Introduction to SAS programming

Summer OFE Workshop 2021

Haiyong Liu

Department of Economics

What Is SAS?

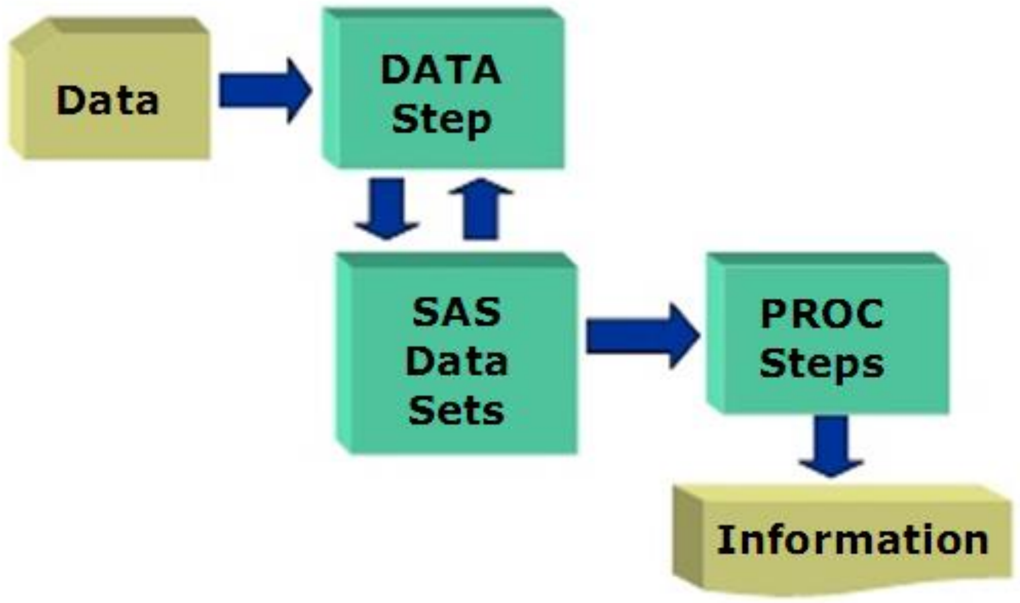
- SAS is a collection of modules that are used to process and analyze data.
- It began in the late ’60s and early ’70s as a statistical package (Statistical Analysis System).
- SAS is also an extremely powerful, general-purpose programming language.
- In recent years, it has been enhanced to provide state-of-the-art data mining tools and programs for Web development and analysis.

Data-Driven Tasks

The functionality of the SAS System is built around the four data-driven tasks common to virtually any application:

- 1. data access:
  - addresses the data required by the application
- 2. data management:
  - shapes data into a form required by the application
- 3. data analysis:
  - summarizes, reduces, or otherwise transforms raw data into meaningful and useful information
- 4. data presentation:
  - communicates information in ways that clearly demonstrate its significance

An Overview of SAS Data Processing



DATA steps are used to create SAS data sets.
PROC steps are used to process SAS data sets.

Why SAS?

- Able to process large data set(s)
- Easy to cope with multiple variables
- Able to track all the operations on the data set(s)
- Generate systematic output
    - Summary statistics
    - Graphs
    - Regression results
- Most government agencies and private sectors use SAS

Roadmap

- Thinking in “SAS”
- Basic rules
- Read in data
- Data cleaning commands
- Summary statistics
- Combine two or more datasets
- Hypothesis testing
- Regression

Thinking in “SAS”

- What is a program?
  - Algorithm, recipe, set of instructions
- How is programming done in SAS?
  - SAS is like programming in any language:
    - Step by step instructions
    - Can create your own routines to process data
    - Most instructions are set up in a logical manner
  - SAS is NOT like other languages:
    - Some syntax is peculiar to SAS
    - Written specifically for statistics so it isn’t all-purpose
    - Canned processes that you cannot edit nor can you see the code

Thinking in “SAS”

- Creating a program
    - What is your problem? (take project 3 as an example)
    - How can you find a solution?
    - What steps need to be taken to find an answer?
        - Do I need to read in data?
            - What variables do I need?
            - Where is the data?
            - What format is the data in?
        - How do I need to clean the data?
            - Are there outliers?
            - Are there any unexpected values in the data?
        - How do I need to transform the data?
            - Are the variables in the form that I need?

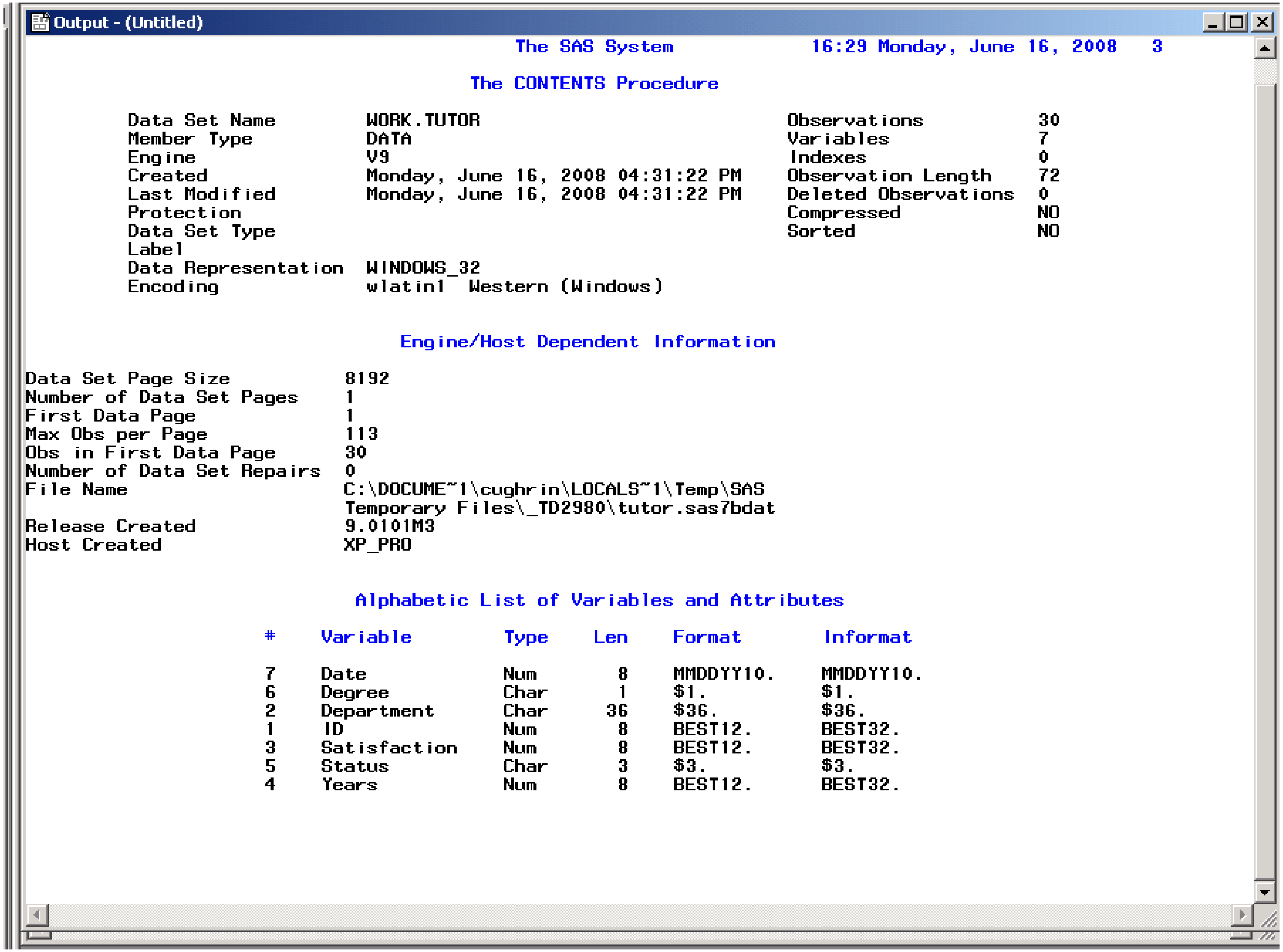
SAS Data Sets

Two Sections

Descriptor Section

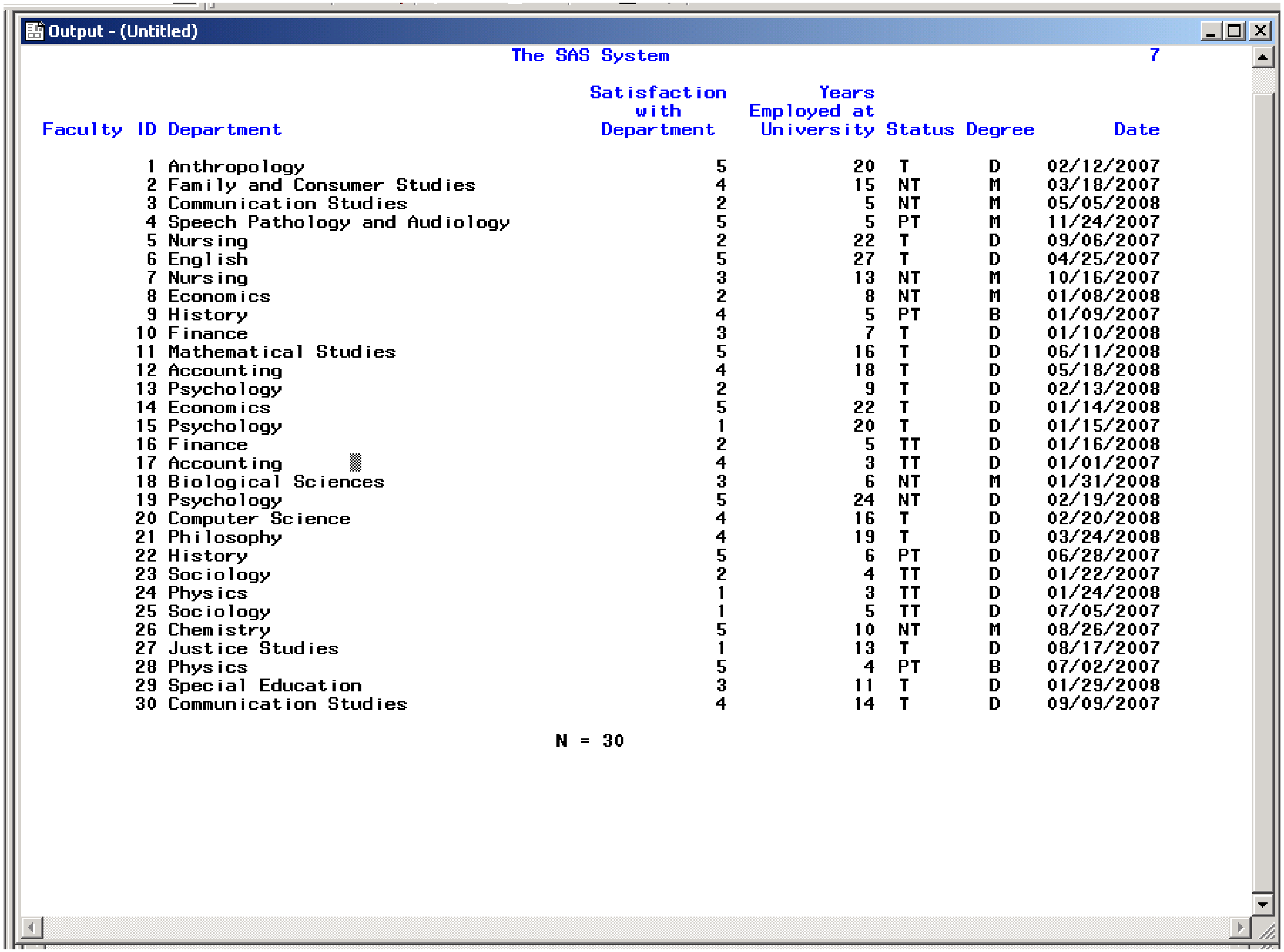
Data Section

Data Set Descriptor Section



```
                                     The SAS System         16:29 Monday, June 16, 2008   3

                                The CONTENTS Procedure

          Data Set Name          WORK.TUTOR                                Observations          30
          Member Type            DATA                                      Variables             7
          Engine                 V9                                        Indexes               0
          Created                Monday, June 16, 2008 04:31:22 PM         Observation Length    72
          Last Modified          Monday, June 16, 2008 04:31:22 PM         Deleted Observations  0
          Protection                                                       Compressed            NO
          Data Set Type                                                    Sorted                NO
          Label
          Data Representation    WINDOWS_32
          Encoding               wlatin1  Western (Windows)


                            Engine/Host Dependent Information

Data Set Page Size            8192
Number of Data Set Pages      1
First Data Page               1
Max Obs per Page              113
Obs in First Data Page        30
Number of Data Set Repairs    0
File Name                     C:\DOCUME~1\cughrin\LOCALS~1\Temp\SAS
                              Temporary Files\_TD2980\tutor.sas7bdat
Release Created               9.0101M3
Host Created                  XP_PRO


                        Alphabetic List of Variables and Attributes

                        #    Variable        Type    Len    Format       Informat

                        7    Date            Num       8    MMDDYY10.    MMDDYY10.
                        6    Degree          Char      1    $1.          $1.
                        2    Department      Char     36    $36.         $36.
                        1    ID              Num       8    BEST12.      BEST32.
                        3    Satisfaction    Num       8    BEST12.      BEST32.
                        5    Status          Char      3    $3.          $3.
                        4    Years           Num       8    BEST12.      BEST32.
```

