ABSTRACT

The features of Release 6.06 SAS/AF and SAS/FSF software provide the tools to develop data entry and retrieval applications that perform extensive field and cross-field validation, referencing and table lookup, and querying. This paper looks at some of the ways Screen Control Language (SCL) functions, selection lists, and extended tables can be used to perform these tasks.

INTRODUCTION

Imagine the ideal data entry and retrieval application. It would provide a user with easy, quick access to a wide variety of data. It would enable the user to update the data, while instilling confidence that field and cross-field validation ensure only meaningful changes. It would permit the user to query the data easily. It would also accomplish all of this in an attractive and comprehensible fashion.

This paper offers suggestions about how to implement these features. It considers validation through the use of secondary data sets, selection lists to simplify user input, and extended tables to modify and update data sets. You might find these suggestions helpful as a starting point for implementing your own data entry and retrieval system.

The discussion focuses on three SAS files: INVENTORY, a list of items in stock; INVOICE, a list of the recent orders; and MAIILLIST, a list of customers on the mailing list.

BASIC CONCEPTS

It is assumed that you are already familiar with using the BUILD procedure to create entries in a SAS/AF application or using the FSEDIT procedure with the SCREEN= option. However, in order to perform the tasks outlined here, some additional concepts are needed. These are discussed in this section of the paper.

Opening a Data Set

When you submit a DATA or PROC step to the SAS Supervisor, you do not usually need to be concerned about the process of opening or closing a SAS data set. However, many SCL functions that return information about the descriptor portion of the data set or permit you to update a SAS data set require that you open the data set yourself. This is accomplished through the use of the OPEN function.

OPEN(data-set-name, mode)

The function returns a return code whose value is zero if the data set was opened properly.

Data Vectors

Two storage areas exist to contain SCL data and SAS data, the SCL data vector and the data set data vector. Ordinarily, there is no communication between the two. In order to permit the flow of information between them, the vectors must be bridged. This can be done in several ways, but the simplest is to use the SET routine.

CALL SET(dsid)

The SET routine forms a connection between like-named data set and SCL variables, thus permitting initialization of SCL variables with values from a SAS data set, and updating of data sets from values assigned to SCL variables.

Extended Tables

An extended table is a scrollable portion of a SAS/AF PROGRAM entry that can consist of observations from a data set, items defined as elements of an array, or summary lines generated from a SAS procedure. Extended tables can be defined as dynamic, which means that the number of rows of the table can be determined by the current set of conditions. Extended tables can also be defined as selection lists and can be used to update data sets.

Some of the SCL statements used when developing extended tables are

CALL SETROW(arguments)

The SETROW routine determines the number of rows of the table or whether the table is dynamic, identifies how many items a user can select if the extended table is a selection list, and whether the selected rows will move to the top of the list. This statement is most efficiently placed in the INIT labeled section of the SCL program.

CALL ENDTABLE()

If the extended table is dynamic, there must be some way to end the display of the rows. The ENDTABLE routine, executed in the GETROW labeled section (description follows), says that the maximum number of rows in the table has been established.

GETROW:

return;

The GETROW labeled section determines what appears in the rows of the extended table. The statements in the GETROW section execute as often as necessary to display a full window of information. When the user scrolls forward or backward, the statements in GETROW again iterate to fill up a new full window. Each cycle through GETROW increases CURROW by an increment of one. CURROW is a system variable whose value is set to the number of the row being processed.

PUTROW:

return;

If the user updates values in the extended table or selects a row of the extended table, the PUTROW labeled section executes. This section might include
statements to update a data set with a user's changes or to branch to another window based on the selection of a certain row of the table. The system variable _CURROW_ indicates which row of the table has been modified.

DATA ENTRY AND MODIFICATION

SAS/AF Software Calling the FSEDIT Routine

A simple way to implement data entry and modification is to use a SAS/AF PROGRAM entry as a front-end to an FSEDIT session. For example, use a PROGRAM entry to query the user for the data set of interest (Display 1). The user can type the name of the data set into the field or type a question mark to see a list of available data sets.

```
          command =
          choose a data set for editing: BUI.INVOKE
```

Display 1 SAS/AF DISPLAY Window

When the user presses the ENTER key, the FSEDIT window is displayed (Display 2).

```
--- FSEDIT DISPLAY PROGRAM (S) ---
        command =
        Customer ID: 1332
          item 1: A-100          quantity: 1
          item 2: A-401          quantity: 2
          item 3: A-503          quantity: 1
          item 4: M-200          quantity: 
          item 5:               quantity: 
```

Display 2 FSEDIT Window

The SCL statements to call the FSEDIT routine on the data set of the user's choice and with a custom window follow. In this and in all subsequent SCL programs, if any labeled sections are not shown it is because there are no SCL statements in them.

```
        main:
        if dsname ne .blank. then do;
          screen = 'query.11[dsname].screen';
          call fseedit(dsname,screen);
        end;
        return;
```

The PROGRAM entry can be enhanced to accomplish WHERE processing on the data set so that a user sees only a subset of observations.

