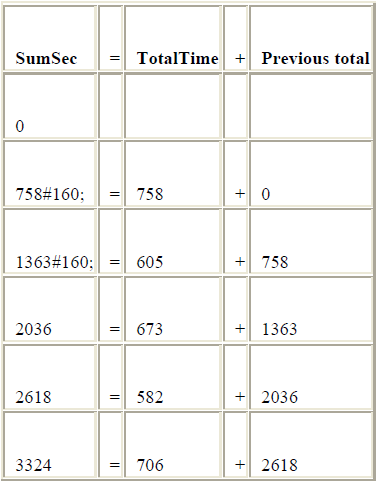
**Chapter X – Creating and Managing Variables**

1. Accumulating Totals

* Basic code

*variable+expression***;**

* + - * *variable* specifies the name of the accumulator variable. This variable must be numeric. The variable is automatically set to 0 before the first observation is read. The variable's value is retained from one DATA step execution to the next.
      * *expression* is any valid SAS expression.

If the *expression* produces a missing value, the sum statement ignores it.

Eg:

**DATA** clinic.stress;

**INFILE** tests;

**INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 MaxHR 31-33

RecHR 35-37 TimeMin 39-40 TimeSec 42-43

Tolerance $ 45;

TotalTime=(timemin\*60)+timesec;

SumSec+totaltime;

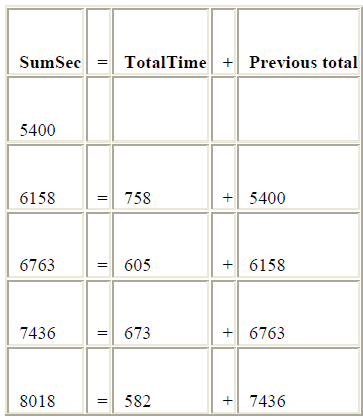
**RUN**;

* Initializing Sum Variables

Basic code:

**RETAIN** *variable* <*initial;-value*>;

* + - * + *variable* is a variable whose values you want to retain
        + *initial-value* specifies an initial value (numeric or character) for the preceding variable.

 Eg:

**DATA** clinic.stress;

**INFILE** tests;

**INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 MaxHR 31-33

RecHR 35-37 TimeMin 39-40 TimeSec 42-43

Tolerance $ 45;

TotalTime=(timemin\*60)+timesec;

**RETAIN** SumSec 5400;

sumsec+totaltime;

**RUN**;

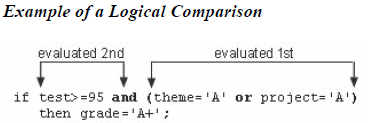
1. **IF** –**THEN/ ELSE** (**PUT**) statement (Assign condition)

For greater efficiency, construct your **IF-THEN/ELSE** statements with conditions of decreasing probability.

* Basic code:

**IF** *expression* **THEN** *statement***;**

**ELSE** *expression* **THEN** *statement***;**

* + - * + *expression* is any valid SAS expression.
        + *statement* is any executable SAS statement
* Logical comparisons that are enclosed in parentheses are evaluated as true or false before they are compared to other expressions.
* 在对比中，1 或者其他numerical value = True

0或者missing (.) = false

* Using ELSE statements with IF-THEN statements can save resources:
* Using IF-THEN statements *without* the ELSE statement causes SAS to evaluate all IF-THEN statements
* Using IF-THEN statements *with* the ELSE statement causes SAS to execute IF-THEN statements until it encounters the first true statement. Subsequent IF-THEN statements are not evaluated.
* **PUT** statement to test the conditional logic.

Eg:

**IF** totaltime>800 **THEN** TestLength='Long';

**ELSE** **IF** 750<=totaltime<=800 **THEN** TestLength='Normal';

**ELSE** **PUT** 'NOTE: Check this Length: ' totaltime=;

1. **LENGTH** statement

* Specify a length (the number of bytes) for the selected variable before the first value is referenced elsewhere in the DATA step.
* If that is a character variable, you must follow the variable name with a dollar sign ($).
* Basic code:

**LENGTH** *variable(s)* <$> *length***;**

* *variable(s)* names the variable(s) to be assigned a length
* $ is specified if the variable is a character variable
* *length* is an integer that specifies the length of the variable.

Eg:

**LENGTH** Type $ 8;

**LENGTH** Address1 Address2 Address3 $200;

**LENGTH** FirstName $12 LastName $16;

1. **DELETE** statement

The DELETE statement stops the processing of the current observation.

