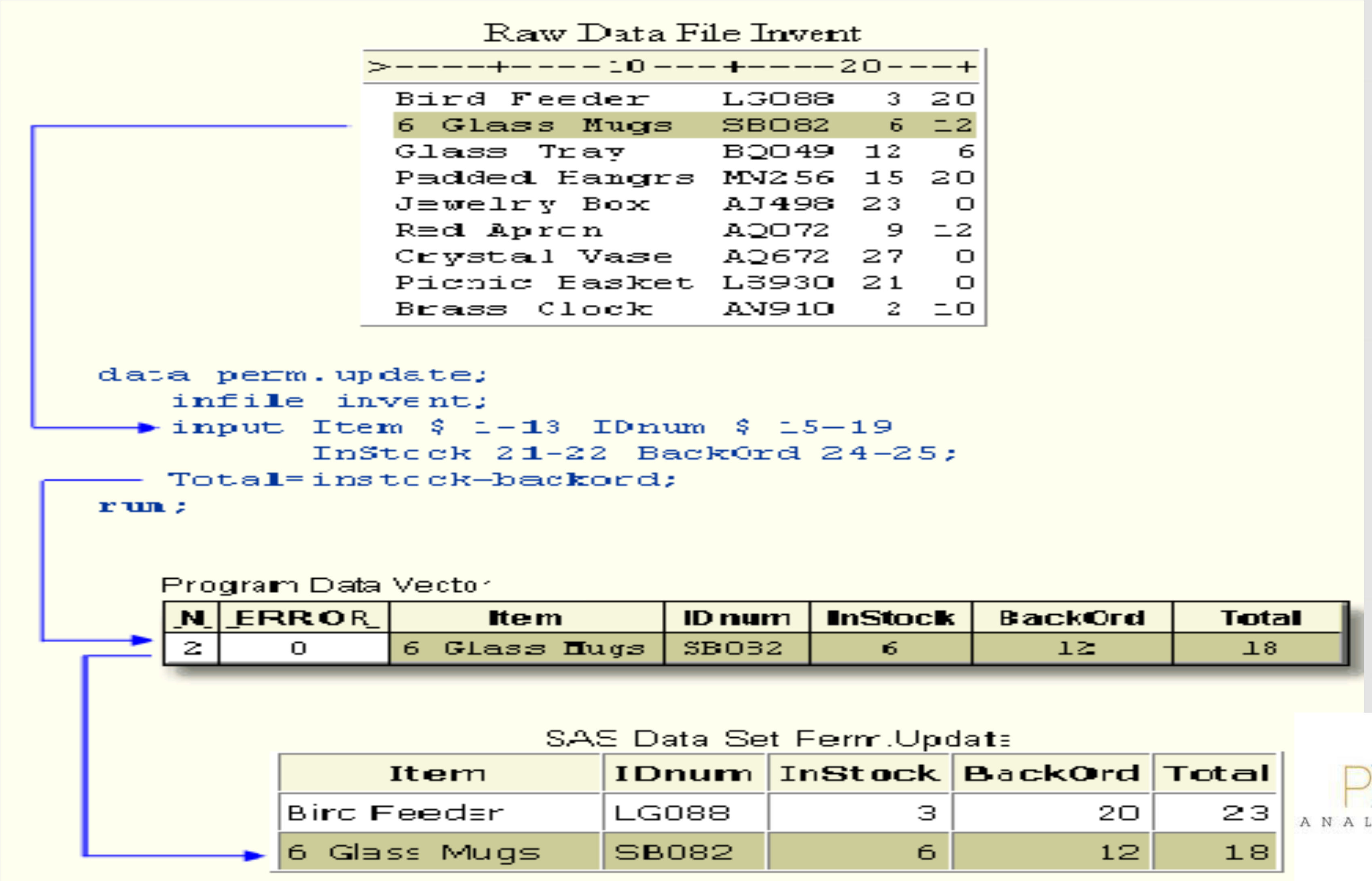
What happens "behind the scenes" when the DATA step reads raw data. Let's examine the **program data vector**, which is a logical framework that SAS uses when creating SAS datasets



Raw Data File Invent

```
>----+----10---+----20---+
 Bird Feeder   LG088  3 20
 6 Glass Mugs  SB082  6 12
 Glass Tray    BQ049 12  6
 Padded Hangrs MN256 15 20
 Jewelry Box   AJ498 23  0
 Red Apron     AQ072  9 12
 Crystal Vase  AQ672 27  0
 Picnic Basket LS930 21  0
 Brass Clock   AN910  2 10
```

```
data perm.update;
   infile invent;
   input Item $ 1-13 IDnum $ 15-19
         InStock 21-22 BackOrd 24-25;
   Total=instock-backord;
run;
```

Program Data Vector

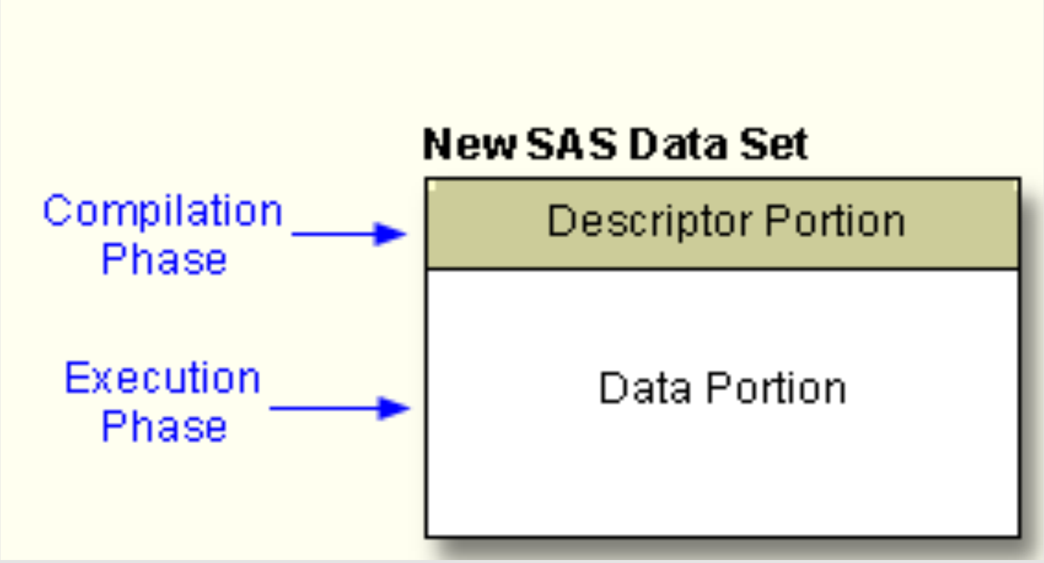
_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
2	0	6 Glass Mugs	SB082	6	12	18

SAS Data Set Perm.Update

Item	IDnum	InStock	BackOrd	Total
Bird Feeder	LG088	3	20	23
6 Glass Mugs	SB082	6	12	18

How SAS Processes Programs

SAS processes the DATA step and then creates a new SAS dataset. A SAS DATA step is processed in two phases:

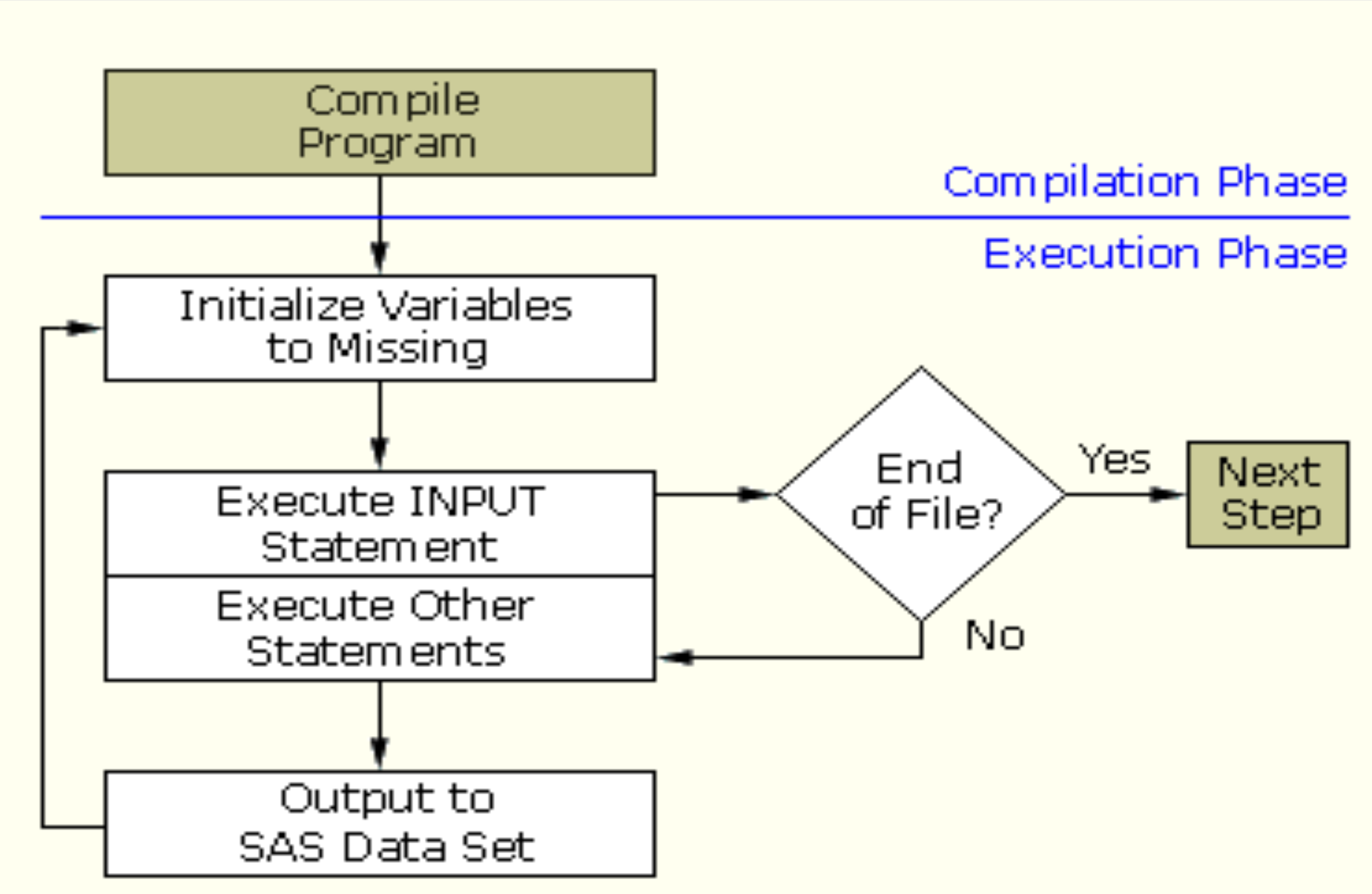


During the **compilation phase**, each statement is scanned for syntax errors.

