

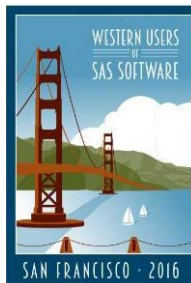


WESTERN USERS OF SAS SOFTWARE

SAN FRANCISCO • 2016

Introduction to DATA Step Programming SAS Basics II

Susan J. Slaughter, Avocet Solutions



DATA versus PROC steps

- Two basic parts of SAS programs

DATA step

Begin with DATA statement

Input and modify data

Create SAS data set

Flexibility of programming

PROC step

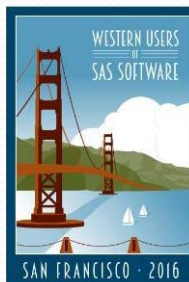
Begin with PROC statement

Perform analysis or task

Produce report

Like filling out a form

Susan says: This is a simplification



DATA versus PROC steps

- A simple example

```
DATA temps;
```

```
    Farenheit = 68;
```

```
    Celsius = (Farenheit - 32) * 0.5556;
```

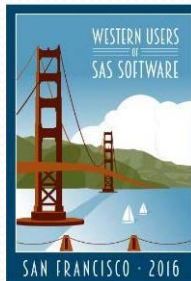
```
PROC PRINT DATA = temps;
```

```
    TITLE 'Temperature Conversions';
```

```
RUN;
```

DATA
step

PROC
step



DATA versus PROC steps

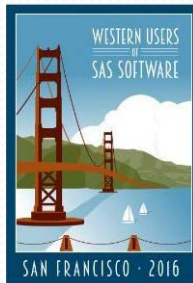
The screenshot shows the SAS interface with the following content:

```
1 DATA temps;  
2   Farenheit = 68;  
3   Celsius = (Farenheit - 32) * 0.5556;  
  
NOTE: The data set WORK.TEMPS has 1 observations and 2 variables.  
NOTE: DATA statement used (Total process time):  
      real time           0.01 seconds  
      cpu time            0.00 seconds  
  
4 PROC PRINT DATA = temps;  
NOTE: Writing HTML Body file: sashtml.htm  
5   TITLE 'Temperature Conversions';  
6   RUN;  
  
NOTE: There were 1 observations read from the data set WORK.TEMPS.  
NOTE: PROCEDURE PRINT used (Total process time):  
      real time           0.54 seconds  
      cpu time            0.32 seconds
```

The Results Viewer window displays the following table:

