

Introduction to SAS

Anita Rocha

Center for Studies in Demography and Ecology

University of Washington

Draft 10/14/2008

Table of contents

| | |
|--|----|
| Introduction..... | 1 |
| Overview..... | 2 |
| Part I..... | 2 |
| Basic Screen Navigation..... | 2 |
| SAS default features..... | 2 |
| PROCs and DATA step..... | 3 |
| Comments..... | 4 |
| Importing and exporting datasets..... | 4 |
| View data..... | 5 |
| Creating and saving datasets..... | 6 |
| Permanent SAS dataset (create a library with LIBNAME)..... | 7 |
| View the “meta-data” of your dataset..... | 8 |
| Variable labels..... | 8 |
| Basic DATA STEP processing..... | 9 |
| More DATA STEP and the creation of variables..... | 9 |
| IF-THEN-ELSE..... | 10 |
| Part II..... | 11 |
| Formats..... | 14 |
| Merging Datasets..... | 15 |
| Descriptive statistics procedures..... | 18 |
| Introduction to regression procedures..... | 20 |

Introduction

This is a course on how to begin using the statistical programming package SAS®, specifically SAS version 9.0, 9.1, 9.2 or 9.3 developed by SAS Institute (www.sas.com). This course assumes no previous experience with SAS.

SAS runs on Windows and UNIX platforms and can be used for data management, statistical analysis, and presentation of results.

First, here are some useful resources.

| |
|---|
| On-line tutorial at the UCLA site: http://www.ats.ucla.edu/stat/sas/notes2/default.htm |
|---|

| | |
|---|---|
| SAS On-line documentation as reference: | http://support.sas.com/onlinedoc/912/docMainpage.jsp |
| <i>The Little SAS Book: A Primer</i> | http://www.sas.com/apps/pubscat/bookdetails.jsp?catid=1&pc=59216 |

Overview

SAS is big, so big that it's hard to be familiar with every aspect. We will look at just a small part, the part that can be useful to us as academic researchers. But there's a whole lot more.

Why use SAS?

- 1) Strong data management capabilities. It can handle large datasets elegantly. It can handle multiple datasets simultaneously.
- 2) Wide variety of statistical procedures.
- 3) Great technical support.
- 4) Your advisor/department/company/industry tells you that SAS is the in-house standard.

Possible weaknesses

- 1) Start-up learning curve.
- 2) Comprehensive documentation can be hard to sort through.
- 3) Some think the syntax is cumbersome (not me!).

There are other packages that have strengths and weaknesses compared to SAS.

Part I

Basic Screen Navigation

Main:

- 1) Editor – create SAS code
- 2) Log – find out information after you've run SAS code
- 3) Output – results of the procedures in your SAS code

Side:

- 4) Explorer – navigate to other objects like libraries
- 5) Results – navigate your Output window

SAS default features

You can customize your work environment in SAS.

Q: How do I change my window layout?

A: See the drop-down menu WINDOW.

Exercise 1:

Go the WINDOW menu and click on Cascade, Tile Vertically, and Tile Horizontally, respectively. To recover original layout, click on Resize.

PROCs and DATA step

We are about to start using the two big workhorses of SAS, the DATA step and the procedures (PROCs).

DATA (step): It creates, saves, changes data

PROC (i.e. procedure): It does something *with* the data like analysis or descriptive statistics.

This first example will begin with a procedure.

Q: What is the basic contents and licensing information for the SAS on this computer?

A: Use PROC SETINIT to see SAS modules and license expiration.

Exercise 2:

Type into the Editor window

```
*Exercise 2;  
proc setinit;  
run;
```

Click on RUN (running man). See the LOG. Notice that information about the procedure is in blue and other information is in black. Later, you'll see warning messages in green and error messages in red. Those colors can be altered.

When does the SAS license on this computer expire?

Why does `*Exercise 2;` show up green in the Editor window?

Save the contents of the Editor window.

- **File**
- **Save As ...**
- Navigate to the appropriate folder and name the file something like **Beg_SAS_class**

Comments

There are two ways to make a comment in SAS code?

1) ** comment* ; as in

```
*Exercise 3;
```

2) */* comment */* as in

```
/* Exercise 3 */
```

Exercise 3:

Try typing both the comments above on separate lines in the Editor window. What happens to its color if a part of the comment is missing (like delete a “*” or “/*”)?

Importing and exporting datasets

Before we begin importing data into SAS, let’s get familiar with the data we will be using for examples and exercises in the first part of this workshop.