SAS/AF Software Calling the FSVIEW Routine

The FSVIEW procedure processes one observation at a time. To see a data set in table form, use the FSVIEW procedure. Once again, a SAS/AF window can front-end the call to the routine, permitting a user to name a data set. Again, the SAS/AF window can be enhanced to perform WHERE processing to limit the observations displayed. The previous SCL program modified to call the FSVIEW procedure with a formula is shown below.

```
        main:
        if dsname ne .blank. then do;
          formula = 'query.11[1][dsname].formula';
          call fsview(dsname,edit,formula);
        end;
        return;
```

A SAS/AF PROGRAM Entry with an Extended Table—Data Modification

Still another way of performing data entry and modification is through the use of an extended table. For data modification, observations from a data set can appear as the rows of an extended table, and any changes made to the rows can be written back to the data set. An extended table for displaying the observations in the MAILLIST data set can be set up as shown in Display 3.

```
        main:
        if dsname ne .blank. then do;
          formula = 'query.11[1][dsname].formula';
          call fsview(dsname,edit,formula);
        end;
        return;
```

Display 3 Building an Extended Table

The window variables (or their aliases) have the same name as the variables in the data set. Some of the fields can have their Protect attribute set to YES so that their values cannot be changed.

1063
When the entry is executed, the user sees the following display (Display 4).

Display 4 Extended Table Showing Observations

If the user changes the information in one of the rows of the table, the new values are written back to the data set.

The SCL program is shown below.

```sas
when the display 4).

Display 4 Extended Table Showing Observations

If the user changes the information in one of the rows of the table, the new values are written back to the data set.

The SCL program is shown below.

```sas
```

The observations are added to a temporary data set named TEMP, which is created by the NEW routine. The MAILIST data set serves as the model for NEW, thus, NEW has the same variables with the same characteristics as MAILIST. No GETROW labeled section is necessary because no information initializes the rows of the extended table.

### SELECTION LISTS

#### Selection Lists through Attributes

The SAS/AF DISPLAY windows described earlier that prompted a user for the name of a SAS data set had a data set selection list built in. By defining a type of INPUT on the ATTR window for the field, you make it possible for a user to enter the prompt character (by default, a question mark) in the field (Display 5) and see a list of all available SAS data sets.

```
```

#### Display 5 Prompt Character in the DSNAME Field

The selection list generated by the INPUT Type attribute is shown in Display 6.

```
```

#### Display 6 Selection List of Data Sets

Similarly, validating fields with other Type values and List values (discretely named or a reference to a LIST or HELP entry) also generates selection lists when a user enters the prompt character.
Selection lists make it easy for a user to make a choice, and they greatly simplify the task of validating the input.

**Selection Lists through Functions**

For more control over what appears on the selection list, use one of the many selection list functions. With them, you can select or exclude list items, customize a message, and limit a user to a specific number of choices.

For example, suppose you wanted to limit the data sets in Display 6 to INVENTRY and MAILLIST and customize the message. Change the SAS/AF DISPLAY window so that a user types an X into an action field or presses ENTER while the cursor is on a push button field (see Display 7).

![Display 7 Selection List from DIRLIST Function](image)

**Display 7 Selection List from DIRLIST Function**

The SCL program to display the selection list is shown below.

```sas
MAIN:
    array items [5] _item1-5;
    FSEINIT;
    return;
    FSETBRM:
    return;
    PSEINIT:
        dsid=opn('sagi.inventry','r');
        if dsid=0 then msg='Warning: no validation on item numbers.';
        return;
    MAIN:
        do i = 1 to 5;
            if (items[i] " '1' or items[i]='') then
                if (locatec(dsid, varnum(dsid,'itemcode'), items[i]) = 9) then
                    items[i]=datalistc(dsid, 'itemcode title', 'Please select an item code');
                end;
        end;
    return;
    PSETERM:
        rc=close(dsid);
        return;
```

An argument to the DIRLIST function enables you to name items to be included or excluded from the list. Here, SUGI.INVOICE is excluded.

**DATA VALIDATION USING A SECONDARY DATA SET**

Selection lists can also be displayed as a result of field validation. For example, open the INVENTRY data set to produce a selection list of valid item values by typing a question mark or invalid value in any of the item fields in a custom FSEDIT data entry window (see Display 8).

![Display 8 Type a Question Mark to See a Selection List](image)

**Display 8 Type a Question Mark to See a Selection List**

The SCL program is shown below.

```sas
array items [5] _item1-5;
FSEINIT:
dsid=opn('sagi.inventry','r');
if dsid=0 then msg='Warning: no validation on item numbers.';
return;
MAIN:
do i = 1 to 5;
    if (items[i] " '1' or (locatec(dsid, varnum(dsid,'itemcode'), items[i]) = 9)) then
        items[i]=datalistc(dsid, 'itemcode title', 'Please select an item code');
    end;
end;
return;
PSETERM:
rc=close(dsid);
return;
```

The FIELD function with the MODIFIED option returns a 1 if a field has been modified. If the user types a question mark, or if the user enters a value not found by the LOCATEC function's search of the ITEMCODE field of the INVENTRY data set, the DATALISTC function displays the valid item codes and titles from the INVENTRY data set. The user may tab to the desired value and press ENTER. The value is written to the field in the FSEDIT window.

**CONCLUSION**

Clearly, the examples shown here are quite simple. However, they demonstrate some powerful techniques available in SAS/AF software and SAS/FSP software, Release 6.06. For more information, please consult the following sources: SAS/FSP Software: Usage and Reference, Version 6, First Edition; SAS/AF Software: Usage and Reference, Version 6, First Edition; and SAS Screen Control Language: Reference, Version 6, First Edition. Also, the Education Division of SAS Institute Inc. offers video, computer-based, and instructor-based training in Screen Control Language in SAS/AF and SAS/FSP software.

SAS/AF, SAS, and SAS/FSP are registered trademarks of SAS Institute Inc., Cary, NC, USA.