

# SAS Date, Date/Time and Time Variables, Formats and Functions

---

*Paige Miller*

Credit Risk Management

1/30/20

**M&T** Bank

# SAS Date and Date/Time variables

- SAS provides three different types of clock and calendar variables
  - Date
  - Datetime
  - Time
- All three types of variables are numeric
  - Date variables: an integer representing the number of days since January 1, 1960. Thus, January 1, 2020 is represented as 21,915
  - Datetime variables: a number representing the number of seconds since midnight on January 1, 1960. Thus, midnight on January 1, 2020 is 1,893,456,000.
  - Time variables: number of seconds after midnight (not discussed further in this talk, but the principles are the same)

## SAS Date and Date/Time variables

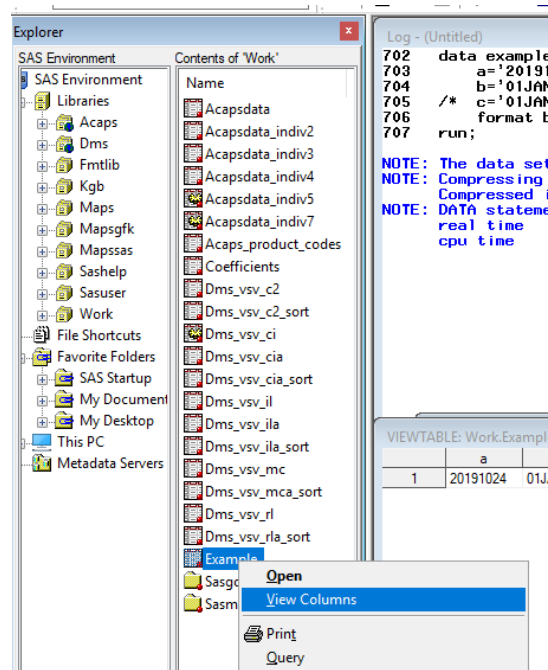
- In order to properly use SAS date and datetime variables, you **first** have to determine in a variables is:
  - Numeric or Character
  - Formatted or unformatted
  - Is or is not a SAS date or datetime value
- Example:

	a	b	c
1	20191024	01JAN20	1894618800

- Is variable A or B or C numeric or character?
- Is either A or B or C formatted or unformatted?
- Sometimes you can't tell (easily) by looking at them

# SAS Date and Date/Time variables

- How can you tell if a variable is numeric or character, and formatted or unformatted?
  - View Columns



The screenshot shows the SAS Explorer interface. The 'Contents of Work' pane lists various files, including 'Example'. A context menu is open over the 'Example' file, with 'View Columns' selected. The Log window shows the following code and output:

```
702 data example
703   a='20191
704   b='01JAN
705   /* c='01JAN
706   format t
707   run;
```

NOTE: The data set  
NOTE: Compressing  
NOTE: DATA statement  
real time  
cpu time

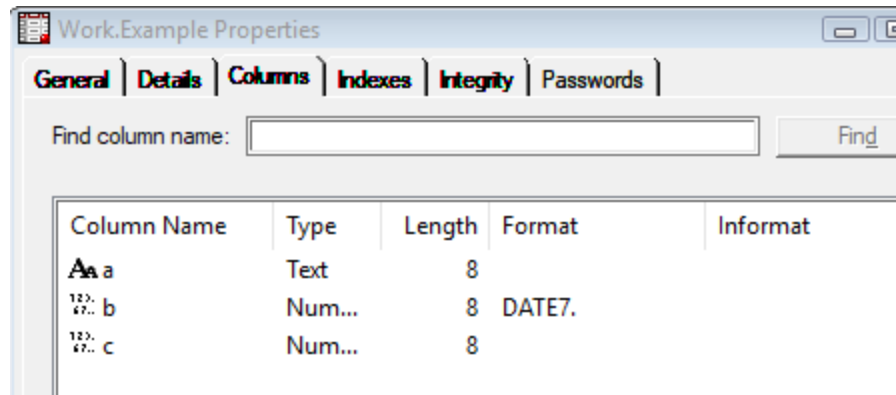
VIEWTABLE: Work.Exempl

	a	
1	20191024	01J

- PROC CONTENTS
- Viewtable

# SAS Date and Date/Time variables

- How can you tell if a variable is numeric or character, and formatted or unformatted?
  - View Columns