* Basic code:

**IF** *expression* **THEN DELETE;**

If the expression is

* *true*, the DELETE statement executes, and control returns to the top of the DATA step (the observation is deleted).
* *false*, the DELETE statement does not execute, and processing continues with the next statement in the DATA step.

Eg:

**IF** RestHR<70 **THEN DELETE**;

1. Selecting variables

* When you need to read and process variables that you don’t want to keep in your data set, use **DROP** and **KEEP** option or statement to specify the variables that you want to drop or keep.
* The DROP and KEEP statements differ from the DROP= and KEEP = data set options in the following ways:
* You cannot use the DROP and KEEP statements in SAS procedure steps.
* The DROP and KEEP statements apply to all output data sets that are named in the DATA statement. To exclude variables from some data sets but not from others, use the DROP= and KEEP data set options in the DATA statement.
* **DROP** and **KEEP** option
* Basic code:

**(DROP=***variable(s)***)**

**(KEEP=***variable(s)***)**

the DROP=KEEP= or option, in parentheses, follows the name of the data set that contains the variables to be dropped or kept

*variable(s)* identifies the variables to drop or keep.

* **DROP** and **KEEP** statement
* Basic code:

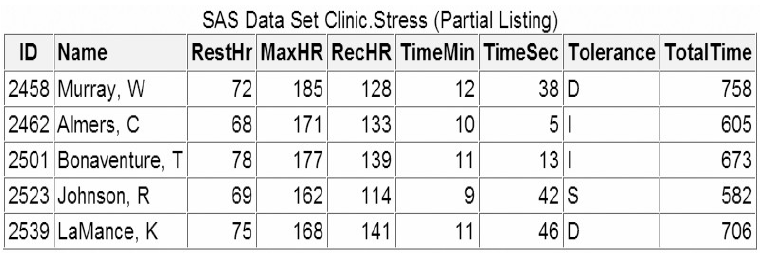
**DROP** variable(s);

**KEEP** variable(s);

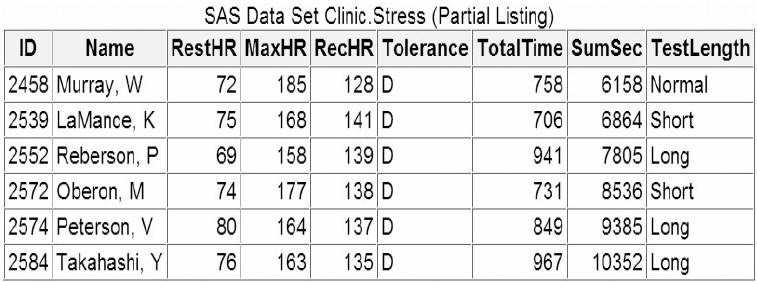
Eg for both statement and option

|  |  |
| --- | --- |
| **KEEP** and **DROP** option | **KEEP** and **DROP** statement |
| **DATA** clinic.stress(**DROP**=timemin timesec); | **DATA** clinic.stress; |
| **INFILE** tests; | **INFILE** tests; |
| **INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 | **INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 |
| MaxHR 31-33 RecHR 35-37 | MaxHR 31-33 RecHR 35-37 |
| TimeMin 39-40 TimeSec 42-43 | TimeMin 39-40 TimeSec 42-43 |
| Tolerance $ 45; | Tolerance $ 45; |
| **IF** tolerance='D'; | **IF** tolerance='D'; |
| TotalTime=(timemin\*60)+timesec; | **DROP** timemin timesec; |
| **RETAIN** SumSec 5400; | TotalTime=(timemin\*60)+timesec; |
| sumsec+totaltime; | **RETAIN** SumSec 5400; |
| **LENGTH** TestLength $ 6; | sumsec+totaltime; |
| **IF** totaltime>800 | **LENGTH** TestLength $ 6; |
| **THEN** testlength='Long'; | **IF** totaltime>800 |
| **ELSE** **IF** 750<=totaltime<=800 | **THEN** testlength='Long'; |
| **THEN** testlength='Normal'; | **ELSE** **IF** 750<=totaltime<=800 |
| **ELSE** **IF** totaltime<750 | **THEN** testlength='Normal'; |
| **THEN** TestLength='Short'; | **ELSE** **IF** totaltime<750 |
| **RUN**; | **THEN** TestLength='Short'; |
|  | **RUN**; |

**Original Data Set**



**Data Set after Modify**



1. Permanent Label and Formats

* Chapter 4 talks about temporarily assign labels and formats within a **PROC** step. To permanently assign labels and formats you need to use **LABEL** and **FORMAT** statement in **DATA** steps
* Most SAS procedures automatically use permanent labels and formats in output, without requiring additional statements or options.
* If you assign temporary labels or formats within a PROC step, they override any permanent labels or formats that were assigned during the DATA step.

