Möglichkeiten der Nach-Verfolgung von Änderungen an SAS-Datensätzen

Ralf Minkenberg
Boehringer Ingelheim Pharma GmbH & Co. KG
Binger Str. 173
55216 Ingelheim
ralf.minkenberg@boehringer-ingelheim.com

Zusammenfassung

Das Nach-Verfolgen von Änderungen an bestimmten SAS-Datensätzen ist eine Möglichkeit, solche Datensätze kontrolliert zu beobachten und eine lückenlose Historie, wie der aktuelle Datenstand erreicht wurde, zu erstellen. In SAS stehen prinzipiell zwei Methoden, die eine solche Nach-Verfolgung erlauben, zur Verfügung.
Für jeden Datensatz kann ein Audit Trail (Protokoll) initiiert werden. In Verbindung mit einer Passwort-Vergabe (für Verändern, Lesen, Schreiben eines Datensatzes) kann so genau dokumentiert werden, wann/wer/was in einem SAS-Datensatz geändert wurde.
Als weitere Möglichkeit kann auch eine Versionierung von Datensätzen in SAS benutzt werden. Hierbei werden die verschiedenen erstellten Versionen eines SAS-Datensatzes abgespeichert und können dann später eingesehen und überprüft werden.

Schlüsselwörter: Audit Trail, Versionierung

1 Motivation

In vielen Situationen kann es sinnvoll oder sogar notwendig sein, alle Änderungen an einem Datensatz festzuhalten, um lückenlos dokumentieren zu können, wie der aktuelle Datenstand entstanden ist oder wer Änderungen vorgenommen hat. Sehr häufig kommen hierbei Datenbanksysteme zum Einsatz, die automatisch einen Audit Trail erstellen und somit diese Aufgabe ohne zusätzliche Bemühungen des Benutzers erledigen. Wird jedoch kein solches System benutzt und sollen trotzdem die Veränderungen an einem Datensatz automatisch erfasst werden, bietet auch SAS entsprechende Möglichkeiten an. Sowohl Einschränkungen der Zugriffskontrolle als auch die Erstellung eines Audit Trails oder die automatische Speicherung verschiedener Datensatzversionen sollen im Folgenden näher vorgestellt werden.

2 Zugriffskontrolle und Audit Trail

2.1 Zugriffskontrolle auf SAS-Datensätze

Auf jeden SAS-Datensatz kann durch entsprechende Datensatz-Optionen nur noch kontrolliert durch Eingabe von Passwörtern zugegriffen werden. Die Optionen müssen je-

des Mal, wenn der Datensatz derart verwendet werden soll, dass die definierte Zugriffseinschränkung greift, angegeben werden. Es stehen drei SAS-Datensatz-Optionen zur Verfügung:

ALTER=	Passwort für Strukturveränderungen
READ=	Passwort zum Lesen der Daten
WRITE=	Passwort zum Modifizieren der Daten

Das Beispiel in Listing 1 setzt 2 Passwörter auf einen Datensatz, eins zum Ändern der Dateistruktur und ein weiteres zum Ändern des Dateiinhalts. Sollen auf die Datei im Folgenden entsprechende Änderungen vorgenommen werden, ist dies nur bei Eingabe des Passworts erlaubt und möglich.

Listing 1: Beispiel für das Setzen von Passwörtern zur Zugriffskontrolle

```
DATA test1 (ALTER=MyPass1 WRITE=MyPass2);
  INPUT name $ 1-17 number $ 19-23;
  DATALINES;
. . .
```

Ob auf einem SAS-Datensatz eingeschränkte Rechte vergeben sind, lässt sich mit Hilfe von PROC CONTENTS erkennen. Bei der Ausgabe werden in der Zeile PROTECTION entsprechende Einschränkungen angegeben (siehe Listing 2).

