Conditionally Executing DO Loops

The iterative DO statement specifies a fixed number of iterations for the DO loop. However, there are times when you want to execute a DO loop until a condition is reached or while a condition exists, but you don't know how many iterations are needed.

Suppose you want to calculate the number of years required for an investment to reach $50,000. In the DATA step below, using an iterative DO statement is inappropriate because you are trying to determine the number of iterations required for Capital to reach $50,000.

```
data work.invest;
   do year=1 to ?;
      Capital+2000;
      capital+capital*.10;
    end;
run;
```

The DO WHILE and DO UNTIL statements enable you to execute DO loops based on whether a condition is true or false.

Using the DO UNTIL Statement

The DO UNTIL statement executes a DO loop until the expression becomes true.

General form, DO UNTIL statement:

```
DO UNTIL(expression);
        SAS statements
END;
```

where expression is a valid SAS expression enclosed in parentheses.

The expression is not evaluated until the bottom of the loop, so a DO UNTIL loop always executes at least once. When the expression is evaluated as true, the DO loop stops.

Using the DO WHILE Statement

Like the DO UNTIL statement, the DO WHILE statement executes DO loops conditionally. You can use the DO WHILE statement to execute a DO loop while the expression is true.

General form, DO WHILE statement:

```
DO WHILE(expression);
        SAS statements
END;
```

where expression is a valid SAS expression enclosed in parentheses.

An important difference between the DO UNTIL and DO WHILE statements is that the DO WHILE expression is evaluated at the top of the DO loop. If the expression is false the first time it is evaluated, the DO loop never executes.