Sas 考试复习要点

西南石油大学理学院-谭兵

第二章：Sas 编程基础

明白逻辑库和数据集的概念，

逻辑库：

SAS 逻辑库是一个逻辑概念，本身不是物理实体，它对应的实体是操作系统下一个文件夹或几个文件夹中的一组 SAS 文件。创建逻辑库

创建逻辑库

```
libname resdata 'd:\resdata';

run;
```

数据集

自动变量

自动变量是由数据步语句自动创建的。这些自动变量被加入到程序数据向量中，但是并不输出到数据集中。自动变量在重复过程中被保留，而不是被设定为缺失。

_Numeric_所有的数值变量

```
data resdata.class3;
set resdata.class;
keep _numeric_;
run;
```

_character_所有的字符串变量

```
data resdata.class3;
set resdata.class;
keep _character_;
run;
```

第三章：SAS 函数与 CALL 子程序

自变量名缩写方法：
　函数名(OF 变量名 1…变量名 n)

概率分布函数

```
data;
Y=probnorm(0.96);
put Y;
run;
```

样本统计函数

Sum（of x1-x10），means（of x1-x10）

随机数函数

```
data rv;
retain _seed_ 0; /*seed用于伪随机数,在0上游走*/,/*retain 赋初值0给_seed_*/
mu=0;
sigma=0;
do _i_=1 to 1000;
normal1=mu+sigma*rannor(_seed_);
output;
end;
drop _seed_ _i_ mu sigma;
proc print;
run;
```

随机数函数自变量 seed

随机数函数使用一个自变量 seed 来选择产生随机数的初始种子值，由这个值开始产生随机数流。

SAS CALL 子程序

用随机数子程序可以更好地控制种子流和随机数流。
同时产生几个随机数流时，用 CALL 子程序比用随机函数的效果更好。因为，用随机数函数同时创建的多个随机数变量都属于同一个随机数流。

```
data;
retain seed3 1234567 seed4 7654321 ;/*两个数据流*/
```

```
do I=1 to 10;
call ranuni(seed3,x3);
call ranuni(seed4,x4);
output;
end;
proc print;
run;
```

产生两个独立的种子流和随机数流

第四章：访问外部数据文件

导入特定文件夹下的 EXCEL 文件

```
proc import out=tb31
datafile= "D:\ResDat\table.xls"/*必须保存为97-2003才行*/或者读入的时候
excel文件后缀为xlsx，但必须和Excel对应
dbms=excel2000 replace;
range="'2#1$'";  /*导入表2.1 */
getnames=yes;
run;
```

导入特定文件夹下的 txt 文件

```
proc import out=b_share_2
datafile= "D:\ResDat\b_shares_1.txt"/*txt文件存放路径*/
dbms=dlm  replace;/*默认空格分割*/
getnames=yes;/*读取变量名*/
datarow=2;/**/
run;
```

第五章：数据步读入原始数据

使用 INPUT 语句读入原始数据

```
data scores;
infile datalines truncover;
input name $ 1-10  sex $11  age 12-15  ;
datalines;
tanbing   F 12
bingtan   M 13
;/*输入数据的时候一定要把位置定好，比如这个12-15存放的是age*/
run;
```

```
proc print;
run;
/*例中，规定记录行的第1到10列为变量NAME的输入值，11列为变量SEX的输入值，12到15列为变量AGE的输入值。NAME和SEX为字符型变量。
*/
```

“:”格式修饰符

‘:’当原始数据是以空格为分隔符时，要想对变量值长度不一致的变量规定统一长度就必须用到该格式修饰符。

```
data;
input Univ : $12. Plc$  Zip ;
cards;
MIT Boston 100023
TsinghuaUniv  Beijing  100084
;
run;
/*当原始数据是以空格为分隔符时，要想对变量值长度不一致的变量规定统一长度就必须用到该格式修饰符":"*/
/*第一个观测变量Univ的值为MIT(只读3个字符，因遇到空格而结束)第二个观测值为TsinghuaUniv(得到先前定义的变量长度12)。
如果只对变量Univ规定长度，而不加格式修饰符(:)，在读入第一条记录时就会出错，如果不对变量Univ规定长度，
读入第二条记录时就会只读入Tsinghua，而不是预先要读入的TsinghuaUniv，
这是因为SAS默认的字符变量的存储长度就是8个字节。*/
proc print;
run;
```

“&”分隔符

’&’，字符型输入值可能包含一个或几个空格。因为空格是列表读入方式默认的分隔符，所以，如果要读入的数据值本身包括空格时就必须用此格式符。

```
data;
input name & $12.  age;
cards;
tan bing   20
bing tan  31
i loveyou  60
;
/*因’&’有以上特性，数据之间应该用两个以上的空格隔开。*/
/*第一个观测name 中Jiang Zhu (包含一个空格)，第二个观测为Annie Zheng （包含一个空格)。分隔符为两个空格。*/
run;
proc print;
run;
```

“~”格式修饰符

’~’，规定读入字符值时保留引号。此选项只在 INFILE 语句中与选项 DSD 一起使用时才有效。

格式化方式输入

格式化输入方式是 INPUT 语句读取非标准数据的唯一方法。即在变量名后面规定输入格式。这种输入方式不仅给出了该输入数据所对应的类型，而且给出了输入数据所在列的长度。

命名方式输入

如果数据行中含有变量的名字，后面跟着等号和变量的值，读取数据时应该使用命名输入方式。

```
data a;
input  date  yymmdd10. fullshr  stkcd=$  lstknm=$  ;
cards;
2001-01-18  1486553100  stkcd=600001 lstknm=邯郸钢铁
;
run;
proc print;
run;
```

第六章：数据步文件管理

data 语句

data _null_;　　/*特殊名，不创建 SAS 数据集，用于输出 */

```
data new (drop=var1);                    /*去掉数据集new中变量var1*/
data new (keep=_numeric_);               /*保留数据集new中所有数值变量*/
data new (label=’股本变动历史’);   /*规定数据集new标签名为”股本变动历史”
*/
data new (rename=(var1=u var2=v)); /*将数据集new中变量var1和var2更名为
u和v*/
```

特殊数据集名

data

data _data_;　　/*等价于语句 data;*/，系统自动为数据集赋名：data1, data2,…,

null

```
/* _null_
 一般和PUT语句一起用
由PUT输出结果，只输出到LOG窗口，不会产生SAS数据集*/
data _null_;
x=exp(5);
y=log(10);
put x= y=;
run;
```

last

_last_是 SAS 系统的一个自动变量，取值为最新创建的 SAS 数据集名。

例 6.7 查看最新创建的 SAS 数据集。

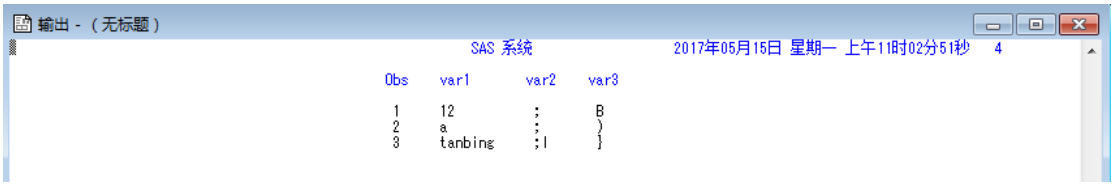
data a;

set _last_;

run;

数据行中含有分号时，必须用 CARDS 4 或 DATALINES4 语句。

```
data;
input var1$ var2$ var3 $;
cards4;
12   ;   B
a    ;   )
tanbing ;l }
;;;;/*这里三个观测变量，所有应该有三个分号表示结束标志，
因为cards4可以将分号读进来，另外一个分号表示cards4结束标志*/
run;
proc print;
run;
```



```
put name 1-8  @12 sex;/*第 1 到 8 位是 name 的显示位置，12 之后是 sex 的显示位置
*/
put a 0-10 .2 b 10-26 .3;/*输出显示的起始列位置和结束的起始列位置*/
```

put 格式化输出

```
data a;
input name & $20. bldg $ room;      /*变量name有空格,20表示name的长度*/
put name @20 (bldg room) ($2."-", 3.);
/*从第20列开始输出第二、三个变量，用‘-’连接变量，后面的2和3分别表示bldg和
room的长度*/
cards;
tan bing  Jb  125
bing bing tan  Cq  233
;
run;
```

```
164  data a;
165  input name & $20. bldg $ room;      /*变量name有空格,20表示name的长度*/
166  put name @20 (bldg room) ($2."-", 3.);
167  /*从第20列开始输出第二、三个变量，用‘-’连接变量，后面的2和3分别表示bldg和room的长度*/
168  cards;

tan bing            Jb-125
bing bing tan       Cq-233
NOTE: 数据集 WORK.A 有 2 个观测和 3 个变量。
NOTE: “DATA 语句”所用时间（总处理时间）：
      实际时间          0.06 秒
      CPU 时间          0.03 秒


171  ;
172  run;
```

by 语句

数据步中，BY 语句规定分组变量。用于控制 SET,MERGE,UPDATE 或 MODIFY 语句的操作。

```
proc sort data=a;/*对a进行排序*/
by year month;
```

set 语句,相当于是列合并

SET 语句从一个或多个已存在的 SAS 数据集中读取观测值，并将这些观测组合在一个数据集中。

```
/*相同变量的数据集连接。*/
data a;
input name $ sex $ score;
cards;
tan M 100
bing F 100
hang F 98
;
run;
data b;
```

```
input name $ sex $ score;
cards;
tan M 100
bing F 10
hang M 50
;
run;
data c;
set work.a work.b;
run;
proc print data=c;
run;
```

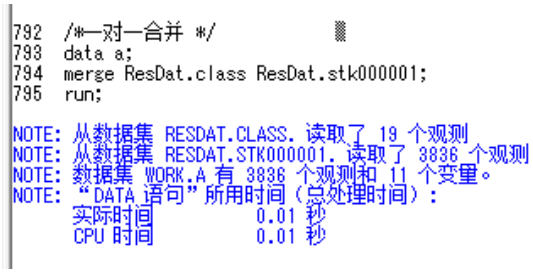
SAS 系统　　2017年05月27日 星期六 上午08时28分53秒 1337

Obs	name	sex	score
1	tan	M	100
2	bing	F	100
3	hang	F	98
4	tan	M	100
5	bing	F	10
6	hang	M	50

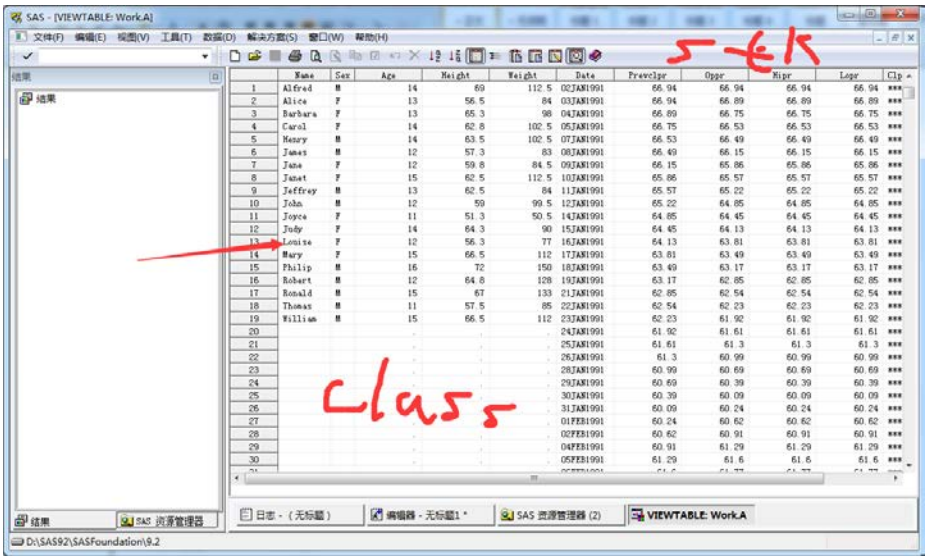
merge 语句，相当于是行合并

MERGE 语句将多个数据集中的观测合并为新数据集中的一个观测。SAS 系统合并观测的方式依赖于 BY 语句的使用。

```
/*一对一合并 */
data a;
merge ResDat.class ResDat.stk000001;
run;
/*例中，一对一合并时，不需要BY语句。 */
```

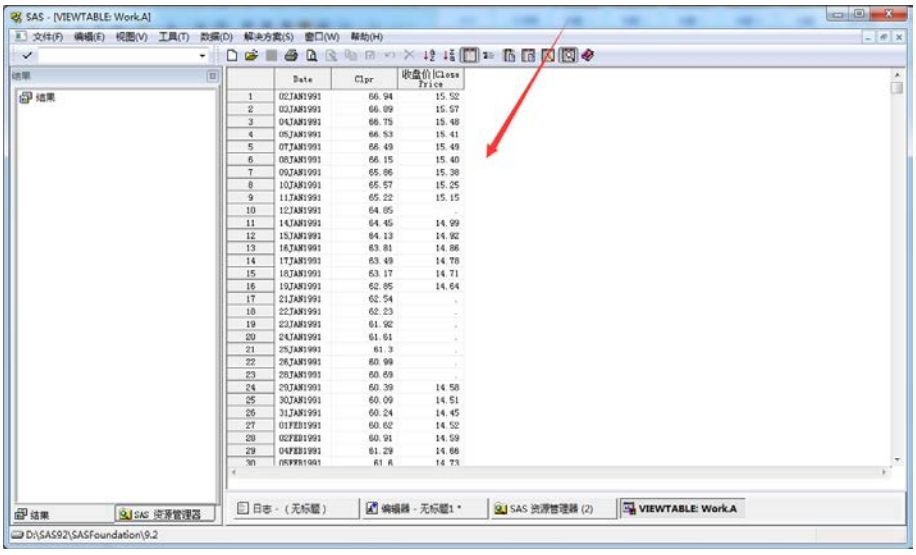


```
792  /*一对一合并 */
793  data a;
794  merge ResDat.class ResDat.stk000001;
795  run;

NOTE: 从数据集 RESDAT.CLASS. 读取了 19 个观测
NOTE: 从数据集 RESDAT.STK000001. 读取了 3836 个观测
NOTE: 数据集 WORK.A 有 3836 个观测和 11 个变量。
NOTE: “DATA 语句”所用时间（总处理时间）：
      实际时间          0.01 秒
      CPU 时间          0.01 秒
```




```
/*匹配合并 */
data a;
merge ResDat.stk000001 (keep=date clpr rename=(clpr=clpr000001))
ResDat.stk000002 (keep=date clpr rename=(clpr=clpr000002));
by date;
run;
/*例中，匹配合并必须有BY语句。 */
```

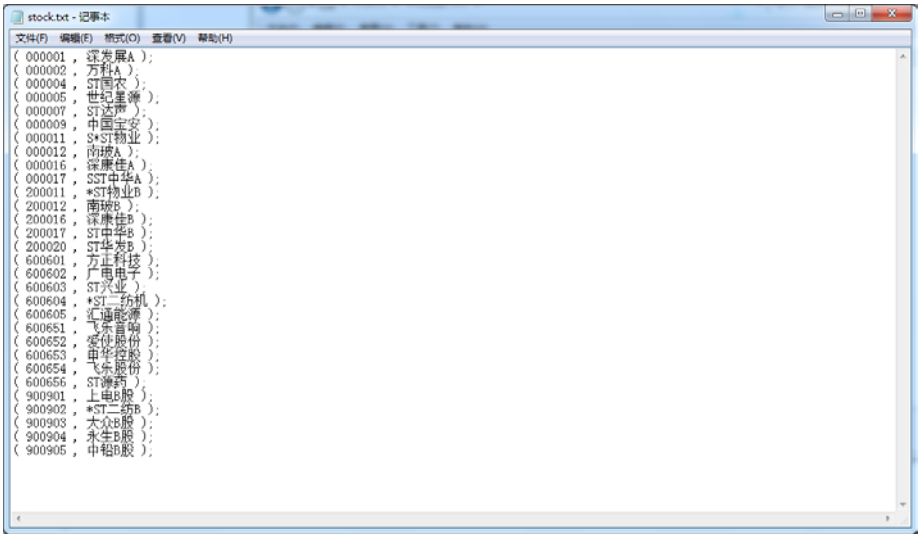
```
796  /*匹配合并 */
797  data a;
798  merge ResDat.stk000001 (keep=date clpr rename=(clpr=clpr000001))
799  ResDat.stk000002 (keep=date clpr rename=(clpr=clpr000002));
800  by date;
801  run;

NOTE: 从数据集 RESDAT.STK000001. 读取了 3836 个观测
NOTE: 从数据集 RESDAT.STK000002. 读取了 4103 个观测
NOTE: 数据集 WORK.A 有 4170 个观测和 3 个变量。
NOTE: “DATA 语句”所用时间（总处理时间）：
      实际时间          0.01 秒
      CPU 时间          0.01 秒


802  /*例中，匹配合并必须有BY语句。 */
```



file 语句

FILE 语句用于规定将要输出的外部文件。FILE 语句一般要与 PUT 语句配合使用。

```
libname resdat 'd:\resdat';
run;
/* 输出规定格式的外部文本文件。*/
data;
set ResDat.lstkinfo;
a='(';
b=',';
c=');';
file "D:\stock.txt";
put a $ stkcd $ b $ lstknm $  c $ ;
run;
/*例中程序产生股票宏文本文档*/
```



infile 语句

INFILE 语句用来定义一个外部数据文件，文件中的数据用 INPUT 语句读取。外部文件可以是已存在的磁盘文件，也可以是从键盘上输入的数据行。

例 6.44 DELIMITER=选项应用

```
data a;
infile cards delimiter=',';/*从 cards 读取数据，用，分割*/
input x y z;
cards;
3,6,9
1,3,5
8,8,8
```

;

例中，要输入的数据用逗号分隔，创建 SAS 数据集时用选项 DELIMITER=’,’。

第七章：数据步修改与选择观测

```
data a;
length lstknm $12; /* 规定变量lstknm的长度为12 */
lstknm='深发展';
lstknm ='大秦铁路';
proc print;
run;
```

SAS 系统 2017

Obs	lstknm
1	大秦铁路

累加语句

```
/*累加语句等于使用SUM函数和一个RETAIN语句*/
data a (keep=name height s_h);
set ResDat.class ;
s_h+height;
proc print;
run;
```

SAS 系统 2017年06

Obs	Name	Height	s_h
1	Alfred	69.0	69.0
2	Alice	56.5	125.5
3	Barbara	65.3	190.8
4	Carol	62.8	253.6
5	Henry	63.5	317.1
6	James	57.3	374.4
7	Jane	59.8	434.2
8	Janet	62.5	496.7
9	Jeffrey	62.5	559.2
10	John	59.0	618.2
11	Joyce	51.3	669.5
12	Judy	64.3	733.8
13	Louise	56.3	790.1
14	Mary	66.5	856.6
15	Philip	72.0	928.6
16	Robert	64.8	993.4
17	Ronald	67.0	1060.4
18	Thomas	57.5	1117.9
19	William	66.5	1184.4

delete 语句

DELETE 语句停止处理当前观测，该观测值**不被读入到创建的数据集**，SAS 系统返回到 DATA 步的开头处理其他观测。

stop 语句

STOP 语句停止处理 DATA 步。**当遇到 STOP 语句时，正被处理的那个观测没有添加到 SAS 数据集上**。

```
/*停止处理DATA步*/
data a;
set resdat.lstkinfo;
if _n_=5 then stop;/*本来有30个观测值，然后到第五个就停止，只读取了四个观测值*/
proc print;  /*OUTPUT窗口照常打印数据集列表 */
run;
/*例中，数据集A从数据集resdat.lstkinfo中读取了4条观测，
因为当指针标识_N_=5时，遇到STOP语句，正被处理的那条观测没有添加到数据集A中。*/
```

abort 语句

ABORT 语句中止执行当前的 DATA 步，继续执行下一个 DATA 或 PROC 步。执行 ABORT 语句时，创建 ABORT 语句执行前已处理观测的数据集。但是，当新创建数据集和已存在的 SAS 数据集同名时，不能覆盖已存在的数据集。

where 语句的性质

- WHERE 语句读入数据集之前选择观测。
- WHERE 语句不是可执行语句，它起不到 IF-THEN 语句的作用。
- 能用 WHERE 语句的地方一定可以用 IF 语句来代替，反之则不行。不过，一旦 WHERE 语句有效，就一定要用它，因为这样的程序效率高。

```
/*例7.22*/
data a;
set ResDat.Idx000001;
where _n_<100; /*错误语句，必须用if */
run;
data a;
set ResDat.Idx000001;
if _n_<100;
run;
/*例中，不能用WHERE语句控制SAS的自动变量。*/
```

output 语句

OUTPUT 语句输出当前的观测到被创建的数据集中。

```
/*一个DATA步创建多个数据集 */
data A B;
set ResDat.lstkinfo;
if Stktype='A' then output A;
else if Stktype='B' then output B;
run;
proc print data=A;
run;
/*例中，一个DATA步创建两个数据集。
数据集A包含变量Stktype=’A’ ’ 的所有观测；
数据集B包含变量Stktype=’ B的所有观测。*/
```

missing 语句

MISSING 语句规定缺失值的符号。
读入含有缺失值的数据源时，必须用 MISSING 语句，否则可能产生读入错误。

```
/*规定缺失值字符 */
data period_a;
missing X I;
input Id $4. Foodpr1 Foodpr2 Foodpr3 Coffeem1 Coffeem2;
datalines;
1001 115 45 65 I 78
1002 86 27 55 72 86
1004 93 52 X 76 88
1015 73 35 43 112 108
1027 101 127 39 76 79
;
run;
proc print;
run;
/*例中，MISSING语句规定用字符X和I表示缺失值。
如果不用MISSING语句，当读入数值变量的缺失值时（这里为X和I），就要产生错误。 */
```

每个观测包含3个数据行　　2017年06月05日 星期

Obs	Id	Foodpr1	Foodpr2	Foodpr3	Coffeem1	Coffeem2
1	1001	115	45	65	I	78
2	1002	86	27	55	72	86
3	1004	93	52	X	76	88
4	1015	73	35	43	112	108
5	1027	101	127	39	76	79

list 语句

- LIST 语句在 SAS 日志窗口上列出正被加工处理观测的输入数据行。

```
/*用INPUT语句读入可疑数据行时使用LIST语句*/
data a;
input x y;
if x<0 then list;
cards;
2 6
4 2
-1 2
-4 6
;
run;
proc print;
run;
```

```
716  /*用 INPUT语句读入可疑数据行时使用LIST语句*/
717  data a;
718  input x y;
719  if x<0 then list;
720  cards;

RULE:      ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+----0
723        -1 2
724        -4 6
NOTE: 数据集 WORK.A 有 4 个观测和 2 个变量。
NOTE: “DATA 语句”所用时间（总处理时间）：
      实际时间          0.01 秒
      CPU 时间          0.01 秒


725  ;
726  run;
```

如果 list 改为 output,那么在日志窗口中就没有输出哦!

第八章：数据步循环与转移控制

do 语句

- DO 语句必须由一个 END 语句来结束。

```
/*IF/THEN语句用中DO.*/
data a;
set ResDat.class;
if age>14 then do;
h_cm=30.5*height/12;
put name= sex= age=  h_cm=;
end;
run;
proc print data=a(keep=name sex age h_cm);
run;
```

Obs	Name	Sex	Age	h_cm
1	Alfred	M	14	.
2	Alice	F	13	.
3	Barbara	F	13	.
4	Carol	F	14	.
5	Henry	M	14	.
6	James	M	12	.
7	Jane	F	12	.
8	Janet	F	15	158.854
9	Jeffrey	M	13	.
10	John	M	12	.
11	Joyce	F	11	.
12	Judy	F	14	.
13	Louise	F	12	.
14	Mary	F	15	169.021
15	Philip	M	16	183.000
16	Robert	M	12	.
17	Ronald	M	15	170.292
18	Thomas	M	11	.
19	William	M	15	169.021

循环 do 语句

DO 组中的语句需要被重复执行时要用到循环 DO 语句。

```
/*用GO TO语句跳出循环体。*/
data a;
input x y;
if x>y then goto skip;  /*skip是循环体外的语句标号*/
y=log(y-x);
yy=y-20;
skip: if y<0 then do;
z=log(x-y);
put x= y= z= ;
end;
cards;
2 6
5 3
5 -1
;
run;
```

```
/*产生指定观测值个数的数据集。*/
data a;
do n=1 to 100 ;
output;
end;
run;
/*例中，产生一个含有100个观测值的数据集。 */
```

例 8.7 自然数求和。

```
data a;
t=0;
do n=1 to 100 ;
t=t+n;
output;
end;
run;
```

例中，求 1 到 100 的自然数之和。

do while 语句

DO WHILE 语句使得当条件成立时重复地执行 DO 组中的语句。

```
/*当N值小于5时重复执行DO组中的语句。*/
data a;
n=0 ;
do while(n lt 5);/*n小于5*/
put n=;
n+1;
end;
run;
```

do until 语句

有条件地执行 DO 组中的语句。

```
do until(n>=5);
```

select 语句

SELECT 语句选择执行 SAS 语句。

```
select;
when (x=2)  put 'two';  /*正确用法*/
end;
run;
```

if 语句

goto 语句

```
/*应用举例。*/
data a;
input x @@;
if 1<=x<=5 then goto ok;
```

```
put x;
count+1;
ok: sumx+x;/*这里没有return,相当于要累加全部的数*/
cards;
1 2 7 2 12 24 22
;
/*例中，COUNT仅对大于5的数计数，SUMX+X对每个观测都执行。 */
proc print;
run;
```

```
/*用DO-END语句替代GOTO语句。*/
data a;
input x @@;
if x<1 or x>5 then do;
put x;
count+1;
end;
sum+x;
cards;
1 2 7 2 12 24 22
;
proc print;
run;
```

Obs	x	count	sumx
1	1	0	1
2	2	0	3
3	7	1	10
4	2	1	12
5	12	2	24
6	24	3	48
7	22	4	70

```
data a;
input x @@;
if 1<=x<=5 then goto ok;
put x; count+1;
return;
ok: sumx+x;/*只累加小于5的数*/
cards;
1 2 7 2 12 24 22
;
run;
proc print;
run;
```

Obs	x	count	sumx
1	1	0	1
2	2	0	3
3	7	1	3
4	2	1	5
5	12	2	5
6	24	3	5
7	22	4	5

return 语句

RETURN 语句告诉 SAS 系统在 DATA 步当前位置上停止执行语句，返回到一个预定位置上继续执行。

```
/*让SAS系统返回到DATA步开头。*/
data survey;
input x y z;
if x=y then return;
x=y+z; a=x**2;
cards;
1 2 3
3 3 4
5 6 7
8 8 9
;
run;
proc print;
run;
/*例中，当X=Y时，RETURN语句被执行。SAS系统添加这个观测到数据集SURVEY，
并返回到 DATA 步的开头。IF 语句后面的二个赋值语句没有被执行*/
```

Obs	x	y	z	a
1	5	2	3	25
2	3	3	4	.
3	13	6	7	169
4	8	8	9	.

leave 语句

LEAVE 语句停止当前 DO 组循环或 SELECT 组的处理过程，并继续执行 DO 组或 SELECT 组后面的语句。

第九章：数据步变量与变量属性控制

array 语句

- ARRAY 语句用于定义数组。数组通常由一组变量构成。

建立临时数组元素列表

temporary;
使用临时数组元素列表可以少占用内存，加快执行时间。

```
array t(3) (5,10,15); 则会产生新变量t1,t2,t3;*/
array t(3) _temporary_ (5,10,15); /*若用临时数组，则不会产生新变量*/
```

引用显式下标数组元素

```
put test{4}=  test{6}=;
```

循环 do 组中引用

```
do i=1 to dim(day);
day(i)=day(i)+10;
end;
```

do while 和 do until 组引用

```
/* DO WHILE语句用法。*/
data test;
input x1-x5 y;
array t(5) x1-x5;/*一维5数组*/
i=1;
do while (t(i)<y);
put t(i)= y=;
i=i+1;
end;
cards;
1 2 3 4 5 3
0 2 4 6 8 6
;
run;
```

informat 语句

INFORMAT 语句把输入格式与变量联系起来。

```
/*规定临时的缺省输入格式。*/
```

```
data a;
informat default=3.1 default=$char4.;
input x1-x5 name $;
put x1-x5 name;
cards;
11 22 33 44 100 johnny
;
run;
proc print;
run;
```

SAS 系统 2017年

Obs	x1	x2	x3	x4	x5	name
1	1.1	2.2	3.3	4.4	10	john

format 语句

```
data ;
format  w $ 3. y 10.3  default=$8. default=8.2;/*设置输出格式*/
w='good morning.';
y=12.1;
x='good morning.';
z=12.1;
put w/y/x/z;
run;
```

```
606  data ;
607  format  w $ 3. y 10.3  default=$8. default=8.2;
608  w='good morning.';
609  y=12.1;
610  x='good morning.';
611  z=12.1;
612  put w/y/x/z;
613  run;

goo
12.100
good mor
12.10
NOTE: 数据集 WORK.DATA6 有 1 个观测和 4 个变量。
NOTE: “DATA 语句”所用时间（总处理时间）：
      实际时间          0.01 秒
      CPU 时间          0.01 秒
```

字符变量长度控制

```
length name $5;/*name的长度为5*/
```

```
data a;
lstknm='深发展';
data b;
lstknm='大秦铁路';
data c;
```

```
length lstknm $12 ; /*length语句放在set语句之前 */
set a b;
proc print;
run;
```

SAS 系统

Obs	lstknm
1	深发展
2	大秦铁路

label 语句

LABEL语句用于为变量加标签。变量标签是对变量的进一步说明，看到标签就能理解变量的意思。

drop 语句

DROP 语句规定输出数据集中要删除的变量，它对 DATA 步正在创建的所有 SAS 数据集都适用。

keep 语句

KEEP语句规定输出数据中要保留的变量

rename 语句

```
data a(keep=date open low high close);
set ResDat.Idx000001;
rename oppr=open lopr=low hipr=high clpr=close;
proc print data=a(obs=3);
run;
```

retain 语句

RETAIN 语句来规定单个变量，变量列表，或数组元素的初始值。

第十章：过程步通用语句

VAR 语句

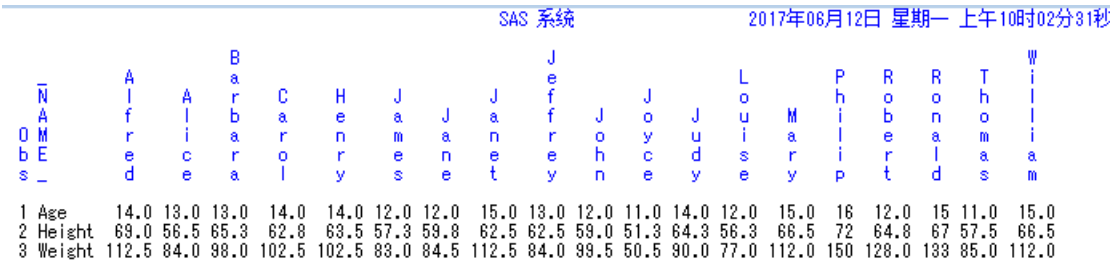
VAR 语句规定要分析的变量名。

```
var  weight  height;
```

ID 语句

ID 语句规定用于识别观测的变量。

```
id name;
```



SAS 系统　　2017年06月12日 星期一 上午10时02分31秒

Obs	_NAME_	Alfred	Alice	Barbara	Carol	Henry	James	Jane	Janet	Jeffrey	John	Joyce	Judy	Louise	Mary	Philip	Robert	Ronald	Thomas	William
1	Age	14.0	13.0	13.0	14.0	14.0	12.0	12.0	15.0	13.0	12.0	11.0	14.0	12.0	15.0	16	12.0	15	11.0	15.0
2	Height	69.0	56.5	65.3	62.8	63.5	57.3	59.8	62.5	62.5	59.0	51.3	64.3	56.3	66.5	72	64.8	67	57.5	66.5
3	Weight	112.5	84.0	98.0	102.5	102.5	83.0	84.5	112.5	84.0	99.5	50.5	90.0	77.0	112.0	150	128.0	133	85.0	112.0

WHERE 语句

WHERE 语句选择 SAS 数据集中的观测。

```
where sex='M';
where sex^='M'; /*删除观测值*/
```

CLASS 语句

CLASS 语句规定分类变量，SAS过程对分类变量的不同值分别进行分析处理。

```
var weight height;
class sex;
```