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

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, unless otherwise directed.

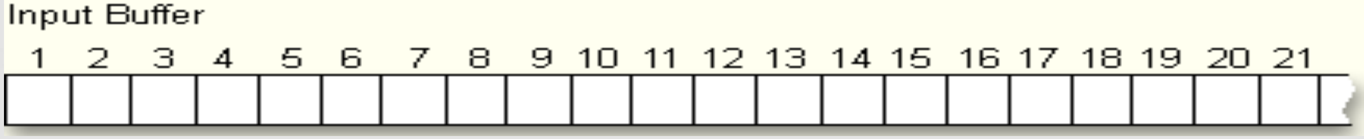
Flow of DATA step processing for reading raw data.



Compilation Phase

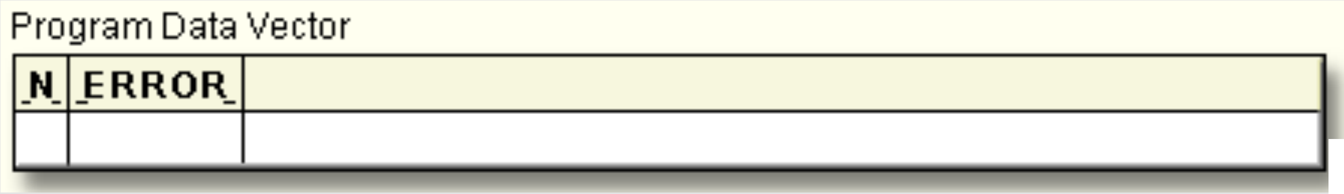
- **Input Buffer**

  At the beginning of the compilation phase, the **input buffer** (an area of memory) is created to hold a record from the external file. The input buffer is created only when raw data is read, not when a SAS data set is read. The term **input buffer** refers to a logical concept; it is not a physical storage area.



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 builds a data set, one observation at a time. Like the term **input buffer**, the term **program data vector** refers to a logical concept.
- The program data vector contains two **automatic variables** that can be used for processing but which are not written to the data set as part of an observation.
- **_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*.



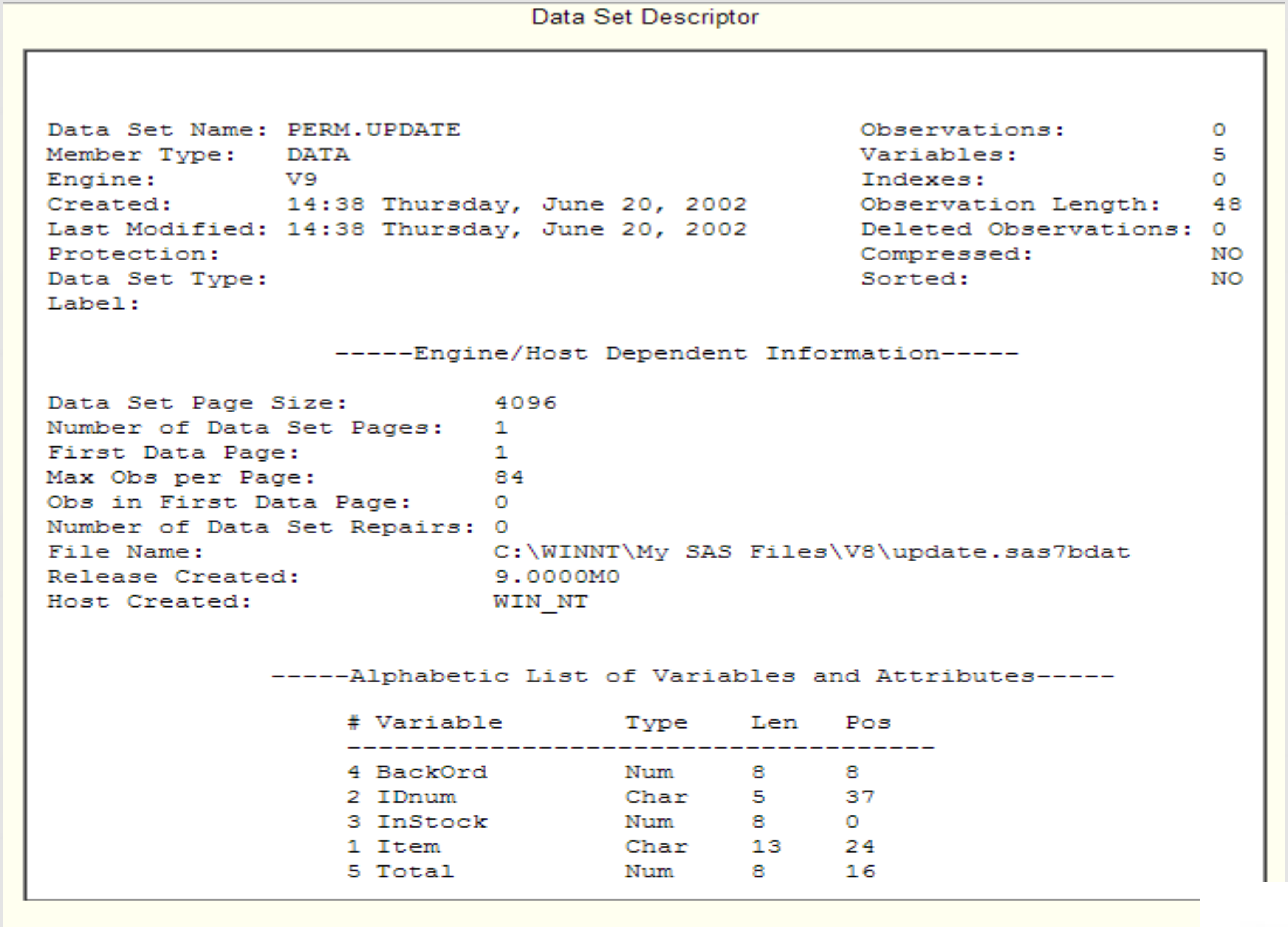
Program Data Vector

_N_	_ERROR_	

Descriptor Portion of the SAS Data Set

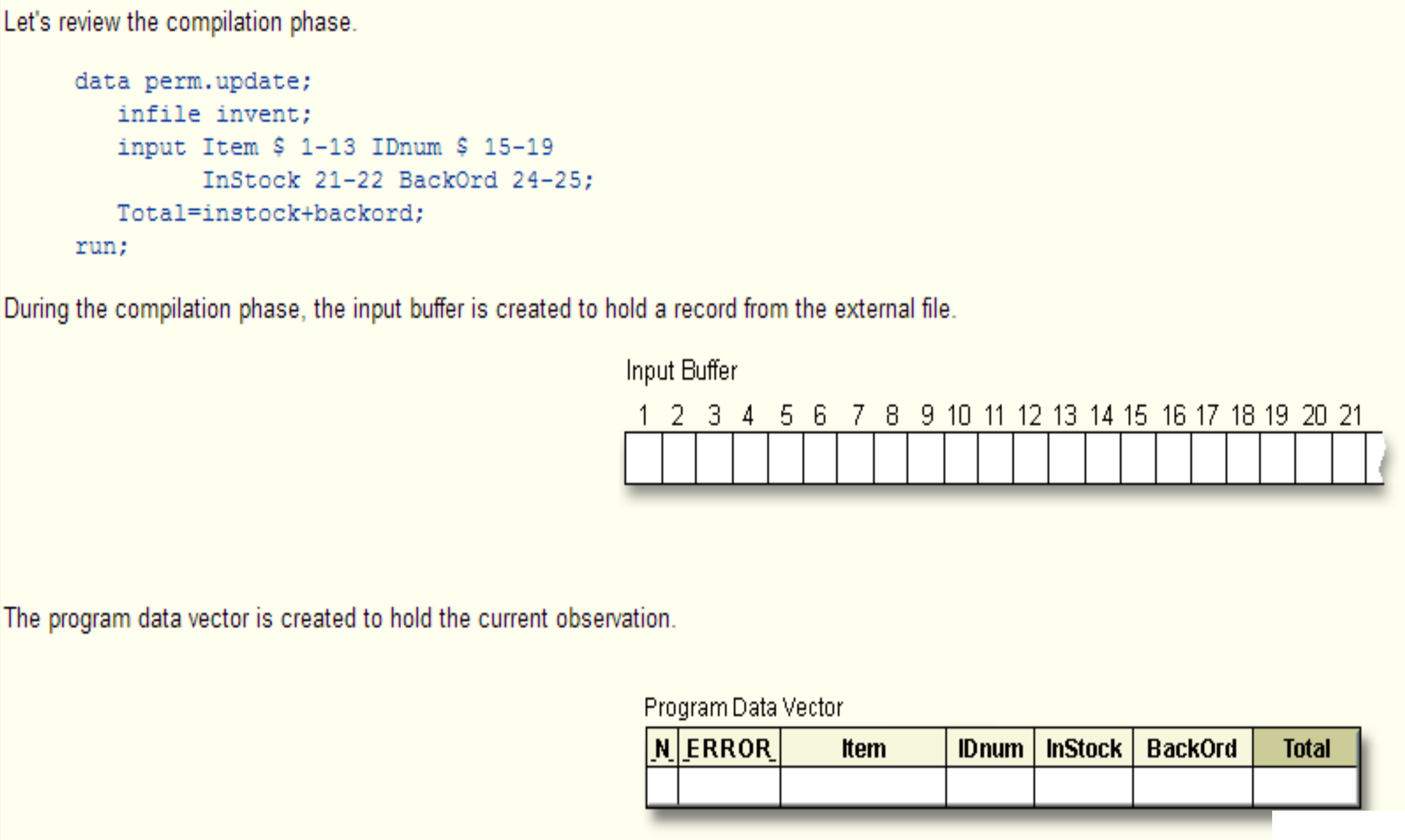
At the bottom of the DATA step (when the RUN statement is encountered), the compilation phase is complete, and the descriptor portion of the new SAS data set is created. The descriptor portion of the data set includes