Listing 2: Ausgabe von PROC CONTENTS

The CONTENTS Procedure

Data Set Name	WORK.TEST1	Observations	6
Member Type	DATA	Variables	2
Engine	V9	Indexes	0
Created	01/02/2017 14:53:35	Observation Length	22
Last Modified	01/02/2017 14:53:35	Deleted Observations	0
Protection	**WRITE/ALTER**	Compressed	NO
Data Set Type		Sorted	NO
Label			

2.2 Arbeiten mit einem Audit Trail

Um alle Änderungen, die an einem Datensatz vorgenommen werden, zu protokollieren, kann ein Audit Trail für einen SAS-Datensatz erstellt werden. Nachdem ein solcher definiert wurde, werden alle Änderungen, die durch die folgenden Befehle veranlasst sind, automatisch protokolliert:

- MODIFY im DATA Step
- UPDATE, INSERT, DELETE in PROC SQL oder PROC APPEND

Zu beachten ist hier, dass Änderungen durch SET oder MERGE im DATA Step nicht berücksichtigt werden. Bei der Verwendung eines Audi Trails muss daher auf diese beiden Befehle verzichtet werden.

Ein Audit Trail wird definiert mittels PROC DATASET, wie folgendes Beispiel zeigt.

Listing 3: Beispiel für die Erstellung eines AUDIT Trails

```
proc datasets library=test;
  audit daten (alter=MyPass3);
  initiate;
  user_var reason $100;
quit;
```

In diesem Beispiel wird für die SAS-Datei TEST.DATEN ein Audit Trail gestartet (INITIATE;) sowie ein Passwort für Änderungen an diesem Datensatz festgelegt (ALTER=MyPass3). Zusätzlich wird für den Audit Trail eine benutzerdefinierte Variable REASON angelegt. In dieser soll dann der Grund für eine Änderung vom Benutzer eingegeben werden.
Neben dem Starten eines Audit Trails mit dem Befehl INITIATE sind noch Befehle für das Unterbrechen des Audit Trails (SUSPEND;), für das Fortsetzen des Audit Trails (RESUME;) sowie für das endgültige Beenden des Audit Trails (TERMINATE;) erlaubt.
Nach der Initiierung eines Audit Trails stehen einige zusätzliche Variablen zur Verfügung (Tabelle 1), in der die notwendigen Informationen einer Änderung automatisch abgelegt werden. Die (wichtigsten) Änderungs-Codes sind in Tabelle 2 zusammengestellt.

Tabelle 1: Zusätzliche Variablen bei Audit Trails

Variable	**Beschreibung**
ATDATETIME	Datum/Zeit der Änderung
ATUSERID	Benutzerkennung der ändernden Person
ATOBSNO	Beobachtungsnummer der geänderten Zeile
ATRETURNCODE	Return Code
ATMESSAGE	Log-Nachricht von SAS bei Änderung
ATOPCODE	Änderungs-Code

Tabelle 2: Liste der Änderungs-Codes (_ATOPCODE_)

ATOPCODE	**Aktion**
AL	Auditing wird fortgesetzt
AS	Auditing wird unterbrochen
DA	Datensatz hinzugefügt
DD	Datensatz gelöscht
DR	Datensatz vor Update
DW	Datensatz nach Update
EA	Hinzufügen einer Beobachtung fehlgeschlagen
ED	Löschen einer Beobachtung fehlgeschlagen
EU	Update einer Beobachtung fehlgeschlagen

Ein Beispiel soll erläutern, wie das Arbeiten mit einem Audit Trail aussieht. Auf einer Kopie des bekannten Datensatzes SASHELP.CLASS wird ein Audit Trail initiiert und zusätzlich die Variable REASON definiert.

```
data class_ksfe; set sashelp.class; run;

proc datasets library=work;
  audit class_ksfe (alter=MyPass);
  initiate;
  user_var reason $1000;
quit;
```

Es werden nun einige Änderungen an dem Datensatz CLASS_KSFE durchgeführt. Zunächst wird das Alter einer Beobachtung geändert, dann eine neue Beobachtung eingefügt und schließlich eine andere Beobachtung gelöscht.

```
data class_ksfe;
  modify class_ksfe; where name = 'John';
  age = '22'; reason = "Correct John's age";
run;

proc sql;
  insert into class_ksfe
    set name = "Ralf", sex = 'M', age = 50, weight = 180,
        height = 68,
        reason = 'Just an example';
quit;

data class_ksfe;
  modify class_ksfe; where name = 'William';
  remove;
run;
```

