CHAPTER 27

SAS Data Sets

Definition

A *SAS data set* is a group of data values that SAS creates and processes. A data set contains

- a table with data, called
  - *observations*, organized in rows
  - *variables*, organized in columns.
- descriptor information that describes such things as the number of variables, variable names, time of last file update, and the length and the format of the data.

There are two types of SAS data sets:

- a *SAS data file* contains both the data and the descriptor information. SAS data files have a member type of DATA.
- a *SAS data view* is a virtual data set that points to data from other sources. SAS data views have a member type of VIEW (See Chapter 29, "SAS Data Views," on page 455).

The term "SAS data sets" is used when SAS data views and SAS data files can be used in the same manner.

An *index* is an auxiliary file that is a summary of a SAS data set. Indexes can provide faster access to specific observations, particularly when you have a large data set.

Audit and *backup files* are auxilary files that are used to audit the changes made to a data file.

Native or *interface files* specify either files that are created by SAS, or files created by other programs.

Native files are SAS data sets that SAS creates. These files have data values and descriptor information formatted by SAS.

Interface files are files created by other programs, such as ORACLE, DB2, or SYBASE. SAS uses special engines to read and write the data. For more information about SAS multiengine architecture, see Chapter 36, "SAS I/O Engines," on page 511.

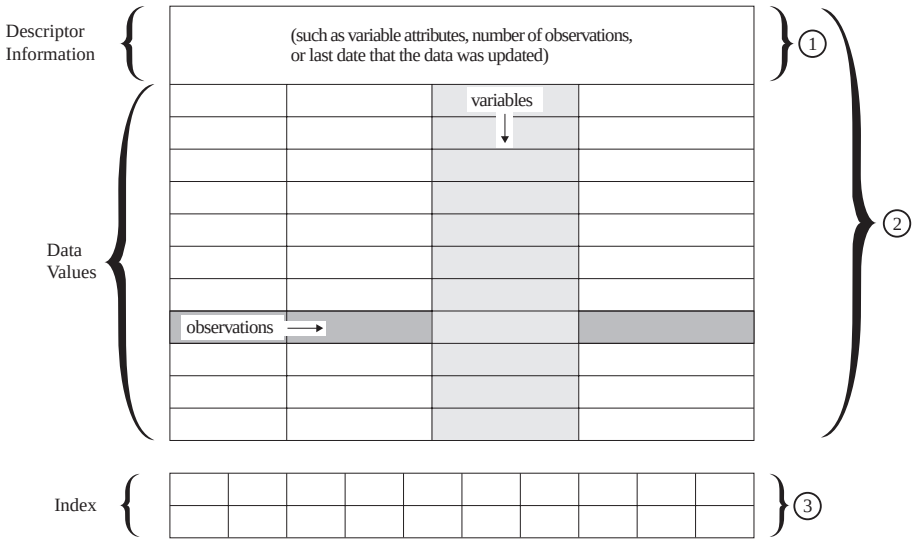
Descriptor Information

The descriptor information for a SAS data set makes the data set self-documenting; that is, each data set can supply the attributes of the data set and of its variables. Once the data is in the form of a SAS data set, you do not have to specify the attributes of the data set or the variables in your program statements. SAS obtains the information directly from the data set.

Descriptor information includes the number of observations, the observation length, the date that the data set was last modified, and other facts. Descriptor information for individual variables includes attributes such as name, type, length, format, label, and whether the variable is indexed.

The following figure illustrates the logical components of a SAS data set.

Figure 27.1 Logical Components of a SAS Data Set



The following three items correspond to the numbers in the figure above:

1 A SAS data view (member type VIEW) contains descriptor information and uses data values from one or more data sets.

2 A SAS data file (member type DATA) contains descriptor information and data values. SAS data sets may be of member type DATA (SAS data file) or VIEW (SAS data view).

