Enhancing reports

Statistics 426: SAS Programming

Module 5

2021

Formats. . . +

Here we will start by revisiting the DATA step to see where to apply formats, informats, titles, and many more things to give reports the desired appearance

General form of DATA step with SET

SET allows to read in an existing SAS dataset as the basis of the new dataset being created; the new dataset specified in the DATA statement is created using an existing SAS dataset specified in the SET statement, often by subsetting it based on conditions

```
DATA datasetname;
SET input-datasetname;
WHERE where-expression;
KEEP variable-list;
LABEL variable1='Label'
variable2='Label'
... ;
FORMAT variable1 format.
variable2 format.
... ;
RUN;
```

Formats

Formats are variable attributes that affect the way data values are written. SAS software offers a variety of character, numeric, and date and time formats. You can also create and store your own formats. To write values out using a particular form, you select the appropriate format

General form of FORMAT statement

```
FORMAT variable format.;
```

Assigns formats to variable values. Format is an instruction that SAS uses to write data values. Using a FORMAT statement in a DATA step permanently associates formats with variables by storing the format in the descriptor portion of the SAS dataset (access with PROC CONTENTS)

Form of FORMAT statement

```
<$>format<w>.<d>
```

`$`: indicates character format
`format`: name of the SAS- or user-defined format

`w`: specifies the total format width including decimal places and special characters
`d`: specifies the number of decimal places in numeric formats

Basic formats

Format	Type	Definition
$w.	character	standard data
w.d	numeric	standard data
COMMAw.d	numeric	comma separates every 3rd digit and period separates decimal fraction
COMMAXw.d	numeric	decimal separates every 3rd digit, comma separates decimal fraction
DOLLARw.d	numeric	leading $ sign, comma separates every 3rd digit, period separates decimal fraction
EUROXw.d	numeric	leading € sign, decimal separates every 3rd digit, comma separates decimal fraction

Examples

Format	Stored	Display
$4.	Programming	Prog
$11.	Programming	Programming
12.	27134.2864	27134
12.2	27134.2864	27134.29
comma12.2	27134.2864	27,134.29
commax12.2	27134.2864	27.134,29
dollar12.2	27134.2864	$27,134.29
eurox12.2	27134.2864	€27.134,29
dollar12.2	27134.2864	$27,134.29
dollar9.2	27134.2864	$27134.29
dollar8.2	27134.2864	27134.29

INFORMAT

Whereas formats write values out by using some particular form, informats read data values *in* certain forms into *standard* SAS values. Informats determine how data values are read into a SAS data set. You must use informats to read numeric values that contain letters or other special characters

An INFORMAT statement in a DATA step permanently associates an informat with a variable. You can specify standard SAS informats or user-written informats, previously defined in PROC FORMAT. A single INFORMAT statement can associate the same informat with several variables, or it can associate different informats with different variables. If a variable appears in multiple INFORMAT statements, SAS uses the informat that is assigned last

They can be specified in either the INPUT line of the DATA step or in an INFORMAT statement in the DATA (or PROC) step

Standard and Nonstandard Numeric Data

Standard numeric data values can contain only numbers, decimal points, numbers in scientific or E-notation (2.3E4, for example), or plus or minus signs

Example: 58 -23 67.23 00.99 5.67e5 1.2e-2

Nonstandard numeric data include values that contain special characters, such as percent signs (%), dollar signs ($), and commas (,), date and time values, data in fraction, integer binary, real binary, and hexadecimal forms

Example: 5,823 (23) $67.23 01/12/1999 12May2006

How SAS Treats Variables with the INFORMAT Statement

Informats that are associated with variables behave like informats that are used with modified list input; the variables by using the scanning feature of list input, but applies the informat. In modified list input, SAS
- does not use the value of `w` in an informat to specify column positions or input field widths in an external file
- uses the value of `w` in an informat to specify the length of previously undefined character variables but ignores the value of `w` in numeric informats
- uses the value of `d` in an informat in the same way it usually does for numeric informats
- treats blanks that are embedded as input data as delimiters unless you change their status with a DLM= option specification in an INFILE statement. If you have coded the INPUT statement to use another style of input, such as formatted input or column input, that style of input is not used when you use INFORMAT

