Chapter 15

Processing Variables with SAS Arrays

An Example: Temperature conversion

Let’s consider a data set (master.temps.sasv7dat) that contains 365 variables **day1-day365** recording daily temperature in °F. We would like to convert the temperatures to °C.

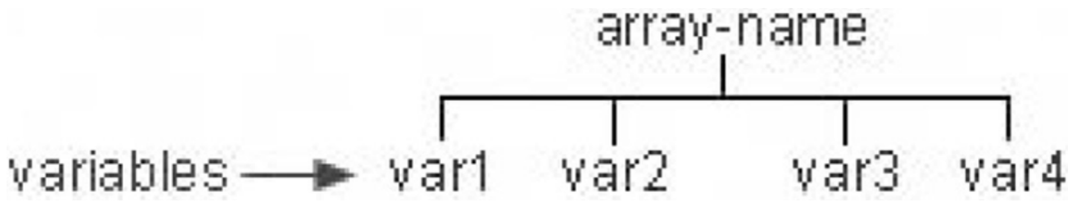
A Possible Solution

Write 365 separate statements to accomplish the task.

```
data work.report;
set master.temps;
      day1celsius=5*(day1-32)/9;
      day2celsius=5*(day2-32)/9;
      day3celsius=5*(day3-32)/9;
      day4celsius=5*(day4-32)/9;
      …
      day365celsius=5*(day365-32)/9;
run;
```

An Alternative Solution: Use Arrays

- Group variables under one name by creating a new variable with an ARRAY statement.



What is an ARRAY?

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.

Why Are ARRAYs Used?

- Perform repetitive calculations.
- Create many variables with same attributes.
- Manipulate variables as a group.
- Compare variables.
- Read data.
- Write shorter code.
- Restructure SAS data sets (wide ←→ long).

Example: Alternative Solution

The previous dataset contains 365 variables **day1-day365** recording daily temperature in °F. We can use a SAS array to convert the temperatures to °C.

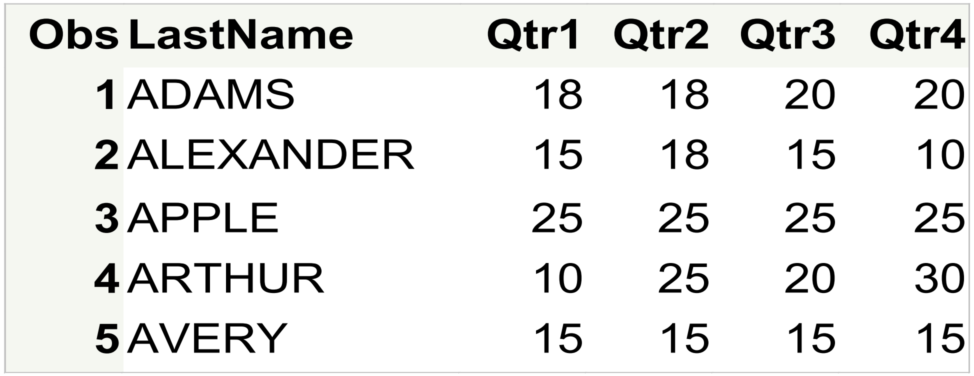
```
data work.report(drop=i);
  set master.temps;
  array daytemp{365} day1-day365;
  do i=1 to 365;
     daytemp{i}=5*(daytemp{i}-32)/9;
  end;
run;
```

General Form of an ARRAY Statement

(One-dimensional array)

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

Example Sales Data:



Obs	LastName	Qtr1	Qtr2	Qtr3	Qtr4
1	ADAMS	18	18	20	20
2	ALEXANDER	15	18	15	10
3	APPLE	25	25	25	25
4	ARTHUR	10	25	20	30
5	AVERY	15	15	15	15

We would like to perform the same operation (say, multiplication) on each Qtr variable.

Use of Arrays

Goal: to convert the sales in dollars to sales in Euros.

```
array quarter{4} qtr1 qtr2 qtr3 qtr4;
    do i=1 to 4;
          quarter{i} = quarter{i}*0.88;
    end;
run;
```

More Ways to Specify Arrays

Specifying array elements:

```
Array sales{4} qtr1 qtr2 qtr3 qtr4;
Array sales{4} qtr1-qtr4;      /*a numbered range of variables*/
```

Specifying array dimension:

```
Array sales(4) qtr1 qtr2 qtr3 qtr4;
Array sales{4} qtr1 qtr2 qtr3 qtr4;
Array sales[4] qtr1 qtr2 qtr3 qtr4;
Array sales{1:4} qtr1 qtr2 qtr3 qtr4;   /*a range of values*/
Array sales{*} qtr1 qtr2 qtr3 qtr4;      /* Let SAS determine the value*/
```

Variable Lists as Array Elements

You can specify all numeric, all character or all variables that have already been defined in the data step:

```
Array sales{*} _NUMERIC_;
Array sales{*} _CHARACTER_;
Array sales{*} _ALL_;
```

Some Restrictions

Specifying a numbered range of variables:

```
Array sales{4} qtr1-qtr4;
```

- The variables must have the same name except for the last character or characters.
- The last character(s) of each variable must be digit(s).
- The variables must be numbered consecutively.

Specifying all variables:

```
Array sales{*} _ALL_;
```

All variable must be the same type (all numeric or all character)

The DIM Function

The DIM function is useful when array length is not user-specified (more flexible):

```
Array quarter{*} qtr1-qtr4;
      do i=1 to dim(quarter);
              quarter{i} = quarter{i}*0.88;
      end;
run;
```

Creating New Variables With Arrays

You can create variables in an ARRAY by omitting ARRAY elements.

```
array score{3};      /*creating numeric variables */
```

It creates 3 variables: score1, score2 and score3.