Example Data: sub-sample of MORG

The data used for this workshop comes from the Merged Outgoing Rotation Groups (MORG), otherwise known as the U.S. Census Bureau’s Current Population Surveys (CPS) Annual Earnings File compiled by the National Bureau of Economic Research (NBER). Fifty or more variables each month are selected for continuity across years. There are 25,000 records or more per month. Here we’re using just a sub-sample of the full 2005 dataset for illustrative purposes. For more information and to gain access to more MORG data, go to the CSDE website (<http://csde.washington.edu/>) or email Anita Rocha (alrocha@u.washington.edu).

Sub-sample contents:

| | |
|---------|---|
| AGE | Age |
| EARNHRE | Earnings per hour in pennies |
| EARNWKE | Earnings per week |
| HHID | Household ID; unique for each record in this sub-sample |
| HOURSLW | Number of hours worked last week |
| IHIGRDC | Imputed highest grade completed |
| LNAME | Last name |
| MARITAL | Marital status |
| | 1 = "Married, Civilian Spouse Present" |
| | 2 = "Married, Armed Forces Spouse Present" |
| | 3 = "Married, Spouse Absent (exc. Separated)" |
| | 4 = "Widowed" |
| | 5 = "Divorced" |
| | 6 = "Separated" |

| | |
|----------|--|
| OWNCHILD | 7 = "Never Married" |
| SEX | Number of biologic children < 18 years old |
| | Sex |
| | 1 = "Male" |
| | 2 = "Female" |
| YRCOLL | Years of college credit |

We will start with importing an Excel file into SAS first through the SAS Import Wizard. The variable names are on the first line of the Excel file. Here are the steps.

- **File**
- **Import Data**
- Choose Excel .xls format (this is actually the default)
- **Click on Next**
- **Click on Browse to select a file: c:\temp\sas\morg05s1.xls**
- The default option is to read variable names from the first line, leave as it is.
- **Click on Next**
- Enter a name (**xls_sample1**) for the data set
- **Click on Finish**

Exercise 4:

Open **morg05s1.xls** as an Excel spreadsheet.

Follow the steps above to import **morg05s1.xls** into SAS.

View your LOG. Does it say that it was successfully created?

What are some of the different types of files SAS can import via the SAS Import Wizard?

View data

There are at least two ways to view your data in SAS.

- 1) In the Explorer window, double-click on Libraries, double-click on Work, and then double-click on the dataset icon. A spreadsheet-type view should open.

Exercise 5:

Follow the steps above for your **xls_sample1** dataset.

Does it look like what is in the original Excel spreadsheet?

Exercise 5b:

Under the EDIT pull-down menu, the default is Browse. Now select Edit to change the data.

2) Use PROC PRINT to view your data.

Exercise 6:

In the Editor window, type

```
* Exercise 6: display MORG05 data;  
proc print data=xls_sample1;  
run;
```

Review the LOG window?

Review the OUTPUT window?

What happened to the Results window? Click back to the Editor window. Can you navigate to your PROC PRINT output via the Results window?

Next, we'll import a comma-delimited ASCII file.

Exercise 7:

Open **morg05s1.csv** in Notepad or Textpad. See this is a comma-delimited file.

In the Editor window, type

```
* Exercise 7: Import morg05s1 data from comma-delimited file;  
* Use the LENGTH statement to ensure that character variable LNAME and  
  HHID has enough spaces to house the entire values;  
data csv_sample1;  
  length LNAME $ 30 HHID $ 15;  
  infile 'c:\temp\csde_sas\morg05s1.csv'  
    delimiter=',' dsd;  
  input LNAME $ SEX MARITAL HHID $ AGE HOURSLOW EARNHRE EARNWKE OWNCHILD  
    YRCOLL IHIGRDC;  
run;  
  
proc print data = csv_sample1;  
run;
```

Run this by highlighting just the portion you want to run and then click RUN.

Note: If you want SAS to read consecutive delimiters (like “,,”) as though there is a missing value between them, specify the DSD option in the INFILE statement.

How many records are in this **csv_sample1** dataset according to the LOG?

Can you find **csv_sample1** dataset in the Work library via the Explorer window?

Creating and saving datasets

You can directly input data into SAS to create a new dataset.

