Sleep Diary Data - Formatting and Outcomes Computation for SHUTi for Researchers

Kelly Shaffer, with edits and review by Lee Ritterband & Katharine E Daniel - UVA CBHT

Compiled for GitHub April 27, 2023

Before you begin...

We ask that you first go through the "README" on the SHUTi for Researcher's github page and read through the PDF document of the RMarkdown code. Focus on the directions between the 'chunks' of syntax code, and look out for places where you will need to modify the code based on your data and files (denoted with "##EDITING (MAY BE) REQUIRED" tags).

For those interested in learning more about the function of individual lines of code, please contact the SHUTi for Researchers team, who can share a more extensively commented version of the general syntax.

We believe that the code we share below should substantially reduce the time needed to manage your SHUTi diary data and compute your diary outcomes. That said, this still can be a time-consuming and tedious process with some trial-and-error involved!

##Version information

If you choose to download this file onto your personal machine, please take care to periodically refer back to the project's github page at https://github.com/BHT/SHUTiForResearchers for potential updates to the below syntax. <<You are currently viewing _v2 of the syntax>>. A full description of updates between each version will be saved as a log on the project's github page. In addition, updates from the last version to the current version will be annotated throughout this syntax.

V2 was built using R version 4.2.3 (2023-03-15 ucrt) – "Shortstop Beagle"

R Markdown, R, and RStudio

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

To use this file, you will need to download R and RStudio (free version is fine). First, install R. R can be downloaded for installation here: https://cran.rstudio.com/ Next, install RStudio. RStudio can be downloaded for installation here: https://www.rstudio.com/products/rstudio/download/

To execute syntax in the R Markdown window, click in a line of code within a syntax "chunk" (lines between the "') and press CTRL + Enter (Windows) or Command + Return (Mac OS). You can also run full "chunks" of code at a time by clicking the green arrow in the top right corner of a chunk ("Run Current Chunk" button). You can also run multiple chunks, or even the full script, at once by selecting options from the "Run" menu at the top right of this syntax window. Until you have edited the syntax and know it works for your files as intended, it is not recommended to run more than one command at a time, but this can be a useful strategy once you have finished editing your syntax file.

R is an open-source data analysis and programming software. Many resources and tutorials can be found online - including at https://education.rstudio.com/. Because it is open-source, many errors and suggestions about code can be found by searching the issue online.

Document Overview

This R Markdown document & accompanying syntax was developed by the SHUTi for Researchers team to ensure all users compute diary variables in the same way. You will at times need to edit the syntax where directed; however, please DO NOT alter the formulas or reuse syntax without permission of the SHUTi for Researchers team.

Companion Manuscript

Our team is developing a companion manuscript to further describe these data management processes. We will update this syntax with the citation and manuscript details and post the manuscript to the github when available.

Citations

For any presentations or published works that leverages files shared here, please cite this github project page (a companion paper is forthcoming):

Shaffer, K.M., Daniel, K.E., & Ritterband, L.M. (2023). Sleep Diary Data - Formatting and Outcomes Computation for SHUTi for Researchers (Version 2.0) [Source code]. https://github.com/BHT/SHUTiForResearchers

(Companion manuscript citation forthcoming)

Data Download and R Environment Setup

Download your study's sleep diary data with the following directions: (1) Log into your SHUTi for Researchers project account as a clinician by going to shuti.org and clicking the Sign In button. (2) Once in, at the top left, there's the Export Wasabi pages button. (3) A drop-down menu will appear. Click on the Study Data Export link and select the Daily Sleep Diaries to get the sleep diaries for all participants. Be advised that any test participants you created may be included, depending on whether you selected the "Include in Reports" option under Additional Notes in the User Info tab. (4) Click Export Data and it will download as a .csv Excel file. Save this file to a folder where you would like your data to be housed. Save the file as "Diaries data_export.csv". (5) Repeat this process to download the Events .csv Excel file. Save this file in the same folder as in step 4 above, and name this file "Events data_export.csv".

Now you will set the folder from steps 4 & 5 above as your 'Working Directory.' This is the location where the R syntax will 'look' for imported data files and 'print' your exported data files.

##EDITING REQUIRED: Set your Working Directory by editing the path below to match the file path that leads to the folder where your data is stored, delete the '#' at the start of the line, and executing the setwd() syntax. Alternatively, you can go to Session > Set Working Directory > Choose Directory to select the folder through 'point-and-click.' Check your Working Directory by executing the getwd() syntax below after deleting the '#' at the start of the line.

```
#setwd("/File/Path/to/FolderWithData")

#getwd()
```

Next, you'll need to make sure you have the R packages that we use in this syntax installed and downloaded.

##EDITING MAY BE REQUIRED: If you have already installed a package on your computer, you don't need to execute the install.packages() command again. If you have not installed a package before, delete the

'#' at the start of the line before attempting to execute the syntax. You do need to execute the library() syntax each time you open your R environment, because this loads the package into your R environment and allows your syntax to pull from it.

```
#install.packages("tidyverse")
#install.packages("lubridate")
#install.packages("Rfast")
#install.packages("imputeTS")

library(tidyverse)
library(lubridate)
library(Rfast)
library(imputeTS)
```

Import the sleep diary data into your R environment.

```
data<-read_csv("Diaries data_export.csv")
```

Now we will rename columns to be easier to use and understand during analyses. Don't change these names.

```
data<-data %>%
  rename(diaryid="Diary ID",
userid="User ID",
PID="PID",
caldate="Calendar Date",
time.bed="Bed At",
time.sleep="Sleep At",
utctime.bed="Bed At (UTC)",
utctime.sleep="Sleep At (UTC)",
sol="Sleep Onset Latency (min)",
wakecount="Interruption Count",
waso="Total Interruption Duration (min)",
time.wake="Final Wake At",
time.rise="Rise At",
utctime.wake="Final Wake At (UTC)",
utctime.rise="Rise At (UTC)",
sleepqual="Sleep Quality",
napduration="Total Nap Duration (min)",
drinkcount="Alcoholic Drink Count",
time.lastdrink="Last Alcoholic Drink At",
utctime.lastdrink="Last Alcoholic Drink At (UTC)",
sleepmeds="Medications for Sleep",
diarytype="Phase",
created="Created At",
updated="Updated At",
timezone="Reference Time Zone",
ema="Early Morning Awakening (min)",
tib="Time in Bed (min)",
tst="Total Sleep Time (min)",
se="Sleep Efficiency",
sleepwindow="Sleep Window ID",
assessment="Assessment")
```

Now, we'll make sure that the time and date data is formatted the right way for us to use for computations.

```
data$utctime.bed<-str_replace(data$utctime.bed, "UTC", "")
data$utctime.sleep<-str_replace(data$utctime.sleep, "UTC", "")
data$utctime.wake<-str_replace(data$utctime.wake, "UTC", "")
data$utctime.rise<-str_replace(data$utctime.rise, "UTC", "")

data$utctime.bed<-ymd_hms(data$utctime.bed, tz = "UTC")
data$utctime.sleep<-ymd_hms(data$utctime.sleep, tz = "UTC")
data$utctime.rise<-ymd_hms(data$utctime.rise, tz = "UTC")
data$utctime.wake<-ymd_hms(data$utctime.wake, tz = "UTC")
```

Data Cleaning

##----------

Round 1 Data Cleaning

##----------

First, you need to ensure that only true participants are included in the dataset. This includes removing any 'test participants' you may have created. You can skip this section if ALL the data in your excel file are from your participants.

