SESUG Paper 228-2019

Utilizing SAS Functions to Generate Accurate Adherence Notifications for Clinical Trials

Lishu Zhang, Jiu Zhao, Wenjun He, the EMMES Company LLC.

ABSTRACT

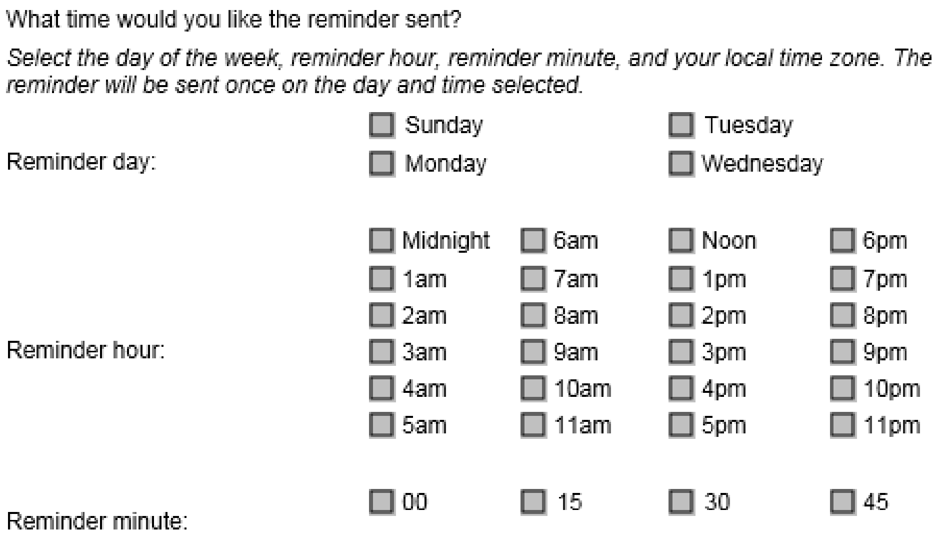
For clinical trial studies, it is critical to keep close contact with participants after main study procedures are administered or during follow-up phases. Furthermore, it could be challenging to monitor participants and reinforce their adherence to medication taking, laboratory sample collection, or periodic doctor's visits outside clinical settings. Currently mobile technology has emerged to play an essential role by sending out automated simple reminders and notifications to maintain communication among participants, study coordinators and principal investigators. Accurate notifications can significantly improve study compliance and treatment efficacy.

This paper will present a real case of how SAS® 9.4 can be utilized as a powerful and flexible tool to create a notification schedule that can be used to remind subjects to take dry blood samples at home for a clinical trial study. Notification dates and time were chosen based on the individual participants' given preferences. The poster will demonstrate step-by-step on how to apply LAG and INTNX functions to generate this type of notification using diagrams and actual programming code. It will also provide examples of how to logically implement changes of participant preferences during the course of study without jeopardizing the scientific rationale and study schedule.

INTRODUCTION

In clinical studies, an individual subject's drop-off rate increases significantly in the first few months after discharge from the hospital. This indicates a poor adherence to certain testing that are recommended for a long testing period after discharge. A simple and effective notification needs to be in placed to remind subjects to take those tests at home with lost cost. We recently worked on a clinical study designed to assess whether simple weekly reminders via phone or email sent to high-risk subjects to self-collect dried blood samples at home would improve study compliance and boost health outcome. The SAS programming team using SAS 9.4/M4 developed the notification schedule. Subjects are required to fill in a form to set the timeframe for receiving reminders or notifications. They will specify the day of the week (Sunday-Wednesday) and the time of the day that they prefer to be reminded right after enrollment. Part of the data collection form is shown in Figure 1.

Figure 1. Part of the reminder form



What time would you like the reminder sent?

Select the day of the week, reminder hour, reminder minute, and your local time zone. The reminder will be sent once on the day and time selected.