Exercise 8:

In the Editor window, type

```
* Exercise 8: Input data directly into a SAS dataset;
```

```
data input_sample1;
  input hhid $ sex marital age hourslw;
  datalines;
02062 2 7 39 40
05382 2 5 51 54
08955 1 1 24 50
;
run;

proc print data=input_sample1;
run;
```

Run this.

Now, suppose we don't need this dataset. Navigate to this dataset via the Explorer window in the Work library. Right click and then Delete (optional).

Permanent SAS dataset (create a library with LIBNAME)

The Work library is temporary. When SAS is closed, all the datasets in the Work library are deleted.

Q: What if you want to save a dataset to continue to work with it later?

A: Create a permanent SAS dataset via a library.

A library can be “created” (not really created but identified) with the following code.

```
libname user-chosen-name 'user-chosen-location';
```

The way you identify a dataset in a library is by writing

```
libname.dataset_name
```

Exercise 9:

Create a new library.

In the Editor window, type

```
* Exercise 9: create a permanent SAS dataset;
libname mylib 'c:\temp\csde_sas';
```

Click RUN.

Look at the Log.

Look at the Explorer window. Do you see the new library?

Now, in the Editor window, type

```
data mylib.permanent;
  set csv_sample1;
run;
```

Run this.

Note: the SET statement just “copies” the content of one dataset to another dataset. What happened in the Explorer window, particularly in the new library?

View the “meta-data” of your dataset

You already know of two ways to view the values in your dataset.

Q: What are they?

A: 1) Explorer window navigation to the dataset to open a spreadsheet-type view, and 2) PROC PRINT.

What if you want to know what type of data is in your dataset? In other words, you want to look at the meta-data, the data about data?

Use PROC CONTENTS.

Exercise 10:

Type in and run the following

```
* Exercise 10: show the contents of a dataset;  
proc contents data=mylib.permanent;  
run;
```

Review the Results window.

How many observations are in the dataset PERMANENT?

How many variables?

What type of variable is AGE?

What type of variable is HHID?

Variable labels

You can add other information to the dataset besides just new records. For example, you can provide labels for your variables.

In the DATA STEP, you can assign a label to a variable with the following SAS code.

```
data new_dataset;  
  set old_dataset;  
  label variable = 'User-written-label-information';  
run;
```

Exercise 11:

Re-run the DATA STEP in Exercise 9, only add a label to the variable EARNWKE (it’s the ‘Earnings per week’). Be mindful of your semicolons.

Then re-run the PROC CONTENTS from Exercise 10.

Next, run this

```
proc print data=mylib.permanent label;  
run;
```

Note: the added option LABEL in PROC PRINT tells SAS to display any available labels in place of variable names in the output.

Exercise 11b optional:

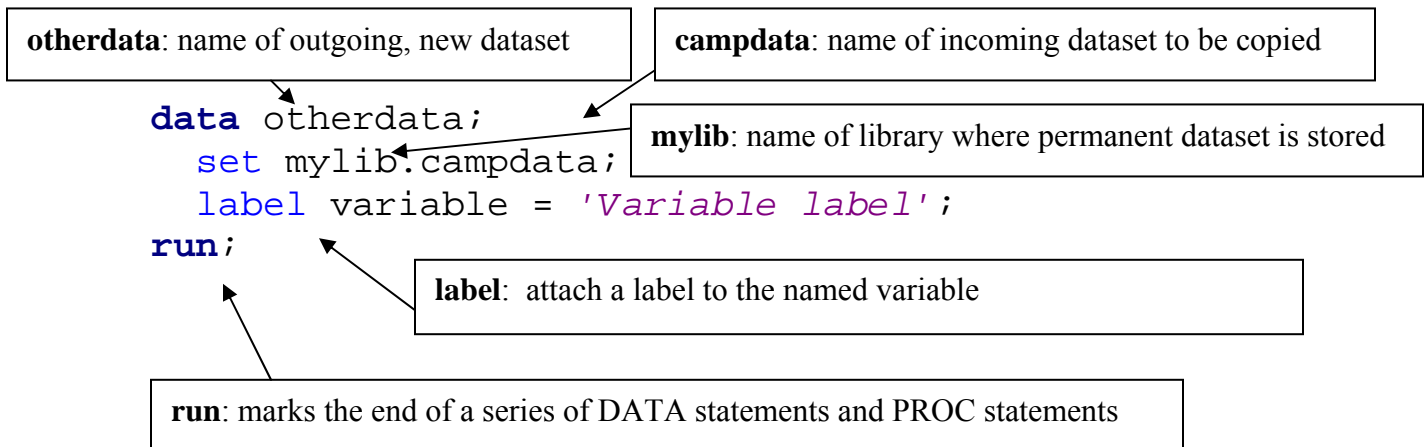
Do the same as in Exercise 11, only add a label to two variables in the same DATA STEP. Note: there's no semicolon between the label assignments, just list them out like:

```
label variable1='label1' variable2='label2';
```

Try it for, say, the variables HOURSLW (hours worked last week) and LNAME (last name) or any other variables of your choice. Check if the dataset has the labels either with a PROC CONTENTS or a PROC PRINT with the LABEL option.