##EDITING REQUIRED: There are two syntax options for you to use below. The first example (lines 171-172) would work best if all your 'real participants' have a participant ID greater than a certain number, and all 'test users' have an ID lower than a certain number. This example syntax selects only participants with PIDs of 25 or higher. The second option (starting line 174) is an example of how to select only a certain set of PIDs. Select an option that will work well for your data, delete the #, edit, and execute the syntax.

```
#data<- data %>%
#  filter(PID >= 25)

#data<- data %>%
#  filter(PID == 66 |
#PID == 67 |
#PID == 69 |
#PID == 71 |
#PID == 83 |
#PID == 94 |
#PID == 99 |
#PID == 112 |
#PID == 127 |
#PID == 139 |
#PID == 148 |
#PID == 152)
```

Next, we'll need to fix any invalid diary times. Start by computing total time in bed to help identify where there might be data entry errors (tib.tot). For our assessments, we compute time in bed as the total sleep opportunity, i.e., rise time - time trying to fall asleep. Compute this as tib.h (time in bed, in hours). The 'tib' variable in the downloaded dataset is computed from the original patient-entered data and will not be used once we clean the data.

```
##V2: updated formulas to print as number (of hours) without formatting

data<- data %>%
  mutate(tib.tot = difftime(utctime.rise, utctime.bed, units = "hours" ))

data<- data %>%
  mutate(tib.h = difftime(utctime.rise, utctime.sleep, units = "hours"))
```

Next, we'll check for common data entry errors. Specifically, participants commonly confuse AM & PM and dates while entering diary times. We continue to work on SHUTi to reduce possible data entry errors, but the syntax below will help you identify and rectify errors that participants continue to commonly make.

This chunk of syntax addresses that participants will sometimes confuse the date, particularly when entering midnight (i.e., 0 on 24 hour clock = 12 AM on dawning day, not 12 AM on yesterday). We only check for date errors for bedtimes and sleep times, which can either occur the 'day before' the diary calendar date or on the same day as the diary calendar date (i.e., 10:30 PM on Monday, Jan 1 or 1 AM on Tuesday, Jan 2). Although it is technically possible for date errors to occur for wake and rise times, these have not been previously detected by our team. This is because wake and rise times generally occur only on the calendar date of the diary (i.e., 6 AM on Tuesday, Jan 2 - in this case, the diary would be tagged with caldate = "Tuesday, 1/2").

This chunk of syntax fixes when the date has (presumably) been listed incorrectly for the time to bed. We assume that users whose diary data indicate they were in bed longer than 24 hours with a bed time of 3 AM (i.e., 3 on 24-hour clock) or earlier on the day before the calendar day of the diary should have the date as the same calendar day of the diary. For example, a user indicating that they went to bed at 2 AM on Monday, Jan 1, who started trying to go to sleep at 2:30 AM on Tuesday, Jan 2, and who reported waking and rising at 7 AM on Tuesday, Jan 2. This syntax will update the time in bed to 2AM on Tuesday, Jan 2.

Throughout our data cleaning, we also create variables to help you keep record of which diaries were modified.

```
data<- data %>%
  mutate(utctime.bed.fix = ifelse(time.bed <= hours(3) & tib.tot >= 24,
                                  utctime.bed + hours(24), utctime.bed))

data$utctime.bed.fix<-as_datetime(data$utctime.bed.fix)

data<- data %>%
  mutate(bed.fix1.YN = ifelse(time.bed <= hours(3) & tib.tot >= 24,
                                 "Yes", "No"))
```

This chunk of syntax fixes when the date has been listed incorrectly for the time starting to try to fall asleep. We assume that users whose diary data indicate they were in bed longer than 24 hours with a sleep time of 3 AM (i.e., 3 on 24-hour clock) or earlier on the day before the calendar day of the diary should have the date as the same calendar day of the diary. For example, a user indicating they went to bed at 2 AM on Monday, Jan 1, who started trying to go to sleep at 2:30 AM on Monday, Jan 1, and who reported waking and rising at 7 AM on Tuesday, Jan 2. This syntax would update the sleep time to 2:30 AM on Tuesday, Jan 2.

```
data<- data %>%
  mutate(utctime.sleep.fix = ifelse(time.sleep <= hours(3) & tib.h >= 24,
                                    utctime.sleep + hours(24),
                                    utctime.sleep))

data$utctime.sleep.fix<-as_datetime(data$utctime.sleep.fix)
```

```
data<- data %>%
  mutate(sleep.fix1.YN = ifelse(time.sleep <= hours(3) & tib.h >= 24,
                                "Yes", "No"))
```

This next chunk of syntax fixes where users have likely AM/PM errors in wake times.

We assume that users whose inputted diary data indicate they were in bed longer than 12 hours with a stated wake time of 3 PM (i.e., 15 on 24-hour clock) or later meant to input the time as AM. For example, someone inputting a bedtime of 10 PM on Monday, Jan 1, who started trying to fall asleep at 10:30 PM on Monday, Jan 1, who reported waking and rising at 5 PM on Tuesday, Jan 2. This syntax would update the wake time to 5 AM on Tuesday, Jan 2.

```
data<- data %>%
  mutate(utctime.wake.fix = ifelse(time.wake >= hours(15) & tib.tot >= 12,
                                   utctime.wake - hours(12),
                                   utctime.wake))

data$utctime.wake.fix<-as_datetime(data$utctime.wake.fix)

data<- data %>%
  mutate(wake.fix.YN = ifelse(time.wake >= hours(15) & tib.tot >= 12,
                              "Yes", "No"))
```

Similarly, this chunk of syntax fixes where users have likely AM/PM errors in rise times.

We assume that users whose inputted diary data indicate they were in bed longer than 12 hours with a stated rise time of 3 PM (i.e., 15 on 24-hour clock) or later meant to input the time as AM. For example, someone inputting a bedtime of 10 PM on Monday, Jan 1, who started trying to fall asleep at 10:30 PM on Monday, Jan 1, who reported waking and rising at 5 PM on Tuesday, Jan 2. This syntax would update the rise time to 5 AM on Tuesday, Jan 2.

```
data<- data %>%
  mutate(utctime.rise.fix = ifelse(time.rise >= hours(15) & tib.tot >= 12,
                                   utctime.rise - hours(12),
                                   utctime.rise))

data$utctime.rise.fix<-as_datetime(data$utctime.rise.fix)

data<- data %>%
  mutate(rise.fix.YN = ifelse(time.rise >= hours(15) & tib.tot >= 12,
                              "Yes", "No"))
```

We'll now update our total time in bed variable to help identify remaining invalid diary entries. We'll also update our assessment time in bed variable to help identify invalid diaries now, too.

```
##V2: updated formulas to print as number (of hours) without formatting

data<- data %>%
  mutate(tib.tot = difftime(utctime.rise, utctime.bed, units = "hours" ))

data<- data %>%
  mutate(tib.h = difftime(utctime.rise, utctime.sleep, units = "hours"))
```

##----------

Round 2 Data Cleaning

##----------

Now for our second round of cleaning updates. This chunk of syntax fixes where users are likely to have mixed-up AM / PM.

We assume that users whose inputted diary data which indicates that they were in bed longer than 12 hours with a stated bed time between 5AM and 3PM (i.e., 15 on 24-hour clock) entered AM / PM incorrectly. For example, someone inputting a bedtime of 1 PM on Monday, Jan 1, who started trying to fall asleep at 2 AM on Tuesday, Jan 2, with a wake and rise time of 6 AM on Tuesday, Jan 2. This syntax would update the bedtime to 1 AM on Tuesday, Jan 2.

```
data<- data %>%
  mutate(utctime.bed.fix = ifelse(c(time.bed <= hours(15) & time.bed >= hours(05))
                                  & tib.tot >= 12,
                                  utctime.bed.fix + hours(12),
                                  utctime.bed.fix))

data$utctime.bed.fix<-as_datetime(data$utctime.bed.fix)
data<- data %>%
  mutate(bed.fix2.YN = ifelse(c(time.bed <= hours(15) & time.bed >= hours(05))
                                  & tib.tot >= 12,
                                 "Yes", "No"))
```

Similarly, this chunk of syntax fixes where users are likely to have mixed-up AM/PM for their reported time of trying to fall asleep.

