ABSTRACT

This paper discusses how we used SAS® AF®, SAS® Screen Control Language (SCL), and SAS Structured Query Language (SQL) products available in SAS version 6.06 software to develop a user-friendly database application. It illustrates how we used short and long selection lists to implement user-interface panels that display all of the correct entries for a field, automatically validate user-input, and supply error messages. For the case where a user input field contained too many possibilities to be listed, we describe a method to combine selections from a series of lists. Lastly, we show how we used extended tables for a database application where the user needs to specify particular SAS dataset observations.

INTRODUCTION

The Johns Hopkins University Applied Physics Laboratory analyzes the performance of submarine-launched, ballistic missile weapon systems for the U.S. Navy. This analysis includes comparing computer simulations of the weapon system during a missile launch to the actual weapon system performance. One of our analysis requirements is to save the simulation data to use as a reference in evaluating future launches and for cumulative studies. We also need to be able to generate charts, tables, listings, reports, files, and statistics from the stored data. As a result, we developed a database application to automate this process of storing and manipulating data. Our main goal during the development was to make the application easy to use.

SAS software was an obvious choice to accomplish this task because it contains powerful analysis capabilities and many features that make it simple to create a user-friendly interface. This paper highlights some of the version 6.06 SAS/AF, SAS SCL, and SAS SQL options we implemented in our application called the Performance Database (PERFDB).

THE NEED FOR A USER-FRIENDLY INTERFACE

Because most of the PERFDB users are unfamiliar with SAS software and database structure, our job was to design the user-interface so the SAS and database environments are invisible to the end-user. We used SAS/AF panels and options to create database interface panels consisting of choice groups and pushbuttons that submit code for execution or display the next related screen. An example of a PERFDB panel that includes a choice group (POR Table choices) and a pushbutton (GOBACK) is shown in Figure 1. To make a selection from a choice group or activate a pushbutton, all the user has to do is move the cursor to the field and press ENTER. These tools not only make the underlying code transparent, but also make the database very easy to use by reducing the amount of input the user must type.

SHORT SELECTION LISTS

On panels where the user is required to supply data, we used the SAS/AF options to list all of the correct entries for a field, automatically validate user-input, and supply error messages. To see the list of all valid entries, the user types a question mark (the default prompt character) in the input...
field. The SAS software then displays a window in which the user can select data by placing the cursor on the desired choice and pressing ENTER. Once a selection is made, the window closes and the selection appears in the input field. When the user wishes to type in the data rather than using the selection list, SAS software checks the input against the list of valid entries. If the data is invalid, the message "HIGHLIGHTED FIELD(S) ARE INVALID" appears on the message line and the field is highlighted in the color specified for "Error Color" on the attributes (ATTR) window.

We implemented the selection list and input validation features by using the "List" option in each variable's ATTR window. For short lists, we typed the valid entries on the "List" line provided in the ATTR window (Figure 2) and verified the prompt character was set to '?' (the default) in the general attributes (GATTR) window. The resulting selection list that is displayed when the user enters a '?' in the Boat Number field is shown in Figure 3.

LONG SELECTION LISTS

Not all selection lists are concise enough to fit on the two lines provided for the "List" option in the ATTR window. Some lists also contain selections that have an imbedded blank, which is the character that separates items in the ATTR "List" option. In these cases, you cannot use the method described above for short selection lists, but must take a different approach.

The approach we took was to use a LIST catalog entry containing the long selection list. For example, to link the catalog entry UPTLSTAT.LIST to the input field we typed

```
\=UPTLSTAT.LIST\n```

into the "List" option of the ATTR window. The prompt character '?' and number of allowed selections (1) were not specified, since we used the defaults.

An added benefit of using a LIST catalog entry to specify a selection list is the availability of the customized error message. This feature allows us to make error messages more meaningful to the user by incorporating the user's own input data into the message. For example, we can set the error message to read "730 IS NOT A VALID BOAT NUMBER - PLEASE REENTER DATA", instead of just informing the user that the field is invalid. To insert the user's input data into the message, we just typed our own message on the "Error msg" line in the LIST catalog entry panel and substituted %S in place of the user's input.

MULTIPLE SCREEN SELECTION LISTS

Some of our user input fields contain too many possibilities to be displayed in one selection list. However, if it were possible to combine responses from a series of shorter selection lists, we could still make inputting these fields user-friendly. This creates two problems. First, how can you combine selections from a series of lists to build the data required for these input fields? Second, how can you implement the standard character ('?') to bring up the series of lists as well as the SAS software generated selection lists? The solution to both problems would be to call a PROGRAM catalog entry from the "List" option in the ATTR window (as
we did with the LIST entry for longer lists). Unfortunately, this is impossible since this option is limited to LIST, CBT, and HELP catalog entries. Therefore, we implemented a two-step solution: 1) override the prompt character; and 2) use SCL to call a PROGRAM catalog entry.

We overrode the prompt character ("?") in the GATTR window for each of the specific fields by using a new prompt character ("{") in the "List" option of the ATTR window (see Figure 4). This new prompt character is not meant to be input by the user. If it is input, however, a HELP catalog entry (BADFIELD.HELP) associated with the new prompt character ("{") is displayed. This screen tells the user that the input is invalid and the key to press to return from the HELP screen. In contrast, if we did not override the prompt character in this manner, a SAS software generated screen "You may enter any standard SAS character data" would be displayed when the user types a "?" in the character field. With the prompt character overridden, our SCL code can check the user input for a "?", take control, and call a PROGRAM entry.

The PROGRAM entry allows us to create a panel with a list of variables belonging to the same choice group. When the user makes a selection, the SCL code checks which item was chosen and calls the next appropriate PROGRAM entry. After the user makes a choice from each screen in the series, control is returned to the original program, and the field is set according to the series of user choices (Figures 5a-5c). This unique method makes input simple and consistent for the user.

Since we used PROGRAM entries to create the selection lists, we were unable to take advantage of
the automatic user-input validation feature of SAS/AF software. Instead, we used SCL code to check the validity of each entry. If the input is incorrect, the following SCL code displays an error message, highlights the field (e.g., OPERTYPE) in its designated error color, and positions the cursor on the field:

```
_MSG_ = 'Operation Type is invalid - ' || OPERTYPE;
ERRORON OPERTYPE;
CURSOR OPERTYPE;
```

The combination of SAS SCL and SAS/AF products creates a powerful tool to aid and control user input.

**EXTENDED TABLES**

The extended table feature of SAS/AF software saves time and energy when the user needs to specify the dataset observations to be used for an application (e.g., to include in a report). The user does not have to get a listing of the database, know what data has already been loaded into the database, or set up some condition to extract the desired observations. Instead, the user can select the observations from a table containing a subset of the dataset variables that uniquely identifies each observation.

We use the SAS SQL interface to extract this subset of data and simultaneously obtain the number of rows (NROWS) in the table from the automatic macro variable “SQLOBS” (see Figure 6). Once the SAS SQL dataset is created and the number of rows is determined, SAS SCL sets up the framework and the available options of the table with the SETROW statement. In the Figure 6 SETROW statement, the first occurrence of NROWS defines the total number of rows (lines) in the table, the second occurrence of NROWS allows any or all of the lines to be selected, and the "Y" automatically moves each line to the top of the table after it is selected.

The GETROW and PUTROW sections in the SAS/AF PROGRAM source code must be included to display and process the table. The GETROW section in the PERFDB source code (Figure 7)

```
GETROW:
/* FETCH DATA FROM CURRENT OBSERVATION */
/* & DISPLAY IT ON THE SCREEN */
RC = FETCHOBS(DATAID, _CURROW_);
RETURN;

PUTROW:
/* PROCESS SELECTED & DESELECTED ROWS */
/* IF COUNTRow ROW WAS SELECTED - ADD */
/* TO THE ARRAY LIST */
IF ISSEL(_CURROW_) THEN DO;
   ARRCNT + 1;
   SELINDX(ARRCNT) = INDXOPER;
END;
/* ELSE COUNTRow ROW WAS DESELECTED */
/* SO DELETE IT FROM ARRAY LIST */
ELSE DO;
   DO I = 1 TO ARRCNT;
      IF SELINDX{I} = INDXOPER THEN DO;
         DO J = I TO ARRCNT - 1;
            SELINDX{J} = SELINDX{J+1}
         END;
         SELINDX{ARRCNT} = 0;
         ARRCNT = ARRCNT - 1;
      I = ARRCNT + 1;
      END;
   END;
END;
END;
RETURN;
```

The GETROW and PUTROW sections in the SAS/AF PROGRAM source code must be included to display and process the table. The GETROW section in the PERFDB source code (Figure 7) reads and displays each observation from the SAS SQL dataset that is currently viewed in the extended table (Figure 8). If the user issues the UP or DOWN command, previous or subsequent observations will be read and displayed on the screen. One of the variables in our observation (INDXOPER) contains a unique index used to identify observations throughout all of the datasets in the database. The user does not need to know the value of INDXOPER, but we need to include INDXOPER in the extended table to determine which observation(s) the user selects. For this reason, we made INDXOPER invisible to the user by setting the
To select countdown(s): position cursor on a data value and press ENTER
(Use PF7 and PF8 to page countdown selections up or down)

When all countdowns to be processed have been selected position the
cursor on CONTINUE and press ENTER

---

To select countdown(s): position cursor on a data value and press ENTER
(Use PF7 and PF8 to page countdown selections up or down)

When all countdowns to be processed have been selected position the
cursor on CONTINUE and press ENTER

---

**CONCLUSION**

Analysts have enjoyed using the PERFDB. They quickly learn how to use the database application because their input is minimal, selection lists are provided, and input is automatically validated. Those users who are familiar with older, less friendly applications are especially happy when they're able to make selections from lists provided to them instead of having to look up the correct input data and type it in. Moreover, by displaying selected data from each dataset observation, the extended table panels give the user confidence that the desired data is in the database. Using SAS software products, we were able to provide an effective user-interface to our database.

**REFERENCES**


**INFORMATION**

For further information, please contact:

The Johns Hopkins University Applied Physics Laboratory
Johns Hopkins Road
Laurel, MD 20723-6099
Attn: Elizabeth Ogden, Bldg 8-132

**TRADEMARKS**

SAS and SAS/AF are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.