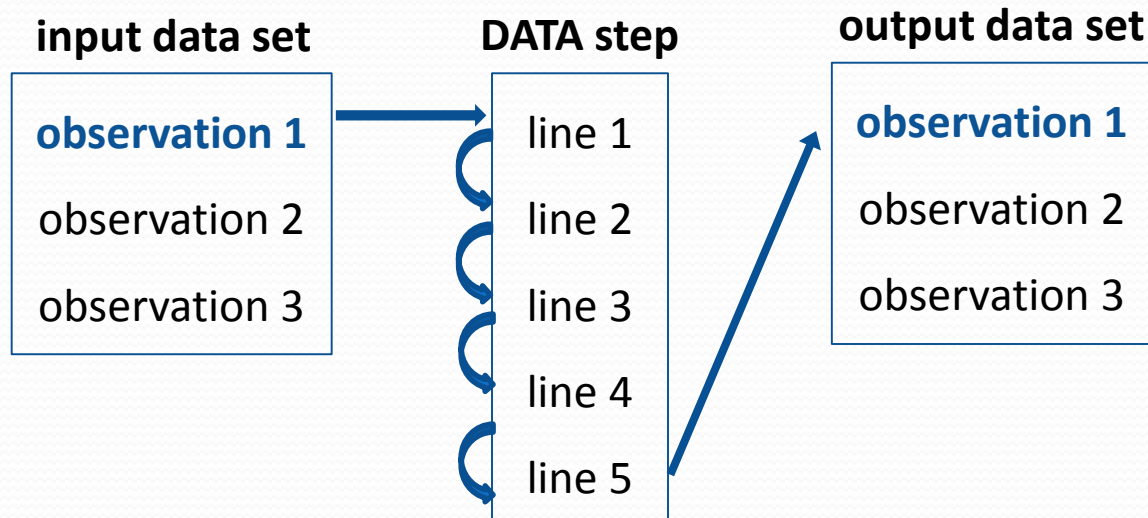
Obs	Farenheit	Celsius
1	68	20.0016

Be sure to check your SAS log!



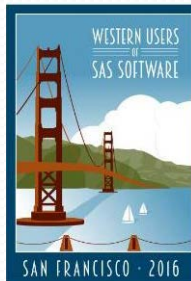
DATA step's built-in loop

- DATA steps execute line-by-line and observation-by-observation



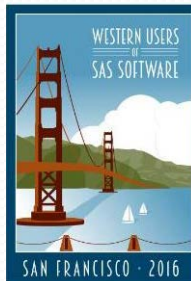
Reading data in a DATA step

- Reading raw data file
 - INPUT statement
- Reading SAS data sets
 - SET statement
 - MERGE statement



Reading raw data

- INPUT statements read raw data
 - External raw data files
 - also called text, ASCII, sequential, flat file, CSV files
 - Internal raw data
 - also called instream data
- Three basic styles of INPUT statements
 - List style
 - Column style
 - Formatted style



Reading raw data

```
* Read internal data;
```

```
DATA students;
```

```
    INPUT ID $ Name $ Age Major $; ← List style input
```

```
    DATALINES;
```

```
78374 Thomas 21 .
```

```
75638 Cathy . STA ← Internal data
```

```
78634 David 20 ENG
```

```
;
```

```
* Print data set named students;
```

```
PROC PRINT DATA = students;
```

```
    TITLE 'Students';
```

```
RUN;
```


Reading raw data

The screenshot shows the SAS interface. The main window is titled "Log - (Untitled)" and contains the following code and output:

```
1 * Read internal data;  
2 DATA students;  
3 INPUT ID $ Name $ Age Major $;  
4 DATALINES;  
  
NOTE: The data set WORK.STUDENTS has 3 observations and 4 variables.  
NOTE: DATA statement used (Total process time):  
      real time      0.01 seconds  
      cpu time       0.00 seconds  
  
8 ;  
9  
10 * Print data set named students;  
11 PROC PRINT DATA = students;  
NOTE: Writing HTML Body file: sashtml.htm  
12 TITLE 'Students';  
13 RUN;
```

Below the log window is a "Results Viewer - SAS Output" window titled "Students" displaying a table with the following data:

Obs	ID	Name	Age	Major
1	78374	Thomas	21	
2	75638	Cathy	.	STA
3	78634	David	20	ENG

The interface also shows a "Results" pane on the left with "Print: Students" and a taskbar at the bottom with "Output - (Untitl...", "Log - (Untitled)", "SAS Essentials.s...", and "Results Viewe...".

Be sure to check your SAS log!

Reading a single SAS data set

- SET statement

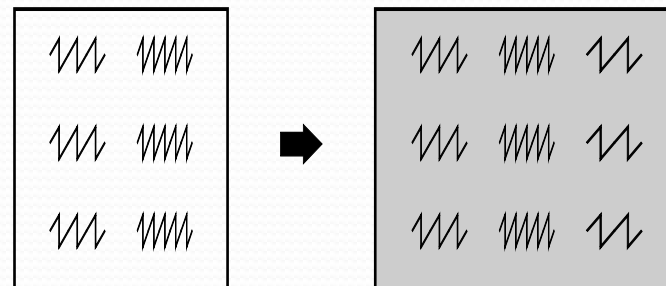
- General form:

```
DATA new-data-set;  
    SET old-data-set;
```

