CHAPTER 13

Dates, Times, and Intervals

SAS Date, Time, and Datetime Values **147**
Definitions **147**
Two-Digit and Four-Digit Years **148**
The Year 2000 **148**
Working with SAS Dates and Times **149**
Informats and Formats **149**
Date and Time Tools by Task **149**
Examples **154**
Example 1: Displaying Date, Time, and Datetime Values as Recognizable Dates and Times **154**
Example 2: Reading, Writing, and Calculating Date Values **155**
International Date, Time and Datetime Formats **156**
Date and Time Intervals **162**
Definitions **162**
Syntax **162**
Intervals By Category **163**
Example 3: Calculating a Duration **164**
Boundaries of Intervals **165**
Single-Unit Intervals **166**
Multiunit Intervals **167**
Multiunit Intervals Other Than Multiweek Intervals **167**
Multiweek Intervals **168**
Shifted Intervals **168**
How to Use Shifted Intervals **169**
How the SAS System Creates Shifted Intervals **169**

SAS Date, Time, and Datetime Values

Definitions

SAS date value

is a value that represents the number of days between January 1, 1960, and a specified date. SAS can perform calculations on dates ranging from A.D. 1582 to A.D. 19,900. Dates before January 1, 1960, are negative numbers; dates after are positive numbers.

- □ SAS date values account for all leap year days, including the leap year day in the year 2000.
- □ SAS date values can reliably tell you what day of the week a particular day fell on as far back as September 1752, when the calendar was adjusted by

dropping several days. SAS day-of-the-week and length-of-time calculations are accurate in the future to A.D. 19,900.

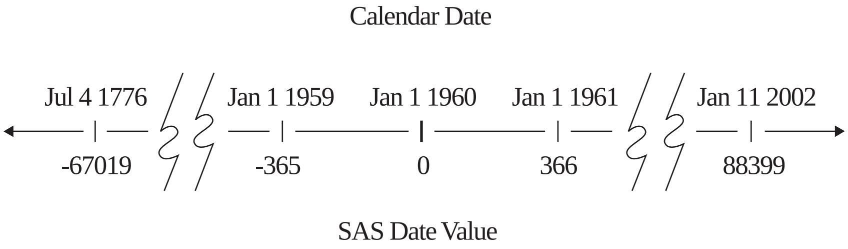
□ Various SAS language elements handle SAS date values: functions, formats and informats.

SAS time value
is a value representing the number of seconds since midnight of the current day. SAS time values are between 0 and 86400.

SAS datetime value
is a value representing the number of seconds between January 1, 1960 and an hour/minute/second within a specified date.

The following figure shows some dates written in calendar form and as SAS date values.

Figure 13.1 How SAS Converts Calendar Dates to SAS Date Values



Two-Digit and Four-Digit Years

SAS software can read two-digit or four-digit year values. If SAS encounters a two-digit year, the YEARCUTOFF= option can be used to specify which century within a 100 year span the two-digit year should be attributed to. For example, YEARCUTOFF=1950 means that two-digit years 50 through 99 correspond to 1950 through 1999, while two-digit years 00 through 49 correspond to 2000 through 2049. Note that while the default value of the YEARCUTOFF= option in Version 8 of the SAS System is 1920, you can adjust the YEARCUTOFF= value in a DATA step to accomodate the range of date values you are working with at the moment. To correctly handle 2-digit years representing dates between 2000 and 2099, you should specify an appropriate YEARCUTOFF= value between 1901 and 2000. See the “How to Read Two-Digit Years Using YEARCUTOFF=” on page 172 section for more information on the YEARCUTOFF= system option.