SAS Data Section



```
                                    The SAS System                                  7

                                                 Satisfaction      Years
                                                     with       Employed at
Faculty ID Department                             Department     University Status Degree        Date

         1 Anthropology                                    5             20  T        D     02/12/2007
         2 Family and Consumer Studies                     4             15  NT       M     03/18/2007
         3 Communication Studies                           2              5  NT       M     05/05/2008
         4 Speech Pathology and Audiology                  5              5  PT       M     11/24/2007
         5 Nursing                                         2             22  T        D     09/06/2007
         6 English                                         5             27  T        D     04/25/2007
         7 Nursing                                         3             13  NT       M     10/16/2007
         8 Economics                                       2              8  NT       M     01/08/2008
         9 History                                         4              5  PT       B     01/09/2007
        10 Finance                                         3              7  T        D     01/10/2008
        11 Mathematical Studies                            5             16  T        D     06/11/2008
        12 Accounting                                      4             18  T        D     05/18/2008
        13 Psychology                                      2              9  T        D     02/13/2008
        14 Economics                                       5             22  T        D     01/14/2008
        15 Psychology                                      1             20  T        D     01/15/2007
        16 Finance                                         2              5  TT       D     01/16/2008
        17 Accounting                                      4              3  TT       D     01/01/2007
        18 Biological Sciences                             3              6  NT       M     01/31/2008
        19 Psychology                                      5             24  NT       D     02/19/2008
        20 Computer Science                                4             16  T        D     02/20/2008
        21 Philosophy                                      4             19  T        D     03/24/2008
        22 History                                         5              6  PT       D     06/28/2007
        23 Sociology                                       2              4  TT       D     01/22/2007
        24 Physics                                         1              3  TT       D     01/24/2008
        25 Sociology                                       1              5  TT       D     07/05/2007
        26 Chemistry                                       5             10  NT       M     08/26/2007
        27 Justice Studies                                 1             13  T        D     08/17/2007
        28 Physics                                         5              4  PT       B     07/02/2007
        29 Special Education                               3             11  T        D     01/29/2008
        30 Communication Studies                           4             14  T        D     09/09/2007

                                     N = 30
```

Attributes of Variables

- Name
    - e.g. Status
- Type
    - Numeric or Character
    - e.g. Status in this example is character (T, TT, PT, or NTT) and Satisfaction is numeric (1 to 5).

SAS Data Set Terminology

- *Variables* – columns in a SAS data set.
- *Observations* – rows in a SAS data set.
- *Numeric Data* – values that are treated as numeric and may include 8 bytes of floating storage for 16 to 17 significant digits.
- *Character Data* – non numeric data values such as letters, numbers, special characters, and blanks. May be stores with a length of 1 to 32, 767 bytes. One byte is equal to one character.

SAS Data Set and Variable Name Criteria

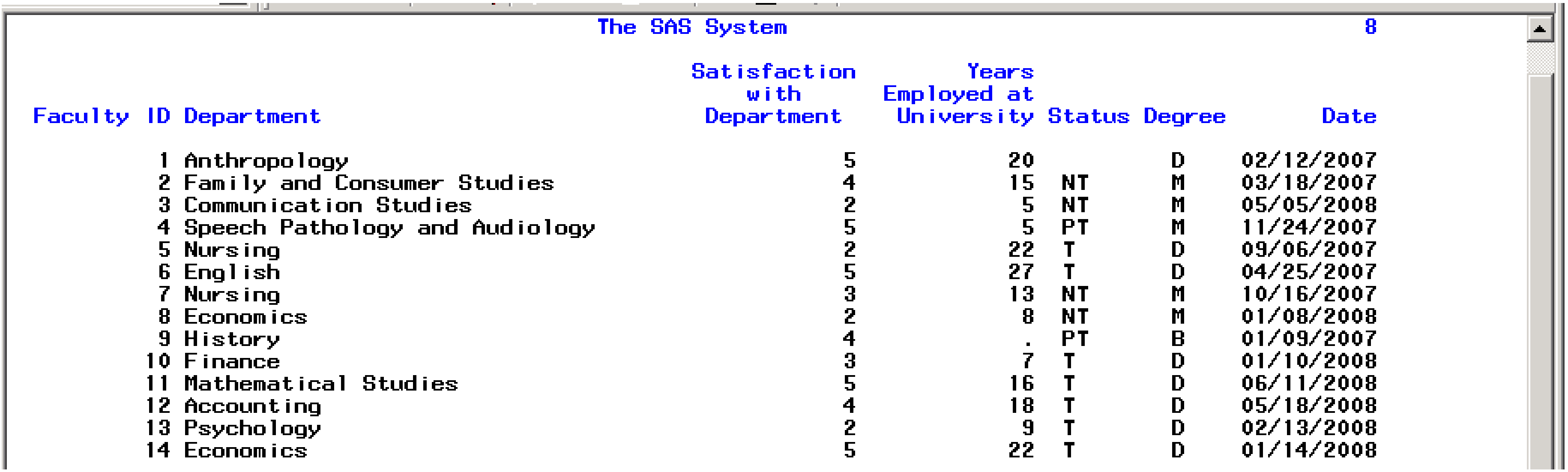
- Can be 32 characters long.
- Can be uppercase, lowercase, or a mixture of the cases.
- Are not case sensitive
- Cannot start with number and cannot contain special characters or blanks.
- Must start with a letter or underscore.

SAS Dates

- Dates are treated as special kind of numeric data.
  - They are the number of days since January 1st, 1960. January 1st 1960 is the 0 point. SAS dates can go back to 1582 (Gregorian Calendar) and forward to the year 20000.
  - Dates are displayed using a format. There are a number of different date formats supported by SAS.
- Time is scored as the number of seconds since midnight. SAS date time is the number of seconds since January 1st, 1960.

Missing Data in SAS

- Missing values are valid values.
  - For character data, missing values are displayed as blanks.
  - For numeric data, missing values are displayed as periods.



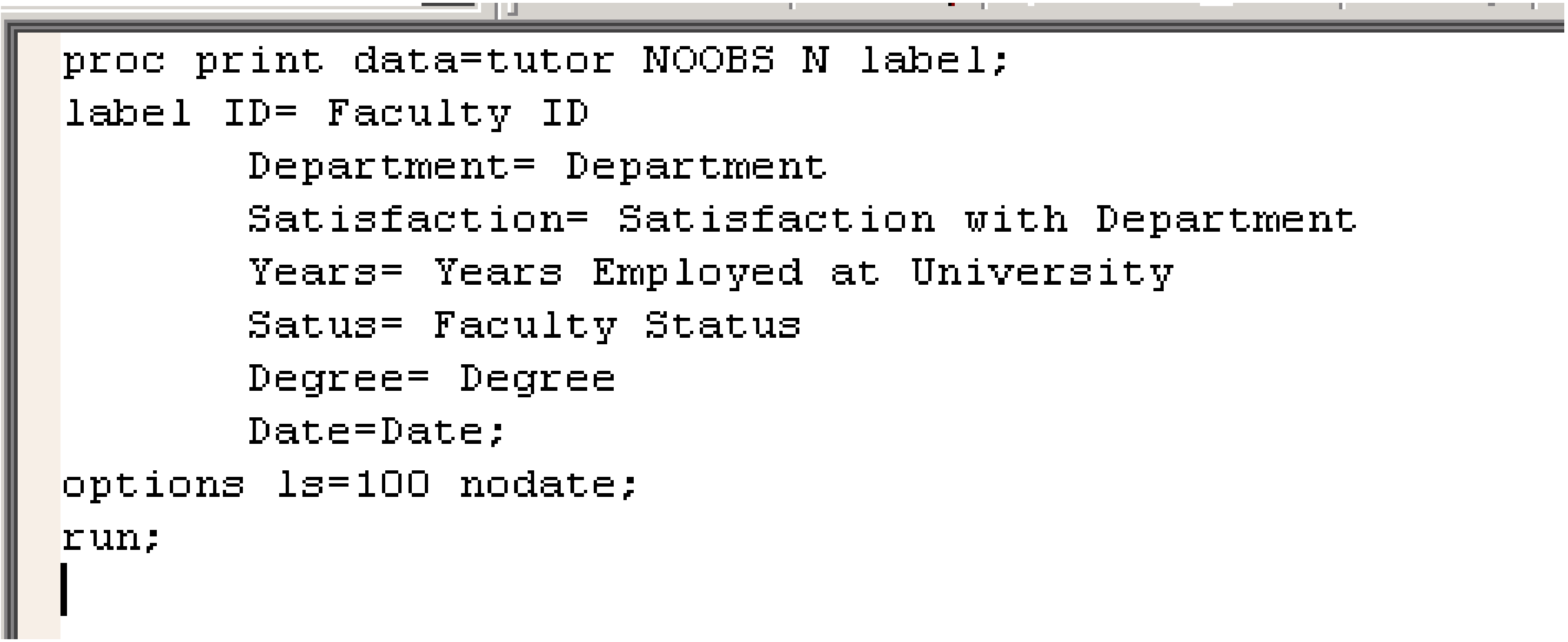
```
                                         The SAS System                                          8

                                                Satisfaction        Years
                                                    with      Employed at
Faculty ID Department                            Department    University Status Degree       Date

         1 Anthropology                                    5           20          D    02/12/2007
         2 Family and Consumer Studies                     4           15  NT      M    03/18/2007
         3 Communication Studies                           2            5  NT      M    05/05/2008
         4 Speech Pathology and Audiology                  5            5  PT      M    11/24/2007
         5 Nursing                                         2           22  T       D    09/06/2007
         6 English                                         5           27  T       D    04/25/2007
         7 Nursing                                         3           13  NT      M    10/16/2007
         8 Economics                                       2            8  NT      M    01/08/2008
         9 History                                         4            .  PT      B    01/09/2007
        10 Finance                                         3            7  T       D    01/10/2008
        11 Mathematical Studies                            5           16  T       D    06/11/2008
        12 Accounting                                      4           18  T       D    05/18/2008
        13 Psychology                                      2            9  T       D    02/13/2008
        14 Economics                                       5           22  T       D    01/14/2008
```

SAS Syntax

SAS Syntax

- Statements in SAS are like sentences. The punctuation though is a semicolon(;)rather than a period (.)
- Most Statements (but not all) start with an identifying key word (e.g. proc, data, label, options, format…)
- Statements are strung together into sections similar to paragraphs. These paragraphs in a Windows OS are ended with the word “run” and a semicolon.