- Example:

```
DATA class;  
    SET class;
```

- If data set names are same, then new replaces old



Stacking SAS data sets

- SET statement
- Stacking = concatenating
- General form:

```
DATA new-data-set;
```

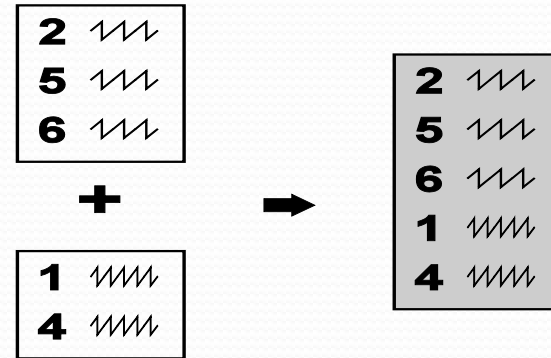
```
    SET old-data-set-1 ... old-data-set-n;
```

- Example:

```
DATA new_stacked;
```

```
    SET class1 class2 class3;
```

- Order of observations in new data set depends on order in SET statement



Interleaving SAS data sets

- SET statement
- General form:

```
DATA new-data-set;
```

```
    SET old-data-set-1 ... old-data-set-n;
```

```
    BY variable-list;
```

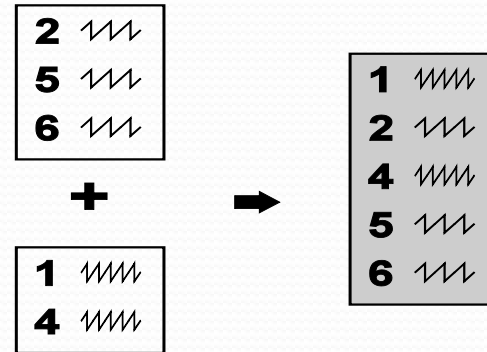
- Example:

```
DATA new_stacked;
```

```
    SET class1 class2 class3;
```

```
    BY ID;
```

- Data sets must be sorted by variables in BY statement
 - Use PROC SORT



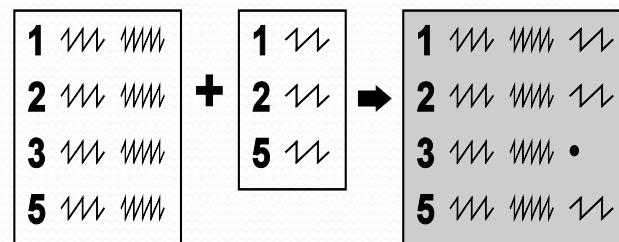
Merging SAS data sets 1-to-1

- MERGE statement
- General form:

```
DATA new-data-set;
```

```
MERGE old-data-set-1 old-data-set-2;
```

```
BY variable-list;
```



- Example:

```
DATA new_merged;
```

```
MERGE class1 demog;
```

```
BY ID;
```

- Data sets must be sorted by values of BY variable
 - If needed, use PROC SORT

Merging SAS data sets 1-to-many

- MERGE statement
- General form:

```
DATA new-data-set;
```

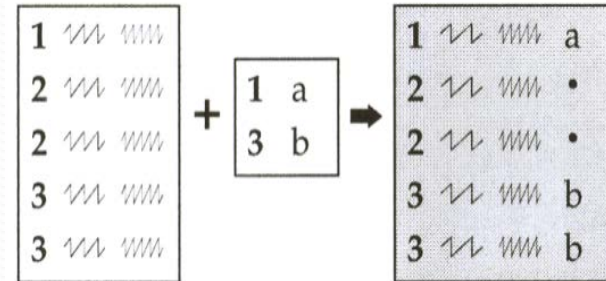
```
MERGE old-data-set-1 old-data-set-2;  
BY variable-list;
```