Eg:

|  |
| --- |
| **DATA** clinic.stress; |
| **INFILE** tests; |
| **INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 |
| MaxHR 31-33 RecHR 35-37 |
| TimeMin 39-40 TimeSec 42-43 |
| Tolerance $ 45; |
| **IF** RestHR < 70 **THEN DELETE;** |
| **IF** tolerance='D'; |
| **DROP** timemin timesec; |
| TotalTime=(timemin\*60)+timesec; |
| **RETAIN** SumSec 5400; |
| sumsec+totaltime; |
| **LENGTH** TestLength $ 6; |
| **IF** totaltime>800 |
| **THEN** testlength='Long'; |
| **ELSE** **IF** 750<=totaltime<=800 |
| **THEN** testlength='Normal'; |
| **ELSE** **IF** totaltime<750 |
| **THEN** TestLength='Short'; |
| **LABEL** SumSec = |
| ‘Cumulative Total Seconds (+5,400)’ |
| **FORMAT** sumsec comma6.; |
| **RUN**; |

1. **SELECT** group (Assign condition)

* Besides **IF – THEN/ELSE** statement, **SELECT** statement also can be used to select data
* If the result of all **SELECT-WHEN** comparisons is false and no **OTHERWISE** statement is present, SAS issues an error message and stops executing the **DATA** step.
* Basic code:

|  |  |  |
| --- | --- | --- |
| Statement | Sub-code | Description |
| **SELECT** | **SELECT** <*(select-expression)*>**;** | Begins a **SELECT** group. |
| **WHEN** | **WHEN**-1 *(when-expression-1*<*..., when-expression-n*>) *statement***;**  **WHEN***-n (when-expression-1* <*..., when-expression-n*>) *statement***;** | Identifies SAS statements that are executed when a particular condition is true. |
| **OTHERWISE (**optional**)** |  | Specifies a statement to be executed if no WHEN condition is met |
| **END** |  | Ends a **SELECT** group |

* SELECT begins a SELECT group.
* the optional *select-expression* specifies any SAS expression that evaluates to a single value.
  + - WHEN identifies SAS statements that are executed when a particular condition is true.
* *when-expression* specifies any SAS expression, including a compound expression. You must specify at least one *when-expression*.
* *statement* is any executable SAS statement. You must specify the *statement* argument.
* **SELECT** Group with/without Expressions
  + - With Expressions

SAS evaluates the select-expression and when-expression, compares the two for equality, and returns a value of true or false.

* 如果*true,* SAS 执行WHEN statement
* 如果*false,* SAS会选择这个WHEN statement中的下一个when-expression，或者直接进入下一个WHEN statement.

Eg:

SAS determines the value of toy and compares it to values in each WHEN statement in turn. If a WHEN statement is true compared to the toy value, SAS assigns the related price and continues processing the rest of the DATA step. If none of the comparisons is true, SAS executes the OTHERWISE statement and writes a debugging message to the SAS log.

**SELECT** (toy);

**WHEN** ("Bear") price=35.00;

**WHEN** ("Violin") price=139.00;

**WHEN** ("Top","Whistle","Duck") price=7.99;

**OTHERWISE** **PUT** "Check unknown toy: " toy=;

**END**;

* + - Without Expressions

If you *don't* specify a select-expression, SAS evaluates each when-expression to produce a result of true or false.

* 如果*true*, 执行WHEN statement
* 如果*false,* SAS会选择这个WHEN statement中的下一个when-expression，或者直接进入下一个WHEN statement.

注意：If more than one WHEN statement has a true when-expression, only the *first* WHEN statement is used. Once a when-expression is true, no other when-expressions are evaluated.

Eg:

In the example below, the SELECT statement does not specify a select-expression. The WHEN statements are evaluated in order, and only one is used.