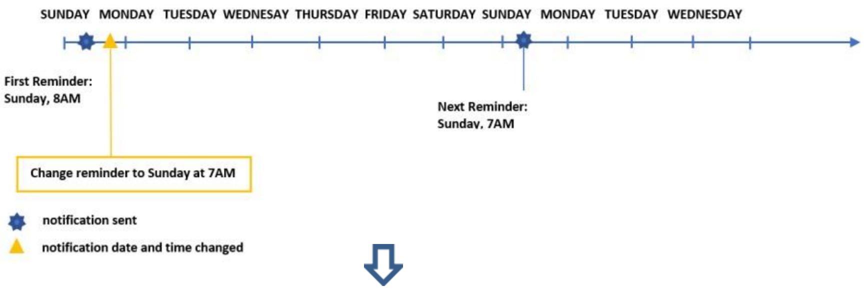
Reminder day:
☐ Sunday
☐ Monday
☐ Tuesday
☐ Wednesday

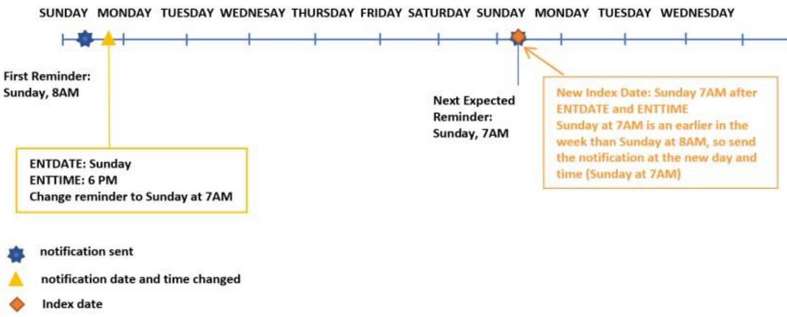
Reminder hour:
☐ Midnight
☐ 1am
☐ 2am
☐ 3am
☐ 4am
☐ 5am
☐ 6am
☐ 7am
☐ 8am
☐ 9am
☐ 10am
☐ 11am
☐ Noon
☐ 1pm
☐ 2pm
☐ 3pm
☐ 4pm
☐ 5pm
☐ 6pm
☐ 7pm
☐ 8pm
☐ 9pm
☐ 10pm
☐ 11pm

Reminder minute:
☐ 00
☐ 15
☐ 30
☐ 45

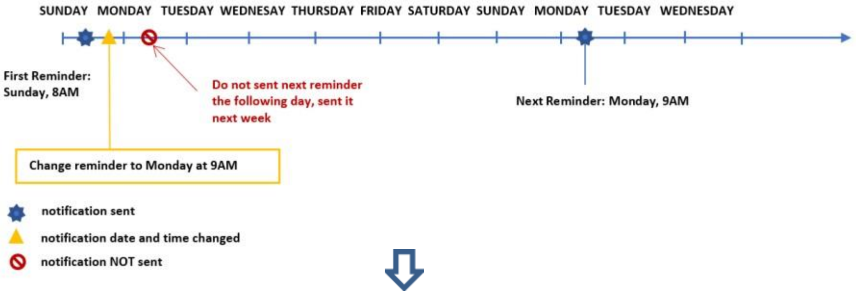
Once the form is filled, the first reminder will be sent within 7 days after enrollment on the day and time selected. Subsequent reminders will be sent once a week until study completion. Subjects may elect to change their preferred notification day and time at any time of the study. If the subject changes preference, the next reminder will be sent according to the following rules (see diagrams below):

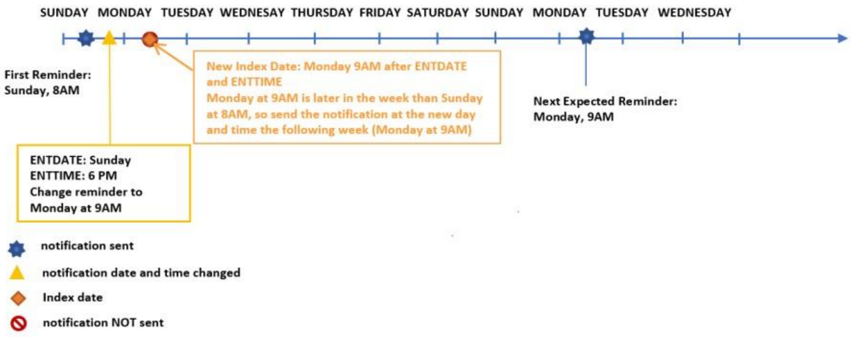
Rule 1: If new day and time are before the last notification sent, then send the next notification at the new day and time:


SUNDAY
MONDAY
TUESDAY
WEDNESAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY
MONDAY
TUESDAY
WEDNESDAY
First Reminder: Sunday, 8AM
Next Reminder: Sunday, 7AM
Change reminder to Sunday at 7AM
notification sent
notification date and time changed

