Putting It Together – The Poster
A Data Set Joining Primer

Janet Stuelpner
Caroline Bahler

Abstract
The ability to combine different types of data from multiple hardware and software platforms is a major strength of the SAS® system. SAS has blessed information analysts with a wealth of different options for joining data values from many different data structures. There are several factors that will determine the type of join that is needed. The source of the data and the outcome are of utmost importance. This paper will discuss the joining techniques offered within the SAS system.

Introduction
Data warehouses can contain data collected and stored in many different physical forms. These data structures can include flat files, database tables, spreadsheets, and SAS data sets. Utilization of this "raw" data by an information analyst can require combining two or more of these data structures through the use of a join (merging and joining are synonymous terms referring to the combination of data structures through the use of common variables/fields). One of the strengths of the SAS system is that it provides many different options for joining data values from many different data structures.

The Software Environment
The "software environment" greatly influences how a group of two or more data structures are joined. Selection of a joining strategy increases in complexity as the number of "software environments" containing data increases. A standard rule of thumb is that data structures from the same environment should be joined within that environment. However, this does not always hold true since system resources and other factors may indicate that it is more "efficient" to join data structures from the same software within a different "environment". (Note: in previous versions of SAS prior to version 7, you could only join a SAS data set to another SAS data set. Any type of raw data, whether an excel spreadsheet, a flat file or a table from a relational database, had to be converted into a SAS data set before it could be merged together with data that was in a SAS data set. In version 7 and beyond, the new LIBNAME engines allow us to join all types of data sources to a SAS data set.)

Figure 1 illustrates a common joining strategy with some of the ambiguities involved. The optimum place to join a SAS data set and a flat file is within SAS. However, whether the database tables are joined within the database environment depends upon the type of join required. For instance, outer joins can be very database resource intensive and the "better" choice might be to join the two database tables within the SAS environment.

Types of Joins
The join operation works on two or more tables at a time. You can find any relationship that exists among data elements. There are several types of joins. We will focus on only three types. These are the cross product, inner join and outer join.

The inner join is also called an equi-join. The result of an inner join is the intersection of the rows in both tables. In other words, it is the rows that are common to both tables.

An outer join is used to return all rows that exist in one table even though the corresponding rows do not exist in the joined table. There are three types of outer joins: left, right, full. A left outer join returns all rows from the left table. A right outer join returns all rows from the right table. A full outer join returns all rows from both tables. The last type of join is a Cross-Join or a Cartesian Product. Some may say that this is not a join at all. In reality, it is the set of all possible combinations of the rows from two tables. Something to note is that if your tables are large, the output from a cross join could result in a table that has millions of rows of data.

SAS Merge vs. PROC SQL
One of the toughest decisions an analyst must make is whether to use a SAS merge or a PROC SQL. There are many pros and cons that must be