Creating character variables with arrays:

```
array name{3} $;      /*default length = 8*/
array name{3} $ 20;  /* specify length of 20*/
```

Creating New Variables with Arrays

```
data score1;
   array score{3};
   input id score{*};
datalines;
1234  99 60 82
5678  80 85 75
;
```

score1	score2	score3	id
99	60	82	1234
80	85	75	5678

Creating New Variables with Arrays

```
data score1;
   array score{3};
   input id score{*};
   do i=1 to 3;
      if score{i}>=75 then
         do;
            NewScore=score{i};
            output;
         end;
   end;
datalines;
1234  99 60 82
5678  80 85 75
;
```

score1	score2	score3	id	i	NewScore
99	60	82	1234	1	99
99	60	82	1234	3	82
80	85	75	5678	1	80
80	85	75	5678	2	85
80	85	75	5678	3	75

Assigning Initial Values to the Elements of an Array

```
array test{3} t1-t3 (90 80 70);
```

Enclose each initial character value in quotation marks

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

- place values after array elements
- specify an initial value for each array element
- separate each value with **a comma or blank**
- enclose initial values in parentheses.

Assigning Initial Values to the Elements of an Array

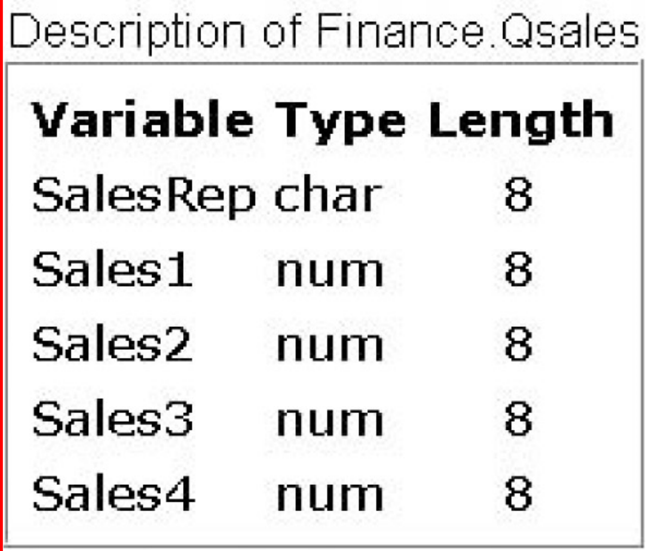
```
 data score1;
   array test{3} t1-t3 (90 80 70);
   array score{3} s1-s3;
   input id score{*};
   do i=1 to 3;
      if score{i}>=test{i} then do;
           NewScore=score{i};
           output;
     end;
   end;
datalines;
1234  99 60 82
5678  80 85 75
;
```

t1	t2	t3	s1	s2	s3	id	i	NewScore
90	80	70	99	60	82	1234	1	99
90	80	70	99	60	82	1234	3	82
90	80	70	80	85	75	5678	2	85
90	80	70	80	85	75	5678	3	75

Assigning Initial Values to the Elements of an Array

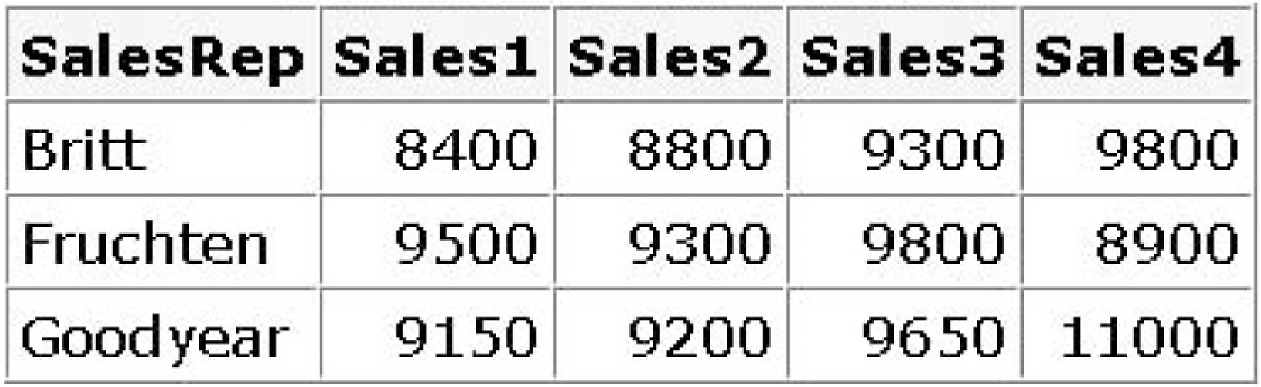
Example:

Compare actual sales figures to sales goals in the Finance.Qsales dataset



Description of Finance.Qsales

Variable	Type	Length
SalesRep	char	8
Sales1	num	8
Sales2	num	8
Sales3	num	8
Sales4	num	8

