Paper CC14

Controlling SAS Datasets Using SAS System and Dataset Options (or I need to track the data!)

David Franklin, New Hampshire, USA

ABSTRACT

Keeping track of modifications to data in a SAS dataset is a very important task when reporting on that data. This is particularly so when databases are close to locked and are in the "Soft Lock" phase. This paper looks at a way that SAS provides of controlling who has access to the datasets for modification, and also looks at two alternatives for tracking changes, first through an audit trail and second using SAS dataset "versioning".

INTRODUCTION

Since SAS version 8 was first released the ability to do an audit trail on a SAS dataset has been available to the SAS user. With this new feature and the existing SAS options to control viewing and modification of data, SAS provides the user to have control over their most valuable asset, their data. While there are other applications out there that will track changes of SAS datasets, the features provided by SAS are robust enough for users to at least consider SAS as a possible answer to keeping track of changes to their data.

It must be noted from the onset that the idea of adding an audit trail to a SAS dataset is to track changes and modifications - rebuilding the dataset deletes any existing audit trail.

USING SAS OPTIONS

There are three SAS Options that are very good for controlling access to a SAS dataset:

- ALTER= password to alter structure
- READ= password to read
- WRITE= password to modify the data

These passwords can be set when the dataset is first created. While there are the three options, in practice on a controlled network only two of these options are used, ALTER and WRITE, as to put a READ password on a SAS dataset will require all users who what to view the data to have that password. Enabling only the ALTER and WRITE password will restrict altering the structure of the dataset, e.g. adding or deleting columns, and also restrict who can modify, add or delete data. The following example demonstrates adding ALTER and WRITE passwords to a SAS dataset that has information on medical monitors:

```
data clindata.invmon
       (alter=James write=Bond
        label='List of Monitors in Study 123-9876');
    infile cards;
    input center $ 1-3
          inv_name $ 5-19
          inv_center $ 20-39;
cards;
001 Barry Nelson   The Casino Clinic
002 Sean Connery   Dr. No's Clinic
003 George Lazenby HM Hospital
004 Roger More     The Moonraker Clinic
;
run;
```

Running a CONTENTS procedure will produce the following output indicating that the dataset has been protected with an ALTER and WRITE password:

```
                    The CONTENTS Procedure

Data Set Name: CLINDATA.INVMON          Observations:         4
Member Type:   DATA                     Variables:            3
Engine:        V8                       Indexes:              0
Created:       12:50 Monday, December   Observation Length:   38
               18, 2006
Last Modified: 12:50 Monday, December   Deleted Observations: 0
               18, 2006
Protection:    WRITE/ALTER              Compressed:           NO
Data Set Type:                          Sorted:               NO
Label:         List of Monitors
               in Study 123-9876


     -----Alphabetic List of Variables and Attributes-----

            #    Variable      Type    Len    Pos
            ------------------------------------
            1    center        Char      3      0
            3    inv_center    Char     20     18
            2    inv_name      Char     15      3
```

With the ALTER and WRITE passwords in place only those who are privy to these would be able to alter the structure of the dataset or add, delete or modify the records.

STARTING AN AUDIT TRAIL IN A SAS DATASET

An audit trail is useful if the dataset is in a state where few if any changes are needed, or it is a dataset that is like a list that is constantly changing. It is not good for the case where the dataset will be rebuilt many times over the life of the dataset – in this case the user should consider versioning which is discussed later in this paper.

You start an audit trail using the AUDIT statement in the DATASETS procedure. The following initiates the audit trail to the dataset INVMON:

```
proc datasets lib=clindata;
  audit invmon (alter=James);
    initiate;
    user_var reason $100;
  quit;
run;
```

Note that a password had to be entered to alter the dataset. A variable REASON has also been created using the USER_VAR statement so that it is possible to put a note to a record that has been changed. The output from a CONTENTS procedure call will produce a similar output as before but with four extra lines added at the bottom of the header indicating that the dataset does have an audit trail, but it does not contain any of the variables used to store the audit information:

```
                    The CONTENTS Procedure

Data Set Name:      CLINDATA.INVMON     Observations:         4
Member Type:        DATA                Variables:            3
Engine:             V8                  Indexes:              0
Created:            12:50 Monday,       Observation Length:   38
                    December 18, 2006
Last Modified:      12:58 Monday,       Deleted Observations: 0
                    December 18, 2006
Protection:         WRITE/ALTER         Compressed:           NO
Data Set Type:                          Sorted:               NO
Label:              List of Monitors
                    in Study 123-9876
Audit:              Active
Audit Before Image: YES
Audit Error Image:  YES
Audit Data Image:   YES

     -----Alphabetic List of Variables and Attributes-----
            #    Variable      Type    Len    Pos
            ------------------------------------
            1    center        Char      3      0
            3    inv_center    Char     20     18
            2    inv_name      Char     15      3
```

