**Chapter XVIII – Reading Data and Time Values**

SAS date values are based on Gregorian Calendar, which valid for dates from A.D.1582 through A.D. 20000; it make adjustments for leap years but leap seconds; but does not adjust daylight saving time

1. How SAS Store Date Value

* Storing dates and times as numeric values enables you to use dates and times in calculations much as you would use any other number.
* Eg:

SAS date value is the number of days from January 1, 1960. (50题, Q1)

|  |  |  |
| --- | --- | --- |
| **Date Expression** | **SAS Date Informat** | **SAS Date Value** |
| 02Jan00 | **DATE*w.*** | 14611 |
| 01-02-2000 | **MMDDYY*w.*** | 14611 |
| 02/01/00 | **DDMMYY*w.*** | 14611 |
| 2000/01/02 | **YYMMDD*w.*** | 14611 |

1. How SAS Store Time Value

A SAS datetime is a special value that combines both date and time information. The datetime value is stored as the number of seconds between midnight to January 1, 1960 and given time

*SAS Calculation of Date and Time Values*



1. Reading Dates and Times with Informats

Using INPUT statement with an informat after a variable to read date and time value into a variable

* Basic code:

**INPUT** <*pointer-control*> *variable informat.***;**

*pointer-control* specifies the absolute or relative position to move the pointer.

*variable* is the name of the variable being read.

*informat*. is any valid SAS informat. Note that the informat includes a final period.

* **MMDDYY*w*.** Informat

基本形式如：10/15/99

* 月日年之间必须有blanks或者delimiters分隔开。如果有特定的delimiter，那他们必须在所有field之间保持一致
* *Date Expressions and Corresponding SAS Date Informats*

|  |  |
| --- | --- |
| **Date Expression** | **SAS Date Informat** |
| 101599 | **MMDDYY6.** |
| 10/15/99 | **MMDDYY8.** |
| 10 15 99 | **MMDDYY8.** |
| 10-15-1999 | **MMDDYY10.** |

* **DATE*w*.** Informat

基本形式如：30May2000

* Can have blanks or other delimiters
* *Date Expressions and Corresponding SAS Date Informats*

|  |  |
| --- | --- |
| **Date Expression** | **SAS Date Informat** |
| 30May00 | **DATE7.** |
| 30May2000 | **DATE9.** |
| 30-May-2000 | **DATE11.** |

* **TIME*w***. Informat

基本形式如：*hh:mm:ss.ss.*

* If you do not enter a value for *ss.ss*, a value of zero is assumed
* *Time Expressions and Corresponding SAS Date Informats*

|  |  |
| --- | --- |
| **Time Expression** | **SAS Time Informat** |
| 17:00:01.34 | **TIME11.** |
| 17:00 | **TIME5.** |
| 2:34 | **TIME5.** |

注意：在最后一个例子中2:34只有4个columns，但是却specify了width of 5，因为width 5是**TIME*w***.的minimum acceptable field width。

* **DATETIME*w*.** Informat

基本形式如：*ddmmmyy hh:mm:ss.ss.*

* You must use delimiters to separate the values for hour, minutes, and seconds
* *Date and Time Expressions and Corresponding SAS Date Informats*

|  |  |
| --- | --- |
| **Date and Time Expression** | **SAS Date Informat** |
| 30May2000:10:03:17.2 | **DATETIME20.** |
| 30May00 10:03:17.2 | **DATETIME18.** |
| 30May2000/10:03 | **DATETIME15.** |

* **YEARCUTOFF**= SAS System Option
* YEARCUTOFF= Option只影响two-digit year，对于four-digit year来说无论它的值是否符合YEARCUTOFF= 的100 years span都会被默认为正确。
* The default YEARCUTOFF= value is 1920.
* Two common error:
  + The informat specifies a w value that is too small to read the entire value, so the last two digits of the year are truncated.

|  |  |  |  |
| --- | --- | --- | --- |
| **Date Expression** | **SAS Date Informat** | **Truncated Date** | **Interpreted As** |
| 17Mar1783 | **DATE7.** | 17Mar17 | 17Mar2017 |

* + Use the wrong informat to read a date or time expression. The SAS log displays an invalid data message, and the variable's values are set to missing



1. Using Dates and Times in Calculations

Eg:

options **YEARCUTOFF**=1920;

**DATA** perm.aprbills;

**INFILE** aprdata;

**INPUT** LastName $8. @10 DateIn **MMDDYY**8. +1 DateOut **MMDDYY**8. +1 RoomRate 6. @35 EquipCost 6.;

****Days=dateout-datein+1;

RoomCharge=days\*roomrate;

Total=roomcharge+equipcost;

**RUN**;

**PROC** **PRINT** data=perm.aprbills;

**RUN**;

****

1. Using Date and Time Formats

Specify how date and time values are displayed

* **WEEKDATE*w*.** Format
* Basic code:

**WEEKDATE***w.*

The **WEEKDATE***w.* format writes date values in the form *day-of-week*, *month-name dd*, *yy* (or *yyyy*).

* Eg:

**PROC** **PRINT** data=perm.aprbills;

**FORMAT** datein dateout **WEEKDATE17**.;

**RUN**;

*PROC PRINT Output for the above Example*



* **WEEKDATE*w*.** Format

|  |  |
| --- | --- |
| **FORMAT Statement** | **Result** |
| **FORMAT** datein **WEEKDATE**3.; | Mon |
| **FORMAT** datein **WEEKDATE**6.; | Monday |
| **FORMAT** datein **WEEKDATE**17.; | Mon, Apr 5, 1999 |
| **FORMAT** datein **WEEKDATE**21.; | Monday, April 5, 1999 |

* The **WORDDATEw.** Format

Similar to **WEEKDATE**, but does not display the day of the week or the two-digit year values

* Basic code:

**WORDDATE***w.*

The **WORDDATE***w*. format writes date values in the form month-name dd, yyyy.

注意：If the w value is too small to write the complete month, SAS abbreviates as needed

* Eg:

**PROC** **PRINT** data=perm.aprbills;

**FORMAT** datein dateout **WORDDATE12**.;

**RUN**;

*PROC PRINT Output for the above Example*

**

* *FORMAT Statements and Corresponding Results*

|  |  |
| --- | --- |
| **FORMAT Statement** | **Result** |
| **FORMAT** datein **WORDDATE** 3.; | Apr |
| **FORMAT** datein **WORDDATE** 5.; | April |
| **FORMAT** datein **WORDDATE** 14.; | April 15, 1999 |

练习

1. An input data file has date expressions in the form 10222001. Which SAS informat should you use to read these dates?
2. DATE6.
3. DATE8.
4. MMDDYY6.
5. MMDDYY8.
6. Suppose your input data file contains the date expression 13APR2009. The YEARCUTOFF=system option is set to 1910. SAS will read the date as
7. 13APR1909
8. 13APR1920
9. 13APR2009
10. 13APR2020
11. Suppose the YEARCUTOFF= system option is set to 1920. An input file contains the date expression 12/08/1925, which is being read with the MMDDYY8. informat. Which date will appear in your data?
12. 08DEC1920
13. 08DEC1925
14. 08DEC2019
15. 08DEC2025
16. Suppose your program creates two variables from an input file. Both variables are stored as SAS date values: FirstDay records the start of a billing cycle, and LastDay records the end of that cycle. The code for calculating the total number of days in the cycle would be
17. TotDays=lastday-firstday;
18. TotDays=lastday-firstday+1;
19. TotDays=lastday/firstday;
20. You cannot use date values in calculations.