The Year 2000

SAS software treats the year 2000 like any other leap year. If you use two-digit year numbers for dates, you’ll probably need to adjust the default setting for the YEARCUTOFF= option to work with date ranges for your data, or switch to four-digit years. The following program changes the YEARCUTOFF= value to 1950. This change means that all two digit dates are now assumed to fall in the 100-year span from 1950 to 2049.

```
options yearcutoff=1950;
data _null_;
   a='26oct02'd;
   put 'SAS date='a;
   put 'formatted date='a date9.;
run;
```

The PUT statement writes the following lines to the SAS log:

```
SAS date=15639
formated date=26OCT2002
```

Note: Whenever possible, specify a year using all four digits. Most SAS date and time language elements support four digit year values. △

Working with SAS Dates and Times

Informats and Formats

The SAS System converts date, time and datetime values back and forth between calendar dates and clock times with SAS language elements called *formats* and *informats*.

- Formats present a value, recognized by SAS, such as a time or date value, as a calendar date or clock time in a variety of lengths and notations.
- Informats read notations or a value, such as a clock time or a calendar date, which may be in a variety of lengths, and then convert the data to a SAS date, time, or datetime value.

Date and Time Tools by Task

The following table correlates tasks with various SAS System language elements that are available for working with time and date data.

Table 13.1 Tasks with Dates and Times, Part 1

To do this ...	Use this ...	List	Input	Result
Write SAS date values in recognizable forms	Date formats	DATE*w*.	14686	17MAR00
		DATE9.	14686	17MAR2000a
		DAY*w*.	14686	17
		DDMMYY*w*.	14686	17/03/00
		DDMMYY10.	14686	17/03/2000
		DDMMYYB*w*.	14686	17 03 00
		DDMMYYB10.	14686	17 03 2000
		DDMMYYC*w*.	14686	17:03:20
		DDMMYYC10.	14686	17:03:2000

To do this ...	Use this ...	List	Input	Result
		DDMMYYD*w*.	14686	17-03-00
		DDMMYYD10.	14686	17-03-2000
		DDMMYYN*w*.	14686	17MAR00
		DDMMYYN10	14686	17MAR2000
		DDMMYYP*w*.	14686	17.03.00
		DDMMYYP10.	14686	17.03.2000
		DDMMYYS*w*.	14686	17/03/00
		DDMMYYS10.	14686	17/03/2000
		DOWNAME.	14686	Friday
		EURDFDE*w*.	14686	17MAR00
		EURDFDE9.	14686	17MAR2000
		EURDFDN*w*.	14686	5
		EURDFDWN*w*.	14686	Friday
		EURDFMY*w*.	14686	MAR00
		EURDFDMY7	14686	MAR2000
		EURDFWDX*w*.	14686	17MAR2000
		EURDFMNw.	14686	March
		EURDFWKX*w*.	14686	Friday, 17 MAR 2000
		JULDAY*w*.	14686	77
		JULIAN*w*.	14686	00077
		MINGUO*w*.	14686	89/03/17
		MINGUO10.	14686	0089/03/17
		MMDDYY*w*.	14686	03/17/00
		MMDDYY10.	14686	03/17/2000
		MMDDYYB*w*.	14686	03 17 00
		MMDDYYB10.*w*.	14686	03 17 2000
		MMDDYYC*w*.	14686	03:17:00
		MMDDYYC10	14686	03:17:2000
		MMDDYYD*w*.	14686	03-17-00
		MMDDYYD10.	14686	03-17-2000
		MMDDYYN*w*.	14686	031700
		MMDDYYN10.	14686	03172000
		MMDDYYP	14686	03.17.00
		MMDDYYP10.	14686	03.17.2000
		MMDDYYS	14686	03/17/00
		MMDDYYS10.	14686	03/17/2000