Format and informat example I

```
data new;
    infile 'S:\Courses\stat-renaes\stat426\data1\sales.csv' dsd missover;
    input Employee_ID First_Name $ Last_Name $
         Gender $ Salary Job_Title $ Country $
         Birth_Date :date.
         Hire_Date :mmddyy.;
    format salary dollar12.;
run;

proc print data=new;
run;
```

Format and informat example I proc print

The SAS System

Obs	Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title	Country	Birth_Date	Hire_Date
1	120102	Tom	Zhou	M	$108,255	Sales Ma	AU	3510	10744
2	120103	Wilson	Dawes	M	$87,975	Sales Ma	AU	-3996	5114
3	120121	Irenie	Elvish	F	$26,600	Sales Re	AU	-5630	5114
4	120122	Christin	Ngan	F	$27,475	Sales Re	AU	-1984	6756
5	120123	Kimiko	Hotstone	F	$26,190	Sales Re	AU	1732	9405
6	120124	Lucian	Daymond	M	$26,480	Sales Re	AU	-233	6999
7	120125	Fong	Hofmeist	M	$32,040	Sales Re	AU	-1852	6999
8	120126	Satyakam	Denny	M	$26,780	Sales Re	AU	10490	17014
9	120127	Sharryn	Clarkson	F	$28,100	Sales Re	AU	6943	14184
10	120128	Monica	Kletschk	F	$30,890	Sales Re	AU	9691	17106

Format and informat example I log

[]figures/informat log.png)

Format example

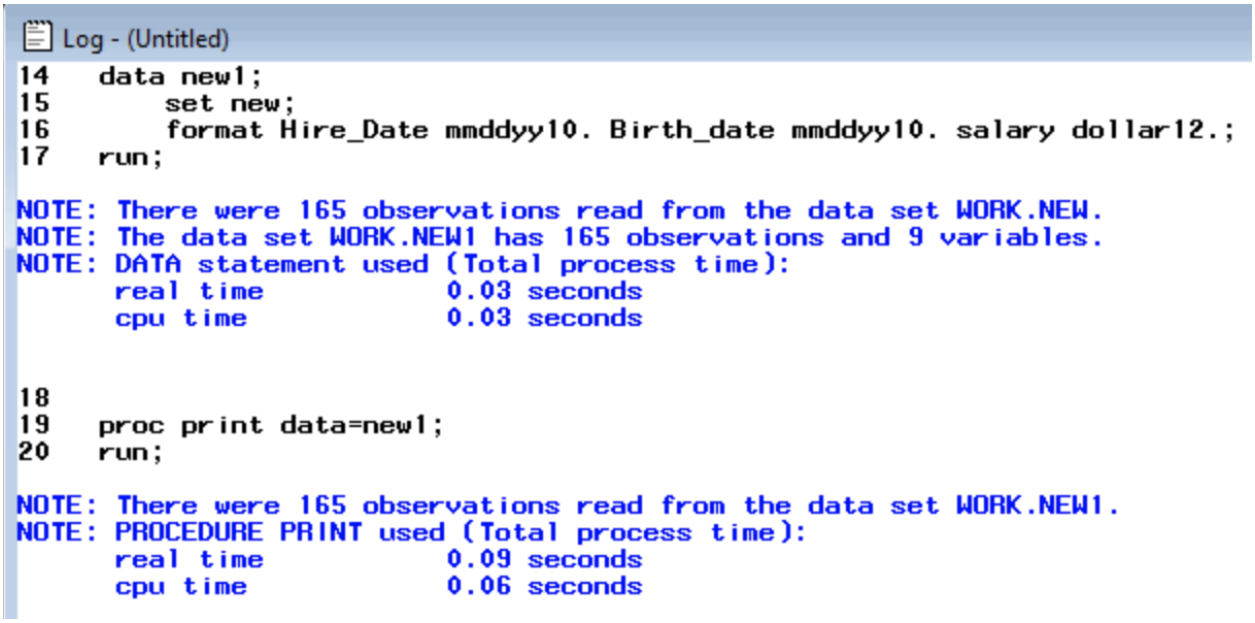
```
data new1;
    set new;
    format Hire_Date mmddyy10. Birth_date mmddyy10. salary dollar12.;
run;

proc print data=new1;
run;
```