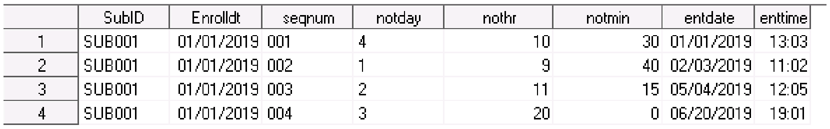

SUNDAY
MONDAY
TUESDAY
WEDNESAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY
MONDAY
TUESDAY
WEDNESDAY
First Reminder: Sunday, 8AM
Next Expected Reminder: Sunday, 7AM
New Index Date: Sunday 7AM after ENTDATE and ENTTIME
Sunday at 7AM is an earlier in the week than Sunday at 8AM, so send the notification at the new day and time (Sunday at 7AM)
ENTDATE: Sunday
ENTTIME: 6 PM
Change reminder to Sunday at 7AM
notification sent
notification date and time changed
Index date

Rule 2: If new day and time are after the last notification sent, then send the next notification at the new time + 7 day


SUNDAY
MONDAY
TUESDAY
WEDNESAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY
MONDAY
TUESDAY
WEDNESDAY
First Reminder: Sunday, 8AM
Do not sent next reminder the following day, sent it next week
Next Reminder: Monday, 9AM
Change reminder to Monday at 9AM
notification sent
notification date and time changed
notification NOT sent


SUNDAY
MONDAY
TUESDAY
WEDNESAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY
MONDAY
TUESDAY
WEDNESDAY
First Reminder: Sunday, 8AM
New Index Date: Monday 9AM after ENTDATE and ENTTIME
Monday at 9AM is later in the week than Sunday at 8AM, so send the notification at the new day and time the following week (Monday at 9AM)
Next Expected Reminder: Monday, 9AM
ENTDATE: Sunday
ENTTIME: 6 PM
Change reminder to Monday at 9AM
notification sent
notification date and time changed
Index date
notification NOT sent

An example of a subject's entered reminder dataset dumdsn is illustrated in Figure 2:



	SubID	Enrolldt	seqnum	notday	nothr	notmin	entdate	enttime
1	SUB001	01/01/2019	001	4	10	30	01/01/2019	13:03
2	SUB001	01/01/2019	002	1	9	40	02/03/2019	11:02
3	SUB001	01/01/2019	003	2	11	15	05/04/2019	12:05
4	SUB001	01/01/2019	004	3	20	0	06/20/2019	19:01

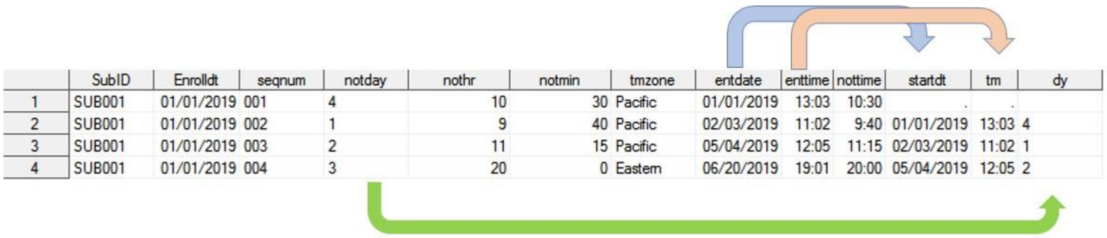
Figure 2. Dataset dumdsn

Subid is the Subject ID. Enrolldt is the Date of Enrollment into the study. Seqnum is the Sequential Number of each preference change. Notday is the Day of the Week the subject selected to be notified: 1- Sunday, 2-Monday, 3-Tuesday, 4-Wednesday. Nothr and notmin are the Time of the Day the notification should be sent. Entdate and enttime present the day and time when a new notification preference is entered. As shown in Figure 2, Subject 001 chose the initial preferred reminder to be sent on every Wednesday at 10:00 am (Seqnum 001). Then, on 03Feb2019, the subject changed it to Sunday at 9 am (Seqnum 002). On 04May2019, the subject changed the preference again to Monday at 11:15 am (Seqnum 003), until on 20Jun2019, the subject changed the final notification schedule to Tuesday at 8 pm (Seqnum 004). The challenge for SAS programmer is to generate a listing of dates and times that could satisfy four different preferences within each window. The following paper will present step by step, on how the SAS code was created to generate this complex listing.

