Subsetting a dataset on simple criteria is an easy coding task in base SAS using the WHERE statement or WHERE data step option. For instance, the code to conditionally select observation by a numeric value could be:

```
where code = 23;
```
or for a character value:

```
where name = 'Jones';
```

The subsetting of datasets in Screen Control Language (SCL) becomes much more difficult. The following examples will use a dataset 'PERSONEL' defined as:

```
LASTNAME $ 15
PAYGRADE N 3
HIREDATE N 8 MMDDYY8.
```

The data file might contain such values:

<table>
<thead>
<tr>
<th>Name</th>
<th>Pay Grade</th>
<th>Hire Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jones</td>
<td>120</td>
<td>03/03/85</td>
</tr>
<tr>
<td>Smith</td>
<td>126</td>
<td>09/12/92</td>
</tr>
<tr>
<td>Meyers</td>
<td>119</td>
<td>12/15/89</td>
</tr>
<tr>
<td>Ball</td>
<td>122</td>
<td>11/19/91</td>
</tr>
<tr>
<td>Greene</td>
<td>119</td>
<td>04/16/87</td>
</tr>
<tr>
<td>Lewis</td>
<td>126</td>
<td>09/30/88</td>
</tr>
</tbody>
</table>

To set up a simple search menu for the user (where all screen control variables are named the same as the dataset variables), the display screen would appear as:

1. Last Name: &lastnam
2. Pay Grade: &pay
3. Hire Date &hiredat

There are certain attributes for each variable which would need to be changed:

- LASTNAME field alias changed to LASTNAME from LASTNAM and CAPS turned off
- PAYGRADE field alias to PAYGRADE from PA and type to NUM (numeric) from default of CHAR
- HIREDATE field alias changed to HIREDATE from HIREDATE, the type to NUM and both format and informat set to MMYYDD8.

The source code for the init section of the program as well as the last name search portion would be:

```
Init:
dsidi = OPEN('ssd.personel','i');
call set(dsidi);
Return;

Main:
*reset WHERE clause;
rc = where(dsidi);

*Search Last Name Field;
If lastname ^= _blank_ then do;
rc = where(dsidi,'lastname = "' || lastname || '"');
anyobs = attrn(dsidi,'ANY');
if anyobs = 0 then do;
_msg_ = 'No match found';
alarm;
lastname = _blank_;
cursor lastname;
return;
else do;
rc = locatec(dsidi, varnum (dsidi,'lastname'), lastname);
end;

Return;
```

When the user presses the ENTER key, the Main section is executed. If a match is found, the data from the first observation meeting the criteria will be displayed on the screen. If no match is found, a message will be sent to the user.

To more specifically understand the code used:

```
rc = where(dsidi,('lastname = "' || lastname || '"');
```

<table>
<thead>
<tr>
<th>Dataset ID</th>
<th>Dataset variable</th>
<th>SCL Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SCL variables cannot be placed in quotes but SCL requires the WHERE clause to be in quotes, thus the unusual configuration of double quotes, single quotes, and concatenation operators surrounding the SCL variable. The LOCATE function is needed to move the data from the selected observation in the dataset to the dataset data vector (DDV). The CALL SET statement found in the init section will automatically cause the data stored in the DDV to move to the screen control data vector (SDV) and vice versa. The ANYOBS function is the only one-line method of testing to see if any observations exist in a subsetted dataset. Many of the functions which would be available in base SAS do not work in SCL to solve this problem.

The code to perform the search dependent on paygrade is as follows:

```
If paygrade ^= _blank_ then do;
   rc = where(dsid,'paygrade ='
            ||paygrade);
   anyobs = attm(dsid,'ANY');
   if anyobs = 0 then do;
      *same error response as above;
   else rc = locaten(dsid,varnum
                    (dsid,'paygrade'),paygrade);
   end;
```

Again, an unusual configuration of quotes and concatenation operators are used, this time to concatenate the SCL numeric variable to the WHERE clause. When this code is compiled, a warning message is displayed stating:

```
WARNING: [Line 12] Converting numeric variable PAYGRADE to character
```

In Observations, Vol.1 No. 2, First Quarter 1992, pgs 57-58, the solution to this warning is discussed, involving the use of put statements and formats. The code in this paper was executed on versions 6.07 and 6.08 on IBM/CMS, UNIX (AIX) and PC-Windows SAS. In testing this 'problem' across these platforms, although the warning is stated, the subset was correctly performed and the appropriate observation(s) was/were selected. Therefore, simply ignoring the warning is this case worked!

To perform the search on the formatted variable, HIREDATE, the code is similar to that for PAYGRADE:

```
If hiredate ^= _blank_ then do;
   rc = where(dsid,'hiredate
            ='||hiredate);
   anyobs = attm(dsid,'ANY');
   if anyobs = 0 then do;
      *same error response as above;
   else rc = locaten(dsid,varnum
                    (dsid,'hiredate'),hiredate);
   end;
```

The same warning message appears during compilation but the appropriate subset occurs.

Multiple selection criteria can also be designed in SCL. There are a variety of methods which can be used. One such method which allows for easy documentation and modification is:

```
clause1 = 'lastname = ""||lastname||""';
clause2 = 'AND hiredate ='||hiredate;
rc = where(dsid,clause1,clause2);
```

Using WHERE clauses in SCL do take a bit of getting used to and there are always the syntactical problems when combining so many symbols, but the subsets do work.