Print

Summary of Lesson 5: Formatting Data Values

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 Formats

A format is an instruction that tells SAS how to display data values in output reports. You can add a FORMAT statement to a PROC PRINT step to specify temporary SAS formats that control how values appear in the report. There are many existing SAS formats that you can use. Character formats begin with a dollar sign, but numeric formats do not.

FORMAT *variable(s) format*;

SAS stores date values as the number of days between January 1, 1960, and a specific date. To make the dates in your report recognizable and meaningful, you must apply a SAS date format to the SAS date values.

Creating and Applying User-Defined Formats

You can create your own user-defined formats. When you create a user-defined format, you don't associate it with a particular variable or data set. Instead, you create it based on values that you want to display differently. The formats will be available for the remainder of your SAS session. You can apply user-defined formats to a specific variable in a PROC PRINT step.

You use the FORMAT procedure to create a format. You assign a format name that can have up to 32 characters. The name of a character format must begin with a dollar sign, followed by a letter or underscore, followed by letters, numbers, and underscores. Names for numeric formats must begin with a letter or underscore, followed by letters, numbers, and underscores. A format name cannot end in a number and cannot be the name of a SAS format.

You use a VALUE statement in a PROC FORMAT step to specify the way that you want the data values to appear in your output. You define value-range sets to specify the values to be formatted and the formatted values to display instead of the stored value or values. The value portion of a value-range set can include an individual value, a range of values, a list of values, or a keyword. The keyword OTHER is used to define a value to display if the stored data value does not match any of the defined value-ranges.

PROC FORMAT;
VALUE *format-name value-or-range1='formatted-value1'*
value-or-range2='formatted-value2'
...;
RUN;

When you define a numeric format, it is often convenient to use numeric ranges in the value-range sets. Ranges are inclusive by default. To exclude the endpoints, use a less-than symbol after the low end of the range or before the high end.

The LOW and HIGH keywords are used to define a continuous range when the lowest and highest values are not known. Remember that for character values, the LOW keyword treats missing values as the lowest possible values. However, for numeric values, LOW does not include missing values.

Sample Programs

Applying Temporary Formats

```
proc print data=orion.sales label noobs;
   where Country='AU' and
```

```
          Job_Title contains 'Rep';
   label Job_Title='Sales Title'
          Hire_Date='Date Hired';
   var Last_Name First_Name Country Job_Title
       Salary Hire_Date;
run;

proc print data=orion.sales label noobs;
   where Country='AU' and
          Job_Title contains 'Rep';
   label Job_Title='Sales Title'
          Hire_Date='Date Hired';
   format Hire_Date mmddyy10. Salary dollar8.;
   var Last_Name First_Name Country Job_Title
       Salary Hire_Date;
run;
```

Specifying a User-Defined Format for a Character Variable

```
proc format;
   value $ctryfmt 'AU'='Australia'
                  'US'='United States'
                  other='Miscoded';
run;

proc print data=orion.sales label;
   var Employee_ID Job_Title Salary
       Country Birth_Date Hire_Date;
   label Employee_ID='Sales ID'
          Job_Title='Job Title'
          Salary='Annual Salary'
          Birth_Date='Date of Birth'
          Hire_Date='Date of Hire';
   format Salary dollar10.
          Birth_Date Hire_Date monyy7.
          Country $ctryfmt.;
run;
```

Specifying a User-Defined Format for a Numeric Variable

```
proc format;
   value tiers low-<50000='Tier1'
               50000-100000='Tier2'
               100000<-high='Tier3';
run;

proc print data=orion.sales;
   var Employee_ID Job_Title Salary
       Country Birth_Date Hire_Date;
   format Birth_Date Hire_Date monyy7.
          Salary tiers.;
run;
```


Close