COLUMBIA UNIVERSITY | MAILMAN SCHOOL of PUBLIC HEALTH

IRVING INSTITUTE FOR clinical and translational research

Biostatistics, Epidemiology & Research Design education initiatives

GETTING STARTED WITH SAS (PART 1)

Christine Mauro, PhD

April 5th, 2018

COLUMBIA UNIVERSITY | MAILMAN SCHOOL of PUBLIC HEALTH BIOSTATISTICS

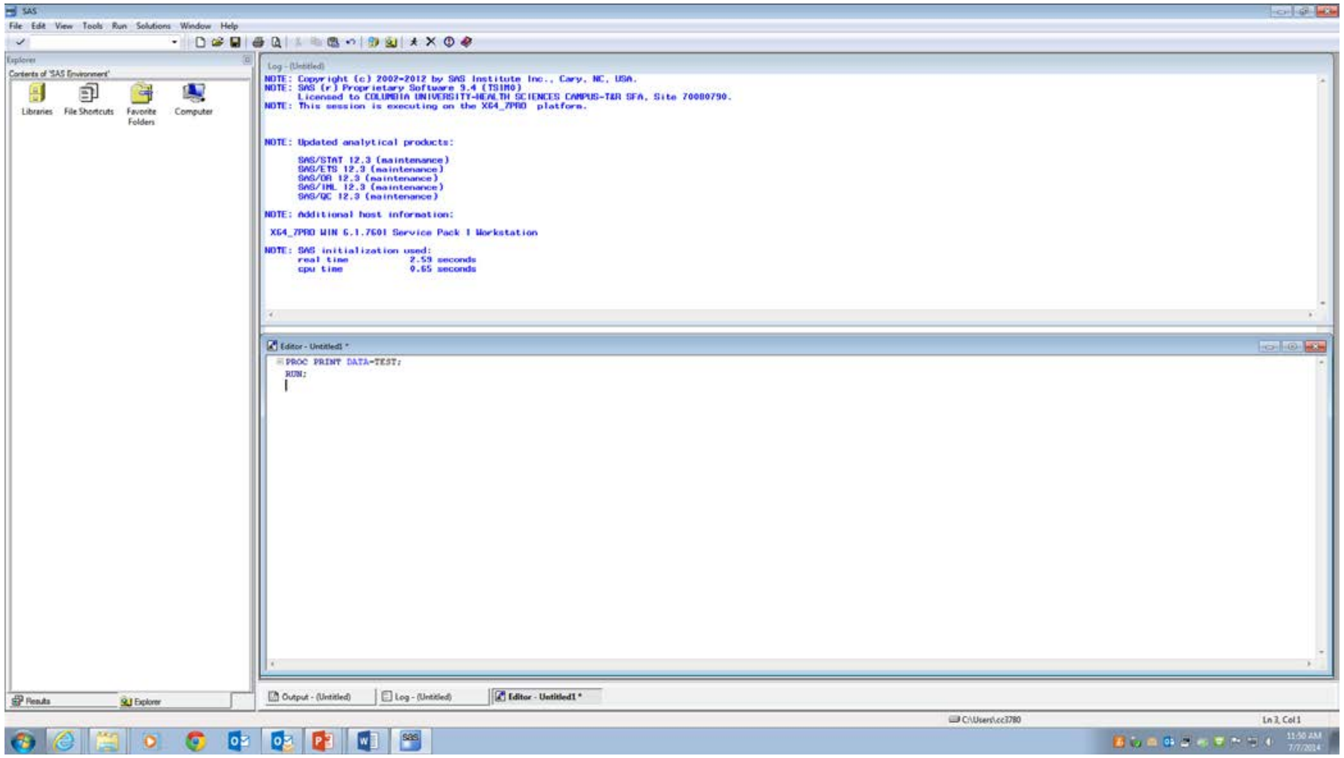
Outline

- SAS Overview
- Importing Data
- Examining Data Attributes
- Manipulating Data

SAS OVERVIEW

SAS Windows

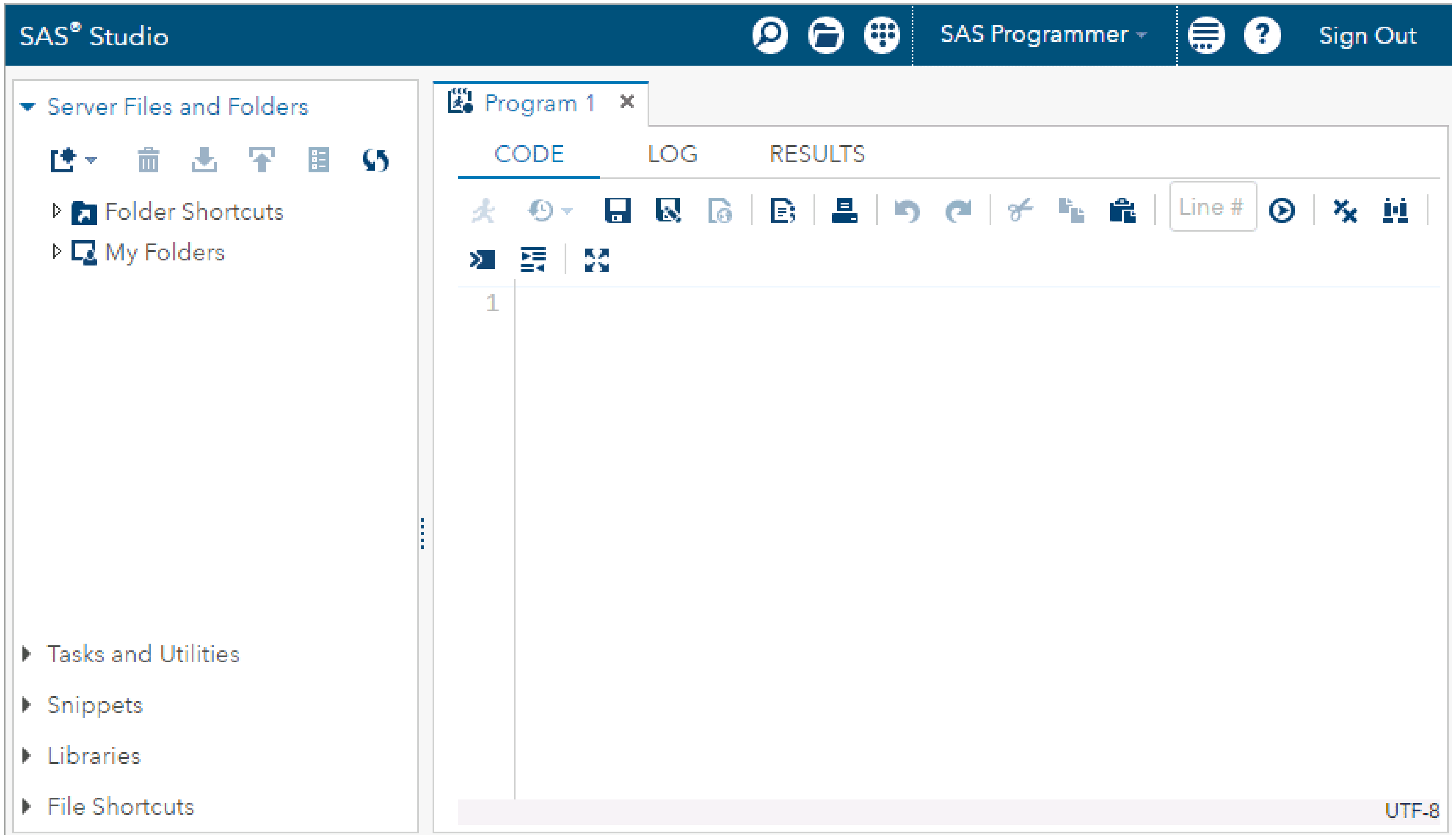
- Results and Explorer
- Programming windows: Editor, Log and Output



```
NOTE: Copyright (c) 2002-2012 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.4 (TS1M0)
      Licensed to COLUMBIA UNIVERSITY-HEALTH SCIENCES CAMPUS-T&R SFA, Site 70080790.
NOTE: This session is executing on the X64_7PRO  platform.



NOTE: Updated analytical products:

      SAS/STAT 12.3 (maintenance)
      SAS/ETS 12.3 (maintenance)
      SAS/OR 12.3 (maintenance)
      SAS/IML 12.3 (maintenance)
      SAS/QC 12.3 (maintenance)

NOTE: Additional host information:

 X64_7PRO WIN 6.1.7601 Service Pack 1 Workstation

NOTE: SAS initialization used:
      real time           2.59 seconds
      cpu time            0.65 seconds
```

```
PROC PRINT DATA=TEST;
RUN;
```