The first step taken was to use LAG function to align the previous reminder day and time on the same observation with the new reminder schedule entered:

```
data dsn1 lastrec; set
    dumdsn;
    by subid seqnum;

    nottime = input(compress(nothr||':'||notmin), time5.); startdt = lag(entdate);
    tm = lag(nottime);
    dy = lag(notday);

    if first.subid then do; call
        missing(startdt); call missing(tm);
        call missing(dy); end;

    format startdt mmddyy10. tm nottime hhmm5.; output dsn1;
    if last.subid then output lastrec;
run;
```

The derived dataset dsn1 generated form the above code is presented in Figure 3. The previous schedule is now preserved on the same row with the new schedule. Variable **startdt**, **tm** and **dy** hold the previous notification values, versus variables **notday** and **nottime** show the new selected values**.**



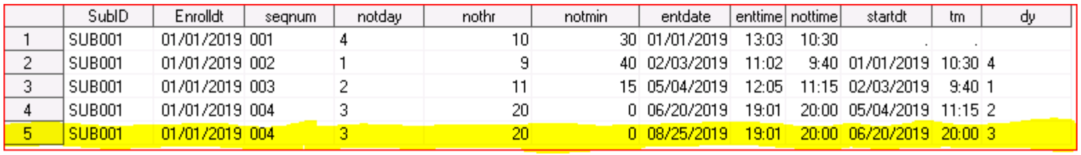
	SubID	Enrolldt	seqnum	notday	nothr	notmin	tmzone	entdate	enttime	nottime	startdt	tm	dy
1	SUB001	01/01/2019	001	4	10	30	Pacific	01/01/2019	13:03	10:30	.	.	
2	SUB001	01/01/2019	002	1	9	40	Pacific	02/03/2019	11:02	9:40	01/01/2019	13:03	4
3	SUB001	01/01/2019	003	2	11	15	Pacific	05/04/2019	12:05	11:15	02/03/2019	11:02	1
4	SUB001	01/01/2019	004	3	20	0	Eastern	06/20/2019	19:01	20:00	05/04/2019	12:05	2

Figure 3. Dataset dsn1

The second step taken was to create a new last observation which allows extension of the final preferred schedule to today's date. Below is code for Step 2:

```
/*to set the last record end date as today (if subject remains in study)*/
data lastrec1;
   set lastrec;

   startdt = entdate;
   dy = notday;
   tm = nottime;
   entdate = today();

   format entdate mmddyy10.;
run;

data dsn2;
   set dsn1 lastrec1;
run;

proc sort;
   by subid entdate;
run;
```

Figure 4 below shows the derived dataset dsn2 generated by this step. The highlighted last row is the new record created to carry the last change to the current date (08/25/2019).



	SubID	Enrolldt	seqnum	notday	nothr	notmin	entdate	enttime	nottime	startdt	tm	dy
1	SUB001	01/01/2019	001	4	10	30	01/01/2019	13:03	10:30	.	.	
2	SUB001	01/01/2019	002	1	9	40	02/03/2019	11:02	9:40	01/01/2019	10:30	4
3	SUB001	01/01/2019	003	2	11	15	05/04/2019	12:05	11:15	02/03/2019	9:40	1
4	SUB001	01/01/2019	004	3	20	0	06/20/2019	19:01	20:00	05/04/2019	11:15	2
5	SUB001	01/01/2019	004	3	20	0	08/25/2019	19:01	20:00	06/20/2019	20:00	3

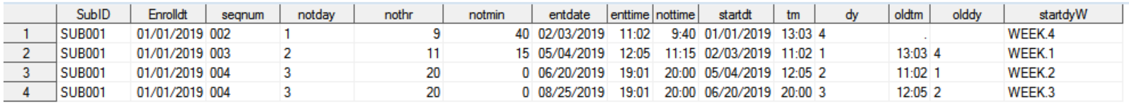
Figure 4. Dataset dsn2

The third step was to utilize LAG function one more time so that three schedules (one current, two previous) could align on the same observation for comparison:

```
data dsn3;
   set dsn2;
   by subid entdate;

   oldtm = lag(tm);
   olddy = lag(dy);
   oldentdt = lag(entdate);
   oldenttm = lag(enttime);

   if first.subid then delete;

   /*create the interval argument for INTNX function*/
   startdyW=compress('WEEK.'||dy);

   format oldentdt mmddyy10. oldenttm oldtm hhmm5.;
run;

proc sort;
   by subid startdt;
run;
```

At this step, a new variable named startdyW was created by compressing word “WEEK” with variable dy. The formed value WEEK4 means the fourth day of a week, and WEEK1 means the first day of a week, etc. In SAS, a week starts with Sunday as Day 1 and ends with Saturday as Day 7. Variable startdyW will be used later as the interval argument for INTNX function. Figure 5 demonstrated the derived dataset of dsn3.

