Indexing and Compressing SAS® Data Sets: How, Why, and Why Not

Andrew H. Karp
Sierra Information Services, Inc.
San Francisco, California USA

Many users of SAS System software, especially those working with "large" data sets, are often confronted with two challenges: a) how can the data set be made 'smaller,' without deleting important variables and/or observations? and, b) can the time required to retrieve subsets of these observations be reduced? Recent releases of the SAS System contains tools which address both of these problems, and this paper discusses how they can be used appropriately to accomplish the desired results. The paper also suggests instances where either compression or indexing are not the appropriate steps to take.

At the outset, it should be understood that compression and indexing SAS data sets are two separate concepts. At times it is appropriate to apply an index to a data set, but not compress it, or vice versa. In many instances, as will be discussed below, inappropriate application of one or both of these tools to a SAS data set may have results contrary to those desired. As we will see, it is possible to increase the size of a data set by use of data set compression tools. It is also possible to increase the amount of computing resources (e.g., time) required to find observations in a data set if an index is applied improperly. This paper will hopefully explain when each tool is potentially appropriate and demonstrate the benefits and drawbacks from both.

Part One: Data Set Compression

A SAS data set contains three (and, perhaps, four, see below) parts: descriptor portion, variables and observations. The descriptor portion serves as the "header" or "roadmap" to the data set, and tells the SAS System supervisor where the variables are located. It also contains information such as when the data set was created, the number of observations and variables, if it has been sorted (and by which variables), the variable labels and the names of formats which have been applied to the variables. PROC CONTENTS, in BASE SAS software, can be used to display the descriptor portion in the SAS output window.

The length of each variable is established when the data set is created. The length of an observation (in bytes) is simply the sum of the byte lengths of the individual variables. It is possible that the byte length of a variable is 'longer' than that which is required to hold its value. This is particularly the case where numeric variables (with a default byte length of 8) hold small values, and where 'long' character variables have blank spaces in them. Since the variable length is the same of each variable in each observation of the data set, if follows that all observations have the same length in a SAS data set. It is therefore easy to conceive of the variables and observations in a SAS data set as forming a "square" or "rectangle."

The SAS System's data set compression option can be used to 'squeeze out' or remove blank spaces and/or repeating values in character variables. When this tool is used, the SAS System does not 'honor' distinctions between each variable in an

observation; rather, the SAS System looks at the entire observation and tries to compress it, rather than compress the individual variables. Once the compression is accomplished, the SAS System adds a small piece of information, or a 'tag' at the start of each observation containing information on how to 'un-compress' the data set when it will be used by a SAS procedure or another data step. After compression, the variables and observations are no longer resemble a square or rectangle, but are "jagged."

After compression, the observations no longer have uniform length, and the SAS System must 'un-compress' the data set every time it is used. CPU (central processing unit) resource requirements are greater when the SAS System reads a compressed rather than a non-compressed data set.

Since compression only 'works' on character variables, SAS data sets composed of only numeric variables, or those with numeric variables and just a few, small (as measured in bytes) character variables are not good candidates for compression. This is the case because the SAS System data set compression tools only operate on character variables. If the data set does not contain any character variable, or the values of the data set's character variables do not have many repeating or blank spaces, there is very little for the compression tool to do. But, the "tag" will still be added to each observation, thus increasing the size of the observation, and hence the overall size of the data set.

Whether or not the compression "works," at least from a data set size reduction standpoint, therefore hinges on how many character variables are present in the data set, and the extent to which these variables contain blanks and/or repeating values. Since SAS System compression tools operate on the entire observation, rather than on the individual variables, placing character variables adjacent to one another in the data set may yield additional opportunities to reduce the size of the data set than if these variables are spread among numeric variables.

How to Compress a SAS Data Set

There are two methods by which to implement data set compression in the SAS System. Perhaps the most common is to add the **COMPRESS=YES** *data set option* in the data step. This option is placed in parentheses adjacent to the name of the new data set being created. The other method is to invoke the COMPRESS=YES *SAS System option*. When invoked, this option will cause ALL data sets created in that SAS session to be compressed. In my opinion, the first option is preferable as it gives you explicit control over which data set(s) in the program will be compressed and which one(s) will not. From an efficiency standpoint (after all, CPU resources are required to carry out the compression), explicit invocation of the SAS System's compression tool means that computing resources are not needlessly squandered.