3 An index is a separate file with the same name as the data set.

Data Set Names

Where to Use

You can use SAS data sets as input for DATA or PROC steps by specifying the name of the data set in

- a SET statement
- a MERGE statement
- an UPDATE statement
- a MODIFY statement
- the DATA= option of a SAS procedure
- the OPEN function.

How and When Names Are Assigned

You name SAS data sets when you create them. Output data sets that you create in a DATA step are named in the DATA statement. SAS data sets that you create in a procedure step are usually given a name in the procedure statement or an OUTPUT statement. If you do not specify a name for an output data set, SAS assigns a default name.

If you are creating SAS data views, you assign the data set name using one of the following:

- the SQL procedure
- the ACCESS procedure
- the VIEW= option in the DATA statement.

If you are using an interface library engine to access the data, the rules for assigning data set names vary according to the engine.

Note: Because you can specify them both as data sets in the same program statements but cannot specify the member type, SAS cannot determine from the program statement which one you want to process. This is why SAS prevents you from giving the same name to SAS data views and SAS data sets in the same library △

Parts of a Data Set Name

The complete name of every SAS data set has three elements. You assign the first two; SAS supplies the third. The form for SAS data set names is as follows:

libref.member-name.membertype

The elements of a SAS data set name include the following:

libref
is the logical name of a SAS data library.

table-name
is the data set name, which can be up to 32 bytes long for the base engine in Version 7. Earlier SAS versions are still limited to 8-byte names.

membertype
is assigned by SAS. The member type is DATA for SAS data files and VIEW for SAS data views.

When you refer to SAS data sets in your program statements, use a one-level or two-level name. Use a one-level name when the data set is in a temporary library, such as USER or WORK. Use a two-level name when the data set is in some other permanent library you have established. A two-level name consists of both the libref and the data set name. A one-level name consists of just the data set name.

Two-level Names

The form most commonly used to create, read, or write to SAS data sets in permanent SAS data libraries is the two-level name as shown here:

libref.data-set-name

When you create a new SAS data set, the libref indicates where it is to be stored. When you reference an existing data set, the libref tells SAS where to find it. The following examples show the use of two-level names in SAS program statements:

```
data revenue.sales;

proc sort data=revenue.sales;
```

One-level Names

You can omit the libref, and refer to data sets with a one-level name in the following form:

data set-name

Data sets with one-level names are automatically assigned to one of two special SAS libraries: WORK or USER. Most commonly, they are assigned to the temporary library WORK and they are deleted at the end of a SAS job or session. If you have associated the libref USER with a SAS data library or used the USER= system option to set the USER library, data sets with one-level names are stored in that library. See Chapter 26, “SAS Data Libraries,” on page 385 for more information on using the USER and WORK libraries. The following examples show how one-level names are used in SAS program statements:

```
data 'test3';

set 'stratifiedsample1';
```

Special SAS Data Sets

Special SAS data set names provide a means for creating null data sets and for naming and using default data sets.

Null Data Sets

If you want to execute a DATA step but do not want to create a SAS data set, you can specify the keyword _NULL_ as the data set name. The following statement begins a DATA step that does not create a data set:

```
data _null_;
```

Using _NULL_ causes SAS to execute the DATA step as if it were creating a new data set, but no observations or variables are written to an output data set. This process can be a more efficient use of computer resources if you are using the DATA step for some function, such as report writing, for which the output of the DATA step does not need to be stored as a SAS data set.

Default Data Sets

SAS keeps track of the most recently created SAS data set through the reserved name _LAST_. When you execute a DATA or PROC step without specifying an input data set, by default, SAS uses the _LAST_ data set. Some functions use the _LAST_ default as well.