SAS Windows (UE)



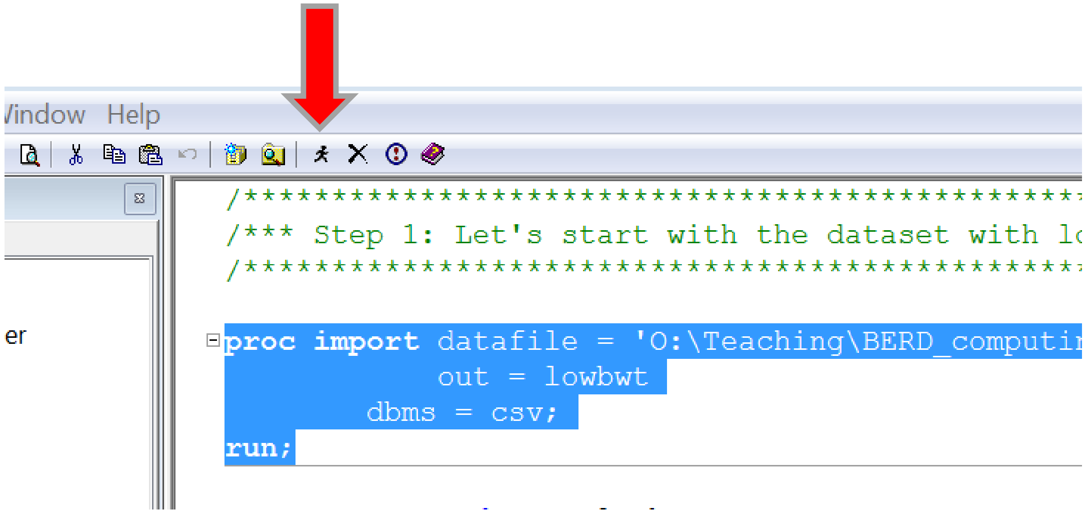
SAS Windows

- **Editor**: use to type in, edit, and submit SAS programs; color coded
  - Make sure to save editor/code for future use!
- **Log**: contains notes about the SAS session including errors and warnings associated with your program
- **Output**: display any printable results
- **Results**: table of contents for the Output window (tree lists)
- **Explorer**: gives access to SAS (data) files and libraries

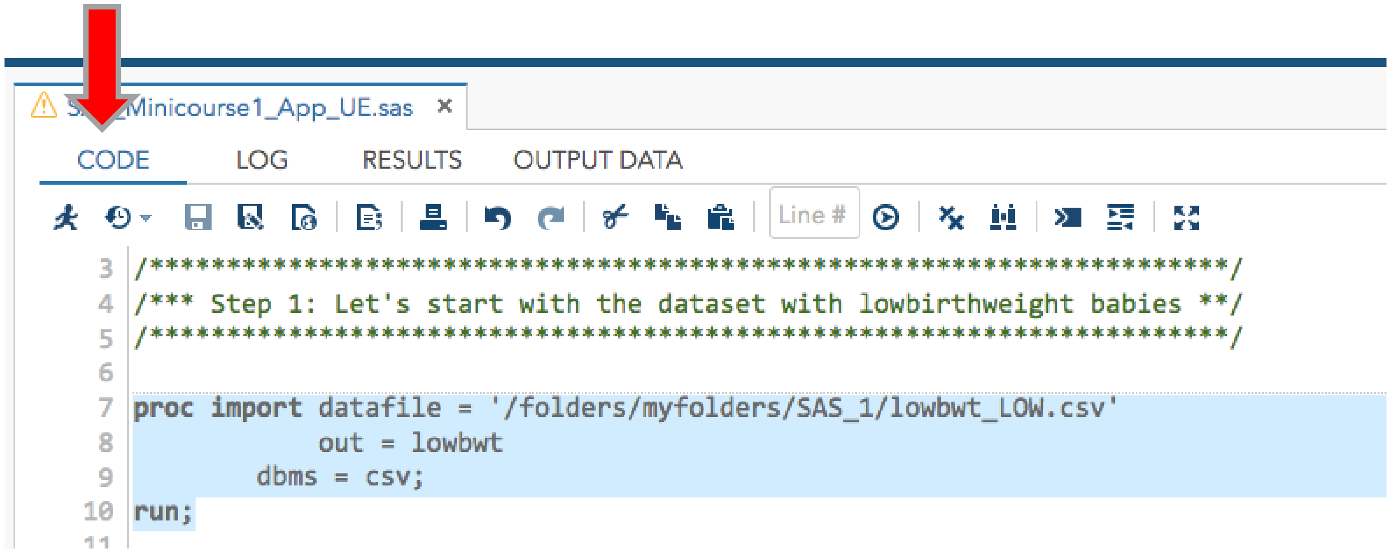
Write your program in the **Editor/CODE** window, select it, and submit it.