Example of SAS Syntax

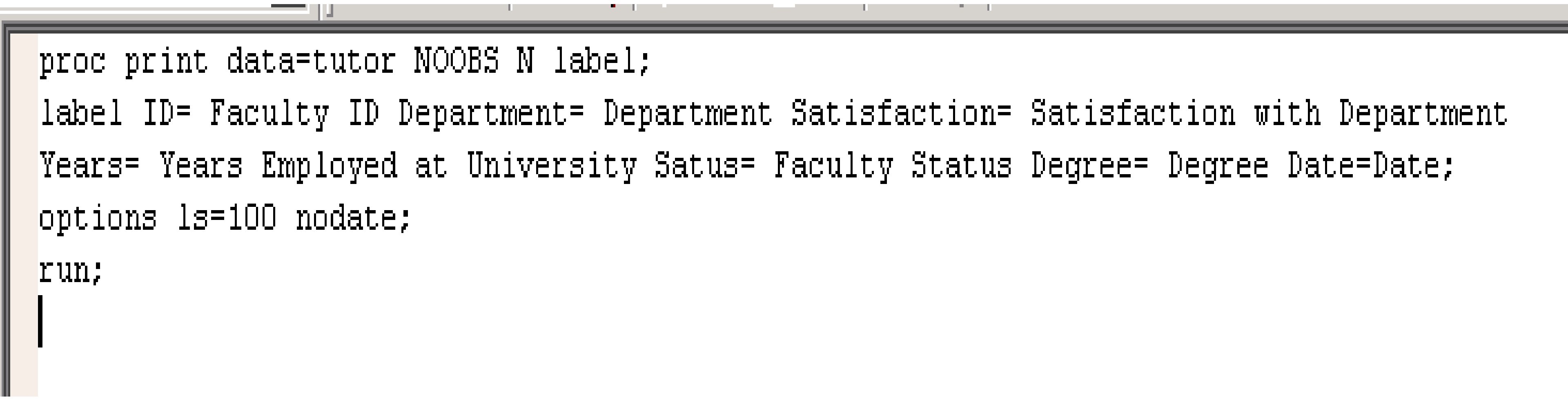


```
proc print data=tutor NOOBS N label;
label ID= Faculty ID
       Department= Department
       Satisfaction= Satisfaction with Department
       Years= Years Employed at University
       Satus= Faculty Status
       Degree= Degree
       Date=Date;
options ls=100 nodate;
run;
```

SAS Syntax Rules

- SAS statements are format free.
- One or more blanks or special characters are used to separate words.
- They can begin and end in any column.
- A single statement can span multiple lines.
- Several statements can be on the same line.

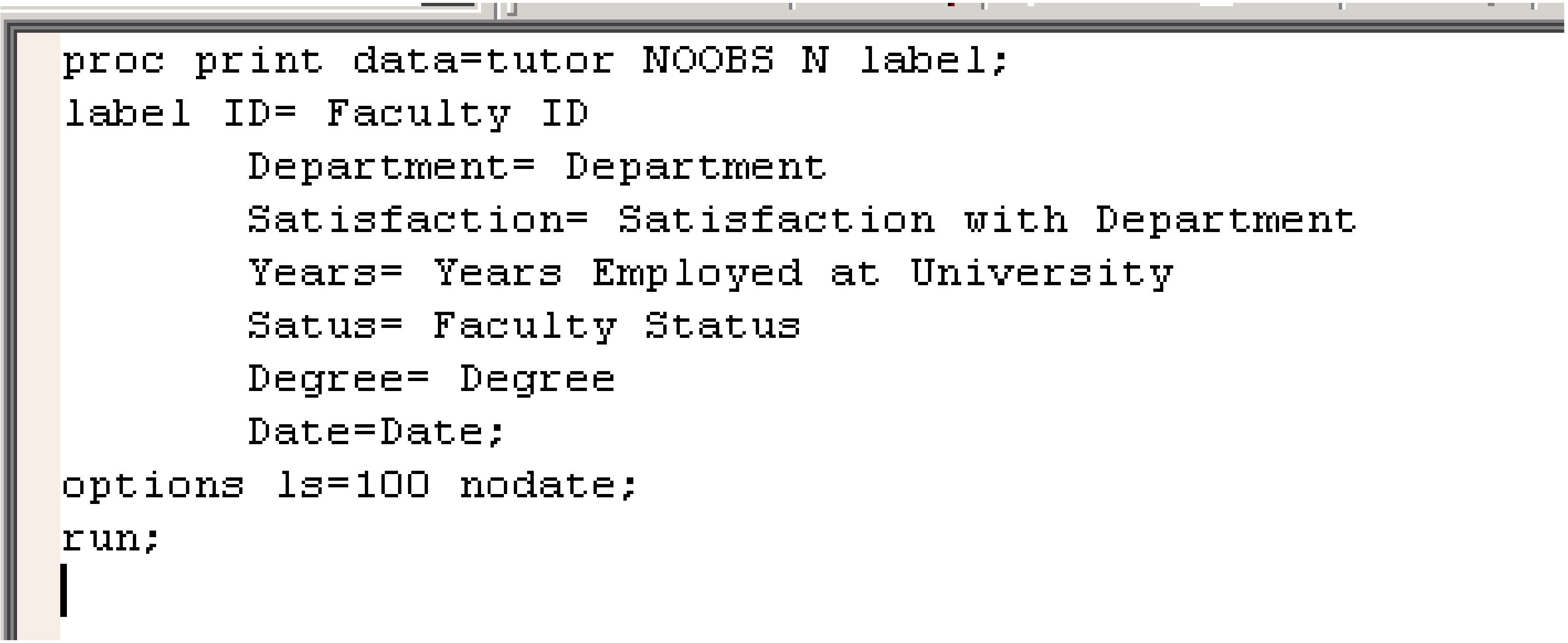
Example of SAS Free Format



```
proc print data=tutor NOOBS N label;
label ID= Faculty ID Department= Department Satisfaction= Satisfaction with Department
Years= Years Employed at University Satus= Faculty Status Degree= Degree Date=Date;
options ls=100 nodate;
run;
```

Using the free-format Syntax rules of SAS though can make it difficult for others (or you) to read your program. This is akin to writing a page of text with little attention to line breaks. You may still have Capital letters and periods, but where a sentence begins and ends may be a bit confusing.

Example of SAS Formatted



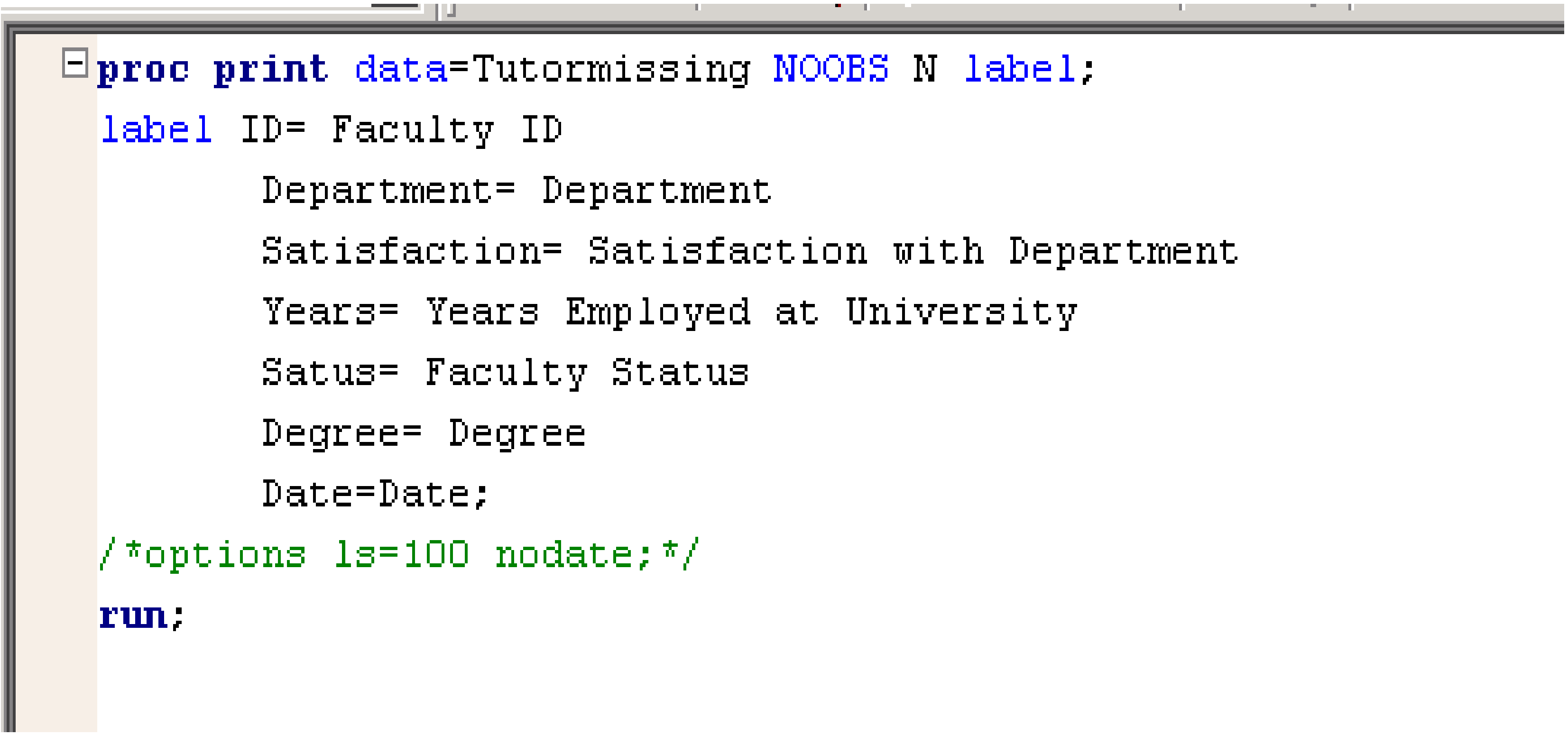
```
proc print data=tutor NOOBS N label;
label ID= Faculty ID
       Department= Department
       Satisfaction= Satisfaction with Department
       Years= Years Employed at University
       Satus= Faculty Status
       Degree= Degree
       Date=Date;
options ls=100 nodate;
run;
```

Using the free-format Syntax rules of SAS though can make it difficult for others (or you) to read your program. This is akin to writing a page of text with little attention to line breaks. You may still have capital letters and periods, but where a sentence begins and ends may be a bit confusing. Isn’t this paragraph a bit easier to read?

SAS Comments

- Type /* to begin a comment.
- Type your comment text.
- Type */ to end the comment.
- Or, type an * at the beginning of a line. Everything between the * and the ; will be commented.
  - e.g. *infile ‘tutor.dat’;
- Alternatively, highlight the text that you would like to comment and use the keys Ctrl / to comment the line. To uncomment a line, highlight and use the Ctrl Shift / keys.

