Understanding the SAS® DATA Step and the Program Data Vector
This presentation was written by Systems Seminar Consultants, Inc.

SSC specializes SAS software and offers SAS:
  • Training Services
  • Consulting Services
  • Help Desk Plans
  • Newsletter subscriptions to The Missing Semicolon™.

SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. The Missing Semicolon is a trademark of Systems Seminar Consultants, Inc.
“The picture’s pretty bleak, gentlemen. … The world’s climates are changing, the mammals are taking over, and we all have a brain about the size of a walnut.”
Abstract

The SAS system is made up of SAS PROCs and the DATA Step

The DATA Step has:
- an excellent, full fledged programming language
- statements to read and write almost any type of data value
- Conversion and calculation of new data
- Looping
- Interfaces
- Arrays
- Much, much more.

In many ways, the design of the DATA step along with its powerful statements, is what makes the SAS language so popular.
This paper will address:
• How the DATA step fits with the rest of the SAS System
• DATA step assumptions and defaults
• internal structures such as buffers, and the Program Data Vector
• compiler statements
• executable statements
Introduction

The SAS system’s origins are in the 1960’s and 1970’s with:

- James Goodnight
- John Sall
- A. J. Barr
- others

Concepts in the design include:

- “self defining files”
- a system of default assumptions
- procedures for commonly used routines
- data handling step that would evolve into the SAS DATA step
Introduction

The DATA step in my opinion:

• extremely simple
• elegant design
• continues today
• 30 years of enhancements.
Structure of SAS

SAS consists of:
1. a data handling language (DATA step)
2. a library of pre-written procedures (PROC step)
Purpose of the DATA Step

“Get the data in shape” for later PROCs and DATA steps.
• SAS PROCs can only read SAS datasets
• We might have some other type of file to process
• SAS dataset has built in descriptor that keeps track of names and attributes
• later steps don’t have to remember as many details

If we don’t have well defined data
• DATA step gives us the power to read and write virtually any kind of file
• We can do calculations and computations on a single row of data
• It has a very powerful data handling language
DATA steps can read and write most types of data stored on your computer.

Notes:
• DATA step output is usually a SAS dataset but can be other files.
• Access to non-SAS database management systems requires a SAS/ACCESS product.
Many assumptions are made to save time and effort.

As computer scientist I study and use many programming languages
• Early on I was intrigued by the cleverness and common sense of SAS
• Many tedious programming tasks were eliminated through defaults
• System still provided a means to override those defaults when necessary
• Our job as a DATA step programmer then is different from other languages

In many ways our tasks are:
• understanding the defaults
• knowing how to work with them
• override them as necessary.
Default Assumptions (continued)

Examples Of SAS Defaults:

• Handling compile and execution naming and storage details
• A dataset descriptor that makes SAS datasets “self defining”
• Generating data set names if omitted
• When reading a data set, assume most recently created dataset if not specified
• Processing all the rows and columns in a file
• Automatically opening and closing of files
• Automatically controlling data initialization
• DATA step looping, data set output, and end of file checking
Default Assumptions (continued)

Examples Of SAS Defaults:

• Automatically defining storage areas for each variable referenced without need to predefine them
• A default length of 8 was assumed for all variables
• A assumption that a variable is numeric if not specified
• LIST input assumed that data values would be separated by blanks rather than specifying exact columns
• SUBSETTING IF statements which imply to continue processing if a condition is true, else delete the observation. (Ex. If rate > 10; )
• When no comparison is made in an IF statement, assume to be checking for 1 (true) (Ex. If eof then put ‘At end’;)
• Abreviated sum statements. (Ex. Salestot+sales)
Compiling a DATA Step

As most languages, the DATA step is first compiled then executed.

- Languages can be compiled, interpreted, SAS is a hybrid language
- Some features from other languages
- Some unique features
- Sometimes difficult to separate compile versus execution events with SAS
- DATA step compiler examines SAS statements for syntax data structures
- generates an executable program
- Unique to SAS compiler checking for the existence of resources
- Assumptions that it “inserts” into the source code.
Data Structures

“Getting the data in shape” needs data structures to hold data as processed.

