/* Some useful things you can do in SAS */
/* I haven't posted data files, these are just examples */

/* 1) repeating an analysis on many variables */
/* simple use of the macro facility */

/* proc glm and proc reg allow you to specify many variables to the */
/*   left of the equal sign in the model statement */
/* Gives you output for each variable one at a time */

/* hypothetical example, no data, commented out so not executed */
/* proc glm;
   class trt block;
   model ht wt condition = block trt;
   run;

Will produce three analyses, one for Y=wt, one for Y=ht, and
one for Y=condition */

/* other procs (e.g. proc mixed) only accept one Y variable */
/* here's how to efficiently analyze many variables */

/* define a macro that is a block of code that can be executed */
/* multiple times */
/* the macro can have none, one, or more arguments, i.e. the variables in () */

/* the example data set is a small part of a metabolomic study */
/* treatment = genotype, 10 metabolites */

/* the macro name is the word after "macro" - this one is called analyze */
/* the macro is all the code between the %macro and the %mend; statement */

%macro analyze(Y);
proc mixed data=esters;
   class Compound;
   model &Y = Compound;
   lsmeans Compound / diff;
   title "Analysis of &Y";
   run;
%mend;

/* notice that inside the macro, the argument is referred to as &Y */
/* when SAS sees this, it replaces &Y with the value of Y, then executes */
/* the code */
/* In most cases " " and ' ' are identical: they define character strings */
/* One detail: SAS expands macros inside " ", but not inside ' ' */
/* so this title will replace &Y by the name of the variable being analyzed */
/* title 'Analysis of &Y'; would always print that exact title */

/* Here's how to use this macro: */

data esters;
iinfile 'esters2.csv' dsd firstobs=2;
   input compound $ C16OH C18OH C20OH C22OH C24OH C26OH C28OH C30OH
                    C32OH C34OH;
run;

%analyze(C16OH);
%analyze(C18OH);
%analyze(C20OH);

/* You are not limited to one proc in a macro. Do what ever you want */
/*   to do! */
/* for example, fit the model then plot residuals vs predicted values */

%macro analyze2(Y);
proc mixed data=esters;
  class Compound;
  model &Y = Compound;
  lsmeans Compound / diff;
  title "Analysis of &Y";
  output out=resids r=resid p=yhat;
run;
proc sgplot;
  scatter x=yhat y=resid;
run;
%mend;

%analyze2(C18OH);

/* 2) Match merging */
/* Sometimes your raw data is in multiple files */
/* e.g. a file with carcass weights and a file with some other characteristic(s) */
/* Set does vertical concatenation (save variables, more observations) */
/* merge does horizontal concatenation (more variables, same observations) */

data weight;
  input steer $ wt;
  cards;
a1 105
a2 112
a4 107
;
data rest;
  input steer $ shear;
  cards;
a1 7
a2 8
a3 5
;
proc sort data=weight;
  by steer;
run;
proc sort data=rest;
  by steer;
run;
data both;
  merge weight rest;
  by steer;
run;
proc print;
  title 'merged data sets';
run;

/* Can also do one-many (or many-one) matching */

data feed;
  input steer $ week feed;
cards;
a1 1 50
a1 2 26
a1 3 45
a2 1 47
a2 2 53
a2 3 60
;
proc sort data=feed;
  by steer;
run;

data manyone;
  merge weight feed;
  by steer;
run;

proc print;
  title 'Weight and feed data';
run;

/* 3) Writing a data set to a file */
/* Can create various types of files, identified by their extension */

/* Excel CSV (comma delimited file) */
proc export data=manyone outfile='manyone.csv';
run;

/* Text file with spaces as delimiters */
proc export data=manyone outfile='manyone.txt';
run;

/* Excel spreadsheet (can create both .xls and .xlsx */
proc export data=manyone outfile='manyone.xlsx';
run;

/* 4) Reading a file with tab characters between fields */
/* Excel .prn files are one example of a tab delimited file */
data something;
  infile 'data.prn' dlm='09'x;
  input XX XX XX;
run;

/* The tab character is 09 in hexadecimal (base 16) ASCII code */
/* 'XX'x represents a hexadecimal constant */
/* dlm= specifies the delimiter between fields */

