CHAPTER
3

The FSEDIT Procedure

Overview

The FSEDIT procedure enables you to edit a SAS data set one observation at a time. You can also use it to create a new SAS data set.

The procedure provides the tools for building applications for entering and editing data. An FSEDIT application provides a custom display in which each data entry field has a set of attributes that can

- □ assign an initial value to the field
- □ restrict the range of values that can be entered in the field
- □ protect the field from editing
- □ require that a value be entered in the field.

The applications can also include a SAS Component Language (SCL) program that

- □ performs sophisticated error-checking and validation of values that are entered in the variable fields
- □ displays computed values in special fields
- □ manipulates values in other SAS data sets.

The FSEDIT procedure also enables you to call the FSLETTER procedure from within an FSEDIT session. This enables you to create form letters or reports that are personalized with information from the observations that are displayed by the FSEDIT procedure.

Note: You can also open the FSEDIT window by issuing an FSEDIT command from any SAS System command line. △

CAUTION:
The FSEDIT procedure edits a data set in place. The FSEDIT procedure does not leave an unedited copy of the original. If you need to preserve a copy of the original data, be sure to make a copy of the data set before you begin editing. △

FSEDIT Procedure Syntax

Restriction: Do not use any of the other statements when you use the NEW= option in the PROC FSVIEW statement. (The VAR statement causes an error; the FORMAT, INFORMAT, LABEL, and WHERE statements are ignored.)

PROC FSEDIT <DATA=*data-set* | NEW=*data-set* <LIKE=*data-set*>>
<KEYS=*keys-entry*>
<SCREEN=*SAS-catalog*<*.screen-entry*>> | <*display-options*>
<*procedure-options*>
<*letter-options*>;

where

- *display-options* can be one or more of the following:

  LABEL

  NC=*n*

  NOBORDER

  NR=*n*

  STCOL=*n*

  STROW=*n*

  TAB=*n*

- *procedure-options* can be one or more of the following:

  ADD | NOADD

  DEBUG

  MODIFY | MOD

  NODEL

  OBS=*n*

  PRINTALL

- *letter-options* can be one or more of the following:

  LETTER=*SAS-catalog* <*.letter-entry* > <SEND=*letter-entry*>

  PRINTFILE=*fileref*

FORMAT *variable-list format* <... *variable-list-n format-n*>;

INFORMAT *variable-list informat* <... *variable-list-n informat-n*>;

LABEL *variable='label'* <... *variable-n='label-n'*>;

VAR *variable* <... *variable-n*>;

WHERE *expression*;

The PROC FSEDIT statement is required. The other statements are optional and are used as follows:

To do this	Use this statement
Associate formats with variables in the input data set	FORMAT
Associate informats with variables in the input data set	INFORMAT
Assign labels that can be used in place of variable names to identify fields in the display	LABEL
Select which variables are available to the procedure	VAR
Specify a condition or set of conditions that observations in the input data set must meet in order to be processed	WHERE

PROC FSEDIT Statement

Initiates the FSEDIT procedure.

Requirement: The FSEDIT procedure must have an input data set. By default, the procedure uses the most recently created data set as its input data set. You can use the DATA= option in the PROC FSEDIT statement to select a particular data set. If you do not specify a data set and none has previously been created in the current SAS session, the procedure terminates with an error message.

Tip: Use the SCREEN= option to identify a SCREEN entry in which to store custom features of the FSEDIT session.

Note: The LETTER= option is required if you want to generate letters and other documents using the FSLETTER procedure within the FSEDIT session.

PROC FSEDIT <DATA=*data-set* | NEW=*data-set* <LIKE=*data-set*>>
<KEYS=*keys-entry*>
<SCREEN=*SAS-catalog*<.*screen-entry*>> | <*display-options*>
<*procedure-options*>
<*letter-options*>;

where

display-options
provide control over the appearance of the FSEDIT window. All of the following options except NOBORDER and NR= are ignored if an existing SCREEN entry is specified with the SCREEN= option.