The _LAST_= system option enables you to designate a data set as the _LAST_ data set. The name you specify is used as the default data set until you create a new data set. You can use the _LAST_= system option when you want to use an existing permanent data set for a SAS job that contains a number of procedure steps. Issuing the _LAST_= system option enables you to avoid specifying the SAS data set name in each procedure statement. The following OPTIONS statement specifies a default SAS data set:

```
options _last_=schedule.january;
```

Automatic Naming Convention

If you do not specify a SAS data set name or the reserved name _NULL_ in a DATA statement, SAS automatically creates data sets with the names DATA1, DATA2, and so on, to successive data sets in the WORK or USER library. This feature is referred to as the DATA*n* naming convention. The following statement produces a SAS data set using the DATA*n* naming convention:

```
data;
```

Sorted Data Sets

A sort indicator is stored with SAS data sets. The sort indicator expresses how the data is sorted. Sort information is used internally for performance improvements, for example, during index creation. For details, see the SORTEDBY data set option in the *SAS Language Reference: Dictionary* and the PROC SORT procedure in the *SAS Procedures Guide*.

Use PROC CONTENTS to view information for a data set.

Generation Data Sets

Definition of Generation Data Sets

Generation data sets
are historical copies of a SAS data set. Beginning with Version 7, you can keep multiple copies of a SAS data set by requesting the generations feature. The multiple copies represent versions of the same data set, which is archived each time it is replaced. The copies are referred to as a *generation group* and are a collection of data sets with the same root member name but with different version numbers. There is a *base version*, which is the most recent version, plus a set of *historical versions*.

You can request generations for both SAS data files and SAS data views; however, there are differences:

- a generation for a data file represents the status of that data file for both the descriptor information and the data.
- a generation for a data view represents the status of that data view for only the descriptor information. The data that the version accesses will be the current data.

Note: Generation data sets provide historical versions of a data set; they do not track observation updates for an individual data set. △

Terminology

The following terms are relevant to generation data sets:

base version
is the most recently created version of a data set. Its name does not have the four-character suffix for the generation number.

oldest version
is the oldest version in a generation group.

generation group
is a group of data sets that represent a series of replacements to the original data set. The generation group consists of the base version and a set of historical versions.

GENMAX=
is an output data set option that specifies how many versions (including the base version and all historical versions) to keep for a given data set.

GENNUM=
is an INPUT data set option that specifies which version of a data set to open. Positive numbers are absolute references to a historical version by its generation number. Negative numbers are a relative reference to historical versions. For example, GENNUM=-1 refers to the youngest version. GENNUM=0 refers to the current version.

generation number
is a monotonically increasing number that identifies one of the historical versions in a generation group. For example, the data set named AIR#272 has a generation number of 272.

historical versions
: are the older copies of the base version of a data set. Names for historical versions have a four-character suffix for the generation number, such as #003.

rolling over
: specifies the process of the version number moving from 999 to 000. When generation number reaches 999, its next value is 000.

shift down
: specifies a demotion of the base version to be the youngest version and a deletion of the oldest version, if applicable. This typically happens when you create a new base version.

shift up
: specifies a promotion of the youngest version to be the base version. This typically happens when you delete the base version.

youngest version
: is the version that is chronologically closest to the base version.

Invoking Generation Data Sets

To invoke generation data sets and to specify the number of versions to maintain, include the output data set option GENMAX= when creating or replacing a data set. For example, the following DATA step creates a new data set and requests that up to four copies be kept (one base version and three historical versions):

```
data a(genmax=4);
     x=1;
     output;
  run;
```

Once generations is in effect, the data set member name is limited to 28 characters (rather than 32), because the last four characters are reserved for a version number. When generations is not in effect (that is, GENMAX=0), the member name can be up to 32 characters. See the GENMAX= data set option in *SAS Language Reference: Dictionary*.

If a password is assigned, all files within a generation group must have the same password. SAS automatically applies any password that you assign to the base version to all of the versions in the group.

Maintaining a Generation Group

The first time a data set with generations in effect is replaced, SAS keeps the replaced data set and appends a four-character version number to its member name, which includes # and a three-digit number. That is, for a data set named A, the replaced data set becomes A#001. When the data set is replaced for the second time, the replaced data set becomes A#002; that is, A#002 is the version that is chronologically closest to the base version. After three replacements, the result is:

A	base (current) version
A#003	most recent (youngest) historical version
A#002	second most recent historical version
A#001	oldest historical version.

