

Lesson 3: Working with SAS Code

Summary

Main Points

Understanding the Structure of SAS Programs

- SAS compiles and executes each DATA or PROC step independently based on step boundaries, such as a RUN statement, a QUIT statement, or the beginning of a new DATA or PROC step.
- The Enhanced Editor illustrates the separation between steps with a horizontal line.

Understanding the Syntax of SAS Programs

- SAS has specific syntax; however, SAS statements are free-format. Each statement must end with a semicolon.
- Although SAS does not require it, you can follow some simple formatting guidelines to make your SAS code easier to read and use.

Adding Comments to SAS Programs

- You can use comments in your code to make it easier for you and others to read and interpret it. A comment is text in your program that SAS ignores during processing. You might use comments to document the purpose of the program, explain segments of the program, or mark SAS code as non-executing text. Comments can be especially useful for testing your code.
- You can use a block comment, which can appear anywhere a single blank can appear and can contain semicolons or unmatched quotation marks. Block comments can also span several lines. You should avoid placing block comments in the first or second columns.
- You can create a comment by beginning any SAS statement with an asterisk, which indicates to SAS that all of the following text up to the next semicolon is a comment. Comments in this form are complete statements, and they can't contain semicolons or unmatched quotation marks.

Diagnosing and Correcting Syntax Errors

- A syntax error is code that doesn't conform to the rules of the SAS language. Common syntax errors include misspelled keywords, unmatched quotation marks, missing semicolons, and invalid operators. The Enhanced Editor highlights potential errors in your SAS code by displaying it in red text.
- SAS code can also contain logic errors.

Lesson 3: Working with SAS Code

- When you submit a program, SAS attempts to interpret any syntax errors that it encounters. If it can't interpret the error, SAS stops processing the step and writes an error message in the log. You must correct your code to fix errors that appear in the log.

Diagnosing and Correcting Unbalanced Quotation Marks

- Unbalanced quotation marks are a syntax error that might not generate a log message. Instead, the main symptom might be a message such as "PROC PRINT running" in the title bar of an active programming window.
- To correct unbalanced quotation marks, you must cancel the submitted statement, correct the error, and resubmit the code. You can use the **Break** tool to cancel submitted statements.

Sample Code

Unformatted Code

```
data work.newsalesemps;length First_Name $ 12
      Last_Name $ 18 Job_Title $ 25; infile
'newemployees.csv' dlm=',';
input First_Name $ Last_Name $ Job_Title $
Salary; run;
title 'New Sales Employees';
proc print data=work.newsalesemps; run;
proc means data=work.newsalesemps; class Job_Title;var
Salary; run;
title;
```

Lesson 3: Working with SAS Code

Formatted Code

```
data work.newsalesemps;
    length First_Name $ 12
           Last_Name $ 18 Job_Title $ 25;
    infile 'newemployees.csv' dlm=',';
    input First_Name $ Last_Name $
          Job_Title $ Salary;
run;

title 'New Sales Employees';
proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
    class Job_Title;
    var Salary;
run;

title;
```

Comments in Code

```
/*
This is a comment.
This title statement won't be processed.
title 'New Sales Employees';
*/

*****
This is a comment;
proc print data=work.newsalesemps;
run;
*This is another comment;
```