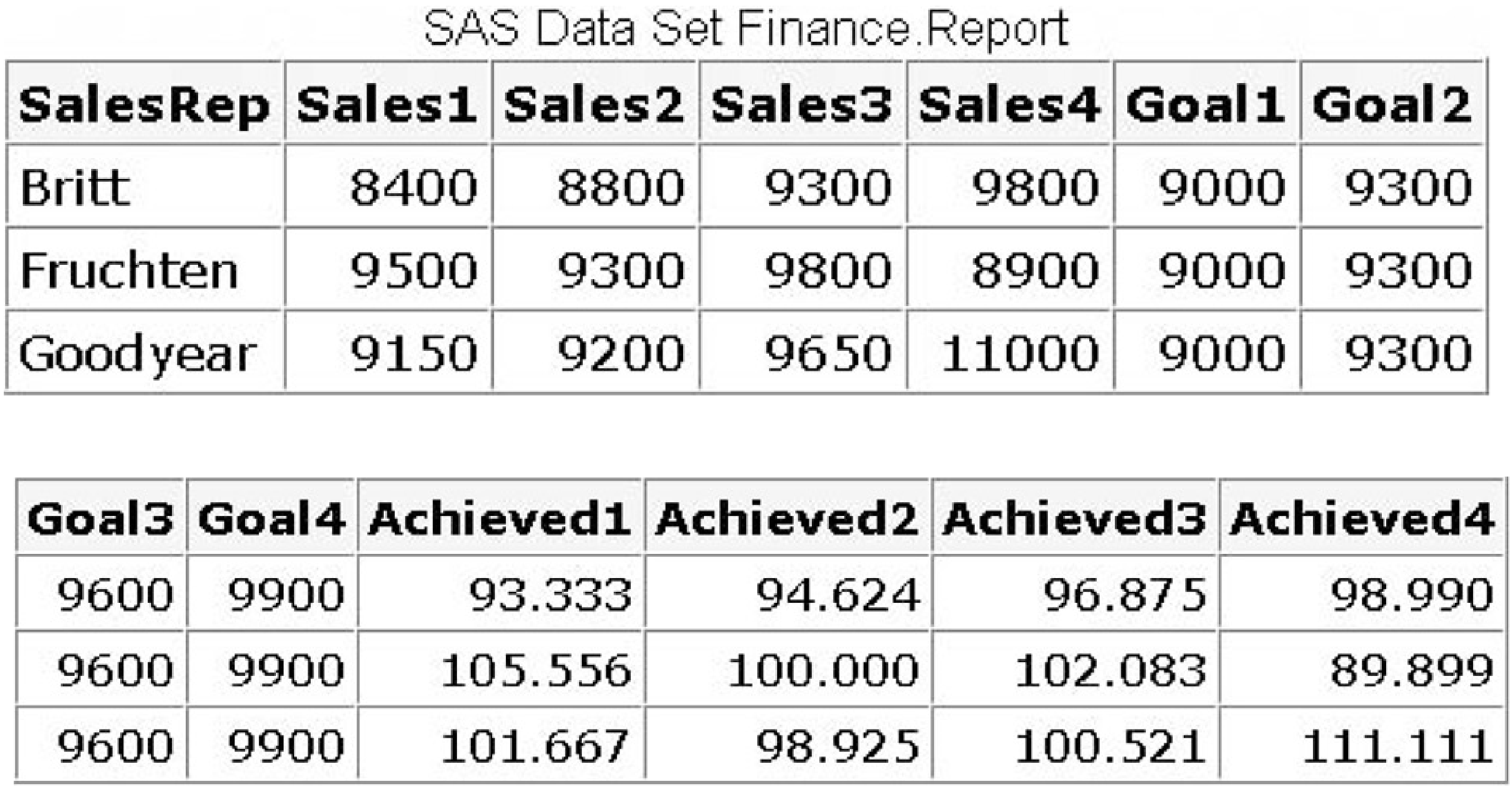


SalesRep	Sales1	Sales2	Sales3	Sales4
Britt	8400	8800	9300	9800
Fruchten	9500	9300	9800	8900
Goodyear	9150	9200	9650	11000

Assigning Initial Values to the Elements of an Array

```
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;
```

Output



SAS Data Set Finance.Report

SalesRep	Sales1	Sales2	Sales3	Sales4	Goal1	Goal2	Goal3	Goal4	Achieved1	Achieved2	Achieved3	Achieved4
Britt	8400	8800	9300	9800	9000	9300	9600	9900	93.333	94.624	96.875	98.990
Fruchten	9500	9300	9800	8900	9000	9300	9600	9900	105.556	100.000	102.083	89.899
Goodyear	9150	9200	9650	11000	9000	9300	9600	9900	101.667	98.925	100.521	111.111