- The name of the data set
- The number of observations and variables
- The names and attributes of the variables



```
Data Set Name: PERM.UPDATE                          Observations:         0
Member Type:   DATA                                 Variables:            5
Engine:        V9                                   Indexes:              0
Created:       14:38 Thursday, June 20, 2002        Observation Length:   48
Last Modified: 14:38 Thursday, June 20, 2002        Deleted Observations: 0
Protection:                                         Compressed:           NO
Data Set Type:                                      Sorted:               NO
Label:

                 -----Engine/Host Dependent Information-----

Data Set Page Size:          4096
Number of Data Set Pages:    1
First Data Page:             1
Max Obs per Page:            84
Obs in First Data Page:      0
Number of Data Set Repairs:  0
File Name:                   C:\WINNT\My SAS Files\V8\update.sas7bdat
Release Created:             9.0000M0
Host Created:                WIN_NT


              -----Alphabetic List of Variables and Attributes-----

                   # Variable        Type    Len    Pos
                   -------------------------------------
                   4 BackOrd         Num       8      8
                   2 IDnum           Char      5     37
                   3 InStock         Num       8      0
                   1 Item            Char     13     24
                   5 Total           Num       8     16
```

Summary of the Compilation Phase



Let's review the compilation phase.

```
data perm.update;
   infile invent;
   input Item $ 1-13 IDnum $ 15-19
         InStock 21-22 BackOrd 24-25;
   Total=instock+backord;
run;
```

During the compilation phase, the input buffer is created to hold a record from the external file.

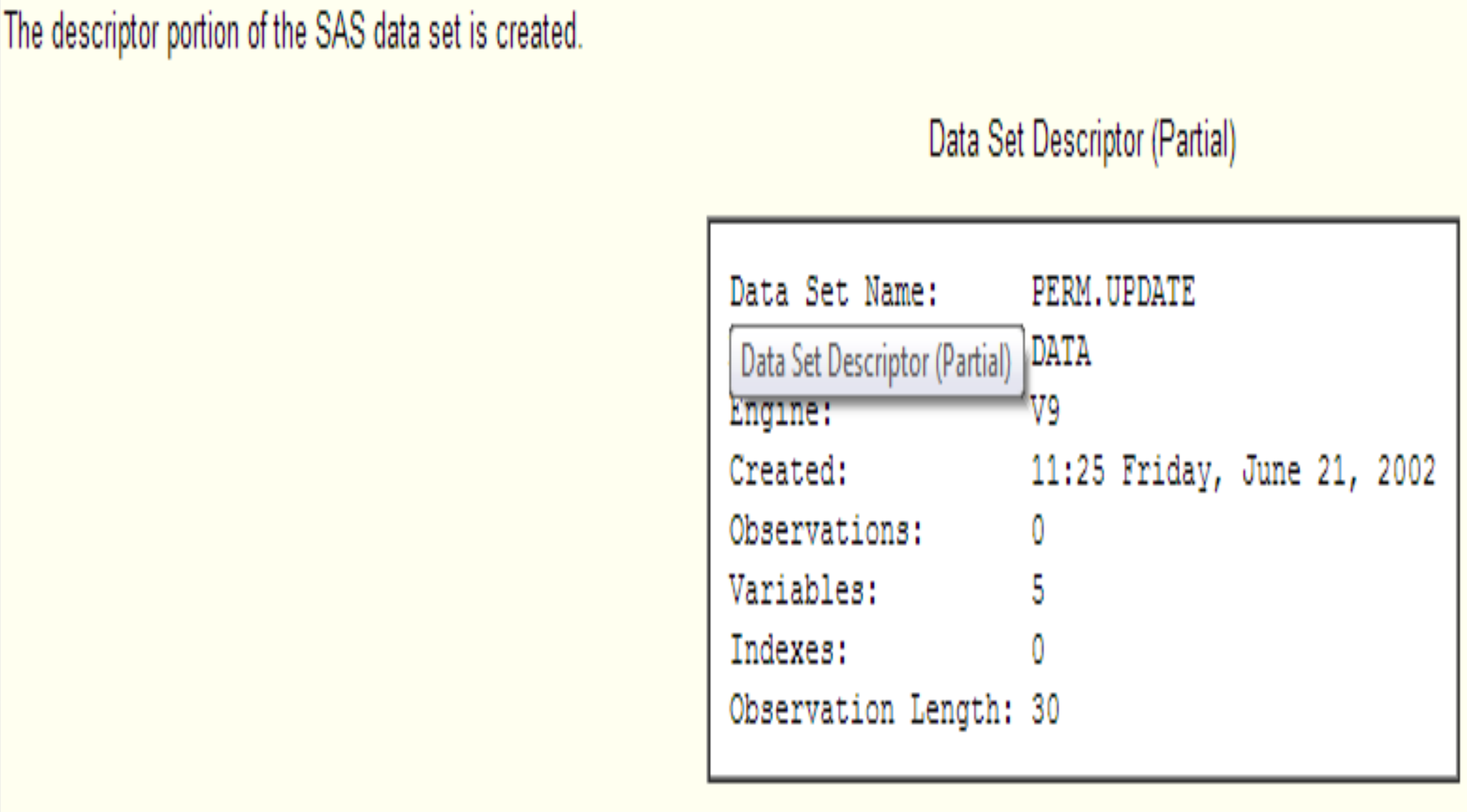
Input Buffer

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

The program data vector is created to hold the current observation.

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total

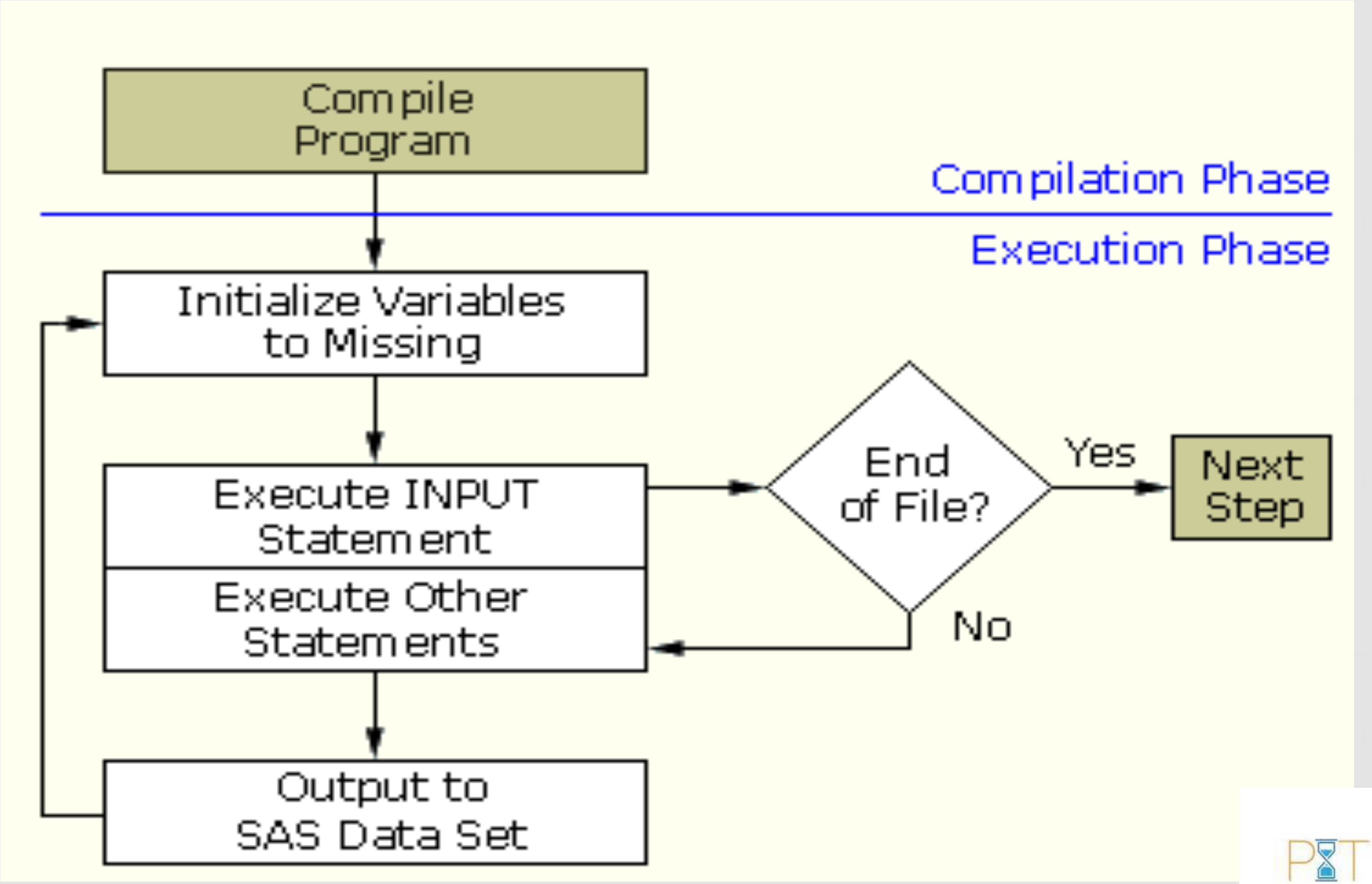


The descriptor portion of the SAS data set is created.