To do this …	Use this …	List	Input	Result
		MMYY.*xw*.	14686	03M2000
		MMYYC*w*.	14686	03:2000
		MMYYD.	14686	03-2000
		MMYYN.	14686	032000
		MMYYP.	14686	03.2000
		MMYYS.	14686	03/2000
		MONNAME.	14686	March
		MONTH.	14686	3
		MONYY.	14686	MAR2000
		NENGO.	14686	H.12/03/17
		PDJULG*w*.	14686	2000077F
		PDJULI*w*.	14686	0100077F
		QTR*w*.	14686	1
		QTRR*w*.	14686	I
		TIME*w.d*	14686	4:04:46
		TIMEAMPM*w.d*	14686	4:04:46 AM
		TOD	14686	4:04:46
		WEEKDATE*w*.	14686	Friday, March 17, 2000
		WEEKDAY*w*.	14686	6
		WORDDATE.*w*.	14686	March 17, 2000
		WORDDATX*w*.	14686	17 MARCH 2000
		YEAR*w*.	14686	2000
		YYMM*w*.	14686	2000M03
		YYMMC*w*.	14686	2000:03
		YYMMDD*w*.	14686	2000-03
		YYMMP*w*.	14686	2000.03
		YYMMS.	14686	2000/03
		YYMMN.	14686	200003
		YYMMDD*w*.	14686	00-03-17
		YYMON.	14686	2000MAR
		YYQ*xw*.	14686	2000Q1
		YYQC*w*.	14686	2000:1
		YYQD*w*.	14686	2000-1
		YYQP*w*.	14686	2000.1

To do this ...	Use this ...	List	Input	Result
		YYQS*w*.	14686	2000/1
		YYQN*w*.	14686	20001
		YYQR*w*.	14686	2000QI
		YYQRC*w*.	14686	2000:I
		YYQRD*w*.	14686	2000-I
		YYQRP*w.w*.	14686	2000.I
		YYQRS*w*.	14686	2000/I
		YYQRN*w*.	14686	III

Table 13.2 Tasks with Dates and Times, Part 2

To do this ...	Use this ...	List	Input	Result
Date Tasks				
Read calendar dates as SAS date Note: YEARCUTOFF=1920	Date informats	DATE*w*.	17MAR2000	-14534
		DATE9.	17MAR2000	14686
		DDMMYY*w*.	170300	14686
		DDMMYY8.	17032000	14686
		JULIAN*w*.	0077	14686
		JULIAN7.	2000077	14686
		MMDDYY*w*.	031700	14686
		MMDDYY10.	03172000	14686
		MONYY*w*.	MAR00	14670
		NENGO*w*.	H.12/03/17	14686
		YYMMDD*w*.	000317	14686
		YYMMDD10.	20000317	14686
		YYQ*w*.	00Q1	14610
Create date values from pieces	Date functions	DATEJUL	2000077	14686
		DHMS	'17MAR2000'D, 00,00,00	14686
		HMS	14,45,32	53132
		MDY	03,17,00	14686
		MDY	03,17,2000	14686
		YYQ	00,1	14610
Extract a date from a datetime value	Date functions	DATEPART	'17MAR00:00:00 'DT	14686

To do this ...	Use this ...	List	Input	Result
Return today’s date as a SAS date	Date functions	DATE() or TODAY() (equivalent)	( )	SAS date for today
Extract calendar dates from SAS	Date functions	DAY	14686	17
		HOUR	14686	4
		JULDATE	14686	0077
		JULDATE7	14686	2000077
		MINUTE	14686	4
		MONTH	14686	3
		QTR	14686	3
		SECOND	14686	46
		WEEKDAY	14686	6
		YEAR	14686	2000
Write a date as a constant in an expression	SAS date constant	’*ddmmmyy*’d or ’*ddmmmyyyy*’	**’17mar00’d** **’17mar2000’d**	14686
Write today’s date as a string	SYSDATE automatic macro variable	SYSDATE	&SYSDATE	Date at time of SAS initialization in DDMMMYY
	SYSDATE9	SYSDATE9	&SYSDATE9	Date at time of SAS initialization in DDMMMYYYY
Time Tasks				
Write SAS time values as time values	time formats	HHMM.	53132	14:46
		HOUR.	53132	15
		MMSS.	53132	885
		TIME.	53132	14:45:32
		TOD.	53132	14:45:32
Read time values as SAS time values	Time informats	TIME	14:45:32	53132
Write the current time as a string	SYSTIME automatic macro variable	SYSTIME	&SYSTIME	Time at moment of execution in HH:MM

