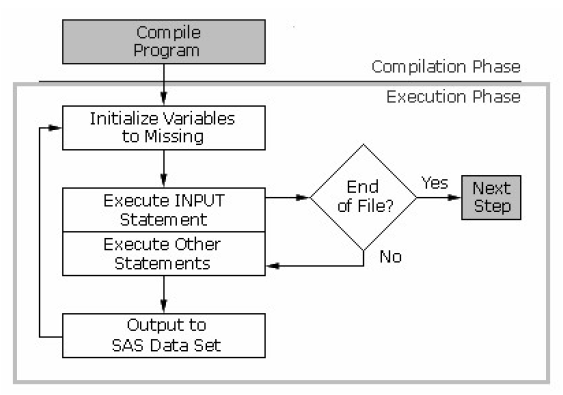
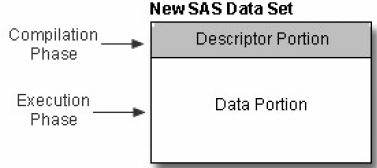
**Chapter VI – Understanding DATA Step Processing**

1. BASIC STEPS:



* **Compilation Phase**

Scanning for syntax errors

When the compilation phase is complete, the descriptor portion of the new data set is created.

* Input Buffer

An area of memory created to hold a record from the external file. Create only when raw data is read, not when SAS data set is read.

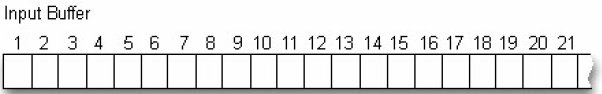
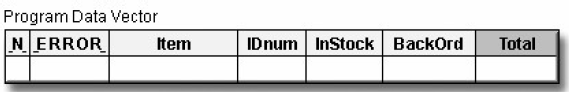
* Program Data Vector

After the input buffer is created, the program data vector is created. The program data vector is the area of memory where SAS holds one observation at a time.

Data vector contains two automatic variables:

* \_**N**\_ counts the number of times that the DATA step begins to execute.
* \_**ERROR**\_ signals the occurrence of an error that is caused by the data during execution. The default value is 0, which means there is no error. When one or more errors occur, the value is set to 1.
* Data Set Variables
* As the **INPUT** statement is compiled, a slot is added to the program data vector for each variable in the new data set.
* Any variables that are created with an assignment statement in the DATA step are also added to the program data vector.
* Descriptor Portion of the SAS Data Set

When compilation phase is complete, and the descriptor portion of the new SAS data set is created, which include:

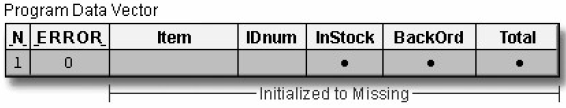
The name of the data set, the number of variables, and the names and attributes of the variables.

* **Execution Phase**

If the DATA step compiles successfully, then the execution phase begins. During the execution phase, the DATA step reads and processes the input data. The DATA step executes once for each record in the input file

The remaining variables are initialized to missing. Missing numeric values are represented by periods, and missing character values are represented by blanks.

* Initializing Variables
* At the beginning of the execution phase, the value of \_**N**\_ is **1**. Because there are no data errors, the value of \_**ERROR**\_ is **0**.



* **INFILE** statement

**INFILE** statement identifies the location of the raw data

* **INPUT** statement
* **INPUT** statement reads a record into the input buffer one element a time from the raw data file. It reads from left to right.
* When the first row is filled up, control returns to the top of the DATA step and the value of \_N\_ increments from 1 to **2**. Finally, the variable values in the program data vector are re-set to missing. Notice that the automatic variable \_ERROR\_ is reset to zero if necessary. The same process repeats.
* The execution phase continues in this manner until end-of-file (there are no more records in the raw data file to be read), then the data portion of the new data set is complete and the DATA step stops.