Format example proc print

The SAS System

Obs	Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title	Country	Birth_Date	Hire_Date
1	120102	Tom	Zhou	M	$108,255	Sales Ma	AU	08/11/1969	06/01/1989
2	120103	Wilson	Dawes	M	$87,975	Sales Ma	AU	01/22/1949	01/01/1974
3	120121	Irenie	Elvish	F	$26,600	Sales Re	AU	08/02/1944	01/01/1974
4	120122	Christin	Ngan	F	$27,475	Sales Re	AU	07/27/1954	07/01/1978
5	120123	Kimiko	Hotstone	F	$26,190	Sales Re	AU	09/28/1964	10/01/1985
6	120124	Lucian	Daymond	M	$26,480	Sales Re	AU	05/13/1959	03/01/1979
7	120125	Fong	Hofmeist	M	$32,040	Sales Re	AU	12/06/1954	03/01/1979
8	120126	Satyakam	Denny	M	$26,780	Sales Re	AU	09/20/1988	08/01/2006
9	120127	Sharryn	Clarkson	F	$28,100	Sales Re	AU	01/04/1979	11/01/1998
10	120128	Monica	Kletschk	F	$30,890	Sales Re	AU	07/14/1986	11/01/2006

Format example log



PROC FORMAT

PROC FORMAT is used to create user-defined formats not already available in SAS; it creates a format that specifies character strings to use to print variable values

