Combining SAS Data Sets
(commands=combine.sas)

There are many ways that SAS data sets can be combined. This handout illustrates combining data sets vertically by adding more cases (stacking or appending data sets) and combining data sets horizontally by adding new variables (merging data sets).

How to Stack Data Sets Vertically (adds new cases):

You can use the `set` statement to combine data sets vertically. It is not necessary for the data sets being combined to have their variables in the same order, or even for them to have the same variables. However, it is critical that if the same variable does appear in both data sets, it should be of the same type (either character or numeric) in both.

If a variable is present in one data set and not in the other, the values for that variable will be missing for all cases for the data set that did not have it. The order of variables in the resulting data set will reflect the order of the first data set listed.

In the example below, the data set BOYS has different variables, which are also in a different order, than the variables in the data set GIRLS. The length of the variable Name in the BOYS dataset will be 8 characters, which is the default, as shown in the output from Proc Contents.

```sas
data boys;
    input name $ sex $ age height teacher $;
    cards;
    Tom  M 12 62 Smith
    Bob  M 13 57 Green
    Joe  M 11 59 Green
    Harry M 12 53 Green
    William M 13 60 Smith
    John M 11 57 Smith
    Richard M 11 55 Green
    ;
    title "Boys Dataset";
    proc contents data=boys varnum;
    run;
```
Boys Dataset

The CONTENTS Procedure

<table>
<thead>
<tr>
<th>Data Set Name</th>
<th>WORK.BOYS</th>
<th>Observations</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Member Type</td>
<td>DATA</td>
<td>Variables</td>
<td>5</td>
</tr>
<tr>
<td>Engine</td>
<td>V9</td>
<td>Indexes</td>
<td>0</td>
</tr>
<tr>
<td>Created</td>
<td>Thu, Sep 08, 2011 12:51:33 PM</td>
<td>Observation Length</td>
<td>40</td>
</tr>
<tr>
<td>Last Modified</td>
<td>Thu, Sep 08, 2011 12:51:33 PM</td>
<td>Deleted Observations</td>
<td>0</td>
</tr>
<tr>
<td>Protection</td>
<td></td>
<td>Compressed</td>
<td>NO</td>
</tr>
<tr>
<td>Data Set Type</td>
<td>WINDOWS_32</td>
<td>Sorted</td>
<td>NO</td>
</tr>
<tr>
<td>Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Representation</td>
<td>WINDOWS_32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Encoding</td>
<td>wlatin1  Western (Windows)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Variables in Creation Order

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>name</td>
<td>Char</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>sex</td>
<td>Char</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>age</td>
<td>Num</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>height</td>
<td>Num</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>teacher</td>
<td>Char</td>
<td>8</td>
</tr>
</tbody>
</table>

Using a Length Statement:

We use a Length statement for the variable Name in the GIRLS dataset, so that we can accommodate the longest name that occurs. Note that you can use any length (Up to 32767 characters) that you want to specify for a character variable; just be sure that it is long enough to accommodate the longest value that will occur within the variable. Note that the Length statement needs to follow immediately after the data statement.

data girls;
  length name $ 10;
  input name $ age sex $ teacher $;
cards;
Sharice  13 F  Smith
Mary     12 F  Smith
Ellen    11 F  Green
Carol    11 F  Green
Chris    13 F  Smith
Claire   12 F  Green
Raye     13 F  Smith
Wilhelmina 12 F Green
;
title "Girls Data";
proc contents data=girls varnum;
run;
Stacking the data using SET (BOYS First):

We now use a Set statement to stack the two datasets. This will make the boys dataset be first in our output dataset, followed by the GIRLS dataset. The characteristics of the variables will be taken from the BOYS data. This means that the length of Name will be too short. To avoid this, we use a Length statement when stacking the two datasets. Again, be sure the Length statement follows the data statement, immediately.

data allkids;
   length $ name 10;
   set boys girls;
run;

title "printout of allkids dataset";
title2 "with boys first in the data set";
proc print data = allkids;
run;

<table>
<thead>
<tr>
<th>Obs</th>
<th>name</th>
<th>sex</th>
<th>age</th>
<th>height</th>
<th>teacher</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tom</td>
<td>M</td>
<td>12</td>
<td>62</td>
<td>Smith</td>
</tr>
<tr>
<td>2</td>
<td>Bob</td>
<td>M</td>
<td>13</td>
<td>57</td>
<td>Green</td>
</tr>
<tr>
<td>3</td>
<td>Joe</td>
<td>M</td>
<td>11</td>
<td>59</td>
<td>Green</td>
</tr>
<tr>
<td>4</td>
<td>Harry</td>
<td>M</td>
<td>12</td>
<td>53</td>
<td>Green</td>
</tr>
<tr>
<td>5</td>
<td>William</td>
<td>M</td>
<td>13</td>
<td>60</td>
<td>Smith</td>
</tr>
</tbody>
</table>
Stacking the data using SET (GIRLS First):