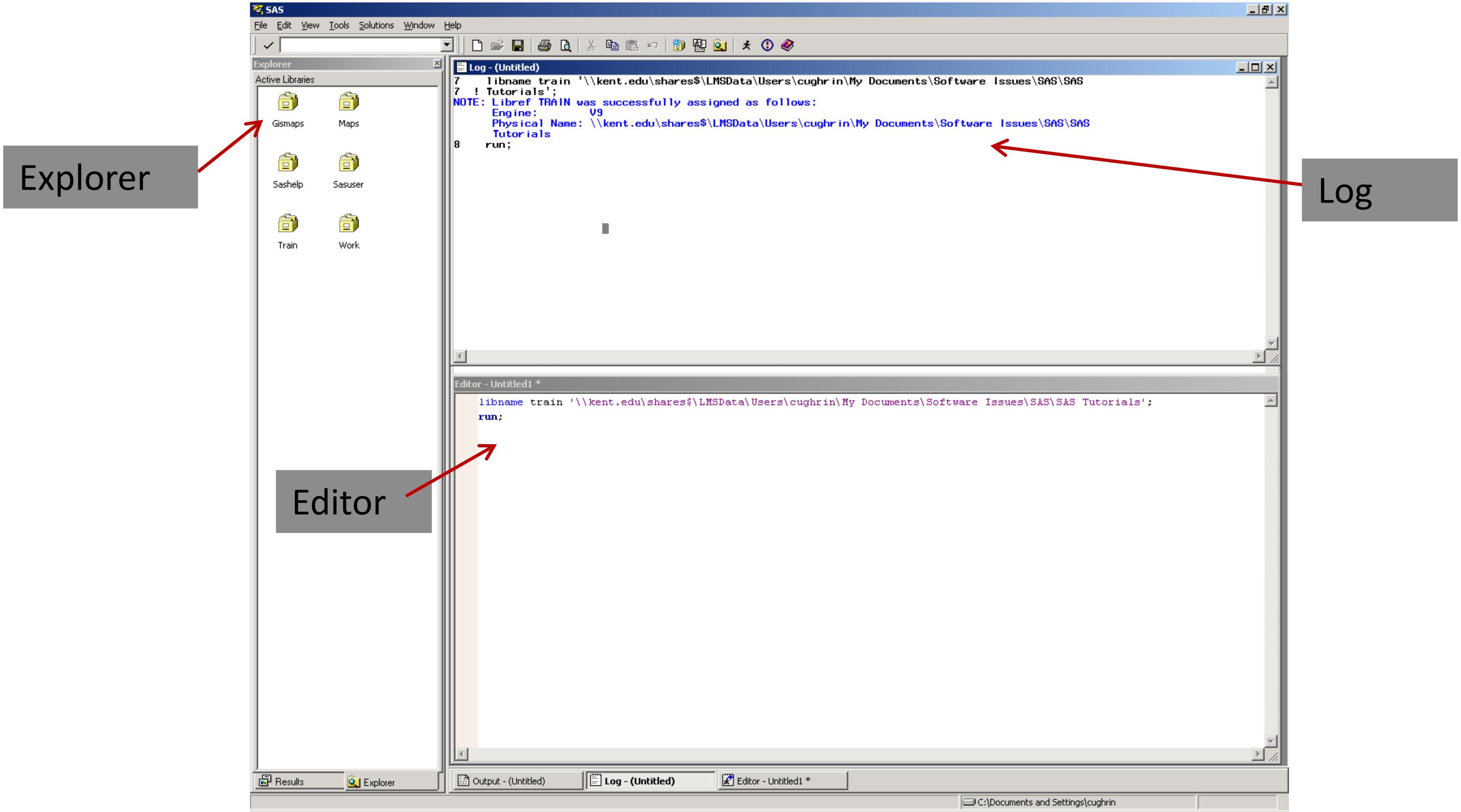
SAS Comments



```
proc print data=Tutormissing NOOBS N label;
 label ID= Faculty ID
       Department= Department
       Satisfaction= Satisfaction with Department
       Years= Years Employed at University
       Satus= Faculty Status
       Degree= Degree
       Date=Date;
 /*options ls=100 nodate;*/
 run;
```

SAS Windows

SAS Windows



Explorer

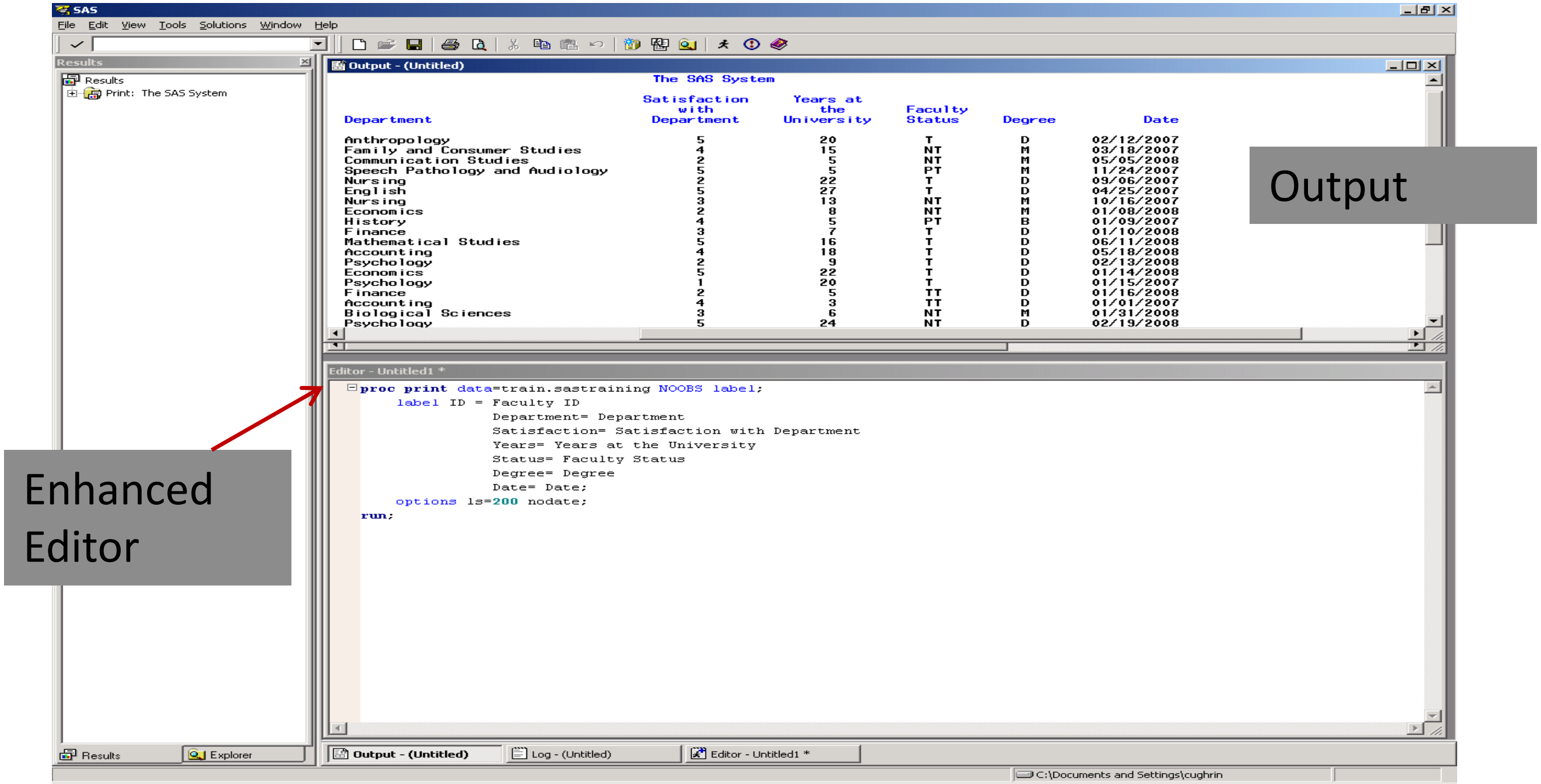
Log

Editor

```
7     libname train '\\kent.edu\shares$\LMSData\Users\cughrin\My Documents\Software Issues\SAS\SAS
7   ! Tutorials';
NOTE: Libref TRAIN was successfully assigned as follows:
      Engine:        V9
      Physical Name: \\kent.edu\shares$\LMSData\Users\cughrin\My Documents\Software Issues\SAS\SAS
      Tutorials
8     run;
```

```
libname train '\\kent.edu\shares$\LMSData\Users\cughrin\My Documents\Software Issues\SAS\SAS Tutorials';
run;
```

Enhanced Editor Window



```
                                        The SAS System

                                      Satisfaction      Years at
                                          with             the        Faculty
Department                             Department     University      Status       Degree          Date

Anthropology                                5              20           T            D          02/12/2007
Family and Consumer Studies                 4              15           NT           M          03/18/2007
Communication Studies                       2               5           NT           M          05/05/2008
Speech Pathology and Audiology              5               5           PT           M          11/24/2007
Nursing                                     2              22           T            D          09/06/2007
English                                     5              27           T            D          04/25/2007
Nursing                                     3              13           NT           M          10/16/2007
Economics                                   2               8           NT           M          01/08/2008
History                                     4               5           PT           B          01/09/2007
Finance                                     3               7           T            D          01/10/2008
Mathematical Studies                        5              16           T            D          06/11/2008
Accounting                                  4              18           T            D          05/18/2008
Psychology                                  2               9           T            D          02/13/2008
Economics                                   5              22           T            D          01/14/2008
Psychology                                  1              20           T            D          01/15/2007
Finance                                     2               5           TT           D          01/16/2008
Accounting                                  4               3           TT           D          01/01/2007
Biological Sciences                         3               6           NT           M          01/31/2008
Psychology                                  5              24           NT           D          02/19/2008
```

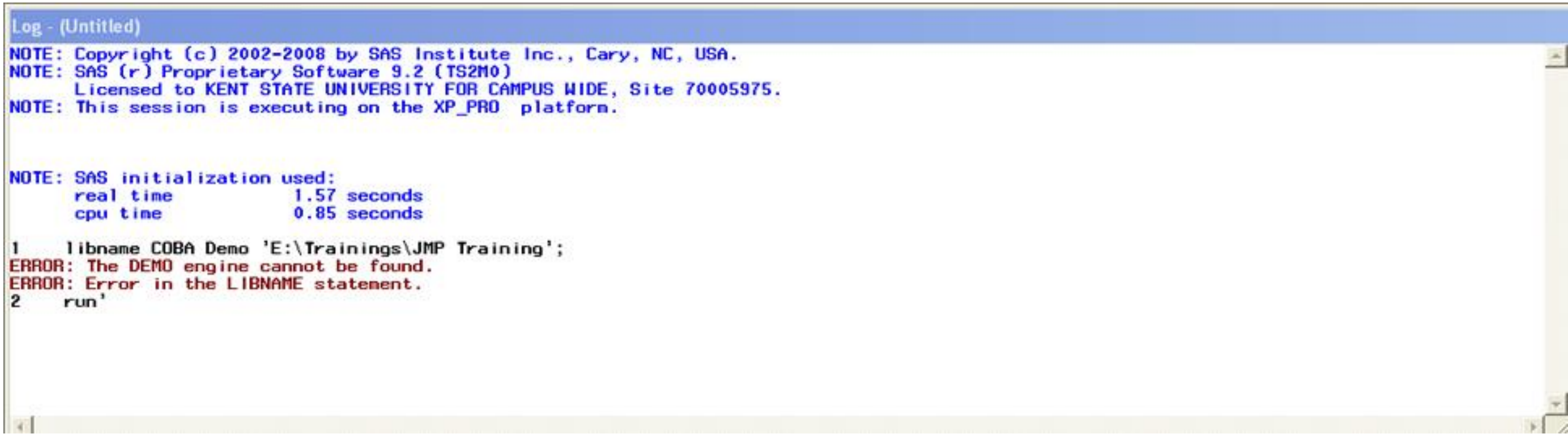
```
proc print data=train.sastraining NOOBS label;
   label ID = Faculty ID
              Department= Department
              Satisfaction= Satisfaction with Department
              Years= Years at the University
              Status= Faculty Status
              Degree= Degree
              Date= Date;
   options ls=200 nodate;
run;
```

Output

Enhanced Editor

- Your program script appears in this window.
- You can either bring it in from a file or type the program right into the window.
- Once the program is in the window, you can Click Submit (or the running guy).