- Example:

```
DATA new_merged;
```

```
MERGE class1 demog;  
BY ID;
```

- Data sets must be sorted by values of BY variable
 - If needed, use PROC SORT



Data for examples

* Input student enrollment data;

```
DATA SASUSER.students;
```

```
    INPUT ID $ Name $ AmtPaid Course $ New;
```

```
    DATALINES;
```

```
78374 Adam      350.00 597 1
```

```
75638 Michele  525.00 221 1
```

```
78634 Jacob     625.00 221 0
```

```
28746 .          .      597 2
```

```
58743 Zina       250.00 435 0
```

```
45378 Amy        250.00 435 0
```

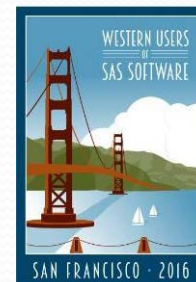
```
87463 Angela    525.00 221 1
```

```
46732 Trevor    450.00 597 0
```

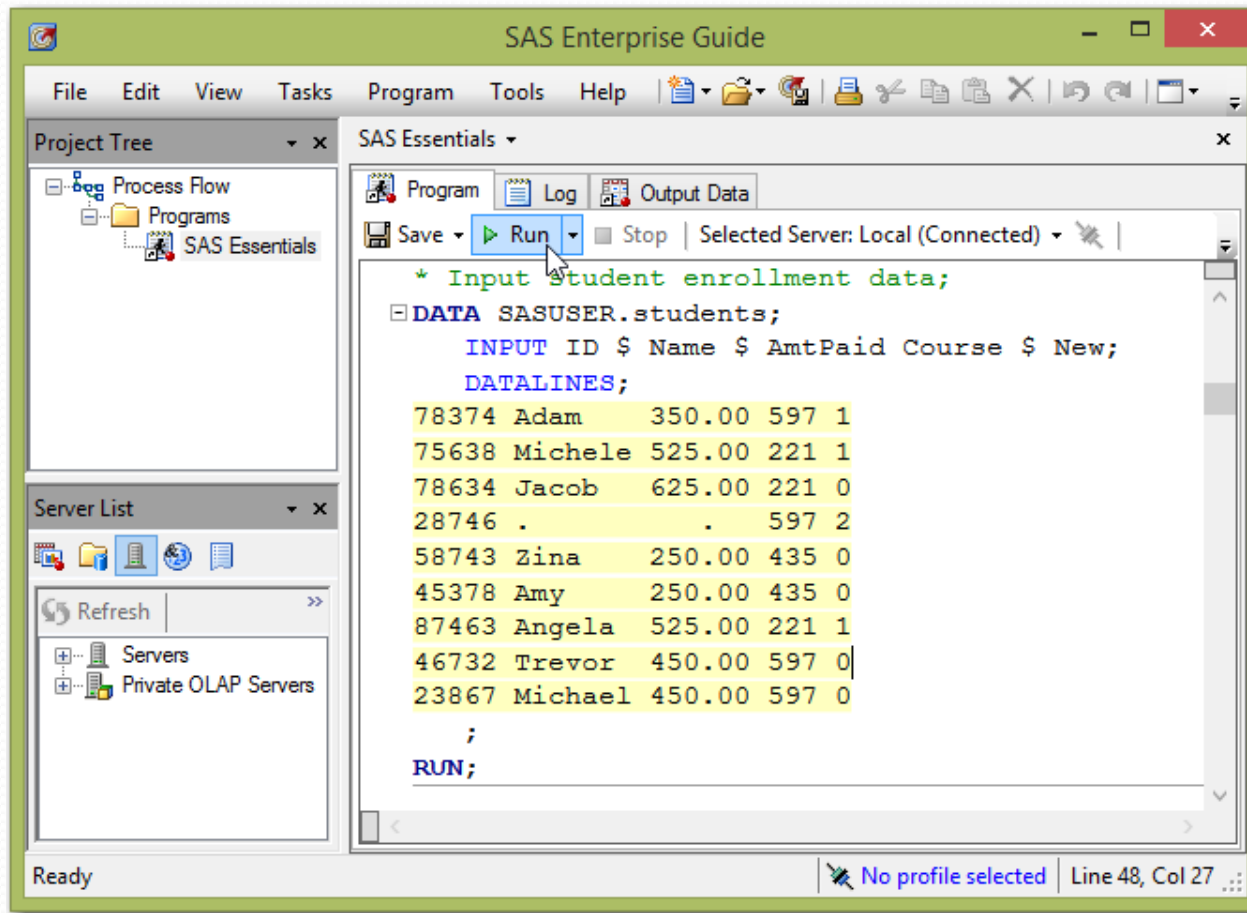
```
23867 Michael  450.00 597 0
```

```
    ;
```

```
RUN;
```