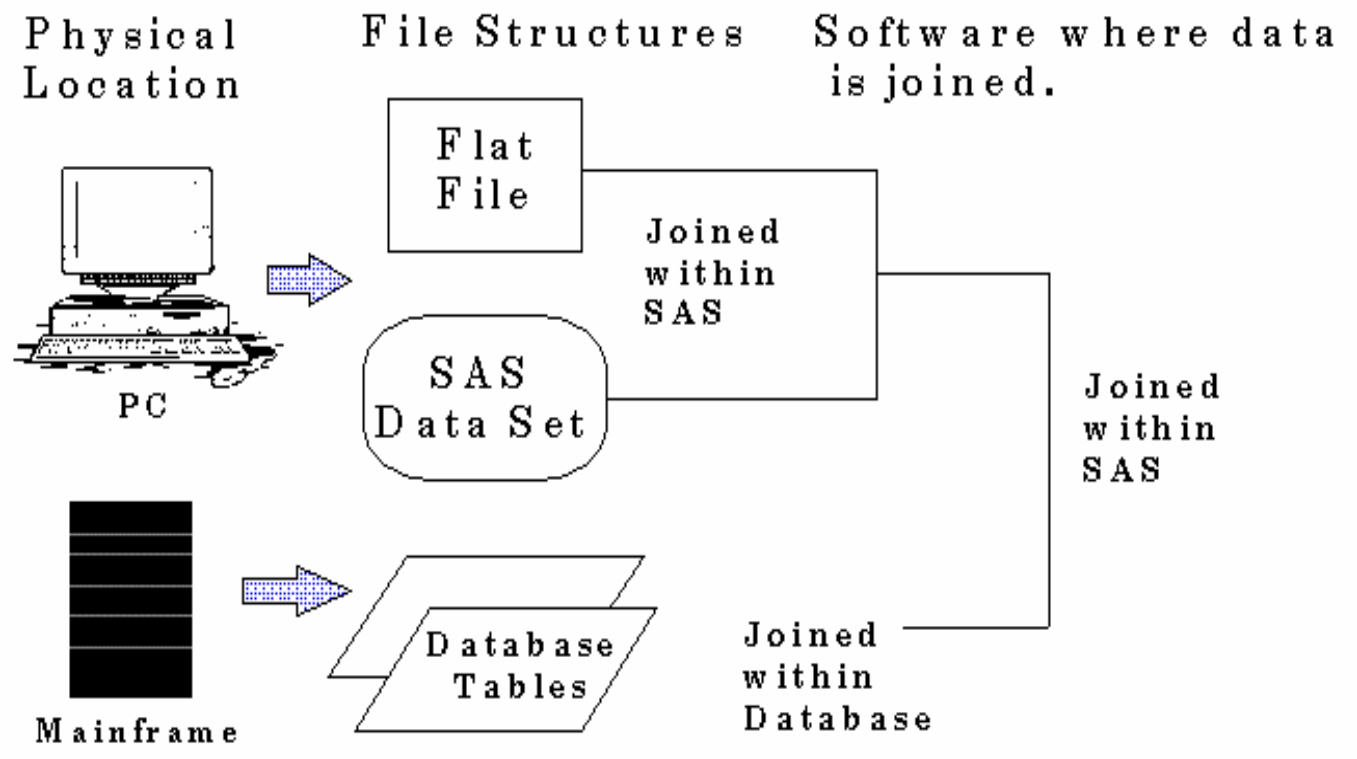

Figure 1. Example of joining strategy.

considered when making this decision. Let's take a look at each type of join and then compare.

Merge

A merge can be performed with a BY statement (match-merge) or without (one-to-one merge). Most often, a MERGE is used to combine two data sets based upon one or more common variables, which means BY group processing will take place. You must ensure that the observations in both data sets are in the correct order, or that they can be retrieved in the correct order. Whether you perform the SORT procedure first, the data is pre-sorted or indexed using the BY variable, it does not matter. The key variable (or BY variable) that is common to both data sets must have the same name and it is suggested that they be of the same length. There can only be one variable name used in the output data set. If you have variables of the same name in both data sets, the variables from the second data set will overwrite the data from the first. This can give you unexpected results in the output data set.

One great advantage of merging the data is the ability to use the IN= option with each data set that is being merged. This will set up a boolean operator that will allow easy coding of an outer join.

PROC SQL

PROC SQL is a powerful procedure combining some of the functionality of the DATA and PROC steps into a single procedure. PROC SQL in many cases can be a more efficient alternative to traditional SAS code. PROC SQL is often used as the interface to other database systems. With PROC SQL, you can retrieve, update, and report on information from SAS data sets or other database products. You can match variables that are not the same name. The data does not need to be pre-sorted. Only with SQL can you directly perform a cross join.

Table 1 illustrated the types of join data sets and the desirable SAS tool that is needed to merge the data. There are 5 scenarios that are defined. For each one several different tools can be used.

Scenario A - Groups of two or more parent data sets are used to build or create a child or output data set. In scenario A all data values from all of the parent

Informational Requirements	Type of Data Set Join	SAS Tool
A. All data values from all data sets.	Match-Merge or Full Outer Join	• MERGE statement with BY statement[4] • PROC SQL[2]
B. All data values from a single data set (base) and all data values from the other data set(s) that match the data values of the joining variables within the base.	Non-base data set(s) are used as • "Look-up" table(s) • Right or Left outer join.	• PROC FORMAT[1] • SET statement with KEYS= option[3] • PROC SQL[2]
C. Only those data values from all data sets that contain the same data values within the joining variables.	Inner Join	• PROC SQL[2] • MERGE statement with IN= option and BY statement4
D. Placement of data sets side by side	One-to-one Merge	MERGE statement[4]
E. Expansion of data set to include all levels of a non-common variable.	Many-to-many Join	PROC SQL[2]

Table 1. The type of parent data set joins required to construct a specific child data set.

data sets are needed to create the child data set (Table 1). The type of joins used to accomplish this are a match-merge or a full join. Missing values are placed within observations that do not occur within all parent data sets.