Check **Output** and **Log** for results.

Running Code



```
/*****************************************************
/*** Step 1: Let's start with the dataset with lo
/*****************************************************

proc import datafile = 'O:\Teaching\BERD_computi
            out = lowbwt
        dbms = csv;
run;
```



```
/*****************************************************************/
/*** Step 1: Let's start with the dataset with lowbirthweight babies **/
/*****************************************************************/

proc import datafile = '/folders/myfolders/SAS_1/lowbwt_LOW.csv'
            out = lowbwt
        dbms = csv;
run;
```

Basic Structure of SAS

- Most programs have two major components:
  - **Data step(s)** – reads data, manipulates/combines it and print reports
  - **Procedure step(s)** – perform analysis on the data and generate output
- Sequential execution of statements
  - Every statement ends in a semicolon ;
  - Statement can be more than one line
  - NOT case sensitive! Exception – "quoted" strings
- Useful messages are written to the SAS log

Helpful Rules

- SAS variable/data set names must be 32 characters or less
  - can only contain letters, numbers, and underscores (do not start with an underscore)
  - Name your variables in a *concise* yet *informative* way
- Missing values are represented by a period (.)
- SAS has no reserved keywords
- Comments can be inserted in two ways

  * … ;   OR   /* …. */

IMPORTING DATA

Reading Data w/ Import Procedure

- Many ways to get data into SAS – we will only learn import (most practical).
- Good for various types of data files
  - Scans the data file, automatically determines the variable types.
- You can also use the import wizard (Base SAS; more complicated in SAS UE)
  - This is a point and click process, with option to save code for later
  - File -> Import Data

PROC IMPORT - Options

- DBMS= specify the file extension if there is none:
  - CSV (comma-delimited), TAB (tab-delimited), XLS (Excel), ACCESS, DTA (Stata), DLM (other), etc.
- Replace=YES
  - overwrite an existing data set with the one specified in out= option
- Delimiter= specify what delimiter is used; default is a blank space
  - e.g., delimiter='/ '
- Getnames=NO do not get variable names from the first line of input file. Default is YES. In NO, the variables are named Var1, Var2, …

PROC IMPORT - EXAMPLES

```
proc import datafile='Path:\Filename'
      out=data-set;
      run;


proc import datafile =
'O:\Teaching\BERD_computing\lowbwt_LOW.csv'
            out = lowbwt
            dbms = csv;
run;

proc import out=work.HEAL
datafile="'O:\Teaching\BERD_computing\Final_data.sav"
                  DBMS=SAV REPLACE;
run;
```

PROC IMPORT – SAS UE

- Easiest to work with CSV files
- For UE, data and program **must be in shared folder** you created during set-up ("myfolders").

```
proc import datafile =
'/folders/myfolders/SAS_1/lowbwt_LOW.csv'
            out = lowbwt
            dbms = csv;
run;
```