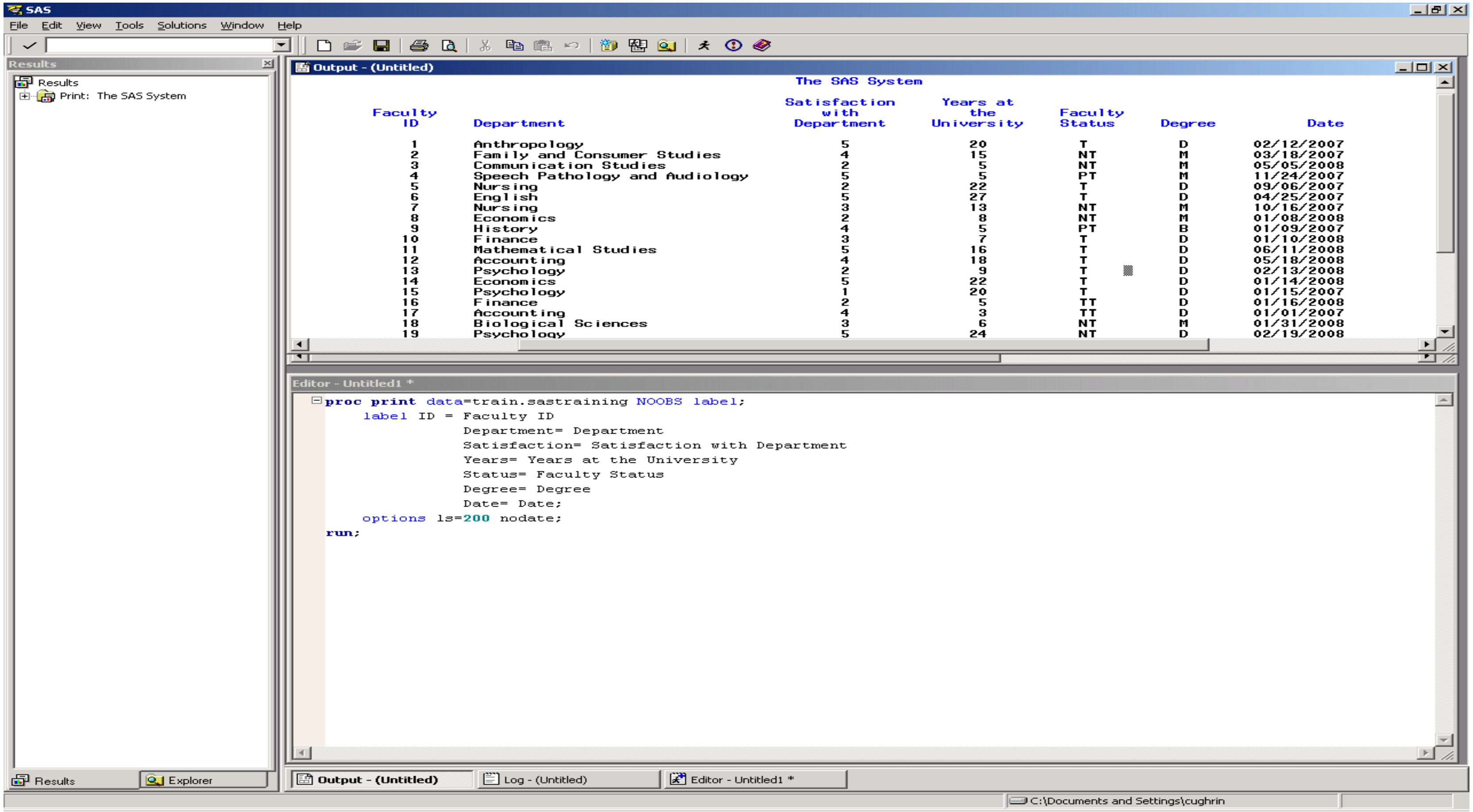
SAS Log



```
NOTE: Copyright (c) 2002-2008 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.2 (TS2M0)
      Licensed to KENT STATE UNIVERSITY FOR CAMPUS WIDE, Site 70005975.
NOTE: This session is executing on the XP_PRO  platform.



NOTE: SAS initialization used:
      real time           1.57 seconds
      cpu time            0.85 seconds

1     libname COBA Demo 'E:\Trainings\JMP Training';
ERROR: The DEMO engine cannot be found.
ERROR: Error in the LIBNAME statement.
2     run'
```

- SAS Log provides a “blow by blow” account of the execution of your program. It includes how many observations were read and output, as well as, errors and notes.
- Note the errors in red.

Output Window

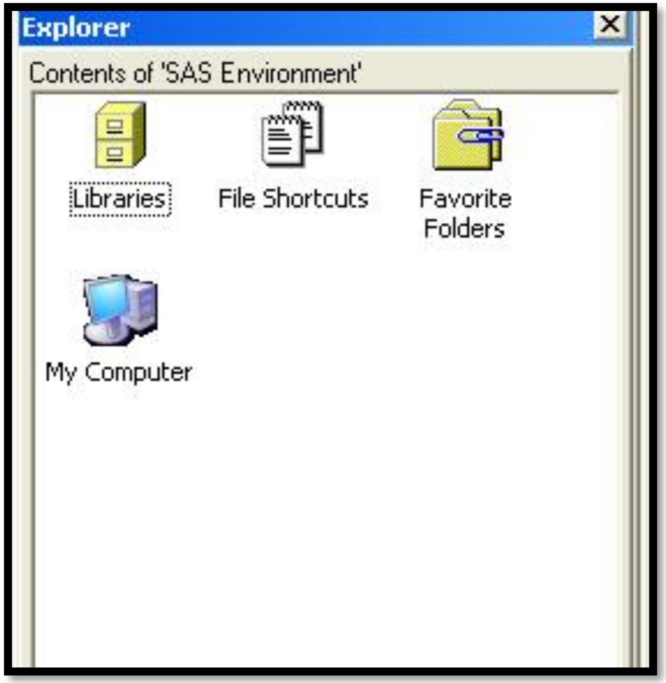


```
                                                The SAS System

                                                  Satisfaction     Years at
     Faculty                                          with           the         Faculty
       ID      Department                          Department     University     Status     Degree          Date

        1      Anthropology                             5             20          T           D         02/12/2007
        2      Family and Consumer Studies              4             15          NT          M         03/18/2007
        3      Communication Studies                    2              5          NT          M         05/05/2008
        4      Speech Pathology and Audiology           5              5          PT          M         11/24/2007
        5      Nursing                                  2             22          T           D         09/06/2007
        6      English                                  5             27          T           D         04/25/2007
        7      Nursing                                  3             13          NT          M         10/16/2007
        8      Economics                                2              8          NT          M         01/08/2008
        9      History                                  4              5          PT          B         01/09/2007
       10      Finance                                  3              7          T           D         01/10/2008
       11      Mathematical Studies                     5             16          T           D         06/11/2008
       12      Accounting                               4             18          T           D         05/18/2008
       13      Psychology                               2              9          T           D         02/13/2008
       14      Economics                                5             22          T           D         01/14/2008
       15      Psychology                               1             20          T           D         01/15/2007
       16      Finance                                  2              5          TT          D         01/16/2008
       17      Accounting                               4              3          TT          D         01/01/2007
       18      Biological Sciences                      3              6          NT          M         01/31/2008
       19      Psychology                               5             24          NT          D         02/19/2008
```

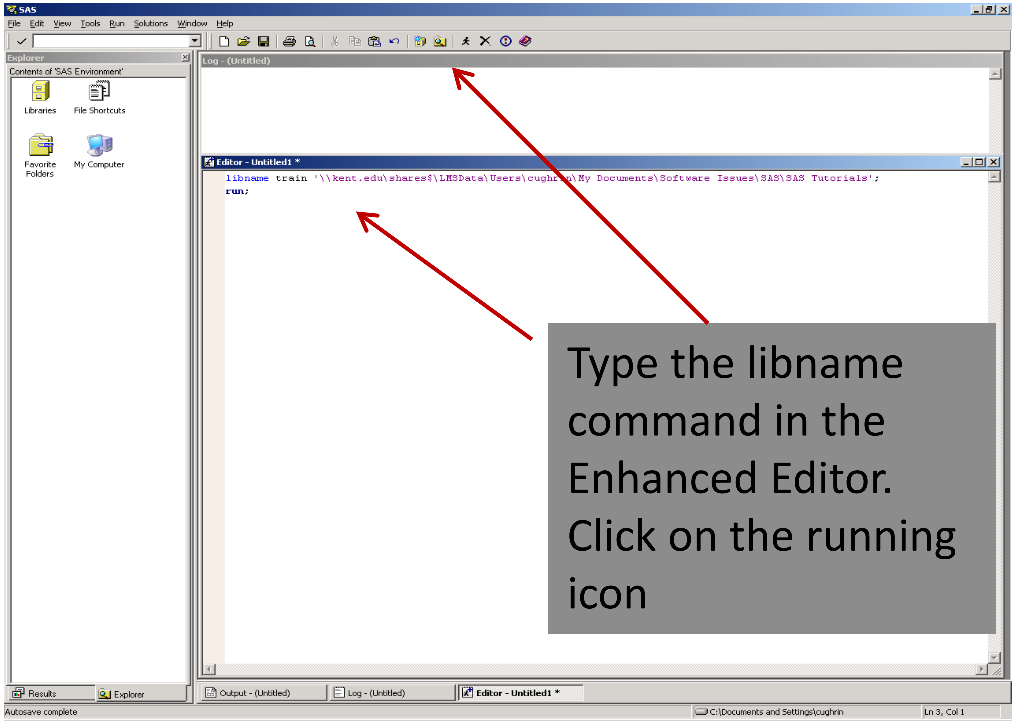
```
proc print data=train.sastraining NOOBS label;
    label ID = Faculty ID
               Department= Department
               Satisfaction= Satisfaction with Department
               Years= Years at the University
               Status= Faculty Status
               Degree= Degree
               Date= Date;
    options ls=200 nodate;
run;
```

SAS Library

- SAS Data Libraries are like drawers in a filing cabinet. The SAS data sets are files within those drawers. Note the icons for the SAS library match that metaphor.
- In order to assign a “drawer”, you assign a library reference name (libref).
- There are two drawers already in your library: work (temporary) and sasuser (permanent).
- You can also create your own libraries (drawers) using the libname statement.