We again use a Set statement to combine the two data sets, but with the GIRLS dataset being first. In this case, we don't need to use a length statement, because the GIRLS dataset has a length of 10 for the variable Name, so SAS will pick up the characteristics of the variable Name from the GIRLS dataset. However, it is probably safer to specify a length here, too, so that the lengths of the character variables will match when combining the data.

data allkids2;
  set girls boys;
run;

title "printout of allkids2 data set";
title2 "with girls first in the data set";
proc print data = allkids2;
run;

<table>
<thead>
<tr>
<th>Obs</th>
<th>name</th>
<th>age</th>
<th>sex</th>
<th>teacher</th>
<th>height</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sharice</td>
<td>13</td>
<td>F</td>
<td>Smith</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>Mary</td>
<td>12</td>
<td>F</td>
<td>Smith</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>Ellen</td>
<td>11</td>
<td>F</td>
<td>Green</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>Carol</td>
<td>11</td>
<td>F</td>
<td>Green</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>Chris</td>
<td>13</td>
<td>F</td>
<td>Smith</td>
<td>.</td>
</tr>
<tr>
<td>6</td>
<td>Claire</td>
<td>12</td>
<td>F</td>
<td>Green</td>
<td>.</td>
</tr>
<tr>
<td>7</td>
<td>Raye</td>
<td>13</td>
<td>F</td>
<td>Smith</td>
<td>.</td>
</tr>
<tr>
<td>8</td>
<td>Wilhelmina</td>
<td>12</td>
<td>F</td>
<td>Green</td>
<td>.</td>
</tr>
<tr>
<td>9</td>
<td>Tom</td>
<td>12</td>
<td>M</td>
<td>Smith</td>
<td>62</td>
</tr>
<tr>
<td>10</td>
<td>Bob</td>
<td>13</td>
<td>M</td>
<td>Green</td>
<td>57</td>
</tr>
<tr>
<td>11</td>
<td>Joe</td>
<td>11</td>
<td>M</td>
<td>Green</td>
<td>59</td>
</tr>
<tr>
<td>12</td>
<td>Harry</td>
<td>12</td>
<td>M</td>
<td>Green</td>
<td>53</td>
</tr>
<tr>
<td>13</td>
<td>William</td>
<td>13</td>
<td>M</td>
<td>Smith</td>
<td>60</td>
</tr>
<tr>
<td>14</td>
<td>John</td>
<td>11</td>
<td>M</td>
<td>Smith</td>
<td>57</td>
</tr>
<tr>
<td>15</td>
<td>Richard</td>
<td>11</td>
<td>M</td>
<td>Green</td>
<td>55</td>
</tr>
</tbody>
</table>

Notice that the order of the variables in the final data set is changed, depending on which data set was listed first in the set statement, but the values in both data sets are the same.
Merge Data Sets Horizontally (adds new variables):

SAS data sets can be merged horizontally in a number of ways. This method of combining data sets allows you to match based on some key variable(s) such as ID or household. You can merge based on the values of one or more variables. Note that if you merge, based on the value of a character variable, be sure that you specify a LENGTH statement as the first statement in your data step, so that the lengths of the variables in both datasets can be accommodated. **You must first sort the data sets that are being merged by the key variable(s), and then merge by the same key variable(s).**

The example below shows how to merge two data sets for the same people. The dataset, EXAM contains data for a hypothetical group of people on a physical exam. The data set LAB contains information for the *some of the same people* on their laboratory results.

```plaintext
data exam;
    input id examdate mmddyy10. sex age height weight sbp dbp;
    format examdate mmddyy10.;
    cards;
    1 10/18/2000 1 25 72 156 128 89
    2 05/29/2000 1 33 68 168 145 96
    3 02/21/2000 1 47 65 182 152 98
    4 06/17/2000 1 29 69 190 139 91
    5 01/11/2000 2 37 62 129 145 93
    6 08/15/2000 2 42 64 156 133 94
;

data lab;
    input id hgb;
    cards;
    1  13.2
    4  12.1
    3  14.5
    6  12.8
    12 13.0
;
proc sort data=exam;
    by id;
run;
proc sort data=lab;
    by id;
run;

data exam_lab;
    merge exam lab;
    by id;
run;
```
title "Printout of Exam_lab Data Set";
proc print;
run;

By default, SAS will include all observations from both data sets in the merged data. Notice in the above example, ID numbers 2 and 5 are in the EXAM data set, but not in the lab data set, while ID number 12 is in the LAB data set, but not in the EXAM data set. However all of these cases are in the merged EXAM_LAB data set.

