03. 导入数据Ⅰ

SAS读取的数据的方法主要有以下几种：

 1. 直接输入；

表格形式输入数据，可以定义变量、设置属性；

程序编辑器data步中直接输入；

 2. 从原始数据文件中创建一个SAS数据集；

data步可以读取任何形式的原始数据文件，也可以用导入向导（Import Wizard）、导入过程（IMPORT procedure）

 3. 将其他软件中的数据文件转换成SAS数据集；

如果安装SAS/ACCESS模块，可以用导入过程和导入向导将Excel、Lotus、dBase和Access文件导入SAS数据集

 4. 直接读取其他软件的数据集；

SAS/ACCESS产品可以不用转换数据格式读取数据库管理系统，包括ORACLE,DB2,INGRES和 SYBASE的数据；使用Excel engine、Access engine和SPSS engine来读取其数据

1. **直接输入**
2. 打开【工具】——【表编辑器】，在表编辑器窗口以表格形式输入数据，可以定义变量、设置属性。（略）
3. data步中直接输入（例如 SAS介绍及基础篇中的例1）

1. **用导入向导**（Import Wizard）**读取文件**

步骤：

1. 打开【文件】——【导入数据】，调出导入向导窗口；
2. 选择要导入的数据类型；
3. 指定要导入的文件位置，SAS默认第一行存放变量名，从第二行开始存放数据（Options可以改变这种默认选择）；
4. 选择数据集要存放的地址，并为数据集命名；
5. （可选）创建一个proc import语句，可以执行它再次导入这个数据。
6. **从外部文件导入数据**
7. 读取空格或分隔符分开的数据

语法：

data 数据集名;

infile ‘文件路径+文件名’ <可选参数>;

input 变量1 变量2 …;

注：infile语句告诉SAS外部数据的存放路径和文件名；

示例：

data students;

infile 'c:\MyRawData\Studens.dat' DLM = ',';

input Name $ Age Height;

注：这是创建临时数据集work.students，若要创建永久数据集，需要指定二级数据集名称。例如，

libname myworks 'D:\SASWorks';

**data** myworks.students;

infile 'c:\MyRawData\Studens.dat' DLM = ',';

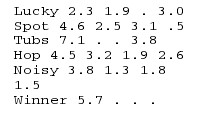
input Name $ Age Height;

* 1. 最简单的数据文件

1. 至少被一个空格，缺失值用“.”表示；
2. 字符串不含空格，少于8个字符；

（SAS默认字符串是8个字符，多于8个字符需要指定列宽）

**例1** 读取路径c:\MyRawData\下的数据文件ToadJump.dat，内容如下：



代码：

**data** toads;

infile 'c:\MyRawData\ToadJump.dat';

input ToadName $ Weight Jump1 Jump2 Jump3;

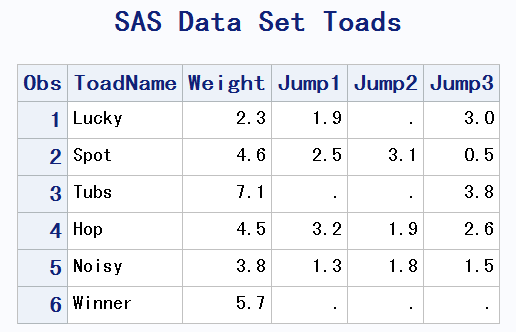
**run**;

**proc print** data = toads;

title 'SAS Data Set Toads';

**run;**

运行结果：



注：（1）Nosiy的数据溢出到第二行了，但这不影响，SAS会按照变量顺序自动跳到下一行读取；

（2）程序将不加选择的逐行逐列的读入所有数据记录。

infile语句可以加上可选参数，

* 1. MISSOVER选项（处理每行数据个数长短不一）

input语句中输入的几个变量，SAS在观测值中就读取几个变量，如果一行未读完，则进入下一行直到输入的变量都读取了变量值。可以让SAS不进入下一行读取，未赋值的变量读为缺失值。