Scenario B - All data values of a base parent data set are kept and only those observations containing matching data values of the common variable(s) are selected from all other parent data sets (Table 1). The terminology right or left join is an indication of which parent data set to use as the base

Scenario C - An inner join is used when the child data set needs to contain only those data records from the parent data sets where the common variable(s) are identical.

Scenario D - One-to-one merging combines all of the parent data sets using a common variable and creates a child data set that is as large as the largest data set within the merge list. The parent data sets are not joined by common variables. Instead parent data sets are placed side by side and each common variable data value is super-imposed by the data values within the last parent data set within the merge.

Scenario E - Expansion of the child data set occurs when one data set has multiple levels of a non-common variable. An example would be where one data set contains a listing of names by city and the other contains city information. The child data set required contains all of the names for each city plus the city information. In this case the data sets are joined by the common variable city and all names within a city are transferred to the child data set.

Conclusions

The pros and cons of using the different joins are listed in Table 2.

The selection of a joining tool is dependent on the environment of the data structures, the required contents of the data sets and what tool is the most system resource efficient. During the application development process, careful bench-marking of joining tools is required to ensure selection of the correct environment and tool for the job.

ACKNOWLEDGMENTS

I would like to take this opportunity to acknowledge all of the help and support given to me by my husband, Robert Stuelpner. He diligently read and

SAS Tool	Pros	Cons
PROC FORMAT[1]	Creates a "look-up" table using either single variables or multiple variables for the key and label components of the format.	The key values must be unique, no duplicate values can occur within the data set used to create the format. Format can be applied to only one data set at a time.
MERGE statement[4]	• Used only for one-to-one merges no sorting required. • Two (2) or more data sets can be joined.	The data sets are not joined by any common variables. If there is a common variable between the data sets, then the common variable will contain the values of the common variable in the last data set joined.
MERGE statement with BY statement[4]	• Two or more data sets can be joined by common variables. • All values of all variables within all data sets will be retained.	All data sets must be sorted by common variable.
MERGE statement with IN= option and BY statement[4]	• Two (2) or more data sets can be joined by common variables. • All data from each data set will be read before subsetting criteria applied.	All data sets must be sorted by common variable.
PROC SQL[2]	• Data sets do not need to be pre-sorted. • Inner joins can occur between two (2) or more data sets.	• All outer joins occur between two data sets only. • Inner join contains only those values of the common variable that match between the data sets joined.
SET statement with KEYS= option[3]	Two (2) or more data sets can be joined by common variables.	Data sets need to be indexed by common variables.

Table 2. Pros and cons for using each type of SAS joining tool.

re-read this paper in an effort to correct the obvious errors and keep me on track. His criticism were constructive and his support never ending.

This has been a tremendous learning experience. Thanks to Caroline Bahler who raised the bar to even greater heights.

References

1. SAS Institute Inc.. SAS Procedures Guide, Version 6, Third Edition. Cary, NC: SAS Institute Inc., 1990. 275-312 pp.
2. SAS Institute. SAS Technical Report P-222, Changes and Enhancements to Base SAS Software, Release 6.07. Cary, NC: SAS Institute Inc., 1991. 91, 207-217 pp.
3. SAS Institute. SAS Language: Reference, Version 6, First Edition. Cary, NC: SAS Institute Inc., 1990. 147-155 pp.
4. Bahler, C. and Clos, S.. To Format or Merge ... That is the Question. Proceedings of the Southeast SAS Users Group. 3:363-367.
5. Bowman, JS, Emerson, SL and Darnovsky, M The Practical SQL Handbook, Addison-Wesley, 1996
6. Plew, RR and Stephens, RK Teach Yourself SQL in 24 Hours, SAMS, 2000

To Contact the authors:
Caroline Bahler - cbahler@hotmail.com
Janet Stuelpner - jstuelpner@usa.net