We assume that users whose inputted diary data indicate they were in bed longer than 12 hours with a stated bed time between 5 AM and 3 PM (i.e., 15 on 24-hour clock) entered AM / PM incorrectly. For example, someone inputting a bedtime of 1 PM on Monday, Jan 1, who started trying to fall asleep at 2 PM on Monday, Jan 1, with a wake and rise time of 6 AM on Tuesday, Jan 2. This syntax would update the sleep time to 2 AM on Tuesday, Jan 2.

```
data<- data %>%
  mutate(utctime.sleep.fix = ifelse(c(time.sleep <= hours(15) & time.sleep >= hours(05))
                                    & tib.h >= 12,
                                    utctime.sleep.fix + hours(12),
                                    utctime.sleep.fix))

data$utctime.sleep.fix<-as_datetime(data$utctime.sleep.fix)

data<- data %>%
  mutate(sleep.fix2.YN = ifelse(c(time.sleep <= hours(15) & time.sleep >= hours(05))
                                    & tib.h >= 12,
                                  "Yes", "No"))
```

We'll now again update our total time in bed variable to help identify remaining invalid diary entries. We'll also update our assessment time in bed variable to help identify invalid diaries now, too.

```
##V2: updated formulas to print as number (of hours) without formatting

data<- data %>%
  mutate(tib.tot = difftime(utctime.rise, utctime.bed, units = "hours" ))
```

```
data<- data %>%
  mutate(tib.h = difftime(utctime.rise, utctime.sleep, units = "hours"))
```

#V2: Added chunk to table diaries updated according to each or any cleaning update For your information, the syntax below will print counts of diaries that were updated according to each cleaning update, as well as the count of diaries with any data changes from one or more of the cleaning updates.

```
table(data$bed.fix1.YN)
```

```
##
##    No   Yes
## 52632   176
```

```
table(data$sleep.fix1.YN)
```

```
##
##    No   Yes
## 52786    22
```

```
table(data$wake.fix.YN)
```

```
##
##    No   Yes
## 52766    42
```

```
table(data$rise.fix.YN)
```

```
##
##    No   Yes
## 52641   167
```

```
table(data$bed.fix2.YN)
```

```
##
##    No   Yes
## 52594   214
```

```
table(data$sleep.fix2.YN)
```

```
##
##    No   Yes
## 52646   162
```

```
data<- data %>%
  mutate(anyfix = ifelse((bed.fix1.YN == "Yes" |
                            bed.fix2.YN == "Yes" |
                            sleep.fix1.YN == "Yes"|
                            sleep.fix2.YN == "Yes" |
                            rise.fix.YN == "Yes" |
                            wake.fix.YN == "Yes"), "Yes", "No"))
table(data$anyfix)
```

```
##
##    No   Yes
## 52252   556
```

##----------

Round 3 Data Cleaning

##----------

Once we've fixed (presumed) diary data entry errors, there are some additional errors that can occur. These are diaries with unresolvable data entry error(s) that need to be deleted due to impossible entries and not being able to make reasonable guesses at what the user intended to enter.

This chunk of syntax first computes total sleep time with updated data and identifies diaries as invalid where total sleep time <0. The 'tst' variable from the downloaded dataset is from the users' originally entered data, and will no longer be accurate from the cleaned diary times.

```
#V2: Simplified computation of tst.fix

data<- data %>%
  mutate(tst.fix = difftime(utctime.wake.fix, utctime.sleep.fix, units = "mins"))
data<- data %>%
  mutate(tst.fix = tst.fix - sol - waso)

data<- data %>%
  mutate(invalid = ifelse(data$tst.fix < minutes(0), "invalid", "valid"))
```

This chunk of syntax identifies diaries as invalid where updated times are out of order (e.g., updated sleep time occurs before updated rise time).

```
data<- data %>%
  mutate(invalid = ifelse((utctime.sleep.fix - utctime.bed.fix) < 0, "invalid", invalid))
data<- data %>%
  mutate(invalid = ifelse((utctime.rise.fix - utctime.wake.fix) < 0, "invalid", invalid))
data<- data %>%
  mutate(invalid = ifelse(tib.h < 0, "invalid", invalid))
```

The above syntax fixes the most common sleep diary entry errors. Now, you should manually review the data and remove any other suspicious or otherwise highly irregular diaries. The first line of code below will show your full dataset at present.

We suggest checking two additional common issues and making a data-driven decision with your team. The next section of code in the chunk below will show you instances where there is an unusually long gap between bedtime and sleeptime. The next section of code will show you instances where a participant reports an unusually long time in bed, which may indicate a problem with bed or rise time.

There may be other issues that you find from running summaries of other variables. We suggest that for each instance where there is an unusual/suspicious entry, you review the surrounding diaries and the personal notes for any clues about whether the diary should be edited or removed, or if it is likely to be valid.

```
#Manually review all data with this syntax:
view(data)
```

```
#V2 Update: Added manual review syntax of all diaries marked 'invalid'
#Review diaries marked 'invalid'
data %>%
  filter(invalid == "invalid") %>%
  view()

#Review long gaps from bed-sleep time (defined here as a gap lasting 8 hours or longer):
data %>%
  mutate(bedsleepgap = utctime.sleep.fix - utctime.bed.fix) %>%
  filter(bedsleepgap >= hours(8)) %>%
  view()

#Review long time in bed (defined here as 14 hours or longer):
data %>%
  filter(tib.tot >= 14) %>%
  view()

data %>%
  filter(tib.h >= 14) %>%
  view()
```

##EDITING MAY BE REQUIRED: If there are diaries to be removed based on your findings from above, uncomment (by deleting the '#' before the line) the syntax lines below and edit as needed. Make a record for yourself of why the diary was removed for your records (don't uncomment this line, as it is in fact a comment!):

```
#data<- data %>%
#  mutate(invalid = ifelse(diaryid == ENTER.NUMBER.HERE, "invalid", invalid))
#data<- data %>%
#  mutate(invalid = ifelse(diaryid == c(ENTER.NUMBER1.HERE, ENTER.NUMBER2.HERE,
#     ENTER.NUMBER3.HERE ), "invalid", invalid))

#Write in reason for removing those diaries (e.g., diaryid ENTER.NUMBER.HERE
#was removed because of XYZ reason)
```