To do this ...	Use this ...	List	Input	Result
Return the current time of day as a SAS time value	Time functions	TIME( )	( )	SAS time value at moment of execution in NNNNN.NN
Return the time part of a SAS datetime value	Time functions	TIMEPART	SAS datetime value in NNNNNNNNNN.N	SAS time value part of date value in NNNNN.NN
Datetime Tasks				
Write SAS datetime values as datetime values	Datetime formats	DATEAMPM	1217083532	26JUL98:02:45 PM
		DATETIME	1268870400	17MAR00:00:00:00
		EURDFDT	1217083532	26JUL98:14:45:32
Read datetime values as SAS datetime values	Datetime informats	DATETIME	17MAR00:00:00:00	1268870400
Return the current date and time of day as a SAS datetime value	Datetime functions	DATETIME()	()	SAS datetime value at moment of execution in NNNNNNNNNN.N
Interval Tasks				
Return the number of specified time intervals that lie between the two date or datetime values	Interval functions	INTCK	week 2 01aug60 01jan01	1055
Advances a date, time, or datetime value by a given interval, and returns a date, time, or datetime value	Interval functions	INTNX	day 14086 01jan60	14086

The SAS System also supports international formats and informats that are equivalent to some of the most commonly used English-language date formats and informats. For details, see "SAS Formats" and "SAS Informats."

Examples

Example 1: Displaying Date, Time, and Datetime Values as Recognizable Dates and Times

The following example demonstrates how a value may be displayed as a date, a time, or a datetime. Remember to select the SAS language element that converts a SAS date, time, or datetime value to the intended date, time or datetime format. See the previous tables for examples.

Note:

□ Time formats count the number of seconds within a day, so the values will be between 0 and 86400.

□ DATETIME formats count the number of seconds since January 1, 1960, so for datetimes that are greater than 02JAN1960:00:00:01, (integer of 86401) the datetime value will always be greater than the time value.

□ When in doubt, look at the contents of your data set for clues as to which type of value you are dealing with.

△

This program uses the DATETIME, DATE and TIMEAMPM formats to display the value 86399 to a date and time, a calendar date, and a time.

```
data test;
options nodate pageno=1 linesize=80 pagesize=60;
Time1=86399;
format Time1 datetime.;
Date1=86399;
format Date1 date.;
Time2=86399;
format Time2 timeampm.;
run;
proc print data=test;
title  'Same Number, Different SAS Values';
footnote1 'Time1 is a SAS DATETIME value';
footnote2 'Date1 is a SAS DATE value';
footnote3 'Time2 is a SAS TIME value'.;
run;
```

Output 13.1 Datetime, Date and Time Values for 86399

```
                    Same Number, Different SAS Values               1

            Obs         Time1           Date1        Time2

             1     01JAN60:23:59:59    20JUL96    11:59:59 PM



                      Time1 is a SAS DATETIME value
                        Date1 is a SAS DATE value
                        Time2 is a SAS TIME value.
```

Example 2: Reading, Writing, and Calculating Date Values

This program reads four regional meeting dates and calculates the dates on which announcements should be mailed.

```
data meeting;
options nodate pageno=1 linesize=80 pagesize=60;
   input region $ mtg : mmddyy8.;
   sendmail=mtg-45;
   datalines;
```

```
N  11-24-99
S  12-28-99
E  12-03-99
W  10-04-99
;

proc print data=meeting;
   format mtg sendmail date9.;
   title 'When To Send Announcements';
run;
```

Output 13.2 Calculated Date Values: When to Send Mail

```
                 When To Send Announcements

        Obs    region         mtg     sendmail

         1       N       24NOV1999    10OCT1999
         2       S       28DEC1999    13NOV1999
         3       E       03DEC1999    19OCT1999
         4       W       04OCT1999    20AUG1999
```

International Date, Time and Datetime Formats

The format for date and time information may vary from language to language. This is in addition to the translation of the words. The format can also vary from country to country for the same language.

