDTC            / style=[just=L cellwidth=200];
define AESER      / style=[just=L cellwidth=100];
run;
ods excel close;
options mprint symbolgen mlogic ;
```

Listing of subjects who crossed over from Placebo to Active Drug

Patient Listing

Subject	Date of Last Placebo Dose Administered	Date of first Dose of Active Drug XXXX	Date of crossover C1D1	Adverse Event	AE Start date	AE End date	Adverse Event Serious
10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Back pain (lower back), intermittent	12-NOV-2017	.	No
10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Blurred vision, intermittent	09-JAN-2018	.	No
10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Bruising	03-NOV-2017	.	No
10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Diarrhea, intermittent	01-NOV-2017	.	No
10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Fatigue, intermittent	05-NOV-2017	.	No
10001	20-NOV-2017	05-DEC-2017	05-DEC-2017	Fever	12-NOV-2017	19-NOV-2017	No

Output 7. Output from the PROC Report Data Listing.

CONCLUSION

PROC SQL is a very powerful with a lot of functions and statements and it can all included in one step. It can replace bulky SAS code as the need for processing the data is minimized.

PROC SQL statement is a combination of data steps and several SAS procedures so minimizing the code and improves the clarity and the efficiency of the program and adding comments at the beginning of the PROC SQL code makes it further more easier to detail exactly what the program is doing.

In conclusion PROC SQL is a powerful tool which balances clarity, descriptiveness and efficiency in generating a code and can be a powerful tool for programmers in the pharmaceutical industry.

REFERENCES

1. Lafler, Kirk Paul (2013, 2011), "Quick Results with PROC SQL,". *Proceedings of the 2013 North East SAS Users Group (NESUG) Conference*; *Proceedings of the 2011 Western Users of SAS Software (WUSS) Conference; Proceedings of the 2011 MidWest SAS Users Group (MWSUG) Conference*, Software Intelligence Corporation, Spring Valley, CA, USA.

2. Wang, Jessica (2014)."Using the Power of SAS SQL". Proceedings of the PharmaSUG,Paper BB10, Regeneron Pharmaceuticals Inc., Basking Ridge, NJ,USA.

3. SAS® 9.3 SQL Procedure User's SAS® 9.3 SQL® Procedure User's Guide. AvailableatURL: https://support.sas.com/documentation/cdl/en/sqlproc/63043/PDF/default/sqlproc.pdf

ACKNOWLEDGMENTS

I would like to thank my manager, Sneha Kadalur, for her constant encouragement and critical Review. I would also like to thank Robert O' Connor and Megan Kellam for providing me the opportunity to program and work on the data listing and also for their suggestions

and assistance in creating the data listing requirements which helped in a great deal to create the final data listing excel sheet with auto filter option.

CONTACT INFORMATION

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

Author Name : Ballari Sen
Company : Agios Pharmaceuticals,INC.
Email : ballari.sen@agios.com