Data Set Manipulations

- A number of data step statements are useful in re-organizing your SAS data sets.

1) DELETE Statement

- Used to prevent an OBSERVATION from being output to a SAS data set.

- General Form:

  IF expression THEN DELETE;

- Two basic uses:
  1. Identify and remove invalid observations
  2. Select a subset of data from a large data set.

SAS Program

```sas
data ok;
  input gender $ weight @@;
  if weight>300 then delete;
  cards;
  m 157 f 123 m 122 f 911
  f 97 f 23 m 212 f 121
  f 97 f 97 f 23
  proc print;

data males;
  input gender $ weight @@;
  if gender='f' then delete;
  cards;
  m 157 f 123 m 122 f 911
  f 97 f 23 m 212 f 121
  f 97 f 23
  proc print;
```

SAS Output

```
The SAS System 1

<table>
<thead>
<tr>
<th>OBS</th>
<th>GENDER</th>
<th>WEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>m</td>
<td>157</td>
</tr>
<tr>
<td>2</td>
<td>f</td>
<td>123</td>
</tr>
<tr>
<td>3</td>
<td>m</td>
<td>122</td>
</tr>
<tr>
<td>4</td>
<td>f</td>
<td>97</td>
</tr>
<tr>
<td>5</td>
<td>f</td>
<td>23</td>
</tr>
<tr>
<td>6</td>
<td>m</td>
<td>212</td>
</tr>
<tr>
<td>7</td>
<td>f</td>
<td>121</td>
</tr>
<tr>
<td>8</td>
<td>f</td>
<td>97</td>
</tr>
<tr>
<td>9</td>
<td>f</td>
<td>23</td>
</tr>
</tbody>
</table>
```

```
The SAS System 2

<table>
<thead>
<tr>
<th>OBS</th>
<th>GENDER</th>
<th>WEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>m</td>
<td>157</td>
</tr>
<tr>
<td>2</td>
<td>m</td>
<td>122</td>
</tr>
<tr>
<td>3</td>
<td>m</td>
<td>212</td>
</tr>
<tr>
<td>4</td>
<td>97</td>
<td></td>
</tr>
</tbody>
</table>
```

2) Subsetting IF Statement

- Selects OBSERVATIONS from a SAS data set.
- An IF statement without a THEN clause deletes observations whose IF expression is false.

- Example:

```sas
  IF SEX = 'M';
  IF SEX ^= 'M' THEN DELETE;
```

Equivalent

```sas
  IF SEX ^= 'M' THEN DELETE;
```

```
```

46
3) **KEEP and DROP Statements**

- **DROP** is used to eliminate selected **VARIABLES** from a SAS data set.

- **KEEP** is used to retain selected **VARIABLES** in a SAS data set. All variables not included in the **KEEP** statement are automatically dropped.

- General Form: 
  
  ```
  DROP variable_1 variable_2 ... variable_n;
  KEEP variable_1 variable_2 ... variable_n;
  ```

**SAS Program**

```sas
data alpha;
  input id $ q1-q10;
  y1=q1+q2+q3+q4+q5 +q6+q7+q8+q9+q10;
  y2=sum(of q1-q10);
  drop q1-q10;
  cards;
  01 1 2 1 4 1 5 2 5 1
  02 2 3 1 5 1 2 . 5 3 2
  03 1 3 1 3 1 1 1 1 2 1
  04 3 2 1 4 1 . 1 1 3 2
  05 ............
  06 5 2 1 5 1 . 1 2 3 3
  07 4 1 1 5 1 3 1 1 2 4
end;
proc print;
```

**SAS Output**

```plaintext
The SAS System

<table>
<thead>
<tr>
<th>OBS</th>
<th>ID</th>
<th>Y1</th>
<th>Y2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>01</td>
<td>.</td>
<td>22</td>
</tr>
<tr>
<td>2</td>
<td>02</td>
<td>.</td>
<td>24</td>
</tr>
<tr>
<td>3</td>
<td>03</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>04</td>
<td>.</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>05</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>6</td>
<td>06</td>
<td>.</td>
<td>23</td>
</tr>
<tr>
<td>7</td>
<td>07</td>
<td>23</td>
<td>23</td>
</tr>
</tbody>
</table>
```

- DROP and KEEP are used to eliminate unnecessary variables from SAS data sets. Eliminating such variables reduces the memory and disk space resources required for your SAS programs.

- use the statement that allows you to specify a shorter variable list. For example, given a program with 10 variables, use KEEP if you want to retain less than 5 variables. Use DROP if you want to retain more than 5 variables.

- **KEEP** and **DROP** lists can also be specified as an option on the **DATA** statement, for example:

```sas
DATA DSM3 (KEEP=ID Y1 Y2);
```
4) **DELETE versus DROP**

- it is important to remember that **DELETE** is used to manipulate **OBSERVATIONS** and **DROP** works on **VARIABLES**.