You can control the observations that get written to the final data set, using the in= data set option. This creates a temporary variable that indicates whether a case is in a particular data set or not. Then you can control which observations get written out, using subsetting if statements. The examples below show three different ways this could be done.

/*How to include only cases that are in both data sets*/

data exam_lab2;
   merge exam(in=a) lab(in=b);
      by id;
         if a and b;
run;

title "Exam_lab2 Data Set Includes Only Those";
title2 "In Both Data Sets";
proc print data=exam_lab2;
run;

/*How to include cases that are in EXAM, regardless of Lab Data*/
data exam_lab3;
  merge exam(in=a) lab(in=b);
  by id;
  if a;
run;

title "Exam_lab3 Data Set Includes Those"
title2 "In Exam Data, Regardless of Lab Data"
proc print data=exam_lab3;
run;

Exam_lab3 Data Set Includes Those
In Exam Data, Regardless of Lab Data

Obs  id  examdate  sex  age  height  weight  sbp  dbp  hgb
 1   1  10/18/2000  1   25    72      156    128    89  13.2
 2   2  10/18/2000  1   33    68      168    145    96   .
 3   3  02/21/2000  1   47    65      182    152    98  14.5
 4   4  06/17/2000  1   29    69      190    139    91  12.1
 5   5  01/11/2000  2   37    62      129    145    93   .
 6   6  08/15/2000  2   42    64      156    133    94  12.8

/*How to include cases that are in LAB, regardless of Exam Data*/
data exam_lab4;
  merge exam(in=a) lab(in=b);
  by id;
  if b;
run;

title "Exam_lab4 Data Set Includes Those"
title2 "In Lab Data, Regardless of Exam Data"
proc print data=exam_lab4;
run;

Exam_lab4 Data Set Includes Those
In Lab Data, Regardless of Exam Data

Obs  id  examdate  sex  age  height  weight  sbp  dbp  hgb
 1   1  10/18/2000  1   25    72      156    128    89  13.2
 2   3  02/21/2000  1   47    65      182    152    98  14.5
 3   4  06/17/2000  1   29    69      190    139    91  12.1
 4   6  08/15/2000  2   42    64      156    133    94  12.8
 5  12  .   .   .     .       .      .     .    13.0
How to merge data sets when the variable names are the same:

When you merge two datasets, SAS requires that the names be different (except for the variables used as Key variables in the merge). If the names are not different in the two datasets, you will overwrite the values of the variables in the first dataset with the values in the second dataset. If the two data sets that you wish to merge have the same variable names, this can be handled by using the rename dataset option for either one or both of the datasets.

data oldsal;
  input name $ idnum sex $ age salary jobcat year;
  cards;
  Roger  518 M 45 7677 2 1989
  Martha 321 F 28 5000 1 1989
  Zeke   444 M 33 6075 1 1989
  Barb 1728 F 40 9023 2 1989
  Bill  993 M 36 7739 3 1989
  Sandy 1002 F 29 6161 3 1989
;    
data newsal;
  input name $ idnum salary jobcat year;
  cards;
  Hank    108 11138 1 1995
  Fred    519 10035 2 1995
  Zeke    444  9697 1 1995
  Martha  321  7987 2 1995
  Sandy  1002  6995 2 1995
  Bill    993 12400 3 1995
  Roxy    773 10119 2 1995
;    
/*merging by idnum*/
proc sort data=oldsal;
  by idnum;
run;

proc sort data=newsal;
  by idnum;
run;

data combine1;
  merge oldsal(rename=(salary=salary89 jobcat=jobcat89))
   newsal(rename=(salary=salary95 jobcat=jobcat95));
  by idnum;
  drop year;
run;
title "printout of combine1 data set";
title2 "matching by id number";
title3 "all cases that were in either data set are included";
proc print data=combine1;
run;

proc print data=combine1;
run;

You can control the observations that are written to the final data set, using in= data set options for this type of merge also.