Operating Environment Information: Some of the details of how date time and datetime formats are implemented depends on your operating environment. For additional information, see the SAS documentation for your operating environment. △

The following table summarizes the details of the available SAS date, time and datetime formats.

Table 13.3 International Date and Datetime Formats

Language	English Format	International Format	Min	Max	Default
Afrikaans (AFR)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	9
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWK.	2	38	28
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFDE.	3	37	29

Language	English Format	International Format	Min	Max	Default
Catalan (CAT)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	9
	MONNAME.	EURDFMN.	1	32	8
	MONYY.	EURDFMY.	5	32	5
	WEEKDATX.	EURDFWKX.	2	40	27
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EUDFWDX.	3	40	16
Croatian (CRO)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	10
	MONNAME.	EURDFMN.	1	32	8
	MONYY.	EURDFMY.	5	32	5
	WEEKDATX.	EURDFWKX.	3	40	27
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	40	16
Czech (CSY)	DATE.	EURDFDE.	10	14	12
	DATETIME.	EURDFDT.	12	40	21
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFWN.	1	32	7
	MONNAME.	EURDFMN.	1	32	8
	MONYY.	EURDFMY.	10	32	10
	WEEKDATX.	EURDFWKX.	2	40	25
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	8	40	16
Danish (DAN)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	7
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	2	31	31
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	18	18
Dutch (NLD)	DATE.	EURDFDE.	5	9	7

Language	English Format	International Format	Min	Max	Default
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	9
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	2	38	28
	WORDDATX.	EURDFWDX.	3	37	29
	WEEKDAY.	EURDFDN.	1	32	1
Finnish (FIN)	DATE.	EURDFDE.	9	10	9
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	10
	DOWNAME.	EURDFDWN.	1	32	11
	MONNAME.	EURDFMN.	1	32	11
	MONYY.	EURDFMY.	8	8	8
	WEEKDATX.	EURDFWKX.	2	37	37
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	20	20
French (FRA)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	8
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	27	27
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	18	18
German (DEU)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	10
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	30	30
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	18	18
Hungarian (HUN)	DATE.	EURDFDE.	8	12	10
	DATETIME.	EURDFDT.	0	40	19

Language	English Format	International Format	Min	Max	Default
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	9
	MONNAME.	EURDFMN.	1	32	10
	MONYY.	EURDFMY.	8	32	8
	WEEKDATX.	EURDFWKX.	3	40	28
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	6	40	18
Italian (ITA)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	9
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	28	28
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	17	17
Macedonian (MAC)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	10
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	32	5
	WEEKDATX.	EURDFWKX.	3	40	29
	WEEKDDATX.	EURDFWDX.	1	32	1
	WORDDATX.	EURDFDN.	3	40	17
Norwegian (NOR)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	7
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	26	26
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	17	17
Polish (POL)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8

Language	English Format	International Format	Min	Max	Default
	DOWNAME.	EURDFDWN.	1	32	12
	MONNAME.	EURDFMN.	1	32	12
	MONYY.	EURDFMY.	5	32	5
	WEEKDATX.	EURDFWKX.	2	40	34
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	40	17
Portuguese (PTG)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	13
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	38	38
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	37	23
Russian (RUS)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	11
	MONNAME.	EURDFMN.	1	32	8
	MONYY.	EURDFMY.	5	32	5
	WEEKDATX.	EURDFWKX.	2	40	29
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	40	16
Spanish (ESP)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	9
	MONNAME.	EURDFMN.	1	32	10
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	1	35	35
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	24	24
Slovenian (SLO)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	10

Language	English Format	International Format	Min	Max	Default
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	32	5
	WEEKDATX.	EURDFWKX.	3	40	29
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	40	17
Swedish (SVE)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	7
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	26	26
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	17	17
Swiss_French (FRS)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	8
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	26	26
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	17	17
Swiss_German (DES)	DATE.	EURDFDE.	5	9	7
	DATETIME.	EURDFDT.	7	40	16
	DDMMYY.	EURDFDD.	2	10	8
	DOWNAME.	EURDFDWN.	1	32	10
	MONNAME.	EURDFMN.	1	32	9
	MONYY.	EURDFMY.	5	7	5
	WEEKDATX.	EURDFWKX.	3	30	30