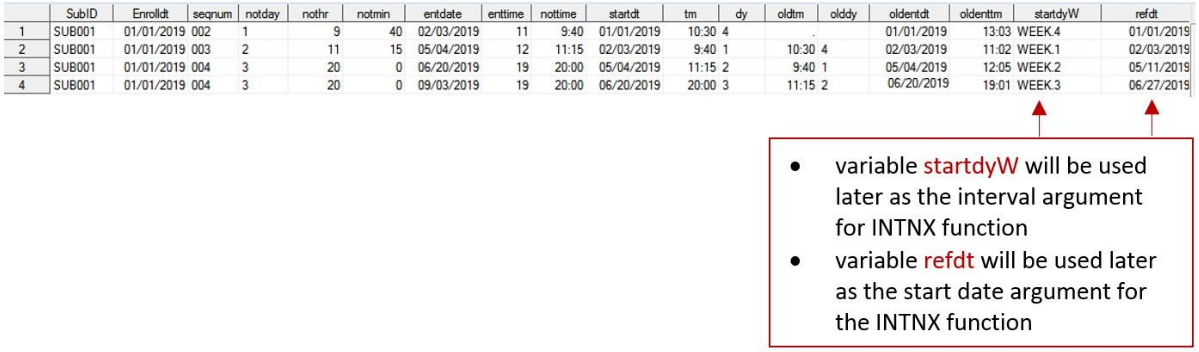
Figure 5. Dataset dsn3



	SubID	Enrolldt	seqnum	notday	nothr	notmin	entdate	enttime	nottime	startdt	tm	dy	oldtm	olddy	startdyW
1	SUB001	01/01/2019	002	1	9	40	02/03/2019	11:02	9:40	01/01/2019	13:03	4	.		WEEK.4
2	SUB001	01/01/2019	003	2	11	15	05/04/2019	12:05	11:15	02/03/2019	11:02	1	13:03	4	WEEK.1
3	SUB001	01/01/2019	004	3	20	0	06/20/2019	19:01	20:00	05/04/2019	12:05	2	11:02	1	WEEK.2
4	SUB001	01/01/2019	004	3	20	0	08/25/2019	19:01	20:00	06/20/2019	20:00	3	12:05	2	WEEK.3

The fourth step taken was to use all information on each observation to define the begin date for new schedule based on Rule 1 and Rule 2. If the new selected day and time are a later day and time, as compared to the previous reminder, then a 7 will be added to the enter date as the new schedule begin date. Otherwise, if the new day and time are an earlier time point, then the begin date is the enter date. Step 4 code is listed below and the derived dataset dsn4 is shown in Figure 6:

```
/*calculate start date for each time the notification record is changed*/
data dsn4; set
    dsn3;
    by subid startdt;

    if first.subid then refdt = startdt;
    else if dy > olddy then refdt = startdt + 7;                          /*Rule 2*/
    else if dy = olddy and tm > oldtm then refdt = startdt + 7; /*Rule 2*/ else refdt = startdt;      /*Rule 1*/

    format refdt mmddyy10.;run;
```



	SubID	Enrolldt	seqnum	notday	nothr	notmin	entdate	enttime	nottime	startdt	tm	dy	oldtm	olddy	oldentdt	oldenttm	startdyW	refdt
1	SUB001	01/01/2019	002	1	9	40	02/03/2019	11	9:40	01/01/2019	10:30	4	.		01/01/2019	13:03	WEEK.4	01/01/2019
2	SUB001	01/01/2019	003	2	11	15	05/04/2019	12	11:15	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019
3	SUB001	01/01/2019	004	3	20	0	06/20/2019	19	20:00	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019
4	SUB001	01/01/2019	004	3	20	0	09/03/2019	19	20:00	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019