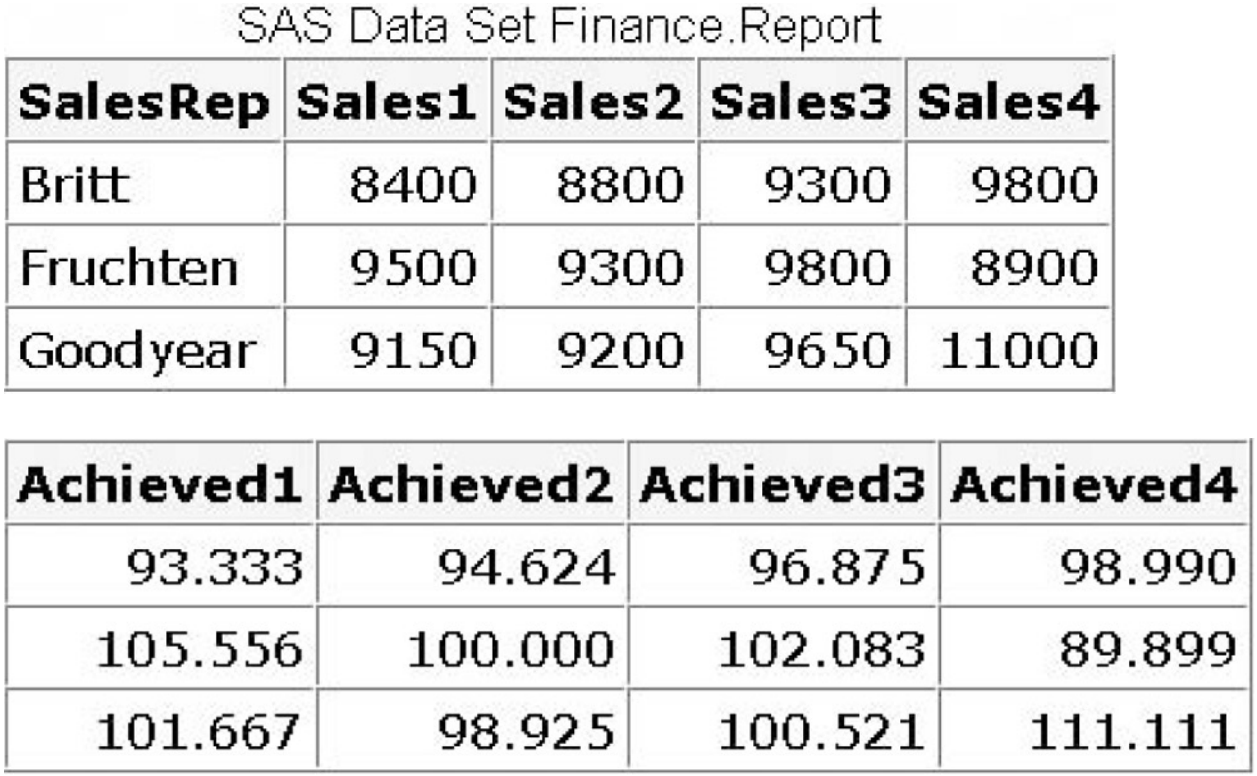
Temporary Arrays

To create temporary arrays without creating new variables, include **_temporary_** after the array 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;
```

Temporary Arrays (cont’d)

New variables are not created for temporary array elements



SAS Data Set Finance.Report

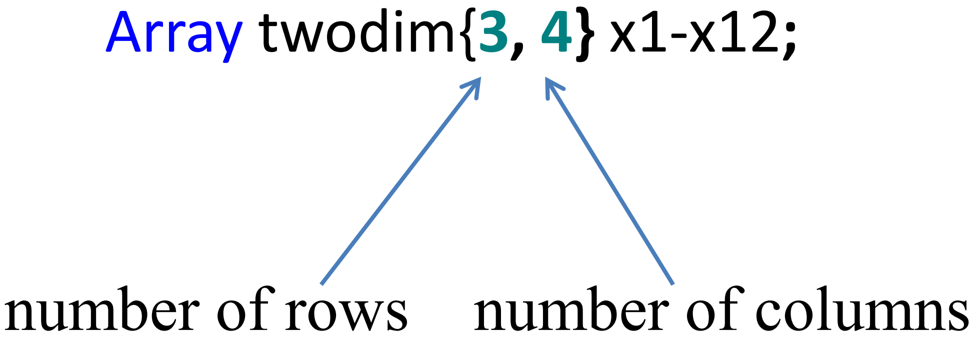
SalesRep	Sales1	Sales2	Sales3	Sales4	Achieved1	Achieved2	Achieved3	Achieved4
Britt	8400	8800	9300	9800	93.333	94.624	96.875	98.990
Fruchten	9500	9300	9800	8900	105.556	100.000	102.083	89.899
Goodyear	9150	9200	9650	11000	101.667	98.925	100.521	111.111

Multidimensional Arrays

Resp1 → Answer1 Answer2 Answer3
Resp2 → Answer4 Answer5 Answer6
Resp3 → Answer7 Answer8 Answer9
Resp4 → Answer10 Answer11 Answer12

Multidimensional Arrays

Two-dimensional arrays – a common type of multidimensional array.



```
Array twodim{3, 4} x1-x12;
```

number of rows

number of columns

X1	X2	X3	X4
X5	X6	X7	X8
X9	X10	X11	X12

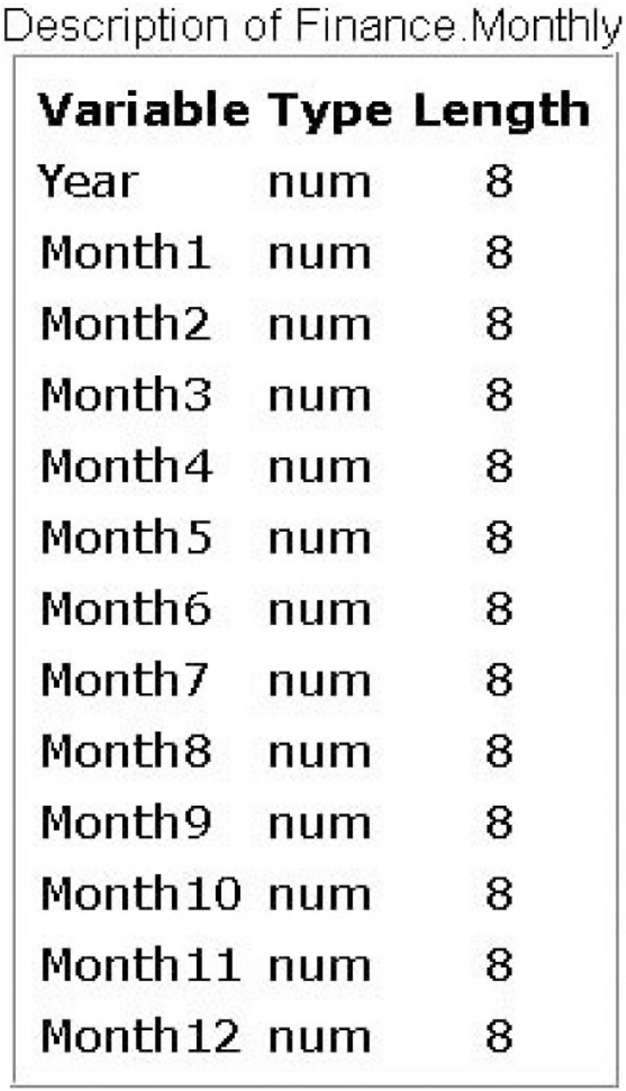
Referencing Elements of a Two-dimensional Array

