**Chapter XVI - Reading Raw Data in Fixed Field**

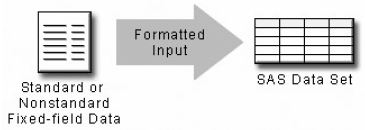
1. Identifying Nonstandard Numeric Data

* Standard Numeric Data

Contains only: numbers; decimal points; numbers in scientific, or E, notation (23E4); minus signs and plus signs.

* Nonstandard numeric data includes

Contains: values that contain special characters, such as percent signs (%), dollar signs ($), and commas (,); date and time values; data in fraction, integer binary, real binary, and hexadecimal forms.



1. Choosing an Input Style

Use formatted input, which combines the features of column input with

the ability to read both standard and nonstandard data.

1. Using Formatted Input

* The General Form of Formatted Input
* Reading both standard and nonstandard data in fixed fields
* Basic code:

**INPUT** <*pointer-control*> *variable* *informat*.;

*pointer-control* positions the input pointer on a specified column

*variable* is the name of the variable

*informat* is the special instruction that specifies how SAS reads the raw data.

* Using the **@n** Column Pointer Control
* The **@*n***is an absolute pointer control that moves the input pointer to a specific column number. The **@** moves the pointer to column *n*, which is the first column of the field that is being read.
* Basic code:

**INPUT *@n*** *variable informat*.;

*variable* is the name of the variable

*informat* is a special instruction that specifies how SAS reads the raw data.

Eg:

the values for FirstName begin in column 9. To point to column 9, use an **@** sign and the column number in the **INPUT** statement：

**INPUT** LastName $7. **@**9 FirstName $5.

* The **+n** Pointer Control
* **+n** 是在current colum number的基础上想前推进n个column, and you can use the notation +(-n) to move the +n pointer backwards
* Basic code:

**INPUT *+n*** *variable informat*.;

*variable* is the name of the variable

*informat* is a special instruction that specifies how SAS reads the raw data.

Eg:

**INPUT** LastName $7. **+**1 FirstName $5. **+**5 Salary comma9. **@**15 JobTitle 3.;

1. Using Informats

* Basic code:

**INPUT** <*pointer-control*> *variable* *informat*.;

1. Each informat contains a w value to indicate the width of the raw data field
2. Each informat also contains a period, which is a required delimiter
3. For some informats, the optional d value specifies the number of implied decimal places
4. Informats for reading character data always begin with a dollar sign ($).

* Selected Informats for Reading Data

|  |  |  |
| --- | --- | --- |
| **PERCENT*w.d*** | **DATE*w*.** | **NENGO*w.*** |
| **$BINARY*w.*** | **DATETIME*w*.** | **PD*w.d*** |
| **$VARYING*w*.** | **HEX*w*.** | **PERCENT*w*.** |
| **$*w.*** | **JULIAN*w*.** | **TIME*w.*** |
| **COMMA*w.d*** | **MMDDYY*w.*** | ***w.d*** |

* Reading Character Values
* The $w. informat enables you to read character data. The w represents the total number of columns that contain the raw data field.
* Eg:

**INPUT** **@**9 FirstName **$**5.;

* Reading Standard Numeric Data

The informat for reading standard numeric data is the ***w.d*** informat.

* Reading Nonstandard Numeric Data
* The **COMMA*w.d***informat is used to read numeric values and to remove embedded
* Eg:

**DATA** sasuser.empinfo;

**INFILE** empdata;

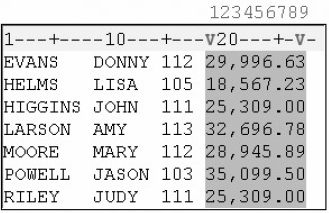
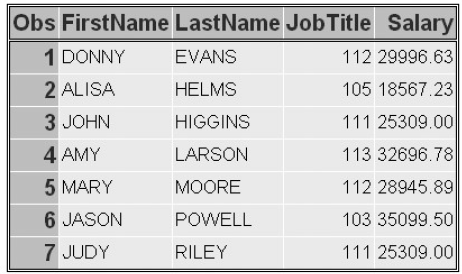
**INPUT @**9 FirstName **$**5. **@**1 LastName **$**7. +7 JobTitle 3. @**19** Salary **comma**.;

**RUN**;

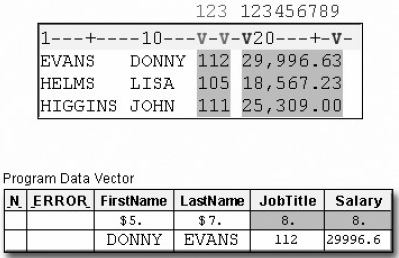
**PROC PRINT** data=sasuser.empinfo;

**RUN**;

***Output from the PRINT Procedure***

* 当SAS读取文件时：

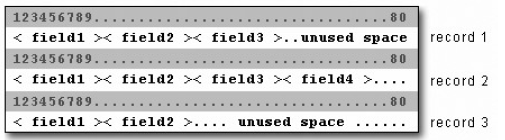


* By default, SAS stores numeric values (no matter how many digits the value contains) as floating-point numbers in 8 bytes of storage. The length of a stored numeric variable is not affected by an informat's width nor by other column specifications in an INPUT statement.

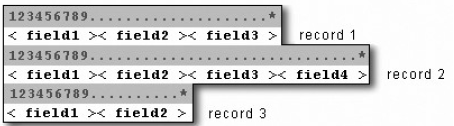
1. Record Formats

* Fixed-Length Records

External files that have a fixed-length record format have an end-of-record marker after a predetermined number of columns. A typical record length is 80 columns.

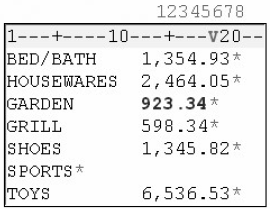


* Variable-Length Records
* Files that have a variable-length record format have an end-of-record marker after the last field in each record.
* When you work with variable-length records that contain fixed-field data, you might have values that are shorter than others or that are missing. This can cause problems when you try to read the raw data into your SAS data set.



* Eg:

**INPUT** Dept $ 1-11 **@**13 Receipts **COMMA**8.;



* **The PAD Option** (50题, Q2)

The **PAD** option pads each record with blanks so that all data lines have the same length.

Eg:

**INFILE** receipts **PAD**;

**INPUT** Dept $ 1-11 **@**13 Receipts **COMMA**8.;



1. Sample Program

**LIBNAME** sasuser 'c:\data\sales';

**FILENAME** vandata 'c:\records\vans.dat';

**DATA** sasuser.vansales;

**INFILE** vandata;

**INPUT** +12 Quarter 1. @1 Region $9. +6 TotalSales comma11.;

**RUN**;

**PROC** **PRINT** data=sasuser.vansales;

**RUN**;

练习

1. Which INPUT statement correctly reads the values for ModelNumber (first field) after the values for Item (second field)? Both Item and ModelNumber are character variables.



1. input +7 Item $9. @1 ModelNumber $5.;
2. input +6 Item $9. @1 ModelNumber $5.;
3. input @7 Item $9. +1 ModelNumber $5.;
4. input @7 Item $9 @1 ModelNumber 5.;、
5. Which INPUT statement correctly reads the numeric values for Cost (third field)?



1. input @17 Cost 7.2;
2. input @17 Cost 9.2.;
3. input @17 Cost comma7.;
4. input @17 Cost comma9.;