##EDITING MAY BE REQUIRED: If there are diaries to be individually edited based on your findings from above, uncomment (by deleting the '#' before the line) the syntax lines below and edit as needed. This should be undertaken with extreme caution! Make sure you are editing the ".fix" variables, which we will use moving forward, rather than the original variables. Make a record for yourself of why the diary was edited for your records (don't uncomment this line, as it is in fact a comment!):

```
#Example 1: Fixing where a user entered 11AM on waking day for time to sleep
#that was not caught with original cleaning. You'll see this syntax is repeated,
#first with a "datacheck" dataset then repeated into our 'real' "data" dataset.
#This allows us to check to make sure we have made correct changes prior to
#finalizing them in our data!

#datacheck<- data %>%
#  mutate(utctime.sleep.fix = ifelse(diaryid == 32530,
#                                    utctime.sleep.fix - hours(12),
#                                    utctime.sleep.fix))
#datacheck$utctime.sleep.fix<- as_datetime(datacheck$utctime.sleep.fix)
#datacheck %>%
```

```
#  filter(diaryid == 32530) %>%
#  view()

#data<- data %>%
#  mutate(utctime.sleep.fix = ifelse(diaryid == 32530,
#                                    utctime.sleep.fix - hours(12),
#                                    utctime.sleep.fix))
#data$utctime.sleep.fix<- as_datetime(data$utctime.sleep.fix)
#data %>%
#  filter(diaryid == 32530) %>%
#  view()
##REASON FOR EDITING DIARY 32530: User entered 11pm bedtime, 11am sleep time
#    next day, 11:30am wake and rise time. Edited to 11pm bed and sleep time.


#Example 2: Fixing where a user entered 4AM on day before waking day for time
#to bed that was not caught with original cleaning. You'll see this syntax is
#repeated, first with a "datacheck" dataset then repeated into our 'real' "data"
#dataset. This allows us to check to make sure we have made correct changes prior
#to finalizing them in our data!

#datacheck<- data %>%
#  mutate(utctime.bed.fix = ifelse(diaryid == 27821,
#                                  utctime.bed.fix + hours(24),
#                                  utctime.bed.fix))
#datacheck$utctime.bed.fix<- as_datetime(datacheck$utctime.bed.fix)
#datacheck %>%
#  filter(diaryid == 27821) %>%
#  view()

#data<- data %>%
#  mutate(utctime.bed.fix = ifelse(diaryid == 27821,
#                                  utctime.bed.fix + hours(24),
#                                  utctime.bed.fix))
#data$utctime.bed.fix<- as_datetime(data$utctime.bed.fix)
#data %>%
#  filter(diaryid == 27821) %>%
#  view()
##REASON FOR EDITING DIARY 27821: User entered 4am bed time, 4am sleep time on
#    following day, and 10am wake / 10:10am rise times. Changed sleep time to
#    be concurrent with sleep time.

#rm(datacheck)
```

Now we'll remove our invalid diaries and create a new dataset, then remove our old dataset with invalid diaries from our environment:

```
data.valid<- data %>%
  filter(invalid == "valid")

rm(data)
```

You can now donwload this dataset for your records using the syntax below. Note that this is NOT the dataset you should use for computing assessment outcomes, as we will continue below to identify, select, and

compute outcomes from a particular set of assessment diaries. However, you now now have a dataset of all valid diaries (all valid diaries entered at assessments + during the SHUTi intervention for only valid users). This syntax will save your 'data' file as a .csv in your working directory folder.

```
write.csv(data.valid, "All_Valid_Sleep_Diaries.csv", row.names = FALSE)
```

##---------- ##----------

Assessment Outcomes

##---------- ##----------

Once you have cleaned the data, you'll want to make sure you only have Assessment diaries included in your study assessment analyses (i.e., remove 'intervention diaries' that were completed while the participant was using SHUTi).

```
assessment.data<- data.valid %>%
  filter(assessment != "NA")
```

##----------

Reliability Guideline 1

##----------

We constrain our valid assessment diaries to those completed within 2 months of the assessment period. For post-assessment, this means diaries must be completed by Day 123 (i.e., 9 weeks from day 0 [63 days] + 2 months [60 days]). For 6-month follow-up (note: 6-months from post-assessment), diaries must be completed by Day 303 (i.e., 9 weeks from day 0 [63 days] + 6 months [180 days] + 2 months [60 days]); 12-month follow-up diaries must be completed by Day 488 (i.e., 9 weeks from day 0 [63 days] + 12 months [365 days] + 2 months [60 days]). This makes sure the sleep diary outcomes are referring to our intended window of assessment, rather than weeks or months later!

To select diaries for each of these assessment time points within the above deadlines, we will upload the Events spreadsheet. This spreadsheet can be downloaded the same way from SHUTi that you downloaded your sleep diary data. Save the file into the same folder as your original data (i.e., your 'working directory')

This chunk of syntax uploads the Events spreadsheet and compares diary calendar dates with "Day 0". "Day 0" refers to the first day of the actual intervention begining. All pre-intervention surveys (which were submitted in block 1) would have been submitted before Day 0.

##EDITING MAY BE REQUIRED: This syntax is written for studies with 4 time points (at baseline, post-assessment [9 weeks post-baseline], 6-month follow-up [from post-assessment], and 12-month follow-up [from post-assessment]). If your study has a different follow-up assessment schedule, you will need to edit the syntax below accordingly.

```
events<-read_csv("Events data_export.csv")

events<- events %>%
  rename(EventType="Event Type",
         Day0="Created At")
```

```
events<-events %>%
  filter(grepl('Day0', EventType))


events<-events %>%
  dplyr::select(c(PID, EventType, Day0))

assessment.data<-merge(assessment.data, events, by = c("PID"))

assessment.data$Day0<-as_date(assessment.data$Day0)

assessment.data<- assessment.data %>%
  mutate(datediffassess = caldate - Day0)

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(assessment == "time_2"
                          & datediffassess > 123, "invalid", "valid"))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(assessment == "time_3"
                          & datediffassess > 303, "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(assessment == "time_4"
                          & datediffassess > 488, "invalid", invalid))

assessment.data<- assessment.data %>%
  filter(invalid == "valid")
```

##----------

Reliability Guideline 2

##----------

We use a minimum of 7 sleep diaries at a time point to compute outcomes for reliability purposes. This chunk of syntax identifies users who have fewer than 7 diaries at a time point, and then marks their diaries at that time point as invalid.

```
grouped.assessment<- assessment.data %>%
  group_by(assessment, PID)

ndiariespertime<-grouped.assessment %>%
  summarise(n.time=n())

assessment.data<-merge(assessment.data, ndiariespertime, by = c("PID", "assessment"))

rm(ndiariespertime)
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(n.time < 7, "invalid", invalid))
assessment.data<- assessment.data %>%
  filter(invalid == "valid")
```

##----------

Reliability Guideline 3

##----------

We look for blocks of diaries that have been completed within 14 days of one another. This chunk of syntax identifies the participants who have 10 valid diaries at a given time point, and marks these as the diaries to use at the assessment time point for outcomes computation.

This chunk of syntax will compute the date difference for the largest available block of diaries (i.e., from 7 to 10) for up to a total of 25 diaries, then select the first block with a date difference of 14 days or less. Note: Lines are repeated in the diary1 section to prevent a 'bug'.