LABEL

NC=*n*

NOBORDER

NR=*n*

STCOL=*n*

STROW=*n*

TAB=*n*

procedure-options
can be one or more of the following:

ADD | NOADD

DEBUG

MODIFY | MOD

NODEL

OBS=*n*

PRINTALL

letter-options	enable you to generate letters, reports, and other documents during an FSEDIT session—things that you can also do independently with the FSLETTER procedure.

LETTER=*SAS-catalog <.letter-entry>*

PRINTFILE=*fileref*

SEND=*letter-entry*

Display and letter options are marked as such in the option descriptions that follow.

Options

Note: Most of the options that are listed here for the FSEDIT procedure are also available for the FSBROWSE procedure. However, the FSEDIT options that relate to creating and editing data sets are not applicable to the FSBROWSE procedure. △

ADD
creates a new blank observation when the procedure is initiated. The new observation is displayed for editing when the FSEDIT window is opened.

DATA=*data-set* <(*data-set-options*)>
names an existing SAS data set to be edited. By default, the FSEDIT procedure uses the most recently created data set.

If you specify both the DATA= option and the NEW= option in the same PROC FSEDIT statement, the DATA= option is ignored.

You can add a list of data set options following the data set name. The list must be enclosed in parentheses. Refer to *SAS Language Reference: Dictionary* for a listing and descriptions of data set options.

Note: The FSEDIT procedure ignores the data set options FIRSTOBS= and OBS=. All other data set options are valid. △

DEBUG
turns on the SAS Component Language (SCL) source-level debugger, which provides step-by-step assistance in resolving errors in SCL programs. This option is useful when you are creating or modifying an application that includes an SCL program.

See *SAS Component Language: Reference* for information about the SCL debugger.

KEYS=*keys-entry*
names the KEYS entry to be associated with the FSEDIT session. The KEYS entry contains function key assignments for the FSEDIT window.

Note: The KEYS= option is ignored when the SCREEN= option is also used in the PROC FSEDIT statement, unless a new SCREEN entry is being created. △

If you specify an existing SCREEN entry with the SCREEN= option, the KEYS entry name that is recorded in the SCREEN entry takes precedence over the entry that is specified in the KEYS= option.

The *keys-entry* value must be a one-level name. The search sequence for the specified entry is as follows:

1. If you also supply the SCREEN= option with the PROC FSEDIT statement, the procedure looks in the catalog named in that option for an entry that has the specified name and the type KEYS.
2. If the KEYS entry is not found in the catalog that contains the SCREEN entry, or if the SCREEN= option is not supplied, the procedure looks for an entry that has the specified name and the type KEYS in the SASUSER.PROFILE catalog (or in WORK.PROFILE if the SASUSER library is not allocated).
3. If the KEYS entry is not found in your personal PROFILE catalog, the procedure looks for an entry that has the specified name and the type KEYS in the SASHELP.FSP catalog.
4. If the KEYS entry is not found in the SASHELP.FSP catalog, the procedure searches the same sequence of catalogs for the default entry, FSEDIT.KEYS.

LABEL

(display option; ignored when an existing SCREEN entry is specified with the SCREEN= option)

specifies that variable labels, rather than variable names, are used to identify variable fields in the FSEDIT window. If a variable has no associated label, the variable name is used to identify that variable’s field.

LETTER=*SAS-catalog<.letter-entry>*

(letter option)

names a SAS catalog that contains LETTER entries, or produces copies of a specified document for all observations in the data set. If the specified catalog does not exist, it is created.

The behavior of the FSEDIT procedure depends on which form of the option you use:

- If you specify only the catalog, the procedure does not automatically produce copies of a document (unless the SEND= option is also used). This form of the option makes the EDIT, LETTER, and SEND commands available during the FSEDIT session; these commands enable you to create and print copies of a document for individual observations.

  The general form of the *SAS-catalog* value is

  <libref.>catalog-name

  If you specify only a one-level name, it is treated as a catalog name in the default library, WORK. You must specify a two-level catalog name if you want to specify a letter name.

