1. **合并数据集**
2. **用 SET 语句拼接合并数据集**

用SET语句可以把两个数据集拼接合并在一起，适用于两个数据集具有相同的变量。

基本形式为：

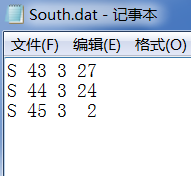
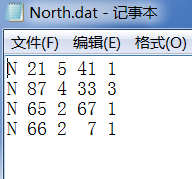
data 新数据集名;

set 旧数据集1 旧数据集2;

注：（1）按原来顺序合并成新数据集（数据集1在上，2在下）；

（2）若一个数据集包含了另一个数据集没有的变量，那么合并后，该变量下将会出现缺失值。

**例1** 路径“C:\MyRawData\”下有关于娱乐公园南北门游客的数据South.dat 和 North.dat，都包括变量 Entrance、PassNumber、PartySize、Age，后者多了一列 Lot（停车）：

先分别读入数据存为数据集再合并成一个新数据集，并创建了新变量，AmountPaid.

代码：

**data** southentrance;

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

input Entrance $ PassNumber PartySize Age;

**proc** **print** data = southentrance;

title 'South Entrance Data';

**run**;

**data** northentrance;

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

input Entrance $ PassNumber PartySize Age Lot;

**proc** **print** data = northentrance;

title 'North Entrance Data';

**run**;

**data** bothentrance;

set southentrance northentrance;

if Age =**.** then AmountPaid =**.**;

else if Age < **3** then AmountPaid = **0**;

else if Age < **65** then AmountPaid = **35**;

else AmountPaid = **27**;

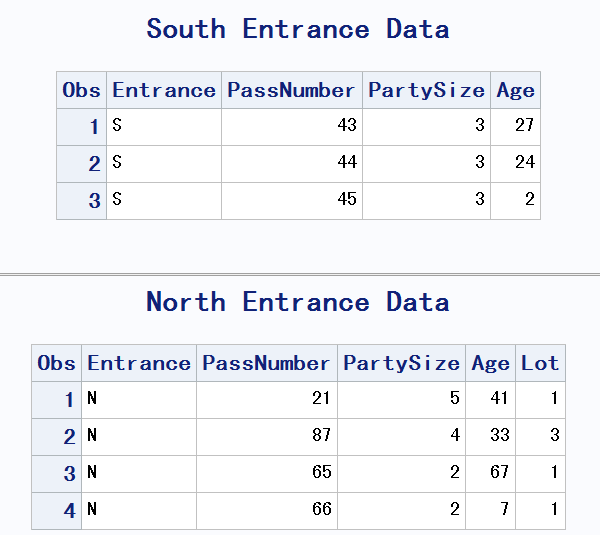
**run**;

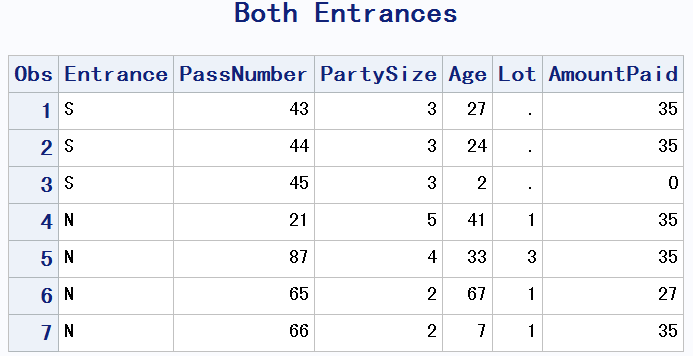
**proc** **print** data = bothentrance;

title 'Both Entrances';

**run**;

运行结果：





注意：南门数据中缺少Lot数据，堆叠合并后的观测值为缺失值。

1. **用SET+BY语句排序拼接合并数据集**

前面是保持数据集原有顺序直接拼接合并，根据需要新数据集可以作排序处理。但这样效率较低，更好的方法是先排好序再合并。

基本形式：

data 新数据集名;

set 旧数据集1 旧数据集2;

by 变量1 变量2…;

注：旧数据集必须是事先排好序的。

**例2** 同样是 **例1** 的数据，对 PassNumber 做排序拼接合并，注意南门数据已经按 PassNumber 排序。

代码：

**data** southentrance;

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

input Entrance $ PassNumber PartySize Age;

**proc** **print** data = southentrance;

title 'South Entrance Data';

**run**;

**data** northentrance;

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

input Entrance $ PassNumber PartySize Age Lot;

**proc** **sort** data = northentrance;

by PassNumber;

**proc** **print** data = northentrance;

title 'North Entrance Data';

**run**;

**data** sortbothentrance;

set northentrance southentrance;

by PassNumber;

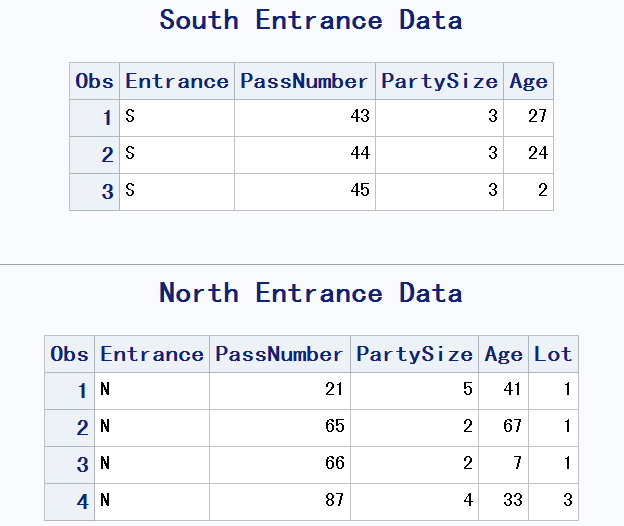
**run**;

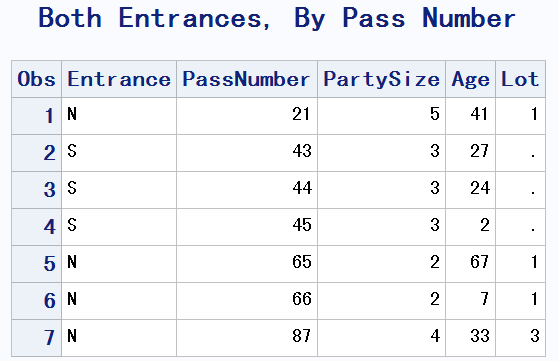
**proc** **print** data = sortbothentrance;

title 'Both Entrances, By Pass Number';

**run**;

运行结果：





1. **一对一匹配合并数据集**

经常会遇到合并两个数据集，它们有共同的变量（其不同取值都是各出现一次），用 MERGE语句可以将两个数据集按共同变量进行一对一匹配合并。注意：要求两数据集事先已按共同变量排序。

基本形式：

DATA new-data-set;

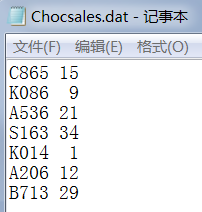
MERGE data-set-1 data-set-2 …;

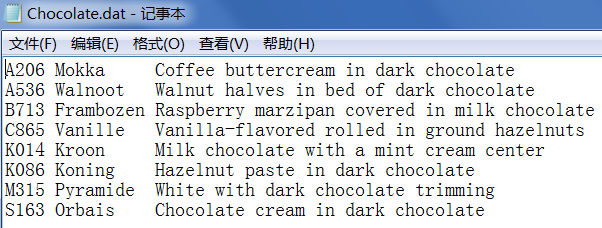
BY variable-list;

注：（1）BY 语句指定共同变量；

（2）若两个数据集有重叠的变量（除了BY指定的共同变量），第 2 个数据集中的变量将覆盖第 1 个数据集中的相同变量。

**例3** 路径“C:\MyRawData\”下有关于某巧克力店的数据，chocsales.dat记录了所卖的巧克力代码、数量；chocolate.dat记录了巧克力代码、所代表的类型、描述：





读入数据，按共同变量“巧克力代码”匹配合并数据集。

代码：

**data** descriptions;

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

input CodeNum $ **1**-**4** Name $ **6**-**14** Description $ **15**-**60**;

**run**;

**data** sales;

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

input CodeNum $ **1**-**4** PiecesSold **6**-**7**;

**proc** **sort** data = sales;

by CodeNum;

**run**;

**data** chocolates;

merge sales descriptions;

by CodeNum;

**run**;

**proc** **print** data = chocolates;

title "Today's Chocolate Sales";

**run**;

运行结果：

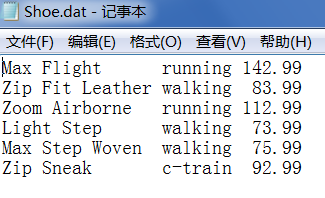
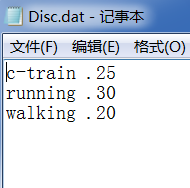


注意：数据 1 中没有代码为“M315”的销售记录，合并后的观测值显示缺失值。

1. **一对多匹配合并数据集**

也是两个包含共同变量的数据集要匹配合并，不同在于：该共同变量的同一取值在一个数据集中出现 1 次，在另一数据集中出现多次。此时就是“一对多匹配合并”。语法同“一对一匹配合并”完全相同。

**例4** 路径“C:\MyRawData\”下有关于鞋子打折的数据：Shoe.dat是关于鞋子的风格、类型、价格；Disc.dat 是关于每个类型鞋子的折扣（训练鞋、跑步鞋、走路鞋的折扣各不同）：

读入数据，按鞋子的类型合并数据集。

代码：

**data** regular;

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

input Style $ **1**-**15** ExerciseType $ RegularPrice;

**run**;

**proc** **sort** data = regular;

by ExerciseType;

**run**;

**data** discount;

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

input ExerciseType $ Adjustment;

**run**;

/\* Perform many-to-one match merge; \*/

**data** prices;

merge regular discount;

by ExerciseType;

NewPrice = ROUND(RegularPrice - (RegularPrice \* Adjustment),**.01**);

**run**;

**proc** **print** data = prices;

title 'Price List for May';

**run**;

运行结果：



1. **在数据步中用 IN = 选项跟踪选择观测值**

在 DATA 步中，IN选项可以被用在 SET, MERGE, 或者 UPDATE 语句（更多是在 MERGE 语句），接在要“追踪选择的数据集”后面。

结合下面的示例，其作用是给原数据集（customer）的每条观测都定义一个临时标签变量（只存在数据步期间不带入数据集，取值 0 或 1）用来指示“BY变量（CustomerNumber）”是否存在于“跟踪选择数据集（orders）”中，若存在，该条观测的临时标签变量 Recent 取值为 1，否则取值为 0. 然后，将满足条件 “Recent = 0”的观测，即 CustomerNumber 号未包含在（orders）中的观测被选出来创建新数据集（noorders）。

示例：

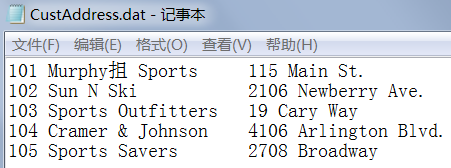
DATA noorders;

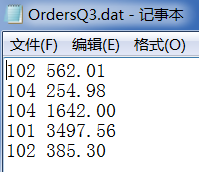
MERGE customer orders (IN = Recent);

BY CustomerNumber;

IF Recent = 0;

**例5** 运动品厂商有两份数据，一是包括所有客户的数据（C:\MyRawData\CustAddress.dat），变量包括客户编号、姓名、地址；一是包括了第三季度订单的数据（C:\MyRawData\OrdersQ3.dat），变量包括客户编号、总价格：





现在想要了解哪些客户在第三季度没有任何订单，即可以用in=option选项。

代码：

**data** customer;

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

input CustomerNumber Name $ **5**-**21** Address $ **23**-**42**;

**data** orders;

infile 'c:\MyRawData\OrdersQ3.dat';

input CustomerNumber Total;

**proc** **sort** data = orders;

by CustomerNumber;

**run**;

/\* Combine the data sets using the IN= option; \*/

**data** noorders;

merge customer orders (IN = Recent);

by CustomerNumber;

if Recent = **0**;

**run**;

**proc** **print** data = noorders;

title 'Customers with No Orders in the Third Quarter';

**run**;

运行结果：