Language	English Format	International Format	Min	Max	Default
	WEEKDAY.	EURDFDN.	1	32	1
	WORDDATX.	EURDFWDX.	3	18	18

Date and Time Intervals

Definitions

duration
is an integer representing the difference, in number of days, between any two dates or times or datetimes.

interval
is a unit of measurement that SAS can count within an elapsed period of time, such as DAYS, TENDAYS and SEMIMONTHS.

Syntax

SAS provides date, time, and datetime intervals for counting different periods of elapsed time. You can create multiples of the intervals and shift their starting point. Use them with the INTCK and INTNX functions and with procedures that support numbered lists (such as the PLOT procedure). The form of an interval is

name<*multiple*><*.starting-point*>

The terms in an interval have the following definitions:

name
is the name of the interval. See the following table for a list of intervals and their definitions.

multiple
creates a multiple of the interval. *Multiple* can be any positive number. The default is 1. For example, YEAR2 indicates a two-year interval.

.starting-point
is the starting point of the interval. By default, the starting point is 1. A value greater than 1 shifts the start to a later point within the interval. The unit for shifting depends on the interval, as shown in the following table. For example, YEAR.3 specifies a yearly period from the first of March through the end of February of the following year.

Intervals By Category

Table 13.4 Intervals Used with Date and Time Functions

Category	Interval	Definition	Default Starting Point	Shift Period	Example	Description
Date	DAY	Daily intervals	Each day	Days	DAY3	Three-day intervals starting on Sunday
	WEEK	Weekly intervals	Each Sunday	Days (1=Sunday ... 7=Saturday	WEEK.7	Weekly with Saturday as the first day of the week
	WEEKDAY <*days*W>	Daily intervals with weekend days treated as part of the preceding weekday. *Days* identifies the weekend days by number (1=Sunday ... 7=Saturday). By default, *days*=17.	Each day	Days	WEEKDAY1W	Six-day week with Sunday as a weekend day
					WEEKDAY35W	Five-day week with Tuesday and Thursday as weekend days (W indicates that day 3 and day 5 are weekend days)
	TENDAY	Ten-day intervals (a U.S. automobile industry convention)	First, eleventh, and twenty-first of each month	TENDAY periods	TENDAY4.2	Four ten-day periods starting at the second TENDAY period
	SEMIMONTH	Half-month intervals	First and sixteenth of each month	SEMIMONTH periods	SEMIMONTH2.2	Intervals from the sixteenth of one month through the fifteenth of the next month

Category	Interval	Definition	Default Starting Point	Shift Period	Example	Description
	MONTH	Monthly intervals	First of each month	Months	MONTH2.2	February-March, April-May, June-July, August-September, October-November, and December-January of the following year
	QTR	Quarterly (three-month) intervals	January 1 April 1 July 1 October 1 April 1 July 1 October 1	Months	QTR3.2	three-month intervals starting on April 1, July 1, October 1, and January 1
	SEMIYEAR	Semiannual (six-month) intervals	January 1 July 1	Months	SEMIYEAR.3	Six-month intervals, March-August and September-February
	YEAR	Yearly intervals	January 1	Months		
Datetime	Add DT	To any date interval			DTMONTH DTWEEKDAY	
Time	SECOND	Second intervals	Each second	Seconds		
	MINUTE	Minute intervals	Each minute	Minutes		
	HOUR	Hourly intervals	Each hour	Hours		

Example 3: Calculating a Duration

This program reads the project start and end dates and calculates the duration between them.

```
data projects;
options nodate pageno=1 linesize=80 pagesize=60;
   input Projid startdate date9. enddate date9.;
   Duration=enddate-startdate;
   datalines;
398 17oct1997 02nov1997
942 22jan1998 10mar1998
167 15dec1999 15feb2000
250 04jan2001 11jan2001
;

proc print data=projects;
   format startdate enddate date9.;
      title 'Days Between Project Start and Project End';
run;
```