<table>
<thead>
<tr>
<th>OBS1</th>
<th>VAR1</th>
<th>VAR2</th>
<th>VAR3</th>
<th>VAR4</th>
<th>DELETE</th>
<th>DROP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBS2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBS3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBS4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SAS Program**

```sas
data dsm3;
  input age gender $ weight;
  if gender='m' then delete;
  drop gender;
cards;
  7 f 50  \(\text{be careful}\)
  7 m 65  \(\text{about the order}\)
  13 f 75
  13 m 92
  27 f 115
  27 m 130
  34 f 120
  34 m 175
  42 f 115
  42 m 186
  60 f 112
  60 m 145
;proc print;
```

**SAS Output**

```
The SAS System 1

<table>
<thead>
<tr>
<th>OBS</th>
<th>AGE</th>
<th>WEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>75</td>
</tr>
<tr>
<td>3</td>
<td>27</td>
<td>115</td>
</tr>
<tr>
<td>4</td>
<td>34</td>
<td>120</td>
</tr>
<tr>
<td>5</td>
<td>42</td>
<td>115</td>
</tr>
<tr>
<td>6</td>
<td>60</td>
<td>112</td>
</tr>
</tbody>
</table>
```

- the example above creates a data set called DSM3 containing only observations whose value for GENDER is not equal to 'm' and only the AGE and WEIGHT variables.
- what happens if one reverses the order of the IF and DROP statements?
5) SET Statement

- the SET statement is used to **transfer** data from an **existing** SAS data set to a new SAS data set. Each time it is executed, SAS retrieves another observation from the old SAS data set and adds it to the new SAS data set.

- the SET statement may be used (in conjunction with other statements; e.g. conditional statements) to **combine several small SAS data sets** into a larger SAS data set or to **subset a large SAS data set** into several smaller SAS data sets.

- TASK — given the names, gender, ages, heights and weights of students in a class, create 3 SAS data sets. The first data set should contain all members of the class, the 2nd data set should consist of males only, and the 3rd data set should consist of females only.

### SAS Program

```sas
data everyone;
  input name$ gender$ age weight;
  cards;
  Alfred   m  14  112.5
  Alice    f  13  84.0
  Brenda   f  13  98.0
  Barbara  f  12  102.5
  Bernadette f  15  83.0
  James    m  12  84.5
  Jane     f  12  101.5
  Henry    m  15  150.7
.ends;

data males;
  set everyone;
  if gender='m';
data females;
  set everyone;
  if gender='f';
proc print data=males;
proc print data=females;
```

### SAS Output

#### The SAS System 1

<table>
<thead>
<tr>
<th>OBS</th>
<th>NAME</th>
<th>GENDER</th>
<th>AGE</th>
<th>WEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alfred</td>
<td>m</td>
<td>14</td>
<td>112.5</td>
</tr>
<tr>
<td>2</td>
<td>James</td>
<td>m</td>
<td>12</td>
<td>84.5</td>
</tr>
<tr>
<td>3</td>
<td>Henry</td>
<td>m</td>
<td>15</td>
<td>150.7</td>
</tr>
</tbody>
</table>

#### The SAS System 2

<table>
<thead>
<tr>
<th>OBS</th>
<th>NAME</th>
<th>GENDER</th>
<th>AGE</th>
<th>WEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alice</td>
<td>f</td>
<td>13</td>
<td>84.0</td>
</tr>
<tr>
<td>2</td>
<td>Brenda</td>
<td>f</td>
<td>13</td>
<td>98.0</td>
</tr>
<tr>
<td>3</td>
<td>Barbara</td>
<td>f</td>
<td>12</td>
<td>102.5</td>
</tr>
<tr>
<td>4</td>
<td>Bernadette</td>
<td>f</td>
<td>15</td>
<td>83.0</td>
</tr>
<tr>
<td>5</td>
<td>Jane</td>
<td>f</td>
<td>12</td>
<td>101.5</td>
</tr>
</tbody>
</table>

- the SET statement can be used in concatenating or interleaving SAS data sets.
# Data Set Manipulations

- **Concatenating SAS Data Sets with SET**

  - The SET statement can be used to combine two or more data sets 'vertically' --- this is called Concatenation.

  - **All variables** found in **any of the small data sets** will be **transferred** to the new SAS data set.

  - Variables defined on two or more of the data sets must be defined **consistently**; e.g., they must be either NUMERIC or CHARACTER on all of the data sets.

  - If a variable was not present on one of the 'component' data sets, then its observations in the 'concatenated' data set will have **missing values** for that variable.

  > have to be defined as same or each dataset

<table>
<thead>
<tr>
<th>ONE</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>OBS2</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>OBSn</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>TWO</strong></td>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>OBS1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>OBS2</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>OBSn</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BOTH</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>OBS2</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>OBSn</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>OBSn+1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>OBSn+2</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>OBSn+m</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

**SAS Code to Concatenate SAS Data Sets**

```
DATA ONE;
  INPUT X Y;
  CARDS;
  3 7
  2 9
  more data lines
  5 5
DATA TWO;
  INPUT X Y;
  CARDS;
  6 3
  7 5
  more data lines
  9 9
DATA BOTH;
  SET ONE TWO;
```
Data Set Manipulations

- Interleaving SAS Data Sets with SET

  - the SET statement can be used to combine two or more SAS data sets into one large SAS data set in which the observations are sorted on the basis of one or more variables.

  - all data sets must be sorted on the variables of interest prior to interleaving.

  - you may interleave data sets using as many 'sort' variables as you want.

