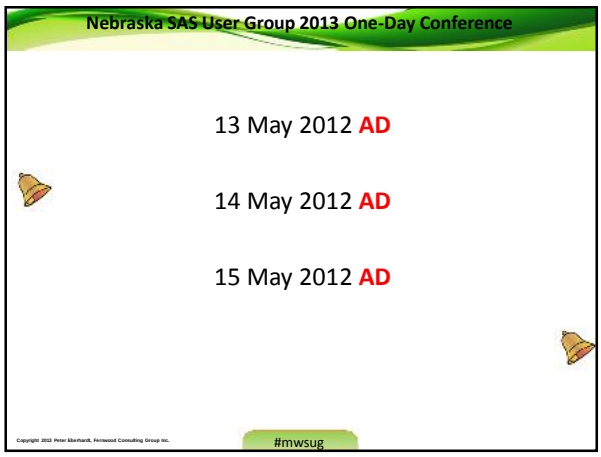
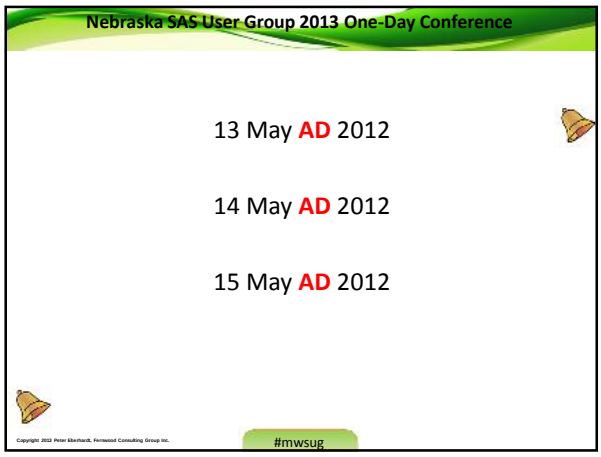
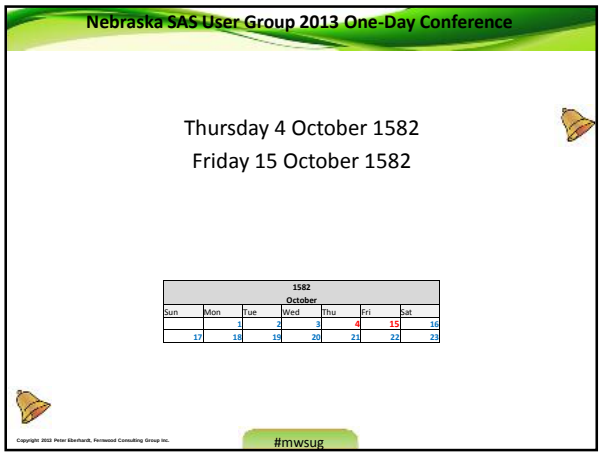
Why the Bell Tolls 108 Times: Stepping Through Time With SAS

Peter Eberhardt
Fernwood Consulting Group Inc.







Nebraska SAS User Group 2013 One-Day Conference

Pope Gregory XIII in the Papal bull

Inter gravissimas

reformed the calendar giving us what is known as the Gregorian Calendar.

This replaced the Julian Calendar which had been in effect since 45 BC.

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS date variable?

- 15 May 2012

```
1    data _null_;
2         toDay = "15MAY2012"d;
3         put toDay= /
4             toDay= yymmdd10. /
5             toDay= worddate23.;
6    run;

toDay=19128
toDay=2012-05-15
toDay=May 15, 2012
```

Date Constant

Date Display

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS date variable?

- 15 May 2012
  - The underlying data representation (**19128**) is independant of its visual representation (**2012-05-16**, **May16, 2012**, etc).
  - **What is 19128?**
    - The number of days from January 1, 1960

      1960-01-01 == 0
      1960-01-02 == 1
      1960-01-03 == 2
      …
      2012-05-15 == 19128

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS date variable?

- 15 May 2012
  - The underlying data representation (**19128**) is independant of its visual representation (**2012-05-16**, **May16, 2012**, etc).
  - **What is 19128?**
    - The number of days from January 1, 1960

1960-01-01 == 0
1959-12-31 == -1
1959-12-30 == -2
…
1592-10-15 == -137774
1592-10-04 == -137785

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS datetime variable?

- 16 May 2012 @ 9:30 am

```
7    data _null_;
8         toDay = "16MAY2012:9:30:0"dt;
9         put toDay= /
10            toDay= datetime. /
11            toDay= datetime18.3 /;
12        toDay = "16MAY2012:9:30:0.5"dt;
13        put toDay= /
14            toDay= datetime. /
15            toDay= datetime18.3;
16   run;
```

Datetime Constant

Datetime Display

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS datetime variable?

