Chapter 2

Data Manipulation in SAS

In this chapter we will use SAS to manipulate data using the DATA statement. We will illustrate the SAS commands using the following example.

2.1 An Example

Nineteen type I diabetic patients were allocated randomly to one of two treatments in a clinical trial. The patients received either drug E or drug N. After receiving the drug, the patients received a single dose of antihypertensive therapy for 14 days. Supine systolic blood pressure (SSBP) and supine diastolic blood pressure (SDBP) were measured at the period baseline and after 14 days of treatment (measuring in mmHg). We are interested in analyzing whether there is a difference in the ability of Drug E and Drug N to lower blood pressure.

The format of the data is

<table>
<thead>
<tr>
<th>number</th>
<th>patient id</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssbp_0</td>
<td>baseline supine systolic blood pressure</td>
</tr>
<tr>
<td>ssbp_14</td>
<td>supine systolic blood pressure after 14 days</td>
</tr>
<tr>
<td>sdbp_0</td>
<td>baseline supine diastolic blood pressure</td>
</tr>
<tr>
<td>sdbp_14</td>
<td>supine diastolic blood pressure after 14 days</td>
</tr>
</tbody>
</table>

The data is available on the website. ‘drugE.txt’ contains the data for the patients who received drug E, and ‘drugN.txt’ contains the data for the patients who received drug N.

The data is adapted from:

2.2 Reading in the data

Go to the class website and save the files `drugE.txt` and `drugN.txt` to the folder My Computer → Drive C: → My SAS files (If the folder My SAS files does not exist, then create it).

Here is the SAS program to read in the data for those patients who received drug E. We then use PROC PRINT to display the data.

```sas
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
   INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
   INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
PROC PRINT data=drugE NOOBS;
   TITLE "Diabetic blood pressure study";
   TITLE2 "Patients receiving drug E";
   FOOTNOTE "NOTE: 8 patients received drug E; 11 patients received drug N.";
RUN;
```

Make sure you understand what this program does (refer to the notes from the previous chapter). In this program we use multiple TITLEs and a FOOTNOTE. SAS remembers the titles and footnotes for output until they are changed. To cancel the footnote just use FOOTNOTE; (without the quotes). It is the same for the TITLE and TITLE2 command.

Submit the program, check the log and then the output window.

2.3 Creating variables

We create new variables in a DATA statement by using the assignment operator, =

```sas
   variable_name = expression;
```

For our drug E dataset we shall create a variable which measures the change in the systolic blood pressure from day 0 to day 14. Change the above program to read:

```sas
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
   INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
   ...
```

INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
diff_ssbp = ssbp_0 - ssbp_14;

PROC PRINT data=drugE NOOBS;
RUN;

Submit the program. We can see that only patient 21 has a systolic blood pressure which increases from day 0 to day 14. All other patients has a decreasing blood pressure of between 11 and 25 (mmHg). Note that this program keeps the TITLEs and FOOTNOTE from the previous program.

Now add this statement and submit your program:

diff_sdbp = sdbp_0 - sdbp_14;

Looking at the output window, we can see there is more variability in the change of the diastolic blood pressure. Here are some more examples of variables we can create:

max_ssbp = MAX(ssbp_0, ssbp_14);
max_sdbp = MAX(sdbp_0, sdbp_14);
drug = "E";

Add each of these statements to your program. What do these variables measure?

Exercise

Create variables for the following.

- The mean of the systolic blood pressure at day 0 and day 14.
- The mean of the diastolic blood pressure at day 0 and day 14.
- The log (to the base 10) of the systolic blood pressure at day 0.