- All computer languages need to address this
- Each language may name the structures differently
- There is a lot of similarity in the way most languages store data.
A Typical SAS Job

data softsale;
   infile rawin;
   input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;

Understanding the SAS® DATA Step and the Program Data Vector
Raw File Buffers

DATA steps reading/writing “raw” or non-SAS data need memory buffers.
• Needed to temporarily hold at least one input record at a time.
• Also times when multiple lines of input can be held in buffers
• Allows the program to logically read later rows before earlier ones.
• Buffer contains the complete input and output record, regardless of whether the INPUT statement reads all of the columns.
• SAS datasets and RDMS (which usually appear as SAS datasets) do not use raw buffers as the files are already in “shape”.

Understanding the SAS® DATA Step and the Program Data Vector 17
data softsale;
  infile rawin;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;

**Understanding the SAS® DATA Step and the Program Data Vector**
The LOGICAL PROGRAM DATA VECTOR (PDV)

A second memory area for data manipulation, conversion, refinement.

Areas are needed for:
- Inputting and input formatting (informatting) desired variables
- Revising existing values
- Computing new variables
- System indicators and flags
- Called Logical Program Data Vector (PDV).
- Many languages have a similar working area. (COBOL Working Storage).
- “Retained” and “Non-retained” variables are stored in separate physical program data vectors, together make Logical Program Data Vector
- All variables referenced be automatically defined in the PDV by compiler using characteristics from the first reference of a variable.

Example:  agemo=age/12;
The LOGICAL PROGRAM DATA VECTOR (PDV)

data softsale;
  infile rawin;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;

Understanding the SAS® DATA Step and the Program Data Vector
The Final SAS Dataset

A “self-defining” dataset.

• The dataset descriptor contains attributes for all kept variables plus dataset labeling information.
• The Dataset Data portion contains data values for all output rows and kept columns.
• Pseudo-variables are not kept.
The Final SAS Dataset

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>Error</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETH</td>
<td>H</td>
<td>12</td>
<td>4822.12</td>
<td>982.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOHN</td>
<td>H</td>
<td>7</td>
<td>678.43</td>
<td>150.11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```sas
data softsale;
  infile rawin;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;
```
Variable Attributes

The compiler assigns each defined variable several characteristics.

- Relative variable number
- Position in the dataset
- Name
- Data type
- Length in bytes
- Informat
- Format
- Variable label
- Flags to indicate dropping, retaining, handling of missing values for each variable
Data Types and Conversion

All SAS values are one of two data types: numeric or character.

- Hundreds of different data types can be read or written via the PDV, only two stored
- In PDV, SAS Dataset every character value is stored as a native EBCDIC or ASCII value with length between 1 and at least 32767
- Numerics stored as double precision floating point values with length between 3 and 8 bytes.
- Storing only two data types greatly simplifies things for SAS datasets
- Moves the complication of converting different data types (packed, binary, etc.) to the INPUT and PUT statement along with appropriate INFORMATS and FORMATS.
- Floating point for numbers with a length of 8 allows for storage of very large numbers (or small) without overflow
- Floating point does have minor mathematical issues of its own.
Data Types and Conversion

data softsale;
  infile rawin;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;

### Understanding the SAS® DATA Step and the Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>Error</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETH</td>
<td>H</td>
<td>12</td>
<td>4822.12</td>
<td>982.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOHN</td>
<td>H</td>
<td>7</td>
<td>678.43</td>
<td>150.11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pseudo Variables

Special variables that the compiler creates that are not added to output file.

Partial list:
_N_ contains the number of times the DATA step has looped.
_ERROR_ is set to 0 if there were no input errors, otherwise 1.
FIRST.<variable> showing beginning of control break
LAST.<variable> showing end of control break
_INFILE_ contains entire input buffer

Others can be requested by the programmer to:
• Detect end of file
• Access control blocks
• Access and alter system information
### Pseudo Variables

**data softsale;**

