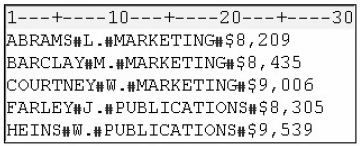
**Chapter XVII - Reading Free-Format Data**

1. Free-Format Data

Raw data that is not arranged in fixed fields，they often separated by blanks or by some other delimiter.

*Raw Data in Free-Format That Is Separated by Blanks*



1. Using List Input

List input can be used to read both standard and nonstandard free-format data

* Basic code:

INPUT *variable* <$>;

*variable* specifies the variable whose value the INPUT statement is to read

$ specifies that the variable is a character variable.

Eg:

**INPUT** Gender $ Age Bankcard FreqBank Deptcard FreqDept;

* Disadvantage:
* Does not specify column locations
* All fields must be separated by at least one blank or other delimiter
* Fields must be read in order from left to right
* You cannot skip or re-read fields.
* Processing List Input

INPUT的默认起点是column 1， variable之间的delimiters是blank space

Eg：

**DATA** sasuser.creditsurvey;

**INFILE** creditcard.dat;

**INOUT** Gender $ Age Bankcard FreqBank Deptcard FreqDept;

**RUN**;

步骤1 – 读取第一个variable from column 1 直到遇到第一个blank space：



步骤2 - SAS scans the record until the next nonblank space is found, and the second value is read until another blank is encountered：



在进行上述步骤时，PDV was created：



步骤3 – 通过**PROC PRINT**将结果print出来



* Working with Delimiters

Use the **DLM**= option in the **INFILE** statement to specify a delimiter other than a blank (the default).

* Basic code:

**DLM=***delimiter(s)*

* + - *delimiter(s)* specifies a delimiter for list input in either of the following forms:
    - *format-of-delimiting-characters'* specifies one or more characters (up to 200) to read as delimiters. The list of characters must be enclosed in quotation marks.
* *character-variable* specifies a character variable whose value becomes the delimiter.
* Eg:

**DATA** sasuser.creditsurvey;

INFILE creditcardcomma.dat DLM=',';

INPUT Gender $ Age Bankcard FreqBank Deptcard FreqDept;

**RUN**;

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

**RUN**;



* 注意：The field delimiter must not be a character that occurs in a data value。如果数字中使用逗号来表示三位数的分隔号，不能使用**DLM** Option
* If you specify a range of character variables, both the variable list and the $ sign must be enclosed in parentheses.

**INPUT** Age (Store1-Store3) ($);

* You can also specify a range of variables using formatted input. If you specify a range of variables using formatted input, both the variable list and the informat must be enclosed in parentheses, regardless of the variable's type.

**INPUT** Age (Score1-Score4) (6.);

* Limitations of List Input
* Although the width of a field can be greater than eight columns, both character and numeric variables have a default length of 8. Character values that are longer than eight characters will be truncated.
* Data must be in standard numeric or character format.
* Character values cannot contain embedded delimiters.
* Missing numeric and character values must be represented by a period or some other character.

1. Reading Missing Values

* Reading Missing Values at the End of a Record
* You can use the **MISSOVER** option in the **INFILE** statement to assign the missing values to variables with missing data at the end of a record. The **MISSOVER** option prevents SAS from reading the next record if, when using list input, it does not find values in the current line for all the **INPUT** statement variables. At the end of the current record, values that are expected but not found are set to missing.
* The MISSOVER option works only for missing values that occur at the end of the record.

Eg:

**DATA** sasuser.creditsurvey;

**INFILE** creditcard missover;

**INPUT** Gender $ Age Bankcard FreqBank Deptcard FreqDept;

**RUN**;

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

**RUN**;

* Reading Missing Values at the Beginning or Middle of a Record
* 因为missing value在中间，如果使用regular list input， SAS会用下一个value来填补这个value的空缺，并set最后几位相应数量的variable为missing value；**MISSOVER**不适用于此处，因为**MISSOVER**止咳作用于missing value在末尾
* 在**DSD** option中默认delimiters是comma；treats two consecutive delimiters as a missing value；removes quotation marks from values.
* 这种情况下，INFILE中写入**Delimiter Sensitive Data** (**DSD**) option会让SAS正确读取数据。在delimiters是comma时，可以不适用**DLM** option来说明delimiters为comma；otherwise：

Eg:

**DATA** sasuser.creditsurvey;

**INFILE** credit3.dat **DSD** **DLM**='\*';

**INPUT** Gender $ Age Bankcard FreqBank Deptcard FreqDept;

**RUN**;

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

**RUN**;





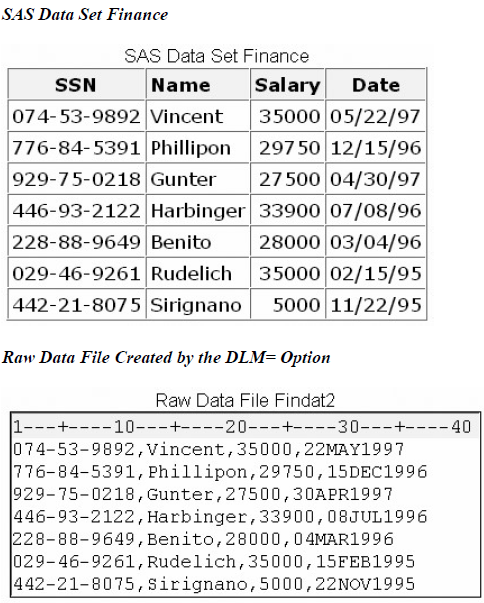
* **DSD** 结合**DLM**= option刻意读取fields that are delimited by blanks