- If you specify a letter name in addition to the catalog name, a copy of the specified document is produced for each observation in the input data set. (If a WHERE statement is used in conjunction with the PROC FSEDIT statement, then copies are produced only for observations that satisfy the specified criteria.) In this case, the procedure does not open an FSEDIT window. A pause occurs while the copies are produced; then the procedure terminates.

  The general form of the *letter-entry* value is *entry-name*<.LETTER>

If you specify a two-level letter name with anything other than LETTER as the second level (entry type), the specified entry type is ignored; LETTER is used instead.

You must specify an existing LETTER entry. The procedure terminates with an error message if the specified LETTER entry does not exist.

LIKE=*data-set* <(*data-set-options*)>

names an existing SAS data set whose structure is copied when a new SAS data set is created. (This option must be used in conjunction with the NEW= option.) When the FSEDIT NEW window is opened, the variable names and attributes of the data set that is specified in this option are displayed.

You can add a list of data set options following the data set name. The list must be enclosed in parentheses. Refer to *SAS Language Reference: Dictionary* for a list and descriptions of data set options.

MODIFY
MOD

opens the FSEDIT Menu window before opening the FSEDIT window. From the FSEDIT Menu window, you can perform tasks that modify the appearance and behavior of the FSEDIT window.

The MODIFY option is ignored (and a warning message is generated) if the PROC FSEDIT statement also includes the SCREEN= option specifying an existing SCREEN entry that is protected by a password.

NC=*n*

(display option; ignored when an existing SCREEN entry is specified with the SCREEN= option)

specifies the width in columns of the FSEDIT window. By default, the FSEDIT window occupies the maximum number of columns that is supported by the output device. The value of *n* must be at least 35. If you specify a value that exceeds the maximum number of columns available on your device, the procedure sets the window width to the maximum available width; no warning message is generated.

NEW=*data-set* <(*data-set-options*)>

creates a new SAS data set. The procedure terminates with an error message if a data set that has the specified name already exists.

When this option is used, the FSEDIT procedure begins by opening the FSEDIT NEW window, in which the names and attributes of the variables in the new data set are defined. Use the LIKE= option in conjunction with the NEW= option to initialize the FSEDIT NEW window that has the variable names and attributes of an existing data set. After the structure of the new data set is defined, the FSEDIT window is opened so that observations can be added. For details, see “Creating a New Data Set” on page 39.

You can add a list of data set options following the data set name. The list must be enclosed in parentheses. Refer to *SAS Language Reference: Dictionary* for a list and descriptions of data set options.

NOADD

prevents users from adding new observations to the data set. When you use the NOADD option, the ADD and DUP commands are disabled in the FSEDIT window, but users can still edit existing observations. When you use the NOADD option in conjunction with the SCREEN= option, the NOADD option takes precedence over the parameter setting that is specified in the SCREEN entry.

The FSEDIT procedure also provides a parameter that you can use to disable the ADD and DUP commands. Refer to “Modifying General Parameters” on page 58 for details.

NOBORDER

suppresses the sides and bottom of the FSEDIT window's border in a character-based display environment.

Note: This option is ignored in graphical windowing environments. △

When this option is used in a supported display environment, text and fields can appear in the columns and row that the border normally occupies.

When the NOBORDER option is used in conjunction with the SCREEN= option, the window size that is specified in the SCREEN entry is ignored. The FSEDIT window always occupies the maximum possible number of rows and columns when the NOBORDER option is specified.

NODEL

prevents users from deleting observations from the data set. When you use the NODEL option, the DELETE command is disabled in the FSEDIT window, but users can still edit existing observations. When you use the NODEL option in conjunction with the SCREEN= option, the NODEL option takes precedence over the parameter setting that is specified in the SCREEN entry.

The FSEDIT procedure also provides a parameter that you can use to disable the DELETE command. Refer to "Modifying General Parameters" on page 58 for details.