Basic DATA STEP processing

Let's identify parts of the DATA STEP.



When you submit a DATA step for execution, it is first compiled and then executed one observation at a time. That's one reason SAS is so efficient with large datasets!

Exercise 12:

Create a new dataset called EARNINGS and make it permanent (i.e. save it by using your library prefix MYLIB) from copying the old dataset MYLIB.PERMANENT. Now let's create a new variable called EARNLW (Earnings last week) by multiplying EARNHRE (Earnings per hour in pennies) and HOURSLW (hours worked last week). Since the EARNHRE is in pennies, let's divide by 100 to make EARNLW in the unit of dollars.

Use PROC PRINT to see the results. Try specifying variables in the PROC PRINT

statement, such as

```
proc print data=mylib.earnings;  
  var hhid hourslw earnhre earnlw;  
run;
```

Exercise 12b optional:

Re-create the permanent dataset EARNINGS by adding another new variable. Create a variable called EARN_GRADE by computing how much a person earns per week (EARNWKE) divided by their highest grade complete (IHIGRDC). Thus we can compute a rate of weekly earnings by years of schooling. The code should look something like this.

```
earn_grade = earnwke/ihigrdc;
```

Use PROC PRINT to see your new variable.

IF-THEN-ELSE

So the usual operations (+, -, *, /) work in SAS. Plus there are a huge number of functions in SAS, both for numeric and character data, like SUM, MIN, MAX, and MEAN. These are all documented, including their proper syntax.

What about other data manipulation commands? One of the most useful is the IF-THEN-ELSE statements. Here are a few examples that we'll try together.

Exercise 13:

```
* 1: Just IF;  
data try1;  
  set mylib.earnings;  
  if earnwke < 500;  
run;  
  
proc print data=try1;  
  var hhid earnwke;  
run;  
  
* 2: IF-THEN;  
data try2;  
  set mylib.earnings;  
  if earnwke < 500 then ewk_group = 1;  
run;  
  
proc print data=try2;  
  var hhid earnwke ewk_group;  
run;  
  
* 3: IF-THEN-ELSE;  
data try3;
```

```
set mylib.earnings;
if earnwke < 500 then ewk_group = 1;
else if earnwke < 1000 then ewk_group = 2;
else ewk_group = 3;
run;

proc print data=try3;
var hhid earnwke ewk_group;
run;
```

Exercise 13b optional:

What are some of the properties of this new categorical variable EWK_GROUP?
Run the following code.

```
* Exercise 13b: PROC FREQ;
proc freq data=try3;
tables ewk_group;
run;
```

Suppose you want to check your logic in creating EWK_GROUP. Try a crosstab. Use asteric (*) to “cross” two variables to get frequencies by categories.

```
proc freq data=try3;
tables earnwke*ewk_group;
run;
```

We’ve completed Part I. Now let’s look at more data manipulation, descriptive statistics and analysis techniques.

Part II

Below is the description of the data we’ll be using for the next sections.

Example Data: sub-sample of Add Health (public version)

We will be using data from the National Longitudinal Study of Adolescent Health (Add Health), a United States nationally representative study that explores the causes of health-related behaviors of adolescents in grades 7 through 12 and their outcomes in young adulthood. Our example data comes from the Wave I, public version file with 6,504 observations that are a random sample of 50 percent of the Add Health Wave I core sample and 50 percent of the Add Health Wave I high education black sample. For simplicity, we’re using just a sub-sample of the full dataset. For more information and to gain access to more Add Health data, both the public and the restricted versions, go to the CSDE website (<http://csde.washington.edu/>) or email Cori Mar (cmmar@u.washington.edu).