Character user-defined format

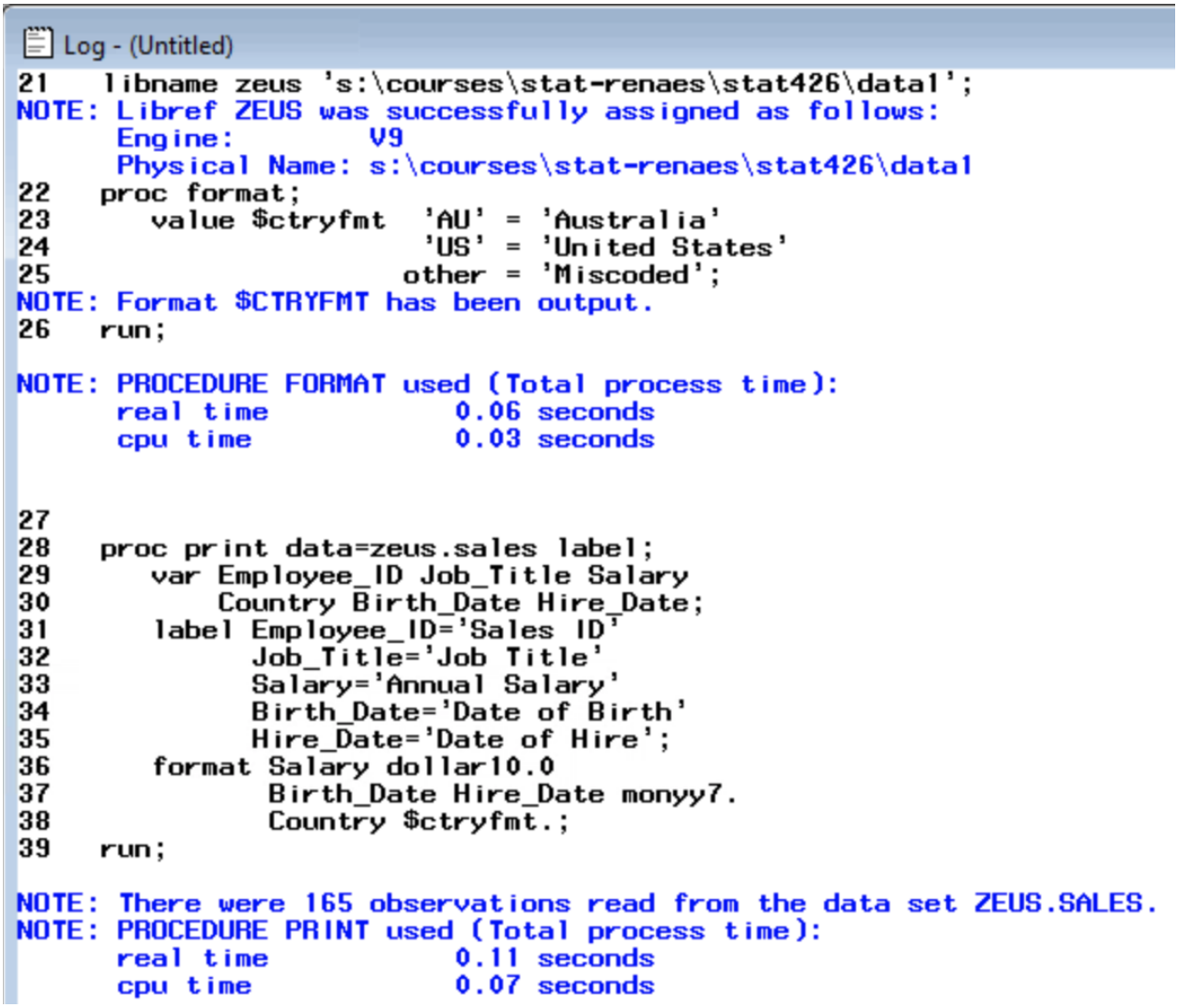
```
libname zeus 's:\courses\stat-renaes\stat426\data1';

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

proc print data=zeus.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 dollar10.0
          Birth_Date Hire_Date monyy7.
          Country $ctryfmt.;
run;
```

Character user-defined format print

The SAS System

Obs	Sales ID	Job Title	Annual Salary	Country	Date of Birth	Date of Hire
1	120102	Sales Manager	$108,255	Australia	AUG1969	JUN1989
2	120103	Sales Manager	$87,975	Australia	JAN1949	JAN1974
3	120121	Sales Rep. II	$26,600	Australia	AUG1944	JAN1974
4	120122	Sales Rep. II	$27,475	Australia	JUL1954	JUL1978
5	120123	Sales Rep. I	$26,190	Australia	SEP1964	OCT1985
6	120124	Sales Rep. I	$26,480	Australia	MAY1959	MAR1979
7	120125	Sales Rep. IV	$32,040	Australia	DEC1954	MAR1979
8	120126	Sales Rep. II	$26,780	Australia	SEP1988	AUG2006
9	120127	Sales Rep. II	$28,100	Australia	JAN1979	NOV1998
10	120128	Sales Rep. IV	$30,890	Australia	JUL1986	NOV2006