The screenshot shows the 'Work.Example Properties' dialog box with the 'Columns' tab selected. It features a search field labeled 'Find column name:' and a 'Find' button. Below is a table with the following data:

Column Name	Type	Length	Format	Informat
A a	Text	8		
<sup>123</sup> <sub>456</sub> b	Num...	8	DATE7.	
<sup>123</sup> <sub>456</sub> c	Num...	8		

- A is text and unformatted
- B is numeric and formatted as DATE7.
- C is numeric and unformatted
- → Knowing these facts and looking at the value of the variable (shown on Slide 3), you can determine if the variable is a date or datetime variable. ←

# SAS Date Variables

- We have determined that B is numeric (so it could be a date variable)
- We have determined that B is formatted as DATE7.
- We see that B appears as 01JAN20
- Now we believe that B is a date variable
  - We can now use any other date format if we don't like DATE7. for variable B
  - The reason we use formats is so humans can understand what date is being referred to
    - Otherwise, SAS doesn't need formats
  - Internally, when performing math or logic, SAS *always* uses unformatted date value of 21915
  - When humans have to enter a date, you can use the format '27DEC2019'D (which could use lower case letters, but no other format)
  - These two statements are equivalent (and it doesn't matter how variable B is formatted)

```
if b>21910 then delete;  
if b>'27DEC2019'd then delete;
```

# Formats for SAS Date Variables

- Other available date formats

[Complete list of SAS date and datetime and time formats in alphabetical order](#)

- List also contains datetime and time formats

**DOWNAME<sub>w</sub>.**  
Format  
Writes **date** values as the name of the day of the week.  
Product: Base SAS  
Document: *SAS Formats and Informats: Reference*

This is a date format

**DTDATE<sub>w</sub>.**  
Format  
Expects a SAS **datetime** value as input and writes the SAS date values in the form *ddmmyy* or *ddmmyyyy*.  
Product: Base SAS

This is a datetime format

- Example (using date format DOWNAME):  
`format b downame3.;`  
makes variable B appear as Wed
- Example 2 (using date format DDMMYYSS):  
`format b ddmmyyss8.;`  
makes variable B appear as 01/01/20

# Functions for SAS Date Variables

- All SAS Date Functions:

[Complete list of SAS date functions in alphabetical order](#)

- List also contains datetime and time functions

## DATEPART Function

DATEPART(**datetime**)

Extracts the date from a SAS datetime value.

Product: Base SAS

Document: *SAS Functions and CALL Routines: Reference*

## DAY Function

DAY(**date**)

Returns the day of the month from a SAS date value.

Product: Base SAS

- You cannot use a SAS datetime function on a SAS date variable
- You cannot use a SAS datetime format on a SAS date variable



# SAS Datetime Variables

- We have determined that C is numeric (so it could be a datetime variable)
- We have determined that C is unformatted
- If we apply a format to C, we see that C appears as 14JAN20:11:00:00
- Now we believe that C is indeed a datetime variable
  - We can now use any datetime format for variable C
  - The reason we use formats is so humans can understand what datetime is being referred to
    - Otherwise, SAS doesn't need formats
  - Internally, when performing math or logic, SAS *always* uses unformatted datetime value of 1894618800
  - When humans need to enter a datetime, you can use the format '14JAN2020:11:00:00'DT (which could use lower case letters, but no other format)
  - These two statements are equivalent (and it doesn't matter how variable C is formatted)

```
if c>1894618800 then delete;  
if c>'14JAN2020:11:00:00'DT then delete;
```

# Formats for SAS Datetime Variables

- Other available datetime formats (same link as before)

[Complete list of SAS date and datetime and time formats in alphabetical order](#)

- List also contains datetime and time formats

<b>DOWNAMEw.</b> Format	<b>DOWNAMEw.</b> Writes <b>date</b> values as the name of the day of the week. Product: Base SAS Document: <i>SAS Formats and Informats: Reference</i>	This is a date format
<b>DTDATEw.</b> Format	<b>DTDATEw.</b> Expects a SAS <b>datetime</b> value as input and writes the SAS date values in the form <i>ddmmyy</i> or <i>ddmmyyyy</i> . Product: Base SAS	This is a datetime format

- Example (using datetime format DTDATE):  
`format c dtdate9.;`  
makes variable C appear as 14JAN2020
- Example 2 (using datetime format B8601DT):  
`format c b8601dt.;`  
makes variable C appear as 20200114T110000

# Functions for SAS Datetime Variables

- All SAS Datetime Functions:

[Complete list of SAS date and datetime and time functions in alphabetical order](#)

- List also contains date and time functions

## DATEPART Function

DATEPART(**datetime**)

Extracts the date from a SAS datetime value.