Sub-sample contents:

| | |
|--------|-----------------------------|
| AID | RESPONDENT IDENTIFIER |
| HIGH50 | USUAL BEDTIME ON WEEKNIGHTS |
| S1 | HOW OLD ARE YOU? |

| | |
|------|--|
| S2 | WHAT SEX ARE YOU? 1 = male 2 = female |
| S3 | WHAT GRADE ARE YOU IN? |
| S27 | HOW MANY PEOPLE LIVE IN HOUSEHOLD? |
| S50 | HOW IS YOUR HEALTH? 1 = excellent 2 = very good 3 = good 4 = fair 5 = poor 9 = multiple responses |
| S10A | GRADE IN ENGLISH 4 = A 3 = B 2 = C 1 = D 0 = F |
| S10B | GRADE IN MATHEMATICS (same as S10A) |
| S10C | GRADE IN HISTORY/SOCIAL STUDIES (same as S10A) |
| S10D | GRADE IN SCIENCE (same as S10A) |
| S59A | SMOKED CIGARETTES-LAST 12 MTHS 0 = never 1 = once or twice 2 = once a month or less 3 = 2 or 3 days a month 4 = once or twice a week 5 = 3 to 5 days a week 6 = nearly every day 99 = multiple responses |

As before, you should have created a folder or already have one in mind where you will temporarily store datasets and save your SAS code. If you haven't done so already, please create or identify a folder now.

We can create a library to identify the location of this folder.
Recall the syntax:

LIBNAME *libref* 'SAS-data-library';

Exercise 14:

Create a library that will “point” to the storage folder you identified. Then display the contents of a SAS dataset called AHW1S1 (Add Health Wave I Sample 1).

```
* Exercise 14: typical libname statement;  
libname MyData 'c:\This\is\a\path\to\my\data\folder';  
  
proc contents data=MyData.ahw1s1;  
run;
```

Exercise 14b optional: Print the AHW1S1 dataset.

Recall that the DATA STEP recognizes all standard operations like addition (+), subtraction (-), multiplication (*), and division (/).

Exercise 15:

Create a new variable called GPA representing the average of the four grades reported (English, Math, History/Social Studies, and Science). Create a new permanent dataset AHGPA (copied from the original permanent dataset AHW1S1) and add a descriptive label to the new variable. Print the revised dataset.

```
* Exercise 15: Create GPA variable by averaging the grades over four  
courses;  
data MyData.ahgpa;  
  set MyData.ahw1s1;  
  gpa = (s10Eng + s10Mat + s10His + s10Sci)/4;  
  label gpa = 'Grade point average for four courses';  
run;  
  
* Add LABEL in the PROC PRINT statement;  
proc print data=MyData.ahgpa label;  
  var aid s10Eng s10Mat s10His s10Sci gpa;  
run;
```

Exercise 16:

Try computing the GPA using the MEAN function in SAS as follows. Call the new variable GPA_MEAN, add it to your permanent dataset AHGPA, and compare its values to the GPA variable values.

```
gpa_mean = mean(s10Eng, s10Mat, s10His, s10Sci);
```

The DATA STEP allows you to do the same operation over and over again, only using a small amount of SAS code. We will do an operation multiple times using a simple ARRAY statement. Keep in mind that multi-dimensional matrices can also be built using ARRAY.

The syntax for the ARRAY statement is

ARRAY *array-name*<{n}><\$><length><elements><(initial-values)>;

Exercise 17:

Suppose Fred would like to increase everyone's GPA and he's going to accomplish this

by increasing every reported grade by 0.5. (Let's not keep this "corrupted" dataset long-term; make it temporary and call it AHGPA_MOD.)

```
* Exercise 17: Add 0.5 to every grade;
data ahgpa_mod;
  set MyData.ahgpa;
  array class{4} s10Eng s10Mat s10His s10Sci;
  do i = 1 to 4;
    class[i] = class[i] + 0.5;
  end;
run;

proc print data=ahgpa_mod;
  var aid s10Eng s10Mat s10His s10Sci;
run;
```

Note: variables in an array have to be of the same type. For example, variables would need to be all numeric or all character.

There's a problem with these grades. What is it? (Consider that grades at least in this school can not exceed 4.0. Is a grade of 4.5 plausible?)