Character user-defined format log



```
Log - (Untitled)
21    libname zeus 's:\courses\stat-renaes\stat426\data1';
NOTE: Libref ZEUS was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\courses\stat-renaes\stat426\data1
22    proc format;
23       value $ctryfmt   'AU' = 'Australia'
24                        'US' = 'United States'
25                     other = 'Miscoded';
NOTE: Format $CTRYFMT has been output.
26    run;

NOTE: PROCEDURE FORMAT used (Total process time):
      real time             0.06 seconds
      cpu time              0.03 seconds


27
28    proc print data=zeus.sales label;
29       var Employee_ID Job_Title Salary
30           Country Birth_Date Hire_Date;
31       label Employee_ID='Sales ID'
32             Job_Title='Job Title'
33             Salary='Annual Salary'
34             Birth_Date='Date of Birth'
35             Hire_Date='Date of Hire';
36       format Salary dollar10.0
37              Birth_Date Hire_Date monyy7.
38              Country $ctryfmt.;
39    run;

NOTE: There were 165 observations read from the data set ZEUS.SALES.
NOTE: PROCEDURE PRINT used (Total process time):
      real time             0.11 seconds
      cpu time              0.07 seconds
```

Numeric user-defined format

```
proc format;
   value tiers  20000-49999  = 'Tier 1'
                50000-99999  = 'Tier 2'
               100000-250000 = 'Tier 3';
run;

proc print data=zeus.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 Birth_Date Hire_Date monyy7.
          Salary tiers.;
```

