

Exploring SAS® Generation Data Sets

Kirk Paul Lafler, Software Intelligence Corporation

Abstract

Users have at their disposal a unique and powerful feature for retaining historical copies of SAS data sets. This collection of copied generation data sets, representing versions of the same data set consists of a root member name and a unique version number, is aptly referred to as a generation group. This presentation explores the power associated with generation data sets by showing users their purpose, how they are created and maintained, the approach used for accessing a specific version of a generation group, as well as other useful techniques. Simple coding examples will be illustrated to reinforce concept and application.

Introducing Generation Data Sets

Copyright 2004-2005 by Software Intelligence Corporation

Introduction / Overview

- Description of Generation Data Sets
- Purpose of Generation Data Sets
- Generation Group
- Maintaining Generation Groups
- Processing Specific Versions
- Examples



Copyright 2004-2005 by Software Intelligence Corporation

Generation Data Sets

- Are archived versions of SAS data sets
- Provide historical versions of a data set
- Similar to the aging process found in PROC DATASETS

```
PROC DATASETS LIBRARY=WORK;  
  AGE MASTER BACKUP1-BACKUP3;  
QUIT;
```

- Can only be specified for a data set (not a view)
- Stored as part of a Generation Group
- SAS processes the base version by default

Copyright 2004-2006 by Software Intelligence Corporation

Terminology

- Base version – most recent data set version
- Generation group – consists of the base version and one or more historical versions
- Historical versions – older copies of the base data set
- Newest version – data set version closest to the base version in a generation group
- Oldest version – data set version farthest from the base version in a generation group

Copyright 2004-2006 by Software Intelligence Corporation

Terminology (continued)

- Generation number – incremental number that identifies a historical version in a generation group
- GENMAX= – specifies the maximum number of versions to be created in a generation group (use with DATA statement) (max is 999)
- GENNUM= – specifies a version from a generation group to use as input to a DATA and/or PROC step process

Copyright 2004-2006 by Software Intelligence Corporation

Purpose of a Generation Data Set

- Provide versions for a data set
- Access a specific data set version in a DATA or PROC step
- Backup and recover a specific data set version
- Compare two or more data sets
 - ✓ for audit purposes
 - ✓ with PROC COMPARE
 - ✓ with DATA step – Adds, deletes, modifications

Copyright 2004-2006 by Software Intelligence Corporation

Exploring Generation Data Sets

Copyright 2004-2005 by Software Intelligence Corporation

Creating Generations

- The output data set option GENMAX= is specified to create a maximum number of historical versions to maintain for a data set, max = 999
- When GENMAX=0, no generations are created
- The maximum number of characters allowed for a data set name is 28 when the GENMAX= option is specified, otherwise it's 32 characters
- The last four characters of a generation are reserved for the version number - #nnn

e.g.: MOVIES#001

Copyright 2004-2006 by Software Intelligence Corporation

Generation Groups and Versions

- SAS appends a 4-character version number to its data set (member) name.
- When GENMAX=4 is specified and the MOVIES data set is replaced four times:

<u>Data Set Name</u>	<u>Version</u>
MOVIES	Base version
MOVIES#001	Oldest historical version
MOVIES#002	2 nd newest historical version
MOVIES#003	Newest historical version

Copyright 2004-2006 by Software Intelligence Corporation

Example – Creating Generations

- During the creation of the MOVIES data set the GENMAX=4 data set option can be specified in a DATA step to create a maximum of four retained versions as follows:

```
data libref.movies (GENMAX=4);  
    < other SAS statements >;  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

Example – Creating Generations

- When the MOVIES data set already exists, the GENMAX=4 data set option can be specified in the MODIFY statement of PROC DATASETS as follows:

```
proc datasets lib = libref;  
    modify MOVIES(GENMAX=4);  
quit;
```

Copyright 2004-2006 by Software Intelligence Corporation

Example – Inspecting CONTENTS

- Once the GENMAX= data set option is established, the CONTENTS statement of PROC DATASETS can be specified to inspect the descriptor portion of the data set as follows:

```
proc datasets lib = libref;  
  contents data = MOVIES;  
quit;
```

Copyright 2004-2006 by Software Intelligence Corporation

Example – Inspecting CONTENTS