/*merging by idnum, but keeping only cases that are in both datasets*/

data combine2;
   merge oldsal(in=a rename=(salary=salary89 jobcat=jobcat89))
      newsal(in=b rename=(salary=salary95 jobcat=jobcat95));
   by idnum;
   if a and b;
   totsal = sum (salary89,salary95);
   format salary89 salary95 totsal dollar12.;
   drop year;
run;
proc print data=combine2;
   title "printout of combine2 data set";
   title2 "matching by id number";
   title3 "and only including cases that are in both data sets";
run;
printout of combine2 data set
matching by id number
and only including cases that are in both data sets

<table>
<thead>
<tr>
<th>Obs</th>
<th>name</th>
<th>idnum</th>
<th>sex</th>
<th>age</th>
<th>salary</th>
<th>jobcat</th>
<th>salary95</th>
<th>jobcat95</th>
<th>totsal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Martha</td>
<td>321</td>
<td>F</td>
<td>28</td>
<td>5000</td>
<td>1</td>
<td>7987</td>
<td>2</td>
<td>$12,987</td>
</tr>
<tr>
<td>2</td>
<td>Zeke</td>
<td>444</td>
<td>M</td>
<td>33</td>
<td>6075</td>
<td>1</td>
<td>9697</td>
<td>1</td>
<td>$15,772</td>
</tr>
<tr>
<td>3</td>
<td>Bill</td>
<td>993</td>
<td>M</td>
<td>36</td>
<td>7739</td>
<td>3</td>
<td>12400</td>
<td>3</td>
<td>$20,130</td>
</tr>
<tr>
<td>4</td>
<td>Sandy</td>
<td>1002</td>
<td>F</td>
<td>29</td>
<td>6161</td>
<td>3</td>
<td>6995</td>
<td>2</td>
<td>$13,156</td>
</tr>
</tbody>
</table>

Include Cases from EXAM regardless of whether they are in LAB or not:

If you want to include only cases from the EXAM dataset, regardless of whether they appear in the LAB dataset or not, you can use the following syntax.

/*INCLUDE CASES THAT ARE IN EXAM, WHETHER THEY ARE IN LAB OR NOT*/

data exam_lab3;
  merge exam(in=a) lab(in=b);
  by id;
  if a;
run;

title "Exam_lab3 Data Set Includes Those";
title2 "In Exam Data, Regardless of Lab Data";
proc print data=exam_lab3;
run;

Note that in the output below, two cases are included for people who were in the EXAM dataset, but had no lab values.
Merging Data from a Table with a Dataset having multiple obs per group

Having the ability to specify which dataset(s) provide cases when merging allows a lot of freedom when combining SAS datasets. For example, combining information on patients who live in several counties with information from the county can be easily accomplished, as shown in the example below.

First, we import the AgeStudy data, and check it using Proc Print:

```sas
PROC IMPORT OUT= WORK.AgeStudy
    DATAFILE= "AgeStudy.xls"
    DBMS=EXCEL REPLACE;
    RANGE="Sheet1$";
    GETNAMES=YES;
    MIXED=NO;
    SCANTEXT=YES;
    USEDATE=YES;
    SCANTIME=YES;
RUN;
title "AgeStudy Data";
proc contents data=agestudy;
run;
proc print data=agestudy;
run;
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>County</th>
<th>Name</th>
<th>Gender</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Washtenaw</td>
<td>Jim</td>
<td>m</td>
<td>54</td>
</tr>
<tr>
<td>2</td>
<td>Washtenaw</td>
<td>Bob</td>
<td>m</td>
<td>49</td>
</tr>
<tr>
<td>3</td>
<td>Washtenaw</td>
<td>Susan</td>
<td>f</td>
<td>29</td>
</tr>
<tr>
<td>4</td>
<td>Wayne</td>
<td>Sally</td>
<td>f</td>
<td>35</td>
</tr>
<tr>
<td>5</td>
<td>Muskegon</td>
<td>Robert</td>
<td>m</td>
<td>53</td>
</tr>
<tr>
<td>6</td>
<td>Osceola</td>
<td>Jill</td>
<td>f</td>
<td>59</td>
</tr>
<tr>
<td>7</td>
<td>Newaygo</td>
<td>Phil</td>
<td>m</td>
<td>37</td>
</tr>
<tr>
<td>8</td>
<td>Wayne</td>
<td>Roger</td>
<td>m</td>
<td>56</td>
</tr>
<tr>
<td>9</td>
<td>Washtenaw</td>
<td>Gipper</td>
<td>m</td>
<td>74</td>
</tr>
</tbody>
</table>

Now, we import the Census data, and take a brief look at it.

