**Chapter XV - Processing Variables with Arrays**

A SAS array is a temporary grouping of SAS variables under a single name. An array exists only for the duration of the DATA step. One reason for using an array is to reduce the number of statements that are required for processing variables.

Eg:

**DATA** Work.Report(**DROP**=day);

**SET** Master.Temps;

**ARRAY** wkday{7} mon tue wed thr fri sat sun;

**DO** day=1 **TO** 7;

wkday{day}=5\*(wkday{day}-32)/9;

**END**;

**RUN**;

1. Creating One-Dimensional Arrays

* Specifying the Array Name (give the array a name)
* Specifying the Dimension

Must specify the dimension of the array after the array name which describe the number and arrangement of element.

* Specify the number of array elements

Eg:

**ARRAY** sales{4} qtr1 qtr2 qtr3 qtr4;

* Specify a range of values

Eg:

**ARRAY** sales{96:99} totals96 totals97 totals98 totals99;

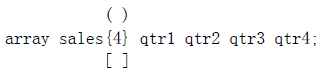
* Using an asterisk (\*)

Eg:

**ARRAY** sales{\*} qtr1 qtr2 qtr3 qtr4;

* Enclose the dimension in either parentheses, braces, or brackets.

Eg:



* Specifying Array Elements
* list each variable name, separate each element with a space, and ends with a semicolon (;)

**ARRAY** sales{4} qtr1 qtr2 qtr3 qtr4;

* Specify array elements as a variable list

|  |  |
| --- | --- |
| **Variables** | **Form** |
| a numbered range of variables | **Var1-Varn** |
| all numeric variables | **\_NUMERIC\_** |
| all character variables | **\_CHARACTER\_** |
| all character or all numeric variables | **\_ALL\_** |

* A numbered range of variables

在这种name方式中，每个variable必须拥有一样的name；最后一位必须是数字；must be numbered consecutively

* Simple example：

**ARRAY** sales{4} qtr1-qtr4;

* Also specify a range of values for the index when you define the array

**ARRAY** sales{96:99} totals96-totals99;

* The ability to reference the elements of an array by an index value is what gives arrays their power. Typically, arrays are used with DO loops to process multiple variables and to perform repetitive calculations. (在这种方式中，subscript values are assigned in the order of the array elements)

Eg:

array quarter{4} jan apr jul oct;

do qtr=1 to 4;

YearGoal=quarter{qtr}\*1.2;

end;

* All numeric variables

**\_NUMERIC\_** specifies all numeric variables that have already been defined in the current DATA step.

Eg:

**ARRAY** sales{\*} **\_NUMERIC\_**;

* All character variables

**\_CHARACTER\_** specifies all character variables that have already been defined in the current DATA step.

Eg:

**ARRAY** sales{\*} **\_CHARACTER\_**;

* All character or all numeric variables

**\_ALL\_** specifies all variables that have already been defined in the current DATA step. The variables must all be of the same type: all character or all numeric.

Eg:

**ARRAY** sales{\*} **\_ALL\_**;

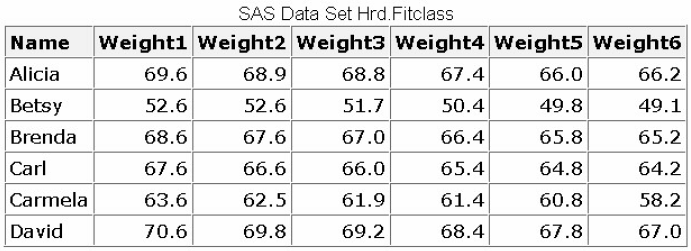
* Basic code:

**ARRAY** *array-name***{***dimension***}** <*elements*>**;**

* *array-name* specifies the name of the array.
* *dimension* describes the number and arrangement of array elements.
* *elements* lists the variables to include in the array. Array elements must be either all numeric or all character. If no elements are listed, new variables will be created with default names.
* 注意：不要将array的名称设置成和DATA step中的variable一样。同时避免使用和SAS function一样的名称，不然将无法在相同的DATA step中使用该function，并且SAS会在log中输出一条warning massage
* 注意：You cannot use array names in LABEL, FORMAT, DROP, KEEP, or LENGTH statements. Arrays exist only for the duration of the DATA step.
* Compilation and Execution

Eg:

The Health Center of a company conducts a fitness class for its employees. Each week, participants are weighed so that they can monitor their progress. The weight data, currently stored in kilograms, needs to be converted to pounds

**DATA** Hrd.Convert(**DROP**=i);

**SET** Hrd.Fitclass;

**ARRAY** wt{6} weight1-weight6;

**DO** i=1 **TO** 6;

wt{i}=wt{i}\*2.2046;

**END**;

**RUN**;

* Compilation

The index values of the array elements are assigned. Note that the array name is not included in the program data vector. The array exists only for the duration of the DATA step.