Note for Step 1: NAs will start to be introduced for participants once you surpass the number of surveys that a given participant submitted. Also, because this code assumes four assessment time points, date1 should have no more than 4 rows per participant (but may have fewer if the participant did not complete sufficient surveys across all four assessment blocks).

```
##Step 1: create new variables that label, for each participant,
#         the specific diaryid number for each submitted survey, in order,
#         within each of the four assessment blocks ranging from survey 1
#         through survey 25

assessment.data<- assessment.data[order(assessment.data$caldate),]

diary1<- assessment.data %>%
  mutate(diarydate1 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate1))

diary1<- as.data.frame(diary1[,1:3])

colnames(diary1)[3] <- "diary1id"

assessment.data <- plyr::join_all(list(assessment.data, diary1),
                            by = c("PID", "assessment"),
                            type = 'full')

diary1<- as.data.frame(diary1[,1:3])
colnames(diary1)[3] <- "diary1id"
assessment.data <- plyr::join_all(list(assessment.data, diary1),
                            by = c("PID", "assessment"),
                            type = 'full')
diary2<- assessment.data %>%
  filter(diaryid != diary1id)
diary2<- diary2 %>%
  mutate(diarydate2 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate2))

diary2<- as.data.frame(diary2[,1:3])
colnames(diary2)[3] <- "diary2id"
assessment.data <- plyr::join_all(list(assessment.data, diary2),
                            by = c("PID", "assessment"),
```

```
                            type = 'full')

diary3<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id)
diary3<- diary3 %>%
  mutate(diarydate3 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate3))
diary3<- as.data.frame(diary3[,1:3])
colnames(diary3)[3] <- "diary3id"
assessment.data <- plyr::join_all(list(assessment.data, diary3),
                            by = c("PID", "assessment"),
                            type = 'full')

diary4<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id)
diary4<- diary4 %>%
  mutate(diarydate4 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate4))
diary4<- as.data.frame(diary4[,1:3])
colnames(diary4)[3] <- "diary4id"
assessment.data <- plyr::join_all(list(assessment.data, diary4),
                            by = c("PID", "assessment"),
                            type = 'full')

diary5<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id)
diary5<- diary5 %>%
  mutate(diarydate5 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate5))
diary5<- as.data.frame(diary5[,1:3])
colnames(diary5)[3] <- "diary5id"
assessment.data <- plyr::join_all(list(assessment.data, diary5),
                            by = c("PID", "assessment"),
                            type = 'full')
diary6<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id)
diary6<- diary6 %>%
  mutate(diarydate6 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate6))
diary6<- as.data.frame(diary6[,1:3])
colnames(diary6)[3] <- "diary6id"
assessment.data <- plyr::join_all(list(assessment.data, diary6),
                            by = c("PID", "assessment"),
                            type = 'full')

diary7<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
```

```
        & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id)
diary7<- diary7 %>%
  mutate(diarydate7 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate7))
diary7<- as.data.frame(diary7[,1:3])
colnames(diary7)[3] <- "diary7id"
assessment.data <- plyr::join_all(list(assessment.data, diary7),
                            by = c("PID", "assessment"),
                            type = 'full')

diary8<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
        & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
        & diaryid != diary7id)
diary8<- diary8 %>%
  mutate(diarydate8 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate8))
diary8<- as.data.frame(diary8[,1:3])
colnames(diary8)[3] <- "diary8id"
assessment.data <- plyr::join_all(list(assessment.data, diary8),
                            by = c("PID", "assessment"),
                            type = 'full')

diary9<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
        & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
        & diaryid != diary7id & diaryid != diary8id)
diary9<- diary9 %>%
  mutate(diarydate9 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate9))
diary9<- as.data.frame(diary9[,1:3])
colnames(diary9)[3] <- "diary9id"
assessment.data <- plyr::join_all(list(assessment.data, diary9),
                            by = c("PID", "assessment"),
                            type = 'full')

diary10<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
        & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
        & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id)
diary10<- diary10 %>%
  mutate(diarydate10 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate10))
diary10<- as.data.frame(diary10[,1:3])
colnames(diary10)[3] <- "diary10id"
assessment.data <- plyr::join_all(list(assessment.data, diary10),
                            by = c("PID", "assessment"),
                            type = 'full')
```

```
diary11<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id)
diary11<- diary11 %>%
  mutate(diarydate11 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate11))
diary11<- as.data.frame(diary11[,1:3])
colnames(diary11)[3] <- "diary11id"
assessment.data <- plyr::join_all(list(assessment.data, diary11),
                            by = c("PID", "assessment"),
                            type = 'full')

diary12<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id)
diary12<- diary12 %>%
  mutate(diarydate12 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate12))
diary12<- as.data.frame(diary12[,1:3])
colnames(diary12)[3] <- "diary12id"
assessment.data <- plyr::join_all(list(assessment.data, diary12),
                            by = c("PID", "assessment"),
                            type = 'full')
diary13<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id)
diary13<- diary13 %>%
  mutate(diarydate13 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate13))
diary13<- as.data.frame(diary13[,1:3])
colnames(diary13)[3] <- "diary13id"
assessment.data <- plyr::join_all(list(assessment.data, diary13),
                            by = c("PID", "assessment"),
                            type = 'full')

diary14<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id)
diary14<- diary14 %>%
  mutate(diarydate14 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
```

```
  slice(which.min(diarydate14))
diary14<- as.data.frame(diary14[,1:3])
colnames(diary14)[3] <- "diary14id"
assessment.data <- plyr::join_all(list(assessment.data, diary14),
                            by = c("PID", "assessment"),
                            type = 'full')

diary15<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id)
diary15<- diary15 %>%
  mutate(diarydate15 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate15))
diary15<- as.data.frame(diary15[,1:3])
colnames(diary15)[3] <- "diary15id"
assessment.data <- plyr::join_all(list(assessment.data, diary15),
                            by = c("PID", "assessment"),
                            type = 'full')

diary16<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id)
diary16<- diary16 %>%
  mutate(diarydate16 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate16))
diary16<- as.data.frame(diary16[,1:3])
colnames(diary16)[3] <- "diary16id"
assessment.data <- plyr::join_all(list(assessment.data, diary16),
                            by = c("PID", "assessment"),
                            type = 'full')

diary17<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id)
diary17<- diary17 %>%
  mutate(diarydate17 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate17))
diary17<- as.data.frame(diary17[,1:3])
colnames(diary17)[3] <- "diary17id"
assessment.data <- plyr::join_all(list(assessment.data, diary17),
```

```
                            by = c("PID", "assessment"),
                            type = 'full')

diary18<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id)
diary18<- diary18 %>%
  mutate(diarydate18 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate18))
diary18<- as.data.frame(diary18[,1:3])
colnames(diary18)[3] <- "diary18id"
assessment.data <- plyr::join_all(list(assessment.data, diary18),
                            by = c("PID", "assessment"),
                            type = 'full')

diary19<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id)
diary19<- diary19 %>%
  mutate(diarydate19 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate19))
diary19<- as.data.frame(diary19[,1:3])
colnames(diary19)[3] <- "diary19id"
assessment.data <- plyr::join_all(list(assessment.data, diary19),
                            by = c("PID", "assessment"),
                            type = 'full')

diary20<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id
         & diaryid != diary19id)
diary20<- diary20 %>%
  mutate(diarydate20 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate20))
diary20<- as.data.frame(diary20[,1:3])
colnames(diary20)[3] <- "diary20id"
assessment.data <- plyr::join_all(list(assessment.data, diary20),
                            by = c("PID", "assessment"),
```

```
                            type = 'full')

diary21<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id
         & diaryid != diary19id & diaryid != diary20id)
diary21<- diary21 %>%
  mutate(diarydate21 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate21))
diary21<- as.data.frame(diary21[,1:3])
colnames(diary21)[3] <- "diary21id"
assessment.data <- plyr::join_all(list(assessment.data, diary21),
                            by = c("PID", "assessment"),
                            type = 'full')

diary22<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id
         & diaryid != diary19id & diaryid != diary20id & diaryid != diary21id)
diary22<- diary22 %>%
  mutate(diarydate22 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate22))
diary22<- as.data.frame(diary22[,1:3])
colnames(diary22)[3] <- "diary22id"
assessment.data <- plyr::join_all(list(assessment.data, diary22),
                            by = c("PID", "assessment"),
                            type = 'full')

diary23<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id