<table>
<thead>
<tr>
<th>ONE</th>
<th>( x )</th>
<th>( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>OBS2</td>
<td>19</td>
<td>8</td>
</tr>
<tr>
<td>OBS3</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>OBS4</td>
<td>32</td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TWO</th>
<th>( x )</th>
<th>( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>OBS2</td>
<td>25</td>
<td>6</td>
</tr>
<tr>
<td>OBS3</td>
<td>43</td>
<td>12</td>
</tr>
<tr>
<td>OBS4</td>
<td>62</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BOTHINT</th>
<th>( x )</th>
<th>( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>OBS2</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>OBS3</td>
<td>19</td>
<td>8</td>
</tr>
<tr>
<td>OBS4</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>OBS5</td>
<td>25</td>
<td>6</td>
</tr>
<tr>
<td>OBS6</td>
<td>32</td>
<td>9</td>
</tr>
<tr>
<td>OBS7</td>
<td>43</td>
<td>12</td>
</tr>
<tr>
<td>OBS8</td>
<td>62</td>
<td>3</td>
</tr>
</tbody>
</table>

SAS Code to Interleave SAS Data Sets

```sas
DATA ONE;
  INPUT X Y;
  CARDS;
  5  2
  19  8
  20  4
  32  9
DATA TWO;
  INPUT X Y;
  CARDS;
  16  1
  25  6
  43  12
  62  3
PROC SORT DATA=ONE;
  BY X;
PROC SORT DATA=TW0;
  BY X;
DATA BOTHINT;
  SET ONE TWO;
  BY X;
```

you can also set first then sorting second
## Data Set Manipulations

### 6) MERGE Statement

- The MERGE statement transfers data from existing SAS data sets to new SAS data sets (as did the SET statement).

- The MERGE statement joins observations from two or more SAS data sets into single observations in a new SAS data set.

- Several types of merging exist including One-to-One Merging, Match Merging, and Table Look-up.

<table>
<thead>
<tr>
<th>ONE</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>OBS2</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>OBS9</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TWO</th>
<th>Z1</th>
<th>Z2</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>OBS2</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>OBS9</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BOTHMGD</th>
<th>X</th>
<th>Y</th>
<th>Z1</th>
<th>Z2</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>1</td>
<td>7</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>OBS2</td>
<td>8</td>
<td>3</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>OBS9</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

### SAS Code to One-to-One Merge SAS Data Sets

```
DATA ONE;
  INPUT X Y;
  CARDS;
  1  7
  8  3
  more data lines
  3  3
DATA TWO;
  INPUT Z1 Z2;
  CARDS;
  2  9
  2  8
  more data lines
  4  4
DATA BOTHMGD;
  MERGE ONE TWO;
```
Data Set Manipulations

- One-to-One Merging - cont.

- if one data set has more observations than the other(s) being merged, the new data set has the greater number of observations. Data values are automatically assigned as missing for the variables in the extra observations.

- if a variable name exists on two or more data sets, the new data set takes on the values from the LAST data set listed in the MERGE statement.

```
data driver;
  input name$ city$ code$;
  cards;
  Cathy Portland a
  Nancy Raleigh b
  Sue Houston c
  Mary Seattle d
  Joanna Tulsa e
;;;
data vehicle;
  input year make$ code$;
  cards;
  1977 Ford z
  1971 Pontiac y
  1989 Chevy x
  1981 Ford v
;;;
data together;
  merge driver vehicle;
  proc print;
```

SAS Program

```
I don't use a by statement
- the second data set will overlay the first
so be careful in the ordering
```

SAS Output

```
The SAS System 1

<table>
<thead>
<tr>
<th>OBS</th>
<th>NAME</th>
<th>CITY</th>
<th>CODE</th>
<th>YEAR</th>
<th>MAKE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cathy</td>
<td>Portland</td>
<td>z</td>
<td>1977</td>
<td>Ford</td>
</tr>
<tr>
<td>2</td>
<td>Nancy</td>
<td>Raleigh</td>
<td>y</td>
<td>1971</td>
<td>Pontiac</td>
</tr>
<tr>
<td>3</td>
<td>Sue</td>
<td>Houston</td>
<td>x</td>
<td>1989</td>
<td>Chevy</td>
</tr>
<tr>
<td>4</td>
<td>Mary</td>
<td>Seattle</td>
<td>v</td>
<td>1981</td>
<td>Ford</td>
</tr>
<tr>
<td>5</td>
<td>Joanna</td>
<td>Tulsa</td>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
**Data Set Manipulations**

- **Match Merging**
  - used to MERGE two or more SAS data sets on the basis of some common variable; e.g., ID.
  - at least one variable must be common to all data sets being merged in a match merge.
  - each data set must be sorted on the basis of the variable on which the merge is to occur.

<table>
<thead>
<tr>
<th>ONE</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>OBS2</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>OBS3</td>
<td>5</td>
<td>59</td>
</tr>
<tr>
<td>OBS4</td>
<td>6</td>
<td>79</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TWO</th>
<th>X</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>OBS2</td>
<td>3</td>
<td>17</td>
</tr>
<tr>
<td>OBS3</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>OBS4</td>
<td>5</td>
<td>24</td>
</tr>
<tr>
<td>OBS5</td>
<td>6</td>
<td>37</td>
</tr>
</tbody>
</table>

**BOTHMM**

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>1</td>
<td>12</td>
<td>5</td>
</tr>
<tr>
<td>OBS2</td>
<td>2</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>OBS3</td>
<td>3</td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>OBS4</td>
<td>4</td>
<td></td>
<td>23</td>
</tr>
<tr>
<td>OBS5</td>
<td>5</td>
<td>59</td>
<td>24</td>
</tr>
<tr>
<td>OBS6</td>
<td>6</td>
<td>79</td>
<td>37</td>
</tr>
</tbody>
</table>

Unique values = 1-6

**SAS Code to Match Merge SAS Data Sets**

```sas
DATA ONE;
  INPUT X Y;
  CARDS;
  1 12
  2 24
  5 59
  6 79
DATA TWO;
  INPUT X Z;
  CARDS;
  1  5
  3 17
  4 23
  5 24
  6 37
PROC SORT DATA=ONE;
  BY X;
PROC SORT DATA=_TWO;
  BY X;
DATA BOTHMM;
  MERGE ONE TWO;
  BY X;
```

By Stuarts
To match OBS.
\section*{Data Set Manipulations}

\subsection*{Match Merging}

- if one data set has variables that are \textbf{not present} on the other data sets being merged, then the new data set has \textbf{missing values} for all variables unique to the other data sets.

- if a variable (which \textbf{is not a merging variable}) exists on two or more data sets, the new data set takes on the values from the \textbf{LAST} data set specified in the \texttt{MERGE} statement.