Product: Base SAS

Document: *SAS Functions and CALL Routines: Reference*

## DAY Function

DAY(**date**)

Returns the day of the month from a SAS date value.

Product: Base SAS

- You cannot use a SAS date function on a SAS datetime variable
- You cannot use a SAS date format on a SAS datetime variable

# INTNX Function

- Increment a date or datetime value by a certain number of intervals
- Syntax:  
INTNX('interval', variablename, increment, 'alignment')
- Example: intnx('week',date\_variable\_name,32,'s') determines what day is 32 weeks after DATE\_VARIABLE\_NAME, same day of the week
- Example: intnx('dtweek',datetime\_variable\_name,32,'s') determines what datetime is 32 weeks after DATETIME\_VARIABLE\_NAME, same day of the week, same time of day

## INTERVALS

Use with Dates	Use with Datetimes
DAY	DTDAY
WEEK	DTWEEK
TENDAY	DTTENDAY
SEMIMONTH	DTSEMIMONTH
MONTH	DTMONTH
QTR	DTQTR
SEMIYEAR	DTSEMIYEAR
YEAR	DTYEAR

## ALIGNMENT

Value	Meaning
'b' (Default)	Beginning
'm'	Middle
'e'	End
's'	Same

Note: the beginning of a week is Sunday. Yes, you can change that.

No, the INTNX function cannot accommodate the Beatles song "Eight Days a Week".

# INTCK Function

- Calculate the number of intervals between two dates
- Syntax:  
INTCK('interval', start\_date, end\_date, 'method')
- Example: intck('month',date\_variable1, date\_variable2,'c') determines the number of months between date variables date\_variable1 and date\_variable2
- Example: intck('dtmonth',datetime\_variable1,datetime\_variable2,'c') determines the number of months between datetime variables datetime\_variable1 and datetime\_variable2.

## INTERVALS

Use with Dates	Use with Datetimes
DAY	DTDAY
WEEK	DTWEEK
TENDAY	DTTENDAY
SEMIMONTH	DTSEMIMONTH
MONTH	DTMONTH
QTR	DTQTR
SEMIYEAR	DTSEMIYEAR
YEAR	DTYEAR

## METHOD

Value	Meaning
'c' (Default)	Continuous (Anniversary)
'd'	Discrete (# of times a boundary is crossed)

## Other things you can do with SAS Date Variables

- What day is 30 days previous?

```
before = b - 30;  
format before yymmdd8.;
```

This works because b is the integer 21915 (regardless of how it is formatted, this subtraction works)

- What day is the first day of the month three months previous?

```
threemonthsearlier = intnx('month',b,-3,'b');  
format threemonthsearlier yymmdd8.;
```

- this does not work with a datetime variable

- How many months since Millard Fillmore's birthday?

```
howmany = intck('month',b,'07JAN1800'd);
```

Answer = -2640

- When you use month or other date interval, both variables must be date variables
- You cannot have one date variable and one datetime variable, or both datetime variables

- How many months since a loan was last delinquent?

```
howmany = intck('month',b,last_delq);
```

where variable last\_delq is a SAS date variable indicating the last time the loan was delinquent

## Other things you can do with SAS Datetime Variables

- What datetime is 27 days previous?

```
before = c - 27*24*60*60;  
format before datetime16.;
```

Because c is the integer 1894618800 (regardless of how it is formatted, this subtraction works)

- What datetime is it at the start of the first day of the month three months previous?

```
threemonthsearlier = intnx('dtmonth',c,-3,'b');  
format threemonthsearlier datetime16.;
```

- When you use the option 'dtmonth', the variable C must be a datetime variable.