         & diaryid != diary19id & diaryid != diary20id & diaryid != diary21id
         & diaryid != diary22id)
diary23<- diary23 %>%
  mutate(diarydate23 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate23))
diary23<- as.data.frame(diary23[,1:3])
colnames(diary23)[3] <- "diary23id"
```

```
assessment.data <- plyr::join_all(list(assessment.data, diary23),
                            by = c("PID", "assessment"),
                            type = 'full')

diary24<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id
         & diaryid != diary19id & diaryid != diary20id & diaryid != diary21id
         & diaryid != diary22id & diaryid != diary23id)
diary24<- diary24 %>%
  mutate(diarydate24 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate24))
diary24<- as.data.frame(diary24[,1:3])
colnames(diary24)[3] <- "diary24id"
assessment.data <- plyr::join_all(list(assessment.data, diary24),
                            by = c("PID", "assessment"),
                            type = 'full')

diary25<- assessment.data %>%
  filter(diaryid != diary1id & diaryid != diary2id & diaryid != diary3id
         & diaryid != diary4id & diaryid != diary5id & diaryid != diary6id
         & diaryid != diary7id & diaryid != diary8id & diaryid != diary9id
         & diaryid != diary10id & diaryid != diary11id & diaryid != diary12id
         & diaryid != diary13id & diaryid != diary14id & diaryid != diary15id
         & diaryid != diary16id & diaryid != diary17id & diaryid != diary18id
         & diaryid != diary19id & diaryid != diary20id & diaryid != diary21id
         & diaryid != diary22id & diaryid != diary23id & diaryid != diary24id)
diary25<- diary25 %>%
  mutate(diarydate25 = as.Date(caldate)) %>%
  group_by(assessment, PID) %>%
  slice(which.min(diarydate25))
diary25<- as.data.frame(diary25[,1:3])
colnames(diary25)[3] <- "diary25id"
assessment.data <- plyr::join_all(list(assessment.data, diary25),
                            by = c("PID", "assessment"),
                            type = 'full')

rm(diary1, diary2, diary3, diary4, diary5, diary6, diary7, diary8, diary9, diary10,
   diary11, diary12, diary13, diary14, diary15, diary16, diary17, diary18, diary19,
   diary20, diary21, diary22, diary23, diary24, diary25)


##Step 2: Isolate the specific date that each diary was submitted on

date1<- assessment.data %>%
  filter(diaryid == diary1id) %>%
  dplyr::select(c(PID, assessment, n.time, caldate))
```

```
date2<- assessment.data %>%
  filter(diaryid == diary2id) %>%
  dplyr::select(c(PID, assessment, caldate))

date3<- assessment.data %>%
  filter(diaryid == diary3id) %>%
  dplyr::select(c(PID, assessment, caldate))

date4<- assessment.data %>%
  filter(diaryid == diary4id) %>%
  dplyr::select(c(PID, assessment, caldate))

date5<- assessment.data %>%
  filter(diaryid == diary5id) %>%
  dplyr::select(c(PID, assessment, caldate))

date6<- assessment.data %>%
  filter(diaryid == diary6id) %>%
  dplyr::select(c(PID, assessment, caldate))

date7<- assessment.data %>%
  filter(diaryid == diary7id) %>%
  dplyr::select(c(PID, assessment, caldate))

date8<- assessment.data %>%
  filter(diaryid == diary8id) %>%
  dplyr::select(c(PID, assessment, caldate))

date9<- assessment.data %>%
  filter(diaryid == diary9id) %>%
  dplyr::select(c(PID, assessment, caldate))

date10<- assessment.data %>%
  filter(diaryid == diary10id) %>%
  dplyr::select(c(PID, assessment, caldate))

date11<- assessment.data %>%
  filter(diaryid == diary11id) %>%
  dplyr::select(c(PID, assessment, caldate))

date12<- assessment.data %>%
  filter(diaryid == diary12id) %>%
  dplyr::select(c(PID, assessment, caldate))

date13<- assessment.data %>%
  filter(diaryid == diary13id) %>%
  dplyr::select(c(PID, assessment, caldate))

date14<- assessment.data %>%
  filter(diaryid == diary14id) %>%
  dplyr::select(c(PID, assessment, caldate))

date15<- assessment.data %>%
```

```
  filter(diaryid == diary15id) %>%
  dplyr::select(c(PID, assessment, caldate))

date16<- assessment.data %>%
  filter(diaryid == diary16id) %>%
  dplyr::select(c(PID, assessment, caldate))

date17<- assessment.data %>%
  filter(diaryid == diary17id) %>%
  dplyr::select(c(PID, assessment, caldate))

date18<- assessment.data %>%
  filter(diaryid == diary18id) %>%
  dplyr::select(c(PID, assessment, caldate))

date19<- assessment.data %>%
  filter(diaryid == diary19id) %>%
  dplyr::select(c(PID, assessment, caldate))

date20<- assessment.data %>%
  filter(diaryid == diary20id) %>%
  dplyr::select(c(PID, assessment, caldate))

date21<- assessment.data %>%
  filter(diaryid == diary21id) %>%
  dplyr::select(c(PID, assessment, caldate))

date22<- assessment.data %>%
  filter(diaryid == diary22id) %>%
  dplyr::select(c(PID, assessment, caldate))

date23<- assessment.data %>%
  filter(diaryid == diary23id) %>%
  dplyr::select(c(PID, assessment, caldate))

date24<- assessment.data %>%
  filter(diaryid == diary24id) %>%
  dplyr::select(c(PID, assessment, caldate))

date25<- assessment.data %>%
  filter(diaryid == diary25id) %>%
  dplyr::select(c(PID, assessment, caldate))

datediff <- plyr::join_all(list(date1, date2, date3, date4, date5, date6, date7, date8,
                                date9, date10, date11, date12, date13, date14, date15,
                                date16, date17, date18, date19, date20, date21, date22,
                                date23, date24, date25), by = c("PID", "assessment"))

rm(date1, date2, date3, date4, date5, date6, date7, date8, date9, date10, date11, date12,
   date13, date14, date15, date16, date17, date18, date19, date20, date21, date22, date23,
   date24, date25)

colnames(datediff)[4:28] <- c("Date1", "Date2", "Date3", "Date4", "Date5", "Date6",
```

```
                              "Date7", "Date8", "Date9", "Date10", "Date11", "Date12",
                              "Date13", "Date14", "Date15", "Date16", "Date17", "Date18",
                              "Date19", "Date20", "Date21", "Date22", "Date23", "Date24",
                              "Date25")


#Step 3: Calculate the amount of calendar days that elapsed between the last 10
#        surveys that were submitted within each block

datediff<- datediff %>%
  mutate(datediff = ifelse(n.time == 7, Date7 - Date1, NA))
datediff<- datediff %>%
  mutate(datediff = ifelse(n.time == 8, Date8 - Date1, datediff))
datediff<- datediff %>%
  mutate(datediff = ifelse(n.time == 9, Date9 - Date1, datediff))
datediff<- datediff %>%
  mutate(datediff = ifelse(n.time >= 10, Date10 - Date1, datediff))
datediff<- datediff %>%
  mutate(datediff2 = ifelse(n.time >= 11, Date11 - Date2, NA))
datediff<- datediff %>%
  mutate(datediff3 = ifelse(n.time >= 12, Date12 - Date3, NA))
datediff<- datediff %>%
  mutate(datediff4 = ifelse(n.time >= 13, Date13 - Date4, NA))
datediff<- datediff %>%
  mutate(datediff5 = ifelse(n.time >= 14, Date14 - Date5, NA))
datediff<- datediff %>%
  mutate(datediff6 = ifelse(n.time >= 15, Date15 - Date6, NA))
datediff<- datediff %>%
  mutate(datediff7 = ifelse(n.time >= 16, Date16 - Date7, NA))
datediff<- datediff %>%
  mutate(datediff8 = ifelse(n.time >= 17, Date17 - Date8, NA))
datediff<- datediff %>%
  mutate(datediff9 = ifelse(n.time >= 18, Date18 - Date9, NA))
datediff<- datediff %>%
  mutate(datediff10 = ifelse(n.time >= 19, Date19 - Date10, NA))
datediff<- datediff %>%
  mutate(datediff11 = ifelse(n.time >= 20, Date20 - Date11, NA))
datediff<- datediff %>%
  mutate(datediff12 = ifelse(n.time >= 21, Date21 - Date12, NA))
datediff<- datediff %>%
  mutate(datediff13 = ifelse(n.time >= 22, Date22 - Date13, NA))
datediff<- datediff %>%
  mutate(datediff14 = ifelse(n.time >= 23, Date23 - Date14, NA))
datediff<- datediff %>%
  mutate(datediff15 = ifelse(n.time >= 24, Date24 - Date15, NA))
datediff<- datediff %>%
  mutate(datediff16 = ifelse(n.time >= 25, Date25 - Date16, NA))


#Step 4: Determine if an assessment block is valid based on inclusion rules listed above

validblock<- datediff %>%
  dplyr::select(c(PID, assessment, datediff:datediff16))
```

```
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff <= 14,
                           "Use.Block1", NA))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 <= 14,
                           "Use.Block2", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 <= 14,
                           "Use.Block3", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 <= 14,
                           "Use.Block4", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 <= 14,
                           "Use.Block5", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 <= 14,
                           "Use.Block6", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 <= 14,
                           "Use.Block7", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 <= 14,
                           "Use.Block8", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 <= 14,
                           "Use.Block9", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 <= 14,
                           "Use.Block10", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 > 14 & datediff11 <= 14,
                           "Use.Block11", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 > 14 & datediff11 > 14 & datediff12 <= 14,
```

```
                            "Use.Block12", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 > 14 & datediff11 > 14 & datediff12 > 14
                           & datediff13 <= 14,
                           "Use.Block13", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 > 14 & datediff11 > 14 & datediff12 > 14
                           & datediff13 > 14 & datediff14 <= 14,
                           "Use.Block14", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 > 14 & datediff11 > 14 & datediff12 > 14
                           & datediff13 > 14 & datediff14 > 14 & datediff15 <= 14,
                           "Use.Block15", useblock))
validblock<- validblock %>%
  mutate(useblock = ifelse(datediff > 14 & datediff2 > 14 & datediff3 > 14
                           & datediff4 > 14 & datediff5 > 14 & datediff6 > 14
                           & datediff7 > 14 & datediff8 > 14 & datediff9 > 14
                           & datediff10 > 14 & datediff11 > 14 & datediff12 > 14
                           & datediff13 > 14 & datediff14 > 14 & datediff15 > 14
                           & datediff16 <= 14,
                           "Use.Block16", useblock))

validblock<- validblock %>%
  mutate(useblock = ifelse(is.na(useblock), "Review.Block", useblock))
```