```sas
inFILE rawin;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;
```

#### Understanding the SAS® DATA Step and the Program Data Vector

<table>
<thead>
<tr>
<th>PROGRAM DATA VECTOR</th>
<th>DATASET DESCRIPTOR PORTION (DISK)</th>
<th>DATASET DATA PORTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: NAME</td>
<td>Name: NAME</td>
<td>BETH</td>
</tr>
<tr>
<td>Type: CHAR</td>
<td>Type: CHAR</td>
<td>H</td>
</tr>
<tr>
<td>Length: 10</td>
<td>Length: 10</td>
<td>12</td>
</tr>
<tr>
<td>Informat: D</td>
<td>Informat: D</td>
<td>4822.12</td>
</tr>
<tr>
<td>Label:</td>
<td>Label:</td>
<td>982.10</td>
</tr>
<tr>
<td>Flags:</td>
<td>Flags:</td>
<td>D</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DATA VECTOR</th>
<th>DATASET DESCRIPTOR PORTION (DISK)</th>
<th>DATASET DATA PORTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: DIVISION</td>
<td>Name: DIVISION</td>
<td>CHRIS</td>
</tr>
<tr>
<td>Type: CHAR</td>
<td>Type: CHAR</td>
<td>H</td>
</tr>
<tr>
<td>Length: 1</td>
<td>Length: 1</td>
<td>233.11</td>
</tr>
<tr>
<td>Informat: D</td>
<td>Informat: D</td>
<td>BETH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DATA VECTOR</th>
<th>DATASET DESCRIPTOR PORTION (DISK)</th>
<th>DATASET DATA PORTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: YEARS</td>
<td>Name: YEARS</td>
<td>JOHN</td>
</tr>
<tr>
<td>Type: NUM</td>
<td>Type: NUM</td>
<td>H</td>
</tr>
<tr>
<td>Length: 8</td>
<td>Length: 8</td>
<td>678.43</td>
</tr>
<tr>
<td>Informat: D</td>
<td>Informat: D</td>
<td>BETH</td>
</tr>
</tbody>
</table>
Output Datasets

Usually a SAS datafile, but raw files and reports can also be created.

- SAS data sets are “self-defining” via dataset descriptor
- Much simpler operation for the programmer
- SAS automatically builds the structures needed
- Outputs the record at DATA step return.
- In addition, all variables except dropped and pseudo variables are kept
- Descriptor info can be printed with PROC CONTENTS, dictionary tables

Notes:
- If a raw file or report is produced, buffers opposite those of INFILE are used.
- Raw files can pass info to other programs.
Questions About The Data Step

If traditional programming experience is applied this DATA step, questions might be:

Where are the opens and closes?
Where do we write out records?
What is looping?
When does the program stop?
Assumptions Made In The Data Step

The default actions of the DATA step easily accommodates *most* programs.

- All rows are read from files starting with the first record until last record.
- Input values from a previous row are cleared before reading next row.
- All variables referenced will be included on the output file.
- All records will be included on the resulting output file.
- All files should be opened at the beginning and closed at the end of step.
- Programs should not continue to loop if no data is read in previous pass.
Assumptions Made In The Data Step

The SAS compiler makes the following assumption and inserts code to do:

• Immediately upon entry check for infinite looping
• All values from non-SAS files are cleared before executing any statements.
• If any reading statement would read a record after end of file, step stops.
• If the program reaches the last statement in the step, or if a RETURN statement is executed, the current PDV contents (all columns for each row) is output to the SAS data set being built.
• A branch is executed to go to the top and enter the DATA step for another pass.
Assumptions Made In The Data Step

Another view it is that the compiler inserts the blue italic code.

data softsale;

_check for looping, initialize PDV_

infile rawin;

_if at EOF then stop_

Input Name $1-10 Division $12
Years 15-16 Sales 19-25
Expense 28-34 State $36-37;

_output to SAS Dataset_

goto top of DATA step

run;

Notes:
• These save effort and make DATA steps virtually infinite-loop proof.
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

**Logical Program Data Vector**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

**Dataset work.softsale**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Input buffer

 logical
Program
Data Vector

Dataset work.softsale

(Data)

(Descriptor)
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Input buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Logical Program Data Vector

Dataset `work.softsale`

(Data) | Name | Division | Years | Sales | Expense | State |
|-------|-------|---------|-------|-------|---------|-------|

Understanding the SAS® DATA Step and the Program Data Vector 35
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
      Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
    init buffer, PDV
    infile rawin;
    if at EOF then stop
    input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
    output to SAS Dataset
    goto top of Data Step
run;
```