Um die während des Audit Trails protokollierten Einträge zu betrachten, kann ein entsprechender Datensatz aus einem Datensatz mit dem Typ AUDIT und demselben Namen wie die beobachtete Datei erstellt werden:

```
data audit_ksfe;
  set class_ksfe(type=audit);
run;
```

Entsprechend der durchgeführten Änderungen erhält man einen Datensatz (read-only) mit vier Beobachtungen (zwei aus der Änderung eines Wertes und jeweils eine beim Hinzufügen bzw. Löschen) und folgenden Werten:

Tabelle 3: Beispiel eines Audit Trails Datensatz

	Name	Sex	Age	Height	Weight	reason
1	John	M	12	59	99.5	
2	John	M	22	59	99.5	Correct John's age
3	Ralf	M	50	68	180	Just an example
4	William	M	15	66.5	112	

ATDATETIME	_ATOBSNO_	_ATRETURNCODE_	_ATOPCODE_	_ATMESSAGE_
28FEB2017:16:34:22	10	.	DR	
28FEB2017:16:34:22	10	.	DW	
28FEB2017:16:34:22	20	.	DA	
28FEB2017:16:34:22	19	.	DD	

3 Versionierung („Generation Data Sets“)

3.1 Verwendung

Es ist möglich, verschiedene Versionen eines SAS-Datensatzes zu speichern. Mit diesen „Generation Data Sets“ werden archivierte Datenstände gespeichert, es sind dann mehrere Versionen eines Datensatzes verfügbar und damit kann die Entwicklung und Veränderung eines Datensatzes nachverfolgt werden. Historische Datenstände sind abrufbar und können auch als aktuelle Version wiederhergestellt werden. Standardmäßig, also ohne besondere Angaben einer Version, wird immer die aktuellste Version verwendet.
Die Versionierung von SAS-Datensätzen und das Abrufen vergangener Versionen sind nur mit SAS-Datensätzen und nicht mit SAS-Views möglich.
Hilfreich kann das Verwenden von verschiedenen Versionen eines Datensatzes z.B. sein, um ein Backup alter Datensätze einfach zu ermöglichen. Auch der Vergleich verschiedener Datensatz-Versionen ist mit versionierten Datensätzen sehr einfach.
Um mehrere Versionen eines Datensatzes ablegen zu können, kann die Datensatz-Option GENMAX verwendet werden. Zugriff auf verschiedene Versionen erlaubt die Datensatz-Option GENNUM.

Listing 4: Erstellung von Datensatz-Versionen und Vergleich der beiden letzten Versionen

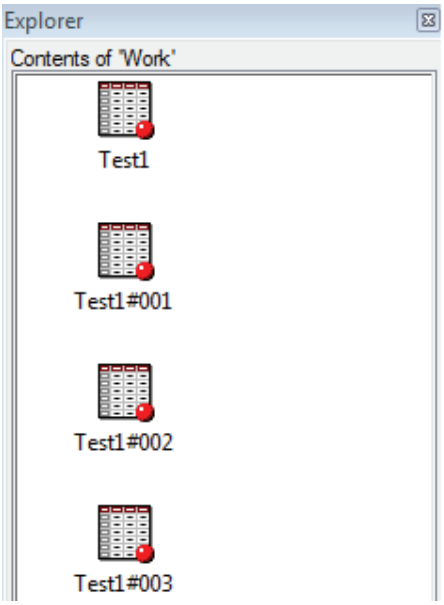
```
data test1 (genmax=5);
  input name $ 1-17 number $ 19-23;
  datalines;
  ...
proc compare base=test1 data=test1(gennum=-1);
run;
```

Wie die verschiedenen Versionen eines Datensatzes abgespeichert und benannt werden, wird deutlich, wenn folgendes Programm ausgeführt wird, in dem die Datei TEST1 mehrfach verändert wird und bis zu vier Versionen dieses Datensatzes gespeichert werden sollen:

```
data test1 (genmax=4);          (1)
  ...

data test1;                     (2)
 set test1;
 ...

data test1;                     (3)
 set test1;
 ...

data test1;                     (4)
 set test1;
 ...

data test1;                     (5)
 set test1;
 ...
```

Die erste erstellte Version von TEST1 wird „Base Version“ genannt (1). Die älteste derzeit gespeicherte Version von TEST1 ist die „Oldest Version“ (2). Die aktuelle („Newest Version“) von TEST1 ist diejenige, die ohne Angabe einer Versionsnummer verwendet wird (5). Alle gespeicherten Versionen des Datensatzes sind auch im Explorer zu sehen. Die aktuelle Version heißt TEST1, alle älteren Versionen erhalten ein #-Zeichen und eine Nummer an den Datensatznamen (hier TEST1) angehängt, z.B. TEST1#002.