Data Set Descriptor (Partial)

```
Data Set Name:       PERM.UPDATE
Data Set Descriptor (Partial) DATA
Engine:              V9
Created:             11:25 Friday, June 21, 2002
Observations:        0
Variables:           5
Indexes:             0
Observation Length:  30
```

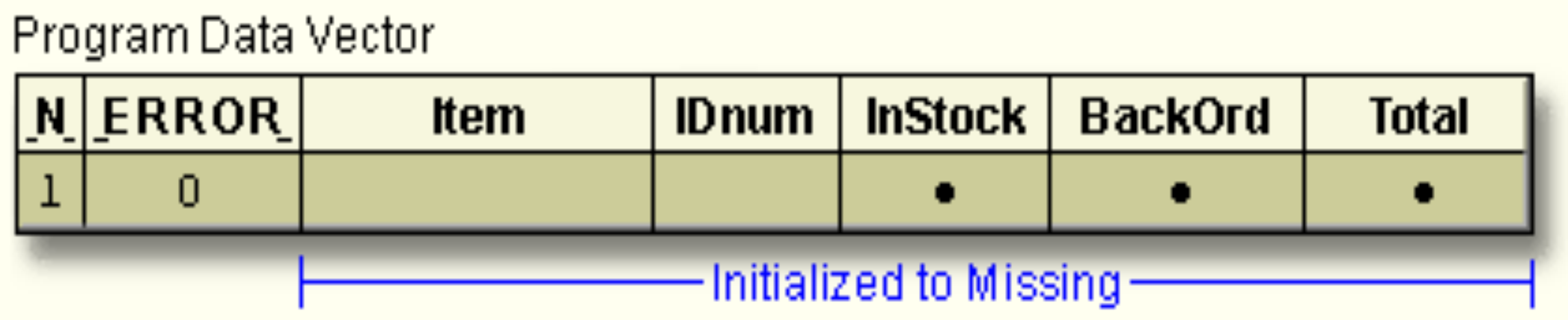
Execution Phase



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*.

```
data perm.update;
infile invent;
input Item $ 1-13 IDnum $ 15-19 InStock 21-22 BackOrd 24-25;
Total=instock+backord;
run;
```



Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0			.	.	.

Initialized to Missing

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

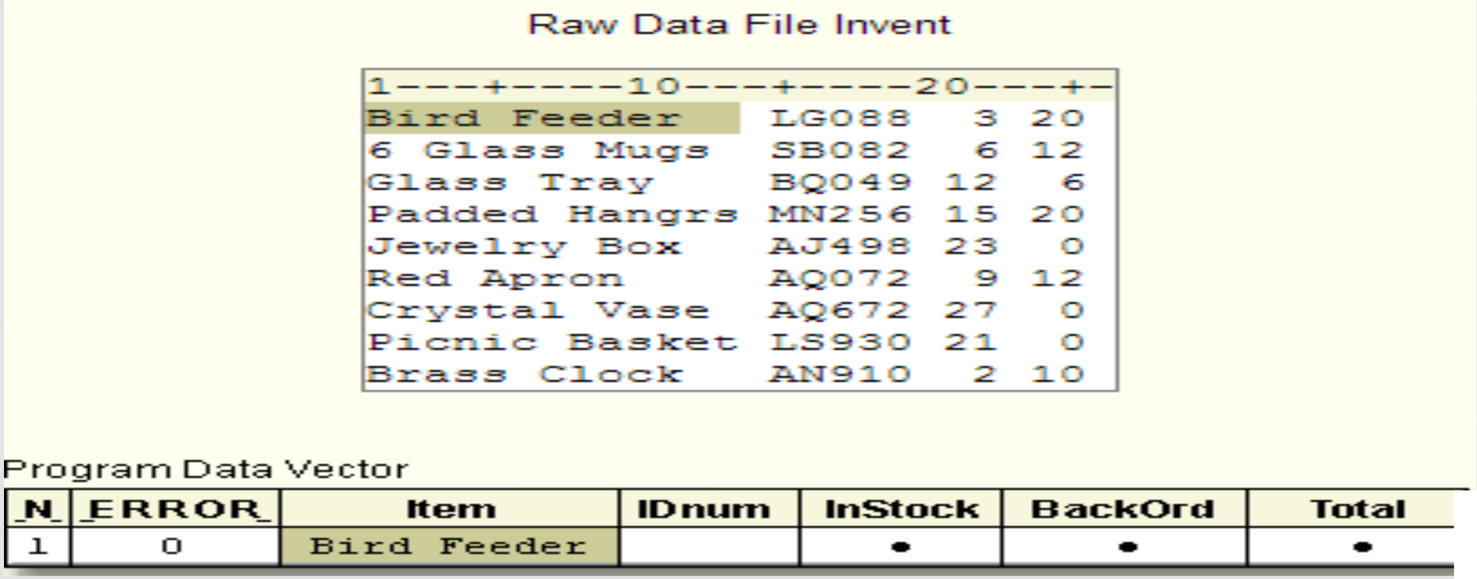
Input Data

- Next, the INFILE statement identifies the location of the raw data.

```
data perm.update;
infile invent;
input Item $ 1-13 IDnum $ 15-19 InStock 21-22
   BackOrd 24-25;
Total=instock+backord;
run;
```

Input Pointer

When an INPUT statement begins to read data values from a record that is held in the input buffer, it uses an input pointer to keep track of its position. The input pointer starts at column 1 of the first record, unless otherwise directed. As the INPUT statement executes, the raw data in columns 1-13 is read and is assigned to Item in the program data vector.



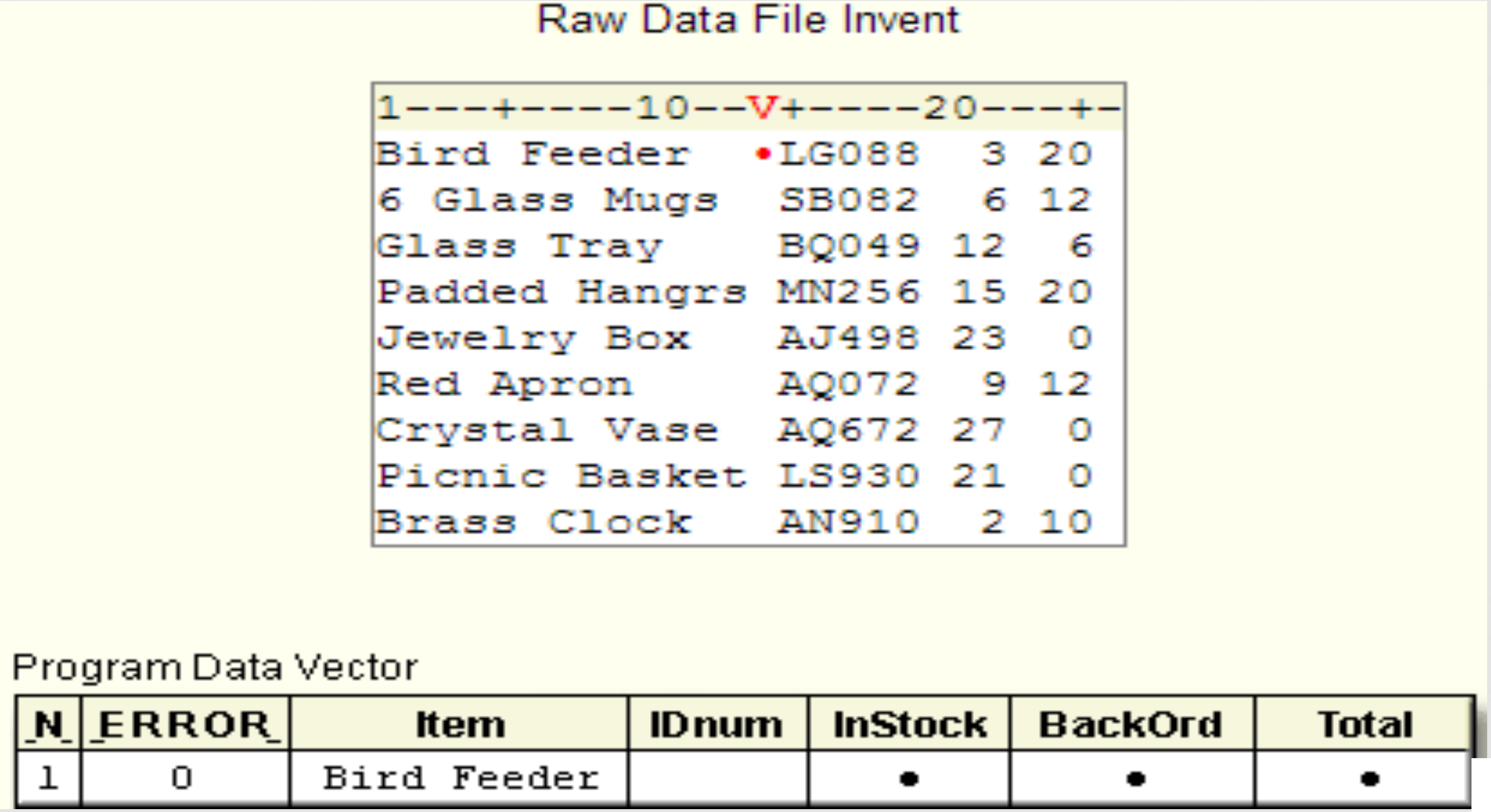
Raw Data File Invent

```
1---+----10---+----20---+-
Bird Feeder   LG088  3 20
6 Glass Mugs  SB082  6 12
Glass Tray    BQ049 12  6
Padded Hangrs MN256 15 20
Jewelry Box   AJ498 23  0
Red Apron     AQ072  9 12
Crystal Vase  AQ672 27  0
Picnic Basket LS930 21  0
Brass Clock   AN910  2 10
```

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0	Bird Feeder		•	•	•

The input pointer now rests on column 14. With column input, the pointer moves as far as the INPUT statement instructs it, and it stops in the column immediately following the last one read.