```sas
PROC IMPORT OUT= WORK.Census
    DATAFILE= "MI_Census_2000.xls"
    DBMS=EXCEL REPLACE;
    RANGE="Sheet1$";
    GETNAMES=YES;
    MIXED=NO;
    SCANTEXT=YES;
    USEDATE=YES;
    SCANTIME=YES;
RUN;
```
title "Census Data";
proc contents data=census;
run;
proc print data=census(obs=12);
run;

Census Data

<table>
<thead>
<tr>
<th>Obs</th>
<th>County</th>
<th>2000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alcona</td>
<td>11719</td>
</tr>
<tr>
<td>2</td>
<td>Alger</td>
<td>9862</td>
</tr>
<tr>
<td>3</td>
<td>Allegan</td>
<td>105665</td>
</tr>
<tr>
<td>4</td>
<td>Alpena</td>
<td>31314</td>
</tr>
<tr>
<td>5</td>
<td>Antrim</td>
<td>23110</td>
</tr>
<tr>
<td>6</td>
<td>Arenac</td>
<td>17269</td>
</tr>
<tr>
<td>7</td>
<td>Baraga</td>
<td>8746</td>
</tr>
<tr>
<td>8</td>
<td>Barry</td>
<td>56755</td>
</tr>
<tr>
<td>9</td>
<td>Bay</td>
<td>110157</td>
</tr>
<tr>
<td>10</td>
<td>Benzie</td>
<td>15998</td>
</tr>
<tr>
<td>11</td>
<td>Berrien</td>
<td>162453</td>
</tr>
<tr>
<td>12</td>
<td>Branch</td>
<td>45787</td>
</tr>
</tbody>
</table>

Now, merge the two datasets, but keep only cases that occur in the Age Study Dataset. Note that each case in Washtenaw county will be matched with the data for Washtenaw County in the Census datasets. Also note that we used a Length statement to be sure the variable County would have sufficient length to accommodate all county names. The IF statement restricts the merged dataset, so that it contains only observations that were originally in the AGESTUDY dataset. Counties that occur in the Census data, but not in the Agestudy data will not be included.

proc sort data=census;
by county;
run;
proc sort data=agestudy;
by county;
run;
data study_census;
  length county $ 16;
  merge agestudy(in=instudy) census(in=incensus);
  by county;
    if instudy;
run;

title "Printout of Merged Data";
proc print data=study_census;
format Census_2000 comma12.;
run;
Note that every observation in Washtenaw County gets the Census for Washtenaw County. The same is true for the cases in Wayne County. In general, SAS will always fill in the values for matching Key variables, regardless of the dataset in which the duplicates occur. However, SAS does not allow duplicates of the same values of key variables to occur in both datasets.

<table>
<thead>
<tr>
<th>Obs</th>
<th>county</th>
<th>Name</th>
<th>Gender</th>
<th>Age</th>
<th>Census_2000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Muskegon</td>
<td>Robert</td>
<td>m</td>
<td>53</td>
<td>170,200</td>
</tr>
<tr>
<td>2</td>
<td>Newaygo</td>
<td>Phil</td>
<td>m</td>
<td>37</td>
<td>47,874</td>
</tr>
<tr>
<td>3</td>
<td>Osceola</td>
<td>Jill</td>
<td>f</td>
<td>59</td>
<td>23,197</td>
</tr>
<tr>
<td>4</td>
<td>Washtenaw</td>
<td>Jim</td>
<td>m</td>
<td>54</td>
<td>322,895</td>
</tr>
<tr>
<td>5</td>
<td>Washtenaw</td>
<td>Bob</td>
<td>m</td>
<td>49</td>
<td>322,895</td>
</tr>
<tr>
<td>6</td>
<td>Washtenaw</td>
<td>Susan</td>
<td>f</td>
<td>29</td>
<td>322,895</td>
</tr>
<tr>
<td>7</td>
<td>Washtenaw</td>
<td>Gipper</td>
<td>m</td>
<td>74</td>
<td>322,895</td>
</tr>
<tr>
<td>8</td>
<td>Wayne</td>
<td>Sally</td>
<td>f</td>
<td>35</td>
<td>2,061,162</td>
</tr>
<tr>
<td>9</td>
<td>Wayne</td>
<td>Roger</td>
<td>m</td>
<td>56</td>
<td>2,061,162</td>
</tr>
</tbody>
</table>