```sas
data person;
  input name$ gender$ code$;
  cards;
  Mary f a
  Ann f b
  Tom m c
  Harv m d
  Julie f e
;
proc sort;
  by name;
data place;
  input name$ city$ region code$;
  cards;
  Jose Erie 7 z
  Mary Miami 5 y
  Ann Tampa 2 x
  Harv Jud 3 v
;
proc sort;
  by name;
data combined;
  merge person place;
  by name;
proc print;
```

\textbf{SAS Program}  
\texttt{Data between play codes}

\textbf{SAS Output}  
\texttt{Combine person place; by name; proc print;}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
\textbf{OBS} & \textbf{NAME} & \textbf{GENDER} & \textbf{CODE} & \textbf{CITY} & \textbf{REGION} \\
\hline
1 & Ann & f & x & Tampa & 2 \\
2 & Harv & m & v & Jud & 3 \\
3 & Jose & m & z & Erie & 7 \\
4 & Julie & f & e & & . \\
5 & Mary & f & y & Miami & 5 \\
6 & Tom & m & c & & . \\
\hline
\end{tabular}
Data Set Manipulations

**Table Look-up**

- table look-up is a variation on a match merge. In this case, one data set contains your table info and has an identifier variable along with corresponding descriptions on it. Other data sets contain the identifier variable along with other variables of interest.

- the identifier variable must be common to all data sets being merged.

- each data set **MUST** be sorted on the basis of the identifier variable prior to merging.

<table>
<thead>
<tr>
<th>ONE</th>
<th>NAME</th>
<th>ST_AB</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>JON SMITH</td>
<td>AK</td>
</tr>
<tr>
<td>OBS2</td>
<td>CHAD JOHNSON</td>
<td>AK</td>
</tr>
<tr>
<td>OBS3</td>
<td>SUE OLSON</td>
<td>AL</td>
</tr>
<tr>
<td>OBS4</td>
<td>JIM BOBLE</td>
<td>AR</td>
</tr>
<tr>
<td>OBS5</td>
<td>SALLY BROWN</td>
<td>AR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TWO</th>
<th>ST_AB</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>AK</td>
<td>ALASKA</td>
</tr>
<tr>
<td>OBS2</td>
<td>AL</td>
<td>ALABAMA</td>
</tr>
<tr>
<td>OBS3</td>
<td>AR</td>
<td>ARKANSAS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BOTHTL</th>
<th>NAME</th>
<th>ST_AB</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBS1</td>
<td>JON SMITH</td>
<td>AK</td>
<td>ALASKA</td>
</tr>
<tr>
<td>OBS2</td>
<td>CHAD JOHNSON</td>
<td>AK</td>
<td>ALASKA</td>
</tr>
<tr>
<td>OBS3</td>
<td>SUE OLSON</td>
<td>AL</td>
<td>ALABAMA</td>
</tr>
<tr>
<td>OBS4</td>
<td>JIM BOBLE</td>
<td>AR</td>
<td>ARKANSAS</td>
</tr>
<tr>
<td>OBS5</td>
<td>SALLY BROWN</td>
<td>AR</td>
<td>ARKANSAS</td>
</tr>
</tbody>
</table>

**SAS Code to Table Look-up**

**SAS Data Sets**

```sas
DATA ONE;
   INPUT NAME $ 1-20 ST_AB $;
   CARDS;
   JON SMITH     AK
   CHAD JOHNSON  AK
   SUE OLSON     AL
   JIM BOBLE     AR
   SALLY BROWN   AR
DATA TWO;
   INPUT ST_AB $ 05 STATE $15,; CARDS;
   AK ALASKA
   AL ALABAMA
   AR ARKANSAS
```

```sas
PROC SORT DATA=ONE;
   BY ST_AB;
PROC SORT DATA= TWO;
   BY ST_AB;
DATA BOTHTL;
   MERGE ONE TWO;
   BY ST_AB;
```
Combining Data Sets Using 1-1 Match Merge

SAS Program

```
title1 'Combining Data Sets --- pg.130-31 Little SAS Book';
data descriptions;
  input CodeNum $ 1-4 Name $ 6-14 Description $ 15-60;
cards;
A206 Mokka Coffee buttercream in dark chocolate
A536 Walnoot Walnut halves in bed of dark chocolate
B713 Frambozen Raspberry marzipan in milk chocolate
C865 Vanille Vanilla-flavored rolled in ground filberts
K014 Kroon Milk chocolate with a mint cream center
K086 Koning Filbert paste in dark chocolate
M315 Pyramide White with dark chocolate trimming
S163 Orbais Chocolate cream in dark chocolate

;:;
```

```
data sales;
  input CodeNum $ 1-4 PiecesSold 6-7;
cards;
C865 15
K086 9
A536 21
S163 34
K014 1
A206 12
B713 29
;:;
```