Input buffer

```
1234567890123456789012345678901234567
CHRIS      H   2   233.11    94.12 WI
MARK       H   5   298.12    52.65 WI
SARAH      S   6   301.21    65.17 MN
```

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Logical Program Data Vector</th>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Dataset work.softsale</th>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Input buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Logical Program Data Vector

(Data)

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Data)

Dataset work.softsale

(Descriptor)

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Descriptor)
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Input buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Logical Program Data Vector...

Dataset `work.softsale`

(Descriptor)

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
</table>

(Data)
Executing the DATA Step

```sas
data softsale;
    init buffer, PDV
    infile rawin;
    if at EOF then stop
    input Name $1-10 Division $12 Years 15-16
        Sales 19-25 Expense 28-34 State $36-37;
    output to SAS Dataset
    goto top of Data Step
run;
```

Input buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Data Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 40
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Input buffer

```
CHRIS H 2 233.11 94.12 WI
```

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16 Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV;
  infile rawin;
  if at EOF then stop;
  input Name $1-10 Division $12 Years 15-16 Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset;
  goto top of Data Step;
run;
```

Input buffer:

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Logical Program Data Vector:

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
</tbody>
</table>

Dataset work.softsale:

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Executing the DATA Step

data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;

Input buffer
  CHRIS H 2 233.11 94.12 WI

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
    init buffer, PDV
    infile rawin;
    if at EOF then stop
    input Name $1-10 Division $12 Years 15-16
        Sales 19-25 Expense 28-34 State $36-37;
    output to SAS Dataset
        goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

```
CHRIS    H   2   233.11    94.12 WI
MARK     H   5   298.12    52.65 WI
SARAH    S   6   301.21    65.17 MN
```

Input buffer

```
CHRIS    H   2   233.11    94.12 WI
```

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
</tbody>
</table>

Dataset work.softsale

```
Name     Division | Years | Sales  | Expense | State
CHRIS    H        | 2     | 233.11 | 94.12   | WI
```
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

**Input buffer**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
goto top of Data Step
run;
```

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>
```

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
infile rawin;
  if at EOF then stop
input Name $1-10 Division $12 Years 15-16
  Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
goto top of Data Step
run;
```

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Each row is handled the same way. Let's skip through MARK's input step.

Logical

Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

**Logical Program Data Vector**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

**Dataset work.softsale**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

data softsale;
init buffer, PDV
infile rawin;
if at EOF then stop
input Name $1-10 Division $12 Years 15-16
  Sales 19-25 Expense 28-34 State $36-37;
output to SAS Dataset
goto top of Data Step
run;

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Logical Program Data Vector</th>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td></td>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td></td>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Input buffer

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Logical Program Data Vector

Data Vector

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 56
Executing the DATA Step

data softsale;
   init buffer, PDV
   infile rawin;
   if at EOF then stop
   input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
   output to SAS Dataset
   goto top of Data Step
run;

Input buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Logical Program Data Vector

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Input buffer: 1234567890123456789012345678901234567890...0

Logical Program Data Vector

<table>
<thead>
<tr>
<th>Logical Program Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Vector</td>
<td></td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 58
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

Input buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 59
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

### Logical Program Data Vector

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

### Dataset work.softsale

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

data softsale;
    init buffer, PDV
    infile rawin;
    if at EOF then stop
    input Name $1-10 Division $12 Years 15-16
        Sales 19-25 Expense 28-34 State $36-37;
    output to SAS Dataset
goto top of Data Step
run;