Data Storage

- SAS data sets can be temporary and permanent
  - Temporary - exists only during the current job and is erased when exit SAS

    - one-level name

    - automatically stored in the WORK library

  - Permanent – remains when the job/session is finished

    - two-level name

    - never stored in WORK library

Data Description

- A data set consists a of two parts:
  - A descriptor portion including variable names, type, size, etc. (PROC CONTENTS)
  - A data portion (PROC PRINT)
- Always check your data before performing any stats

EXAMINING DATA ATTRIBUTES

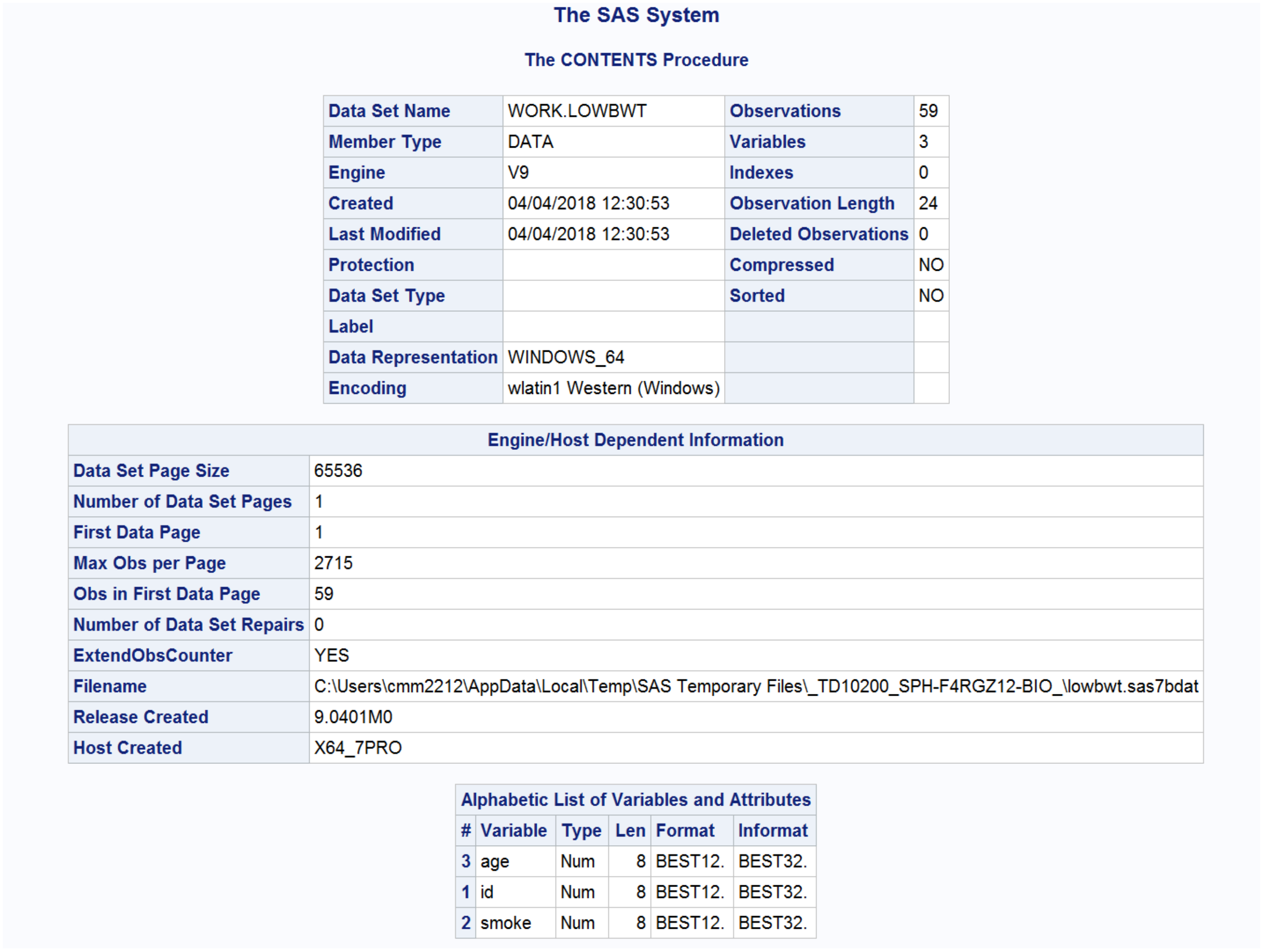
PROC CONTENTS

- Used to examine the descriptor portion of a SAS data set

```
proc contents data=data-name options;
run;
```