Eg:

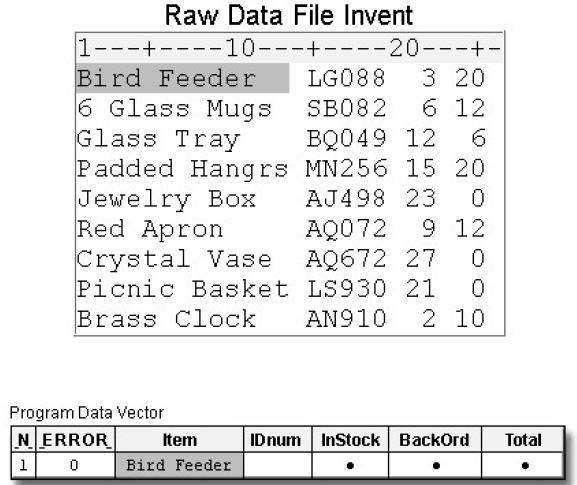
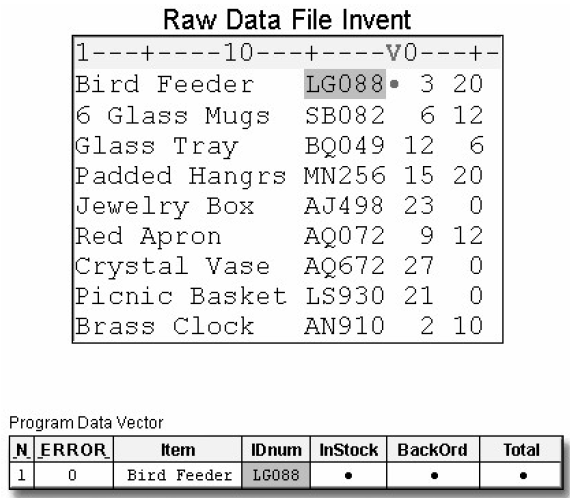
**DATA** perm.update;

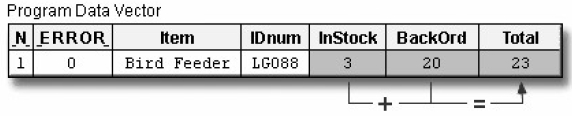
**INFILE** invent;

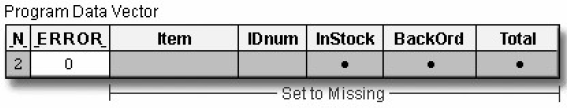
**INPUT** Item $ 1-13 IDnum $ 15-19

InStock 21-22 BackOrd 24-25;

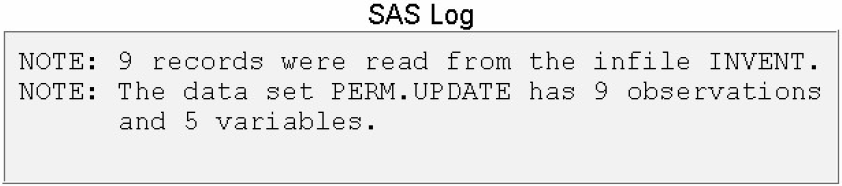
Total=instock+backord;

**RUN**;





* At the end of the execution phase, the SAS log confirms that the raw data file was read, and it displays the number of observations and variables in the data set.



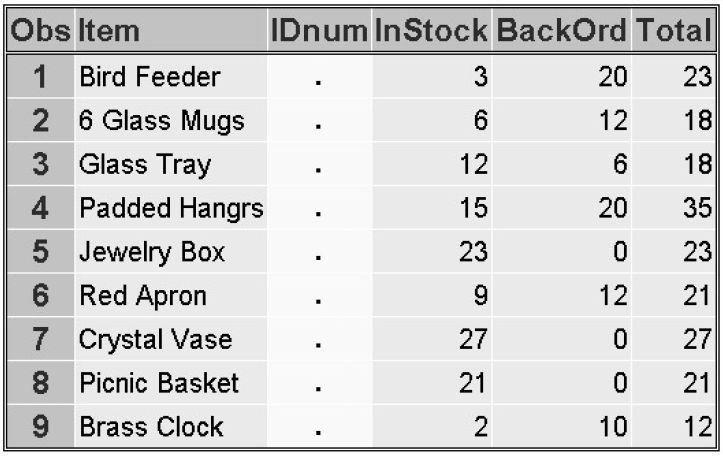
1. DEBUGGING A DATA STEP

* Diagnosing Error in the Compilation Phase

Syntax Errors on:

* misspelled keywords and data set names
* unbalanced quotation marks
* invalid options.
* Diagnosing Error in the Execution Phase

When SAS detects an error in the execution phase depending on the type of error:

