Print

Summary of Lesson 9: Manipulating Data

This summary contains topic summaries, syntax, and sample programs.

Topic Summaries

To go to the movie where you learned a task or concept, select a link.

Using SAS Functions

You use an assignment statement in a DATA step to evaluate an expression and assign the result to a new or existing variable. The expression on the right side of an assignment statement can include calls to SAS functions. A SAS function is a routine that accepts arguments and returns a value.

```
variable=expression;
```

The SUM function, a descriptive statistics function, returns the sum of its arguments and ignores missing values. The arguments can be numeric constants, numeric variables, or arithmetic expressions, but the arguments must be numeric values and must be enclosed in parentheses and separated with commas. The parentheses are required, even if no arguments are passed to the function.

```
SUM(argument1,argument2, ...)
```

In addition to descriptive statistics functions, many SAS date functions are available. Some of these functions create SAS dates, and others extract information from SAS dates. The MONTH function extracts and returns the numeric month from a SAS date.

```
MONTH(SAS-date)
```

Conditional Processing

The IF-THEN statement is a conditional statement. It executes a SAS statement for observations that meet specific conditions. The statement includes an expression and a SAS program statement. The expression defines a condition that must be true for the statement to be executed. The expression is evaluated during each iteration of the DATA step. If the condition is true, the statement following the THEN statement is executed; otherwise, SAS skips the statement.

```
IF expression THEN statement;
```

A program often includes a sequence of IF statements with mutually exclusive conditions. When SAS encounters a true condition in this series, evaluating the other conditions isn't necessary. You can use the ELSE statement to specify an alternative action to be performed when the condition in an IF-THEN statement is false. This increases the efficiency of the program.

You can use the logical operators AND and OR to combine conditions in an IF expression. You use the AND operator when both conditions must be true, and you use the OR operator when only one of the conditions must be true. An optional final ELSE statement can be used at the end of a series of IF-THEN/ELSE statements. The statement following the final ELSE executes if none of the IF expressions is true.

Use a DO group with an IF-THEN or an ELSE statement when multiple statements must be executed based on one condition. The DO group consists of a DO statement, the SAS statements to be executed, and an END statement. Each DO statement must have a corresponding END statement.

```
IF expression THEN
    DO;
        executable statements
    END;
```

ELSE IF *expression* **THEN**
    DO;
        executable statements
END;

Truncation can occur when new variables are assigned values within conditional program statements. During compilation, SAS creates a variable in the PDV the first time it encounters the variable in the program. If this is in conditional code, be sure that it is created with a length long enough to store all possible values. It is a best practice to use a LENGTH statement to explicitly define the length.

Sample Programs

Creating Variables by Using Functions

```
data work.comp;
   set orion.sales;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;

proc print data=work.comp;
   var Employee_ID First_Name Last_Name
       Salary Bonus Compensation BonusMonth;
run;
```

Assigning Values Conditionally

```
data work.comp;
   set orion.sales;
   if Job_Title='Sales Rep. IV' then
      Bonus=1000;
   if Job_Title='Sales Manager' then
      Bonus=1500;
   if Job_Title='Senior Sales Manager' then
      Bonus=2000;
   if Job_Title='Chief Sales Officer' then
      Bonus=2500;
run;

proc print data=work.comp;
   var Last_Name Job_Title Bonus;
run;
```

Using Compound Conditions

```
data work.comp;
   set orion.sales;
   if Job_Title='Sales Rep. III' or
      Job_Title='Sales Rep. IV' then
      Bonus=1000;
   else if Job_Title='Sales Manager' then
      Bonus=1500;
   else if Job_Title='Senior Sales Manager' then
      Bonus=2000;
   else if Job_Title='Chief Sales Officer' then
      Bonus=2500;
   else Bonus=500;
run;

proc print data=work.comp;
   var Last_Name Job_Title Bonus;
run;
```

Using IF-THEN/ELSE Statements

```
data work.bonus;
   set orion.sales;
   if Country='US' then Bonus=500;
   else Bonus=300;
run;

proc print data=work.bonus;
   var First_Name Last_Name Country Bonus;
run;
```

Creating Two Variables Conditionally

```
data work.bonus;
   set orion.sales;
   if Country='US' then
      do;
         Bonus=500;
         Freq='Once a Year';
      end;
   else if Country='AU' then
      do;
         Bonus=300;
         Freq='Twice a Year';
      end;
run;
proc print data=work.bonus;
   var First_Name Last_Name Country Bonus Freq;
run;
```

Adjusting the Program

```
data work.bonus;
   set orion.sales;
   length Freq $ 12;
   if Country='US' then
      do;
         Bonus=500;
         Freq='Once a Year';
      end;
   else if Country='AU' then
      do;
         Bonus=300;
         Freq='Twice a Year';
      end;
run;

proc print data=work.bonus;
   var First_Name Last_Name Country Bonus Freq;
run;

data work.bonus;
   set orion.sales;
   length Freq $ 12;
   if Country='US' then
      do;
         Bonus=500;
         Freq='Once a Year';
      end;
   else do;
         Bonus=300;
         Freq='Twice a Year';
      end;
run;

proc print data=work.bonus;
   var First_Name Last_Name Country
       Bonus Freq;
run;
```

SAS Programming 1: Essentials

Close