/* 5) Manipulating dates and times */
/* We have seen character and numeric variables */
/* SAS has a huge number of other types of variables */
/* these are described in the documentation about informats */
/* A date format variable is represented internally as the # days from Jan 1, 1966 */
/* A time format variable is represented internally as the # seconds since midnight */
/* This means you can do all sorts of things with dates and times */
/* You tell SAS that a variable is to be considered a date or a time */
/* by describing how to read that variable. This is called an INFORMAT by SAS */
/* INFORMATs (for input) and FORMATs (for printing) have periods at the end */
/* mmddyyW. is the format for dates of the form 11/28/2012 or 11/28/12 */
/* W is the number of characters in the field (10 for 11/28/2012, 8 for 11/28/12) */
/* dateW. is the format for 28 Nov 2012 and all sorts of other ways to represent dates */
/* timeW. is the format for a time: hh:mm or hh:mm:ss or hh:mm:ssAM */
/* the input function allows you to convert a character string (constant or variable) */
/* to a formatted variable */

data dates;
  input day mmddyy8. start time7. end time7.;
  length = end - start;
  julian = day - input('12/31/11', mmddyy8.);
/* julian is day # in the year */
cards;
11/26/12 7:00 7:35
11/27/12 1:15pm 3:35pm
11/28/12 9:00 1:30pm
; run;
/* Notice extra spaces after 'short' dates so there are 7 characters */

proc print;
  title 'Fun with dates and times';
  run;
/* If you want to print something that looks like a date, attach a format to the variable */
/* If the format statement is in the data step, it is permanently attached to the variable */
/* In the proc step, it applies only to that proc */
proc print;
  format day mmddyy8. start time6. end time6. length time5.;
  run;
/* 5) Keyboard shortcuts */
/* Many common menu choices have keyboard shortcuts */
/* The two I use all the time are: */
/* ctrl-e: clear the contents of an output, log, or program window */
/* F3 when program window active: execute the contents of the program window */
/* if code selected, execute the selected code */
Result of merging two data sets

<table>
<thead>
<tr>
<th>Obs</th>
<th>steer</th>
<th>wt</th>
<th>shear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a1</td>
<td>105</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>a2</td>
<td>112</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>a3</td>
<td>.</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>a4</td>
<td>107</td>
<td>.</td>
</tr>
</tbody>
</table>

Result of many-one merge of two data sets

<table>
<thead>
<tr>
<th>Obs</th>
<th>steer</th>
<th>wt</th>
<th>week</th>
<th>feed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a1</td>
<td>105</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>a1</td>
<td>105</td>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>a1</td>
<td>105</td>
<td>3</td>
<td>45</td>
</tr>
<tr>
<td>4</td>
<td>a2</td>
<td>112</td>
<td>1</td>
<td>47</td>
</tr>
<tr>
<td>5</td>
<td>a2</td>
<td>112</td>
<td>2</td>
<td>53</td>
</tr>
<tr>
<td>6</td>
<td>a2</td>
<td>112</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>7</td>
<td>a4</td>
<td>107</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

Result of printing data read as dates and times

<table>
<thead>
<tr>
<th>Obs</th>
<th>day</th>
<th>start</th>
<th>end</th>
<th>length</th>
<th>julianday</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>19323</td>
<td>25200</td>
<td>27300</td>
<td>2100</td>
<td>331</td>
</tr>
<tr>
<td>2</td>
<td>19324</td>
<td>47700</td>
<td>56100</td>
<td>8400</td>
<td>332</td>
</tr>
<tr>
<td>3</td>
<td>19325</td>
<td>32400</td>
<td>48600</td>
<td>16200</td>
<td>333</td>
</tr>
</tbody>
</table>

Result of printing the same data set with date and time formats

<table>
<thead>
<tr>
<th>Obs</th>
<th>day</th>
<th>start</th>
<th>end</th>
<th>length</th>
<th>julianday</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11/26/12</td>
<td>7:00</td>
<td>7:35</td>
<td>0:35</td>
<td>331</td>
</tr>
<tr>
<td>2</td>
<td>11/27/12</td>
<td>13:15</td>
<td>15:35</td>
<td>2:20</td>
<td>332</td>
</tr>
<tr>
<td>3</td>
<td>11/28/12</td>
<td>9:00</td>
<td>13:30</td>
<td>4:30</td>
<td>333</td>
</tr>
</tbody>
</table>