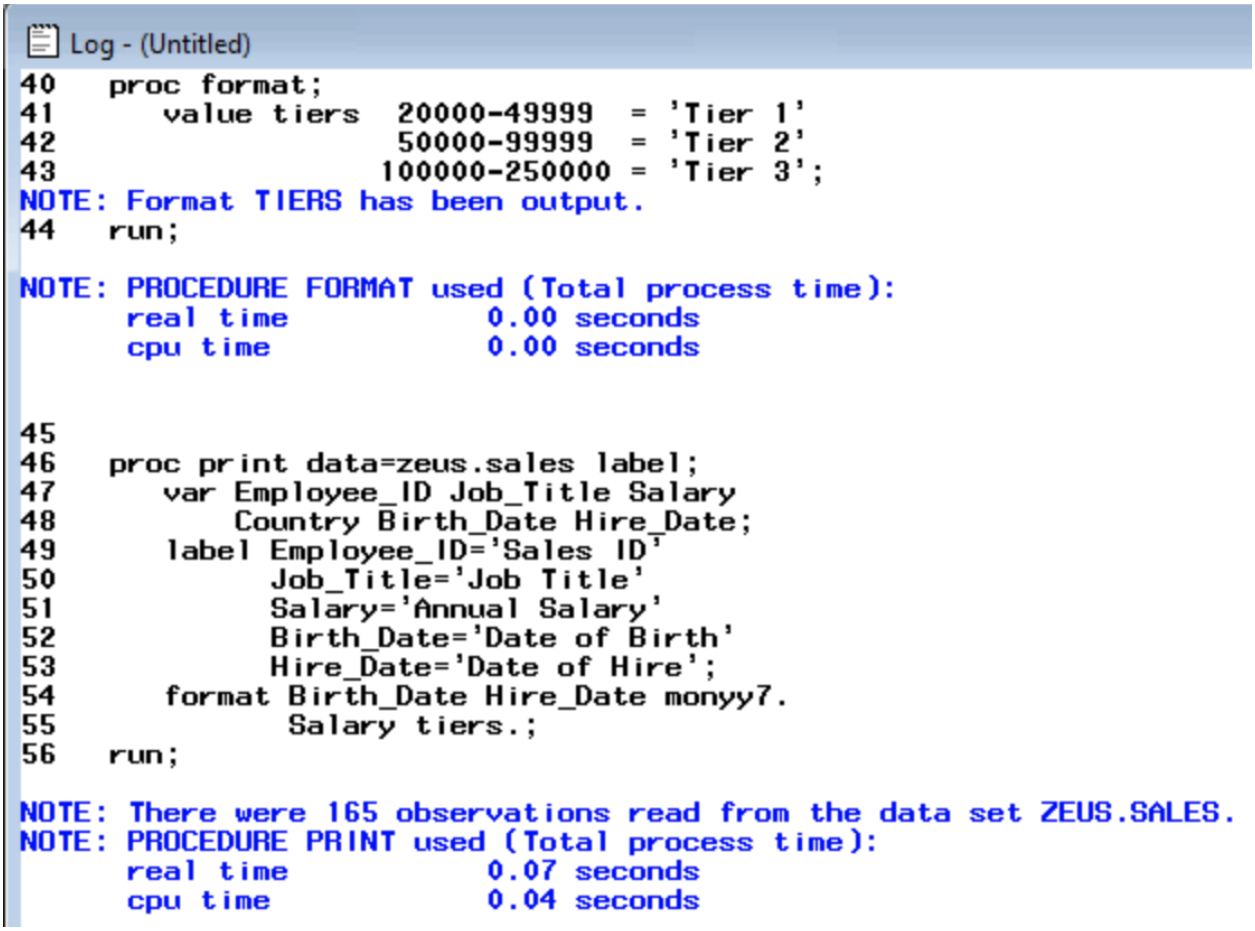
```
run;
```

Numeric user-defined format print

The SAS System

Obs	Sales ID	Job Title	Annual Salary	Country	Date of Birth	Date of Hire
1	120102	Sales Manager	Tier 3	AU	AUG1969	JUN1989
2	120103	Sales Manager	Tier 2	AU	JAN1949	JAN1974
3	120121	Sales Rep. II	Tier 1	AU	AUG1944	JAN1974
4	120122	Sales Rep. II	Tier 1	AU	JUL1954	JUL1978
5	120123	Sales Rep. I	Tier 1	AU	SEP1964	OCT1985
6	120124	Sales Rep. I	Tier 1	AU	MAY1959	MAR1979
7	120125	Sales Rep. IV	Tier 1	AU	DEC1954	MAR1979
8	120126	Sales Rep. II	Tier 1	AU	SEP1988	AUG2006
9	120127	Sales Rep. II	Tier 1	AU	JAN1979	NOV1998
10	120128	Sales Rep. IV	Tier 1	AU	JUL1986	NOV2006

Numeric user-defined format log



```
Log - (Untitled)
40    proc format;
41       value tiers  20000-49999  = 'Tier 1'
42                    50000-99999  = 'Tier 2'
43                   100000-250000 = 'Tier 3';
NOTE: Format TIERS has been output.
44    run;

NOTE: PROCEDURE FORMAT used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds


45
46    proc print data=zeus.sales label;
47       var Employee_ID Job_Title Salary
48           Country Birth_Date Hire_Date;
49       label Employee_ID='Sales ID'
50             Job_Title='Job Title'
51             Salary='Annual Salary'
52             Birth_Date='Date of Birth'
53             Hire_Date='Date of Hire';
54       format Birth_Date Hire_Date monyy7.
55              Salary tiers.;
56    run;

NOTE: There were 165 observations read from the data set ZEUS.SALES.
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.07 seconds
      cpu time            0.04 seconds
```

Dates in SAS

SAS stores date values as numeric values. A SAS date value is stored as the number of days between January 1, 1960 and a specific date. Dates after 1/1/1960 are positive numbers and dates before 1/1/1960 are negative numbers

Date formats

Date	Format	Stored	Display
1/1/1960	MMDDYY6.	0	010160
1/1/1960	MMDDYY8.	0	01/01/60
1/1/1960	MMDDYY10.	0	01/01/1960
12/31/60	DDMMYY6.	365	311260
12/31/60	DDMMYY8.	365	31/01/60
12/31/60	DDMMYY10.	365	31/01/1960
12/31/59	DATE7.	-1	31Dec59
12/31/59	DATE9.	-1	31Dec1959
1/1/60	WORDDATE.	0	January 1, 1960
1/1/60	WEEKDATE.	0	Friday, January 1, 1960
1/1/60	MONYY.	0	Jan1960
1/1/60	YEAR4.	0	1960

