Print

Summary of Lesson 6: Reading SAS Data Sets

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.

Reading a SAS Data Set

You use a DATA step to create a new SAS data set from an existing SAS data set. The DATA step begins with a DATA statement, which provides the name of the SAS data set to create. Include a SET statement to name the existing SAS data set to be read in as input.

You use the WHERE statement to subset the input data set by selecting only the observations that meet a particular condition. To subset based on a SAS date value, you can use a SAS date constant in the WHERE expression. SAS automatically converts a date constant to a SAS date value.

```
DATA output-SAS-data-set;
    SET input-SAS-data-set;
    WHERE where-expression;
RUN;
```

You use an assignment statement to create a new variable. The assignment statement evaluates an expression and assigns the resulting value to a new or existing variable. The expression is a sequence of operands and operators. If the expression includes arithmetic operators, SAS performs the numeric operations based on priority, as in math equations. You can use parentheses to clarify or alter the order of operations.

```
variable=expression;
```

Customizing a SAS Data Set

By default, the SET statement reads all of the observations and variables from the input data set and writes them to the output data set. You can customize the new data set by selecting only the observations and variables that you want to include. You can use a WHERE statement to select the observations, as long as the variables included in the condition come from the input data set. You can use a DROP statement to list the variables to exclude from the new data set, or use a KEEP statement to list the variables to include. If you use a KEEP statement, you must include every variable to be written, including any new variables.

```
DROP variable-list;
KEEP variable-list;
```

SAS processes the DATA step in two phases: the compilation phase and the execution phase.

You can subset the original data set with a WHERE statement for variables that are defined in the input data set, and a subsetting IF statement for new variables that are created in the DATA step. Remember that, although IF expressions are similar to WHERE expressions, you cannot use special WHERE operators in IF expressions.

```
IF expression;
```

To subset observations in a PROC step, you must use a WHERE statement. You cannot use a subsetting IF statement in a PROC step. To subset observations in a DATA step, you can always use a subsetting IF statement. However, a WHERE statement can make your DATA step more efficient because it subsets on input.

Adding Permanent Attributes

When you use the LABEL statement in a DATA step, SAS permanently associates the labels to the variables by storing the labels in the descriptor portion of the data set. Using a FORMAT statement in a DATA step permanently associates formats with variables. The format information is also stored in the descriptor portion of the data set. You can use PROC CONTENTS to view the label and format information. PROC PRINT does not display permanent labels unless you use the LABEL or SPLIT= option.

LABEL *variable='label'*
variable='label'
... ;

FORMAT *variable(s) format ...*;

Sample Programs

Subsetting Observations in the DATA Step

```
proc print data=orion.sales;
run;

data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
run;

proc print data=work.subset1;
run;
```

Subsetting Observations and Creating a New Variable

```
data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep' and
         Hire_Date<'01jan2000'd;
   Bonus=Salary*.10;
run;

proc print data=work.subset1 noobs;
   var First_name Last_Name Salary
       Job_Title Bonus Hire_Date;
   format Hire_Date date9.;
run;
```

Subsetting Variables in a DATA Step: DROP and KEEP

```
data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   Bonus=Salary*.10;
   drop Employee_ID Gender Country Birth_Date;
run;

proc print data=work.subset1;
run;

data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   Bonus=Salary*.10;
   keep First_Name Last_Name Salary Job_Title Hire_Date Bonus;
run;
```

```
proc print data=work.subset1;
run;
```

Selecting Observations by Using the Subsetting IF Statement

```
data work.auemps;
   set orion.sales;
   where Country='AU';
   Bonus=Salary*.10;
   if Bonus>=3000;
run;

proc print data=work.auemps;
run;
```

Adding Permanent Labels to a SAS Data Set

```
data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   Bonus=Salary*.10;
   label Job_Title='Sales Title'
         Hire_Date='Date Hired';
   drop Employee_ID Gender Country Birth_Date;
run;

proc contents data=work.subset1;
run;

proc print data=work.subset1 label;
run;
```

Adding Permanent Formats to a SAS Data Set

```
data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   Bonus=Salary*.10;
   label Job_Title='Sales Title'
         Hire_Date='Date Hired';
   format Salary Bonus dollar12.
          Hire_Date ddmmyy10.;
   drop Employee_ID Gender Country Birth_Date;
run;


proc contents data=work.subset1;
run;

proc print data=work.subset1 label;
run;
```


Close