Data for examples

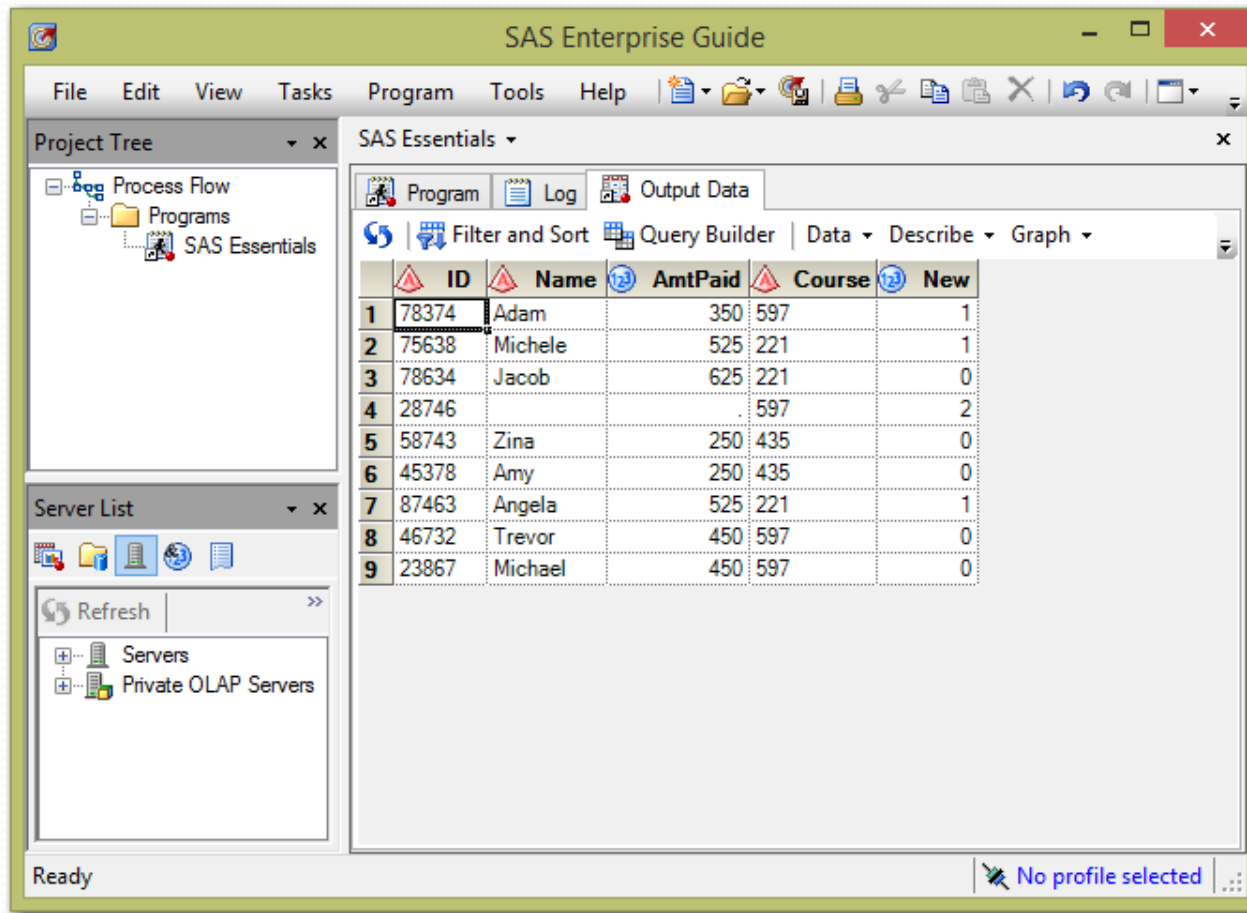


The screenshot displays the SAS Enterprise Guide interface. The main window shows a SAS program with the following code:

```
* Input student enrollment data;  
DATA SASUSER.students;  
  INPUT ID $ Name $ AmtPaid Course $ New;  
  DATALINES;  
78374 Adam      350.00 597 1  
75638 Michele  525.00 221 1  
78634 Jacob    625.00 221 0  
28746 .         .      597 2  
58743 Zina     250.00 435 0  
45378 Amy      250.00 435 0  
87463 Angela   525.00 221 1  
46732 Trevor   450.00 597 0  
23867 Michael  450.00 597 0  
;  
RUN;
```

The interface includes a Project Tree on the left showing 'SAS Essentials', a Server List at the bottom left, and a status bar at the bottom indicating 'Ready' and 'No profile selected | Line 48, Col 27'.

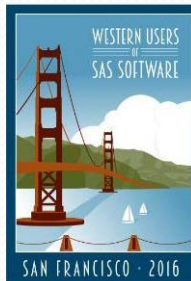
Data for examples



The screenshot displays the SAS Enterprise Guide interface. The main window shows a data table with the following columns: ID, Name, AmtPaid, Course, and New. The data is as follows:

	ID	Name	AmtPaid	Course	New
1	78374	Adam	350	597	1
2	75638	Michele	525	221	1
3	78634	Jacob	625	221	0
4	28746			597	2
5	58743	Zina	250	435	0
6	45378	Amy	250	435	0