```
array twodim{3,4} x1-x12;
twodim[2,3] = 5;
```

X1	X2	X3	X4
X5	X6	**5**	X8
X9	X10	X11	X12

Referencing Elements of a Two-dimensional Array

Example: Company sales data stored by month. You need to generate a new data set of quarterly sales (total sales for each quarter)



Description of Finance.Monthly

Variable	Type	Length
Year	num	8
Month1	num	8
Month2	num	8
Month3	num	8
Month4	num	8
Month5	num	8
Month6	num	8
Month7	num	8
Month8	num	8
Month9	num	8
Month10	num	8
Month11	num	8
Month12	num	8

The Finance.Monthly Data Set

Year	Month1	Month2	Month3	Month4	Month5	Month6	Month7	Month8	Month9	Month10	Month11	Month12
1989	23000	21500	24600	23300	20000	21100	23000	21600	24600	24300	22700	24800
1990	23400	24400	25300	25100	22500	24400	28700	27400	27100	28700	26500	27600
1991	23500	24600	25300	28700	25500	27600	28700	28400	28100	29700	28800	29300

Forming a Two-dimensional Array

Define the array m(4,3) to arrange variables month1-month12 into four groups (quarters)

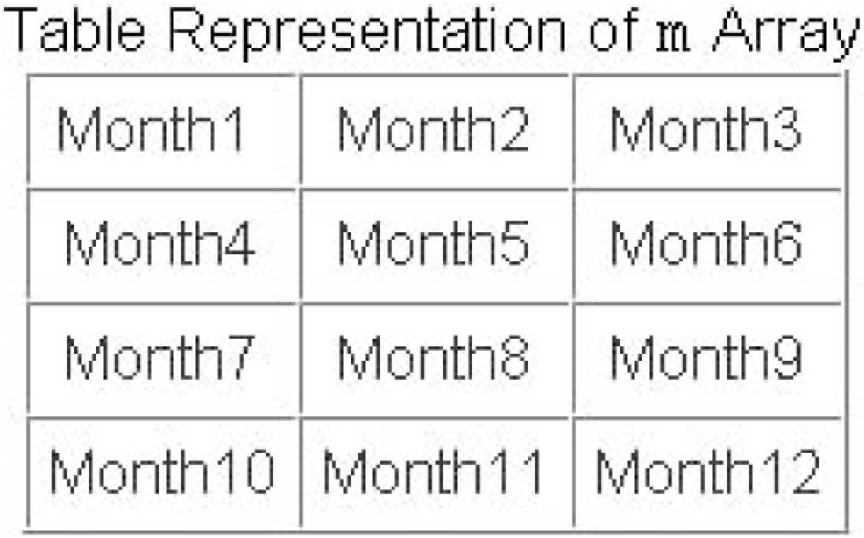


Table Representation of m Array

Month1	Month2	Month3
Month4	Month5	Month6
Month7	Month8	Month9
Month10	Month11	Month12

```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
```

Define an ARRAY of New Variables

Define an array Qtr[4] that creates variables qtr1-qtr4 which store quarterly sales figures

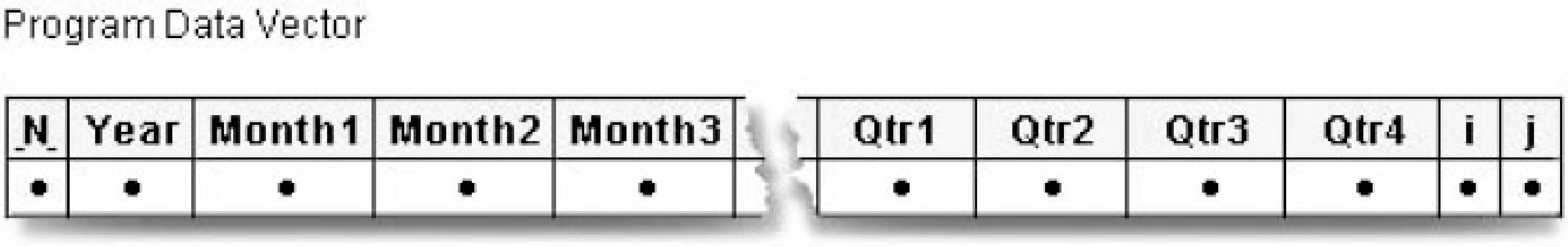
```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
```

Use a Nested DO Loop to sum monthly sales figures for every quarter

```
data finance.quarters;
   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;
```

After the following program is compiled:

```
data finance.quarters;
   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;
```

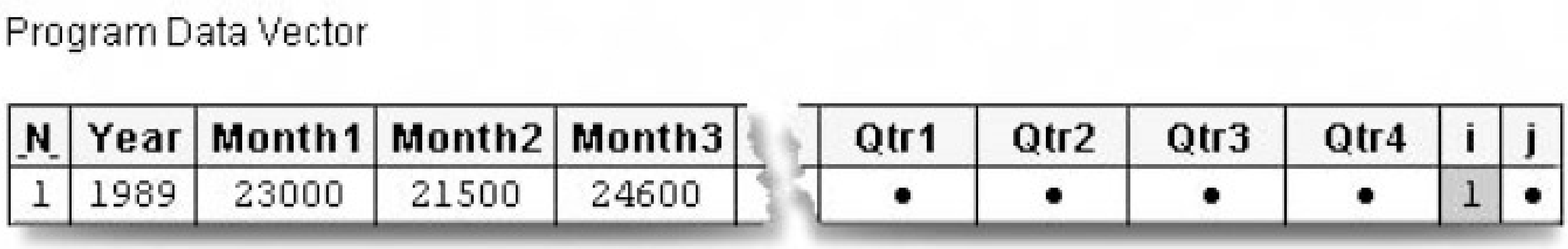


Program Data Vector

_N_	Year	Month1	Month2	Month3	Qtr1	Qtr2	Qtr3	Qtr4	i	j
.	.	.	.	.	.	.	.	.	.	.

The 1st Observation Read in in the 1st Execution of the DATA Step

