**Getting Started With SAS**

**Things to Note**

* SAS programs typically end with “Run”;
* Semi-colons separate individual commands in a SAS program or script”;”
* Capitalization does not matter
* Comment by placing text between a dash-star combo (e.g. /\* I can now comment because I am in between the dashes and stars\*/

**SAS Libraries**

* Folders in SAS that store SAS datasets and formats (how we label variables)
* Connected to a real folder on your computer
* If you don’t use a library, all datasets will be stored in the work library and then be deleted once you close shop

**Non-syntax Steps**

1. Open libraries in the left-hand side of the SAS workplace
2. Right-click in the SAS library section
3. Click “new library”
4. Give name
   1. Can only be eight characters long and cannot have spaces
   2. We will use the name “CAT\_lab1”
5. Click Browse under “path” to choose the folder on the computer where you want the SAS library linked to
   1. Since you probably don’t have this external folder created on the computer, it’s probably best to first create the folder and then specify the path
6. Click the “re-create this library at start-up” box if you always want this library to be present on the opening of SAS
   1. If you don’t click this, while the dataset’s and formats will save to the external folder you specify the library will not be accessible on SAS after closing and you will have to recreate every time
   2. Some people prefer this so not to clutter their SAS workspace

**Syntax** – This is useful if you want to create a library quickly without permanently saving it on SAS

libname CAT\_lab1 '/folders/myfolders/Cat\_lab1';

/\*between dashes stars is a comment in sas and won’t mess with program\*/

/\*libname = SAS function\*/

/\*CAT\_lab1 = name of library\*/

/\*'/folders/myfolders/Cat\_lab1'= path to external folder that you are connecting your library to\*/

**Creating a dataset from scratch in SAS**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **name** | **date** | **job** | **friends** | **alcohol** | **income** | **neuroticism** |
| **Leo** | **1-Jan-60** |  | **5** | **10** | **20000** | **10** |
| **Martin** | **24-May-69** | **1** | **2** |  | **40000** | **17** |
| **Andy** | **21-Jun-73** | **1** | **0** | **20** | **35000** | **14** |
| **Paul** | **16-Jul-70** | **1** | **4** | **5** | **22000** | **13** |
| **Graham** | **10-Oct-49** | **1** | **1** | **30** | **50000** | **21** |
| **Carina** | **5-Nov-83** | **2** | **10** | **25** | **5000** | **7** |
|  | **8-Oct-87** | **2** | **12** | **20** | **100** | **13** |
| **Doug** | **16-Sep-89** | **2** | **15** | **16** | **3000** | **9** |
| **Mark** | **20-May-73** | **2** | **12** | **17** | **10000** | **14** |
| **Sam** | **28-Mar-90** | **2** | **17** | **18** | **10** | **13** |

* note you don’t need all this code below to create dataset, but I added extra to show you some extra things you can do
* To create Dataset at a minimum, you need the following lines: DATA, Input, Cards, Run

DATA CAT\_lab1.first\_data;  
/\*creates data name to use\*/

/\* the name before the period “.” specifies the library to save to & the name after the period “.” specifies the name of the SAS dataset\*/  
informat name $ 10.;  
/\*informat changes string variable length to different length number (default is 8)\*/  
INPUT name $  
 dateofbirth date9.   
job friends alcohol  
income neuroticism;

/\* input means we have variable names following\*/  
/\*date9. makes underlying information into the date form yy-mm-dd\*/  
format dateofbirth date9.;  
/\*this line makes a date actually visible as a date\*/  
format friends 6.3;  
/\*this specifies that are numeric variable friends will have 6 total characters with 3 decimals\*/  
CARDS;  
Leo 01JAN1960 1 5 10 20000 10  
Martin 24May1969 . 2 15 40000 17  
Andy 21Jun1973 1 0 . 35000 14  
Paul 16Jul1970 1 4 5 22000 13  
Graham 10Oct1949 1 1 30 50000 21  
Carina 05Nov1983 2 10 25 5000 7  
'' 08Oct1987 2 12 20 30000 13  
Doug 16Sep1989 2 15 16 3000 9  
Mark 20May1973 2 12 17 10000 14  
Sam 28Mar1990 2 17 18 100000 13  
;  
/\* cards is place to write data\*/

/\* could also use “datalines” instead of “CARDS” \*/

/\* use '' to represent missing string value\*/

/\* use . to represent missing numeric value\*/  
RUN;

**Adding Grouping and Variable Labels**

* Note we could do all this above when we create the dataset, but I wanted to show you how to do it separately also to show you how to create a new dataset from an old dataset

PROC FORMAT;  
VALUE job  
1= ‘Lecturer’  
2= ‘Student’  
;  
RUN;

/\*this whole thing creates a format code to tell a grouping variable later the names of the levels\*/

/\*technically we are just creating a custom format function\*/

/\*this proc format is not necessarily needed, but I find it easier than creating new variable using if-then statements, deleting the old variable, and finally renaming the new variable the old variable name\*/

DATA CAT\_lab1.first\_data\_add\_labels;   
/\*data tells the program to make new dataset\*/  
SET CAT\_lab1.first\_data;  
/\*set tells what old dataset to create a new one from\*/  
FORMAT job job. ;  
/\* this uses previously created job function to label our categorical variable\*/  
LABEL date = “Date of Birth”  
job = “Position”  
friends = “Number of Friends”  
alcohol = “Alcohol Units Consumed”  
income = “Income”  
neuroticism = “Neuroticism Score”  
;

/\*label function allows us to give longer and better descriptions of our variables if need be\*/  
RUN;

**Permanently Rename a variable**

DATA CAT\_lab1.first\_data;   
SET CAT\_lab1.first\_data;