Establishing the libname

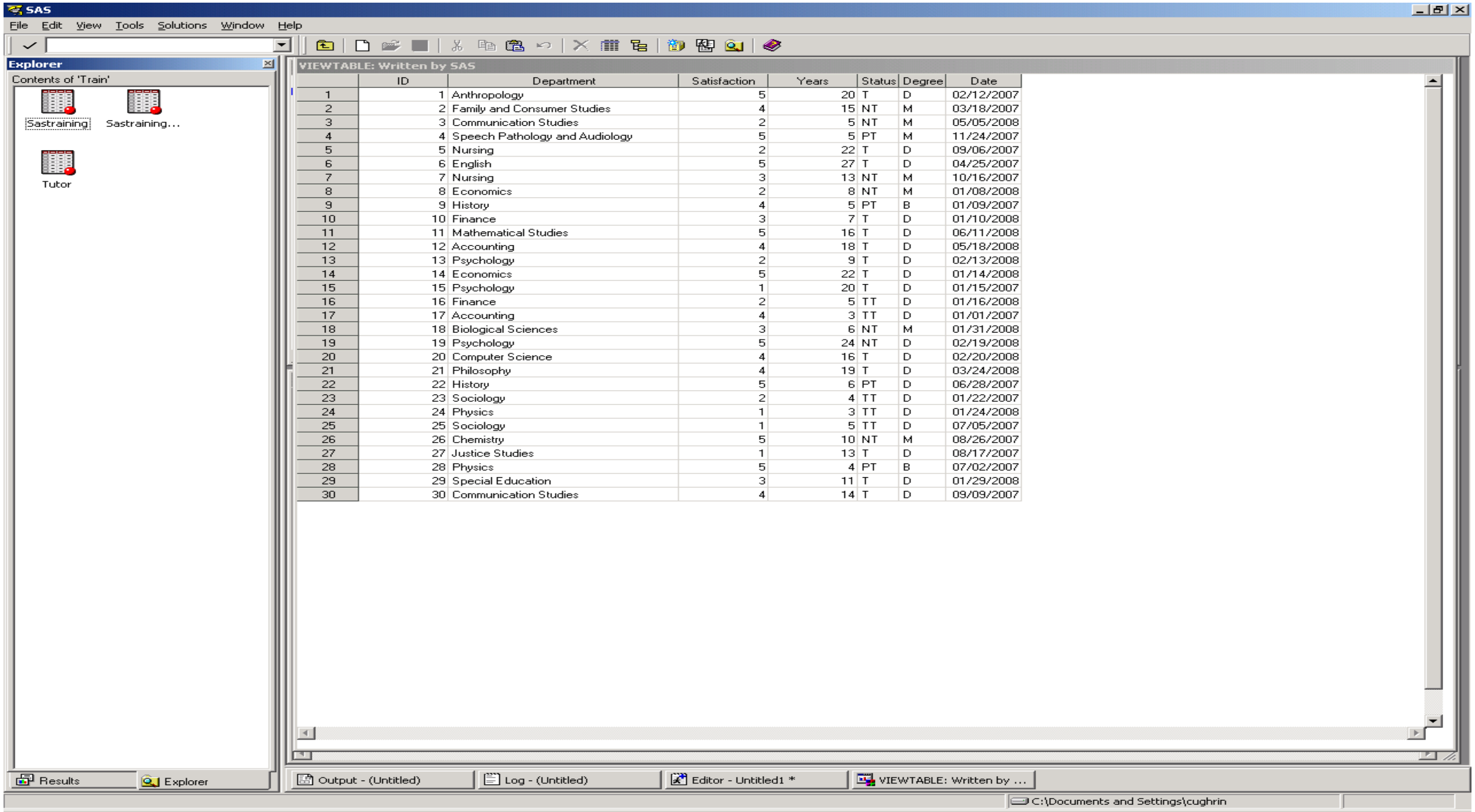


```
libname train '\\kent.edu\shares$\LMSData\Users\cughrin\My Documents\Software Issues\SAS\SAS Tutorials';
run;
```

Type the libname command in the Enhanced Editor. Click on the running icon

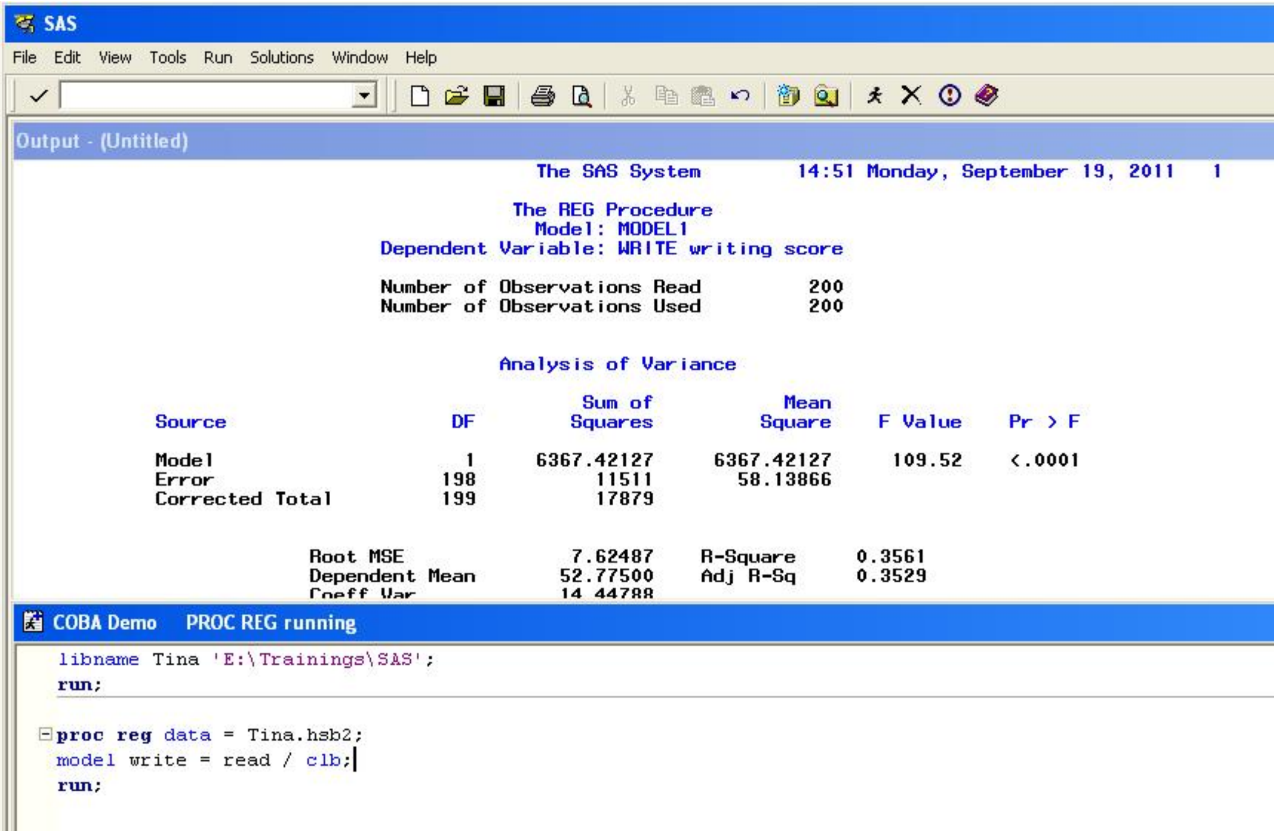
```
libname Tina 'E:\Trainings\JMP Training';
run;
```

Viewtable Window



	ID	Department	Satisfaction	Years	Status	Degree	Date
1	1	Anthropology	5	20	T	D	02/12/2007
2	2	Family and Consumer Studies	4	15	NT	M	03/18/2007
3	3	Communication Studies	2	5	NT	M	05/05/2008
4	4	Speech Pathology and Audiology	5	5	PT	M	11/24/2007
5	5	Nursing	2	22	T	D	09/06/2007
6	6	English	5	27	T	D	04/25/2007
7	7	Nursing	3	13	NT	M	10/16/2007
8	8	Economics	2	8	NT	M	01/08/2008
9	9	History	4	5	PT	B	01/09/2007
10	10	Finance	3	7	T	D	01/10/2008
11	11	Mathematical Studies	5	16	T	D	06/11/2008
12	12	Accounting	4	18	T	D	05/18/2008
13	13	Psychology	2	9	T	D	02/13/2008
14	14	Economics	5	22	T	D	01/14/2008
15	15	Psychology	1	20	T	D	01/15/2007
16	16	Finance	2	5	TT	D	01/16/2008
17	17	Accounting	4	3	TT	D	01/01/2007
18	18	Biological Sciences	3	6	NT	M	01/31/2008
19	19	Psychology	5	24	NT	D	02/19/2008
20	20	Computer Science	4	16	T	D	02/20/2008
21	21	Philosophy	4	19	T	D	03/24/2008
22	22	History	5	6	PT	D	06/28/2007
23	23	Sociology	2	4	TT	D	01/22/2007
24	24	Physics	1	3	TT	D	01/24/2008
25	25	Sociology	1	5	TT	D	07/05/2007
26	26	Chemistry	5	10	NT	M	08/26/2007
27	27	Justice Studies	1	13	T	D	08/17/2007
28	28	Physics	5	4	PT	B	07/02/2007
29	29	Special Education	3	11	T	D	01/29/2008
30	30	Communication Studies	4	14	T	D	09/09/2007

Proc Reg


SAS
File Edit View Tools Run Solutions Window Help
Output - (Untitled)
COBA Demo PROC REG running

```
                        The SAS System         14:51 Monday, September 19, 2011   1

                       The REG Procedure
                         Model: MODEL1
                 Dependent Variable: WRITE writing score

                 Number of Observations Read          200
                 Number of Observations Used          200

                      Analysis of Variance

                                   Sum of          Mean
Source                   DF       Squares        Square    F Value    Pr > F

Model                     1    6367.42127    6367.42127     109.52    <.0001
Error                   198         11511      58.13866
Corrected Total         199         17879

            Root MSE              7.62487    R-Square     0.3561
            Dependent Mean       52.77500    Adj R-Sq     0.3529
            Coeff Var            14.44788
```

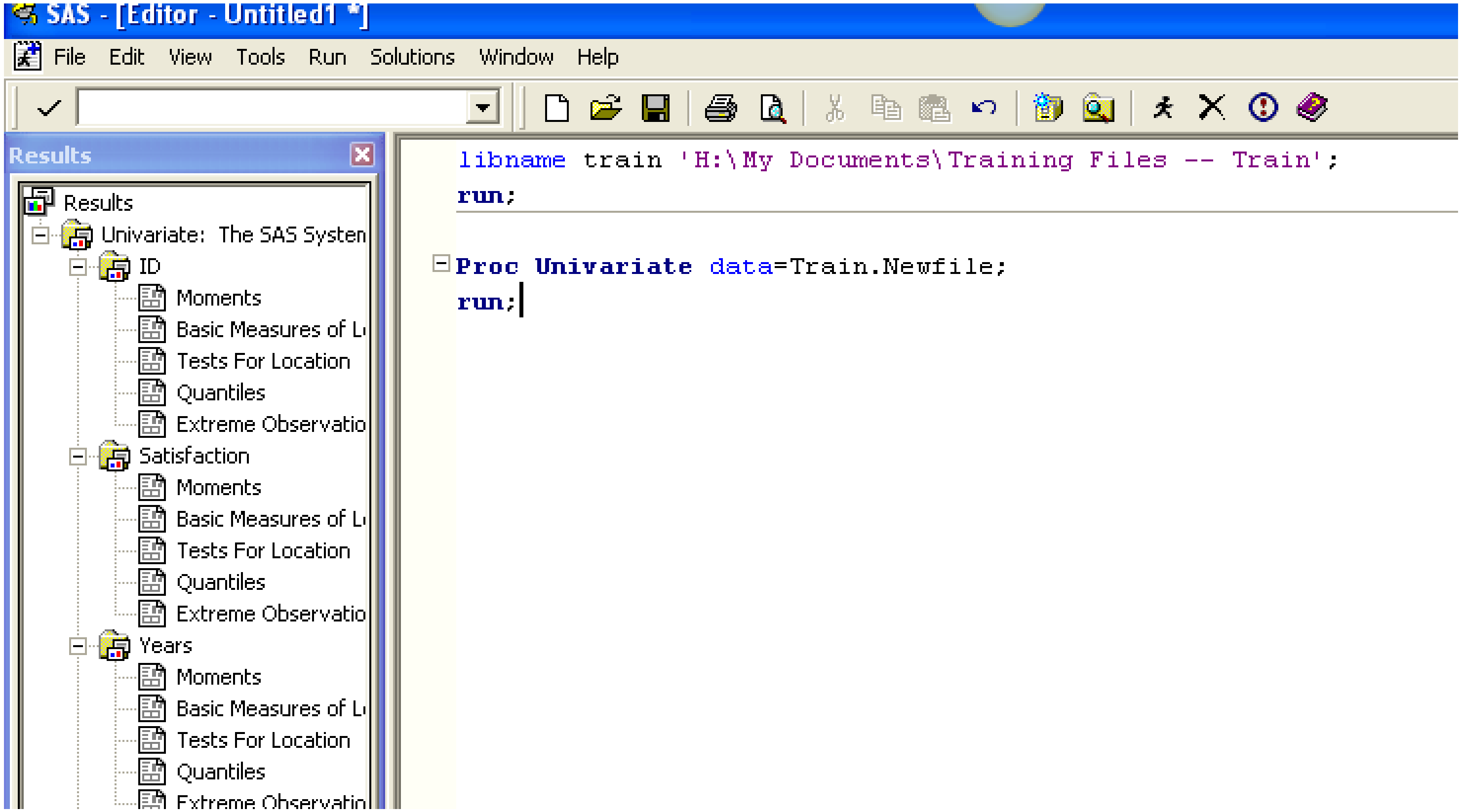
```
libname Tina 'E:\Trainings\SAS';
run;

proc reg data = Tina.hsb2;
model write = read / clb;
run;
```