LABEL statement

Sometimes, you just have to change the default label in a report. Adding permanent attributes is possible by using LABEL and FORMAT statements. The descriptor portion of a SAS dataset (PROC CONTENTS) stores variable attributes including name, type and length of the variable. Labels and formats can also be stored in the descriptor portion of the SAS dataset. Use of LABEL assigns a descriptive label to the variable names

General form of LABEL

```
LABEL variable='Label'
variable='Label'
... ;
```

A label can:
- have up to 256 characters
- any number of variables can be associated with a single LABEL statement
- using a LABEL statement in a DATA step permanently associated label with the variables by storing the label in the descriptor portion of the dataset
- to see the labels in PROC PRINT, you need to specify a label option

```
PROC PRINT data=SAS-dataset-name label;
run;
```

LABEL example

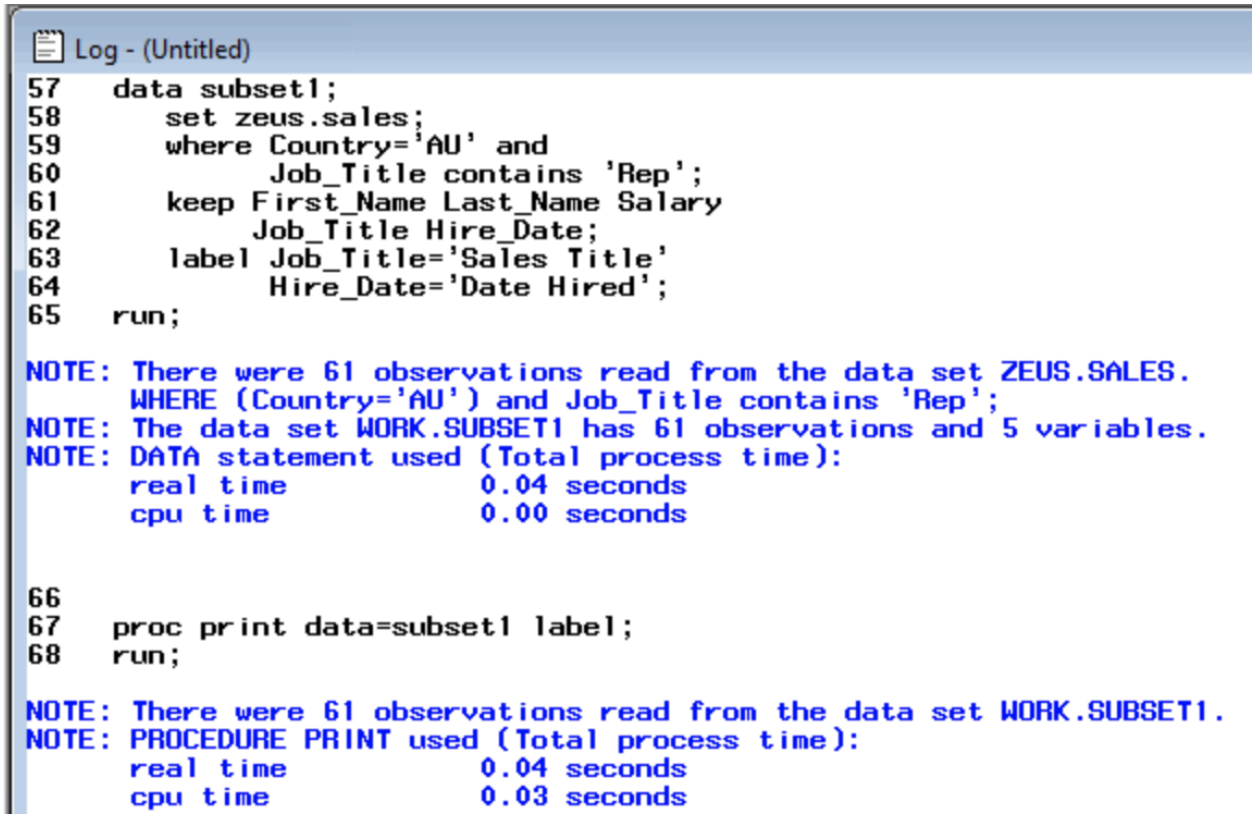
```
data subset1;
   set zeus.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   keep First_Name Last_Name Salary
        Job_Title Hire_Date;
   label Job_Title='Sales Title'
         Hire_Date='Date Hired';
run;

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

LABEL example print

The SAS System

Obs	First_Name	Last_Name	Salary	Sales Title	Date Hired
1	Irenie	Elvish	26600	Sales Rep. II	5114
2	Christina	Ngan	27475	Sales Rep. II	6756
3	Kimiko	Hotstone	26190	Sales Rep. I	9405
4	Lucian	Daymond	26480	Sales Rep. I	6999
5	Fong	Hofmeister	32040	Sales Rep. IV	6999
6	Satyakam	Denny	26780	Sales Rep. II	17014
7	Sharryn	Clarkson	28100	Sales Rep. II	14184
8	Monica	Kletschkus	30890	Sales Rep. IV	17106
9	Alvin	Roebuck	30070	Sales Rep. III	9405
10	Kevin	Lyon	26955	Sales Rep. I	16922

LABEL example log



Length statement in a DATA step

This is used in a DATA step when reading in an external file or using the CARDS (or DATALINES) statements (whenever you use the INPUT statement, you can use the LENGTH statement)

A variable's length (the number of bytes used to store it) is related to its type
- Character variables can be up to 32,767 bytes long
- All numeric variables have a default length of 8. Numeric values (no matter how many digits they contain) are stored as floating-point numbers in 8 bytes of storage, unless you specify a different length

General form of LENGTH in DATA step

`LENGTH var<\$>n...;`

`var`: name of variable
`<\$>`: used when variable is character
`n`: the number of bytes for that variable

LENGTH example