Results of Data Set Compression

Following data set compression, the SAS System places a note in the SASLOG showing the results (percentage decrease (or increase) in data set size) resulting from this action. Users should examine this note very carefully and make sure that the data set is in fact smaller after compression. If it is not, other tools (see below) can be employed to reduce the size of the data set. If possible, compressing a random subset of the larger data set should be attempted; the results will give a good idea, at least in percentage terms, of how much smaller the data set will be after compression is applied. If no (or very small) reduction in data set size occurs when the test data set is compressed, then you should question whether compression is

an appropriate tool to apply to the 'main' data set.

As mentioned earlier, each time the SAS System reads (e.g., applies a Procedure or another Data Step) to a compressed data set, the data set must be uncompressed. Users have no control over whether or not the data set is uncompressed; the SAS System does this automatically. Additional CPU resources are required to uncompress the data set before it can be used again by the SAS System

Deciding whether or not to compress a data set therefore hinges on several issues: a) how often is the data set going to be used? Infrequently used data sets are better candidates for compression than data sets that are used often. b) how many (and where are the) the character variables in the data set? Data sets without 'long' character variables are poor candidates for compression.

Using Other File Compression Tools with SAS System Data Sets

Users of the SAS System's Release 6.12 for Windows 95 may want to consider using proprietary data set compression tools such as PKZIP™ or WinZIP™ on their SAS data sets they wish to archive. These products are often quite effective in reducing the size of SAS data sets. Keep in mind, however, that the SAS System cannot read a data set which has been compressed using these products, so you will have to 'unzip' the data set before the SAS System can use it.

A Powerful Alternative to Data Set Compression: The LENGTH Statement

As mentioned above, there are several situations where using SAS System data set compression tools are not indicated. In these instances, consider using the LENGTH Statement in the data set to explicitly set the byte length of variables. The LENGTH Statement is particularly effective with numeric variables (and, remember, the SAS System's compression tools only work with character, not numeric variables) that take on relatively small values. The SAS System Companion for your specific operating system should be consulting for the largest integers which can be stored in a specific number of bytes. In general, however, using the LENGTH Statement to set the byte lengths of numeric variables, with or without subsequent compression of the character variables, often yields significant data set size reductions. More specifically, more observations can be fit it a single data set memory "page," which can reduce the number of I/O (Input/Output) or EXCP (Executive Channel Program) operations required to read a SAS data set.

Part Two: Using an INDEX in the SAS System

By default, the SAS System reads the observations in a data set sequentially; that is, one at a time in the order they are arranged in the data set. When a subset of observations from a data set is required, users usually specify a WHERE clause, WHERE data set option, or "Subsetting IF Statement" in a Data Step to "select" or "filter" the observations of interest. Without an INDEX on the variable(s) of interest, the SAS System will examine *each* observation in the data set, one at a time, to see if they meet the conditions of interest.

An INDEX is a physical file structure that serves as an adjunct to a SAS data set. Like the index pages at the rear of a book, an INDEX can be used to locate observations based on the values of the variables upon which an INDEX has been created. These variables are often called KEY or INDEX variables. When an INDEX has been created (see below), the SAS System can use it to extract those observations for which the values of the index variable meet the

conditions of interest *without* having to read each observation sequentially. Rather, the index structure allows the SAS System to "go right to" the observations of interest.

Care must be taken to select the appropriate variable(s) when creating an index. In general, it is preferable to select those variables whose values "discriminate well" amongst observations in the data set. In other words, select variables whose values help identify small subsets of the data set. Using PROC FREQ or PROC UNIVARIATE to assess the distribution of values of a "index candidate variable" is always recommended; blind application indexes to variables which do a poor job of discriminating among observations leads to unnecessary CPU and other resource utilization.

In general, SAS Institute recommends that indexes be considered for those variables which: a) have values which are frequently used in WHERE clauses, WHERE data set options, or subetting IF statements; b) less than about 25% of the observations have the same value of the index variable; and, c) the values of the variables are approximately uniformly distributed. Also, small data sets (three or few pages of memory) are usually not good candidates for indexing.

