

SAS/FSP® AND SCL: USING SCREENS WITHIN SCREENS TO CAPTURE REPEATING DATA ELEMENTS

Denise Svendsgaard Williamson
John Iwaniszek
Hoffmann Research Associates, Inc.
Chapel Hill, North Carolina

Abstract

Some data entry plans require the entry of data elements which are repeating (e.g., work history on employment applications). An alternative solution to setting up a finite number of fields on a data entry screen(s) for one main data set is to create a sub data set that will contain only the repeating (e.g., work history) information and to access this sub data set screen from the main data set screen. When you access this sub screen, you can pass variables from the main data set so that you have a way of linking each observation in the sub data set to an observation in the main data set. This process can be accomplished by using SAS® Screen Control Language (SCL) in SAS/FSP in the main data set screen(s) to call the sub data set screen(s).

Introduction

Some data entry projects require the use of repeating fields (work histories on employment applications, for example) along with non-repeating fields (such as name, social security number, and position number applied for). A convenient solution is to imbed SAS/FSP screens for the repeating data elements within a main SAS/FSP data entry screen.

The Background

In the following example, a data entry plan was designed for electronically storing employment applications. In defining the data elements of interest on the applications, it was determined that all work history records were to be input. While the application itself had space for up to 5 records of work history, applicants were told to attach additional sheets of paper, if necessary, to describe their complete work history. The data was not available for review at the data entry design stage, so no upper limits on work history were known. In an effort to capture as many work history records as there might be, a data entry plan was designed to make the work history a separate data set from the main data set which was to

contain information such as applicant name, address, and position applied for. Screen Control Language (SCL) was used to open a SAS/FSP window from the main fsedit window.

The Process

The elements that repeat were defined from the set of variables needed for analysis. For the following example, work history, or prior work experience, is the only repeating element.

Next, a data set was generated from the main data elements to contain all non-repeating variables of interest and for the sub data elements to contain all repeating variables. Note that if there were other repeating elements in addition to work history, they should form their own sub data set. In addition to containing repeating data elements, the sub data sets contained variables in common with the main data set in order to link the sub data to the main data. In the following example, the Social Security Number, Position Number Applied for, and Application Date serve as the "link" variables since they uniquely identify an application.

SAS/FSP windows were then customized for the two data sets separately using a text editor and the SCL programs were compiled separately, then integrated into the process.

Example

The following example describes how to set up programs for an employment application with work history as the repeating data element. The first program defines the variable specifications for the main data set.

```
* crscl.sas > create main screen data set;

libname sesug '\client\sesug2';

data sesug.main;
  length ssn appdate posn 8
         firstnm $15 middlei $1 lastnm $15 workhist $1;
  format ssn ssnll.
         appdate mmdyy8.;
  informat appdate mmdyy8.;
  label ssn = 'Social Security Number'
        appdate = 'Date of Application'
        posn = 'Position Number Desired'
        firstnm = 'First Name'
        middlei = 'Middle Initial'
        lastnm = 'Last Name'
        workhist = 'Is there Work History?';
run;
```

The next program sets up the specifications for the sub (work history) data set.

```
* crscl2.sas > create sub screen data set (work history);

libname sesug '\client\sesug2';

data sesug.sub;
  length ssn appdate posn 8
         employer $30 stadd $30 city $15 state $2 zipcode $5
         fromdate todte 8;
  format ssn ssnll.
         appdate fromdate todte mmdyy8.;
  informat appdate fromdate todte mmdyy8.;
  label ssn = 'Social Security Number'
        appdate = 'Date of Application'
        posn = 'Position Number Desired'
        employer = 'Employer Name'
        stadd = 'Street Address'
        city = 'City'
        state = 'State'
        zipcode = 'Zip Code'
        fromdate = 'From Date'
        todte = 'To Date';
run;
```

Below is the result of the customization of the main window.

```
Command ==> Obs 1

Application Form X-0127 - Main Screen

-----
Position Number Desired _____

Date of Application ____/____/____

Applicant SSN _____

Applicant Name _____
                First      Middle      Last
                Initial

Is there work history? _ (Y/N)
```

Below is the result of the customization of the sub window. The sub window has been customized to protect the "link" fields.

```
Command ==> Obs 1

Application Form X-0127 - Sub Screen (Work History)

-----
Position Number Desired _____

Date of Application ____/____/____

Applicant SSN _____

Employer _____
Employer Street Address _____

City _____ State _____ Zip _____

Date of Employment: From ____/____/____ To ____/____/____
```

In the main data set, a variable named WORKHIST is used to indicate whether an applicant has any work history. A response of 'Y' (Yes) prompts the program to call the sub fsedit screen. The following SCL program passes the "link" variables to a macro table. If an operator responds 'Y' to 'Is there work history?' and the field has been modified, the window for the sub data set is opened at a new observation (since the ADD option is specified in the FSEDIT CALL). Note that this program is intended for data entry only. A separate process is necessary to browse or review the data.

```
*****
* SCL program for main data set -- application form
*****

init:
return:
main:
  /* pass variables to macro table */
  call symputn('socsec',ssn);
  call symputn('appdt',appdate);
  call symputn('posno',posn);

  /* call fsedit screen if response is 'Y' to 'Is there work history?'
  if workhist='Y' and modified(workhist) then do;
    call fsedit('sesug.sub','sesug.sub','add');
  done;

end;
return:
term:
return;
```

The following SCL program is used with the sub data set. Macro values are passed from the macro table from the current observation on the main data set to the current observation on the sub data set, provided the current observation on the sub data set is new (that is, the values for Social Security Number, Position Applied for, and Application Date have not been filled). The conditional IF statement is provided in order to restrict passing of the macro values if one moves to a previously edited sub observation. After entering history records one observation at a time, ending the sub screen returns the user to the current observation of the main window and data entry may proceed.

```

.....
* SCL program for sub screen (work history)
.....
init:
  cursor employer;

  /* pass macro variables if current observation is new */
  if ssn = . then ssn = symgetn('ssnsec');
  if appdate = . then appdate = symgetn('appdt');
  if posn = . then posn = symgetn('posno');
return;
main:
  /* pass macro variables if current observation is new */
  if ssn = . then ssn = symgetn('ssnsec');
  if appdate = . then appdate = symgetn('appdt');
  if posn = . then posn = symgetn('posno');
return;
term:
return;

```

Conclusions

In addition to allowing for an unlimited number of work histories to be entered, the separation of the main and sub data set also saves disk storage space. That is, this method allows the user to take advantage of a relational data base design, avoiding unnecessary storage of non repeating fields such as those stored in the main program.

The use of SCL to open SAS/FSP windows from within SAS/FSP sessions has other applications. For example, when there is a large amount of data to input that exceeds SAS limitations, the data can be divided into sections, with one data set and one SAS/FSP window per section. It is then possible to make the transition from one section of the data to another without exiting from the SAS/FSP session.

The authors may be contacted at:

Hoffmann Research Associates, Inc.
 111 Providence Road
 Chapel Hill, NC 27514
 (919) 489-8000

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