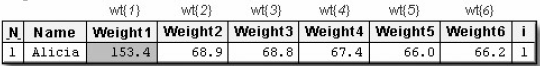
* Executing

在这个步骤中，SAS开始向PDV中写入从data set中读取的data

**ARRAY** wt{6} weight1-weight6;



wt{i}=wt{i}\*2.2046;



* Using the **DIM** Function in an Iterative **DO** Statement

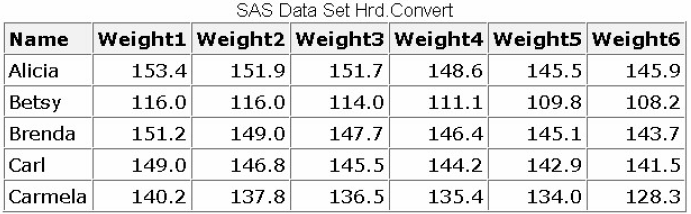
可以用DIM function来获取array的长度。

* Basic code：

**DIM**(*array-name*)

*array-name* specifies the array.

* Eg:

**DATA** Hrd.Convert;

**SET** Hrd.Fitclass;

**ARRAY** wt{\*} weight1-weight6;

**DO** i=1 **TO** **DIM**(wt);

wt{i}=wt{i}\*2.2046;

**END**;

**RUN**;

这个例子中**DIM**(wt) returns a value of 6.

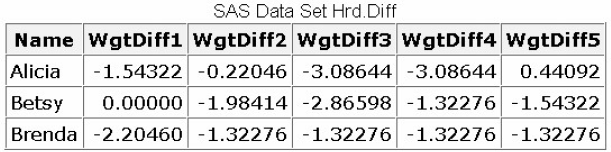
1. Expanding Your Use of Arrays

如果在建立array的时候没有specify variable的话，系统自动会create新的variable。这个自动建立的variable followed by consecutive numbers 1 to the dimension of the array

Eg:

继example in Compilation and Execution

Suppose you need to calculate the weight gain or loss from week to week for each member of a fitness class, shown below. 上一步中，建立了一个SAS表格，读入了数据并且做了必要的计算。这个步骤中，我们需要建立一个新的array来实现建立weight gain or loss from的目标。

 **DATA** Hrd.Diff;

**SET** Hrd.Convert;

**ARRAY** wt{6} weight1-weight6;

**ARRAY** WgtDiff{5};

**DO** i=1 **TO** 5;

wgtdiff{i}=wt{i+1}-wt{i};

**END**;

**RUN**;

注意：To create an array of character variables, add a dollar sign ($) after the array dimension.

系统默认所有character variables的长度为8。如果需要，系统允许用户specify variable的length

* Assigning Initial Values to Arrays
* Basic code:

**ARRAY** goal{4} g1 g2 g3 g4 (initial values);

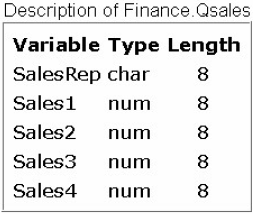
在给array variable assign初始值的时候，在variable list之后的括号中输入与各个variable相对应的初始值，并用空格或逗号将它们分隔开，如果初始值为character，必须将其放置于quotation marks中。

Eg:

**ARRAY** goal{4} g1 g2 g3 g4 (9000 9300 9600 9900);

**ARRAY** col{3} $ color1-color3 ('red','green','blue');

* Eg:

**DATA** Finance.Report(**DROP**=i);

**SET** Finance.Qsales;

**ARRAY** sale{4} sales1-sales4;

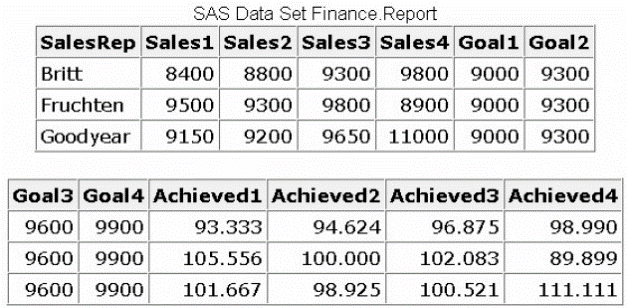
**ARRAY** Goal{4} (9000 9300 9600 9900);

**ARRAY** Achieved{4};

**DO** i=1 **TO** 4;

achieved{i}=100\*sale{i}/goal{i};

**END**;

**RUN**;

* Creating Temporary Array Elements

To create temporary array elements for DATA step processing without creating new variables, specify **\_TEMPORARY\_** after the array name and dimension

**DATA** Finance.Report(**DROP**=i);

**SET** Finance.Qsales;

**ARRAY** sale{4} sales1-sales4;

**ARRAY** Goal{4}**\_TEMPORARY\_** (9000 9300 9600 9900);

**ARRAY** Achieved{4};

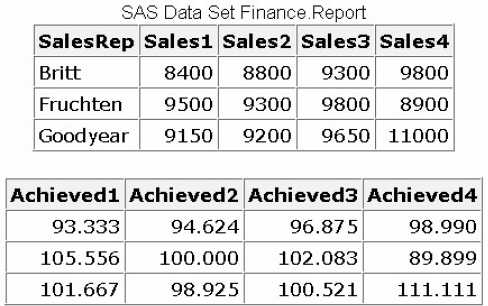
**DO** i=1 **TO** 4;

achieved{i}=100\*sale{i}/goal{i};

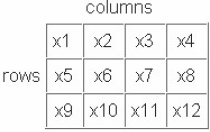
**END**;

**RUN**;

注意：Temporary array elements do not appear in the resulting data set.