```
proc sort;
  by CodeNum;
run;
```

```
data chocolates;
  merge sales descriptions;
  by CodeNum;
proc print;
  title1 "Today's Chocolate Sales";
```

SAS Output

Today's Chocolate Sales

<table>
<thead>
<tr>
<th>Code</th>
<th>Pieces</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obs</td>
<td>Num</td>
<td>Sold</td>
<td>Description</td>
</tr>
<tr>
<td>1</td>
<td>A206</td>
<td>12</td>
<td>Mokka</td>
</tr>
<tr>
<td>2</td>
<td>A536</td>
<td>21</td>
<td>Walnoot</td>
</tr>
<tr>
<td>3</td>
<td>B713</td>
<td>29</td>
<td>Frambozen</td>
</tr>
<tr>
<td>4</td>
<td>C865</td>
<td>15</td>
<td>Vanille</td>
</tr>
<tr>
<td>5</td>
<td>K014</td>
<td>1</td>
<td>Kroon</td>
</tr>
<tr>
<td>6</td>
<td>K086</td>
<td>9</td>
<td>Koning</td>
</tr>
<tr>
<td>7</td>
<td>M315</td>
<td></td>
<td>Pyramide</td>
</tr>
<tr>
<td>8</td>
<td>S163</td>
<td>34</td>
<td>Orbais</td>
</tr>
</tbody>
</table>
Combining Data Sets Using 1-Many Match Merge

SAS Program

options ls=80;
title1 'Combining Data Sets --- One-to-Many Match Merge';
title2 'pg.132-33 Little SAS Book';

data videos;
   input Name $ 1-29 ExerciseType $ RegularPrice;
   cards;
   Adorable Abs         aerobics 12.99
   Aerobic Childcare for Parents aerobics 13.99
   Judy Murphy's Fun Fitness step 12.99
   Lavonnes' Low Impact Workout aerobics 13.99
   Muscle Makers         weights 15.99
   Rock N Roll Step Workout step 12.99
   ;
;
proc sort data=videos;
   by ExerciseType;
   run;

data discount;
   input ExerciseType $ Adjustment;
   cards;
   aerobics .20
   step .30
   weights .25
   ;
;
   data prices;
   merge videos discount;
   by ExerciseType;
   NewPrice=round(RegularPrice-(RegularPrice*Adjustment), .01);
   ;
proc print;
   run;

SAS Output

Combining Data Sets --- One-to-Many Match Merge
pg.132-33 Little SAS Book

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Exercise Type</th>
<th>Regular Price</th>
<th>Adjustment</th>
<th>New Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Adorable Abs</td>
<td>aerobics</td>
<td>12.99</td>
<td>0.20</td>
<td>10.39</td>
</tr>
<tr>
<td>2</td>
<td>Aerobic Childcare for Parents</td>
<td>aerobics</td>
<td>13.99</td>
<td>0.20</td>
<td>11.19</td>
</tr>
<tr>
<td>3</td>
<td>Lavonnes' Low Impact Workout</td>
<td>aerobics</td>
<td>13.99</td>
<td>0.20</td>
<td>11.19</td>
</tr>
<tr>
<td>4</td>
<td>Judy Murphy's Fun Fitness</td>
<td>step</td>
<td>12.99</td>
<td>0.30</td>
<td>9.09</td>
</tr>
<tr>
<td>5</td>
<td>Rock N Roll Step Workout</td>
<td>step</td>
<td>12.99</td>
<td>0.30</td>
<td>9.09</td>
</tr>
<tr>
<td>6</td>
<td>Muscle Makers</td>
<td>weights</td>
<td>15.99</td>
<td>0.25</td>
<td>11.99</td>
</tr>
</tbody>
</table>
Combining Data Sets Using 1-Many Match Merge

SAS Program

```sas
Title1 'Combining Data Sets - Merging Summary Stats with Original Data';
Title2 'pg.134-35 Little SAS Book, 2nd Edition';

Data videos;
  Input Name $ 1-29 ExerciseType $ Sales;
  Cards;
  Adorable Abs aerobics 1930
  Aerobic Childcare for Parents aerobics 2250
  Judy Murphy's Fun Fitness step 4150
  Lavonnes' Low Impact Workout aerobics 1130
  Muscle Makers weights 2230
  Rock N Roll Step Workout step 1190
;

Proc sort data=videos;
  By ExerciseType;
  Run;

*** Use Means procedure to calculate Sales Totals to
*** merge back with original data.;
Proc means nocrunch data=videos;
  Var Sales;
  By ExerciseType;
  Output Out=SummaryData Sum(Sales)=Total;
  Run;

Proc print data=SummaryData;
  Title2 'Summary Data Set';
  Run;

*** Merge the summary data back with original data by Exercise Type.;
Data videosSummary;
  Merge videos SummaryData;
  By ExerciseType;
  Percent=Sales/Total*100;
  ;