Creating an INDEX in the SAS System

Indexes are created using the INDEX data set option in a Data Step or the INDEX CREATE option in PROC DATASETS. Users can create multiple indexes (i.e., two or more index variables) in a single invocation of either the INDEX data set option or PROC DATASETS. Composite indexes are also permitted.

Using the INDEX to Select Subsets of Observations

Once the INDEX has been created, the SAS System will determine whether or not is should be used to extract observations from a data set. Users have no control over when an INDEX will be utilized or not utilized. Adding the MSGLEVEL=(I) option to your program will result in a message in the SASLOG notifying you if the index was utilized.

The SAS System will consider using the index when WHERE clause or WHERE data set option indicate that a subset of observations, based on the value(s) of variables upon which a simple or composite index has been applied, is desired. Assuming you have chosen variables whose values provide are able to 'discriminate' among observations, the amount of CPU required to retrieve the desired subset should be reduced. Additional CPU savings are usually obtained in the data set is sorted by the values of the index variables prior to applying the index itself.

While indexes are often intuitively appealing, users should also consider several factors that may militate against their use in certain situations. First, the index becomes the fourth part of a SAS data set, thus increasing its size. Creating indexes on rarely used in WHERE clauses will increase the size of the data set unnecessarily. Second, CPU resources are required to create the index. Assume that one minute of CPU time is required to apply an index to a data set, and that 10 seconds of CPU time is saved each time a WHERE clause is applied to the data set following index application. Clearly, the index will "pay for itself" in reduced CPU utilization only if it is used seven or more times. Lastly, CPU resources are required to rebuild the index each time observations are added to (or deleted from) a data set. If the data set is frequently modified, the CPU requirements to constantly re-build the index may exceed

any savings gained in extracting subsets from it.

Indexing should therefore, in my opinion, only be considered for those large data sets which are relatively "stable" relative to the number of times a WHERE clause will be applied to them to obtain a subset of observations. In addition, only those variables whose values 'discriminate among observations' and, additionally, are used frequently in a WHERE clause should be used in creating indexes.

Match-Merging using an INDEX

Indexes can also be used in a Data Step to perform efficient match-merging of two data sets. This requirement often occurs when a small number of observations in one data set is to be merged on a common variable in a larger data set, and only those observations with a common value of the matching variable are desired in the new data set. Before the availability of the index facility in SAS System software, the only tool available to SAS programmers to accomplish this type of task was to perform a match-merge in a data step between the two data sets with a subsetting IF statement restricting output of just the desired observations.

While this approach "works"—and, for many years was the only method available to SAS System software programmers—indexes often yield dramatic results when used in the data step. This approach has received extensive treatment in other papers presented in recent years at NESUG and other regional US SAS Software user conferences, as well as at SUGI and SEUGI. Those interested in learning more about these techniques may want to consult the two excellent papers presented at recent NESUG conferences listed in the references below.

Conclusions

Compression and indexing tools available in recent releases of SAS System software often provide useful methods to either reduce the size of a data set or speed retrieval of records from it. There are, however, drawbacks associated with each which must be considered before deciding if they are appropriate for a particular programming or data management scenario.

Note:

SAS is a registered trademark of SAS Institute, Inc. in the United States and other countries. ® indicates USA registration.

The author may be contacted at:

Sierra Information Services, Inc.
1489 Webster Street
Suite 1308
San Francisco, CA 94115 USA
415/441-0702 (voice)
SierraInfo @ AOL.COM
www.SierraInformation.com

References:

Horowitz, Lisa A., *Techniques for Managing Large Data Sets: Compression, Indexing and Summarization*, Proceedings of the 1997 NorthEast SAS Users Group, pp. 30-37

Loren, Judy, *Using Indexes for Direct Access to SAS Datasets:* Proceedings of the 1995 NorthEast SAS Users Group, pp. 281-282

Moorman, Denise J. and Deanna Warner, *Indexing in the SAS System, Version 6,* SAS Institute: Observations #8, 1998. http://www.sas.com/service/doc/periodicals/obs/obswww08/index.html