**SELECT**;

**WHEN** (toy="Bear" and month IN ('OCT', 'NOV', 'DEC')) price=45.00;

**WHEN** (toy="Bear" and month IN('JAN', 'FEB')) price=25.00;

**WHEN** (toy="Bear") price=35.00;

**OTHERWISE;**

**END;**

1. **DO** group

IF-THEN/ELSE statement and SELECT group execute only a single SAS statement when conditions is true. DO groups can execute a group of statement as a unit

* Basic code:

**DO;**

*SAS statements*

**END;**

|  |  |
| --- | --- |
|  | In the SELECT group below, the statements between DO and END are performed only when the value of Payclass is **hourly**. Notice that an IF-THEN statement appears in the DO group; the PUT statement executes only when Hours is greater than **40**. The second END statement in the program closes the SELECT group. |
| **DATA** clinic.stress; | **DATA** payroll; |
| **INFILE** tests; | **SET** salaries; |
| **INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 | **SELECT**(payclass); |
| MaxHR 31-33 RecHR 35-37 | **WHEN** ('monthly') amt=salary; |
| TimeMin 39-40 TimeSec 42-43 | **WHEN** ('hourly') |
| Tolerance $ 45; | **DO**; |
| TotalTime=(timemin\*60)+timesec; | amt=hrlywage\*min(hrs,40); |
| **RETAIN** SumSec 5400; | **IF** hrs>40 |
| sumsec+totaltime; | **THEN** PUT 'CHECK TIMECARD'; |
| **LENGTH** TestLength $ 6 Message $ 20; | **END**; |
| **IF** totaltime>800 **THEN** | **OTHERWISE** **PUT** 'PROBLEM OBSERVATION'; |
| **DO**; | **END**; |
| testlength='Long'; | **RUN**; |
| message='Run blood panel'; |  |
| **END**; |  |
| **ELSE** **IF** 750<=totaltime<=800 |  |
| **THEN** testlength='Normal'; |  |
| **ELSE** **IF** totaltime<750 |  |
| **THEN** TestLength='Short'; |  |
| **RUN**; |  |

1. Syntax

**LIBNAME** *libref 'SAS-data-library'***;**

**FILENAME** *fileref 'filename'***;**

**DATA** *SAS-data-set*(**DROP=***variable(s)*|**KEEP=***variable(s)*)**;**

**INFILE** *file-specification* <OBS=*n*>**;**

**INPUT** *variable* <$> *startcol-endcol...***;**

**DROP** *variable(s)***;**

**KEEP** *variable(s)***;**

**RETAIN** *variable initial-value***;**

*variable+expression***;**

**LENGTH** *variable(s)* <$> *length***;**

**IF** *expression***THEN** *statement*;

**ELSE** *statement***;**

**IF** *expression***THEN DELETE;**

**LABEL** *variable1***=***'label1' variable2***=***'label2' ...***;**

**FORMAT** *variable(s) format-name*;

**SELECT** <*(select-expression)*>;

**WHEN**-1 (*when-expression-1* <*..., when-expression-n*>) *statement***;**

**WHEN**-n (*when-expression-1* <*..., when-expression-n*>) *statement***;**

<OTHERWISE*statement*;>

**END;**

**RUN;**

**PROC PRINT DATA=***SAS-data set* **LABEL;**

**RUN;**

1. Sample Program

**DATA** clinic.stress;

**INFILE** tests;

**INPUT** ID $ 1-4 Name $ 6-25 RestHR 27-29 MaxHR 31-33

RecHR 35-37 TimeMin 39-40 TimeSec 42-43

Tolerance $ 45;

**IF** tolerance='D'AND resthr ge 70 **THEN** **DELETE**;

**DROP** timemin timesec;

TotalTime=(timemin\*60)+timesec;

**RETAIN** SumSec 5400;

sumsec+totaltime;

**LENGTH** TestLength $ 6;

**IF** totaltime>800 **THEN** testlength='Long';

**ELSE** **IF** 750<=totaltime<=800 **THEN** testlength='Normal';

**ELSE** **IF** totaltime<750 **THEN** TestLength='Short';

**LABEL** sumsec='Cumulative Total Seconds (+5,400)';

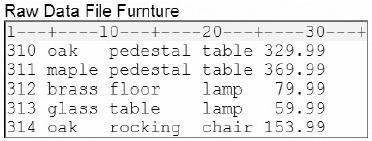
**FORMAT** sumsec comma6.;

**RUN**;

题目

1. Which program creates the output shown below?

***Output 1 Output 2***

1. **DATA** test2;

**INFILE** furnture;

**INPUT** StockNum $ 1-3 Finish $ 5-9 Style $ 11-18

Item $ 20-24 Price 26-31;

**IF** finish='oak' **THEN DELETE**;

**RETAIN** TotPrice 100;

totalprice+price;

**DROP** price;

**RUN**;

**PROC PRINT** data=test2 **NOOBS**;

**RUN**;

1. **DATA** test2;

**INFILE** furnture;

**INPUT** StockNum $ 1-3 Finish $ 5-9 Style $ 11-18

Item $ 20-24 Price 26-31;