Input buffer

Logical
Program
Data Vector

Dataset work.softsale

Each row is handled the same way. Let's skip through SARAH’S input step.
Executing the DATA Step

```sas
data softsale;
    init buffer, PDV
    infile rawin;
    if at EOF then stop
    input Name $1-10 Division $12 Years 15-16
        Sales 19-25 Expense 28-34 State $36-37;
    output to SAS Dataset
    goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

**Logical Program Data Vector**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

**Dataset work.softsale**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
       Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

---

<table>
<thead>
<tr>
<th>Logical Program Data Vector</th>
<th>Dataset work.softsale</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Division</strong></td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

**Logical Program Data Vector**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

**Dataset work.softsale**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>
```sas
data softsale;
    init buffer, PDV
    infile rawin;
    if at EOF then stop
    input Name $1-10 Division $12 Years 15-16
        Sales 19-25 Expense 28-34 State $36-37;
    output to SAS Dataset
    goto top of Data Step
run;
```

### Logical Program Data Vector

<table>
<thead>
<tr>
<th>Logical Program Data Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

### Dataset work.softsale

<table>
<thead>
<tr>
<th>Descriptor</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Division</td>
</tr>
<tr>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>CHRIS</td>
<td>H</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 65
Executing the DATA Step

```sas
data softsale;

    init buffer, PDV
infil rawin;

    if at EOF then stop
input Name $1-10 Division $12 Years 15-16
    Sales 19-25 Expense 28-34 State $36-37;

    output to SAS Dataset
    goto top of Data Step
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```sas
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16
                Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
  goto top of Data Step
run;
```

**Input buffer**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

**Dataset work.softsale**

<table>
<thead>
<tr>
<th>Name</th>
<th>Division</th>
<th>Years</th>
<th>Sales</th>
<th>Expense</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td>WI</td>
</tr>
<tr>
<td>MARK</td>
<td>H</td>
<td>5</td>
<td>298.12</td>
<td>52.65</td>
<td>WI</td>
</tr>
<tr>
<td>SARAH</td>
<td>S</td>
<td>6</td>
<td>301.21</td>
<td>65.17</td>
<td>MN</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Executing the DATA Step

```
data softsale;
  init buffer, PDV
  infile rawin;
  if at EOF then stop
  input Name $1-10 Division $12 Years 15-16 Sales 19-25 Expense 28-34 State $36-37;
  output to SAS Dataset
go to top of Data Step
run;
```

Input buffer

Logical Program Data Vector .. .

```
Name         Division   Years      Sales    Expense   State
CHRIS        H          2          233.11  94.12     WI
MARK         H          5          298.12  52.65     WI
SARAH        S          6          301.21  65.17     MN
```

Understanding the SAS® DATA Step and the Program Data Vector
Overriding Data Step Assumptions

Not all programs fit scenario above; can we alter behavior of the program?

- The FIRSTOBS= and OBS= system options begin and end logically after the first record and before the last record respectively.
- The RETAIN compiler statement instructs the step to *not* initialize variables.
- The STOP statement (usually used with IF statement) stops step early.
- The DELETE, subsetting IF statements, exit the DATA step early; never reach OUTPUT.
- RETURN exits the DATA step early but does output to the SAS file.
- DROP, KEEP statements, dataset options exclude or include columns.
- GOTO and various DO groups alter the looping path for the program.
Retaining Data Values

Sometimes variables should not be cleared upon exit/reentry.

The RETAIN compiler statement:
- Specifies listed variables should not be initialized
- Can also set an initial value
- If first reference sets compiler attributes
- Is essentially a flag set by compiler to tell execution don’t clear
- Can be coded anywhere in DATA Step
- Variables read with SET, MERGE, or UPDATE, MODIFY are retained.
Retaining Data Values

Example: Read a file with CITY in the first row, RETAIN the value.