Output 13.3 Output from the PRINT Procedure

```
              Days Between Project Start and Project End run                8

             Obs    Projid    Startdate      Enddate    Duration

              1       398      17OCT1997  02NOV1997       16
              2       942      22JAN1998  10MAR1998       47
              3       167      15DEC1999  15FEB2000       62
              4       250      04JAN2001  11JAN2001        7
```

Boundaries of Intervals

The SAS System associates date and time intervals with fixed points on the calendar. For example, the MONTH interval represents the time from the beginning of one calendar month to the next, not a period of 30 or 31 days. When you use date and time intervals (for example, with the INTCK or INTNX functions), the SAS System bases its calculations on the calendar divisions that are present. Consider the following examples:

Table 13.5 Using INTCK And INTNX

Example	Results	Explanation
mnthnum1= **intck(** **’month’,** **’25aug2000’d,** **’05sep2000’d);**	mnthnum1=1	The number of MONTH intervals the INTCK function counts depends on whether the first day of a month falls within the period.
mnthnum2= **intck(** **’month’,** **’01aug2000’d,** **’31aug2000’d);**	mnthnum2=0	
next=intnx(**’month’,** **’25aug2000’d,1);**	next represents 01sep2000	The INTNX function produces the SAS date value that corresponds to the beginning of the next interval.

Note: The only intervals that do not begin on the same date in each year are WEEK and WEEKDAY. A Sunday can occur on any date because the year is not divided evenly into weeks. △

Single-Unit Intervals

Single-unit intervals begin at the following points on the calendar:

Table 13.6 Single-Unit Intervals

These single-unit intervals	Begin at this point on the calendar
DAY and WEEKDAY	each day
WEEK	each Sunday
TENDAY	the first, eleventh, and twenty-first of each month
SEMIMONTH	the first and sixteenth of each month
MONTH	the first of each month
QTR	the first of January, April, July and October
SEMIYEAR	the first of January and July
YEAR	the first of January

Single-unit time intervals begin as follows:

Table 13.7 Single-Unit Time Intervals

These single-unit time intervals	Begin at this point
SECOND	each second
MINUTE	each minute
HOUR	each hour

Multiunit Intervals

Multiunit Intervals Other Than Multiweek Intervals

Multiunit intervals, such as MONTH2 or DAY50, also depend on calendar measures, but they introduce a new problem: the SAS System can find the beginning of a unit (for example, the first of a month), but where does that unit fall in the interval? For example, does the first of October mark the first or the second month in a two-month interval?

For all multiunit intervals except multiweek intervals, the SAS System creates an interval beginning on January 1, 1960, and counts forward from that date to determine where individual intervals begin on the calendar. As a practical matter, when a year can be divided evenly by an interval, think of the intervals as beginning with the current year. Thus, MONTH2 intervals begin with January, March, May, July, September, and November. Consider this example:

Table 13.8 Month2 Intervals

SAS statements	Results
howmany1=intck ('month2','15feb2000'd,'15mar2000'd);	howmany1=1
count=intck ('day50','01oct2000'd,'01jan2000'd);	count=1

In the above example, the SAS System counts 50 days beginning with January 1, 1960; then another 50 days; and so on. As part of this count, the SAS System counts one DAY50 interval between October 1, 1998 and January 1, 1999. As an example, to determine the date on which the next DAY50 interval begins, use the INTNX function, as follows:

Table 13.9 Using the INTNX Function

SAS statements	Results
start=intnx ('day50','01oct98'd,1);	SAS date value 14200, or Nov 17, 1998

The next interval begins on November 17, 1998.

Time intervals (those that represent divisions of a day) are aligned with the start of the day, that is, midnight. For example, HOUR8 intervals divide the day into the periods 00:00 to 08:00, 8:00 to 16:00, and 16:00 to 24:00 (the next midnight).