**IF** finish='oak' and price<200 **THEN DELETE**;

totalprice+price;

**DROP** price;

**RUN**;

**PROC PRINT** data=test2 **NOOBS**;

**RUN**;

1. **DATA** test2 (**DROP** =price);

**INFILE** furnture;

**INPUT** StockNum $ 1-3 Finish $ 5-9 Style $ 11-18

Item $ 20-24 Price 26-31;

**IF** finish='oak' and price<200 **THEN DELETE**;

totalprice+price;

**RUN**;

**PROC PRINT** data=test2 **NOOBS**;

**RUN**;

1. **DATA** test2;

**INFILE** furnture;

**INPUT** StockNum $ 1-3 Finish $ 5-9 Style $ 11-18

Item $ 20-24 Price 26-31;

**IF** finish='oak' and price<200 **THEN DELETE**;

totalprice+price;

**RUN**;

**PROC PRINT** data=test2 **NOOBS**;

**RUN**;

1. How is the variable Amount labeled and formatted in the PROC PRINT output?

**DATA** credit;

**INFILE** creddata;

**INPUT** Account $ 1-5 Name $ 7-25 Type $ 27

Transact $ 29-35 Amount 37-50;

**LABEL** amount='Amount of Loan';

**FORMAT** amount dollar12.2;

**RUN**;

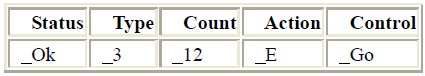
**PROC** **PRINT** data=credit label;

**LABEL** amount='Total Amount Loaned';

**FORMAT** amount comma10.;

**RUN**;

1. label Amount of Loan, format DOLLAR12.2
2. label Total Amount Loaned, format COMMA10.
3. label Amount, default format
4. The PROC PRINT step does not execute because two labels and two formats are assigned to the same variable.
5. For the observation shown below, what is the result of the IF-THEN statements?



**IF** status='OK' **AND** type=3

THEN Count+1;

**IF** status='S' **OR** action='E'

**THEN** Control='Stop';

1. Count = 12 Control = Go
2. Count = 13 Control =Stop
3. Count = 12 Control =Stop
4. Count = 13 Control = Go
5. Which of the following can determine the length of a new variable?
6. the length of the variable's first reference in the DATA step
7. the assignment statement
8. the LENGTH statement
9. all of the above
10. Which set of statements is equivalent to the code shown below?

IF code='1' THEN Type='Fixed';

IF code='2' THEN Type='Variable';

IF code^='1' AND code^='2' THEN Type='Unknown';

1. IF code='1' THEN Type='Fixed';

ELSE IF code='2' THEN Type='Variable';

ELSE Type='Unknown';

1. IF code='1' THEN Type='Fixed';

IF code='2' THEN Type='Variable';

ELSE Type='Unknown';

1. IF code='1' THEN type='Fixed';

ELSE code='2' AND type='Variable';

ELSE type='Unknown';

1. IF code='1' AND type='Fixed';

THEN code='2' AND type='Variable';

ELSE type='Unknown';

1. What is the length of the variable Type, as created in the DATA step below?

**DATA** finance.newloan;

**SET** finance.records;

TotLoan+payment;

**IF** code='1' **THEN** Type='Fixed';

**ELSE** Type='Variable';

**LENGTH** type $ 10;

**RUN**;

1. 5
2. 8
3. 10
4. It depends on the first value of Type.
5. Which program contains an error?

DATA clinic.stress(drop=timemin timesec);

INFILE tests;

INPUT ID $ 1-4 Name $ 6-25 RestHR 27-29 MaxHR 31-33

RecHR 35-37 TimeMin 39-40 TimeSec 42-43

Tolerance $ 45;

TotalTime=(timemin\*60)+timesec;

SumSec+totaltime;

RUN;

PROC PRINT data=clinic.stress;

LABEL totaltime='Total Duration of Test';

FORMAT timemin 5.2;

DROP sumsec;

RUN;

PROC PRINT data=clinic.stress(keep=totaltime timemin);

LABEL totaltime='Total Duration of Test';

FORMAT timemin 5.2;

RUN;

DATA clinic.stress;

INFILE tests;

INPUT ID $ 1-4 Name $ 6-25 RestHR 27-29 MaxHR 31-33

RecHR 35-37 TimeMin 39-40 TimeSec 42-43

Tolerance $ 45;

TotalTime=(timemin\*60)+timesec;

KEEP id totaltime tolerance;

RU;