- variable startdyW will be used later as the interval argument for INTNX function
- variable refdt will be used later as the start date argument for the INTNX function

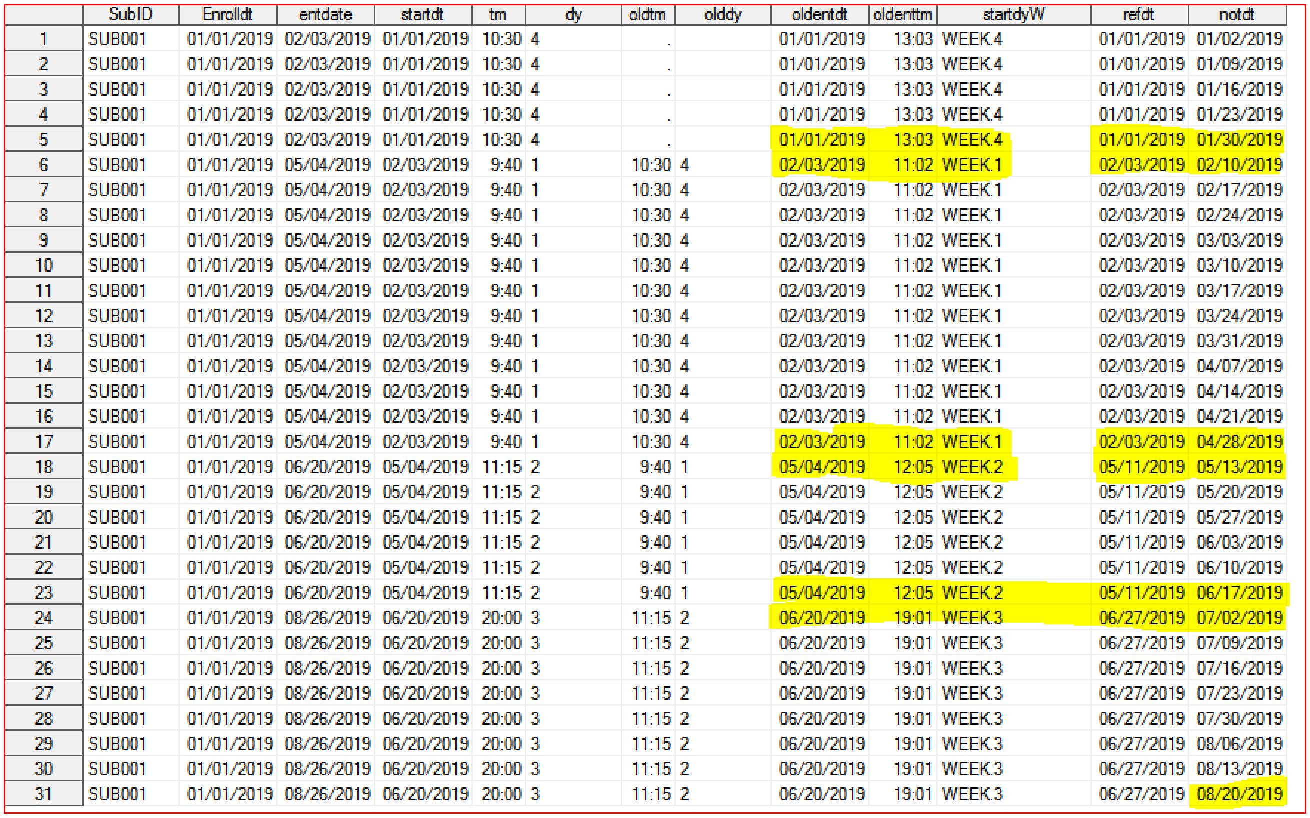
Figure 6. Dataset dsn4

Now we have all elements ready for putting into INTNX function to generate the final schedule. The syntax for INTNX function is: **INTNX**(*interval* <*multiple*><*.shift-index*>, *start-from*, *increment* <, '*alignment*'>). In this case, we only need to specify *interval* argument (**startdyW** variable), *start-from* argument (**refdt** variable) and *increment* argument (the nth week, represented by **&i** in the macro). A macro program is developed to iterate the date generation. An arbitrary number of 260 weeks is chosen conservatively based on the length of study designed. Below is the code for macro named “gendt”:

```
/*macro to generate notification dates based on DTR*/
%macro gendt;
    data dsn5;
           set dsn4;
        by subid startdt;

    /*260 weeks should be long enough to cover the entire course of study,
          if not enough, please re-adjust*/
             %do i=0 %to 260;
                notdt = intnx(startdyW, refdt, &i.);
                if refdt < notdt < entdate or
                      (notdt = refdt and oldenttm < oldtm) then output;%end;
             format notdt mmddyy10.; run;
    %mend gendt;
    %gendt;
```