With GENMAX=4, a fourth replacement deletes the oldest version, which is A#001. As replacements occur, SAS will always maintain four copies. For example, after ten replacements, the result is:

A	base (current) version
A#010	most recent (youngest) historical version
A#009	2nd most recent historical version
A#008	oldest historical version

The limit for version numbers that SAS can append is #999. That is, after 999 replacements, the youngest version is #999. After 1,000 replacements, SAS rolls over the youngest version number to #000. After 1,001 replacements, the youngest version number is #001. For example, using data set A with GENNUM=4, the results would be:

999 replacements	□ A (current) □ A#999 (most recent) □ A#998 (2nd most recent) □ A#997 (oldest)
1,000 replacements	□ A (current) □ A#000 (most recent) □ A#999 (2nd most recent) □ A#998 (oldest)
1,001 replacements	□ A (current) □ A#001 (most recent) □ A#000 (2nd most recent) □ A#999 (oldest)

The following figure shows how names are assigned to generation data sets:

Table 27.1 Naming Generation Group Data Sets

Time	SAS Code	Data Set Name(s)	GENNUM= Absolute Reference	GENNUM= Relative Reference	Explanation
1	data air (genmax=3);	AIR	1	0	AIR data set created at time 1, and three generations requested
2	data air;	AIR	2	0	New AIR is created at time 2. AIR from time 1 is renamed AIR#001.
		AIR#001	1	-1	
3	data air;	AIR	3	0	New AIR is created at time 3. AIR from time 2 is renamed AIR#002.
		AIR#002	2	-1	
		AIR#001	1	-2	

Time	SAS Code	Data Set Name(s)	GENNUM= Absolute Reference	GENNUM= Relative Reference	Explanation
4	data air;	AIR	4	0	New AIR is created at time 4. AIR from time 3 is renamed AIR#003. AIR#001 from time 1, which is the oldest, is deleted.
		AIR#003	3	-1	
		AIR#002	2	-2	
5	data air (genmax=2);	AIR	5	0	New AIR is created at time 5, and the number of generations is changed to two. AIR from time 4 is renamed AIR#004. The two oldest versions are deleted.
		AIR#004	4	-1	

Processing Specific Versions of a Generation Group

Once a generation group exists, SAS processes the base version by default. For example, the following PRINT procedure prints the base version:

```
proc print data=a;
run;
```

To request a specific version from a generation group, use the GENNUM= input data set option. There are two methods that you can use:

- A positive integer (excluding zero) references a specific historical version number. For example, the following statement prints the historical version #003:

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

  Note: After 1,000 replacements, if you want historical version #000, specify GENNUM=1000. △

- A negative integer is a relative reference to a version in relation to the base version, from the youngest predecessor to the oldest. For example, GENNUM=-1 refers to the youngest version. The following statement prints the data set three versions back from the base version:

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

Table 27.2 Requesting Specific Generation Data Sets

This SAS statement ...	produces this result ...
proc print data=air(gennum=0); **proc print data=air;**	Prints the current (base) version of the AIR data set.
proc print data=air(gennum=-2);	Prints the version two generations back from the current version.

This SAS statement ...	produces this result ...
`proc print data=air(gennum=3);`	Prints the file AIR #003.
`proc print data=air(gennum=1000);`	After 1,000 replacements, prints the file AIR#000, which is the file that is created after AIR #999.

Managing Generation Data Sets

Displaying Data Set Information

A variety of statements in PROC DATASETS process a specific historical version. For example, you can display data set version numbers for historical copies using the

- CONTENTS procedure
- CONTENTS statement in PROC DATASETS.

In addition, you can display the contents for an individual historical version.