NR=*n*

(display option)

specifies the height in rows of the FSEDIT window. By default, the FSEDIT window occupies the maximum number of rows that is supported by the output device. The value of *n* must be at least 10. If you specify a value that exceeds the maximum number of rows available on your device, the procedure sets the window height to the maximum available height; no warning message is generated.

Note: When the NR= option is used in conjunction with the SCREEN= option, the window size that is specified in the NR= option overrides the number of rows that is specified in the SCREEN entry. △

OBS=*n*

specifies the number of the observation that is displayed when the FSEDIT window is opened. By default, the first observation in the data set (observation 1) is initially displayed. If the *n* value is greater than the number of observations in the input data set, then the last observation in the data set is displayed when the FSEDIT window is opened.

This option is not valid if a WHERE statement is used with the PROC FSEDIT statement.

PRINTALL

prints a copy of the FSEDIT display for each observation in the data set. (If a WHERE statement is used in conjunction with the PROC FSEDIT statement, then only observations that meet the specified criteria are printed.) If the SCREEN= option is also specified, the custom display format is used.

The procedure output is written to the location that has been designated for SAS System output. If you are using the SAS windowing environment, the output is written to the OUTPUT window.

When you use the PRINTALL option, the procedure does not open an FSEDIT window. A pause occurs while the output is created; then the procedure terminates.

PRINTFILE=*fileref*
PRTFILE=*fileref*
PRINT=*fileref*
DDNAME=*fileref*

(letter option)

names an external file to which documents that are produced during the FSEDIT session are written. By default, output is sent to the output destination that is specified in the FORM entry that is associated with the LETTER entry. When this option is used, output is written to the specified file instead.

You must use a FILENAME statement to assign the fileref to an external file before submitting a PROC FSEDIT statement that contains this option.

SCREEN=*SAS-catalog<.screen-entry>*

names a catalog or a specific SCREEN entry that contains information for a custom FSEDIT application, or in which the procedure can store custom features that are defined during the current session. If the specified catalog does not already exist, it is created.

The general form of the *SAS-catalog* value is

<libref.>catalog-name

If you specify only a one-level name, it is treated as a catalog name in the default library, WORK. You must use a two-level catalog name if you want to specify a SCREEN entry name.

The general form of the *screen-entry* value is

entry-name<.SCREEN>

If you specify a two-level screen name that has anything other than SCREEN as the second level (entry type), the specified entry type is ignored; SCREEN is used instead.

When the SCREEN= option is used, the procedure attempts to load a SCREEN entry when the FSEDIT session is initiated:

- If only the catalog name is provided, the procedure attempts to load an entry that is named FSEDIT.SCREEN.
- If both the catalog name and the entry name are provided, the procedure attempts to load the specified entry.

If the entry is not found, the FSEDIT session is initiated that has default FSEDIT window characteristics.

If you do not supply the SCREEN= option, any changes that you make to the display format are available only during the current FSEDIT session.

SEND=*letter-entry*

(letter option)

generates a copy of the specified document for each observation in the input data set. (If a WHERE statement is used in conjunction with the PROC FSEDIT statement, then letters are generated only for observations that meet the specified criteria.) This option is valid only when the LETTER= option is also used.

The *letter-entry* value must be the name of an existing LETTER entry in the catalog that is identified in the LETTER= option. The value should be a one-level name; the entry type LETTER is assumed. The procedure terminates with an error message if the specified entry does not exist.

When you specify the SEND= option, the procedure does not open an FSEDIT window. A pause occurs while the copies of the document are produced; then the procedure terminates.

STCOL=*n*
(display option; ignored when an existing SCREEN entry is specified with the SCREEN= option)
specifies the display column in which the leftmost column of the FSEDIT window is positioned. By default, the FSEDIT window begins at the leftmost column of the display (STCOL=1).

STROW=*n*
(display option; ignored when an existing SCREEN entry is specified with the SCREEN= option)
specifies the display row in which the top row of the FSEDIT window is positioned. By default, the window begins at the top row of the display (STROW=1).