The final dataset dsn5 was generated with notification dates and times matched the subject's multiple preferences. As illustrated in Figure 7, the subject was enrolled on 01/01/2019. The initial notification was set on each Wednesday at 10:30 am. The first reminder was set on the first Wednesday 01/02/2019 within the 7-day window after the enrollment. The first change was entered on 02/03/2019. The change was from every Wednesday at 10:30 am to Sunday at 9:40 am. Since Sunday is an earlier day of the week, Rule 1 was applied. The reference date (refdt) was the date the change was entered (on 02/03/2019). Both the second and third changes were a later day of the week; therefore, Rule 2 was applied. The reference dates were a week later after changes were entered. Please note, for the first change, the first Sunday after 01/30/2019 is 02/03/2019. However, since the new schedule was entered on 02/03/2019 at 11:02 am, a later time than the previous notification time at 9:40 am, the following Sunday 02/10/2019 became the first reminder created for this new schedule.

Figure 7. Dataset dsn5



	SubID	Enrolldt	entdate	startdt	tm	dy	oldtm	olddy	oldentdt	oldenttm	startdyW	refdt	notdt
1	SUB001	01/01/2019	02/03/2019	01/01/2019	10:30	4	.		01/01/2019	13:03	WEEK.4	01/01/2019	01/02/2019
2	SUB001	01/01/2019	02/03/2019	01/01/2019	10:30	4	.		01/01/2019	13:03	WEEK.4	01/01/2019	01/09/2019
3	SUB001	01/01/2019	02/03/2019	01/01/2019	10:30	4	.		01/01/2019	13:03	WEEK.4	01/01/2019	01/16/2019
4	SUB001	01/01/2019	02/03/2019	01/01/2019	10:30	4	.		01/01/2019	13:03	WEEK.4	01/01/2019	01/23/2019
5	SUB001	01/01/2019	02/03/2019	01/01/2019	10:30	4	.		01/01/2019	13:03	WEEK.4	01/01/2019	01/30/2019
6	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	02/10/2019
7	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	02/17/2019
8	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	02/24/2019
9	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	03/03/2019
10	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	03/10/2019
11	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	03/17/2019
12	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	03/24/2019
13	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	03/31/2019
14	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	04/07/2019
15	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	04/14/2019
16	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	04/21/2019
17	SUB001	01/01/2019	05/04/2019	02/03/2019	9:40	1	10:30	4	02/03/2019	11:02	WEEK.1	02/03/2019	04/28/2019
18	SUB001	01/01/2019	06/20/2019	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019	05/13/2019
19	SUB001	01/01/2019	06/20/2019	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019	05/20/2019
20	SUB001	01/01/2019	06/20/2019	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019	05/27/2019
21	SUB001	01/01/2019	06/20/2019	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019	06/03/2019
22	SUB001	01/01/2019	06/20/2019	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019	06/10/2019
23	SUB001	01/01/2019	06/20/2019	05/04/2019	11:15	2	9:40	1	05/04/2019	12:05	WEEK.2	05/11/2019	06/17/2019
24	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	07/02/2019
25	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	07/09/2019
26	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	07/16/2019
27	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	07/23/2019
28	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	07/30/2019
29	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	08/06/2019
30	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	08/13/2019
31	SUB001	01/01/2019	08/26/2019	06/20/2019	20:00	3	11:15	2	06/20/2019	19:01	WEEK.3	06/27/2019	08/20/2019

CONCLUSION

SAS INTNX function is a very useful function to return any date from a specific date given a number of time units. Time units can be a date, time or datetime. In addition, the function beautifully auto-calculates leap years for users. The power and flexibility of this function provide us a perfect tool to generate date schedule as presented in this clinical trial study. The notification schedule provided to the clinical study team greatly helped the subjects to collect samples on time and improved compliance and effectiveness of the clinical trial.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at Lishu

Zhang
The Emmes Company LLC
lishu8@gmail.com