Exercise 18:

Fix the problem with grades that look like 4.5 (4.0 is the maximum grade achievable in these schools) by re-computing these grades. But this time let's create some new grade variables rather than overwrite the old ones using a second array statement.

```
* Exercise 18: Add 0.5 to every grade less than 4.0 and create new
grade variables;
data ahgpa_mod;
  set MyData.ahgpa;
  array class{4} s10Eng s10Mat s10His s10Sci;
  array newclass{4} s10NewEng s10NewMat s10NewHis s10NewSci;
  do i = 1 to 4;
    if class[i] < 4.0 then newclass[i] = class[i] + 0.5;
    else newclass[i] = class[i];
  end;
run;

proc print data=ahgpa_mod;
  var aid s10Eng s10NewEng s10Mat s10NewMat s10His s10NewHis s10Sci
  s10NewSci;
run;
```

Formats

We can assign formats to variable values. For example, S59A (SMOKED CIGARETTES-LAST 12 MTHS) has values with the corresponding interpretation: 0 = never, 1 = once or twice, 2 = once a month or less, etc. Formats can remind you of the meaning behind the category by displaying text instead of numbers. Note that formats do not change the actual value of the variable, just how it's displayed.

Formats are created in the FORMAT procedure with the following syntax. Note: format names are limited to eight (8) characters in length!

```
PROC FORMAT <option(s)>;  
  VALUE <$>name <(format-option(s))>  
  value-range-set(s);  
RUN;
```

Using formats is a two step process.

- 1) Create the format using PROC FORMAT.
- 2) Assign the format to the variable.

Exercise 19:

Format the variable SEX to be 1 = Male and 2 = Female. Add this format to the permanent dataset AHGPA. Display the assigned format by the CONTENTS and the PRINT procedures.

```
* Exercise 19: Format S2 (SEX) as 1=Male and 2=Female;  
* Part 1: Create the format;  
proc format;  
  value sexfmt  
    1 = 'Male'  
    2 = 'Female';  
run;  
  
* Part 2: Assign the format;  
data MyData.ahgpa;  
  set MyData.ahgpa;  
  format s2 sexfmt.;  
run;  
  
* Display the new format two ways;  
proc contents data=MyData.ahgpa;  
run;  
  
* Try the option (OBS= ) to limit the number of displayed observations;  
proc print data=MyData.ahgpa (obs=20);  
run;
```

Exercise 19b optional:

Format the variable S50 (How is your health?) to be 1 = excellent, 2 = very good, 3 = good, 4 = fair, 5 = poor, 9 = multiple responses. Assign this format in the DATA STEP to recreate the permanent dataset AHGPA. Display it.

Merging Datasets

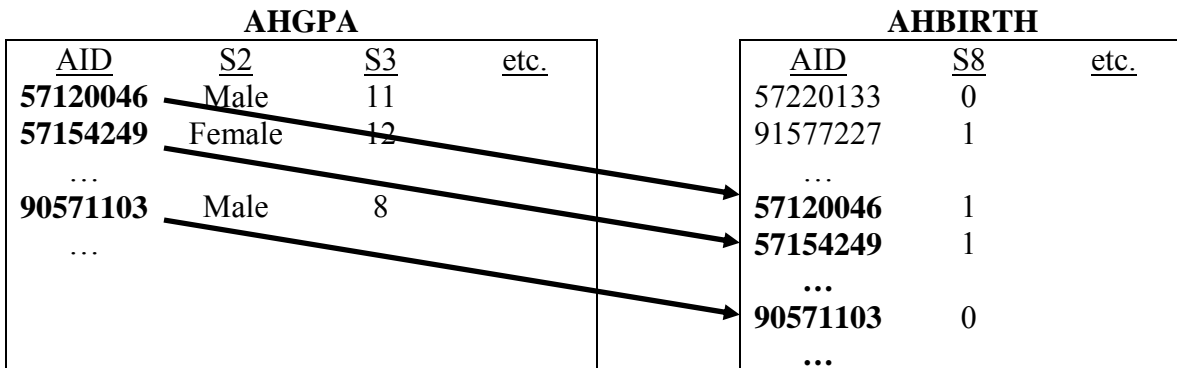
What if we want to combine two datasets by matching on a key variable?

