Paper CT-16

Manage Hierarchical or Associated Data with the RETAIN Statement

Alan R. Mann, Independent Consultant, Harpers Ferry, WV

ABSTRACT

For most of the history of computing machinery, hierarchical data has existed, and will undoubtedly persist through the next several decades. Lining up parent and child relationships by several key fields can be a challenge, and in most cases, could be served by joins or merges. This short presentation will show SAS programmers and analysts of all levels a trick to line up parent and child data using an associated key and ultimately arranging them via a surrogate key using the RETAIN statement in a data step. What perplexes many programmers is where to place the RETAIN and the logic to make use of it. This short paper clears up this issue, demonstrating a real-world application the audience can take away and start using immediately.

Introduction

In designing a SAS® DATA Step to feed a report or de-normalized table with data sorted by a common key field and arranged by subordinate values, the initial thought would be to sort or index the data and move to the next problem. However, sorting will not work when the one of the identifying keys is changed due to either a revaluation, a status change or inactivation of the record in the table. When hunting and pecking through an interactive screen is the only alternative to finding associated records, there is a better solution, which is the use of the RETAIN Statement and a good key sort to arrange all present and historic records under a single identifying key held in memory while the DATA step iterates through all associated records.

The Problem

At issue for our session is the need to arrange all records by a uniform control number:

INTAKE_NO	ORIG_CONTROL_NO
6471905	9999661
6471907	9999661
6471908	9999661
8365980	9999661
8365980	9999999 (bucket value indicating a defunct account number)
8365981	9999661

This particular problem has its real-world application at a bank in the credit card division, where defunct account numbers are given a control number with a special bucket value to indicate loss or fraud activity. Hint: when you lose a credit card, the number gets retired, not recycled.

Normally, the only way to find associated accounts would be to follow the Intake Number in a separate interactive online record system, take a copy or screen scrape, and paste it to a report of the portfolio to follow the activity of all lost and active account numbers. If your portfolio is in the thousands, or millions, this is a daunting and impossible task. Remember, you can base the sort on the Intake Number, but if you have sorted on the Control Number, all lost account numbers will sink to the bottom.

The Process

Base SAS® has a list of alternatives to hierarchically arrange data, such as:

PROC REPORT
PROC TABULATE
INTERLEAVING
MERGING
SORTING

While one could do a sort on INTAKE_NO and CONTROL_NO, remember that Intake Numbers are not uniformly sorted and unique per account, so that by itself will not pass as a solution. Merging or Interleaving (MERGE or SET) will assist, but cannot complete the job without a surrogate key, and this would be a time consuming process, causing a full stop at CONTROL_NO. PROC TABULATE or PROC REPORT could arrange the records in a pure hierarchy, but without a continuous surrogate key initiated when the portfolio was created, the lost account bucket values will always sink to the bottom. What is needed is the ability to keep the original Control Number persisting in memory until the next Intake Number is read. We would then need to add another column to indicate the true, original Control Number.

The Solution

Every SAS® DATA Step returns to the top an observation is complete. Additional logical looping may take place to determine a logical branch or situation to modify the data being read. Knowing this, we can construct our program:

a. For purposes of bookkeeping and QA, it is recommended that non-Loss Account Numbers and Loss Account Numbers be stored in separate datasets. NOTE: Sort or index all upstream datasets coming into the DATA Step by INTAKE_NO.

```
data bank.pop._lookup_table;
  set bank.pop._lookup_table1 bank.pop._lookup_table2;
```

b. Set the RETAIN Statement anywhere within the DATA Step. This author's preference is to place the RETAIN after the SET Statement.

```
retain x;
```

c. Persist the SORT in the DATA Step to ensure the loop will maintain its sort order processing.

```
by INTAKE_NO ;
 if first.INTAKE_NO then do;

 x = ORIG_CONTROL_NO;
```

d. This is the current value to persist and reside next to the Control Number to allow an auditor comparative information through the active and lost account history.

```
    end;
```

e. We then create a new, non-Loss Control Number. We then drop the x value

```
    CONTROL_NO = x;
    drop x;
run;
```

The Result

INTAKE_NO	CONTROL_NO	ORIG_CONTROL_NO
6471905	9999661	9999661
6471907	9999661	9999661
6471908	9999661	9999661
8365980	9999661	9999661
8365980	9999661	9999999 (loss bucket number)
8365981	9999661	9999661

CONCLUSION

Things to remember:

- Sort by a key you will not want to persist;
- Use the RETAIN statement anywhere in the DATA Step.
- Store the persisting value within the DO Loop, and assign a comparative field after breaking out of the loop for storage in memory when the DATA Step loops back to input the next observation.
- Use the above code in your tool kit of tricks.

REFERENCES

SAS Institute, Inc. 2011. SAS OnlineDoc® 9.3 Cary, NC.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Alan Mann
Enterprise:	Independent Consultant
Address:	370 Sylvan Lane
City, State ZIP:	Harpers Ferry, WV 25425
Work Phone:	(540) 336-7873
Fax:	
E-mail:	amann1@earthlink.net
Web:	