To get the audit trail variables for the dataset the TYPE=AUDIT option needs to be used in the CONTENTS procedure call, as shown below:

```
proc contents data=clindata.invmon (type=audit);
run;
```

with the following output:

```
                    The CONTENTS Procedure

Data Set Name: CLINDATA.INVMON.AUDIT   Observations:         0
Member Type:   AUDIT                   Variables:            10
Engine:        V8                      Indexes:              0
Created:       12:58 Monday, December  Observation Length:   204
               18, 2006
Last Modified: 13:00 Monday, December  Deleted Observations: 0
               18, 2006
Protection:    WRITE/ALTER             Compressed:           NO
Data Set Type: AUDIT                   Sorted:               NO
Label:         List of Monitors
               in Study 123-9876

     -----Alphabetic List of Variables and Attributes-----
    #    Variable          Type    Len    Pos    Format
   ------------------------------------------------------------
    5    _ATDATETIME_      Num       8    138    DATETIME19.
   10    _ATMESSAGE_       Char      8    196
    6    _ATOBSNO_         Num       8    146
    9    _ATOPCODE_        Char      2    194
    7    _ATRETURNCODE_    Num       8    154
    8    _ATUSERID_        Char     32    162
    1    center            Char      3      0
    3    inv_center        Char     20     18
    2    inv_name          Char     15      3
    4    reason            Char    100     38
```

Descriptions of the variables created for the audit trail are listed below:

Variable	Description
ATDATETIME	Date/time of modification
ATUSERID	User ID of person making modification
ATOBSNO	Observation number of record being modified (NOTE: if the option REUSE=YES is on then value will always be '0')
ATRETURNCODE	Success/Failure return code (blank if successful, error code from SAS Log if unsuccessful)
ATMESSAGE	Stores the SAS log message from SAS when the modification occurred
ATOPCODE	Modification code. Valid return codes are: DA=Record added DD=Record deleted DR=Copy of record before update DW=Copy of record after update EA=Record added failure ED=Record deleted failure EW=Record update failure.
REASON	User created variable for purpose of noting a reason.

It is possible to suspend, resume and terminate the audit trail for a dataset using the following code examples:

```
proc datasets lib=clindata;
  *Suspend an audit trail;
    audit invmon (alter=James); suspend;
  *Resume an audit trail;
    audit invmon (alter=James); resume;
  *Terminate an audit trail;
    audit invmon (alter=James); terminate;
  quit;
run;
```

While it is possible to do these actions on an audit trail within a SAS dataset these are not encouraged since it will invalidate the integrity of the dataset.

EDITING THE SAS DATASET WHEN AN AUDIT TRAIL IS ACTIVE

In the Introduction it was indicated that if a SAS Dataset was rebuilt then audit trail information pertaining to that dataset will be lost so some particular strategies have to be adopted when editing data in the SAS dataset.

The three tasks that are generally done in a SAS dataset are:

- modify an existing observation
- add a new observation
- delete an existing observation

Whenever a DATA step SET statement is used the dataset is rebuilt so using this statement in the code to modify the dataset should be highly discouraged. However using the SQL procedure with the UPDATE, INSERT and DELETE statements is one way of modifying the data while keeping the audit trail. The following examples show how to update the dataset while keeping the audit trail:

```
proc sql;

  *Correct record;
  update clindata.invmon (write='Bond')
    set inv_name='Roger Moore',
        reason='Corrected name, ref. memo 2006-12-18'
        where center='004';

  *Delete record;
  delete
    from clindata.invmon (write='Bond')
    where center='001';

  *Add new record;
  insert into clindata.invmon (write='Bond')
    set center='005',
        inv_name='Timothy Dalton',
        inv_center='HM Hospital',
        reason='New data after softlock, ref. memo 2006-12-18';
  quit;
run;
```