- The output displays info about the data set, such as number of variables and their types, number of observations, dates of creation, etc.
- List is in alphabetical order, from uppercase to lowercase

PROC CONTENTS



The SAS System

The CONTENTS Procedure

Data Set Name	WORK.LOWBWT	**Observations**	59
Member Type	DATA	**Variables**	3
Engine	V9	**Indexes**	0
Created	04/04/2018 12:30:53	**Observation Length**	24
Last Modified	04/04/2018 12:30:53	**Deleted Observations**	0
Protection		**Compressed**	NO
Data Set Type		**Sorted**	NO
Label			
Data Representation	WINDOWS_64		
Encoding	wlatin1 Western (Windows)		

Engine/Host Dependent Information	
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	2715
Obs in First Data Page	59
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\Users\cmm2212\AppData\Local\Temp\SAS Temporary Files_TD10200_SPH-F4RGZ12-BIO_\lowbwt.sas7bdat
Release Created	9.0401M0
Host Created	X64_7PRO

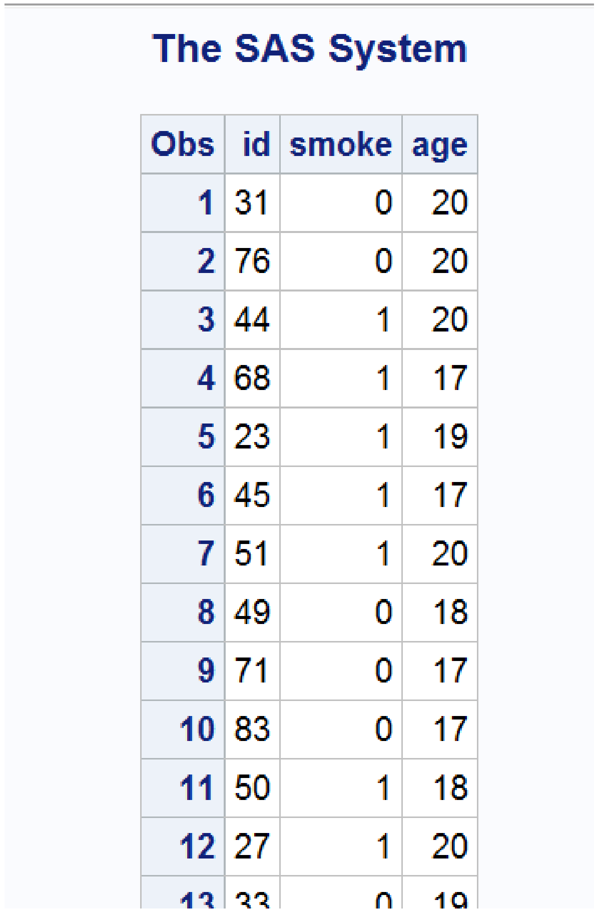
Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat
3	age	Num	8	BEST12.	BEST32.
1	id	Num	8	BEST12.	BEST32.
2	smoke	Num	8	BEST12.	BEST32.

PROC PRINT

- Used to list the data

```
proc print data=data-name;
  run;
```



The SAS System

Obs	id	smoke	age
1	31	0	20
2	76	0	20
3	44	1	20
4	68	1	17
5	23	1	19
6	45	1	17
7	51	1	20
8	49	0	18
9	71	0	17
10	83	0	17
11	50	1	18
12	27	1	20
13	33	0	19

MANIPULATING DATA

SET Statement

- SET statement reads a temporary SAS data set into a new one, so that you can modify it (add new variables, subset, etc.)

```
data new-data-set;
    set old-data-set;
  run;
```