Proc reg data= Tina.hsb2;
Model write = read / clb;
Run;

Proc Univariate

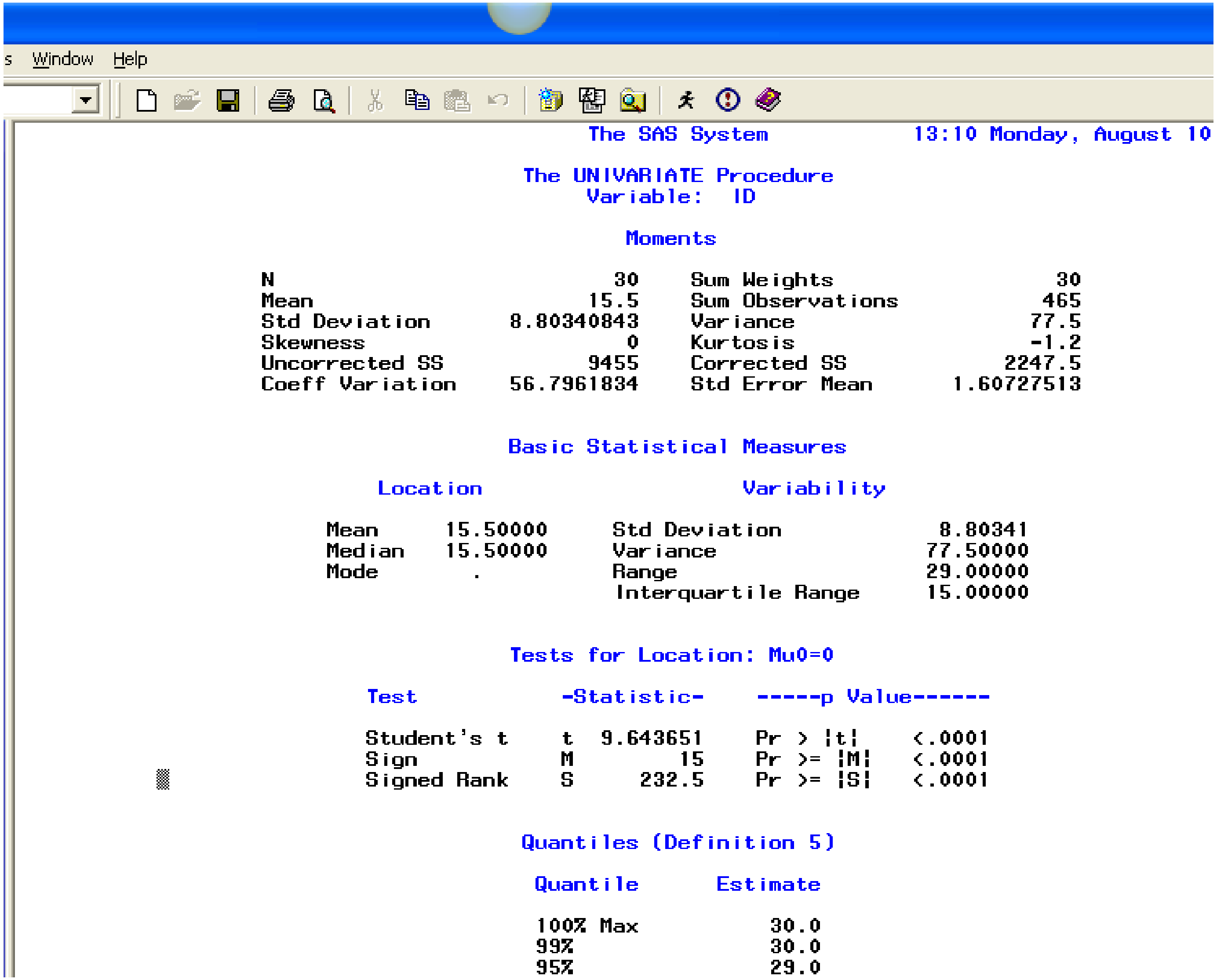
Proc Univariate

Proc Univariate



```
libname train 'H:\My Documents\Training Files -- Train';
run;

Proc Univariate data=Train.Newfile;
run;
```

Proc Univariate



```
                              The SAS System              13:10 Monday, August 10

                            The UNIVARIATE Procedure
                                 Variable:  ID

                                    Moments

             N                          30    Sum Weights                 30
             Mean                     15.5    Sum Observations           465
             Std Deviation      8.80340843    Variance                  77.5
             Skewness                    0    Kurtosis                  -1.2
             Uncorrected SS           9455    Corrected SS            2247.5
             Coeff Variation    56.7961834    Std Error Mean      1.60727513


                          Basic Statistical Measures

                  Location                    Variability

                Mean     15.50000     Std Deviation            8.80341
                Median   15.50000     Variance                77.50000
                Mode      .           Range                   29.00000
                                       Interquartile Range    15.00000


                          Tests for Location: Mu0=0

                    Test           -Statistic-    -----p Value------

                    Student's t    t  9.643651    Pr > |t|    <.0001
                    Sign           M        15    Pr >= |M|   <.0001
                    Signed Rank    S     232.5    Pr >= |S|   <.0001


                          Quantiles (Definition 5)

                           Quantile       Estimate

                           100% Max           30.0
                           99%                30.0
                           95%                29.0
```

Basic rules (1) – organize files

- .sas – program file
- .log – notes, errors, warnings
- .lst – output
- .sas7bdat – data file
- library – a cabinet to put data in
  - Default: Work library
    - temporary, erased after you close the session
  - Permanent library
    - libname mylib “m:\”;
    - mylib.mydata

      = a sas data file named “mydata” in library “mylib”
- run and recall .sas

Basic rules (2) -- program

- every command ends with ;
- format does not matter

  if x=1 then y=1; else y=2; is the same as

```
if x=1     then     y=1;
            else     y=2;
```

- case insensitive
- comment

```
* this is comment;
/* this is comment */;
```

Basic rule (3) – variable

- Type
  - numeric (default, 8 digit, . stands for missing value)
  - character ($, default 8 digit, blank stands for missing)
- Variable names
  - <=32 characters if SAS 9.0 or above
  - <=8 characters if SAS 8 or below
  - case insensitive
- Must start with letter or "_"

  _name, my_name, zip5, u_and_me

  -name, my-name, 5zip, per%, u&me, my@w, my$sign

Common Flow of a SAS Program

- **Beginning:** Create a SAS data set
- **Middle:** Work with data using SAS procedures (PROCs)
- **End:** RUN the program

SAS and Data: SAS Data Sets

- SAS is flexible. Can read data from many sources
- Sometimes you can get SAS data sets from data sources (BLS, etc.)
- First step is to convert raw data to a SAS data set

A SAS Program: Beginning

```
*BEGINNING: Create a SAS data set containing data.  2 steps.;

* Step 1:  Create a SAS data set;

data htwt;                             * create data set named HTWT     ;
input name $ sex $ age height weight;  * input variables by name and type;
x = height + weight;                   * create x                       ;
y = age**2;                            * create y - ** exponentiation   ;
z = 3*age - 5;                         * create z - * multiplication    ;
```

SAS Statements: `data, input, x =, y =, z =`

Beginning: Data Step Processing

```
*BEGINNING: Create a SAS data set containing data.  2 steps.;

* Step 1:  Create a SAS data set;

data htwt;                              * create data set named HTWT      ;
input name $ sex $ age height weight;   * input variables by name and type;
x = height + weight;                    * create x                        ;
y = age**2;                             * create y - ** exponentiation    ;
z = 3*age - 5;                          * create z - * multiplication     ;
```

`data:` Tells SAS the name of the SAS data set being created.

Beginning

```
*BEGINNING: Create a SAS data set containing data.  2 steps.;

* Step 1:  Create a SAS data set;

data htwt;                             * create data set named HTWT      ;
input name $ sex $ age height weight;  * input variables by name and type;
x = height + weight;                   * create x                        ;
y = age**2;                            * create y - ** exponentiation    ;
z = 3*age - 5;                         * create z - * multiplication     ;
```

`input:` Tells SAS the names of the variables being read. `varname $` means character data.

Beginning

```
* Step 2:  Input observations                      ;
* the cards statement precedes data. The data lines;
* DO NOT have semi-colons                          ;
*input name $ sex $ age height weight;

cards;
alfred     M 14 69 112
alice      F 13 56  84
barbara    F 14 62 102
henry      M 15 67 135
john       M 16 70 165
sally      F 16 63 120
;
```

`cards:` Tells SAS the the following lines are data. Data must follow

Delimiters

- Must separate variables on cards or external files
- Accomplished with “delimiters”
- Spaces are common, SAS default
- Can also use other characters, but must tell SAS

Middle: Work with data

```
*MIDDLE: Work with the data.  1 Step.;

* Step 3:  Operate with the SAS data;

proc print;                       * print the data;
title 'Height-Weight Example #1'; * put title with data;
```

`proc:`A SAS procedure. These are how you work with the data in SAS. There are many SAS procedures.

`print:` SAS procedure to create an output file. By default, uses the data from the last data statement.

Summary Statistics in SAS

- Means and Standard Deviations can be easily calculated for variables in a SAS data set using the `means` procedure
- **Format:**

```
 proc means;
    var v1 v2 v3;
```

- List all the variables you want summary statistics for on the second line

Output from `proc means`

```
 Summary Statistics   07:22 Thursday, October 28, 1999   3

Variable  N        Mean     Std Dev      Minimum      Maximum
-----------------------------------------------------------------
X         6 184.1666667  32.4679329  140.0000000  235.0000000
Y         6 216.3333333  35.4664160  169.0000000  256.0000000
Z         6  39.0000000   3.6331804   34.0000000   43.0000000
-----------------------------------------------------------------
```

End: Run program

```
*END: Run the program.  1 step;

* Step 4:  Run the program;

run;                     *Run the above statements;
```

```
data htwt;
input name $ sex $ age height weight;
x = height + weight;
y = age**2;
z = 3*age - 5;

cards;
alfred     M 14 69 112
alice      F 13 56  84
barbara    F 14 62 102
henry      M 15 67 135
john       M 16 70 165
sally      F 16 63 120
;

proc means;
 var x y z;
title 'Summary Statistics';

proc print;
title 'Height-Weight Example #1';

run;
```

Errors in SAS Programs

- You ***will*** make them
- Common ones:
  - Leaving off a semi-colon from the end of a SAS statement
  - Misspelling
  - Omitting one quote (‘) in infile or title statement
- SAS Log will help you to find errors

Some Definitions

- **Field:** Smallest unit of data. One observation of a variable. Can be either character (*letters and numbers*) or numeric (*numbers only*).
- **Record:** A single line of input. Contains one or more fields
- **File:** A collection of records

A Character Field

```
*input name $ sex $ age height weight;
```

```
alfred     M 14 69 112
alice      F 13 56  84
barbara    F 14 62 102
henry      M 15 67 135
john       M 16 70 165
sally      F 16 63 120
;
```

A Numeric Field

```
*input name $ sex $ age height weight;
```

```
alfred     M 14 69 112
alice      F 13 56  84
barbara    F 14 62 102
henry      M 15 67 135
john       M 16 70 165
sally      F 16 63 120
;
```

A Record

```
*input name $ sex $ age height weight;

alfred     M 14 69 112
alice      F 13 56  84
barbara    F 14 62 102
henry      M 15 67 135
john       M 16 70 165
sally      F 16 63 120
;
```

A File

**input name $ sex $ age height weight;*

```
alfred     M 14 69 112
alice      F 13 56  84
barbara    F 14 62 102
henry      M 15 67 135
john       M 16 70 165
sally      F 16 63 120
;
```

Reading External Files

```
data capm;                             * create the dataset capm;
infile 'a:\TABLE.TXT';                 * open the data file Table.txt;
input x1 x2 m;                         * input the variables;

proc print;                            * print;
var x1 x2 m;                           * variables;
title 'CAPM Data';                     * print title;

run;                                   * run;
```

Input Styles: List Input

```
input x1 x2 m;                          * input the variables;
```

This statement reads in the data in a SAS program.