7	87463	Angela	525	221	1
8	46732	Trevor	450	597	0
9	23867	Michael	450	597	0

The interface also includes a Project Tree on the left showing 'Process Flow', 'Programs', and 'SAS Essentials'. Below it is a Server List with 'Servers' and 'Private OLAP Servers'. The status bar at the bottom indicates 'Ready' and 'No profile selected'.



Assignment statements

- Set one thing equal to another
- General form:

variable-name = expression;

Example

Type of expression

x = 5;

numeric constant

x = '5';

character constant

x = y;

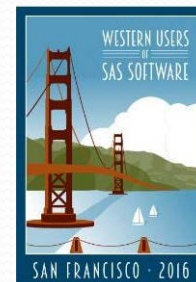
a variable

x = y + 1;

addition

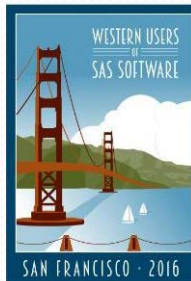
x = y * z;

multiplication



SAS functions

- Functions perform calculation or transformation
- SAS has hundreds of functions
- Types of functions
 - Character
 - Date and time
 - Descriptive statistics
 - Mathematical
 - Probability
 - Random number
 - State and ZIP code



SAS functions

- Functions are often used in assignment statements

- General form:

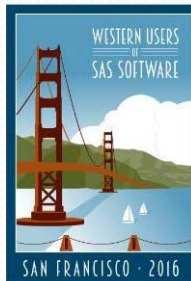
```
var = function-name(argument, argument,...);
```

- Example:

```
PhoneNum = '(714) 637-9398';
```

```
AreaCode = SUBSTR(PhoneNum, 2, 3);
```

Value of AreaCode is '714'