2.4 Making Comparisons

We use the IF, THEN, and ELSE commands to make comparisons of variables in the DATA statement. The first form of the command is

IF some_comparison THEN action;
If `some_comparison` is true then we carry out `action`. Otherwise we do nothing. Another form of the command is is

```
IF some_comparison THEN action;
    ELSE another_action;
```

If `some_comparison` is true then carry out `action`, otherwise carry out `another_action`. We can combine multiple IFs as follows

```
IF some_comparison THEN action;
    ELSE IF another_comparison THEN another_action;
    ELSE yet_another_action;
```

Consider the following program for the diabetic patients who have taken drug E.

```
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
    INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
    INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
    IF (ssbp_0 > ssbp_14) THEN ssbp_lower = "Yes";

PROC PRINT data=drugE NOOBS;
RUN;
```

In this program `ssbp_lower` is equal to `Yes` if the systolic blood pressure decreased after 14 days. Note that the variable has no value if the blood pressure increased after 14 days. Change the IF command to

```
IF (ssbp_0 GT ssbp_14) THEN ssbp_lower = "Yes";
```

GT stands for **greater than**. In SAS there are both commands and symbols for greater than, less than, etc. Here is a table of the commands or symbols you can use for making comparisons in SAS:

<table>
<thead>
<tr>
<th>Command</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>EQ</td>
<td>=</td>
<td>is equal to</td>
</tr>
<tr>
<td>NE</td>
<td>^=</td>
<td>is not equal to</td>
</tr>
<tr>
<td>LT</td>
<td>&lt;</td>
<td>is less than</td>
</tr>
<tr>
<td>LE</td>
<td>&lt;=</td>
<td>is less than or equal to</td>
</tr>
<tr>
<td>GT</td>
<td>&gt;</td>
<td>is greater than</td>
</tr>
<tr>
<td>GE</td>
<td>&gt;=</td>
<td>is greater than or equal to</td>
</tr>
<tr>
<td>AND</td>
<td>&amp;</td>
<td>and</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td>or</td>
</tr>
</tbody>
</table>
Here is a program which defines variables which indicates over the 14 days, whether the systolic blood pressure decreased, the diastolic blood pressure decreased, and if both blood pressures decreased.

```
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
  INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
  INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
  /* ssbp_lower is "Yes" if systolic blood pressure decreased, "No" otherwise. */
  IF (ssbp_0 > ssbp_14) THEN ssbp_lower = "Yes";
    ELSE ssbp_lower="No";
  /* sdbp_lower is "Yes" if diastolic blood pressure decreased, "No" otherwise. */
  IF (sdbp_0 > sdbp_14) THEN sdbp_lower = "Yes";
    ELSE sdbp_lower="No";
  /* both_lower is "Yes" if both blood pressures decreased, "No" otherwise. */
  IF ((ssbp_0 > ssbp_14) & (sdbp_0 > sdbp_14)) THEN both_lower = "Yes";
    ELSE both_lower="No";

PROC PRINT data=drugE NOOBS;
RUN;
```

**Exercise:** Add a line to the above program which creates a variable `either_lower`, which is "Yes" if either the systolic or the diastolic decreased over the 14 days, or "No" otherwise.

We need to use the DO END command if we need to make multiple actions in an IF statement. Consider the following example:

```
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
  INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
  INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
  IF (ssbp_0 > ssbp_14) THEN DO;
    ssbp_lower="Yes"
    ssbp_higher="No"
  END;
  ELSE DO;
    ssbp_lower="No"
    ssbp_higher="Yes"
  END;
```

PROC PRINT data=drugE NOOBS;
RUN;

There is one problem with this program. Look at the output window.

<table>
<thead>
<tr>
<th>number</th>
<th>ssbp_0</th>
<th>ssbp_14</th>
<th>sdbp_0</th>
<th>sdbp_14</th>
<th>ssbp_lower</th>
<th>ssbp_higher</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>136</td>
<td>120</td>
<td>78</td>
<td>85</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>21</td>
<td>120</td>
<td>132</td>
<td>70</td>
<td>80</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>115</td>
<td>96</td>
<td>68</td>
<td>55</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>134</td>
<td>118</td>
<td>91</td>
<td>79</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>16</td>
<td>131</td>
<td>106</td>
<td>93</td>
<td>88</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>19</td>
<td>119</td>
<td>108</td>
<td>71</td>
<td>77</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>124</td>
<td>112</td>
<td>76</td>
<td>72</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>24</td>
<td>127</td>
<td>113</td>
<td>82</td>
<td>57</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Observation two of ssbp_higher is Ye not Yes. This because when we define ssbp_higher at observation one the name was set to a string of two characters. When we get to the second observation SAS truncate Yes to two characters. We use the FORMAT statement to say that the variables ssbp_lower and sdbp_lower are character variable of maximum length 3. Here is the new program:

OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
  INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
  INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
  FORMAT ssbp_lower $3. ssbp_higher $3.;
  IF (ssbp_0 > ssbp_14) THEN DO;
    ssbp_lower="Yes";
    ssbp_higher="No";
  END;
  ELSE DO;
    ssbp_lower="No";
    ssbp_higher="Yes";
  END;
PROC PRINT data=drugE NOOBS;
RUN;
2.5 Subsetting Data

2.5.1 Selecting observations

In the DATA step, we can use the IF (without THEN) command to select out observations which match a certain criteria, or IF THEN DELETE to remove observation which match a certain criteria.

As a first example here is a program to read in the blood pressure data for those patients taking drug E. We will keep only those observations with a baseline systolic blood pressure (ssbp_0) that is greater than or equal to 120.

```sas
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE_ssbp_above_120;
   INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
   INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
   /* keep only those with a systolic BP >= 120 */
   IF (ssbp_0 >= 120);

PROC PRINT data=drugE_ssbp_above_120 NOOBS;
   TITLE "Diabetic blood pressure study";
   TITLE2 "Patients receiving drug E, with systolic BP greater than or equal to 120";
   FOOTNOTE;
RUN;
```

There are six patients in this case (numbers 9, 21, 12, 16, 20 and 24).

Here is a program which omits those with a baseline systolic blood pressure (ssbp_0) that is greater than or equal to 120.

```sas
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE_ssbp_below_120;
   INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
   INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
   /* delete those with a systolic BP >= 120 */
   IF (ssbp_0 >= 120) THEN DELETE;

PROC PRINT data=drugE_ssbp_below_120 NOOBS;
   TITLE "Diabetic blood pressure study";
```

There are two individuals in this case (numbers 8 and 19).

2.5.2 Selecting and removing variables

We use DROP to remove variables from a dataset. For example to remove the diastolic blood pressures from our dataset.

```sas
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
   INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
   INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
   DROP sdbp_0 sdbp_14;

PROC PRINT data=drugE NOOBS;
   TITLE "Diabetic blood pressure study";
   TITLE2 "Patients receiving drug E";
   FOOTNOTE;

RUN;
```

We use KEEP to keep only those variable names named in the KEEP statement. To keep just the number and diastolic blood pressures we would use the following program:

```sas
OPTIONS NODATE; /* Turn off the date in the output window */

DATA drugE;
   INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
   INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
   KEEP number sdbp_0 sdbp_14;

PROC PRINT data=drugE NOOBS;
   TITLE "Diabetic blood pressure study";
   TITLE2 "Patients receiving drug E";
   FOOTNOTE;

RUN;
```
RUN;

2.6 Temporary and Permanent datasets

Here is the contents of the log window when we read in the drugE dataset:

NOTE: 8 records were read from the infile "C:\My SAS Files\drugE.txt".
The minimum record length was 17.
The maximum record length was 17.
NOTE: The data set WORK.DRUGE has 8 observations and 3 variables.
NOTE: DATA statement used:
  real time 0.00 seconds
  cpu time 0.00 seconds

Note how the dataset is called WORK.DRUGE. A WORK dataset is a temporary SAS dataset that is not saved when we leave the program. To make a file permanent we put quotes, " " , around the dataset name. We can also supply a folder where we store the dataset, e.g.,

DATA "C:\My SAS Files\drugE";
  INFILE "C:\My SAS Files\drugE.txt" FIRSTOBS=2;
  INPUT number ssbp_0 ssbp_14 sdbp_0 sdbp_14;
RUN;

Submit the program. Now use the windows explorer to find the folder My Computer → Drive C: → My SAS files. SAS has created a file called "drugE.sas7bdat". We can use this SAS dataset later without worrying about INPUTs.

For example. Delete the DATA statement program above and enter this new program

PROC PRINT data="C:\My SAS Files\drugE";
RUN;

This second program will work even if we exited and restarted SAS.

NOTE: Remember to make a backup of your SAS files that you create.