Multiweek Intervals

Multiweek intervals, such as WEEK2, present a special case. In general, weekly intervals begin on Sunday, and the SAS System counts a week whenever it passes a Sunday. However, the SAS System cannot calculate multiweek intervals based on January 1, 1960, because that date fell on a Friday, as shown:

Figure 13.2 Calculating Multi Week Intervals

Dec	Su	Mo	Tu	We	Th	Fr	Sa	Jan
1959	27	28	29	30	31	1	2	1960

Therefore, the SAS System begins the first interval on Sunday of the week containing January 1, 1960—that is, on Sunday, December 27, 1959. The SAS System counts multiweek intervals from that point. The following example counts the number of two-week intervals in the month of August, 1998:

Table 13.10 Counting Two-Week Intervals

SAS statements	Results
count=intck ('week2','01aug98'D, '31aug98'D);	count=3

To see the beginning date of the next interval, use the INTNX function, as shown here:

Table 13.11 Using INTNX to See The Beginning Date of an Interval

SAS statements	Results
begin=intnx('week2','01aug1998'd,1);	"Begin" represents SAS date 14093 or August 02, 1998

The next interval begins on August 16.

Shifted Intervals

Shifting the beginning point of an interval is useful when you want to make the interval represent a period in your data. For example, if your company's fiscal year begins on July 1, you can create a year beginning in July by specifying the YEAR.7 interval. Similarly, you can create a period matching U.S. presidential elections by specifying the YEAR4.11 interval. This section discusses how to use shifted intervals and how the SAS System creates them.

How to Use Shifted Intervals

When you shift a time interval by a subperiod, the shift value must be less than or equal to the number of subperiods in the interval. For example, YEAR.12 is valid (yearly periods beginning in December), but YEAR.13 is not. Similarly, YEAR2.25 is not valid because there is no twenty-fifth month in the two-year period.

In addition, you cannot shift an interval by itself. For example, you cannot shift the interval MONTH because the shifting subperiod for MONTH is one month and MONTH contains only one monthly subperiod. However, you can shift multi-unit intervals by the subperiod. For example, MONTH2.2 specifies bimonthly periods starting on the first day of the second month.

How the SAS System Creates Shifted Intervals

For all intervals except those based on weeks, the SAS System creates shifted intervals by creating the interval based on January 1, 1960, by moving forward the required number of subperiods, and by counting shifted intervals from that point. For example, suppose you create a shifted interval called DAY50.5. The SAS System creates a 50-day interval in which January 1, 1960 is day 1. The SAS System then moves forward to day 5. (Note that the *difference*, or amount of movement, is 4 days.) The SAS System begins counting shifted intervals from that point. The INTNX function demonstrates that the next interval begins on January 5, 1960:

Table 13.12 Using INTNX to Determine When an Interval Begins

SAS statements	Results
start=intnx ('day50.5','01jan1960'd,1);	SAS date value 4, or Jan 5, 1960

For shifted intervals based on weeks, the SAS System first creates an interval based on Sunday of the week containing January 1, 1960 (that is, December 27, 1959), then moves forward the required number of days. For example, suppose you want to create the interval WEEK2.8 (biweekly periods beginning on the second Sunday of the period). The SAS System measures a two-week interval based on Sunday of the week containing January 1, 1960, and begins counting shifted intervals on the eighth day of that. The INTNX function shows the beginning of the next interval:

Table 13.13 Using the INTNX Function to Show the Beginning of the Next Interval

SAS statements	Results
start=intnx ('week2.8','01jan1960'd,1);	SAS date value 2, or Jan 3, 1960

You can also shift time intervals. For example, HOUR8.7 intervals divide the day into the periods 06:00 to 14:00, 14:00 to 22:00, and 22:00 to 06:00.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS Language Reference: Concepts*, Cary, NC: SAS Institute Inc., 1999. 554 pages.

SAS Language Reference: Concepts

ISBN 1–58025–441–1

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, November 1999

The Institute is a private company devoted to the support and further development of its software and related services.