SAS® GLOBAL FORUM 2019

USERS PROGRAM

APRIL 28 – MAY 1, 2019 | DALLAS, TX

3677 Reducing the Space Requirements of SAS® Data Sets without Sacrificing any Variables or Observations

Stephen Sloan
Accenture

Stephen is a Data Science Senior Principal, has presented at 20 SAS Users Group events, and has been published in professional journals.

Stephen has a B.A. in Mathematics from Brandeis University, M.S. degrees in Mathematics and Computer Science from Northern Illinois University, an MBA from Stern Business School, and a graduate certificate in Financial Analytics from Stevens Institute.

3677 Reducing the Space Requirements of SAS® Data Sets without Sacrificing any Variables or Observations

Stephen Sloan
Accenture

ABSTRACT

The efficient use of space can be very important when working with large SAS data sets, many of which have millions of observations and hundreds of variables. We are often constrained to fit the data sets into a fixed amount of available space. Many SAS data sets are created by importing Excel or Oracle data sets or delimited text files into SAS and the default length of the variables in the SAS data sets can be much larger than necessary. When the data sets don't fit into the available space, we sometimes need to make choices about which variables and observations to keep, which files to zip, and which data sets to delete and recreate later.

There are things that we can do to make the SAS data sets more compact and thus use our space more efficiently. These things can be done in a way that allows us to keep all the desired data sets without sacrificing any variables or observations.

SAS has compression algorithms that can be used to shrink the space of the entire data set. In addition, there are tests that we can run that allow us to shrink the length of different variables and evaluate whether they are more efficiently stored as numeric or as character variables. These techniques often save a significant amount of space; sometimes as much as 90% of the original space is recouped. We can use macros so that data sets with large numbers of variables can have their space reduced by applying the above tests to all the variables in an automated fashion.

3677 Reducing the Space Requirements of SAS® Data Sets without Sacrificing any Variables or Observations

Stephen Sloan
Accenture

INTRODUCTION

We often run into situations where we just don't have enough room to store the temporary or permanent SAS data set we are creating. It could be due to a crowded disk drive, very long variables, a large number of observations, new restrictions on space used by data sets, or some combination of the above.

This paper outlines a 4-step procedure that can reduce the space occupied by SAS data sets without changing any values.

If the data set is very large, it probably has a large number of variables. Since we don't want to have to hard-code the size reduction tests and implementations for each variable, we use the meta-data in the SAS file sashelp.vtable to identify the numeric and character variables and their length.

3677 Reducing the Space Requirements of SAS® Data Sets without Sacrificing any Variables or Observations

Stephen Sloan
Accenture

MODIFYING THE DATA - A FOUR-STEP PROCESS

1. **Initialize the program**
2. **Reduce the size of the numeric variables**
3. **Reduce the size of the character variables**
4. **Concatenate the data sets with numeric and character variables**

3677 Reducing the Space Requirements of SAS® Data Sets without Sacrificing any Variables or Observations

Stephen Sloan
Accenture

Step 1 – Initialize the program

Set the compression to reduce the data set's footprint. We usually use COMPRESS=BINARY, which will cause the data set to use less space when the observations have "several hundred" bytes or more, according to the SAS web site. COMPRESS=YES can be used when there are not enough variables to justify COMPRESS=BINARY, which can cause the program to run longer.

Set REUSE=YES to reduce the amount of memory used while the program is running.

Set ERRORS=0 to reduce the size of the log file.

```
OPTIONS COMPRESS=BINARY ERRORS=0 REUSE=YES;
```

Identify the input and output data sets.

Initialize the tracking data so that the size can be compared after each step

Separate the input SAS data set into two data sets, one with numeric variables and one with character variables:

```
*** Split data into numeric and character variables ***;
DATA numeric(KEEP=_NUMERIC_) character(KEEP=_CHARACTER_);
```

SET x;

RUN;

Stephen Sloan
Accenture

Step 2 – Reduce the size of the numeric variables

Find the numeric variables that are all integers or missing

```
IF var=INT(var) THEN flag /* for this variable */ =1;
ELSE flag=0;
```

Use PROC SUMMARY with the MIN function to isolate the variables that are all-integers, they will have the flag=1.

