SAS - Intro
Statistical Analysis System
is a computer software
system for the analysis
of data.

Data - letters or numbers, organized in order to represent information, e.g. the values of a variable

Variable - a characteristic or attribute for which data is collected
EX: 68 inches is a data value for the variable height.

Observation - A set of data values for a single individual.

Note: An individual does not have to be a person.

Data Set - A collection of observations. Can think of a dataset as a rectangular array (or matrix) of information. Each row (or group of rows) corresponds to one observation.
Each column corresponds to a single variable.

General Structure of a Dataset

`Var1 Var2 Var3 ... Var k`

`obs1 obs2 obs3 .. obs n`

A raw data set can be stored as a text file (ASCII).

Data can also be stored in other "structures" e.g. Excel file, Access file, Oracle file
**Codebook** - a description of the variables, variable names, variable types, and allowable values for the observations in a data set.

<table>
<thead>
<tr>
<th>Ex: Variable Name</th>
<th>Variable Type</th>
<th>Desc</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>num</td>
<td>Age (yrs)</td>
<td>0 - 116</td>
</tr>
<tr>
<td>Weight</td>
<td>num</td>
<td>Current Weight in lbs</td>
<td>80 - 500</td>
</tr>
<tr>
<td>Gender</td>
<td>char</td>
<td>gender</td>
<td>M, F</td>
</tr>
<tr>
<td>Marital</td>
<td>char</td>
<td>Marital Status</td>
<td>1=Single, 2=Married, 3=Divorced, 4=Other</td>
</tr>
</tbody>
</table>
Rules for naming variables and data sets:

1) 1-32 characters long
2) must start with letter
3) can use letters A-Z, numbers 0-9, underscore _
4) names are case insensitive
   AGE is same as age
5) try to choose names which have meaning
SAS dataset - a dataset created by SAS, typically from some source external to SAS.

SAS dataset contains:

1) data

2) "descriptive" info about the variables (names, types, formats, etc.) sometimes called "meta-data"

3) indices

SAS datasets can be temporary or permanent.
Getting Your Data Into SAS

SAS programs are comprised of SAS statements.

SAS statements instruct SAS to perform a task or activity, e.g., create a dataset, create a new variable, modify value of a variable, execute a statistical procedure.

SAS statements consist of:
- Keywords - SAS recognizes
- User-supplied Info -
  Variable names
  Labels
  Titles
SAS statements

1) may begin in any column
2) may end in any column
3) may use more than 1 line
4) two (or more) statements may appear on the same line
5) contain (at least) one SAS keyword
6) may contain variable names, options, parameters, dataset names
7) ALWAYS end with semicolon ;

SAS program consist of two basic actions (called steps)
Data step
Procedure step
Can have several data steps and procedure steps in a single SAS program

The Data Step

Used to get the data into SAS and create a SAS data set.

Data statement

Input statement

Datalines statement (CORDS)

Infile statement

LIBNAME statement
DATA Statement
Is used to create a SAS dataset.
Has the form

DATA datasetname (options);

Ex: DATA mydata;

Ex: DATA error (DROP= b_day);

The datasetname must conform to standard naming conventions.

The options include:

DROP =
KEEP =
RENAME =
Label =
Simplest form of DATA stmt is
DATA;

No dataset name was specified; so SAS will create a dataset name (data1, data2, etc.)
Datasets can have a 2-level name:
library.member

Ex: DATA ab1.names;
The dataset "names" will be stored in the library "ab1" (assuming we have identified library ab1 using a LIBNAME statement).
If the user specifies a 1-level dataset name, the dataset is stored in a library named WORK. The dataset is a temporary SAS dataset, and will disappear when the user exits SAS.

Ex: DATA mine;

If the user wants to store a permanent copy of a SAS dataset, the user must:

1) Create (or identify) a library (which is just a folder or a directory)
2) Tell SAS where the library is located (use a LIBNAME statement)

3) Use a two-level dataset name. The first level "points to" the permanent SAS library. The second level name is the name of the permanent SAS data set.

*EX: LIBNAME ab1 "path to directory" ;
DATA ab1.perm ;*
INPUT statement

1) Allows the user to read "external" data into SAS.