TAB=*n*
(display option; ignored when an existing SCREEN entry is specified with the SCREEN= option)
specifies the interval that is used for column spacing when more than one column is necessary to display variables in the default screen format.

FORMAT Statement

Associates formats with variables in the input data set. *Formats* are patterns that the SAS System uses to determine how variable values are displayed. The formats can be either SAS formats or custom formats that you have defined with the FORMAT procedure.

Reminder: FORMAT statements are ignored if you use them in conjunction with a PROC FSEDIT statement that includes the NEW= option.

FORMAT *variable-list format <... variable-list-n format-n>*;

Arguments

At least one pair of the following arguments is required:

variable-list
consists of one or more variable names from the input data set.

format
is the SAS format or user-defined format to associate with the specified variable or variables.

Using the FORMAT Statement

You can use a single FORMAT statement to assign the same format to several variables or to assign different formats to different variables. You can use any number of FORMAT statements with each PROC FSEDIT statement.

Formats that are specified in a FORMAT statement take precedence over formats that are defined in the data set itself. Formats that are assigned in a FORMAT statement remain in effect only for the duration of the procedure. The FORMAT statement does not affect any format assignments that are stored in the data set.

If you are creating a new application, or if you do not use a custom FSEDIT display, then the format widths that you specify may affect the widths of the fields for the associated variables in the FSEDIT window. If you are using an existing application, the assigned formats determine how variable values are displayed, but they do not affect field widths in the FSVIEW window.

Be aware that the format you assign to a variable affects the informats you can assign with the INFORMAT statement. For example, suppose the data set that is displayed by the FSVIEW procedure contains a variable AMOUNT that is assigned the format DOLLAR10.2 but an informat of 10.2. Because of the format, values in the column for the variable AMOUNT are displayed with commas and a leading dollar sign, so the value 1250 would be displayed as $1,250.00. However, if you edit this value (for example, changing it to $1,150.00) and press ENTER, an error condition occurs. The 10.2 informat does not allow the dollar sign ($) or comma characters in entered values. An appropriate informat for this variable is COMMA., which does allow these characters.

Refer to *SAS Language Reference: Dictionary* for more detailed information about formats and on the FORMAT statement. See the description of the FORMAT procedure in *SAS Procedures Guide* for information about defining your own formats.

INFORMAT Statement

Associates informats with variables in the input data set. *Informats* are patterns that the SAS System uses to determine how values that are entered in variable fields should be interpreted. The informats can be either SAS informats or custom informats that you have defined with the FORMAT procedure.

Reminder: INFORMAT statements are ignored if you use them in conjunction with a PROC FSEDIT statement that includes the NEW= option.

INFORMAT *variable-list informat* <... *variable-list-n informat-n*>;

Arguments

At least one pair of the following arguments is required:

variable-list
consists of one or more variable names from the input data set.

informat
is the SAS informat or user-defined informat to associate with the specified variable or variables.

Using the INFORMAT Statement

You can use a single INFORMAT statement to assign the same informat to several variables or to assign different informats to different variables. You can use any number of INFORMAT statements with each PROC FSEDIT statement.

Informats that are specified in an INFORMAT statement take precedence over informats that are defined in the data set itself. Informats that are assigned in an INFORMAT statement remain in effect only for the duration of the procedure; the INFORMAT statement does not affect any informat assignments that are stored in the data set.

When using INFORMAT statements with the FSEDIT procedure, you should make sure the informats that you assign to variables are compatible with the formats for those variables. Otherwise, you will complicate the process of editing the values of the variables. For example, suppose the data set that is displayed by the FSEDIT procedure contains a variable AMOUNT that is assigned the informat 10.2 and the format DOLLAR10.2. Because of the format, values in the field for the variable AMOUNT are displayed with commas and a leading dollar sign:

```
AMOUNT:   $1,250.00
```

However, if you edit this field value (for example, changing it to $1,150.00) and press ENTER, an error condition occurs. The 10.2 informat does not allow the dollar sign ($) or comma characters in entered values. An appropriate informat for this variable is COMMA., which does allow these characters.