```
data softsale;
  infile rawin missover;
  if _N_ = 1 then do;
    input city $2-8;
    delete; end;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
  retain city;
run;
```

Understanding the SAS® DATA Step and the Program Data Vector 71

<table>
<thead>
<tr>
<th>NAME</th>
<th>DIVISION</th>
<th>YEARS</th>
<th>SALES</th>
<th>EXPENSE</th>
<th>CITY</th>
<th><em>ERROR</em></th>
<th><em>N</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>MADISON</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BETH</td>
<td>H</td>
<td>12</td>
<td>4822.12</td>
<td>982.10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOHN</td>
<td>H</td>
<td>7</td>
<td>678.43</td>
<td>150.11</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>DIVISION</th>
<th>YEARS</th>
<th>SALES</th>
<th>EXPENSE</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETH</td>
<td>H</td>
<td>12</td>
<td>4822.12</td>
<td>982.10</td>
<td></td>
</tr>
<tr>
<td>CHRIS</td>
<td>H</td>
<td>2</td>
<td>233.11</td>
<td>94.12</td>
<td></td>
</tr>
<tr>
<td>JOHN</td>
<td>H</td>
<td>7</td>
<td>678.43</td>
<td>150.11</td>
<td></td>
</tr>
</tbody>
</table>
Stopping Early

Normally jobs run until EOF, OBS= or STOP can stop step early.

Example: Stop after 50 records

```sas
data softsale;
  infile rawin;
  if _N_ = > 50 then stop;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
run;
```
Stopping Late

Sometimes a program should continue even if the last record was read. Example: Calculate a percentage of total sales.

```sas
DATA PCTDS;
IF _N_ = 1 THEN
  DO UNTIL(EOF);
    SET CONCAT(KEEP=NAME)
    SALES);
  END=EOF;
  TOTSALES+SALES;
END;
SET CONCAT(KEEP=NAME SALES);
SALESPCT=(SALES*100)/TOTSALES;
RUN;
```

<table>
<thead>
<tr>
<th>CONCAT DATA SET</th>
<th>OBS</th>
<th>NAME</th>
<th>YEARS</th>
<th>SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BETH</td>
<td>12</td>
<td>4822.12</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CHRIS</td>
<td>2</td>
<td>233.11</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>JOHN</td>
<td>7</td>
<td>678.43</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MARK</td>
<td>5</td>
<td>298.12</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ANDREW</td>
<td>24</td>
<td>1762.11</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>BENJAMIN</td>
<td>3</td>
<td>201.11</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>JANET</td>
<td>1</td>
<td>98.11</td>
<td></td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
Default is to output the current row if DATA step reaches implied OUTPUT.

Statements to discard rows:
• DELETE (usually after an IF) leaves the DATA step without OUTPUT.
• False cases of Subsetting IF statements exit without OUTPUTing.
• RETURN exits the step but does OUTPUT.
• Explicit OUTPUT statement OUTPUTs when executed, but no implied OUTPUT at bottom.
• You may prefer those positive statements such as Subsetting IF, OUTPUT etc.
• You might use a negative statement such as DELETE to filter unwanted rows.

Note:
• WHERE also filters rows in engine, not in DATA step.
Example: Only output records with more than 4000 in Sales.

```sas
data softsale;
  if no input last time thru then stop
  initialize PDV
  infile rawin;
    if at EOF then stop
    input Name $1-10
             Division $12
             Years  15-16
             Sales  19-25
             Expense 28-34
             State  $36-37;
  If sales > 4000;
  If sales not gt 4000 then goto top of DATA step
  output to SAS Dataset
  goto top of DATA step
run;
```

Note:
- WHERE also filters rows in engine, not in DATA step.
Losing the Crowd

AS YOU CAN CLEARLY SEE IN SLIDE 397...

GAAAAH!

"POWERPOINT" POISONING.
Accumulating Totals in a DATA Step

We should be able to write formulas to count or sum across observations.

In a DATA step, read the following dataset and do the following:
  • Count all employees and print running counts.
  • Sum hours and print running sums.