1. Multidimensional Arrays

* Defining a Multidimensional Array

Specify the number of elements in each dimension, separated by a comma.

* Basic code:

**ARRAY** new{r,c}

The first dimension in the **ARRAY** statement specifies the number of rows, and the second dimension specifies the number of columns.

* You can reference any element of the array by specifying the two dimensions.

Eg:

You can perform an action on the variable x7 by specifying the array reference new(2,3). You can easily locate the array element in the table by finding the row (2), then the column (3).

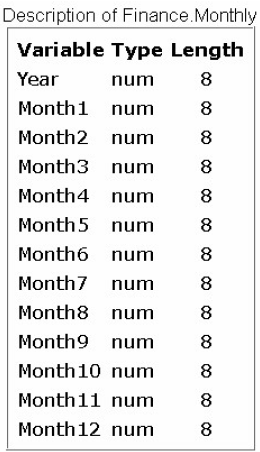
**ARRAY** new{3,4} x1-x12;

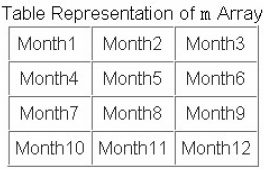
new(2,3)=0;

* Referencing Elements of a Two-Dimensional Array

Multidimensional arrays are typically used with nested DO loops.

Eg:

 Your company's sales figures are stored by month in the SAS data set Finance.Monthly. Your task is to generate a new data set of quarterly sales rather than monthly sales.

 **DATA** finance.quarters(**DROP**=i j);

**SET** finance.monthly;

**ARRAY** m{4,3} month1-month12;

**ARRAY** Qtr{4};

**DO** i=1 **TO** 4;

qtr{i}=0;

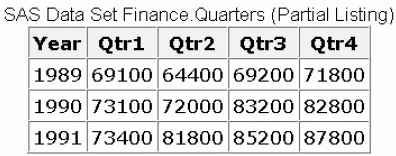
**DO** j=1 **TO** 3;

qtr{i}+m{i,j};

**END**;

**END**;

**RUN**;



You can use arrays for rotating (transposing) a SAS data set. When you rotate a SAS data set, you change variables to observations or observations to variables.

Eg:

Suppose you want to rotate the Finance.Funddrive data set to create four output observations from each input observation

**DATA** Work.Rotate(**DROP**=qtr1-qtr4);

**SET** Finance.Funddrive;

**ARRAY** contrib{4} qtr1-qtr4;

**DO** Qtr=1 **TO** 4;

Amount=contrib{qtr};

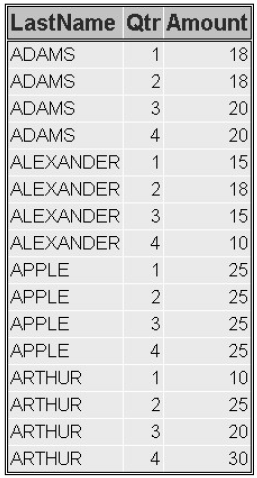
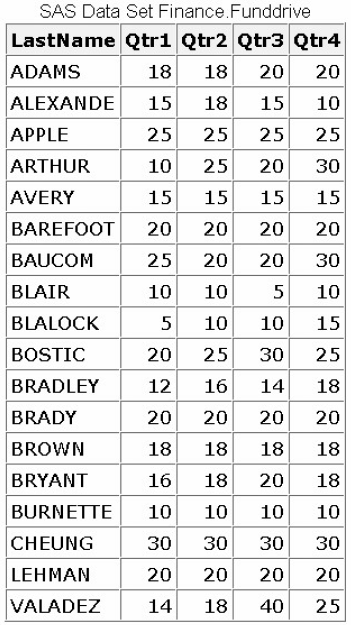
**OUTPUT**;

**END**;

**RUN**;

**PROC** **PRINT** data=**ROTATE**(obs=16) **NOOBS**;

**RUN**;



练习

1. For the program below, select an iterative DO statement to process all elements in the

contrib array.

**DATA** work.contrib;

**ARRAY** contrib{4} qtr1-qtr4;

**...**

contrib{i}=contrib{i}\*1.25;

**END**;

**RUN**;

1. do i=4;
2. do i=1 to 4;
3. do until i=4;
4. do while i le 4;