SAS Basics Workshop #1

Creating Variables, pt 2
Presented by Paul Fisher
November 6, 2006

Data

Go to the URL

http://www.econ.wayne.edu/agoodman/fisher_sas/

And download the Excel file under Workshop #3. (You will have to right click on the Excel icon next to the words "Example Data Set"

Turn this Excel file into a tab delimited text file and then import it into SAS.

Syntax

When creating variables, you might need to know how to express the following concepts:

- Equal to: =
- Not equal to: ^=
- Less than: <
- LTOET: <=
- Missing value: .

Eliminating Observations

Sometimes you need to eliminate observations for not meeting certain criteria. We use the word “delete” to do this.

```
DATA datafile;
SET datafile;
IF condition THEN delete;
RUN;
```
Eliminating Observations

Look at our data. Suppose we only want to include people whose income is over $3,000.

```sas
DATA work1;
SET work1;
IF INCOME<3000 THEN delete;
RUN;
```

Suppose we can only use observations if we know the number of hours worked.

```sas
DATA work1;
SET work1;
IF HOURS=. THEN delete;
RUN;
```

Using Arrays

Arrays are sequences. Sometimes it’s easier to refer to a whole sequence of variables than it is to refer to each individual variable.

First, you have to define the array:

```sas
ARRAY aname{n} VAR1 VAR2;
```

Examples:

```sas
array INCOME{5} INC90 INC91 INC92 INC93 INC94;
array INCOME{5} INC90-INC94;
array INCOME{5} INC90 Y91 INCOME92 YD93 INC94;
```

Using Arrays

In our data set, we have information about expenditures on things that are bad for you: beer, cigarettes, marijuana, fast food. Suppose we want to take the log of each of these.

So, we have to define an array for the variables we’re starting with, but ALSO for the variables we’ll end up with:

```sas
array EXP{4} beer cigs mari fastf;
array LNEXP{4} lnbeer lncigs lnmari lnfastf;
```

Using Arrays

Array commands have to include a “do” command, which tells SAS where in the array to start and how far to go and an “end” command which tells SAS to stop using the arrays.

Typically, we want to start at the beginning and go to the end:

```sas
array ANAME{n} VAR1 VAR2 ... VARn;
array NEW{n} NEW1 NEW2 ... NEWn;
do i=1 to n;
  (commands using the arrays)
end;
```
Using Arrays

Since we want to calculate the log of the expenditures, our whole data step would look like this:

```plaintext
data work1;
set work1;
array EXP{4} beer cigs mari fastf;
array LNEXP{4} lnbeer lncigs lnmari lnfastf;
do i=1 to n;
    LNEXP{i}=log(EXP{i});
end;
run;
```

Dropping Variables

We might not want to keep old variables after we change them. If we want to keep our data set from getting crowded with variables we don’t need anymore, we should drop some as we go:

```plaintext
data datafile(drop=var1 var2 var3);
set datafile;
run;
```

In our case, it would look like this:

```plaintext
data work1(drop=beer cigs mari fastf);
set work1;
run;
```

Complicated Example

Notice how in our data set, some members of the same household (see HHID) disagree on the highest level of education attained by the head of household (HOHEDU)?

One solution to this might be to take the average reported HOHEDU for each household. But how to do this?

Let’s work through this example one step at a time.

```plaintext
First, we have to get rid of the observations that did not report a HOHEDU (even if their siblings did). Now, we don’t want to delete them for good, but only while we work on this problem, so we should create a new datafile.

data work2;
set work1;
if HOHEDU=. then delete;
run;
```
Complicated Example

Now, as SAS moves down through the data, we want SAS to know when a new household starts, so we have to sort the data and use a "first" and "last" command:

This is how you sort the data:

```
proc sort data=work2;
  by hhid;
run;
```

This is how you identify first and last:

```
data work2;
  set work2;
  by hhid;
  if first.hhid then first=1;
  else first=0;
  if last.hhid then last=1;
  else last=0;
run;
```

Look at the data to see what this will do.

Complicated Example

We will need to know how many people are in each household, so we need SAS to count the number of people in the household as it moves through the data:

```
data work2;
  set work2;
  retain count;
  by hhid;
  if first.hhid then
    do;
    count=0;
    end;
  count = count + 1;
run;
```

Complicated Example

In order to eventually get a mean, we need a sum. So we want SAS to add up the HOHEDU values for each household as it moves through the data. This makes a cumulative sum:

```
data work2;
  set work2;
  retain sumhohed;
  by hhid;
  if first.hhid then
    do;
    sumhohed=0;
    end;
  sumhohed = sumhohed + hohedu;
run;
```
Complicated Example

Now we can calculate a cumulative mean:

```sas
data work2;
set work2;
   avgHOHED = SUMHOHED/count;
run;
```

Then we output JUST the mean for the last members of each household:

```sas
data work3;
set work2;
by hhid;
if last.hhid then output;
run;
```

Complicated Example

Then we drop the other variables from our new data set, leaving just avgHOHED and hhid:

```sas
data work3(drop=PID AGE EDU HOHEDU RACE INCOME HOURS lnBEER lnCIGS lnMARI lnFASTFOOD first last sumhohed count);
set work3;
run;
```

Complicated Example

Now, we can just merge our last datafile with our original one, (by HHID) and it will add our avgHOHed to every observation.

```sas
data work4;
   merge work1 work3;
   by hhid;
run;
```

Conclusion

If you want to use the “first” or “last” commands, you must also use the “by” statement.

As SAS moves downward through the data, the “retain” statement tells it to remember a value from one observation when it moves on to the next.

You’ve seen many, many examples of data manipulation, but we’ve just scratched the surface. Your own work will require you to find ways to apply these commands (and others) in creative ways to do whatever it is you’ll need to do.