1. Specifying the Length of Character Variables

如果在读取character variable时，character长于默认的8，SAS会在写入PDV时自动截取8个首字母。这种情况下，我们使用LENGTH statement来限定variable的长度，以满足显示所有variable中character的要求



1. Modifying List Input (50题, Q1)

* Two modifiers that can be used with list input:
* The ampersand (&) modifier is used to read character values that contain embedded blanks. (嵌入式空格的意思是在一个词之间会有空格出现，比如说New York， Des Moines)
* The colon (:) modifier is used to read nonstandard data values and character values that are longer than eight characters, but which contain no embedded blanks.
* The ampersand (&) modifier
* The value is read until two or more consecutive blanks are encountered. The & modifier precedes a specified informat if one it used
* Eg:

**DATA** sasuser.cityrank;

**INFILE** topten.dat;

**LENGTH** City $ 12;

**INPUT** Rank city &;

* The colon (:) modifier

The : indicates that values are read until a blank (or other delimiter) is encountered, and then an informat is applied. If an informat for reading character values is specified, the w value specifies the variable’s length, overriding the default length.

Eg:

**DATA** sasuser.cityrank;

**INFILE** topten.dat;

**INPUT** Rank City & $12. Pop86 : comma.;

1. Creating Free-Format Data

* Chapter V explained use PUT statement to write observations from SAS data set to raw data file with column output. PUT also can be used with list output to create free-format raw data file.
* Eg:

在这里，我们将使用DLM= option with a FILE statement来建立一个character-delimited data file

**DATA \_NULL\_;**

**SET** sasuser.finance;

**FILE** 'c:\data\findat2' **DLM**=',';

**PUT** ssn name salary date : **DATE**9.;

**RUN**;

* For creating a simple raw data file, an alternative to the **DATA** step is the **EXPORT** procedure

Basic code:

**PROC EXPORT** DATA = SAS-data-set;

**OUTFILE**=filename<DELIMITER=’delimiter’>;

**RUN**;

* + - SAS-data-set names the input SAS data set
    - Filename specifies the complete path and filename of the output
    - ’delimiter’ specifies the delimiter to separate columns of data in the output file
* Using the DSD option

在FILE dtatement中使用DSD option可以在含有comma的variable前后加上quotation mark

Eg:

**DATA** \_**NULL**\_;

**SET** sasuser.finance;

**FILE** 'c:\data\findat2' DSD;

**PUT** ssn name salary : comma. date : **DATE**9.;

**RUN**;



1. Mixing Input Styles

***Input Styles and the Types of Information They Read***

|  |  |
| --- | --- |
| **Input Style** | **Reads** |
| Column | Standard data values in fixed fields |
| Formatted | Standard and nonstandard data value in fixed fields |
| List | Data value that are not arranged in fixed, but are separated by blanks or other delimiters |

Eg:

***Raw Data Showing Mixed Input Styles***



以上raw data file中，我们需要使用多种input style。

* First field是standard character value 而且是fixed column，所以可以使用column input
* Second 和third fields虽然是fixed columns，当时values需要informat，所以使用formatted input
* Fourth field的observations不是在同一个column中end，所以使用list input。当时其中有一些characters length大 于8，所以需要使用：format modifier。
* Fifth field的begin和end都不是在同一个column，所以需要使用list input

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Description | Staring Column | Field Width | Data Type | Input Style |
| Social Security# | 1 | 11 | Character | Column |
| Date of Birth | 13 | 7 | Date | Formatted |
| Annual Salary | 21 | 6 | Numeric | Formatted |
| Department | 28 | 5 to 9 | Character | List |
| Phone Extension | ?? | 4 | Character | List |

**DATA** sasuser.mixedstyles;

**INFILE** rawdata.dat;

**INPUT** SSN $ 1-11 @13 HireDate **DATE**7. @21 Salary **COMMA**6. Department : $9. Phone $;

**RUN**;

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

**RUN**;



练习

1. Which type of input should be used to read the values in the raw data file that is referenced by the fileref University?



1. column
2. formatted
3. list
4. modified list
5. Which SAS statement correctly reads the values for Flavor and Quantity? Make sure the length of each variable can accommodate the values that are shown.



1. input Flavor & $9. Quantity : comma.;
2. input Flavor & $14. Quantity : comma.;
3. input Flavor : $14. Quantity & comma.;
4. input Flavor $14. Quantity : comma.;
5. Which SAS statement correctly reads the raw data values in order and assigns them to these corresponding variables: Year (numeric), School (character), Enrolled (numeric)?