*** By statement separates Exercise types on printout.;
*** ID statement causes Exercise Type to replace OBS on output.;
Proc print;
  By ExerciseType;
  Id ExerciseType;
  Var Name Sales Total Percent;
  Title2 'Sales Share by Type of Exercise';
  Run;
```
Combining Data Sets Using 1-Many Match Merge

SAS Output

Combining Data Sets --- Merging Summary Stats with Original Data 1
Summary Data Set
02:20 Monday, March 19, 2001

<table>
<thead>
<tr>
<th>Obs</th>
<th>Exercise</th>
<th>Type</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>aerobics</td>
<td>0</td>
<td>3</td>
<td></td>
<td>5310</td>
</tr>
<tr>
<td>2</td>
<td>step</td>
<td>0</td>
<td>2</td>
<td></td>
<td>5340</td>
</tr>
<tr>
<td>3</td>
<td>weights</td>
<td>0</td>
<td>1</td>
<td></td>
<td>2230</td>
</tr>
</tbody>
</table>

Combining Data Sets --- Merging Summary Stats with Original Data 2
Sales Share by Type of Exercise
02:20 Monday, March 19, 2001

<table>
<thead>
<tr>
<th>Exercise Type</th>
<th>Name</th>
<th>Sales</th>
<th>Total</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>aerobics</td>
<td>Adorable Abs</td>
<td>1930</td>
<td>5310</td>
<td>36.347</td>
</tr>
<tr>
<td></td>
<td>Aerobic Childcare for Parents</td>
<td>2250</td>
<td>5310</td>
<td>42.373</td>
</tr>
<tr>
<td></td>
<td>Lavonnes' Low Impact Workout</td>
<td>1130</td>
<td>5310</td>
<td>21.281</td>
</tr>
<tr>
<td>step</td>
<td>Judy Murphy's Fun Fitness</td>
<td>4150</td>
<td>5340</td>
<td>77.715</td>
</tr>
<tr>
<td></td>
<td>Rock N Roll Step Workout</td>
<td>1190</td>
<td>5340</td>
<td>22.285</td>
</tr>
<tr>
<td>weights</td>
<td>Muscle Makers</td>
<td>2230</td>
<td>2230</td>
<td>100.000</td>
</tr>
</tbody>
</table>
Combining Data Sets -- Grand Total with Original Data

SAS Program

title1 'Combining Data Sets --- Combining Grand Total with Original Data';
title2 'pg.136-37 Little SAS Book, 2nd Edition';

data videos;
  input Name $ 1-29 ExerciseType $ Sales;
  cards;
  Adorable Abs     aerobics 1930
  Aerobic Childcare for Parents aerobics 2250
  Judy Murphy's Fun Fitness step 4150
  Lavonnes' Low Impact Workout aerobics 1130
  Muscle Makers     weights 2230
  Rock N Roll Step Workout step 1190
  ;
  proc means noprint data=videos;
  var Sales;
  output out=summarydata SUM(Sales)=GrandTotal;
  proc print data=summarydata;
  title2 'Summary Data Set';
  run;

data videosummary;
  if _n_=1 then set summarydata; *** Set retains value of GrandTotal.;
  set videos;
  Percent=Sales/GrandTotal*100;
  ;
  proc print;
  var Name ExerciseType Sales GrandTotal Percent;
  title2 'Sales Share by Type of Exercise';
  run;

Combining Data Sets --- Combining Grand Total with Original Data  1
Summary Data Set

<table>
<thead>
<tr>
<th>Obs</th>
<th>TYPE_</th>
<th>FREQ</th>
<th>Grand Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>6</td>
<td>12880</td>
</tr>
</tbody>
</table>

Combining Data Sets --- Combining Grand Total with Original Data  2
Sales Share by Type of Exercise

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Exercise Type</th>
<th>Sales</th>
<th>Grand Total</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Adorable Abs</td>
<td>aerobics</td>
<td>1930</td>
<td>12880</td>
<td>14.9845</td>
</tr>
<tr>
<td>2</td>
<td>Aerobic Childcare for Parents</td>
<td>aerobics</td>
<td>2250</td>
<td>12880</td>
<td>17.4689</td>
</tr>
<tr>
<td>3</td>
<td>Judy Murphy's Fun Fitness</td>
<td>step</td>
<td>4150</td>
<td>12880</td>
<td>32.2205</td>
</tr>
<tr>
<td>4</td>
<td>Lavonnes' Low Impact Workout</td>
<td>aerobics</td>
<td>1130</td>
<td>12880</td>
<td>8.7733</td>
</tr>
<tr>
<td>5</td>
<td>Muscle Makers</td>
<td>weights</td>
<td>2230</td>
<td>12880</td>
<td>17.3137</td>
</tr>
<tr>
<td>6</td>
<td>Rock N Roll Step Workout</td>
<td>step</td>
<td>1190</td>
<td>12880</td>
<td>9.2391</td>
</tr>
</tbody>
</table>
Using DO Loops to Simplify Data Entry

- Iterative DO loops can be used to greatly reduce the amount of data that needs to be entered when each observation has one or more categorical or index variables.

- General form of an iterative DO loop:

  DO index-variable = specification-1, specification-2, ......, specification-n;

  additional SAS statements

  END;

- Where:

  index-variable - names a variable whose value governs the execution of the DO group.

  specification - refers to a single value or an expression of the form start TO stop BY increment. WHILE and UNTIL clauses may be part of the specification.

  additional SAS statements - can be any data step statements like assignment statements or conditional statements.

- The index variable may be either character or numeric with special date-time informats possible as well.

- Examples:

  do i = 5 to 25 by 5;
  do x = 3, 7, 9, 15;
  do day = 'Friday', 'Saturday', 'Sunday';
  do date = '01JAN60'd to '30JUN60'd by 1;
  do z = .1 to .9 by .1, 1 to 10 by 1, 20 to 100 by 10;
  do y = 5 to 50 while(abd<thx);
  do counter = 10 to 0 by -2 while(type='C');
  do i = 0.5 to 2.5 by .02 until((x/5) < z);
Using DO Loops to Simplify Data Entry

SAS Program