When only the variables are listed, with $ to indicate character variables, it’s called “*List Input*”, the simplest input style in SAS.

You will use different input styles, depending on what the data look like.

Rules for List Input

- Fields must be separated by at least 1 blank
- Each field must appear in order
- Missing values must be represented by a placeholder (a period . in this case)
- No embedded blanks in character fields
- Maximum length of character fields is 8 characters
- Data must be in a standard format (e.g. text file)

Looking at Data in SAS

After creating a SAS data set, it’s a good idea to look at the data to make sure it was read correctly.

You can use proc print to write the data to the output window, or you browse the data interactively.

Let’s browse the data interactively.

SAS Libraries

- Notice that the SAS data file CAPM has another descriptor when we used the “Data Access” menu to browse the data

SAS Libraries

- Notice that the SAS data file CAPM has another descriptor when we used the “Data Access” menu to browse the data
- The first column is headed “Libname”
  - Means “SAS Library Name”
  - CAPM is in Libname “WORK”
- SAS organizes data into “Libraries”, which are subdirectories

SAS Libraries

- CAPM is in Library “WORK”
- SAS automatically creates a Library called WORK in temporary memory.
- Anything in WORK is erased when you end your SAS session
- SAS data sets can be identified by a two-part name: libname.filename

  `work.capm` is equivalent to capm

SAS Libraries

- Permanent SAS data files are kept in libraries. To permanently save a SAS data set, you must define a library other than WORK using a LIBNAME statement
- **Format:**

`LIBNAME` *libref ‘your-data-library’;*

- *libref* is the SAS name for your library
- *‘your-data-library’* is a subdirectory

Data cleaning (1) – if then

Format:

```
IF condition THEN action;
ELSE IF condition THEN action;
ELSE action;
```

Note:

(1) the if-then-else can be nested as many as you want

(2) if you need multiple actions instead of one action, use “DO; action1; action2; END; ”

Data cleaning (1) – if then

- = or EQ means equals
- ~= or NE means not equal
- > or GT means greater than
- < or LT means less than
- >= or GE means greater than or equal
- <= or LE means less than or equal
- in means subset
  - if gender in (‘M’, ‘F’) then ..;
- Multiple conditions: AND (&), OR(|)

Data cleaning (1) – if then

```
*reading in program of proj3rawdata3 is on page 21;
data proj3rawdata3;
set proj3rawdata3;
IF fracuninsured<0.15 THEN uninsuregrp=0;
ELSE uninsuregrp=1;
run;
proc contents data=proj3rawdata3; run;
proc print data=proj3rawdata3; run;
```

Note: (1) the code is less efficient if you replace ELSE ..; with

```
 IF fracuninsured>=0.15 THEN ..;
```

(2) missing value is always counted as the smallest negative, so fracuninsured=. will satisfy the condition fracuinsured<0.15. If you want to ignore the missing obs set the condition as 0<=fracuninsured<0.15.

Data cleaning (1) – if then

```
* Multiple actions in each branch;

data proj3rawdata3;
set proj3rawdata3;
IF fracuninsured<0.15 AND uninsured>1000000 THEN DO;
    uninsuredgrp=0; uninsuredpop=‘over 1 million';
    END;
ELSE DO;
    uninsuredgrp=1; uninsuredpop=‘less than 1 m
    END;
run;
proc print data=proj3rawdata3; run;
```

the do-end pair acts as brackets

Data cleaning (1) – if then

```
*Use if commands to choose a subsample;

data proj3subsample; /* note here we generate a new data set */
set proj3rawdata3;
IF fracuninsured=. Then delete;
If fracuninsured<=0.1;
run;
proc print data=proj3subsample; run;
```

Data cleaning (1) – exercise

still use proj3rawdata.

define newgrp = 1 if fracuninsured <0.1 (low)

2 if 0.1<=fracuninsured<0.15 (mid-low)

3 if 0.15<=fracuninsured<0.2 (mid-high)

4 if fracuninsured>=0.2 (high).

Data cleaning (1) – exercise answer

```
data proj3rawdata3;
  set proj3rawdata3;
  if fracuninsured<0.1 then  newgrp=1;
  else if fracuninsured<0.15 then newgrp=2;
         else if fracuninsured<0.2 then newgrp=3;
             else newgrp=4;
run;
proc contents data=proj3rawdata3; run;
proc print data=proj3rawdata3; run;
```

Question: What if one observation has fracuninsured=.?

Save data

```
* Save in sas format;
libname mylib “M:\”;
data mylib,proj3rawdata3;
    set proj3rawdata3;
run;

* Export data to excel;
Proc export data=proj3rawdata3
     outfile=“M:\proj3data-fromsas.xls”
     dbms=excel replace;
Run;
```

No ; here

You can also export a sas data file into a comma delimited text file if you write dbms=csv.

proc sort

```
proc sort data=proj3rawdata3;
 by year state;
 run;
proc sort data=proj3rawdata3
  out=proj3rawdata3_sorted;
 by year descending fracuninsured;
 run;
```

* note that missing value is always counted as the smallest;

proc means and proc univariate

```
proc means data=proj3rawdata3;
   class newgrp;
   var insured uninsured fracuninsured;
   run;
```

By default, proc means report mean, stdev, min, max
Could choose what to report:
proc means data=proj3rawdata3 n mean median;

```
proc sort data=proj3rawdata3; by newgrp; run;
```

```
proc univariate data=proj3rawdata3;
   by newgrp;
   var insured uninsured fracuninsured;
   run;
```

By default, proc univariate report median, and many other statistics

Notes on proc means and proc univariate

*if you do not use class or by command, the statistics are based on the full sample. If you use class or by var x, the statistics are based on the subsample defined by each value of var x.

*You can use class or by in proc means, but only by in proc univariate;

*whenever you use “by var x”, the data set should be sorted by var x beforehand;

proc means and proc univariate allow multiple groups

```
data proj3rawdata3;
    set proj3rawdata3;
    if totalpop<6000000 then popgrp=“low”;
    else popgrp=“high”;
    run;
proc means data=proj3rawdata3;
    class newgrp popgrp;
    var fracuninsured;
    run;
proc sort data=proj3rawdata3;
    by newgrp popgrp;
    run;
proc univariate data=proj3rawdata3;
    by newgrp popgrp;
    var fracuninsured;
    run;
```

proc freq

```
* Remember we already generate a variable called newgrp to
  indicate categories of fraction uninsured and a variable called
  popgrp to indicate categories of population size;
```

```
proc freq data=proj3rawdata3;
  tables newgrp
         popgrp
         newgrp*popgrp;
  run;
```

One dimension frequency table

Two-dimension frequency table

proc chart – histogram for categorical variables

```
proc chart data=proj3rawdata3;
  title ‘histogram for newgrp’;
  vbar newgrp;
  run;

proc chart data=proj3rawdata3;
  title ‘frequency by two variables’;
  vbar newgrp / group=popgrp;
  run;
```

proc chart – histogram for continuous variable

```
proc chart data=proj3rawdata3;
  title “histogram for continuous variable’;
  vbar fracuninsured;
  run;
proc chart data=proj3rawdata3;
  title ‘histogram with specific midpoints’;
  vbar fracuninsured / midpoints=0 to 1 by 0.05;
  run;
```

proc plot – scatter plot

```
proc plot data=proj3rawdata3;
  title ‘scatter plot of fracuninsured and totalpop’;
  plot fracuninsured*totalpop;
  run;
```

scatter plot is less informative for categorical variables

```
proc plot data=proj3rawdata3;
  title ‘scatter plot of newgrp and popgrp’;
  plot newgrp*popgrp;
  run;
```

fancy proc means

```
proc means data=proj3rawdata3;
  class newgrp popgrp;
  var uninsured fracuninsured;
  output    out  = summary1
            mean = avguninsured avgfracuninsured;
  run;
proc print data=summary1;
  run;
```

some summary stat. in proc print

```
* Assume we have already defined newgrp and
  popgrp in proj3rawdata3;

proc sort data=proj3rawdata3; by popgrp; run;
proc print data=proj3rawdata3 n;
  where fracuninsured>=0.1;
  by popgrp;
  sum totalpop;
  var totalpop insured uninsured fracuninsured;
  run;
```

How to handle multiple data sets?

- Add *more observations* to an existing data and the new observations follow the same data structure as the old one ➔ append
- Add *more variables* to an existing data and the new variables refer to the same subjects as in the old data ➔ merge
- Sometimes we may need to change data structure to fit in append or merge ….

merge and append

proj3rawdata3:

year	state	totalpop	…	fracuninsured	newgrp	popgrp
2009	MA	6420947	…	0.0548	1	high

summary1:

newgrp	popgrp	avguninsured	avgfracuninsured
1	high	7500000	0.073

merged:

year	**state**	**totalpop**	**….fracuninsured**	**newgrp**	**popgrp**	**avguninsure**	**avgfracuninsured**
2009	MA	6420947	… 0.0548	1	high	7500000	0.073

appended:

year	**state**	**totalpop**	**….fracuninsured**	**newgrp**	**popgrp**	**avguninsure**	**avgfracuninsured**
2009	MA	6420947	… 0.0548	1	high	.	.
.		.	.	1	high	7500000	0.073

merge two datasets

```
proc sort data=proj3rawdata3;
   by newgrp popgrp;
   run;
proc sort data=summary1;
   by newgrp popgrp;
   run;
data merged;
   merge proj3rawdata3 (in=one) summary1 (in=two);
   by newgrp popgrp;
   if one=1 & two=1;
   run;
```

What if this line is
“if one=1 OR two=1;”?

Keep track of matched and unmatched records

```
data allrecords;
   merge proj3rawdata3 (in=one) summary1 (in=two);
   by newgrp popgrp;
   myone=one;
   mytwo=two;
   if one=1 or two=1;
   run;
proc freq data=allrecords;
   tables myone*mytwo;
   run;
```

SAS will drop variables "one" and "two" automatically at the end of the DATA step. If you want to keep them, you can copy them into new variables "myone" and "mytwo"

be careful about merge!

- always put the merged data into a new data set
- must sort by the key variables before merge
- ok for one-to-one, multi-to-one, one-to-multi, but no good for multi-to-multi
- be careful of what records you want to keep, and what records you want to delete
- what if variable x appears in both datasets, but x is not in the “by” statement?
    - after the merge x takes the value defined in the last dataset of the “merge” statement

append

```
data appended;
  set proj3rawdata3 summary1;
  run;
proc print data=appended;
  run;
proc print data=merged;
  run;
```

Class example of merge and append: reshape and summarize

Task1: reshape proj3rawdata3 from long to wide

Task2: generate average fracuninsured per state and merge it back to the main data

Source format of Proj3rawdata3 (long):

year	state	totalpop	fracuninsured	….
2009	MA	6420947	0.0548	….
2009	HI	1257622	0.078	….
….				
2008	MA	6339513	0.0536	….
2008	HI	1267409	0.075	…..

Target format (wide)

state	totalpop2009	fracuninsured2009	. ..	Totalpop2008	fracuninsured2008	…..
MA	6420947	0.0548	….	6339513	0.0536	……
HI	1257622	0.078	….	1267409	0.075	……..