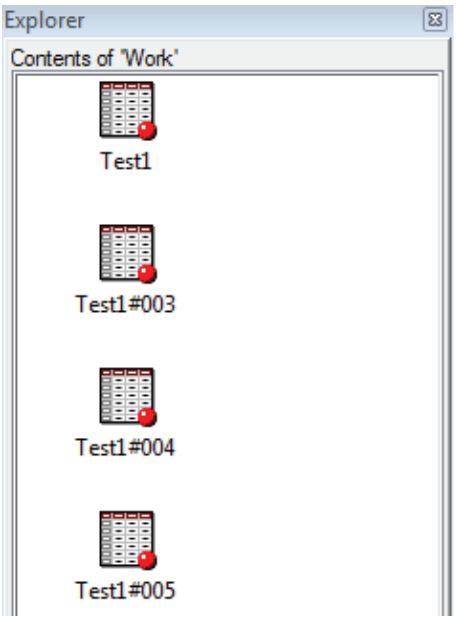
Abbildung 1: Verschiedene Versionen eines Datensatzes mit GENMAX=4



Werden nun weitere Versionen des Datensatzes erstellt, also die maximale Anzahl an Versionen überschritten, dann wird weitergezählt, aber der älteste Datensatz wird jeweils gelöscht, so dass die kleinste angehängte Zahl nicht mehr 001 ist:

```
data test1;
 set test1;
 ...

data test1;
 set test1;
 ...
```

Abbildung 2: Weitere Versionen des Datensatzes mit GENMAX=4



3.2 Löschen verschiedener Versionen

In allen Prozeduren können beliebige existierende Versionen eines Datensatzes verwendet werden. Das Löschen bestimmter (oder aller) Versionen ist sehr einfach mit PROC DATASETS möglich. Beim DELETE-Befehl kann die Datensatz-Option GENNUM verwendet werden, um die zu löschenden Versionen anzugeben. Hierbei sind folgende Möglichkeiten erlaubt:

ALL	Löschen aller Versionen inkl. der aktuellen Version
HIST	Löschen aller Versionen außer der aktuellen Version
REVERT	Löschen der aktuellen Version, die vorherige Version wird zur aktuellen Version
Integer	Löschen einer speziellen Version

Listing 5: Löschen verschiedener Versionen eines Datensatzes

```
proc datasets library=work;
  * Löscht alle Versionen außer der aktuellen;
  delete test1 (gennum=hist);
  * Löscht aktuelle Version, macht vorige zur aktuellen;
  delete test1 (gennum=revert);
  * Löscht alle Versionen;
  delete test1 (gennum=all);
run;
quit;
```

4 Zusammenfassung

Das Nach-Verfolgen von Änderungen an SAS-Datensätzen ist eine Möglichkeit, solche Datensätze kontrolliert zu beobachten und eine lückenlose Historie, wie der aktuelle Datenstand erreicht wurde, zu erstellen. In SAS stehen prinzipiell zwei Methoden, die eine solche Nach-Verfolgung erlauben, zur Verfügung.

Durch das Anlegen eines Audit-Trails kann jede Änderung an einem Datensatz protokolliert werden. Damit ist eine exakte Nachverfolgung aller Änderungen möglich.

Ist man nicht an den einzelnen Änderungen, sondern nur an den verschiedenen Versionen eines Datensatzes interessiert, kann mit der Versionierung von SAS-Datensätzen eine festgelegte Anzahl von Versionen eines Datensatzes gespeichert werden und jede dieser Versionen kann direkt im Programm angesprochen werden.