



Vienna 2015
Clinical Data Science
October 11th–14th 2015
Intercontinental Wien Hotel, Vienna



SQL by Example: A Comparative Introduction to PROC SQL

Jeanina (Nina) Worden, Santen,
Emeryville, USA



A few notes before starting...

- There are numerous techniques to achieve the same result in SAS®
 - The examples used here, or in the paper, are not meant to be the definitive solution
 - This paper is NOT an argument for the sole use of PROC SQL, rather an introduction to a potential alternative technique for programmers
- This presentation will not cover each example presented in my paper, instead it is a consolidation of several



First the Basics...



Basic Syntax

- The basic syntax for PROC SQL is:
proc sql;
create table <new dsn> as
select *
from <original dsn>;
quit;
- CREATE TABLE is used to designate the new dataset name
- The SELECT line specifies the variables to be added to the new dataset
 - In this example “*” is used to denote all variables should be selected
- The FROM line denotes the data source
- The code is ended with QUIT;



Today's Example



Example

- Today's example will be a comparison between the DATA step method and PROC SQL to:
 - Creating a variable, MAXVISIT, that contains the maximum visit for a subject
 - Merge the new subject dataset with the SITE dataset to get the PI name



Example

- Using the following two simple datasets

SITE dataset

SITEID	NAME
1	John Smith
2	Sam Jones

SUBJECT dataset

SUBJID	SITE	VISIT
123	1	1
123	1	4
456	2	1
456	2	2

DATA Step Method



Does this look familiar?

```
proc sort data=subject;  
    by subjid site;  
run;
```

```
proc transpose data=subject out=subj;  
    by subjid site;  
    var visit;  
run;
```

```
data max_visit;  
    set subj;  
    maxvisit=max(of col1-col2);  
    drop _name_ col1 col2;  
run;
```

- SUBJECT dataset is sorted, then transposed, to create variables to hold the individual visits
- The MAX function is used against the new variables to determine the maximum value for the new variable MAXVISIT



Dataset code continues...

```
proc sort data=max_visit  
    out=subj(rename=(site=siteid));  
    by site;  
run;
```

```
proc sort data=site;  
    by siteid;  
run;
```

```
data ds_new;  
merge subj site;  
by siteid;  
run;
```

- Each dataset that will be merged must be sorted before merging
 - Merging variable names are different so one needs renamed
- Final dataset can be produced by merging the sorted datasets



Using PROC SQL



Same code in SQL would be...

```
proc sql;  
create table sql_new as  
select distinct subjid, site,  
       max(visit) as maxvisit, name  
from subject as a, site as b  
where a.site=b.siteid  
group by subjid;  
quit;
```

- SORT is done in the background so presorting of the datasets is not needed
 - There is a time “cost” so this must be considered when using this method on larger datasets

- PROC SQL requires the source dataset to be designated for the merging variables
 - This is also used when selecting common variables for the new dataset
- PROC SQL can handle the differing names of the merging variables as long as they are of the same type
- MAX function can be used on the variable directly
 - The MAX function is controlled by the GROUP BY statement...if excluded the MAX would be for all records



A few closing notes

- PROC SQL can be an additional tool for programmers to complete common tasks, sometimes with less typing.
- Today's PROC SQL example demonstrated how:
 - Functions can be applied directly to the variable, avoiding data transformation coding
 - No explicit PROC SORT needed before merging, though there is still a processing “cost”
 - Merge keys can be of differing names



A few closing notes

- Looking for more info?
 - SAS® SQL Procedure User Guide is a wealth of information
 - My paper provides additional examples to include comparisons for:
 - “De-duping” datasets
 - Renaming of variables
 - Creating new variables
 - Merging
 - Full merges
 - One-sided
 - Conditional





Questions?



Contact Info

Name: Jeanina (Nina) Worden

Company: Santen, Inc.

Address: 2100 Powell Street, 15th Floor

City, State ZIP: Emeryville, CA 94608

Work Phone: (415) 268-9095

E-mail: jworden@santeninc.com