Refer to *SAS Language Reference: Dictionary* for more detailed information about informats and on the INFORMAT statement. See the description of the FORMAT procedure in *SAS Procedures Guide* for information about defining your own informats.

LABEL Statement

Associates descriptive labels with variables in the input data set.

Restriction: You must specify the LABEL option in the PROC FSEDIT statement in order for fields in the FSEDIT window to be identified with labels rather than with variable names. Thus, the LABEL statement is useful only in conjunction with the LABEL option.

Reminder: Labels that you specify in the LABEL statement are ignored if you specify an existing SCREEN entry in the SCREEN= option of the PROC FSEDIT statement.

LABEL *variable='label'* <... *variable-n='label-n'*>;

Arguments

At least one pair of the following arguments is required:

variable
is the name of a variable from the input data set.

label
is a string up to 256 characters long.

Using the LABEL Statement

By default, the FSEDIT procedure identifies each field in the FSEDIT window with the corresponding variable name. Labels can be longer and more informative than variable names.

Variable labels that are specified in a LABEL statement take precedence over labels that are stored in the data set itself. The LABEL statement does not affect any variable labels that are stored in the data set.

VAR Statement

Selects which variables to display and what order to display then in.

Reminder: The VAR statement is ignored if an existing SCREEN entry is specified in the SCREEN= option of the PROC FSEDIT statement.

Restriction: The VAR statement causes an error if it is used in conjunction with a PROC FSEDIT statement that includes the NEW= option.

Tip: When more than one variable name is supplied, any valid form of variable list can be used. See *SAS Language Reference: Concepts* for details about variable lists.

VAR *variable* <... *variable-n*>;

Argument

variable	is the name of a variable from the input data set.

Using the VAR Statement

By default, the FSEDIT procedure displays all of the variables in the data set and arranges the variables in the order in which they appear in the data set. You can use the VAR statement to control which variables appear in the FSEDIT window and in what order they appear. The maximum number of variable fields that are visible at any given time depends on the width and height of the FSEDIT window.

During an FSEDIT session, you can modify the display to add or remove variable fields.

WHERE Statement

Defines criteria that observations must meet in order to be displayed for editing.

Restriction: The WHERE statement causes an error if it is used in conjunction with a PROC FSEDIT statement that includes the NEW= option.

WHERE *expression*;

Argument

expression	is any valid WHERE expression that includes one or more of the variables in the input data set. Refer to the description of the WHERE statement in *SAS Language Reference: Dictionary* for details about the operators and operands that are valid in WHERE expressions.

Using the WHERE Statement

By default, the FSEDIT procedure displays all of the observations in the data set. The WHERE statement is useful when you want to process only a subset of the observations in a SAS data set. For example, to process only observations for which the value of the variable YEAR is less than 5, follow the PROC FSEDIT statement with this statement:

```
where year<5;
```

The FSEDIT procedure displays only observations that meet the specified condition(s). Observations that do not satisfy the condition(s) are not shown and cannot be edited. If no observations meet the specified condition(s), a blank observation is displayed that has observation number 0. In this case, a warning message is also displayed in the window's message line.

Conditions that are imposed by a WHERE statement (or, equivalently, by a WHERE= data set option) are called permanent WHERE clauses because they remain in effect for the duration of the FSEDIT session and cannot be canceled or modified while the procedure is active.

When you use a WHERE statement in conjunction with the PROC FSEDIT statement, the behavior of the FSEDIT procedure is affected in the following ways:

- □ The word *Subset* appears in parentheses following the FSEDIT window title to indicate that a permanent WHERE clause is in effect.
- □ Observation numbers might not be sequential because some observations might be excluded.
- □ You cannot scroll directly to a particular observation by typing the corresponding observation number on the command line.

FSEDIT Command Syntax

Tip: The FSEDIT command provides an easy way to open an FSEDIT window from any SAS System command line.

FSEDIT <? | *data-set* <*screen-name*>>

FSEDIT Command

Initiates an FSEDIT session.