```
proc datasets lib = libref;  
  contents data = MOVIES;  
quit;
```

Data Set Name	libref.MOVIES	Observations	22
Member Type	DATA	Variables	6
Engine	V9	Indexes	0
Created	Tuesday, March 28, 2006 01:02:48 AM	Observation Length	88
Last Modified	Tuesday, March 28, 2006 01:02:48 AM	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Max Generations	4		
Next Generation Num	1		
Label			

Copyright 2004-2006 by Software Intelligence Corporation

More on Versions

- Assuming GENMAX=4 is specified, only four generations are maintained.
- After the MOVIES data set is replaced 100 times:

<u>Data Set Name</u>	<u>Version</u>
MOVIES	Base current version
MOVIES#098	Oldest historical version
MOVIES#099	2 nd newest historical version
MOVIES#100	Newest historical version

Copyright 2004-2006 by Software Intelligence Corporation

Versions and Rolling Over

- The process of version roll over is where a version number moves from 999 to 000.
- After 1000 replacements, the newest version number is automatically rolled over to #000, e.g.: MOVIES#000.

Copyright 2004-2006 by Software Intelligence Corporation

and More on Versions

- Assuming GENMAX=4 is specified, only four generations are maintained.
- After the MOVIES data set is replaced 1001 times:

<u>Data Set Name</u>	<u>Version</u>
MOVIES	Base current version
MOVIES#999	Oldest historical version
MOVIES#000	2 nd newest historical version
MOVIES#001	Newest historical version

Copyright 2004-2006 by Software Intelligence Corporation

Selecting a Generation

- The GENNUM= data set option is used to reference a specific historical version in a generation group
- The GENNUM= data set option can be specified in a DATA and/or PROC step.

Copyright 2004-2006 by Software Intelligence Corporation

Processing the Base Version

- Using PROC PRINT with our MOVIES example, we'll specify the base version of our generation group as follows:

```
proc print data=movies; /* Base version */  
run;
```

< or >

```
proc print data=movies (gennum=0);  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

Processing the Newest Version

- Using PROC PRINT with our MOVIES example, we'll specify the newest (most recent) version in our generation group as follows:

```
proc print data = movies (gennum = -1);  
run;
```

< or >

```
proc print data = movies (gennum = 3);  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

Processing 2 Generations Back

- Using PROC PRINT with our MOVIES example, we'll specify the version two generations back from the base version as follows:

```
proc print data = movies (gennum = -2);  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

Processing 3 Generations Back

- Using PROC PRINT with our MOVIES example, we'll specify the version three generations back from the base version as follows:

```
proc print data = movies (gennum = -3);  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

Modifying Data in a Data Set

- We'll modify our data in the MOVIES data set by adding an additional minute to the movie length for the movie "Jaws" as follows:

```
data movies;  
  set movies;  
  if upcase(title)='JAWS' then  
    length=length+1;  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

Comparing 2 Data Set Versions

- To determine the changes between two data sets PROC COMPARE can be used as follows:

```
proc compare base=movies  
              compare=movies(gennum = -1);  
run;
```

Copyright 2004-2006 by Software Intelligence Corporation

PROC COMPARE Results

```
Data Set Summary
Dataset      Created      Modified  NVar  NObs
WORK.MOVIES  28MAR06:01:48:51  28MAR06:01:48:51  6     22
WORK.MOVIES#001  28MAR06:01:02:48  28MAR06:01:02:48  6     22

Variables Summary
Number of Variables in Common: 6.

Observation Summary
Observation   Base   Compare
First Obs      1       1
First Unequal   9       9
Last Unequal    9       9
Last Obs      22      22

Number of Observations in Common: 22.
Total Number of Observations Read from WORK.MOVIES: 22.
Total Number of Observations Read from WORK.MOVIES#001: 22.

Number of Observations with Some Compared Variables Unequal: 1.
Number of Observations with All Compared Variables Equal: 21.

Values Comparison Summary
Number of Variables Compared with All Observations Equal: 5.
Number of Variables Compared with Some Observations Unequal: 1.
Total Number of Values which Compare Unequal: 1.
Maximum Difference: 1.
```

Copyright 2004-2006 by Software Intelligence Corporation

Managing Generations Groups

- The DATASETS procedure provides the tools to manage generation groups.
- The following options can be specified using the DELETE statement in PROC DATASETS:
 - ✓ ALL – deletes all data sets including historical versions
 - ✓ HIST – deletes only historical versions
- Only generation groups can be copied with the COPY statement (not individual versions)
- A specific version or the entire generation group can be renamed with the CHANGE statement

Copyright 2004-2006 by Software Intelligence Corporation

In Conclusion

- Generation data sets are archived versions of SAS data sets
- Provide historical versions of data sets
- Can only be specified for a data set, not views
- Provide access to a specific data set version for processing
- SAS processes the base version by default
- Generation groups can be managed using PROC DATASETS

Copyright 2004-2006 by Software Intelligence Corporation

Acknowledgments

The author would like to thank Peter Eberhardt and Mark Tabladillo, SESUG Simple but Clever Section Co-Chairs, for accepting this paper and presentation, as well as the SESUG Leadership for their support of a great Conference.

Contact Information

Kirk Paul Lafler, a SAS Certified Professional® has been using SAS software since 1979. As a consultant and trainer with Software Intelligence Corporation, he provides consulting services, custom programming, and instructor-led and/or hands-on SAS training to users around the world. Kirk has written four books including PROC SQL: Beyond the Basics Using SAS by SAS Institute, Power SAS and Power AOL by Apress, and more than one hundred peer-reviewed articles and papers for professional journals and SAS User Group proceedings. His popular SAS Tips column appears regularly in the BASAS, HASUG, SANDS, SAS, and SESUG Newsletters and websites. Kirk can be reached at:

Kirk Paul Lafler
Software Intelligence Corporation
P.O. Box 1390
Spring Valley, California 91979-1390
E-mail: KirkLafler@cs.com



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.