Reduce the size of these numeric variables

- Take the absolute value of the variables in each observation
- Use PROC SUMMARY with the MAX function to determine the largest absolute value of each variable
- Use the chart on the following slide to determine the minimum length for each variable

Stephen Sloan
Accenture

Step 2 – Reduce the size of the numeric variables

Length in Bytes	Largest Integer Represented Exactly	Exponential Notation	Significant Digits Retained
3	8,192	2^{13}	3
4	2,097,152	2^{21}	6
5	536,870,912	2^{29}	8
6	137,438,953,472	2^{37}	11
7	35,184,372,088,832	2^{45}	13
8	9,007,199,254,740,992	2^{53}	15

Stephen Sloan
Accenture

Step 2 – Reduce the size of the numeric variables

Change small numeric variables to character variables

The chart on the previous page shows the lowest possible length for numeric variables in different operating systems. Small numeric variables could have a smaller length if they are character variables. For example, numeric variables with a value from 0 to 9 only need a length of 1 as character variables, while they would need a length of 2 or 3 as numeric variables.

```
LENGTH old_value $ 1; /* Set the length and the character value */
SET ds(RENAME=old_value=new_value); /* Rename the numeric variable */
old_value=new_value; /* This changes the numeric to the character variable */
DROP new_value;  /* Clean-up */
```

Stephen Sloan
Accenture

Step 3 – Reduce the size of the character variables

Calculate the maximum length of each character variable

- Use the LENGTH function to get the length of each value of each character variable
- Use PROC SUMMARY with the MAX function to get the largest length for each character variable.
- Use the LENGTH statement to re-set the length of any character variable whose maximum length is less than the existing length

Stephen Sloan
Accenture

Step 3 – Reduce the size of the character variables

Convert all-digit character variables to numeric variables, as numeric variables take up less space (refer to slide 8 in this deck)

See which character variables contain only digits.

```
no_digits=COMPRESS(var,d);
length_without_digits=LENGTH(no_digits);
```

Use PROC SUMMARY with MAX. If the max of length_without_digits is 0, the variable is an all-digit character variable.

Convert the all-digit character variables to numeric variables

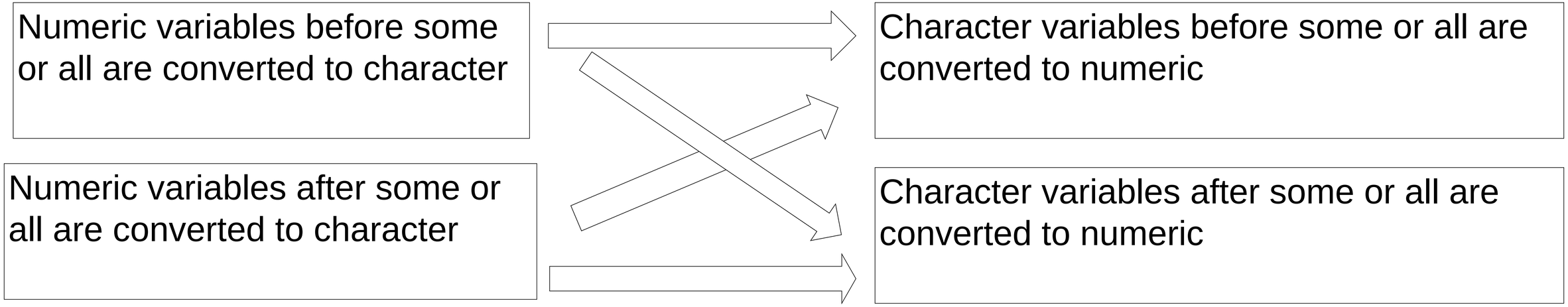
```
LENGTH old_value 3; /* Set the length of the numeric variable */
SET ds(RENAME=old_value=new_value); /* Rename the character variable */
old_value=new_value; /* This changes the character to the numeric variable */
DROP new_value;  /* Clean-up */
```

Stephen Sloan
Accenture

Step 4 – Concatenate the numeric and character data sets

Due to the different binary compression algorithms for missing numeric values and blank character values, the conversions between numeric and character variables could increase instead of decrease the size of the data sets.

Therefore we do 4 concatenations and look for the lowest amount of space:



Stephen Sloan
Accenture

- Use macro variables for all calculations
  - We might have hundreds of variables to be evaluated, so we can't identify them by name
  - Instead we use PROC CONTENTS to output a data set with the names of the variables
  - We create macro variables containing the names of the variables
- Track the reduction in size from each step of the process. A sample table is below

Statistic	Value
Initial size	57,747,456,000
After COMPRESS=BINARY	4,788,715,520
Numeric	3,762,552,832
Character	1,027,080,192
Revised Numeric	3,424,649,216
Revised Character	821,428,224
Revised Concatenated	4,242,669,568
Revised Numeric after Character Conversion	2,842,296,320
Revised Character after Numeric Conversion	826,015,744
Size of Final Data Set	3,641,311,232
Reduction	93.69%

Stephen Sloan
Accenture

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:
Email: stephen.b.sloan@accenture.com
A copy of the program will be supplied on request

Reminder:

Complete your session survey in the conference mobile app.