One-to-one match merge

Suppose each observation in a dataset has a unique matching observation in another dataset. We would like to combine this into one dataset that includes all information

from both datasets. This is a one-to-one match merge. An example of a one-to-one match merge is combining the dataset AHGPA to the dataset AHBIRTH on the key variable AID. Effectively, we will add birth origin of the student and his or her parents to the larger dataset.

| Example Data: sub-sample of Add Health (public version) | |
|---|--|
| For the purpose of the example below, we have another permanent dataset called AHBIRTH containing information about birth origin. Like the previous dataset, this is a sub-sample taken from the larger Add Health Wave I public dataset. | |
| Sub-sample contents: | |
| AID S8 | RESPONDENT IDENTIFIER BORN IN THE UNITED STATES? 0 = no 1 = yes |
| S13 | MOTHER BORN IN U.S.? 0 = no 1 = yes 7 = legitimate skip 8 = I don't know |
| S19 | FATHER BORN IN U.S.? 0 = no 1 = yes 7 = legitimate skip 8 = I don't know |



Note: Match-merge requires the SORT procedure to order records before matching if the dataset isn't already sorted on the key variable.

Exercise 20: one-to-one match merge

Merge datasets AHGPA and AHBIRTH on the key variable AID. Be sure to sort on AID before merging. Create new permanent dataset AH1. Print the resultant dataset.

Note: the **out=dataset_name** option on the PROC SORT saves the sorted data to a new

dataset named in the statement. This is only necessary when you want to preserve the ordering in the original dataset. Otherwise you can omit this option and the original dataset will be overwritten with the sorted version.

```
* Exercise 20: Merge AHGPA with AHBIRTH on AID;
proc sort data=MyData.ahgpa out=ahgpa;
  by aid;
run;

proc sort data=MyData.ahbirth out=ahbirth;
  by aid;
run;

data MyData.ah1;
  merge
    ahgpa
    ahbirth;
  by aid;
run;

proc print data=MyData.ah1;
run;
```

One-to-many match merge

Not all datasets of interest have precisely a one-to-one observation match structure. Sometimes it is of interest to propagate information from one dataset to another based on other characteristics besides a unique identification variable. This is an example of a one-to-many match merge. Here we merge datasets AH1 and AHAGE, matching on key variable S3 (Grade). Effectively, we will be assigning the average age by each grade to the observations at the individual level.

Example Data: average age by grade from Add Health Wave I (public version)

For this example, a dataset called AHAGE contains the average age by grade from the Add Health Wave I public dataset.

Contents:

| | |
|---------|------------------------|
| S3 | WHAT GRADE ARE YOU IN? |
| MEANAGE | AVERAGE AGE BY GRADE |

| AHGPA | | | | AHAGE | |
|----------|--------|-----|------|-------|---------|
| AID | S2 | S3 | etc. | S3 | MEANAGE |
| 57120046 | Male | 11 | | 7 | 12.4332 |
| 57154249 | Female | 12 | | 8 | 13.4559 |
| ... | | ... | | 9 | 14.3858 |
| 90571103 | Male | 8 | | 10 | 15.3423 |
| 90676053 | Female | 11 | | 11 | 16.3099 |
| | | ... | | 12 | 17.2329 |

Exercise 21: one-to-many match merge

Merge datasets AHGPA and AHAGE on the key variable S3 (Grade). Create a new permanent dataset AH2. Print the resultant dataset.

```
* Exercise 21: Merge AH1 and AHAGE on S3 (Grade);  
proc sort data=MyData.ah1;  
  by s3;  
run;  
  
proc sort data=MyData.ahage;  
  by s3;  
run;  
  
data MyData.ah2;  
  merge  
    MyData.ah1  
    MyData.ahage;  
  by s3;  
run;  
  
proc print data=MyData.ah2;  
run;
```

Descriptive statistics procedures

There are many procedures in SAS. For basic descriptive statistics for continuous variables, use UNIVARIATE.

```
PROC UNIVARIATE < options > ;  
  BY variables ;  
  VAR variables ;  
  HISTOGRAM ;  
  QQPLOT ;
```

Exercise 22:

What are the mean, the standard deviation, the median, and the minimum of the variable S27 (HOW MANY PEOPLE LIVE IN HOUSEHOLD?) in the AH2 dataset?

```
* Exercise 22: Descriptive statistics for variable S27 (HOW MANY PEOPLE  
LIVE IN HOUSEHOLD?);  
proc univariate data=MyData.ah2;  
  var s27;  
run;
```

The distribution of the data can be visualized quickly with the use of the HISTOGRAM statement. Also, a QQPLOT is useful in determining how well the data resembles a normal distribution.

Exercise 23:

Plot a histogram of the distribution of S27 for all people in this sample. Also, add a QQPLOT.

```
* Exercise 23: Add a histogram and qqplot;  
proc univariate data=MyData.ah2;  
  var s27;  
  histogram;  
  qqplot;  
run;  
quit;
```

Note: **quit** is used to end this procedure because it helps tell SAS that the graphing portion of the procedure is concluded.