```
Data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
   do i=1 to 4;                                    /*i=1*/
     qtr{i}=0;
     do j=1 to 3;
        qtr{i}+m{i,j};
     end;
   end;
run;
```

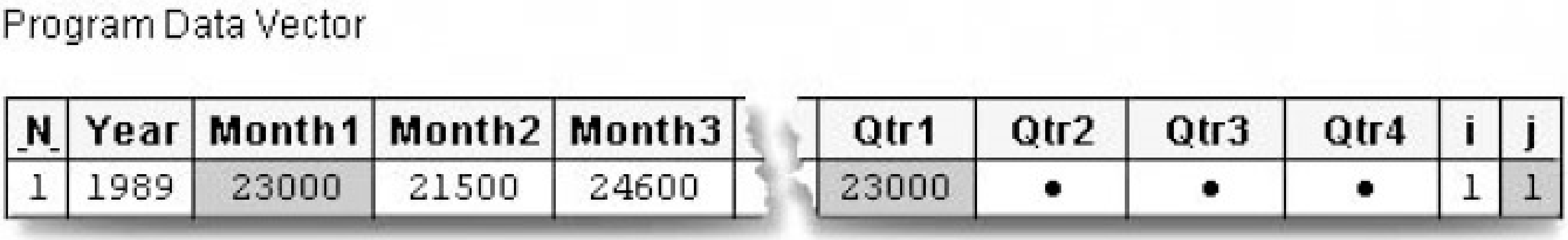


Program Data Vector

_N_	Year	Month1	Month2	Month3	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1	1989	23000	21500	24600	•	•	•	•	1	•

The 1st Addition in the 1st Inner and Outer Do Loops

```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
   do i=1 to 4;                          /*i=1*/
     qtr{1}=0;
     do j=1 to 3;                        /*j=1*/
        qtr{1}+m{1,1};
     end;
   end;
run;
```

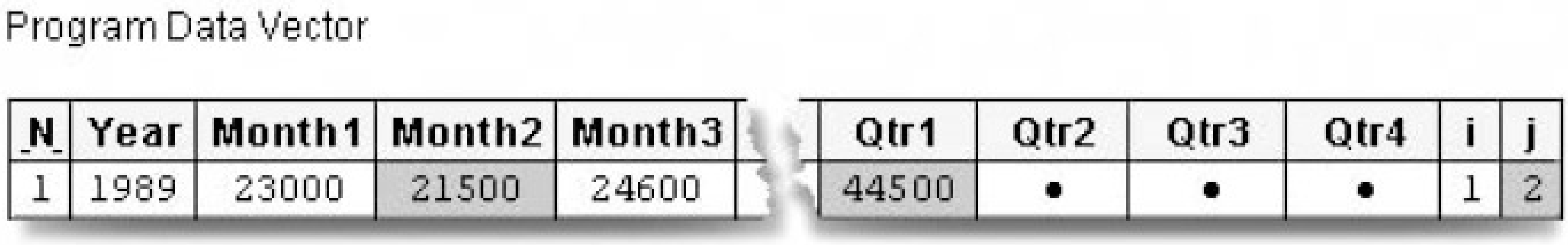


Program Data Vector

_N_	Year	Month1	Month2	Month3	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1	1989	23000	21500	24600	23000	•	•	•	1	1

The 2nd Addition in the 2nd Inner and 1st Outer Do Loops

```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
   do i=1 to 4;                          /*i=1*/
     qtr{1}=0;
     do j=1 to 3;                        /*j=2*/
        qtr{1}+m{1,2};
     end;
   end;
run;
```

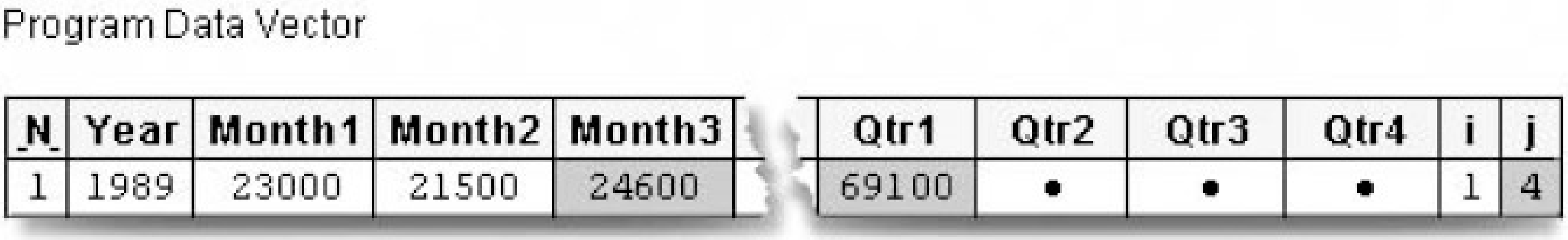


Program Data Vector

_N_	Year	Month1	Month2	Month3	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1	1989	23000	21500	24600	44500	●	●	●	1	2

The 3rd Addition in the 3rd Inner and 1st Outer Do Loops

```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
   do i=1 to 4;                              /*i=1*/
     qtr{1}=0;
     do j=1 to 3;                            /*j=3*/
        qtr{1}+m{1,3};
     end;
   end;
run;
```



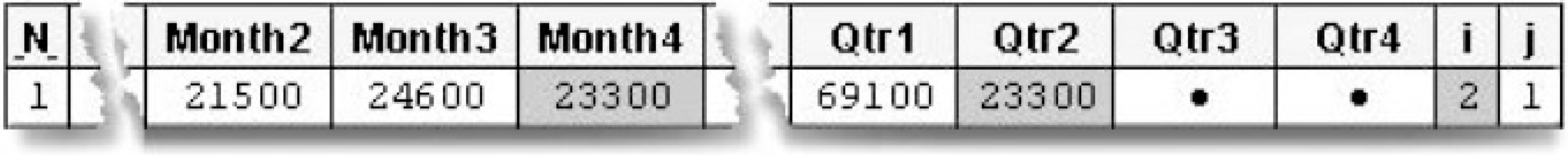
Program Data Vector

_N_	Year	Month1	Month2	Month3	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1	1989	23000	21500	24600	69100	•	•	•	1	4

The 1st Addition in the 1st Inner and 2nd Outer Do Loops