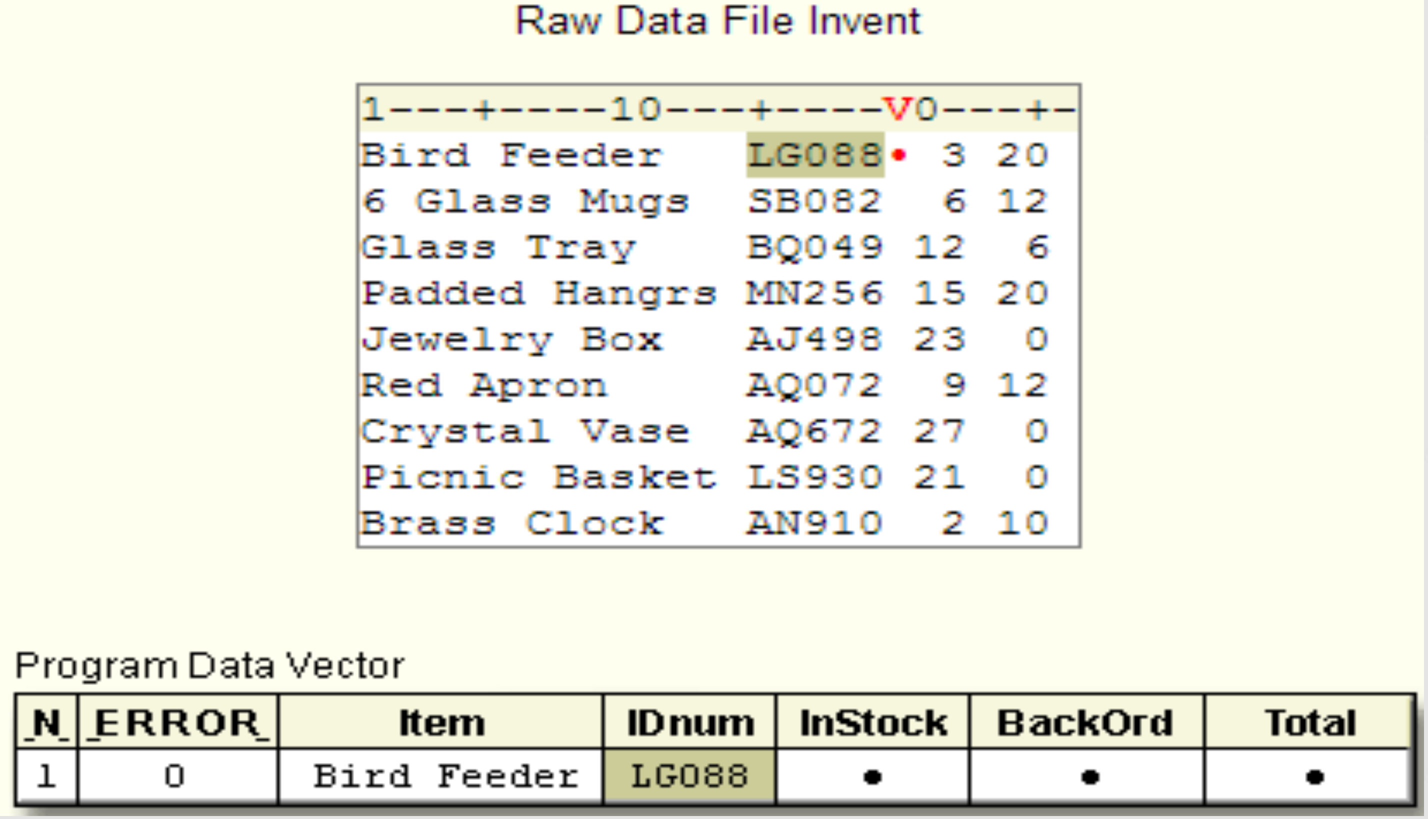
Raw Data File Invent

```
1---+----10--V+----20---+-
Bird Feeder  •LG088   3 20
6 Glass Mugs  SB082   6 12
Glass Tray    BQ049  12  6
Padded Hangrs MN256  15 20
Jewelry Box   AJ498  23  0
Red Apron     AQ072   9 12
Crystal Vase  AQ672  27  0
Picnic Basket LS930  21  0
Brass Clock   AN910   2 10
```

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0	Bird Feeder		•	•	•

- Next, the data in columns 15-19 is read and is assigned to IDnum in the program data vector.



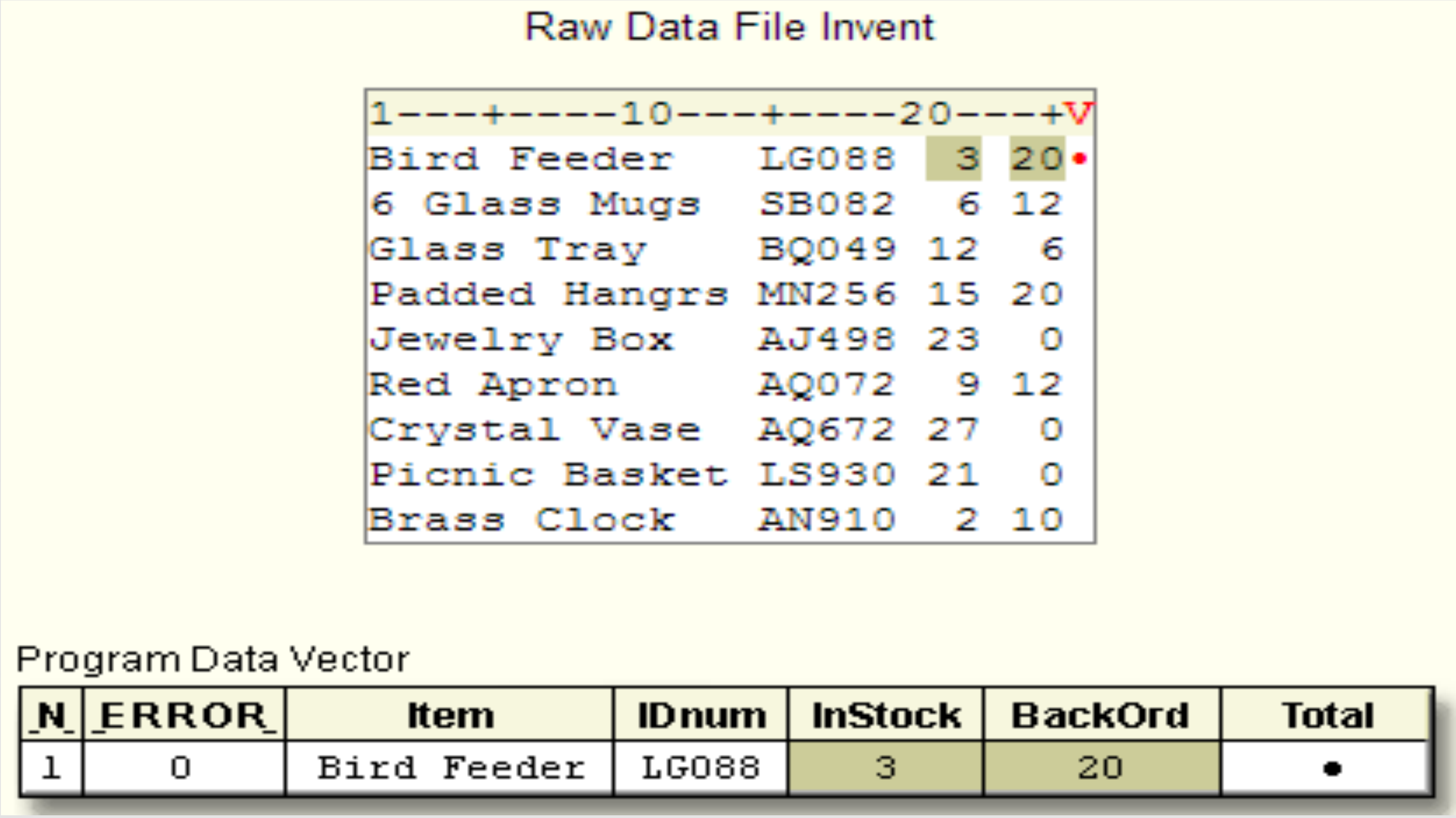
Raw Data File Invent

```
1---+----10---+----V0---+-
Bird Feeder    LG088  3 20
6 Glass Mugs   SB082  6 12
Glass Tray     BQ049 12  6
Padded Hangrs  MN256 15 20
Jewelry Box    AJ498 23  0
Red Apron      AQ072  9 12
Crystal Vase   AQ672 27  0
Picnic Basket  LS930 21  0
Brass Clock    AN910  2 10
```

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0	Bird Feeder	LG088	•	•	•

- Likewise, the INPUT statement reads the values for InStock from columns 21-22, and it reads the values for BackOrd from columns 24-25.