2) Assigns names to the variables and describes the location of their values in each data record. So the programmer must know the structure of the external data set.

3) If an error is made when specifying the location of data, subsequent SAS procedures may produce incorrect results.

GIGO
4) Programmer can indicate whether variables contain numeric values, character values, or dates. (Actually, date values are stored as numeric values).

Many different options/forms of the INPUT statement.

**LIST INPUT**

List INPUT is sometimes called "free format" input. This type of INPUT stmt can be used if all following conditions are satisfied:
1) Each data value on a data record is separated from the next data value by at least one blank space (or some other delimiter, e.g., a comma).

2) Values of character variables consist of 8 characters or fewer.

3) Values of character variables do not contain blank(s).

4) Numeric values contain all necessary decimal points.

5) The dataset contains no missing values; or missing values are represented by periods.
IF ALL 5 conditions are satisfied, we do not have to specify the actual column locations of the values of each variable.

GENERAL FORM

INPUT input list specifications;

SAS assumes that a variable is a numeric variable, unless we tell SAS otherwise. To indicate that a variable contains character information place a $ after its name in the INPUT statement.
EX: `INPUT name $ age height sex $;`

This `INPUT` statement tells SAS that:

1) Each record contains 4 variables.

2) No missing values in the data set.

3) The first variable (name) contains character data.

4) The fourth variable (sex) contains character data.

5) The variables age and height contain numeric data.
Note: OK to continue the INPUT statement onto the next line in SAS program.

Missing Values in a Record

When using LIST INPUT, if any variables have a missing value, the missing value must be indicated by using a period for that missing value.
Fixed Format (or Column) INPUT

If data values of each variable are located in the same columns on every record, we can use column input. We must tell SAS which columns contain data for which variables. Specify the column locations in the INPUT statement.

EX: INPUT name $ 1-10
     sex $ 12  age 14-15
     height 17-18  weight 19-21;
Note: When using column input, there is no need for a blank space (or other delimiter) between data values on a record.

**READING SELECTED VARIABLES**

If a data set contains data that we do not need to read, we can "skip over" that data without reading it.

**Example:**

```plaintext
INPUT name $ 1-10
age 14-15;
```
We can INPUT the data values in any order we desire:

**Ex:**  INPUT name $1-10$

    age 14-15

    weight 19-21

    sex $12$ ;
DATA LINES statement
1) Follows the DATA and INPUT statements (and any other assignment statements)
2) Indicates the beginning of "In-stream" data
3) Data records immediately follow the DATA LINES statement
4) A single semi-colon (;) on a data record indicates the end of the In-Stream dataset
INFILE statement

1) Used to indicate location of raw data external file

2) Also can be used to specify options for reading raw data.

`INFILE "path/filename";`

`EX: INFFILE " /home/abbillings/public_html/HTWT.DTA";`

When used with DATALINES filename, can be used to read files with comma (or other) delimiters.
The `INFILE` statement **MUST** be placed before the associated `INPUT` statement in a SAS program (after the `DATA` stmt).

```
DATA    ;
INFILE  ;
INPUT   ;
```
COMMA - Delimited Data in LIST INPUT

We can use the INFILE statement to indicate that a file has data values separated by commas (or other delimiters) by using the DLM option.

EX: DATA ~;
INFILE DATALINES DLM="'','";
INPUT ~ ~ ~ ~;
DATALINES;
    data goes here
    separated by commas
;

Can choose any delimiter we want

EX: INFILE DATALINES DLM='/';
the data values are separated
by the delimiter /
John/23/49/251

The DSD option on the INFILE
statement tells SAS to treat
two consecutive delimiters as
containing a missing value.
IF ODS is used without specifying DLM=
SAS assumes that commas are being used as delimiters.

EX: INFILE DATALINES ODS;
SAS assumes commas are delimiters
two consecutive commas indicates a missing value

John,, 43, 241
↑
2nd variable's value is missing
Can use DLM and DSD together in an INFILE statement

Ex: INFILE DATALINES
    DLM= '/', DSD;

The data record
    John/23//241

means the third variable has a missing value.