```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
   do i=1 to 4;                        /*i=2*/
     qtr{2}=0;
     do j=1 to 3;                      /*j=1*/
        qtr{2}+m{2,1};
     end;
   end;
run;
```

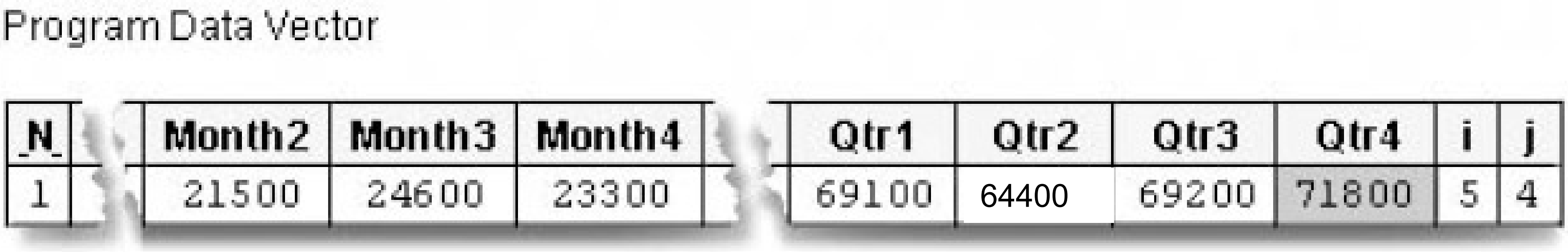
Program Data Vector



_N_	Month2	Month3	Month4	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1	21500	24600	23300	69100	23300	•	•	2	1

The Final Stage: All the Loops End