```
fileref rawin
  JOE 40
  PETE 20
  STEVE .
  TOM 35
```
data timecard;
    infile rawin;
    input Name $ Hours;
    Ktr=ktr+1;
    Hourstot=hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>OBS</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>PETE</td>
<td>20</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>STEVE</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>TOM</td>
<td>35</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

**Question:** Why didn't the above program work correctly?
Adding values across records has three problems:
1. All variables are set to missing on every record.
2. Totals normally need to start at 0, not missing.
3. When a missing value is added to any value, the result is a missing value.

The SUM statement is a special assignment statement to add values across observations.

SYNTAX:

`variable+expression;`

Notes:
- SUM variables start at 0, not missing.
- SUM variables are retained from pass to pass.
- Any missing values encountered are ignored.
- RETAIN and SUM function could also be used.
data timecard;
    infile rawin;
    input Name $ Hours;
        Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An Accumulation Solution

data timecard;
  infile rawin;
  input Name $ Hours; Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Set at compile time.
data timecard;
    infile rawin;
    input Name $ Hours; Ktr+1; Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td>RM</td>
<td></td>
</tr>
</tbody>
</table>
data timecard;
    infile rawin;
    input Name $ Hours;
    Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An Accumulation Solution

data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>0</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td>RM</td>
<td>0</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
data timecard;
    infile rawin;
    input Name $ Hours; Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 85
data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;
data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
data timecard;
    infile rawin;
    input Name $ Hours; Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>flags</td>
<td></td>
<td></td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PDV</td>
<td></td>
<td></td>
<td>1</td>
<td>40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
</tbody>
</table>
data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>40</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An Accumulation Solution

data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETE</td>
<td>20</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
</tbody>
</table>
data timecard;
   infile rawin;
   input Name $ Hours;
   Ktr+1;
   Hourstot+hours;
run;

proc print data=timecard;
   title 'SOFTCO PAYROLL';
run;

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

data timecard;
    infile rawin;
    input Name $ Hours; Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

```sas
data timecard;
    infile rawin;
    input Name $ Hours; Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### Obs

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 93
data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Obs Name Hours Ktr Hourstot
1  JOE  40  1  40
2  PETE 20  2  60
An Accumulation Solution

data timecard;
    infile rawin;
    input Name $ Hours;
    Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector 96
An Accumulation Solution

data timecard;
infile rawin;
input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td>60</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td>RM</td>
<td>RM</td>
</tr>
</tbody>
</table>

Obs | Name | Hours | Ktr | Hourstot |
1   | JOE  | 40    | 1   | 40       |
2   | PETE | 20    | 2   | 60       |
data timecard;
infile rawin;
input Name $ Hours;
    Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
### An Accumulation Solution

```sas
data timecard;
  infile rawin;
  input Name $ Hours; Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>40</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>RM</td>
<td>60</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>3</td>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
data timecard;
  infile rawin;
  input Name $ Hours; fileref rawin
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>RM</td>
<td>RM</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOM</td>
<td>35</td>
<td>3</td>
<td>60</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

```sas
data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;
```

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td>4</td>
<td>60</td>
</tr>
</tbody>
</table>
```

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

data timecard;
    infile rawin;
    input Name $ Hours; Ktr+1;
    Hourstot+hours;
run;

proc print data=timecard;
    title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td></td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOM</td>
<td>35</td>
<td>4</td>
<td>95</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
<td>4</td>
<td>95</td>
</tr>
</tbody>
</table>

Understanding the SAS® DATA Step and the Program Data Vector
An Accumulation Solution

data timecard;
  infile rawin;
  input Name $ Hours;
  Ktr+1;
  Hourstot+hours;
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>JOE</td>
<td>40</td>
</tr>
<tr>
<td>PETE</td>
<td>20</td>
</tr>
<tr>
<td>STEVE</td>
<td>.</td>
</tr>
<tr>
<td>TOM</td>
<td>35</td>
</tr>
</tbody>
</table>
data timecard;
  infile rawin;
  input Name $ Hours;  
  Ktr+1;               
  Hourstot+hours;      
run;

proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Hours</th>
<th>Ktr</th>
<th>Hourstot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOE</td>
<td>40</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>PETE</td>
<td>20</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>3</td>
<td>STEVE</td>
<td>.</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>TOM</td>
<td>35</td>
<td>4</td>
<td>95</td>
</tr>
</tbody>
</table>
Another Accumulation Solution

Retain and the SUM function can also accumulate correctly.

data timecard;
  infile rawin;
  input Name $ Hours;
  ktr=ktr+1;
  hourstot=sum(hourstot,hours);
  retain Ktr Hourstot 0;
run;
proc print data=timecard;
  title 'SOFTCO PAYROLL';
run;
Compiler Instructions

A series of statements that instruct the compiler to alter attributes.