**例2** 读入下面数据（c:\MyRawData\AllScores.dat），一个学生应该有5门课的成绩，但由于最后两门是自学课程，不是所有学生都完成，故而缺失：此时就需要加上MISSOVER。

C:\Documents and Settings\Administrator\Application Data\Tencent\Users\342407528\QQ\WinTemp\RichOle\QSL]Q@5MG{]0O)[C~]JBZDC.jpg

代码：

**data** class102;

infile 'c:\MyRawData\AllScores.dat' MISSOVER;

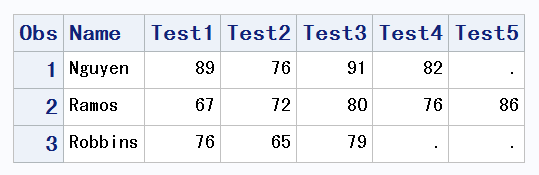
input Name $ Test1 Test2 Test3 Test4 Test5;

**run**;

**proc** **print** data = class102;

**run;**

运行结果：



* 1. DLM和DSD选项

默认读入的数据是空格分隔，若是其它分隔符分隔，在infile语句中加上DLM=’分隔符’ ：

逗号分隔——DLM = ‘,’

制表符分隔——DLM=’09’X （制表符的十六进制值是09）

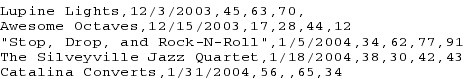
有时后面需要再加上DSD，有三个作用：

a. 忽略引号中数据的“假分隔符”（例2）；

b. 自动将字符串中的引号去掉；

c. 将两个相邻的分隔符当作缺失值来处理。

**例3** 读取路径c:\MyRawData\下的数据文件Bands.csv，内容如下：



注意第3行引号中的逗号并不是分隔符，另外，每行数据长短不一样，所以还需要加上MISSOVER.

代码：

**data** music;

infile 'c:\MyRawData\Bands.csv' DLM = ',' DSD MISSOVER;

input BandName:$30. GigDate:MMDDYY10. EightPM NinePM TenPM ElevenPM;

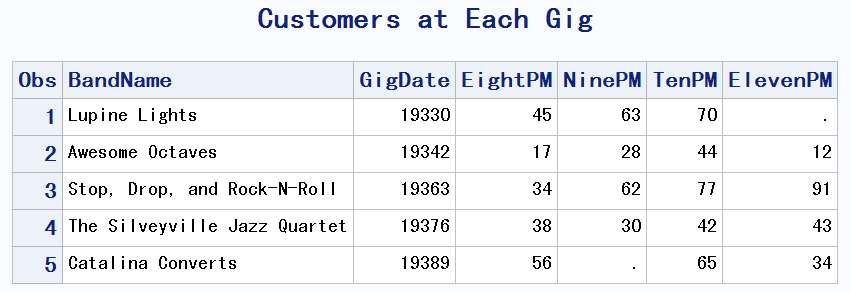
**run**;

**proc** **print** data = music;

title 'Customers at Each Gig';

**run;**

运行结果：



* 1. FIRSTOBS = m OBS=n选项

有的数据文件包括数据的描述，需要用该选项告诉SAS从第m行开始读取到第n行结束。

**例4** 读取如下的数据文件（c:\MyRawData\ IceCreamSales2.dat）：

![C:\Documents and Settings\Administrator\Application Data\Tencent\Users\342407528\QQ\WinTemp\RichOle\](J0D[FCRE(J0O$~OH)T0[8.jpg](data:image/jpeg;base64,)

注意第3行到第5行是有效数据。

代码：

**data** icecream2;

infile 'c:\MyRawData\IceCreamSales2.dat' FIRSTOBS = **3** OBS=**5**;