- 16 May 2012 @ 9:30 am

```
toDay=1652779800
toDay=16MAY12:09:30:00
toDay=16MAY12:09:30:00.0

toDay=1652779800.5
toDay=16MAY12:09:30:01
toDay=16MAY12:09:30:00.5
```

```
put toDay= /
      toDay= datetime. /
      toDay= datetime18.3 /
```

```
put toDay= /
        toDay= datetime. /
        toDay= datetime18.3
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS datetime variable?

- 16 May 2012 @ 9:30 am
  - **What is 1652779800?**
    - The number of seconds since January 1, 1960 @ midnight
    - Datetime variables are floating point numbers (can contain partial seconds)

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS time variable?

- 9:30

```
17   data _null_;
18        morning = "9:30"t;
19        put morning= /
20            morning= time. /
21            morning= timeampm. /;
22        morning = "9:30:00.6 PM"t;
23        put morning= /
24            morning= time. /
25            morning= timeampm12.3 /;
26   run;
```

Time Constant

Time Display

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS time variable?

- 9:30

```
morning=34200
morning=9:30:00
morning=9:30:00 AM
```

```
put morning= /
      morning= time. /
      morning= timeampm. /;
```

```
morning=77400.6
morning=21:30:01
morning=9:30:00.6 PM
```

```
put morning= /
      morning= time. /
      morning= timeampm12.3 /;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

What is a SAS time variable?

- 9:30
  - **What is 34200?**
    - The number of seconds since midnight
    - Time variables are floating point numbers (can contain partial seconds)

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

The underlying data representation

- **18613**
- **1608197400**
- **34200**

is independant of its visual representation

- **2010-12-17, December 17, 2010**
- **17DEC10:09:30:00, 17DEC10:09:30:00.0**
- **9:30:00, 9:30:00 AM**

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

SAS Formats

Visual representation of SAS date, datetime, and time variables is controlled through SAS formats

- Numerous built-in SAS formats for dates
- Fewer built-in SAS formats for datetime and time variables
- PROC format
  - Create custom formats

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

SAS Informats

- "mirror image" of formats
  - Turn a visual representation into the underlying data representation

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

SAS Informats

```
data _null_;
     input toDay          yymmdd10.  +1
           tomorrow          date9.  +1
           toDayDT      datetime16.  +1
           morning           time5.;
     put toDay= tomorrow= toDayDT= morning=;
datalines;
2010-12-17 18dec2010 17DEC10:09:30:00 09:30
2010-12-18 19dec2010 17DEC10:19:30:00 21:30
;;
run;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

SAS Informats

```
put toDay= tomorrow= toDayDT= morning=;

2010-12-17 18dec2010 17DEC10:09:30:00 09:30
2010-12-18 19dec2010 17DEC10:19:30:00 21:30
```

toDay=18613 tomorrow=18614 toDayDT=1608197400 morning=34200
toDay=18614 tomorrow=18615 toDayDT=1608233400 morning=77400

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- date()/today()
  - Returns the current date

```
data _null_;
  td = today();
  dt = date();
run;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- date()/today()
  - Useful to capture the run date for a report

```
38   %let titledate = %sysfunc(today(),yymmdd10.);
39   %put titleDate= &titledate;
titleDate= 2010-12-17


footnote1 "Produced on &titledate";
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- time()
  - Returns the current time

```
data _null_;
  now = time();
run;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- time()
  - Returns the current time

```
40   %let titledate = %sysfunc(date(),date9.);
41   %let titletime = %sysfunc(time(), timeampm.);
42   %put titleDate= &titledate titletime= &titletime;
titleDate= 17DEC2010 titletime= 9:30:34 PM

footnote1 "Produced on &titledate at &titletime";
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- datepart()
  - Returns the datepart of a datetime variable