- In general, declarative statements
- Can be coded in any order
- Allow the program to be very explicit in the definition of SAS structures.

Examples of these statements are:

- LENGTH statement to set a variables internal length
- INFORMAT to set input format
- FORMAT to set output display format
- LABEL to define a variable label
- ATTRIB to define any or all of the above in one statement
- DROP to indicate which variables are to be left behind on SAS file
- KEEP to indicate which variables to include on the SAS file
- RETAIN to set initial values and instruct SAS to never clear
data softsale;
  infile rawin;
  length name $20;
  attr division length=$2;
  format sales comma10.2;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
  drop sales years;
run;

Understanding the SAS® DATA Step and the Program Data Vector
Debugging Features

There are a number of debugging features in the DATA step.

- Excellent Interactive DATA step debugger available.
- Simple DATA step statements that display data.
- The LIST statement can display the input buffer from the most recent INPUT.
- The FILE LOG with PUT can display any text and variable from the PDV in the SAS log.
- The PUTLOG statement can also display text to the SAS log.
- PROC CONTENTS and PROC PRINT/REPORT can be used to display the dataset descriptor and data values of the final dataset.

Notes:
- By putting the LIST and PUT/PUTLOG statements at strategic points in the DATA step may be the simplest and best way to debug.
A Debugging Example

Example: The program below selects 0 records. Which statement is causing the problem?

<table>
<thead>
<tr>
<th>1234567890123456789012345678901234</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETH   H 12 4822.12 982.10</td>
</tr>
<tr>
<td>CHRIS  H 2  233.11  94.12</td>
</tr>
<tr>
<td>JOHN   H 7  678.43  150.11</td>
</tr>
</tbody>
</table>

data softsale;
  infile rawin missover;
  input name $1-10 division $12 years 15-16 sales 19-25
  expense 27-34;
  if sales > 4000;
  if division='h';
run;

NOTE: The data set WORK.SOFTSALE has 0 observations and 5 variables.
A Debugging Example

Now use PUTLOG to display messages and values around the IF statements.

data softsale;
  infile rawin missover;
  input name $1-10 division $12 years 15-16 sales 19-25 expense 27-34;
  putlog '$$$ before sales if ' _n_ = sales= division=;
  if sales > 4000;
  putlog '$$$ before division if ' _n_ = sales= division=;
  if division='h';
run;

$$$ before sales if _N_=1 sales=4822.12 division=H
$$$ before division if _N_=1 sales=4822.12 division=H
$$$ before sales if _N_=2 sales=233.11 division=H
$$$ before sales if _N_=3 sales=678.43 division=H

NOTE: The data set WORK.SOFTSALE has 0 observations and 5 variables.
The SAS DATA step has many, many more statements that can read, write, and process data in almost any form.

• over 500 DATA step functions
• interfaces to the SAS macro system
• much more
• much written about those features
• beyond the scope of this paper to try to cover them all.
• the DATA step is an extremely versatile and full featured programming language.
As powerful and well designed as the DATA step is, it is different from other languages.

- Perhaps a more standardized language such as SQL might be more auditable and more desirable.
- Terrible DATA step code can be written.
- Good design and adequate documentation make a well written program.
- PROC SQL is also a great tool that adds that language’s features to SAS.
- SAS programs can be well written programs that can be used for everything from one time programs to full fledged production programs.
Conclusions

The SAS DATA step is an excellent programming language with unique features and extremely versatile features.
Not Good On the Flight Home

“Say... what's a mountain goat doing way up here in a cloud bank?”
Contact Us
Questions, comments or to subscribe to the free “Missing Semicolon” newsletter.

SYSTEMS SEMINAR CONSULTANTS, INC.
SAS® Training, Consulting, & Help Desk Services

Steven J. First
President

(608) 278-9964
FAX (608) 278-0065
sfirst@sys-seminar.com
www.sys-seminar.com

2997 Yarmouth Greenway Drive • Madison, WI 53711