/\*notice how the dataset after data and set are the same – this is because we are changing the current dataset & not creating a new one\*/  
RENAME neuroticism = crazy;   
/\*this is the relevant code\*/;  
Run;

**Dropping a variable**

DATA CAT\_lab1.first\_data\_DropVar;   
SET CAT\_lab1.first\_data;   
drop crazy ;

Run;

**Applying a mathematical transformation to a numeric variable**

* note that SAS has easier to use tasks and utilities to do this for most common transformations

DATA CAT\_lab1.first\_data\_trans;   
SET CAT\_lab1.first\_data;   
income\_in\_thousands = income / 1000;

/\*technically we are creating a new transformed variable & not overwriting the old variable\*/  
run;

**Using If-Then Statement to Create new Variables**

Common SAS language

GT or >– Greater Than  
LT or < – Less Than  
GE or >= – Greater than or Equal to  
LE or <= – Less than or Equal to  
EQ or = – Equal to  
NE or ^= – Not Equal to

Note – Usually, SAS treats missing values by using deletion, however when specifying if-then statements, it treats them as 0’s which messes with stuff. As such, we must plan for this in our if-then statements by specifying missing as periods “.”.

**Deleting Case (deleting row) – to create a missing value**

DATA CAT\_lab1.first\_data\_Delete;

SET CAT\_lab1.first\_data;

IF alcohol = 10 THEN DELETE;

RUN;

**Replacing Case**

/\*replacing variable\*/

DATA CAT\_lab1.first\_data\_Replace;

SET CAT\_lab1.first\_data;

IF alcohol = 10 THEN alcohol = .;

RUN;

**Creating new string variable from numeric variable**

DATA CAT\_lab1.first\_data\_NewStringVar;   
SET CAT\_lab1.first\_data;  
informat too\_much\_alcohol $ 20.;

/\*creating a new variable named “too\_much\_alchohol” which is a string variable and can take 20 characters\*/  
if alcohol = . then too\_much\_alcohol = .;

/\* saying that if alcohol is missing then so to is our new variable\*/  
else if alcohol <= 20 then too\_much\_alcohol = 'Not\_Drunk';

/\* saying that if alcohol is less than or equal to 20, we are labeling the person as “not drunk”\*/

/\*notice how we use “else if” after the first “if” is used\*/  
else if alcohol >=21 and alcohol <= 25 then too\_much\_alcohol = 'Drunk';  
else if alcohol >25 then too\_much\_alcohol = 'Too\_Drunk';  
run;

**Creating new numeric variable from numeric variable**

DATA CAT\_lab1.first\_data\_NewNumVar;   
SET CAT\_lab1.first\_data;  
if alcohol = . then too\_much\_alcohol = .;  
else if alcohol <= 20 then too\_much\_alcohol = 0;  
else if alcohol >=21 and alcohol <= 25 then too\_much\_alcohol = 1;  
else if alcohol >25 then too\_much\_alcohol = 2;  
run;

**Listing Variable Names**

proc contents  
 data = CAT\_lab1.first\_data  
 noprint  
 out = data\_info  
 (keep = name varnum);

/\* “data\_info” is name of new dataset (saved in work) that contains variable names\*/  
run;

**Some descriptive and other result options**

**Print data table**

PROC PRINT DATA= CAT\_lab1.first\_data;  
RUN;

**Frequency table**

PROC FREQ DATA= CAT\_lab1.first\_data;  
RUN;

/\*creates all freq tables from CAT\_lab1.first\_data dataset\*/

/\*Or\*/

PROC FREQ DATA= CAT\_lab1.first\_data(keep=job);

/\* “keep = job” tells SAS to only create freq table for job variable\*/  
RUN;

**Mean stdv min max**

PROC means DATA= CAT\_lab1.first\_data;

RUN;

/\*Or\*/

PROC means DATA= CAT\_lab1.first\_data;

Var job;

/\*this line just tells the program to use only the job variable\*/

RUN;

**Detailed Summary Statistics**

PROC UNIVARIATE data = CAT\_lab1.first\_data;  
RUN;

/\*Or\*/

PROC UNIVARIATE data = CAT\_lab1.first\_data;  
 var job;  
RUN;

**Data table information**

PROC CONTENTS DATA= CAT\_lab1.first\_data;

RUN;

**Saving Results**

1. Run some function to get results
2. At the top of the results page click one of three download buttons
   1. Download results as a HTML file
   2. Download results as a pdf file
   3. Download results as a rtf file
      1. This last option is word

**Loading more than one function in the SAS result section**

* Either turn on “go interactive” in the SAS code box
  + In vague terms this button in the upper right corner of the code section and looks like a greater than symbol nested over a box
* Or simply place all relevant proc functions before “run;” such as . . .

PROC PRINT DATA= CAT\_lab1.first\_data;

Proc FREQ DATA = CAT\_lab1.first\_data;

PROC MEANS DATA= CAT\_lab1.first\_data;

PROC CONTENTS DATA= CAT\_lab1.first\_data;

RUN;

**Importing from Excel**

%web\_drop\_table(CAT\_lab1.first\_data\_excel);

/\* CAT\_lab1.first\_data\_excel = name of library (CAT\_lab1) we are saving SAS data to, and name of sas dataset (first\_data\_excel)\*/

FILENAME REFFILE '/folders/myfolders/CAT\_lab1/lab1.xlsx';

/\* '/folders/myfolders/CAT\_lab1/lab1.xlsx' file path location of excel on computer \*/

PROC IMPORT DATAFILE=REFFILE

DBMS=XLSX

OUT=CAT\_lab1.first\_data\_excel;

GETNAMES=YES;

RUN;

PROC CONTENTS DATA=CAT\_lab1.first\_data\_excel; RUN;

%web\_open\_table(CAT\_lab1.first\_data\_excel);