SAS Data Manipulation

ARRAY Statement and DO loops

WHY DO WE NEED ARRAYS?

There are sometimes when you want to do the same thing to many variables, For example:

1) you want to convert the numeric variables every 99 to a missing value or

2) you want to take every numeric variables to square root transformation….

You could use a serious of IF statement or other assignment statement to do it. But using ARRAY and a Do Loop will simplify and shorten your program.

INDEXED ARRAY SYNTAX

There are two types of arrays that can be specified in SAS. The first is what I call an indexed array and the second is a non-indexed array. All arrays are set up and accessed only within a DATA step. The syntax for an indexed array is as follows:

INDEXED ARRAY SYNTAX

ARRAY arrayname {n} [$] [length] list_of_array_elements;

- *ARRAY* is a SAS keyword that specifies that an array is being defined
- *arrayname* a valid SAS name that is not a variable name in the data set.
- *{n}* the index used to give the number of elements in the array, optional
- *[$]* used to specify if the elements in the array are character variables, the default type is numeric
- *[length]* used to define the length of new variables being created in the array, optional
- *list_of_array_elements* a list of variables of the same type (all numeric or all character) to be included in the array

- All ARRAY will followed by a DO loop

INDEXED ARRAY SYNTAX

- The array itself is not stored with the data set. It is temporary defined the DATA step. To reference an array variable, give the array name and the subscript for that variable. The first variable in the variable list has subscript 1, the second has subscrip2, and so forth.

- ARRAY color (5) red yellow green blue pink;

- Color (1) is the variable red, color (2) is the variable yellow….

NON-INDEXED ARRAY SYNTAX

- In addition to the indexed array, SAS also provides the option of using a non-indexed array. Here you don’t specify the number of elements in the array, {n}. Rather, during the creation of the array, SAS determines the number of elements of the array based on the set of variables listed. The syntax for a non-indexed array is as follows:

NON-INDEXED ARRAY SYNTAX

ARRAY arrayname [$] [length] list_of_array_elements;

- *ARRAY* is a SAS keyword that specifies that an array is being defined;
- *arrayname* a valid SAS name that is not a variable name in the data set;
- *[$]* used to specify if the elements in the array are character variables, the default type is numeric;
- *[length]* used to define the length of new variables being created in the array, optional
- *list_of_array_elements* a list of variables of the same type (all numeric or all character) to be included in the array.

EXAMPLE OF A NON-INDEXED ARRAY

Using the CES-D item reversal example, the SAS code that would be to define a non-indexed array containing the 4 CES-D items (0,1,2,3 become 3,2,1,0) that need to be reversed is

```
data cesd;
set in.cesd1;
array areverse cesd4 cesd8 cesd12 cesd18;
```

SAS DO LOOPS

So we have now defined our arrays, but now we have to use them to manipulate the data. We use a DO loop to perform the data manipulations on the arrays.

SAS DO LOOPS

There are four different types of DO loops available in SAS.

1. DO index=, an iterative, or indexed, DO loop used to perform the operations in the DO (Rename)loop at a specified start and ending index value for an array

2. 2. DO OVER loop used to perform the operations in the DO loop over ALL elements in the array

3. 3. DO UNTIL (logical condition) loop used to perform the operations in the DO loop until the logical condition is satisfied

4. 4. DO WHILE (logical condition) loop used to perform the operations in the DO loop while the logical condition is satisfied

DO LOOP WITH INDEXED ARRAY

```
/**CESD: self-report scale to measure depressive symptom**/
/** Reverse CESD4-16 score 0→3, 1 → 2, 2 → 1, 3 → 0**/
DATA cesd;
SET in.cesd1;
ARRAY areverse {4} cesd4 cesd8 cesd12 cesd16;
DO i=1 to 4;
   areverse[i]=3-areverse[i];
END;
```

DO OVER LOOP WITH A NON-INDEXED ARRAY
(Do not rename)

```
/** Compare with the previous one, What is the
   differences?**/
DATA cesd;
SET in.cesd1;
ARRAY areverse cesd4 cesd8 cesd12 cesd16;
DO over areverse;
    areverse=3-areverse;
END;
```

DO OVER LOOP WITH A NON-INDEXED ARRAY
(Rename)

```
DATA cesd;
SET in.cesd1;
ARRAY aold cesd4 cesd8 cesd12 cesd16;
ARRAY arev rcesd4 rcesd8 rcesd12 rcesd16;
DO OVER aold;
  arev=3-aold;
END;
```

DO OVER LOOP WITH A NON-INDEXED ARRAY
(Do not rename)

```
DATA list;
SET in.list1;
ARRAY aquest q1-q20;
        DO OVER aquest;
        if aquest=. then aquest=0;
        END;
RUN;
```

DO OVER LOOP WITH A NON-INDEXED ARRAY (Rename)

```
DATA list;
SET in.list1;
   ARRAY aquest q1-q20;
   ARRAY anquest nq1-nq20;
      DO OVER aquest;
   anquest=aquest;
   if anquest=. Then anquest=0;
      END;
```