* A note, warning, or error message is displayed in the log.
* The values that are stored in the program data vector are displayed in the log.
* The processing of the step either continues or stops.
* SAS program can detect invalid data:
* **PROC PRINT**

Eg:

**PROC PRINT** data=perm.update;

**RUN**;

The result shows the IDnum missed its value

* **PROC FREQ**

Using **PROC FREQ** to identify any variables that were not given an expected value

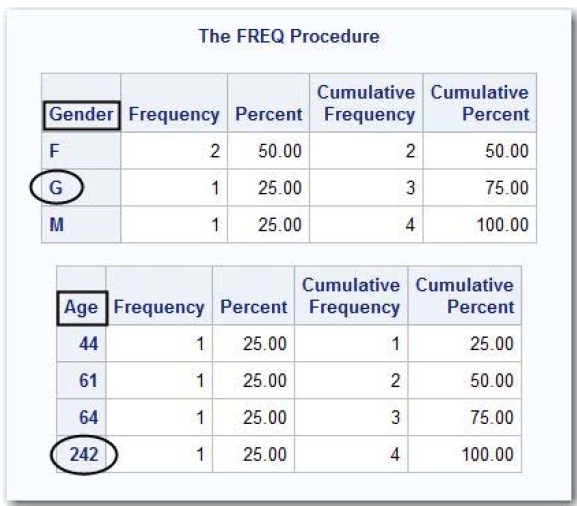
Basic code:

**PROC FREQ DATA=***SAS-data-set* <*NLEVELS*>;

**TABLES** *variable(s);*

**RUN**;

The **TABLES** statement specifies the frequency tables to produce.

The **NLEVELS** option displays a table that provides the number of distinct values for each variable named in the TABLES statement.

Eg:

**PROC FREQ** data=work.Patients;

**TABLES** Gender Age;

**RUN**;

In the output you can see both the valid (**M** and **F**) and invalid (**G**) values for Gender, and the valid and invalid (242) values for age

* **PROC MEANS**

The **MEANS** procedure can also be used to validate data because it produces summary reports displaying descriptive statistics

Basic code:

**PROC MEANS DATA=***SAS-data-set* <*statistics*>;

**VAR** *variable(s)*;

**RUN**;

The VAR statement specifies the analysis variables and their order in the results.

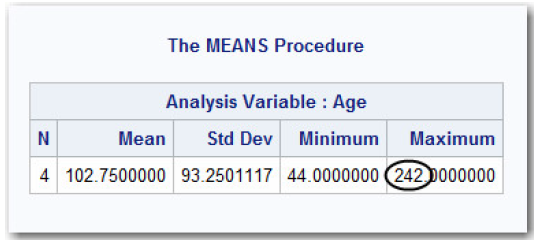
The statistics to display can be specified in the PROC MEANS statement.

Eg:

**PROC MEANS** data=work.Patients;

**VAR** Age;

**RUN**;

The output for the MEANS procedure displays a range of 44 to 242, which clearly indicates that there is invalid data somewhere in the Age column.

1. Sample Program:

**DATA** perm.update;

**INFILE** invent;

**INPUT** Item $ 1-13 IDnum $ 15-19

InStock 21-22 BackOrd 24-25;

Total=instock+backord;

**RUN**;

**DATA** work.test;

**INFILE** loan;

**INPUT** Code $ 1 Amount 3-10;

if code='1' then type='variable';

else if code='2' then type='fixed';

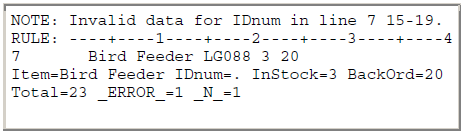
else put 'MY NOTE: invalid value: '

code=;

**RUN**;

练习

1. Unless otherwise directed, the DATA step executes
2. once for each compilation phase.
3. once for each DATA step statement.
4. once for each record in the input file.
5. once for each variable in the input file.
6. If SAS cannot interpret syntax errors, then
7. data set variables will contain missing values.
8. the DATA step does not compile.
9. the DATA step still compiles, but it does not execute.
10. the DATA step still compiles and executes.
11. Look carefully at this section of a SAS session log. Based on the note, what was the most likely problem with the DATA step?



1. A keyword was misspelled in the DATA step.
2. A semicolon was missing from the INFILE statement.
3. A variable was misspelled in the INPUT statement.
4. A dollar sign was missing in the INPUT statement.