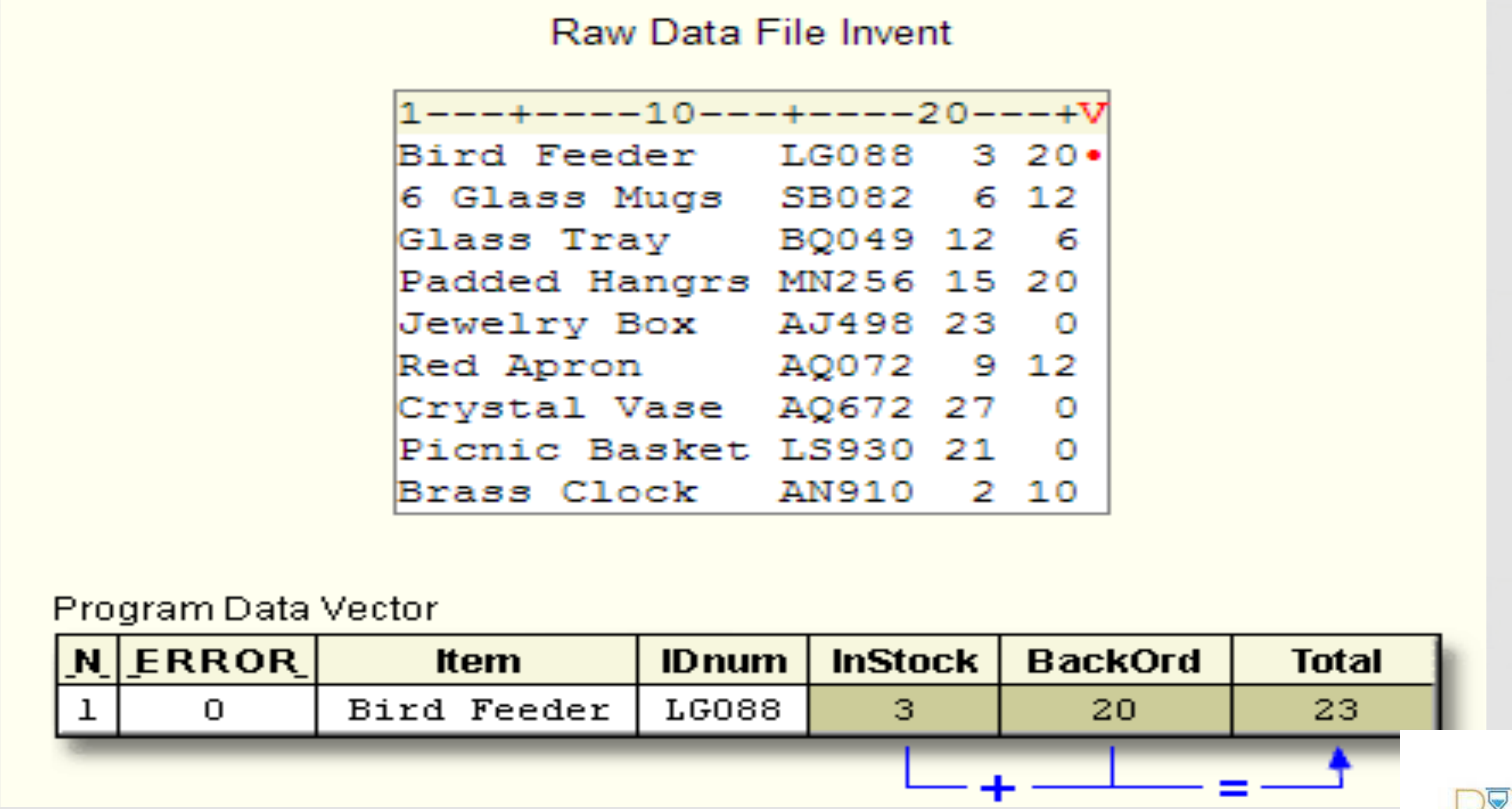
Raw Data File Invent

```
1---+----10---+----20---+V
Bird Feeder   LG088  3 20•
6 Glass Mugs  SB082  6 12
Glass Tray    BQ049 12  6
Padded Hangrs MN256 15 20
Jewelry Box   AJ498 23  0
Red Apron     AQ072  9 12
Crystal Vase  AQ672 27  0
Picnic Basket LS930 21  0
Brass Clock   AN910  2 10
```

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0	Bird Feeder	LG088	3	20	•

- Next, the assignment statement executes. The values for InStock and BackOrd are added to produce the values for Total.



Raw Data File Invent

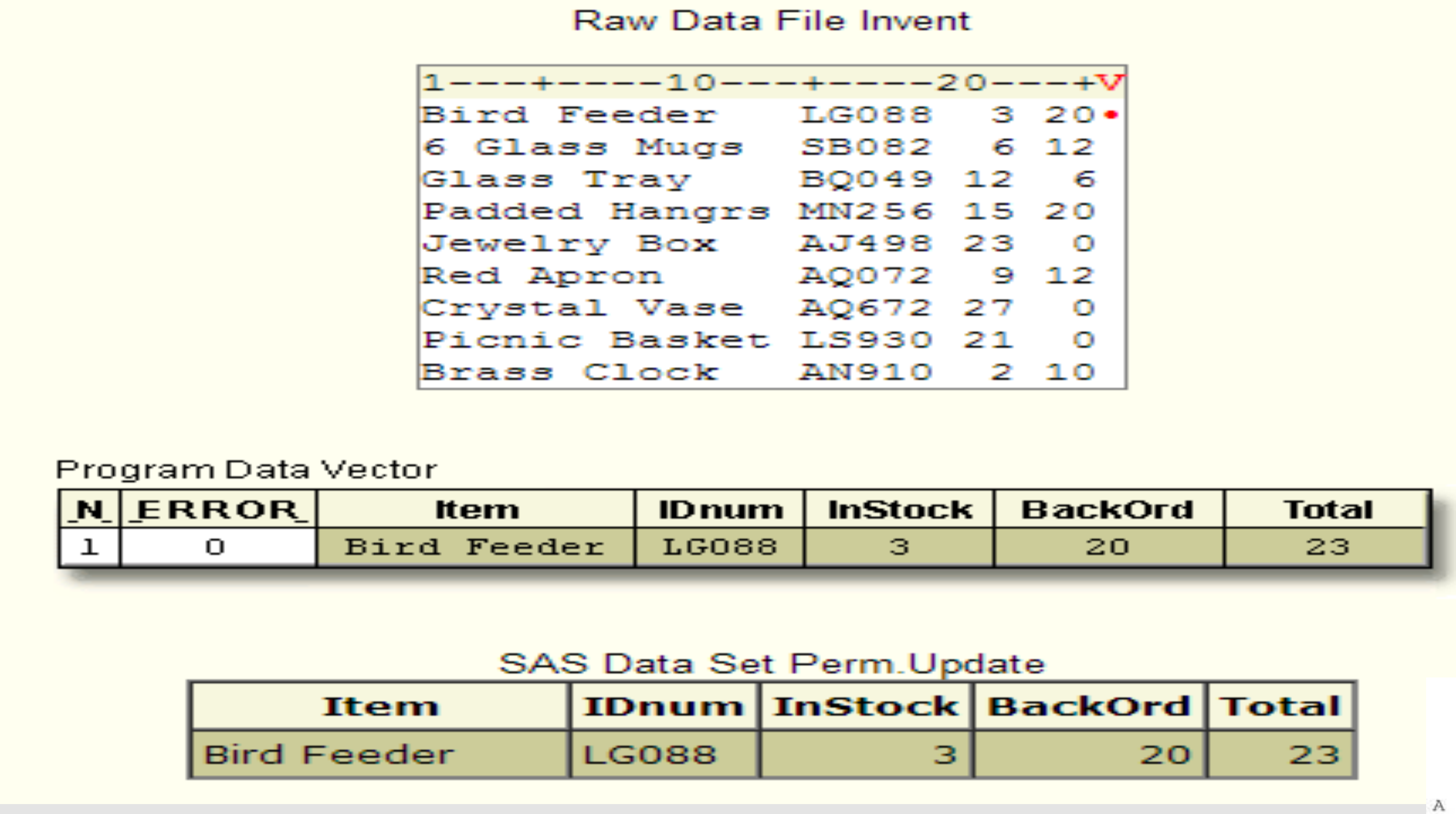
```
1---+----10---+----20---+V
Bird Feeder    LG088  3 20•
6 Glass Mugs   SB082  6 12
Glass Tray     BQ049 12  6
Padded Hangrs  MN256 15 20
Jewelry Box    AJ498 23  0
Red Apron      AQ072  9 12
Crystal Vase   AQ672 27  0
Picnic Basket  LS930 21  0
Brass Clock    AN910  2 10
```

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0	Bird Feeder	LG088	3	20	23

End of the DATA Step

At the end of the DATA step, several actions occur. First, the values in the program data vector are written to the output data set as the first observation.



Raw Data File Invent

```
1---+----10---+----20---+V
Bird Feeder    LG088  3 20•
6 Glass Mugs   SB082  6 12
Glass Tray     BQ049 12  6
Padded Hangrs  MN256 15 20
Jewelry Box    AJ498 23  0
Red Apron      AQ072  9 12
Crystal Vase   AQ672 27  0
Picnic Basket  LS930 21  0
Brass Clock    AN910  2 10
```

