USING A SAS DATABASE

Carol Lambros, Warner-Lambert/Parke-Davis Pharmaceutical Research

The most compelling reason for using a permanently-stored SAS database is the relative ease with which one can be created, maintained, and used by programming and nonprogramming personnel alike. Using SAS with its data-handling, statistical, and report-generation features means using only one major software package for many users, an advantage that is hard to surpass in the data-processing environment. With the addition of procedures defining the steps from data-entry to data retrieval, the application of fairly straightforward, user-designated CLISTS, and the addition of permanently stored SAS format libraries, a nearly total SAS "system" can be devised which suits the needs of multiple, diverse users.

To give some historical perspective, three years ago we created our first SAS database for use in pharmaceutical, clinical research. This initial database was built for a relatively small study of short duration. Raw data were keyed from case report forms to disk with some input editing and data correction occurring at entry. Then, on a routine basis, these data were released into a master file and then processed by a nonSAS program which did further data editing and created five tapes. Because of the editing/data correction cycle included in this preprocessing program, data were picked up at this point in this initial endeavor in order to avoid having to further duplicate efforts by doing any preprocessing of data.

The final database design decided upon consisted of six subdatasets which were to be permanently stored and accessed at retrieval time by a "link" program which would merge them together on a temporary basis. It was felt that this approach would provide maximum efficient use of computer space. These six subdatasets were related by key, identifying parameters. These common variables provided sort and merge criteria for the eventual linking of all datasets into one. In order to maintain efficient storage of data, the LENGTH statement was used extensively to cut down on the default storage length of eight bytes per variable.

The actual loading of raw data into the SAS database consisted of reading the data from tape in one step, thus, immediately converting it all to SAS datasets. Next, a series of merges brought related types of data together. Special calculations were done during this series of merges to provide additional variables for use during data retrieval. Most notable of these was the calculation of a study day (day from start of study) for every date recorded. This field was then stored along with the other variables in the dataset. Some additional editing was also done during these merges using SAS's ability to track from which dataset an observation comes. This helped to find nonmatches in the merging process.

Responsible documentation is key to good system development, and ways were sought to let SAS do as much as possible in this area. Documentation of the datasets was achieved through the use of the LABEL statement. Judicious use of the 40 characters allowed in a label statement provided key information on the raw source of the variable along with other pertinent information. These labels, along with the remaining information printed with the PROC CONTENTS procedure provides adequate documentation on the contents of the database.

Further system documentation was obtained by using PROC DATASETS in the final creation step. This final step occurs just prior to permanently storing the datasets. If the SAS program which creates the datasets has completed successfully (a condition code check is performed to ascertain this), then a SAS "housekeeping" program is run which, using first PROC DATASETS, then PROC COPY, deletes the old version of the datasets and copies into the database the new version. By using the PROTECT= option when the final datasets are created, the datasets cannot be deleted or altered in anyway without use of the appropriate password.

With the subdatasets permanently stored in the master database, the next step was the design of a SAS program to link the datasets together for purposes of data retrieval. This program was the key to any attempt to access the database. It consisted of another series of merges on key identifying variables and linked the subdatasets together. The final completely merged dataset consisted of one observation per patient per clinic visit. For this application it was felt that for retrieval and report-generation this was the most reasonable.

Handling text material (comments) was a somewhat difficult aspect of the database design. It was decided to simply store the comment records line-by-line a subdataset in the permanent SAS database. Then, when the linking program was run, these comments were processed for retrieval. This was achieved by the creation of two macros which when called in any retrieval program, would scan the appropriate records and format the text into 90-character blocks for printing.

The initial database along with the linking procedure was quite successful in the first study for which it was designed. But, as happens with most new methods, this model did not fit the next application quite as well. The basic design was fine, but the next project involved a rather lengthy and voluminous study. Procedural problems arose around the time required for the link program to run before one could gain access to the data. This was especially troublesome in the interactive environment and required us to make some major revisions in the SAS database. The final linking of the subdatasets was moved back into the database program itself and the entire merged dataset now became the permanent database. The linking, merging, and preparation of comments remained in what was previously the link program, a program which was now used only if one needed to have access to comments as well as data. Thus, the link program has come to mean comment preparation. There was some increase in storage required as some of the expansion of repeating fields was also done in the link; now more fields were stored permanently. No real measurement of this increase exists, though one might well argue the virtue of efficiency of human resources versus machine resources.

Recently, we have modified our approach even more, to use the features of SAS datasets as a potential relational database. Still using the subdatasets approach, instead of merging all datasets together for access we now leave them as separate datasets with keyed parameters which can be used for merging, if needed. This has helped to solve problems with variables which repeat within an observation. These variables are now held together in a dataset by sequence number, which allows as many repeats as are necessary. Typically, in the clinical research application this involves medication and adverse reaction data,

In addition to continuing to modify the design of our databases, we have rounded-out the system by adding a permanently-stored format library, special backups and user-CLISTS which give both programming and nonprogrammng users easy access to data interactively and in batch. The CLISTS are written with nonprogramming users in mind, consisting of a series of prompts which elicit the appropiate information from the user. These CLISTS allocate appropriate files, including the format library, contain JCL required to submit jobs in batch, and even create tape backup upon request.

We now support approximately 20 SAS databases representing the various stages of design development, with the most recent ones the most fully developed. We continue to look for ways to provide users with the best possible set of tools with which to work and that means finding more ways to use SAS.