SAS functions

- Examples:

```
Test1 = 80;
```

```
Test2 = .;
```

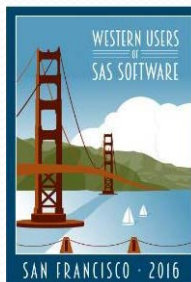
```
Test3 = 100;
```

```
AverageScore1 = Test1 + Test2 + Test3 / 3;
```

```
AverageScore2 = MEAN(Test1, Test2, Test3);
```

Value of AverageScore1 is missing

Value of AverageScore2 is 90



Assignment statements

- Example:

* Student enrollment data;

```
DATA SASUSER.students;
```

```
  SET SASUSER.students;
```

```
  Quarter = 'Fall';
```

```
  FirstInitial = SUBSTR(Name, 1,1);
```

```
RUN;
```

	ID	Name	AmtPaid	Course	New	Quarter	FirstInitial
1	78374	Adam	350	597	1	Fall	A
2	75638	Michele	525	221	1	Fall	M
3	78634	Jacob	625	221	0	Fall	J
4	28746			597	2	Fall	
5	58743	Zina	250	435	0	Fall	Z
6	45378	Amy	250	435	0	Fall	A
7	87463	Angela	525	221	1	Fall	A
8	46732	Trevor	450	597	0	Fall	T
9	23867	Michael	450	597	0	Fall	M

Subsetting IF

- Special form of IF statement

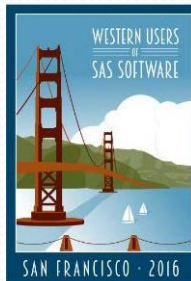
- General form:

IF condition;

- Example:

IF Age >= 21;

SAS will keep only observations with ages 21 or over.



Conditional logic: IF-THEN

- IF-THEN statements
- General form:

```
IF condition THEN action;
```

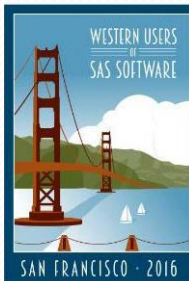
```
IF condition AND condition THEN action;
```

```
IF condition OR condition THEN action;
```

- Examples:

```
IF State = 'AZ' THEN Region = 'West';
```

```
IF State = 'CA' AND County = 'Yolo'  
THEN Area = 'Central';
```



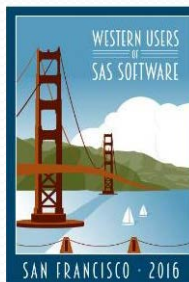
Conditional logic: IF-THEN/ELSE

- IF-THEN/ELSE statements
- General form:

IF condition THEN action;

ELSE IF condition THEN action;

ELSE action;



Conditional logic: IF-THEN/ELSE

- IF-THEN/ELSE statements

- Bad example:

```
IF State = 'CA' THEN Region = 'West' ;
```

```
IF State = 'MA' THEN Region = 'East' ;
```

```
IF State = 'TX' THEN Region = 'Gulf' ;
```

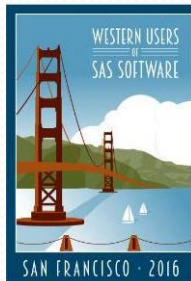
- Better example:

```
IF state = 'CA' THEN Region = 'West' ;
```

```
ELSE IF State = 'MA' THEN Region = 'East' ;
```

```
ELSE IF State = 'TX' THEN Region = 'Gulf' ;
```

```
ELSE Region = 'None' ;
```



Conditional logic

- Example:

* Student enrollment data;

```
DATA SASUSER.students;
```

```
  SET SASUSER.students;
```

```
  IF Name NE ' ';
```

```
  IF New = 1 THEN NewStudent = 'yes';
```

```
  ELSE IF New = 0 THEN NewStudent = 'no';
```

```
  ELSE NewStudent = '?';
```

```
RUN;
```