```
data finance.quarters;
   set finance.monthly;
   array m{4,3} month1-month12;
   array Qtr{4};
   do i=1 to 4;                          /*i=5 -> Loop ends*/
     qtr{i}=0;
     do j=1 to 3;
        qtr{i}+m{i,j};
     end;
   end;
run;
```



Program Data Vector

_N_	Month2	Month3	Month4	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1	21500	24600	23300	69100	64400	69200	71800	5	4

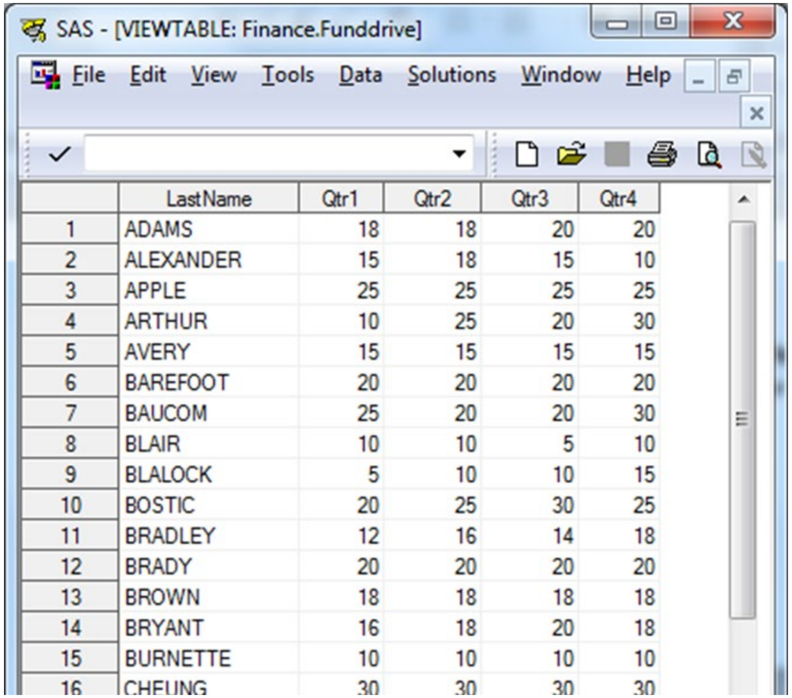
Quarterly Sales Figures Data Set

Year	Month1	Month2	Month3	Month4	Month5	Month6	Month7	Month8	Month9	Month10	Month11	Month12	Qtr1	Qtr2	Qtr3	Qtr4	i	j
1989	23000	21500	24600	23300	20000	21100	23000	21600	24600	24300	22700	24800	69100	64400	69200	71800	5	4
1990	23400	24400	25300	25100	22500	24400	28700	27400	27100	28700	26500	27600	73100	72000	83200	82800	5	4
1991	23500	24600	25300	28700	25500	27600	28700	28400	28100	29700	28800	29300	73400	81800	85200	87800	5	4

Restructuring Data

Wide format to long format restructuring:

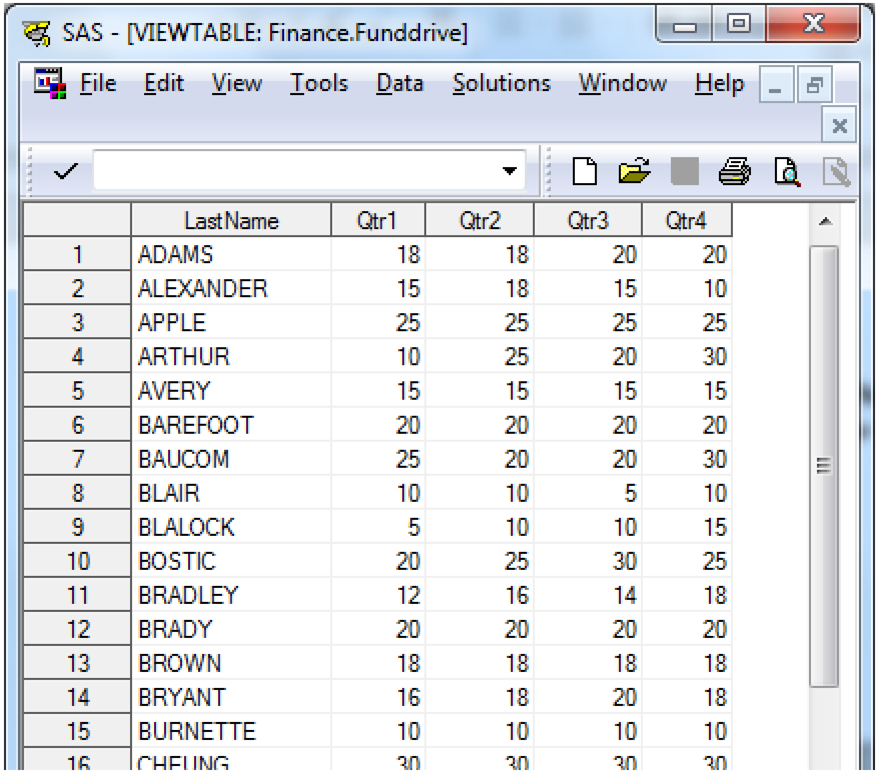
```
data funddrive_long(drop=qtr1-qtr4);
    set finance.funddrive;
    array contrib[4] qtr1-qtr4;
    do qtr=1 to 4;
        Amount=contrib[qtr];
        output;
    end;
run;
```



	LastName	Qtr1	Qtr2	Qtr3	Qtr4
1	ADAMS	18	18	20	20
2	ALEXANDER	15	18	15	10
3	APPLE	25	25	25	25
4	ARTHUR	10	25	20	30
5	AVERY	15	15	15	15
6	BAREFOOT	20	20	20	20
7	BAUCOM	25	20	20	30
8	BLAIR	10	10	5	10
9	BLALOCK	5	10	10	15
10	BOSTIC	20	25	30	25
11	BRADLEY	12	16	14	18
12	BRADY	20	20	20	20
13	BROWN	18	18	18	18
14	BRYANT	16	18	20	18
15	BURNETTE	10	10	10	10
16	CHEUNG	30	30	30	30

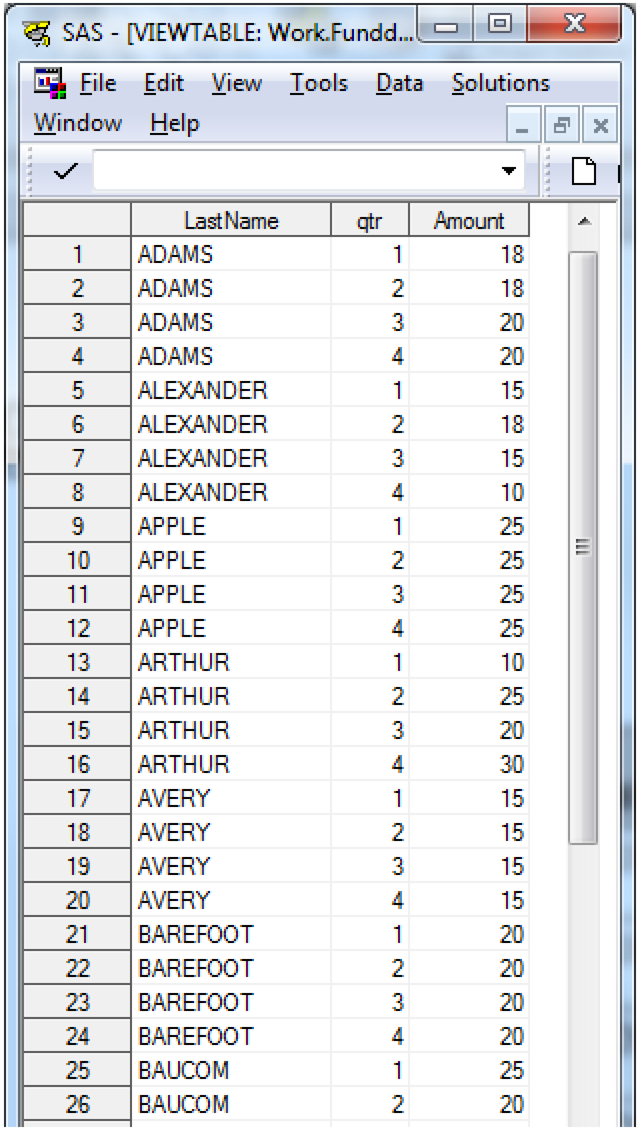
Restructuring Data

Wide



	LastName	Qtr1	Qtr2	Qtr3	Qtr4
1	ADAMS	18	18	20	20
2	ALEXANDER	15	18	15	10
3	APPLE	25	25	25	25
4	ARTHUR	10	25	20	30
5	AVERY	15	15	15	15
6	BAREFOOT	20	20	20	20
7	BAUCOM	25	20	20	30
8	BLAIR	10	10	5	10
9	BLALOCK	5	10	10	15
10	BOSTIC	20	25	30	25
11	BRADLEY	12	16	14	18
12	BRADY	20	20	20	20
13	BROWN	18	18	18	18
14	BRYANT	16	18	20	18
15	BURNETTE	10	10	10	10
16	CHEUNG	30	30	30	30

Long



	LastName	qtr	Amount
1	ADAMS	1	18
2	ADAMS	2	18
3	ADAMS	3	20
4	ADAMS	4	20
5	ALEXANDER	1	15
6	ALEXANDER	2	18
7	ALEXANDER	3	15
8	ALEXANDER	4	10
9	APPLE	1	25
10	APPLE	2	25
11	APPLE	3	25
12	APPLE	4	25
13	ARTHUR	1	10
14	ARTHUR	2	25
15	ARTHUR	3	20
16	ARTHUR	4	30
17	AVERY	1	15
18	AVERY	2	15
19	AVERY	3	15
20	AVERY	4	15
21	BAREFOOT	1	20
22	BAREFOOT	2	20
23	BAREFOOT	3	20
24	BAREFOOT	4	20
25	BAUCOM	1	25
26	BAUCOM	2	20