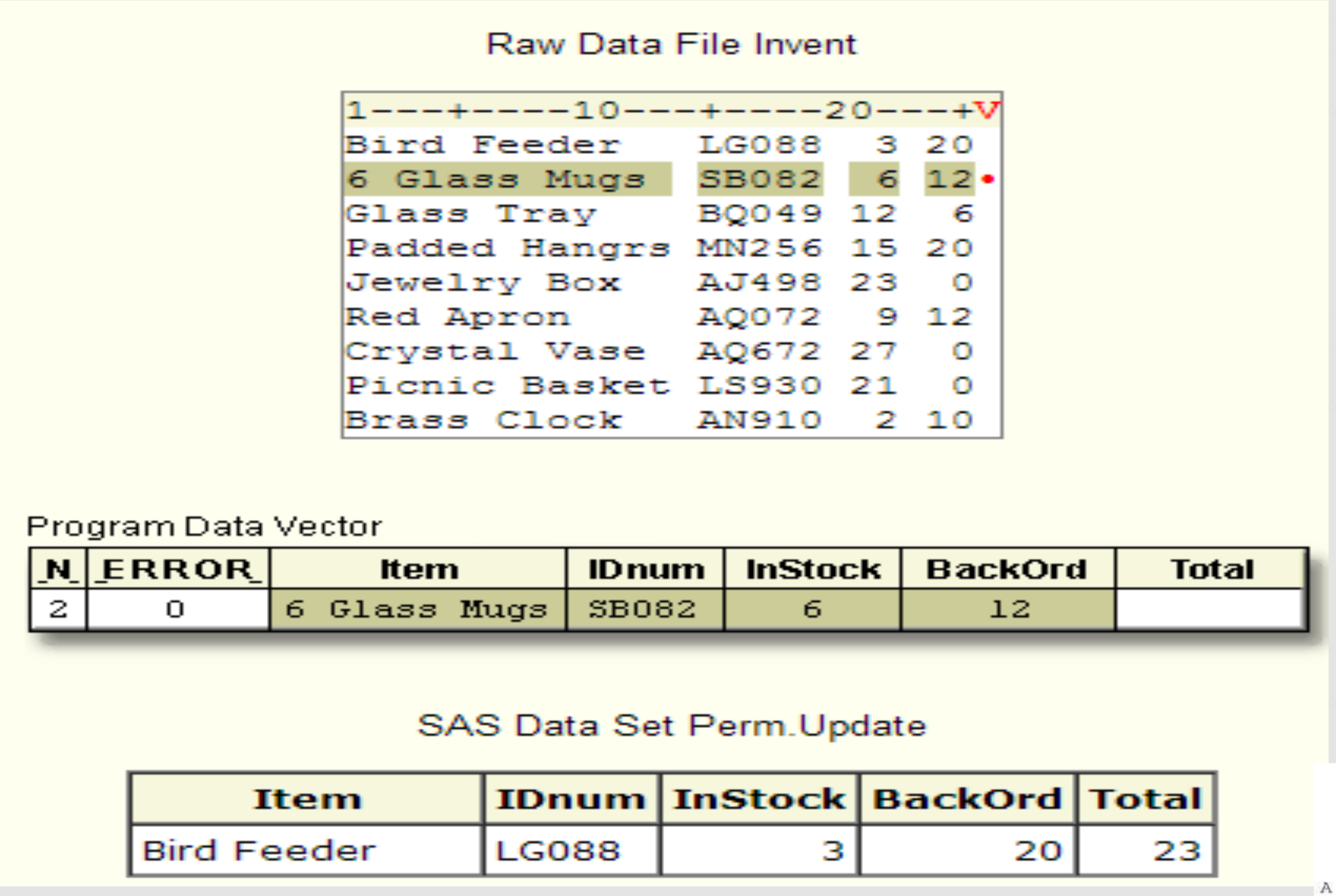
Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
1	0	Bird Feeder	LG088	3	20	23

SAS Data Set Perm.Update

Item	IDnum	InStock	BackOrd	Total
Bird Feeder	LG088	3	20	23

As the INPUT statement executes for the second time, the values from the second record are held in the input buffer and then read into the program data vector.



Raw Data File Invent

```
1---+----10---+----20---+V
Bird Feeder   LG088  3 20
6 Glass Mugs  SB082  6 12
Glass Tray    BQ049 12  6
Padded Hangrs MN256 15 20
Jewelry Box   AJ498 23  0
Red Apron     AQ072  9 12
Crystal Vase  AQ672 27  0
Picnic Basket LS930 21  0
Brass Clock   AN910  2 10
```

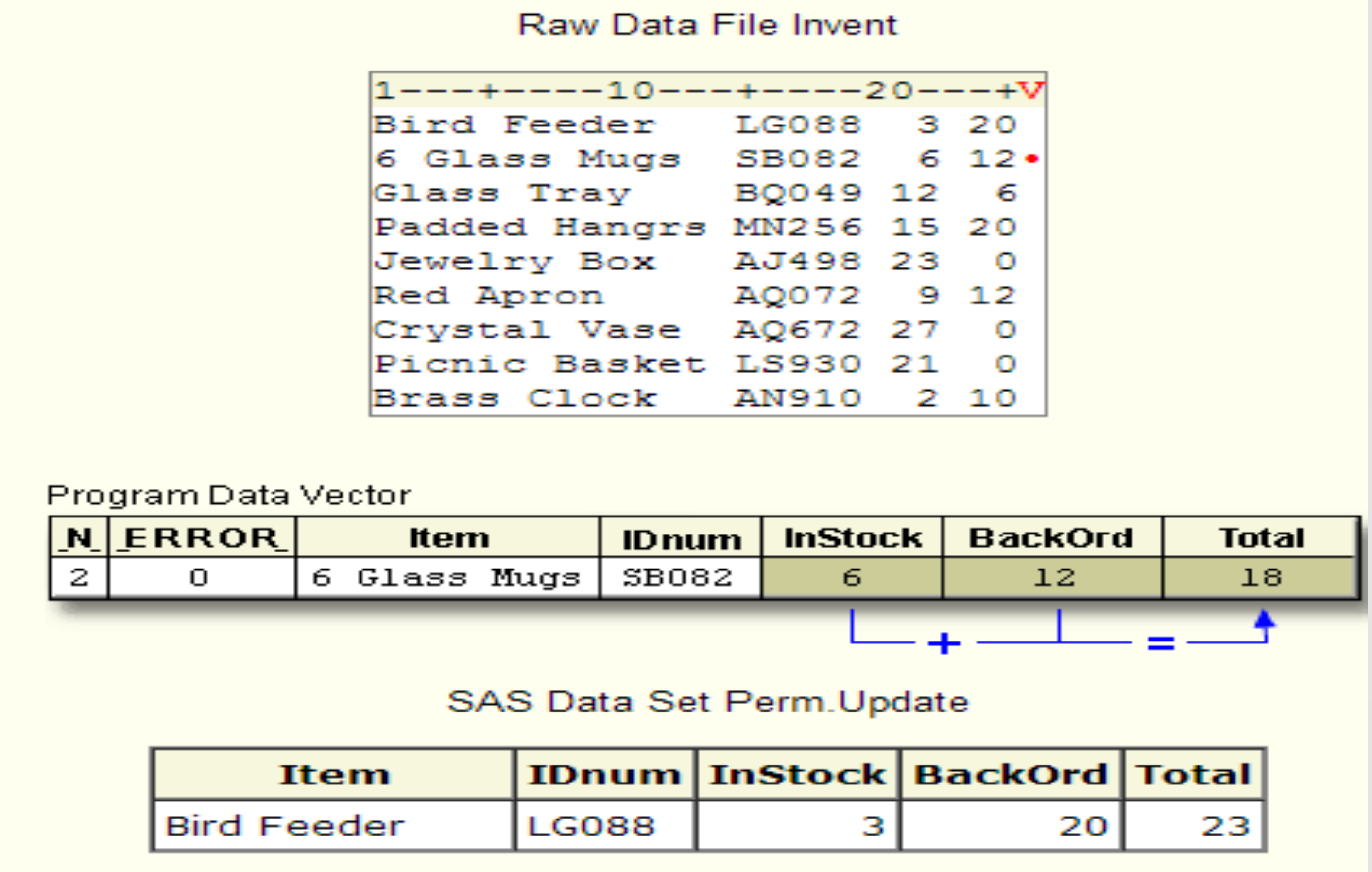
Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
2	0	6 Glass Mugs	SB082	6	12	

SAS Data Set Perm.Update

Item	IDnum	InStock	BackOrd	Total
Bird Feeder	LG088	3	20	23

- Next, the value for Total is calculated based on the current values for InStock and BackOrd. The RUN statement indicates the end of the DATA step.



Raw Data File Invent

```
1---+----10---+----20---+V
Bird Feeder    LG088   3 20
6 Glass Mugs   SB082   6 12•
Glass Tray     BQ049  12  6
Padded Hangrs  MN256  15 20
Jewelry Box    AJ498  23  0
Red Apron      AQ072   9 12
Crystal Vase   AQ672  27  0
Picnic Basket  LS930  21  0
Brass Clock    AN910   2 10
```