Now consider information about S27 (HOW MANY PEOPLE LIVE IN HOUSEHOLD) grouped by S8 (BORN IN THE US). Before using the BY statement in the PROC UNIVARIATE, we need to sort the data. Any BY statement requires pre-sorting by the variables in that statement.

Exercise 24:

What is the median of the variable S27 (how many people in household) in the AH2 dataset for students that were born in the US and those that were not born in the US (S8), separately?

```
* Exercise 24: Descriptive statistics for variable S27 (HOW MANY PEOPLE  
LIVE IN HOUSEHOLD) by S8 (BORN IN THE US);  
proc sort data=MyData.ah2;  
  by s8;  
run;  
  
proc univariate data=MyData.ah2;  
  var s27;  
  by s8;  
run;
```

Question:

1) What is the median number of people living in a household for those born in the US and for those not born in the US?

One of the most popular procedures is PROC FREQ, specifically used for categorical data.

```
PROC FREQ < options > ;  
  TABLES requests < / options > ;
```

Exercise 25:

What proportion of students report FAIR health (a category of S50)?

```
* Exercise 25: Frequency of the variable S50 (HEALTH);  
proc freq data=MyData.ah2;  
  tables s50;  
run;
```

Questions:

- 1) What percentage of students report FAIR health? (Think about your denominator with regard to missing values.)
- 2) Among those that responded, what percentage of students report GOOD health or better?

You can also use PROC FREQ to perform cross-tabulations.

Exercise 26:

Is there a difference in reported health by sex?

```
* Exercise 26: Crosstabulation of S50 (Health) with S2 (Sex);  
proc freq data=MyData.ah2;  
  tables s2 * s50;  
run;
```

Questions:

- 1) What percent of the overall number of people are male?
- 2) What percent of the overall number of people report excellent health?
- 3) Among females, what percent report good health?
- 4) Among people that very good health, what percent are male?

Introduction to regression procedures

SAS has a procedure for nearly any kind of analysis. One important analysis method is Ordinary Least Squares (OLS) regression. This can be performed using any number of procedures. Below we feature the procedure MIXED.

```
PROC MIXED < options > ;  
  CLASS variables < / option > ;  
  MODEL dependents=independents < / options > ;
```

Exercise 27:

Perform an ordinary least squares regression with GPA as the dependent variable, and S50 (health), S2 (sex), and S8 (born in the US) as the independent variables.

```
* Exercise 27: OLS with outcome GPA and predictors S50 (health), S2  
(Sex), and S8 (BORN IN THE US);  
proc mixed data=MyData.ah2;  
  class s50;  
  model gpa = s50 s2 s8 / intercept solution;
```

```
run;
```

Questions:

- 1) Why is S50 (health) in the class statement?
- 2) What is the estimated coefficient for S8 (born in the US) in the model? Is it statistically significant?

For binary outcomes, we can use LOGISTIC regression. The linear logistic model has the form

$$\text{logit}(\pi) \equiv \log\left(\frac{\pi}{1-\pi}\right) = \alpha + \beta'x$$

where

π = probability of the “event” occurring as indicated by the binary outcome variable

α = intercept term

$\beta'x$ = is a linear combination of coefficients multiplied by their corresponding covariates

The SAS syntax looks like

```
PROC LOGISTIC < options > ;  
  CLASS variables < / option > ;  
  MODEL dependents=independents < / options > ;
```

Exercise 28:

Perform a logistic regression with ever smoked in the last 12 months (you need to create this from S59A) as the dependent variable, and S1 (age) and GPA as the independent variables.

```
* Exercise 28: Logistic regression with outcome smoked in the last 12 months;
```

```
* Create a variable SMOKED12MO to be 0 = did not smoke in the last 12 months and 1 = ever smoked in the last 12 months;
```

```
data mydata.ah2;  
  set mydata.ah2;  
  if s59a eq . then smoked12mo = .;  
  else if s59a eq 0 then smoked12mo = 0;  
  else smoked12mo = 1;  
  label smoked12mo = 'EVER SMOKED IN LAST 12 MONTHS';  
run;
```

```
proc logistic data=MyData.ah2;  
  model smoked12mo = s1 gpa;  
run;
```

Question:

- 1) Does this model fit?

Exercise 28b optional:

Adjust the reference category to reflect SMOKED12MO = 0.