	ID	Name	AmtPaid	Course	New	Quarter	FirstInitial	NewStudent
1	78374	Adam	350	597	1	Fall	A	yes
2	75638	Michele	525	221	1	Fall	M	yes
3	78634	Jacob	625	221	0	Fall	J	no
4	58743	Zina	250	435	0	Fall	Z	no
5	45378	Amy	250	435	0	Fall	A	no
6	87463	Angela	525	221	1	Fall	A	yes
7	46732	Trevor	450	597	0	Fall	T	no
8	23867	Michael	450	597	0	Fall	M	no

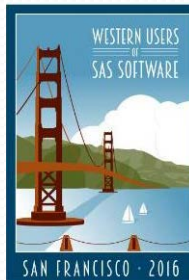
DO groups

- A single IF-THEN statement can have only one action
- For multiple actions, use DO/END statements
- General form:

```
IF condition THEN DO;  
action;  
action;  
END;
```

- Example:

```
IF State = 'HI' THEN DO;  
Region = 'West';  
Capital = 'Honolulu';  
END;
```



DO groups

- Example:

* Student enrollment data;

```
DATA SASUSER.students;
```

```
  SET SASUSER.students;
```

```
  IF Course = '221' THEN DO;
```

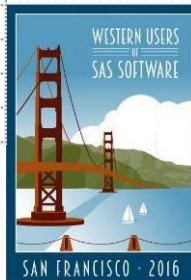
```
    CourseName = 'Introduction to SAS';
```

```
    Instructor = 'Susan Slaughter';
```

```
  END;
```

```
RUN;
```

	ID	Name	AmtPaid	Course	New	Quarter	FirstInitial	NewStudent	CourseName	Instructor
1	78374	Adam	350	597	1	Fall	A	yes		
2	75638	Michele	525	221	1	Fall	M	yes	Introduction to SAS	Susan Slaughter
3	78634	Jacob	625	221	0	Fall	J	no	Introduction to SAS	Susan Slaughter
4	58743	Zina	250	435	0	Fall	Z	no		
5	45378	Amy	250	435	0	Fall	A	no		
6	87463	Angela	525	221	1	Fall	A	yes	Introduction to SAS	Susan Slaughter
7	46732	Trevor	450	597	0	Fall	T	no		
8	23867	Michael	450	597	0	Fall	M	no		



Pop quiz

- 1) What statement do you use to read data from a raw data file?

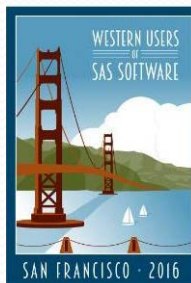
INPUT statement

- 2) What statement do you use to concatenate two SAS data sets?

SET statement

- 3) What statement do you use to match observations from one data set with observations from another data set?

MERGE statement



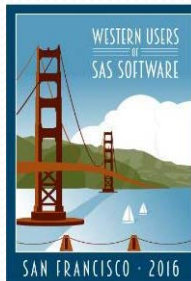
Pop quiz

- 4) Write a statement that assigns the value of 10 to a variable named Score.

```
Score = 10;
```

- 5) Write a statement that assigns the value of Alaska to a variable named State.

```
State = 'Alaska';
```



Pop quiz

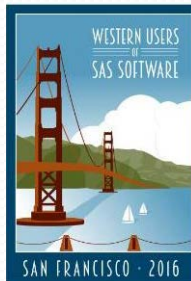
6) What statement do you use to end a DO group?

END statement

7) What will this statement do?

```
IF Name = 'Joe' ;
```

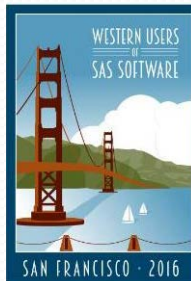
This subsetting IF statement will keep all observations where the value of the variable Name equals Joe.



Pop quiz

Extra credit: Can an ELSE statement be used without an IF-THEN statement?

No, SAS will give you an error if you use an ELSE without an IF-THEN statement.



Thank you!

I hope you can stay for the next presentation.

Susan Slaughter
Avocet Solutions

Can download slides from
www.avocetsolutions.com