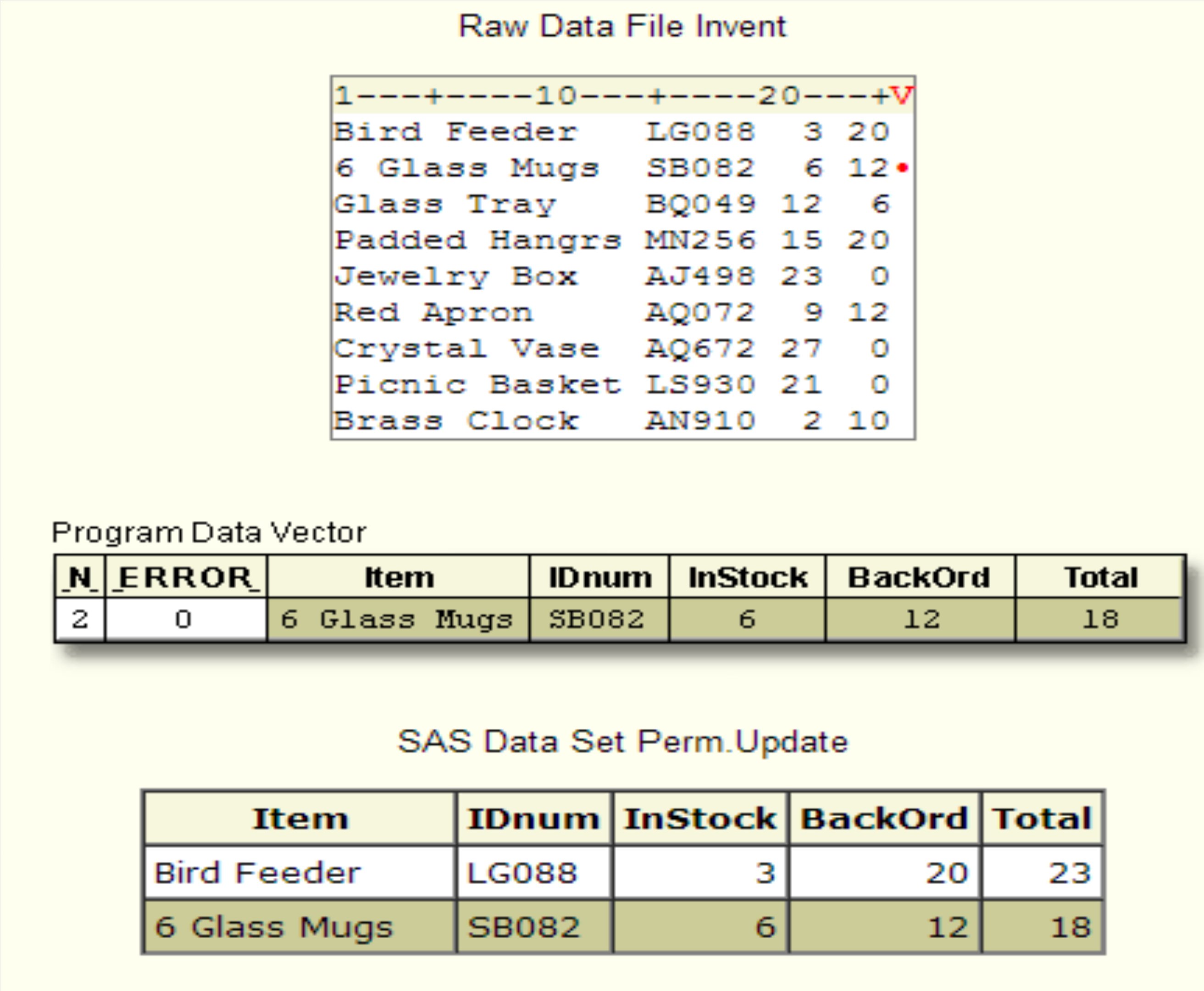
Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
2	0	6 Glass Mugs	SB082	6	12	18

SAS Data Set Perm.Update

Item	IDnum	InStock	BackOrd	Total
Bird Feeder	LG088	3	20	23

At the bottom of the DATA step, the values in the program data vector are written to the data set as the second observation.



Raw Data File Invent

```
1---+----10---+----20---+V
Bird Feeder    LG088   3 20
6 Glass Mugs   SB082   6 12•
Glass Tray     BQ049  12  6
Padded Hangrs  MN256  15 20
Jewelry Box    AJ498  23  0
Red Apron      AQ072   9 12
Crystal Vase   AQ672  27  0
Picnic Basket  LS930  21  0
Brass Clock    AN910   2 10
```

Program Data Vector

_N_	_ERROR_	Item	IDnum	InStock	BackOrd	Total
2	0	6 Glass Mugs	SB082	6	12	18

SAS Data Set Perm.Update

Item	IDnum	InStock	BackOrd	Total
Bird Feeder	LG088	3	20	23
6 Glass Mugs	SB082	6	12	18

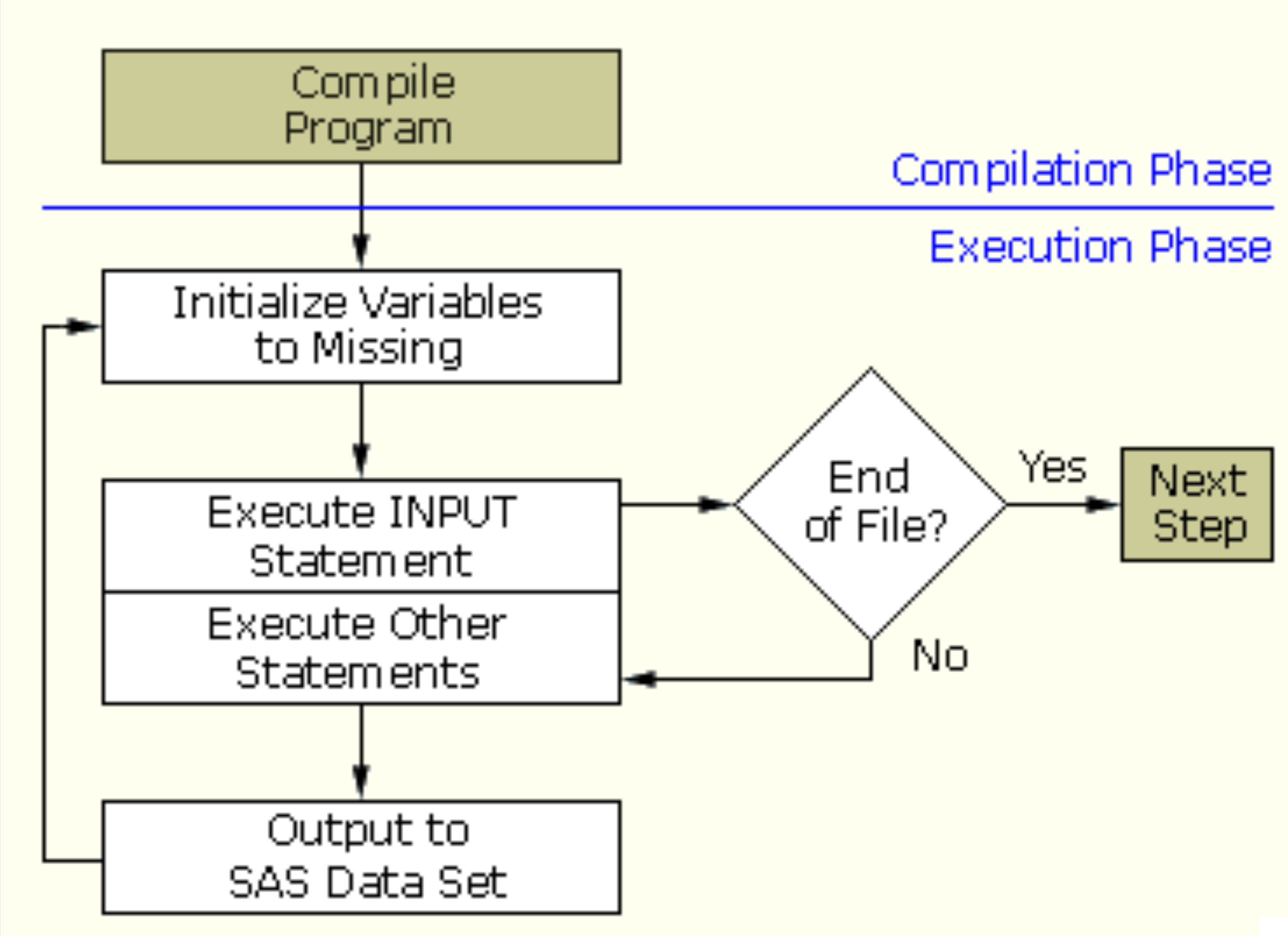
End-of-File Marker

The execution phase continues in this manner until the end-of-file marker is reached in the raw data file. When there are no more records in the raw data file to be read, the data portion of the new data set is complete.

Summary of the Execution Phase

- During the execution phase variables in the program data vector are initialized to missing before each execution of the DATA step
- Each statement is executed sequentially
- The INPUT statement reads the next record from the external file identified by the INFILE statement, and it writes the values into the program data vector
- Other statements can then further modify the current observation
- The values in the program data vector are written to the SAS data set at the end of the DATA step
- Program flow is returned to the top of the DATA step
- The DATA step is executed until the end-of-file marker is reached in the external file.

`



Compile Program

Compilation Phase

Execution Phase

Initialize Variables to Missing

Execute INPUT Statement

Execute Other Statements

End of File?

Yes

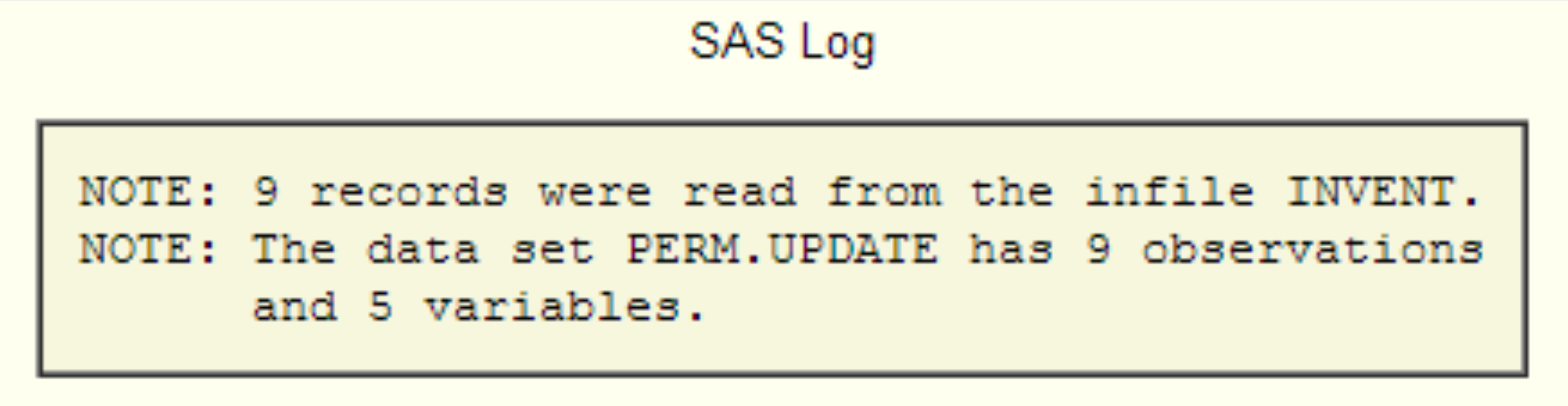
No

Next Step

Output to SAS Data Set

End of the Execution Phase

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.



SAS Log

```
NOTE: 9 records were read from the infile INVENT.
NOTE: The data set PERM.UPDATE has 9 observations
      and 5 variables.
```