```
data fromSQLServer;
  set sql.dataTable;
  sasDate = datepart(sqlDate);
  drop sqlDate;
  format sasDate yymmmdd10.;
run;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- timepart()
  - Returns the time part of a datetime variable

```
data fromSQLServer;
  set sql.dataTable;
  sasTime = timepart(sqlDate);
  drop sqlDate;
  format sasTime timeAMPM.;
run;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- day(dateVar)
  - Returns the day (1 – 31) of a SAS date variable
- month(dateVar)
  - Returns the month (1 – 12) of a SAS date variable
- year(dateVar)
  - Returns the year of a SAS date variable
- qtr(dateVar)
  - Returns the quarter (1 – 4) of a SAS date variable

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- mdy(mm, dd, yy)
  - Returns a SAS date from the three integer variables for month (mm), day (dd), and year (yy) input
- hms(hh, mm, ss)
  - Returns a SAS time variable for the three variables hour (hh), minute (mm), and second (ss). The seconds input can be floating point
- dhms(date, hh, mm, ss)
  - Returns a SAS date time variable from the four variables for date, hour, minute, and second

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- intck()
  - Returns the count of the number of interval boundaries between two dates, two times, or two datetime values
- intnx()
  - Increments a date, time, or datetime value by a given time interval, and returns a date, time, or datetime value.

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- Interval
  - Character value
  - Common values are (not case sensitive):
    - 'day'
    - 'week'
    - 'month'
    - 'year'
    - 'qtr'

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- Boundary
  - starting point for each interval
    - 'day'
    - 'week' sundays
    - 'month' first of the month
    - 'year' Jan 1
    - 'qtr' Jan 1, Apr 1, Jul 1, Oct 1

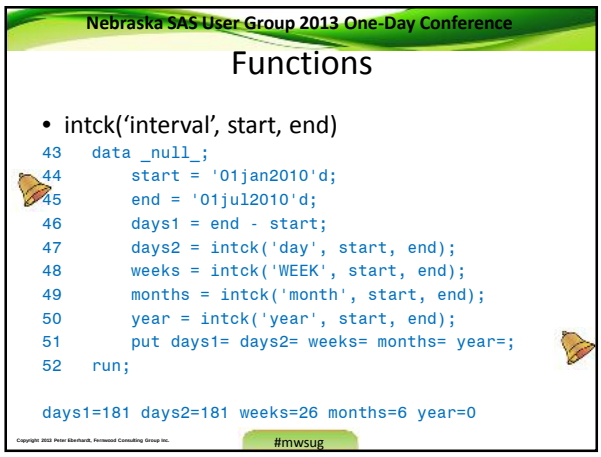
#mwsug

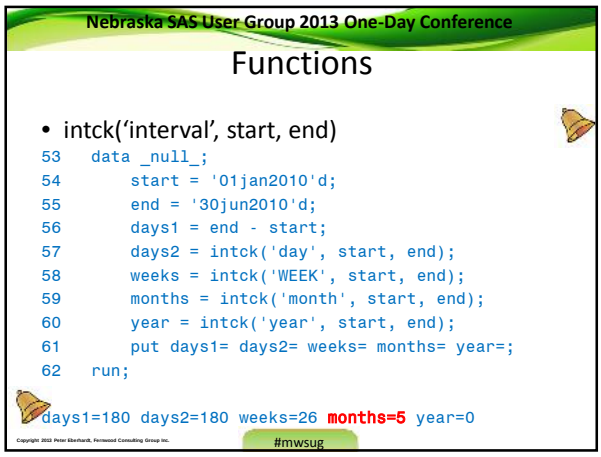
Nebraska SAS User Group 2013 One-Day Conference

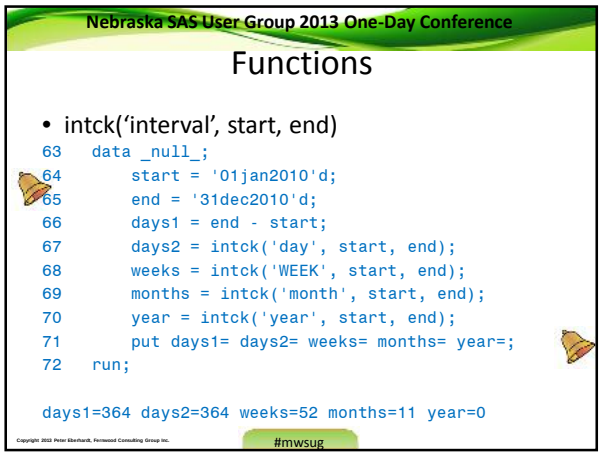
Functions

- intck('interval', start, end)
  - start: can be date, time, or datetime
  - end: must be same type as start

#mwsug







Nebraska SAS User Group 2013 One-Day Conference

Functions

- intck('interval', start, end)

```
73   data _null_;
74        start = '31dec2010'd;
75        end = '01jan2011'd;
76        days1 = end - start;
77        days2 = intck('day', start, end);
78        weeks = intck('WEEK', start, end);
79        months = intck('month', start, end);
80        year = intck('year', start, end);
81        put days1= days2= weeks= months= year=;
82   run;
```

```
days1=1 days2=1 weeks=0 months=1 year=1
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