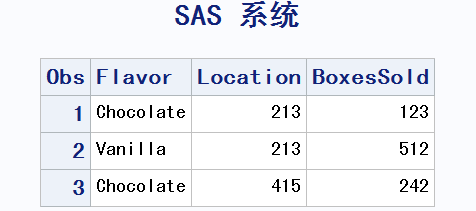
input Flavor $ **1**-**9** Location BoxesSold;

**run**;

**proc** **print** data = icecream2;

**run**;

运行结果：



1. 读取按固定列排列的数据
2. 同一变量的值都占据相同范围的列内；
3. 变量值是字符串或者标准数值。

注：标准数值是指数据、小数点、正负号、和科学计算法的E。逗号数据和日期都不是标准数值。比第1种的优势在于：

* 不要求变量值之间有空格；
* 缺失值可以直接用空格代替；
* 字符串中可以包含空格；
* 可以跳过不需要的变量。

语法：

input 变量1 n-m…;

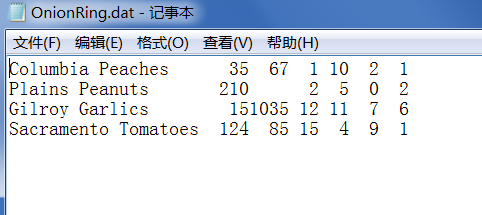
注：“n-m”表示变量1数据所占的列范围，第n列至第m列。

示例：

input Name $ **1**-**10** Age **11**-**13** Height **14**-**18**;

* 1. 标准按固定列排列的数据

**例5** 读取路径c:\MyRawData\下的数据文件OnionRing.dat，内容如下：



代码：

**data** sales;

infile 'c:\MyRawData\OnionRing.dat';

input VisitingTeam $ **1**-**20** ConcessionSales **21**-**24** BleacherSales **25**-**28** OurHits **29**-**31** TheirHits **32**-**34** OurRuns **35**-**37** TheirRuns **38**-**40**;

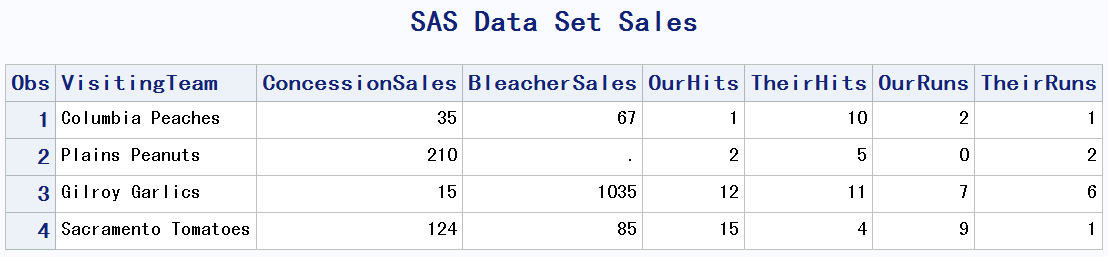
**run**;

**proc print** data = sales;

title 'SAS Data Set Sales';

**run;**

运行结果：



* 1. TRUNCOVER选项

使用按固定列的input或控制格式（下节）的 input输入时，若某行的数据（+空格）没有占到指定列的宽度，可能会转到下一行读取，此时必须用TRUNCOVER选项，以避免发生这种错误。

**例6** 读取如下数据（c:\MyRawData\Address.dat）：

C:\Documents and Settings\Administrator\Application Data\Tencent\Users\342407528\QQ\WinTemp\RichOle\I8X6)YGZ%%)VJ}ZXXC}ENPT.jpg

注意三行的长度都不一样，input中只能指定最长的一行。

代码：

**data** homeaddress;

infile 'c:\MyRawData\Address.dat' TRUNCOVER;

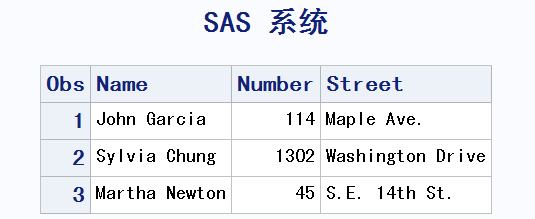
input Name $ **1**-**15** Number **16**-**19** Street $ **22**-**37**;

**run**;

**proc** **print** data = homeaddress;

**run**;

运行结果：



1. 读取非标准格式的数据文件

非标准格式的数据，包括日期数据、“8,765,432”、含美元符号、十六进制数等。

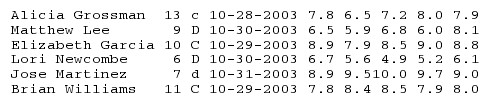
语法：

input 变量名 变量格式 …;

示例：

input Name $ **10.** Age **3.** Height **5.1** BirthDate MMDDYY10.;

**例7** 读取路径c:\MyRawData\下的数据文件Pumpkin.dat，内容如下：



代码：

**data** contest;

infile 'c:\MyRawData\Pumpkin.dat';

input Name $16. Age **3.** +**1** Type $1. +**1** Date MMDDYY10.

(Score1 Score2 Score3 Score4 Score5) (**4.1**);

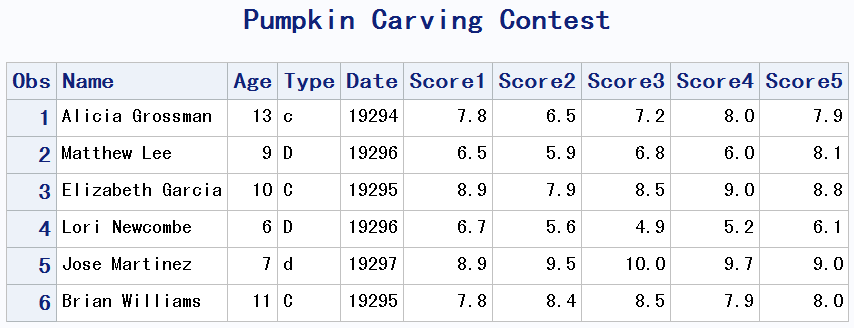
**run**;

**proc print** data = contest;

title 'Pumpkin Carving Contest';

**run**;

运行结果：



程序说明：

（1）“Name $ **10.**”表示字符型变量Name，共占10列宽度，无小数位；

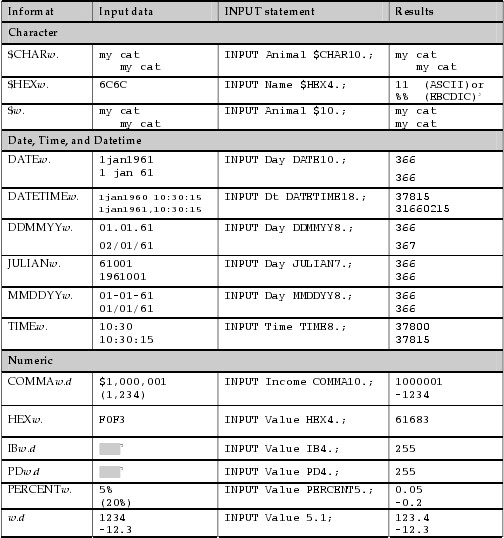
（2）“Height **5.1**”表示数值型变量Height，共占5列宽度，其中1位小数位；

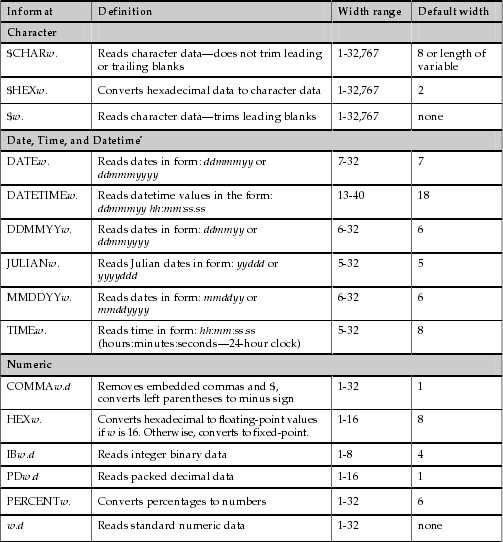
（3）“+1”表示跳过一列，即原始数据中Age后面有一个空格；

（4）“MMDDYY10.”共占10位的日期格式；

（5）Score1 Score2 Score3 Score4 Score5五个变量格式相同，可以用小括号共同指定格式；

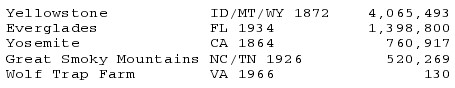
附：变量格式及实例列表





通常可以混合使用前面的三种方式，例如，

**例8** 读取路径c:\MyRawData\下的数据文件NatPark.dat，内容如下：



代码：

**data** nationalparks;

infile 'c:\MyRawData\NatPark.dat';

input ParkName $ **1**-**22** State $ Year @**40** Acreage COMMA9.;

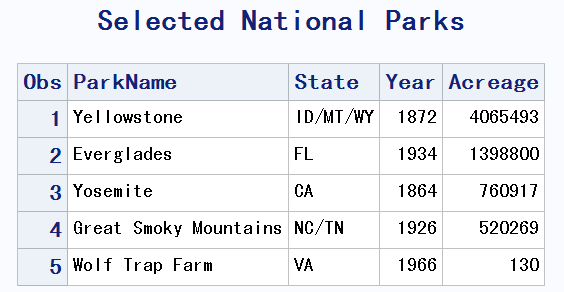
**run**;

**proc** **print** data = nationalparks;

title 'Selected National Parks';

**run**;

运行结果：



程序说明：

符号“@”是列指示器，“@40”告诉SAS在读取Acerage变量之前，移动到第40列去；若没有“@40”，Comma9告诉SAS读取9列，将会读取包括空格在内的9列，这便会导致输出结果有问题：



1. 读取凌乱的数据

有的数据排列混乱，长度不一，不知道从哪列开始，但是要读取的数据前面的字符或单词是固定的。可以利用列指示器“@”+固定字符串，定位位置来读取数据。

语法：

input @ ‘字符串’ 变量名 …;

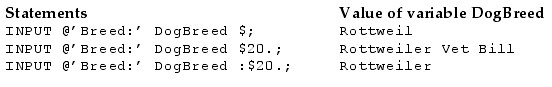
示例：

input @'Breed:' DogBreed $;

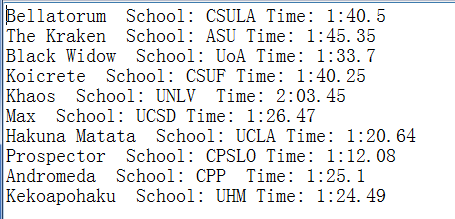
另外，input读取字符串变量默认为8个字符，若超过8个字符，则需要定义长度，定义为$length，在该长度中，空格也算在内。若要使SAS读取过程中遇到空格则不再继续读取，则要在$length.前面加冒号“:”。比如原始数据中有这么一行：

My dog Sam Breed: Rottweiler Vet Bills: $478

用下述述不同方法读取，会有不同结果：



**例9** 读取路径c:\MyRawData\下的数据文件Canoes.dat，内容如下：

****

读取学校名和时间。

代码：

**data** canoeresults;

infile 'c:\MyRawData\Canoes.dat';

input @'School:' School $ @'Time:' RaceTime :STIMER8.;

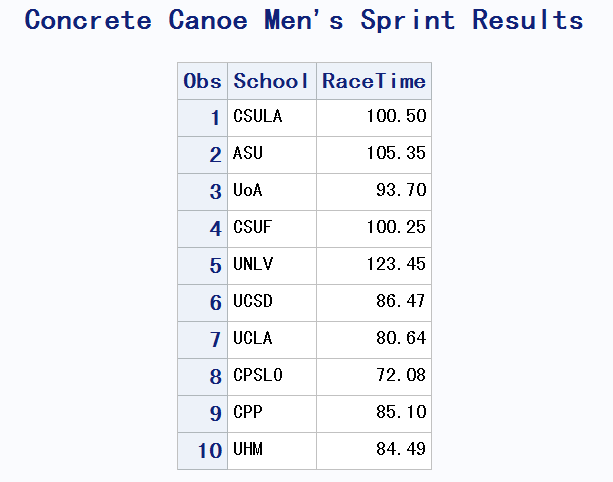
**run**;

**proc** **print** data = canoeresults;

title "Concrete Canoe Men's Sprint Results";

**run**;

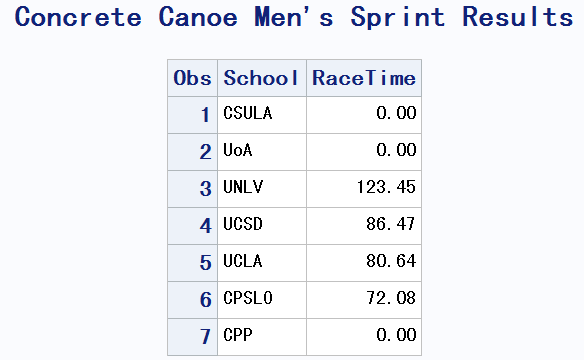
运行结果：（注意：时间转化为以秒为单位）



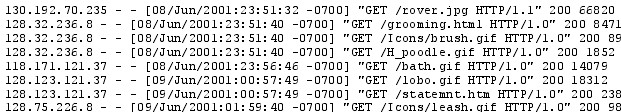
程序说明：

（1）@’School:’和 @’Time:’来指定读取学校名和时间的起始位置；

（2）由于Time占的字节数不同，用了冒号修饰符，:STIMER8. 来读取时间，STIMER8是时间格式。若没有冒号修饰符，当遇到字符位数不够时，SAS将会跳到下一个数据行去接着读取，输出错误结果：



**例10** 读取路径c:\MyRawData\下的数据文件weblogs.txt，内容如下：



读取其中的访问日期和访问的文件名。

代码：

**data** weblogs;

infile 'C:\MyRawData\weblogs.txt';

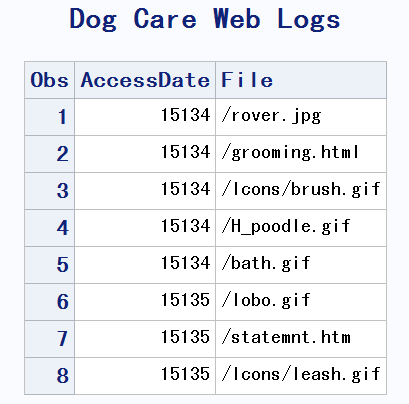
input @'[' AccessDate DATE11. @'GET' File :$20.;

**proc** **print** data = weblogs;

title 'Dog Care Web Logs';

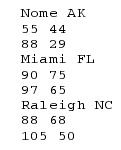
**run**;

运行结果：



1. 读取非“一行一个观测值”数据、有选择地读取观测值
   1. 跨行观测值数据

**例11** 读取路径c:\MyRawData\下的数据文件Temperature.dat，内容如下：



注意一个观测值占据3行。

代码：

**data** highlow;

infile 'c:\MyRawData\Temperature.dat';

input City $ State $

/ NormalHigh NormalLow

#**3** RecordHigh RecordLow;

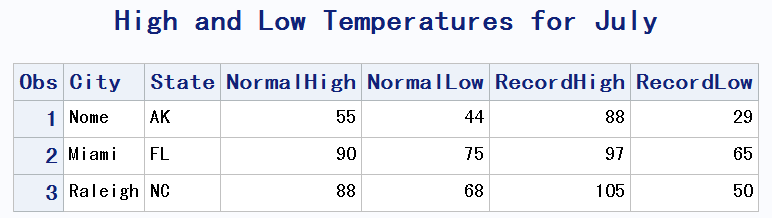
**run**;

**proc** **print** data = highlow;

title 'High and Low Temperatures for July';

**run**;

运行结果：



程序说明：

（1）SAS先读取第一行的city变量和state变量，斜线 /告诉SAS移动到下一行的第一列，以便读取normalhigh和normallow.

（2）#3告诉SAS移动到第三行的第一列以便继续读取观测值的recordhigh变量和recordlow变量。

* 1. 一行有多个观测值的数据

当一行出现多个观测值时，可以在input语句结尾加一个停止符号@@.

**例12** 读取路径c:\MyRawData\下的数据文件Precipitation.dat，内容如下：

C:\Documents and Settings\Administrator\Application Data\Tencent\Users\342407528\QQ\WinTemp\RichOle\8N1[IS4F@YGQJ%%JRCFRSRM.jpg

注意第一行包含2个观测值。

代码：

**data** rainfall;

infile 'c:\MyRawData\Precipitation.dat';

input City $ State $ NormalRain MeanDaysRain @@;

**run**;

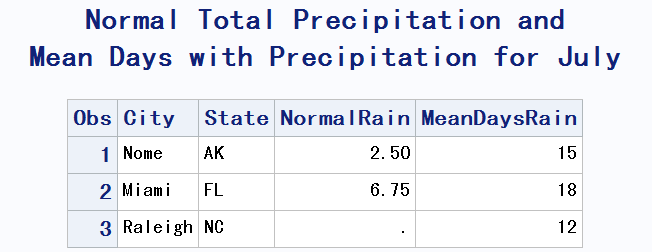
**proc** **print** data = rainfall;

title1 'Normal Total Precipitation and';

title2 'Mean Days with Precipitation for July';

**run**;

运行结果：

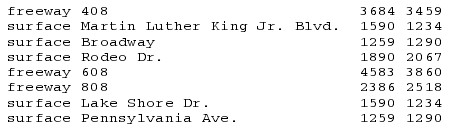


* 1. 有选择地读取部分观测值

有时候只需要读取原始数据的部分观测值，比如只需要年鉴中的女性数据、收入超过10万的人口数据等。实现方法：

在SAS读取某一行观测值时，先读取到“判断”变量，然后在input语句结尾加符号@，（叫做trailing at），告诉SAS先停在此行，同时用if语句检测此观测值是否满足条件，若满足，那么可以再用一个input语句来读取其它变量。

**例13** 交通数据（c:\MyRawData\Traffic.dat）包含街道的类型（freeways和surface）、街道名称、早晨每小时的机动车流动量、晚上每小时机动车流动量。



要求只读取freeway的数据。

代码：

**data** freeways;

infile 'c:\MyRawData\Traffic.dat';

input Type $ @;

if Type = 'surface' then DELETE;

input Name $ **9**-**38** AMTraffic PMTraffic;

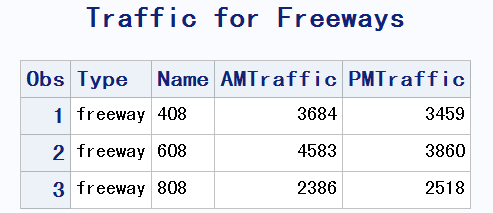
**run**;

**proc** **print** data = freeways;

title 'Traffic for Freeways';

**run**;

运行结果：



注：@的作用类似于@@，都是行停留指示符，不同地方在于停留多久，@能使SAS停留到下一个input语句（也不换行），@@能使停留的时间到下一个data步（也不换行）。

比如这段代码：

**data** test;

infile cards ;

input x @;

input y;

input z @@;

cards;

1 2 3 4 5 6

7 8 9 10 11 12

13 14 15 16 17

;

**run**;

**proc** **print** data = test;

**run**;

test输出结果就是：