After the amendments the dataset has the following data:

Data After Modifications

Obs	center	inv_name	inv_center
2	002	Sean Connery	Dr. No's Clinic
3	003	George Lazenby	HM Hospital
4	004	Roger Moore	The Moonraker Clinic
5	005	Timothy Dalton	HM Hospital

To list the audit data the option TYPE=AUDIT must be used, as shown in the following SAS code and output:

```
proc print data=clindata.invmon (type=audit);
   title1 'Audit Trail After Modifying INVMON Dataset';
run;
```

Audit Trail After Modifying INVMON Dataset

Obs	center	inv_name	inv_center	reason
1	004	Roger More	The Moonraker Clinic	
2	004	Roger Moore	The Moonraker Clinic	Corrected name, ref. memo 2006-12-18
3	001	Barry Nelson	The Casino Clinic	
4	005	Timothy Dalton	HM Hospital	New data after softlock, ref. memo 2006-12-18

Obs	_ATDATETIME_	_ATOBSNO_	_ATRETURNCODE_	_ATUSERID_	_ATOPCODE_	_ATMESSAGE_
1	18DEC2006:13:27:19	4	.	dfrankli	DR	
2	18DEC2006:13:27:19	4	.	dfrankli	DW	
3	18DEC2006:13:27:19	1	.	dfrankli	DD	
4	18DEC2006:13:27:19	5	.	dfrankli	DA	

Note that in the listing from the audit trail it recorded that the record for center 004 was changed, the record for center 001 was deleted and a new record for center 005 was added. Also added was the UserID of the person who added it and the date/time it occurred.

TRACKING CHANGES WITH SAS DATASET VERSIONING

As mentioned earlier the audit information is lost when a SAS dataset is rebuilt using a statement like SET in the DATA step. However it is sometimes necessary to keep versions of a SAS dataset. This can be achieved using the SAS COPY procedure and placing that copy into another directory or placing it on another media.

SAS has another feature that is available if "versions" of a dataset have to be made instead of using the audit trail facilities. To set a SAS dataset up for "versions" the GENMAX= option is invoked, as shown in the example below:

```
data clindata.invmon
       (alter=James write=Bond
        label='List of Monitors in Study 123-9876'
        genmax=2);
    infile cards;
    input center $ 1-3
          inv_name $ 5-19
          inv_center $ 20-39;
cards;
001 Barry Nelson   The Casino Clinic
002 Sean Connery   Dr. No's Clinic
003 George Lazenby HM Hospital
004 Roger More     The Moonraker Clinic
;
run;
```

The data step above allows for three versions of the dataset to exist at any one time with number 0 as the current version, 2 as the most recent version, and 1 as the oldest version. A maximum of 999 versions, excluding the latest version, can be set for a dataset with the oldest version "dropping off" if the new version is created and the number of versions exceed that allowable.

To access a particular version of a dataset the option GENNUM= is used and can be used both in a direct and relative reference, as the following examples show:

```
*Print current version of dataset;
proc print data=clindata.invmon;
run;

*Print previous version of dataset;
proc print data=clindata.invmon (gennum=-1);
run;
```

Using this same technique it is possible to use the COMPARE procedure tool to compare versions of the dataset using the following code:

```
proc compare base=clindata.invmon data=clindata.invmon (gennum=-1);
run;
```

Versions of a SAS dataset can be deleted using the DELETE statement in the DATASETS procedure, as the following example shows:

```
proc datasets library=clindata;
  *Deletes all versions except current version;
    delete invmon (gennum=hist);
  *Deletes current version and moves previous version to current;
    delete invmon;
  *Deletes all versions of dataset;
    delete invmon (gennum=all);
  quit;
run;
```

The disadvantage of using this method is that there is no note as to why a version was needed or any record as to why an individual record was changed. However changes can be quickly seen using tools like the COMPARE procedure.

CONCLUSION

With careful planning it is possible to use SAS for tracking and securing SAS datasets. The methods looked at here are good and can track changes using their own features but it is important to know that with each method there are limitations.

CONTACT INFORMATION

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

David Franklin
16 Roberts Road
Litchfield, NH 03052
Tel/Fax 603/216-2232
Email 100316.3451@compuserve.com
http://ourworld.compuserve.com/homepages/dfranklinuk

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies.