**08. 日期时间数据及数据格式**

**（一）SAS日期和时间数据**

SAS将所有的日期转化成一个以1960年1月1日为起点的数值（日期数值）。比如：

January 1, 1960 0

January 1, 1959 -365

January 1, 1961 366

January 1, 2020 21915

SAS处理日期数据包括：读取日期格式的数据，使用日期函数处理日期，按指定格式输出日期。

1. **读取日期格式的数据**
2. 即按格式读入数据（formatted input），见系列03。比如，

input BirthDate MMDDYY10.;

1. 设定默认的百年

07/04/76这样的数据可能是1976，也可能是2076、1776。因此需要YEARCUTOFF = 来指定一个一百年的第一年，默认的是1920年。

例如，下面的语句就是告诉SAS将一个两位年份的日期解释为1960年到2049年之间：

OPTIONS YEARCUTOFF = 1950;

1. SAS日期的存取

日期一旦被以SAS以日期格式读取之后，就将此数据像其他数值数据一样用在表达式中（日期数值）。

例如，像为图书馆的书设定21天的还书日期，只需要在借书日期上加上21：

DueDate = CheckDate + 21;

通过引号和字母D，可以将一个日期当作常数来使用，下面的代码创建了一个EarthDay05的日期变量，其值等于April 22, 2005

EarthDay05='22APR2005'D;

1. **使用日期函数处理日期**

SAS提供的日期函数使得处理日期的操作大大简便。

例如，today( ) 返回今天的日期；

计算一本书应归还的剩余期限，就可以用：

DaysOverDue = DateDue - today();

用出生日期Birthdate计算年龄，可以用：

CurrentAge = INT (YRDIF (BirthDate, TODAY(), 'AGE') );

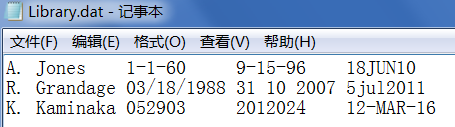
1. **按按指定格式输出日期**

按指定格式输出日期时，还需要将日期数值转换成日期。

例如，下面的FORMAT语句告诉SAS用WORDATE18.格式打印日期变量BirthDate。

format BirthDate WORDATE18.;

**例1** 图书馆有借书卡数据，Dates.dat，包含持卡人姓名、出生日期、卡办理日期，到期日：



读取数据，计算剩余期限（DaysOverDue）；计算持卡人年龄（CurrentAge）；接着用IF语句来判断一个卡是否为新卡，在2012年1月1日之后办理的，为新卡。

代码：

**data** librarycards;

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

input Name $11. + **1** BirthDate MMDDYY10. +**1** IssueDate ANYDTDTE10. DueDate DATE11.;

DaysOverDue = DueDate - TODAY();

CurrentAge = INT(YRDIF(BirthDate, TODAY(), 'AGE'));

if IssueDate > **'01JAN2012'D** then NewCard = 'yes';

**run**;

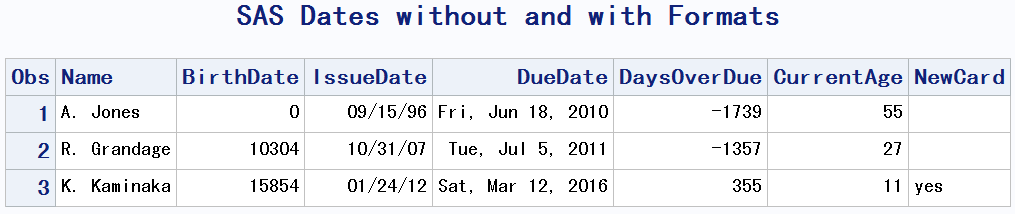
**proc** **print** data = librarycards;

format Issuedate MMDDYY8. DueDate WEEKDATE17.;

title 'SAS Dates without and with Formats';

**run**;

运行结果：



程序说明：

1. BirthDate变量没有指定输出格式，默认输出日期数值；
2. “ANYDTDTE10.” 以各种格式（非单一格式）读入日期；
3. 歧义日期值，例如01-02-03，可以用DATESTYLE = MDY来设置（默认是MDY）。
4. **时间数据**
5. SAS的时间数值（time），表示从当天零点开始到当前时间的秒数，大小介于0和 86400之间。

hms(h,m,s) ——由小时h、分钟m、秒s生成SAS时间值；

t1 = ‘8:45’T;——时间常数值赋值；

hour(time);——返回时间数值的小时数；minute, second类似；

1. SAS还有日期时间数值（datetime），表示从1960年1月1日零点到当前时刻的秒数。datetime 对应的几种格式：

30May2000:10:03:17.2 DATETIME20.

30May00 10:03:17.2 DATETIME18.

30May2000/10:03 DATETIME15.

conn = '01Feb2011:8:45'DT;——日期时间常数值赋值；

dhms(d,h,m,s)——由SAS日期值d、小时h、分钟m、秒s生成SAS日期时间值；

datepart(dt) ——返回SAS日期时间值dt的日期部分；

intnx(interval,from,n)——返回从from开始经过n个interval间隔后的SAS日期，其中interval 可以取'YEAR'、'QTR'、'MONTH'、'WEEK'、'DAY'等；例如，

intnx('MONTH', '16Dec1997'd, 3);

结果为1998年3月1日。注意：它总是返回一个周期的开始值。

intck(interval,from,to)——返回从日期from到日期to中间经过的interval间隔的个数，其中interval取'MONTH'等。例如，

intck('YEAR', '31Dec1996'd, '1Jan1998'd);

返回两个日期的年间隔数，结果得2，尽管这两个日期之间实际只隔1年。

**（二）SAS日期时间格式、函数及示例**

**日期时间格式：**

|  |  |  |
| --- | --- | --- |
| 格式 | 说明 | 默认宽度 |
| ANYDTDTEw | 以各种格式读入（非单一格式） | 9 |
| DATEw. | ddmmmyy 或 ddmmmyyyy | 7 |
| DDMMYYw. | ddmmyy 或ddmmyyyy | 6 |
| MMDDYYw. | mmddyy 或 mmddyyyy  例：01/01/1961（10.） | 6 |
| JULIANw | Julian日期，yyddd or yyyyddd | 5 |
| EURDFDDw. | dd.mm.yy  例：01.01.61（8. 01.01.1961（10.） | 8 |
| WEEKDATEw. | day-of-week, month-name dd, yy or yyyy  例：Sunday（9.）Sunday, January 1, 1961（29.） | 29 |
| WORDDATEw. | month-name dd, yyyy  例：Jan 1, 1961（12.）January 1, 1961（18.） | 18 |
| TIMEw.d | hh:mm:ss （8.）或hh:mm:ss.ss（11.2） | 8 |
| DATETIMEw.d | ddmmmyy:hh:mm:ss  例：01JAN60:03:23（13.） 01JAN60:03:23:02.0（18.1） | 16 |

**日期时间函数：**

|  |  |
| --- | --- |
| datejul(day) | 将日期数值day转化为Julian日期值 |
| today() | 返回今天的日期数值 |
| mdy(mm,dd,yy) | 将月mm、日dd、年yy转化为日期数值 |
| day(date) | 返回日期数值的日 |
| month(date) | 返回日期数值的月（1-12） |
| year(date) | 返回日期数值的年 |
| qtr(date) | 返回日期数值的季（1-4） |
| weekday(date) | 返回日期数值是周几（1=星期日） |
| time(t) | 返回时间数值t对于的时间 |
| yrdif(date1,date2, ‘AGE’) | 计算两个日期的“年龄差”  例：a=mdy(4,13,2000); b=mdy(8,13,2012);  yrdif(a,b,'AGE'); 返回 12.3342 |

注：Julian日期：5位或7位，前2或4位是年，后3位为该年的第多少天，例如1960年2月1日是60032或1960032.

**（三）字符和数值格式**

1. **字符格式：**

|  |  |  |
| --- | --- | --- |
| 格式 | 说明 | 默认宽度 |
| $UPCASEw. | 转化为大写字母 | 8 |
| $w. | 同$CHARw. 标准格式字符（不忽略开头空格） | 1 |

1. **数值格式：**

|  |  |  |
| --- | --- | --- |
| 格式 | 说明 | 默认宽度 |
| w.d | 控制位宽，共占w位，其中d位小数 | 无 |
| BESTw. | SAS自动选择最佳数值格式（默认） | 12 |
| COMMAw.d | 逗号数值格式  例：1,200,001（9.） 1,200,001.00（12.2） | 6 |
| Ew. | 科学计数法  例：1.2E+6（E7.） | 12 |
| PERCENTw.d | 百分数  例：5.00%（PERCENT9.2） | 6 |
| DOLLARw.d | 美元金额：$开头+逗号数值格式  例：$1,200,001.00（DOLLAR13.2） | 6 |
| EUROXw.d | 欧元金额：  例：€1.200.001,00 （EUROX13.2） | 6 |

1. **自定义格式**

数据集中为了处理方便有时候用数字代表实际的变量值，比如1代表男性，2代表女性。这种代码在输出的时候不好解读，可以用proc format自定义格式，然后打印出想要的值。

基本形式为：

proc format;

value 格式名 变量值1 = '输出形式1'

变量值2 = '输出形式2'

……

变量值n = '输出形式n';

注：（1）若变量值是字符值，格式名前要加上$;

（2）格式名长度不能超过32个字节（包括$），不能以数字结尾，除了下划线不能包含其他任何特殊符号。且名字不能与已有的格式名冲突；

（3）输出形式必须放在单引号里，中间要想输出单引号，需要用两个单引号。例如，

'employee’ ’s jobtitle'——> employee’s jobtitle

示例：

'A' = 'Asia'

**1**, **3**, **5**, **7**, **9** = 'Odd'

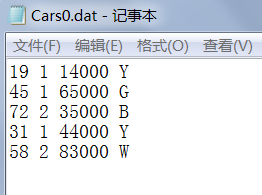
**500000** - HIGH = 'Not Affordable'

**13** -< **20** = 'Teenager'

**0** <- HIGH = 'Positive Non Zero'

OTHER = 'Bad Data'

**例2** 关于汽车公司客户的调查信息数据（C:\MyRawData\Cars0.dat）。包括客户年龄、性别（1为男性，2为女性）、每年收入、偏爱的汽车颜色：



读取数据，并使用proc format自定义格式过程，为颜色、性别和汽车创建输出格式。

代码：

**data** carsurvey;

infile 'c:\MyRawData\Cars0.dat';

input Age Sex Income Color $;

**proc** **format**;

value gender **1** = 'Male'

**2** = 'Female';

value agegroup **13** -< **20** = 'Teen'

**20** -< **65** = 'Adult'

**65** - HIGH = 'Senior';

value $col 'W' = 'Moon White'

'B' = 'Sky Blue'

'Y' = 'Sunburst Yellow'

'G' = 'Rain Cloud Gray';

\* Print data using user-defined and standard (DOLLAR8.) formats;

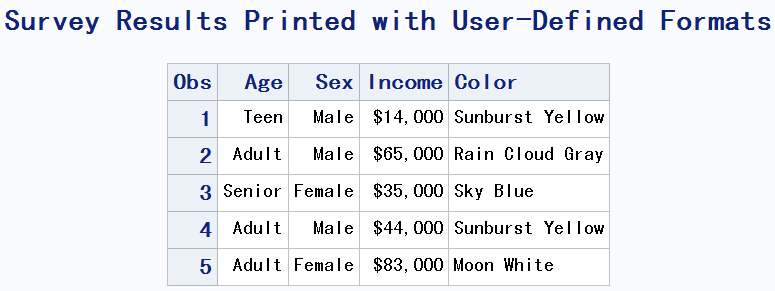
**proc** **print** data = carsurvey;

format Sex gender. Age agegroup. Color $col. Income DOLLAR8.;

title 'Survey Results Printed with User-Defined Formats';

**run**;

运行结果：



程序说明：

（1）定义格式时，格式名后面没有句点，而在设定输出格式时，格式名后面带句点（表示位宽无小数位）；

（2）该代码是定义临时格式（默认存到临时逻辑库work），也可以另指定路径定义永久格式。