FSEDIT <? | *data-set* <*screen-name*>>

Arguments

?

opens a selection window from which you can choose the data set to be processed by the FSEDIT procedure. The selection list in the window includes all data sets in all SAS data libraries that have been identified in the current SAS session (all data libraries that have defined librefs).

To select a data set, position the cursor on the desired data set name and press ENTER.

data-set

specifies the data set to be processed by the FSEDIT procedure. The general form of the argument is

<*libref.*>*data-set-name* <(*data-set-options*)>

If you omit the libref, then the default library, WORK, is assumed.

If you specify a data set that does not exist, a selection window is opened showing all available data sets. An error message in the selection window indicates that the specified data set does not exist.

If you omit this argument altogether and do not specify **?** for a selection window, then the most recently created data set (the data set that is identified in the _LAST_= system option) is selected. If no data set has previously been created in the current SAS session, a selection window is opened showing all available data sets. An error message in the selection window indicates that no default data set is available.

You can add a list of data set options following the data set name. The list must be enclosed in parentheses. The FIRSTOBS= and OBS= options are ignored; all other data set options are valid. Refer to *SAS Language Reference: Dictionary* for a list and descriptions of data set options.

screen-name

specifies a SCREEN entry that contains custom field attributes and window characteristics for the FSEDIT session. The general form of the argument is

<*libref.*>*catalog-name*<*.entry-name*<.SCREEN>>

You can specify a one-, two-, three-, or four-level name:

- If a one-level name is specified, it is treated as a catalog name in the default library, WORK. If the specified catalog does not already exist in the WORK library, it is created. The procedure attempts to load the default SCREEN entry FSEDIT.SCREEN from the catalog when the FSEDIT session is initiated. If the default entry does not exist, it is created when a MODIFY command is used during the FSEDIT session.

  Remember that all catalogs in the WORK library are deleted when you terminate your SAS session.
- If a two-level name is specified, it is treated as *libref.catalog-name.* If the specified catalog does not already exist in the specified library, it is created. The procedure attempts to load the default SCREEN entry FSEDIT.SCREEN from the catalog when the FSEDIT session is initiated. If the default entry does not exist, it is created when a MODIFY command is used during the FSEDIT session.
- If a three-level name is specified, it is treated as *libref.catalog-name.entry-name*. The entry type is assumed to be SCREEN. The procedure attempts to load the specified SCREEN entry when the FSEDIT session is initiated.

  If the specified SCREEN entry is not found, the FSEDIT session is initiated with the default display format. However, if the MODIFY command is used during the FSEDIT session, a SCREEN entry that has the specified name is created to store customization information.
- If a four-level name is specified, the fourth level (entry type) should be SCREEN. Any other value for the type is ignored; SCREEN is used instead. The procedure attempts to load the specified SCREEN entry when the FSEDIT session is initiated.

  If the specified SCREEN entry is not found, the FSEDIT session is initiated with the default display format. However, if the MODIFY command is used during the FSEDIT session, a SCREEN entry that has the specified name is created to store customization information.

Using the FSEDIT Command

The FSEDIT command can be issued in any SAS System window. When you end an FSEDIT session that was initiated with the FSEDIT command, control returns to the window from which the command was issued.

The FSEDIT command does not permit you to specify procedure options such as KEYS= and ADD. You must use a PROC FSEDIT statement rather than the FSEDIT command to initiate the procedure with these options. You must use the PROC FSEDIT statement to produce letters or other documents from the FSEDIT session because the FSEDIT command does not provide a substitute for the statement's LETTER= option. You must also use the PROC FSEDIT statement if you want to modify the procedure's behavior using the FORMAT, INFORMAT, LABEL, VAR, or WHERE statements.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/FSP® Software Procedures Guide, Version 8* , Cary, NC: SAS Institute Inc., 1999.

SAS/FSP® Software Procedures Guide, Version 8

Copyright © 1999 by SAS Institute Inc., Cary, NC, USA.

ISBN 1–58025–517–5

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

1st printing, October 1999