```
data new;
   length First_Name $ 12 Last_Name $ 18
          Gender $ 1 Job_Title $ 25
          Country $ 2;
    infile 'S:\Courses\stat-renaes\stat426\data1\sales.csv' dsd missover;
   input Employee_ID First_Name $ Last_Name $
         Gender $ Salary Job_Title $ Country $
```

```
         Birth_Date :date.
         Hire_Date :mmddyy.;
run;

proc print data=new;
run;
```

Titles

Title statement:

The TITLE statement specifies title lines for SAS output. General form:
`TITLEn 'Text';`
- Titles appear at the top of the output page
- The default title is `The SAS System`
- The value of n can take on values from 1-10
- An unnumbered TITLE is equivalent to TITLE1
- Titles remain in effect until canceled, changed or the SAS session ends
- A null title statement will cancel all titles (TITLE;)

Footnotes

The FOOTNOTE statement specifies footnote lines for SAS output. General form:
`FOOTNOTEn 'Text';`
- Footnotes appear at the bottom of the output page
- The value of n can take on values from 1-10
- An unnumbered FOOTNOTE is equivalent to FOOTNOTE1
- Footnotes remain in effect until canceled, changed or the SAS session ends
- A null footnote statement will cancel all footnotes (FOOTNOTE;)

Enhancing Reports

One aesthetic feature with SAS is to enhance reports with the ODS (Output Destination Service) statement. Enhance reports with labels, formats, global options, and output styles (through ODS statements)

ODS general form

`ODS destination FILE='filename.ext' ...;`

`destination`: type of file you are writing your report to (RTF, PDF, HTML, etc.)
`filename.ext`:‘ filename (and also preceding the filename is usually the physical address of your file)

Use a STYLE= option in the ODS statement to specify a style definition.

```
ODS destination FILE='filename.ext'
STYLE=style-definition;
⋮
ODS destination CLOSE;
```

`style-definition`: describes how to display the presentation aspects such as color and fonts of SAS output. Note that STYLE= option cannot be used with the LISTING destination