The syntax above is written to review up to 25 Sleep Diaries per participant per time point to identify a block of 7-10 diaries completed in 14 days, and within 2 months of the assessment period. This chunk of syntax identifies if you have any participants with more than 25 diaries at a single time point AND who do not have a valid block in their first 25 diaries. If you have such a participant, you will 'manually' select their diaries to be included by editing and executing the syntax below.

The syntax below will create two datasets: the "check" and "review" datasets. The "check" dataset will show you any participants who have more than 25 diaries at any given time point. The "review" dataset will show you participants who do not have a valid block of diaries selected from their first 25 diaries.

```
assessment.data.addl<- assessment.data %>%
  filter(n.time >= 26)

grouped.assessment.addl<- assessment.data.addl %>%
  group_by(assessment, PID)

addldiaries<- grouped.assessment.addl %>%
  mutate(addldiaries = "Yes - check")

addldiaries<- addldiaries %>%
```

```
  distinct(PID, .keep_all = TRUE)

assessment.data <- plyr::join_all(list(assessment.data, addldiaries),
                            by = c("PID", "assessment"),
                            type = 'full')

check<- assessment.data %>%
  filter(addldiaries == "Yes - check")


review<- validblock %>%
  filter(useblock == "Review.Block")
```

At this point, we want to ensure that we're not accidentally failing to label and retain any valid block of diaries that are currently labeled in the variable useblock as "Review.Block". This review process must be done manually by completing the following steps:

First, we want to know if there is overlap between the "check" and "review" datasets which were created above. If there is overlap (i.e., the same participant ID shows up in both data sets), that means a user may have a valid block of 7-10 diaries that extends beyond the first 25 diaries. If this is the case, it will change the data you need to consider when checking to see if a valid block of diaries is available.

To determine validity of diary blocks, check to see if any of the below five conditions are met. Go in order and stop checking conditions once one is satisfied. You want to move forward with the condition that is labeled as close to (1) as possible, given the conditions are listed in order of preference: (1) first block of 10 diaries in 14 days (e.g., caldate = 2022/01/01 to caldate = 2022/01/15; any other diaries should be marked as invalid and later removed) (2) if no block of 10, select the first block of 9 diaries in 14 days (any other diaries should be marked as invalid and later removed) (3) if no block of 9, select the first block of 8 diaries in 14 days (any other diaries should be marked as invalid and later removed) (4) if no block of 8, select the first block of 7 diaries in 14 days (any other diaries should be marked as invalid and later removed) (5) if no block of 7 diaries, then all diaries will be marked as invalid and removed, so no outcomes will be computed for this participant at this time point

```
assessment.data<-assessment.data %>%
  mutate(invalid = "invalid")

repCheck <- intersect(check$PID, review$PID)
```

If the output above does not print any PIDs, ignore the check database, and skip the next "chunk" to go look at the "review" data frame.

If the output prints any PIDs, you will use the "check" database with the syntax chunk below to manually review diaries for the identified participants to see whether there is a valid block of diaries across all of their submitted diaries using the decision rules numbered above.

Selecting Assessment Diaries for overlapping PIDs in "check" and "review"

```
#EXAMPLE SYNTAX FOR SELECTING ASSESSMENT DIARIES FOR USERS IN THE 'CHECK' AND
#'REVIEW' DATABASES -- used to label specific diaryids as invalid based on your
#'manual review of the "check" and "review" datasets:

view(check)

#assessment.data <- assessment.data %>%
#  mutate(invalid = ifelse(diaryid == ENTER.NUMBER |
```

```
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER |
#                             diaryid == ENTER.NUMBER, "valid", invalid))
```