- intnx('interval', start, increment <, *'align'*>)
  - Increment: is number of intervals to add
    - Can be negative
  - 'align': controls position of the resultant date within the interval. Optional
    - 'beginning' ('b')
      - This is the default
    - 'middle' ('m')
    - 'end' ('e')
    - 'same' ('s')

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

intnx('interval', start, increment <, *'align'*>)

```
83   data _null_;
84        start = '01jan2010'd;
85        weeks = intnx('week', start, 1);
86        weeksE = intnx('week', start, 1, 'e');
87        weeksS = intnx('week', start, 1, 's');
88        weeksM = intnx('week', start, 1, 'M');
89        put weeks= yymmdd10. weeksE= yymmdd10. weeksS=
  yymmdd10. weeksM= yymmdd10.;
90   run;
```

```
weeks=2010-01-03 weeksE=2010-01-09 weeksS=2010-01-08
 weeksM=2010-01-06
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

intnx('interval', start, increment <, *'align'*>)

```
start =2010-01-01
weeks =2010-01-03
weeksE=2010-01-09
weeksS=2010-01-08
weeksM=2010-01-06
```

increment=1

2010 January						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

intnx('interval', start, increment <, *'align'*>)

– Determine first/last day of the month

```
91   data _null_;
92        start = '11jan2010'd;
93        monthStart = intnx('month', start, 0, 'b');
94        monthEnd   = intnx('month', start, 0, 'e');
95        put monthStart= yymmdd10. monthEnd= yymmdd10.;
96   run;

monthStart=2010-01-01 monthEnd=2010-01-31
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

intnx('interval', start, increment <, *'align'*>)

– 'interval' can be compound

- 'year.10': 1 year interval where October is the start
- 'year2.4': 2 year interval where April is the start

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

intnx('interval', start, increment <, *'align'*>)

– Fiscal years

```
97  data _null_;
98  start = '11JUN2010'd;
99  yearStart = intnx('year', start, 0, 'b');
100 fiscalStartApr= intnx('year.4',  start, 1, 'b');
101 fiscalStartOct= intnx('year.10', start, 1, 'b');
102 fiscalEndApr   = intnx('year.4',  start, 1, 'e');
103 fiscalEndOct   = intnx('year.10', start, 1, 'e');
104 put yearStart= yymmdd10. /
105 fiscalStartApr= yymmdd10. fiscalEndApr= yymmdd10. /
106 fiscalStartOct= yymmdd10. fiscalEndOct= yymmdd10.;
107 run;
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Functions

intnx('interval', start, increment <, *'align'*>)

– Fiscal years

```
start = '11JUN2010'd;

yearStart=2010-01-01
fiscalStartApr=2011-04-01 fiscalEndApr=2012-03-31
fiscalStartOct=2010-10-01 fiscalEndOct=2011-09-30
```


#mwsug

Nebraska SAS User Group 2013 One-Day Conference

There and Back Again

The underlying data representation

– **18613**

– **1608197400**

– **34200**

is independant of its visual representation

– **2010-12-17, December 17, 2010**

– **17DEC10:09:30:00, 17DEC10:09:30:00.0**

– **9:30:00, 9:30:00 AM**

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

There and Back Again

Converting data representation to visual representation is controlled by FORMATS

Converting visual representation to data representation is controlled by INFORMATS

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

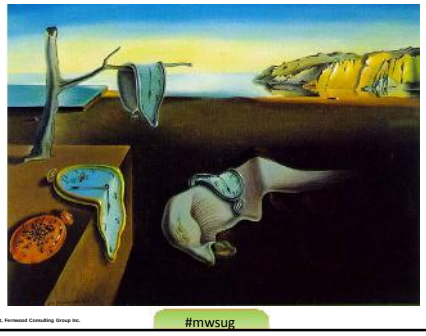
There and Back Again

SAS Functions can do it all

#mwsug

Nebraska SAS User Group 2013 One-Day Conference

Time is the thing that keeps everything from happening at once



#mwsug

Questions

Peter Eberhardt
peter@fernwood.ca

Nebraska SAS User Group 2013 One-Day Conference

In China, one of traditional activities at the Chinese New Year's Eve is people gather together at temples and wait for the bell to ring 108 times. People believe this can expel unhappiness and bring good luck in the New Year.

Why the bell is struck 108 times? It was recorded in the "Manuscript of Seven Categories" made during the Ming Dynasty that 108 times of striking the bell was equivalent to one year, as there were twelve months, twenty-four Qi (solar terms) and seventy-two Hou (12+24+72=108).

In the ancient system of timekeeping, five days were called a Hou (pentad), and six Hou made up a month, so there were seventy-two Hou periods in one year, therefore the number 108 was calculated according to these time measurements

#mwsug

Nebras onference