```sas
options linesize=80;
title 'U.S. Population Decennial Census Values';
data uspop;
  do year=1790 to 1980 by 10;
    input pop @@;
    year2=year**2;
    output;
  end;
cards;
  3929214 5308483 7239881 9638453 12860702 17063353
  23191876 31443321 38558371 50189209 62979766 76212168
  92228496 106021537 123202624 132164569 151325798 179323175
  203302031 226545805
proc print;
```

SAS Output

<table>
<thead>
<tr>
<th>OBS</th>
<th>YEAR</th>
<th>POP</th>
<th>YEAR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1790</td>
<td>3929214</td>
<td>3204100</td>
</tr>
<tr>
<td>2</td>
<td>1800</td>
<td>5308483</td>
<td>3240000</td>
</tr>
<tr>
<td>3</td>
<td>1810</td>
<td>7239881</td>
<td>3276100</td>
</tr>
<tr>
<td>4</td>
<td>1820</td>
<td>9638453</td>
<td>3312400</td>
</tr>
<tr>
<td>5</td>
<td>1830</td>
<td>12860702</td>
<td>3348900</td>
</tr>
<tr>
<td>6</td>
<td>1840</td>
<td>17063353</td>
<td>3385600</td>
</tr>
<tr>
<td>7</td>
<td>1850</td>
<td>23191876</td>
<td>3422500</td>
</tr>
<tr>
<td>8</td>
<td>1860</td>
<td>31443321</td>
<td>3459600</td>
</tr>
<tr>
<td>9</td>
<td>1870</td>
<td>38558371</td>
<td>3496900</td>
</tr>
<tr>
<td>10</td>
<td>1880</td>
<td>50189209</td>
<td>3534400</td>
</tr>
<tr>
<td>11</td>
<td>1890</td>
<td>62979766</td>
<td>3572100</td>
</tr>
<tr>
<td>12</td>
<td>1900</td>
<td>76212168</td>
<td>3610000</td>
</tr>
<tr>
<td>13</td>
<td>1910</td>
<td>92228496</td>
<td>3648100</td>
</tr>
<tr>
<td>14</td>
<td>1920</td>
<td>106021537</td>
<td>3686400</td>
</tr>
<tr>
<td>15</td>
<td>1930</td>
<td>123202624</td>
<td>3724900</td>
</tr>
<tr>
<td>16</td>
<td>1940</td>
<td>132164569</td>
<td>3763600</td>
</tr>
<tr>
<td>17</td>
<td>1950</td>
<td>151325798</td>
<td>3802500</td>
</tr>
<tr>
<td>18</td>
<td>1960</td>
<td>179323175</td>
<td>3841600</td>
</tr>
<tr>
<td>19</td>
<td>1970</td>
<td>203302031</td>
<td>3880900</td>
</tr>
<tr>
<td>20</td>
<td>1980</td>
<td>226545805</td>
<td>3920400</td>
</tr>
</tbody>
</table>
Using DO Loops to Simplify Data Entry

SAS Program

options linesize=80;
title 'Split-Plot Analysis of Rat Data from Gill & Hafs, 1971';
data;
do rat=1 to 12;
do period=1 to 4;
   input wtgain @;
   pause
   if rat <=6 then treat='Pregnant';
   if rat > 6 then treat='Non-preg';
   output;
end;
end;
cards;
7.5 8.6 6.9 0.8 10.6 11.7 8.8 1.6 12.4 13.0 11.0 5.6
11.5 12.6 11.1 7.5 8.3 8.9 6.8 0.5 9.2 10.1 8.6 3.8
13.3 13.3 12.9 11.1 10.7 10.8 10.7 9.3 12.5 12.7 12.0 10.1
8.4 8.7 8.1 5.7 9.4 9.6 8.0 3.8 11.3 11.7 10.0 8.5
proc print;
proc glm;
class treat rat period;
model wtgain = treat rat(period) period treat*period rat*period(period); test h=treat e=rat(treat); test h=period treat*period e=rat*period(treat); means rat treat period period*period;

SAS Output

<table>
<thead>
<tr>
<th>OBS</th>
<th>RAT</th>
<th>PERIOD</th>
<th>WTGAIN</th>
<th>TREAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7.5</td>
<td>Pregnant</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>8.6</td>
<td>Pregnant</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>6.9</td>
<td>Pregnant</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>4</td>
<td>6.8</td>
<td>Pregnant</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>10.4</td>
<td>Pregnant</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>2</td>
<td>11.7</td>
<td>Pregnant</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>3</td>
<td>8.8</td>
<td>Pregnant</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>4</td>
<td>11.6</td>
<td>Pregnant</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>1</td>
<td>12.4</td>
<td>Pregnant</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>2</td>
<td>13.8</td>
<td>Pregnant</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>3</td>
<td>11.0</td>
<td>Pregnant</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>4</td>
<td>11.6</td>
<td>Pregnant</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>1</td>
<td>13.5</td>
<td>Pregnant</td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>2</td>
<td>12.6</td>
<td>Pregnant</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>3</td>
<td>11.1</td>
<td>Pregnant</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>4</td>
<td>7.5</td>
<td>Pregnant</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>1</td>
<td>8.3</td>
<td>Pregnant</td>
</tr>
<tr>
<td>18</td>
<td>5</td>
<td>2</td>
<td>8.9</td>
<td>Pregnant</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>3</td>
<td>6.8</td>
<td>Pregnant</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>4</td>
<td>6.5</td>
<td>Pregnant</td>
</tr>
<tr>
<td>21</td>
<td>6</td>
<td>1</td>
<td>9.2</td>
<td>Pregnant</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
<td>2</td>
<td>10.1</td>
<td>Pregnant</td>
</tr>
<tr>
<td>23</td>
<td>6</td>
<td>3</td>
<td>8.6</td>
<td>Pregnant</td>
</tr>
<tr>
<td>24</td>
<td>6</td>
<td>4</td>
<td>3.8</td>
<td>Pregnant</td>
</tr>
<tr>
<td>25</td>
<td>7</td>
<td>1</td>
<td>13.3</td>
<td>Non-preg</td>
</tr>
<tr>
<td>26</td>
<td>7</td>
<td>2</td>
<td>13.3</td>
<td>Non-preg</td>
</tr>
<tr>
<td>27</td>
<td>7</td>
<td>3</td>
<td>12.9</td>
<td>Non-preg</td>
</tr>
<tr>
<td>28</td>
<td>7</td>
<td>4</td>
<td>11.1</td>
<td>Non-preg</td>
</tr>
<tr>
<td>29</td>
<td>8</td>
<td>1</td>
<td>15.7</td>
<td>Non-preg</td>
</tr>
<tr>
<td>30</td>
<td>8</td>
<td>2</td>
<td>16.8</td>
<td>Non-preg</td>
</tr>
<tr>
<td>31</td>
<td>8</td>
<td>3</td>
<td>18.7</td>
<td>Non-preg</td>
</tr>
<tr>
<td>32</td>
<td>8</td>
<td>4</td>
<td>9.3</td>
<td>Non-preg</td>
</tr>
<tr>
<td>33</td>
<td>9</td>
<td>1</td>
<td>12.5</td>
<td>Non-preg</td>
</tr>
<tr>
<td>34</td>
<td>9</td>
<td>2</td>
<td>12.7</td>
<td>Non-preg</td>
</tr>
<tr>
<td>35</td>
<td>9</td>
<td>3</td>
<td>12.0</td>
<td>Non-preg</td>
</tr>
<tr>
<td>36</td>
<td>9</td>
<td>4</td>
<td>16.1</td>
<td>Non-preg</td>
</tr>
<tr>
<td>37</td>
<td>10</td>
<td>1</td>
<td>8.4</td>
<td>Non-preg</td>
</tr>
<tr>
<td>38</td>
<td>10</td>
<td>2</td>
<td>8.7</td>
<td>Non-preg</td>
</tr>
<tr>
<td>39</td>
<td>10</td>
<td>3</td>
<td>8.1</td>
<td>Non-preg</td>
</tr>
<tr>
<td>40</td>
<td>10</td>
<td>4</td>
<td>8.0</td>
<td>Non-preg</td>
</tr>
<tr>
<td>41</td>
<td>11</td>
<td>1</td>
<td>9.4</td>
<td>Non-preg</td>
</tr>
<tr>
<td>42</td>
<td>11</td>
<td>2</td>
<td>9.6</td>
<td>Non-preg</td>
</tr>
<tr>
<td>43</td>
<td>11</td>
<td>3</td>
<td>8.0</td>
<td>Non-preg</td>
</tr>
<tr>
<td>44</td>
<td>11</td>
<td>4</td>
<td>7.8</td>
<td>Non-preg</td>
</tr>
<tr>
<td>45</td>
<td>12</td>
<td>1</td>
<td>12.3</td>
<td>Non-preg</td>
</tr>
<tr>
<td>46</td>
<td>12</td>
<td>2</td>
<td>11.7</td>
<td>Non-preg</td>
</tr>
<tr>
<td>47</td>
<td>12</td>
<td>3</td>
<td>10.9</td>
<td>Non-preg</td>
</tr>
<tr>
<td>48</td>
<td>12</td>
<td>4</td>
<td>8.5</td>
<td>Non-preg</td>
</tr>
</tbody>
</table>
Using Explicit Arrays in SAS

- arrays can be used in conjunction with DO loops to perform certain actions repetitively. Arrays and DO loops together enable the programmer to perform 'horizontal manipulation' in a SAS data step.

- general form of an explicit array:

  ```
  ARRAY array-name { subscript } array-elements ;
  ARRAY array-name { subscript } $ array-elements ;
  ```

  - where:

    array-name - names the array being created.
    subscript - refers to a single value or expression.

- given a 10 question survey with responses ranging from 1 to 5. Even numbered items are reverse scaled and need to be recoded (i.e., 1 should be changed to 5, 2 to 4, 4 to 2, and 5 to 1). Write a SAS program to recode the selected items and print the recoded data.

```
SAS Program

data likert;
  input q1 - q10;
  array qs(5) q2 q4 q6 q8 q10;
  do i = 1 to 5;
    qsi = abs(qs(i)-6);
  end;
  drop i;
  cards;
  1 1 1 1 1 1 1 1 1
  2 2 2 2 2 2 2 2 2
  3 3 3 3 3 3 3 3 3
  4 4 4 4 4 4 4 4 4
  5 5 5 5 5 5 5 5 5
  1 5 1 5 1 5 1 5 1 5
proc print;
```

```
SAS Output

      14:12 Tuesday, November 5, 1991  1

   OBS  Q1  Q2  Q3  Q4  Q5  Q6  Q7  Q8  Q9  Q10
     1     1     5     1     5     1     5     1     5     1     5
     2     2     4     4     4     4     4     4     4     4     4
     3     3     3     3     3     3     3     3     3     3     3
     4     4     2     4     2     4     2     4     4     4     2
     5     5     1     5     1     5     1     5     1     5     5
     6     1     1     1     1     1     1     1     1     1     1

This does a reverse scale

Part F between array+do include
    cntr=0;
    do =
      if qs(i)>3 then cntr+1;
    end;
    if cntr
```