Create/Modify Variables

- The assignment statement can be used to create/modify variables in the DATA step

Example:

```
data new-data-set;
 set old-data-set;
 los = discharge_time-admission_time;
 new_counts = log(counts);
 weight = weight*1.5;
run;
```

- A variety of arithmetical operators and statistical functions
- A new variable becomes automatically the right-most column in the data set

Operators in SAS

Arithmetic operators	Comparison operators	
* multiplication	= eq equal to	^= ne not equal to
+ addition	> gt greater than	>= ge greater than or equal to
- subtraction	< lt less than	<= le less than or equal to
** exponentiation		
/ division		

Delete/Rename Variables

- In data step use:

- KEEP = variable-list
  - tells SAS which variables to keep

- DROP = variable-list
  - tells SAS which variables to drop

- RENAME = (oldvar=newvar)
  - tells SAS to rename certain variables

Delete/Rename Variables

Examples:

```
data new-data-set;
 set old-data-set (KEEP = var1 var2 var3);

proc print data = new-data-set (DROP =
var2);

data new-data-set (RENAME = (var1 = new-var1
var3 = new-var3));
 set new-data-set;
```

- KEEP or DROP?

IF-THEN Statements

- Very useful when you want the assignment statement to apply to *some* observations and *not all*

- General syntax:

```
IF condition THEN action;
```

- You can specify multiple conditions with the keywords AND OR

```
IF condition1 AND condition2 THEN action;
```

IF-THEN/ELSE Statements

- Very useful for grouping observations based on multiple conditions

General syntax:

```
IF condition1 THEN action;
    ELSE IF condition2 THEN action;
    ELSE action;
```

- Note: unless you are sure that your data has no missing values, you should allow for missing values when writing the IF-THEN/ELSE statements

Subsetting a SAS Data Set

- Select observations from one data set by defining selection criteria, usually using IF or WHERE statements
- WHERE and IF play similar roles
  - WHERE can also be used in PROC steps
  - IF is preferred for more complex conditions

```
data new-data-set2;
  set new-data-set;
       where condition; /* or IF condition */
run;
```

Sorting a Data Set

- A data set can be sorted by one or more variables
- In SAS, use PROC SORT:

General syntax:

```
proc sort data=data-set;
  by variable-list;
run;
```

- Overwrites the existing data set
- Sorts in ascending order (default)
- Sorts with respect to the order of variable-list

PROC SORT - Options

- OUT option – creates a new data set containing the sorted version of the original data set

- DESCENDING option – sorts from largest to smallest or Z -> A

```
proc sort data=data-set out=new-data-set;
          by var1 descending var2;
run;
```

- New-data-set is sorted by var1 (ascending) and var2 (descending)

Concatenating Data Sets (Stacking)

- Combine data sets with all or most of the same variables but different observations

data 1
ID
1
4
6

+

data 2
ID
2
5
7

=

combined
ID
1
4
6
2
5
7

Concatenating Data Sets (Stacking)

General syntax:

```
data new-data-set;
  set data-set-1 … data-set-n;
run;
```

- The number of observations in the new data set equals the sum of the observations in the n data sets
- If one of the data sets has a variable not contained in the other, missing values will be added instead

Merging Data Sets

- Useful when you need to combine data from different sources, or at different times

data 1

ID	Age
1	15
2	20
3	18

+

data 2

ID	Age	Weight
1	15	115
2	20	134
6	22	140

=

combined

ID	Age	Weight
1	15	115
2	20	134
3	18	.
6	22	140

Merging Data Sets

General syntax:

```
data new-data-set;
  merge data-set-1 … data-set-n;
   by matching-variable; /* usually ID */
run;
```

- All data sets must be SORTED first by the matching variable
  - The matching variable should be of the same type (numeric/character), and same length
  - If you merge two data sets and they have other variables in common, then the variables from the second data set will overwrite variables with the same name in the first data set

Merging (IN= option)

- Helpful to know which data set an observation comes from
- Creates an "indicator variable" that takes on either the value 0 or 1 depending on whether or not the current observation comes from the input data set
- Make sure that only complete records are output, and create a data set with missing observations from one of the merged data sets