Copying and Appending Generation Data Sets

You can use the COPY statement in PROC DATASETS or the COPY procedure to copy a generation group.
For example, the following DATASETS procedure uses the COPY statement to copy a generation data group MYGEN1 from library MYLIB1 to library MYLIB2.

```
libname mylib1 'SAS-data-library1';
libname mylib2 'SAS-data-library2';

   proc datasets;
     copy in=mylib1 out=mylib2;
     select mygen1;
run;
```

You can use the GENNUM= data set option to append a specific historical version. For example, the following DATASETS procedure uses the APPEND statement to append a historical version of data set B to data set A. Note that by default, SAS uses the base version for the BASE= data set.

```
proc datasets;
   append base=a data=b(gennum=2);
run;
```

Modifying the Number of Generations

When modifying the attributes of a data set, you can increase or decrease the number of copies for an existing generation group. If you decrease the number of versions, SAS deletes the oldest version(s) so as not to exceed the new maximum. For example, the following statement can be used in a data step to change the number of copies maintained for data set A to three:

```
modify a(genmax=3);
```

You can also use the MODIFY statement of the DATASETS procedure to modify the number of generations on an existing file:

```
libname mylib SAS-data-library;
proc datasets lib=mylib;
   modify air(genmax=4);
run;
```

The previous statements modify the number of generations for MYLIB.AIR to 4. If the modification reduces the number of generations, then SAS deletes the oldest versions above the new limit.

Deleting Versions of Generation Data Sets

When deleting data sets, you can delete a specific version as well as delete an entire generation group. The following table shows the types of delete operations and effects on generation data sets when you delete versions of a generation group. For this data set, assume that the base version of AIR and two historical versions (AIR#001 and AIR#002) exist already for each command.

These SAS statements in PROC DATASETS ...	produce this result ...
delete air; **delete air(gennum=0);**	Deletes the base version and shifts up historical versions. AIR#002 is renamed to AIR and becomes the new base version.
delete air(gennum=2);	Deletes AIR#002.
delete air(gennum=-2);	Deletes the second youngest version (AIR#001). If the referenced file does not exist, this causes an error.
delete air(gennum=all);	Deletes all data sets in the generation group, including the base file.
delete air(gennum=hist);	Deletes all data sets in the generation group, except the base file.

A complete set of GENNUM= specifications is listed under the DATASETS procedure, DELETE statement, in the *SAS Language Reference: Dictionary*.

Renaming Versions of Generation Data Sets

When renaming a data set, you can rename an entire generation group:

```
change a=newa;
```

Or you can rename a single copy using the CHANGE statement in PROC DATASETS. Note that if the single copy is the base (gennum=0), the youngest historical version automatically becomes the base.

```
change a(gennum=2)=newa;
```

Tools for Managing Data Sets

To copy, rename, delete, or obtain information about the contents of SAS data sets, use the same windows, procedures, functions and options you do for SAS data libraries. For a list of those windows and procedures, see Chapter 26, “SAS Data Libraries,” on page 385.

Beginning with Version 6.12, there are functions available that allow you to work with your SAS data set. The list below gives a brief description of each function. See each individual function for more complete information.

Viewing and Editing SAS Data Sets

The VIEWTABLE window enables you to browse, edit, or create data sets. This window provides two viewing modes:

Table View
uses a tabular format to display multiple observations in the data set.

Form View
displays data one observation at a time in a form layout.

You can customize your view of a data set, for example, by sorting your data, changing the color and fonts of columns, displaying variable labels instead of variable names, or removing or adding variables. You can also load an existing DATAFORM catalog entry in order to apply a previously-defined variable, data set, and viewer attributes.

To view a data set, select the following:

Tools ► Table Editor

This brings up VIEWTABLE or FSVIEW (MVS and CMS only). You can also double-click on the data set in the Explorer window.

SAS files supported within the VIEWTABLE window are:

- SAS data files
- SAS data views
- MDDB files.

For more information, see the online help for VIEWTABLE in base SAS.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS Language Reference: Concepts*, Cary, NC: SAS Institute Inc., 1999. 554 pages.

SAS Language Reference: Concepts

ISBN 1–58025–441–1

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, November 1999