- How many months since Millard Fillmore's birthday?

```
howmany = intck('dtmonth',c,'07JAN1800:00:00:00'dt);
```

Answer = -2640

- When you use 'dtmonth' or other dt period, both variables must be datetime variables
- You cannot have one date variable and one datetime variable, or both date variables

- How many months since a loan was last delinquent?

```
howmany = intck('dtmonth',c,last_delq);
```

where variable last\_delq is a SAS datetime variable indicating the last datetime the loan was delinquent

## Converting SAS Datetime Variables to Date Variables

- You use the DATEPART function (which operates on datetime values)

```
date=datepart(c);  
format date date7.;
```

- To determine the month (or year or day of month) of a datetime value, you can use the MONTH() (or YEAR() or DAY() ) functions (but since these are date functions, you have to first convert C to a date variable using the DATEPART function). These functions return integers for month (1=January, 2=February, *etc.*), and integers for year or for day.

```
m=month(datepart(c));  
y=year(datepart(c));  
d=day(datepart(c));
```



## Converting SAS Date Variables to Datetime Variables

- You use the DHMS function, the first argument must be a SAS date value  

```
datepart=dhms(b,0,0,0); /* Day, hour, minute, second */  
format datepart datetime16.;
```

## Converting numeric or character variables that look like dates to SAS Dates

- Suppose variable D is numeric, not formatted, and has the value 20191123
  - Looks like November 23, 2019, but SAS just thinks of it as an integer with no special meaning
  - It is NOT a SAS date variable (remember, SAS date variables are the number of days since 1/1/60 and November 23, 2019 is 21876)
  - You can convert this to an actual SAS date via first turning it into a character string '20191123' using the PUT function, and then using the INPUT function with the proper informat (in this case the ANYDTDTE. informat) to cause SAS to create a SAS date variable

```
date = input(put(d,8.),anydtdte.);  
format date yymmdds8.;
```

- Remember variable A which was a character variable with the value 20191024?  
To convert it to an actual SAS date value

```
date = input(a,anydtdte.);  
format date yymmdds8.;
```

[Complete list of all SAS date and datetime informats](#)

## Using formats to get aggregate statistics in PROC MEANS, PROC SUMMARY, PROC REPORT, etc.

- Suppose you have a SAS data set with PROCESSDATE (a date/time variable) and Loan Balance.
- To obtain the total origination amounts by month of processdate

```
proc report data=mydataset;  
  columns processdate originationamount;  
  define processdate/group format=dtmonyy. order=internal;  
  define originationamount/sum format=dollar16.0;  
run;
```

ProcessDate	OriginationAmount
JAN19	\$3,319,716
FEB19	\$3,396,848
MAR19	\$3,547,712
APR19	\$3,630,067
MAY19	\$3,235,314
JUN19	\$3,746,573
JUL19	\$4,058,305
AUG19	\$4,086,302
SEP19	\$4,622,079
OCT19	\$4,840,816
NOV19	\$3,405,527
DEC19	\$2,905,803

## APPENDIX — How to do calculations if you want weeks starting on Monday (or any other day of the week)

- Days of the week in SAS: 1=Sunday, 2=Monday, *etc.*
- INTNX('week.2',b,10,'b')  
The WEEK.2 indicates that the weeks should be considered starting on Monday
- Also works for INTCK

## APPENDIX — Suppose you want two-week time periods or two-month time periods

- INTNX('week2',b,10, 'b')  
INTNX('month2',b,10, 'b')  
The WEEK2 indicates that two weeks is the time period.
- Also works for INTCK
- Combine both examples on this page, we want three-week time periods beginning on Mondays  
INTNX('week3.2',b,10,'b')

## APPENDIX — Should macro variables be formatted?

- In general, NO!!!
- The only time you format macro variables is for use in Titles or Labels when humans have to view and understand the date or datetime value
- Example: suppose you want all records from a database where the origination date (which is a datetime variable) from 48 months ago to 24 months ago
  - All the math works because formatting is not needed to do math in SAS; all math is done using the unformatted values anyway.

```
%let today=%sysfunc(datetime()); Note: not formatted
%let _24monthsago=%sysfunc(intnx(dtmonth,&today,-24,b)); Note: not formatted and
when used in %sysfunc, you don't enclose dtmonth or the b option in quotes
%let _48monthsago=%sysfunc(intnx(dtmonth,&today,-48,b)); Note: not formatted
proc sql;
    create table loans as select application_id,app_date_adjudicated
    from acaps.i10_master where app_date_adjudicated>=&_48monthsago and
    app_date_adjudicated<&_24monthsago;
quit;
```

- How to format macro variables for use in Titles or Labels

```
%let macrodate = %sysfunc(putn(&_24monthsago,datetime9.));
%put &=macrodate; formatted result is 13FEB2018
```

**Official Hashtag**

**#TimesOnMySide**

## Contact Information

SAS Communities: PaigeMiller (note: no space between the first and last name)

E-mail: pmiller1@mtb.com