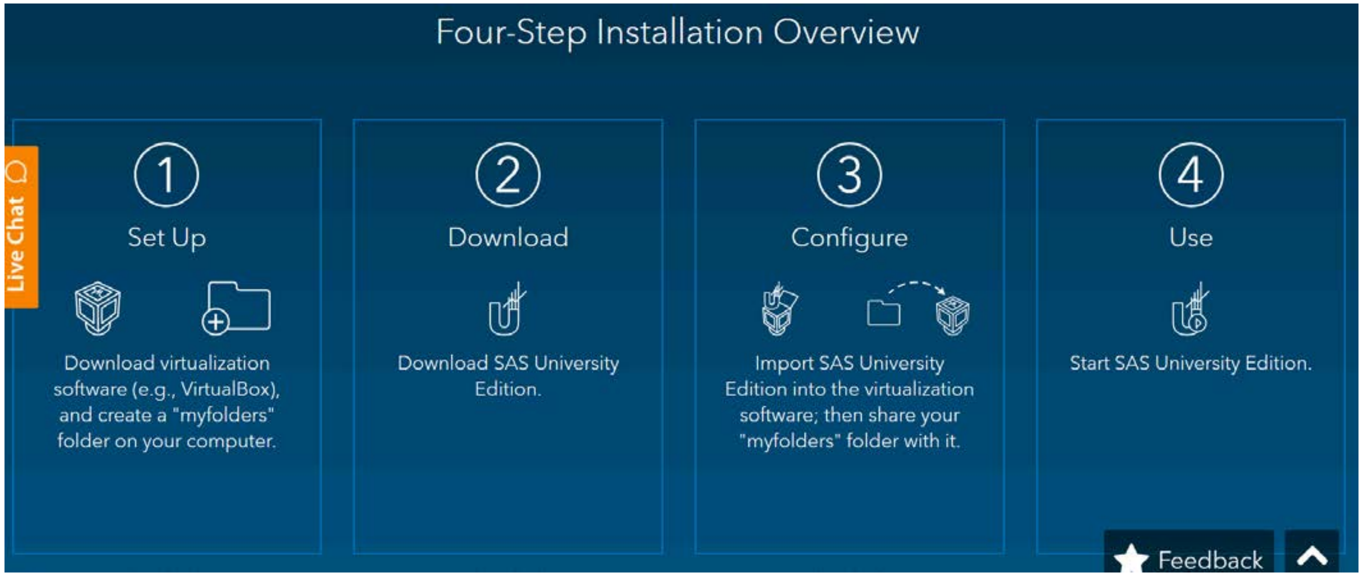
Merging (IN= option)

Example:

```
        data both problem;
         merge data1(in=one) data2(in=two);
            by id;
            if one and two then output both;
          /* if one=1 AND two=1, i.e., same ID present in
both data sets */
                              else output problem;
        run;
```

SAS University Edition

- https://www.sas.com/en_us/software/university-edition.html (FREE and works on Macs!)
- http://localhost:10080


Four-Step Installation Overview

1 Set Up
Download virtualization software (e.g., VirtualBox), and create a "myfolders" folder on your computer.

2 Download
Download SAS University Edition.

3 Configure
Import SAS University Edition into the virtualization software; then share your "myfolders" folder with it.

4 Use
Start SAS University Edition.

APPLICATION

Application

- A study was conducted to identify risk factors for low infant birth weight using data from 189 live births at Bay State Medical Center in Massachusetts. Low birthweight was defined as a <2500grams.

- We have one data set for low birthweight-babies (lowbwt_LOW.csv) and another for normal birthweight babies (lowbwt_Normal.csv).
  - id = ID number of infant
  - smoke = smoking during pregnancy = 1 if yes; 0 if no
  - age = mother's age in years

- We have a separate dataset with data on # of visits (lowbwt_ADMIN.csv).
  - id = ID number of infant
  - visits = number of physician visits during 1st trimester = 0 if none; 1 if one; 2 if two or more

Thank you!

BERD EDU link:

http://irvinginstitute.columbia.edu/resources/biostat_educational_initiatives.html

Acknowledgements: Cody Chiuzan, PhD