Selecting assessment diaries for PIDs in the "review" database

```
view(review)

#EXAMPLE SYNTAX -- used to look at diaries for specific participants at
#specific time points who did not have a block of 10 identified within 14 days.

#Participant 360 had no valid block identified at Time 3, and there was no
#block of 7-9 or more diaries within 14 days:

#pt360t3<- assessment.data %>%
#  filter(PID == 360 & assessment == "time_3")
#view(pt360t3)
##NOTE ABOUT DATA: No block of >=7 within a 14 day time period - no valid
#                  assessment diary block at Time 3 for this user


#Participant 1018 had no valid block identified at Time 2, and there was
#no block of 7-9 or more diaries within 14 days:
#pt1018t2<- assessment.data %>%
#  filter(PID == 1018 & assessment == "time_2")
#view(pt1018t2)
##NOTE ABOUT DATA: No block of >=7 within a 14 day time period - no valid
#                  assessment diary block at Time 2 for this user

#Participant 1018 also had no valid block identified at Time 3, but there was
#     a block of 8 diaries within 14 days from 11/2-11/16
#pt1018t3<- assessment.data %>%
#  filter(PID == 1018 & assessment == "time_3")
#view(pt1018t3)

#assessment.data <- assessment.data %>%
#  mutate(invalid = ifelse(diaryid == 36544 |
#                             diaryid == 36613 |
#                             diaryid == 36804 |
#                             diaryid == 36875 |
#                             diaryid == 36926 |
#                             diaryid == 36978 |
#                             diaryid == 37290 |
#                             diaryid == 37347, "valid", invalid))
##NOTE ABOUT DATA: No block of >=9 within 14 day time period,
#                  but 8 diaries from 11/2-11/16
```

```
#Participant 1447 had no valid block identified at Time 3, but there was a
#block of 9 diaries in 14 days

#pt1447t3<- assessment.data %>%
#  filter(PID == 1447 & assessment == "time_3")
#view(pt1447t3)

#Block of 9 diaries from Dec 13 through Dec 23
#assessment.data <- assessment.data %>%
#  mutate(invalid = ifelse(diaryid == 38780 |
#                            diaryid == 38815 |
#                            diaryid == 38868 |
#                            diaryid == 38975 |
#                            diaryid == 38976 |
#                            diaryid == 39233 |
#                            diaryid == 39234 |
#                            diaryid == 39236 |
#                            diaryid == 39324, "valid", invalid))
```

It's a good idea to clean up your data environment a bit at this point.

##EDITING MAY BE REQUIRED: Add in any of the datasets you created to review for blocks of 7-9 diaries above

```
rm(addldiaries, assessment.data.addl, grouped.assessment.addl, check, review, repCheck, events)
```

##----------

Select diaries to be used for computing final outcomes

##----------

##EDITING MAY BE REQUIRED: This syntax labels which 10 diaries within each block are valid and should therefore be used when calculating outcomes. For example, if Block 1 was identified previously in the script as the block of data to be used for calculating outcomes, then diaries 1-10 will be marked as "valid" and diaries 11-25 will be marked as "invalid." However, if Block 2 was identified as the block of data to be used, then diaries 2-11 will be marked as "valid" and diaries 1 and 12-25 will be marked as "invalid."

```
assessment.data<- merge(assessment.data,
                        validblock[ , c("PID", "assessment", "useblock")],
                        by = c("PID", "assessment"), all.x = TRUE)

rm(validblock)

assessment.data <- assessment.data %>%
  mutate_at(vars(diary8id:diary25id), list(~na_replace(., 0)))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary1id, "Diary1", 0))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary2id, "Diary2", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary3id, "Diary3", diaryn))
```

```
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary4id, "Diary4", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary5id, "Diary5", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary6id, "Diary6", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary7id, "Diary7", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary8id, "Diary8", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary9id, "Diary9", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary10id, "Diary10", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary11id, "Diary11", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary12id, "Diary12", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary13id, "Diary13", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary14id, "Diary14", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary15id, "Diary15", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary16id, "Diary16", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary17id, "Diary17", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary18id, "Diary18", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary19id, "Diary19", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary20id, "Diary20", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary21id, "Diary21", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary22id, "Diary22", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary23id, "Diary23", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary24id, "Diary24", diaryn))
assessment.data<- assessment.data %>%
  mutate(diaryn = ifelse(diaryid == diary25id, "Diary25", diaryn))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block1" & (diaryn == "Diary1" |
                                                      diaryn == "Diary2" |
                                                      diaryn == "Diary3" |
                                                      diaryn == "Diary4" |
                                                      diaryn == "Diary5" |
                                                      diaryn == "Diary6" |
                                                      diaryn == "Diary7" |
                                                      diaryn == "Diary8" |
```

```
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10"), "valid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block1" & (diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block2" & (diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block2" & (diaryn == "Diary1" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block3" & (diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
```

```
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block3" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block4" & (diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block4" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block5" & (diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
```

```
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block5" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block6" & (diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14"|
                                                    diaryn == "Diary15"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block6" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
```

```
  mutate(invalid = ifelse(useblock == "Use.Block7" & (diaryn == "Diary7" |
                                                      diaryn == "Diary8" |
                                                      diaryn == "Diary9" |
                                                      diaryn == "Diary10" |
                                                      diaryn == "Diary11" |
                                                      diaryn == "Diary12" |
                                                      diaryn == "Diary13" |
                                                      diaryn == "Diary14"|
                                                      diaryn == "Diary15" |
                                                      diaryn == "Diary16"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block7" & (diaryn == "Diary1" |
                                                      diaryn == "Diary2" |
                                                      diaryn == "Diary3" |
                                                      diaryn == "Diary4" |
                                                      diaryn == "Diary5" |
                                                      diaryn == "Diary6" |
                                                      diaryn == "Diary17" |
                                                      diaryn == "Diary18" |
                                                      diaryn == "Diary19" |
                                                      diaryn == "Diary20" |
                                                      diaryn == "Diary21" |
                                                      diaryn == "Diary22" |
                                                      diaryn == "Diary23" |
                                                      diaryn == "Diary24" |
                                                      diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block8" & (diaryn == "Diary8" |
                                                      diaryn == "Diary9" |
                                                      diaryn == "Diary10" |
                                                      diaryn == "Diary11" |
                                                      diaryn == "Diary12" |
                                                      diaryn == "Diary13" |
                                                      diaryn == "Diary14" |
                                                      diaryn == "Diary15" |
                                                      diaryn == "Diary16" |
                                                      diaryn == "Diary17"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block8" & (diaryn == "Diary1" |
                                                      diaryn == "Diary2" |
                                                      diaryn == "Diary3" |
                                                      diaryn == "Diary4" |
                                                      diaryn == "Diary5" |
                                                      diaryn == "Diary6" |
                                                      diaryn == "Diary7" |
                                                      diaryn == "Diary18" |
                                                      diaryn == "Diary19" |
                                                      diaryn == "Diary20" |
                                                      diaryn == "Diary21" |
                                                      diaryn == "Diary22" |
                                                      diaryn == "Diary23" |
                                                      diaryn == "Diary24" |
```

```
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block9" & (diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block9" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block10" & (diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block10" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
```

```
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block11" & (diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block11" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block12" & (diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block12" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
```

```
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block13" & (diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block13" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block14" & (diaryn == "Diary14" |
                                                    diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block14" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
```

```
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "invalid", invalid))

assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block15" & (diaryn == "Diary15" |
                                                    diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" ), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block15" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary25"), "invalid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block16" & (diaryn == "Diary16" |
                                                    diaryn == "Diary17" |
                                                    diaryn == "Diary18" |
                                                    diaryn == "Diary19" |
                                                    diaryn == "Diary20" |
                                                    diaryn == "Diary21" |
                                                    diaryn == "Diary22" |
                                                    diaryn == "Diary23" |
                                                    diaryn == "Diary24" |
                                                    diaryn == "Diary25"), "valid", invalid))
assessment.data<- assessment.data %>%
  mutate(invalid = ifelse(useblock == "Use.Block16" & (diaryn == "Diary1" |
                                                    diaryn == "Diary2" |
                                                    diaryn == "Diary3" |
                                                    diaryn == "Diary4" |
                                                    diaryn == "Diary5" |
```

```
                                                    diaryn == "Diary6" |
                                                    diaryn == "Diary7" |
                                                    diaryn == "Diary8" |
                                                    diaryn == "Diary9" |
                                                    diaryn == "Diary10" |
                                                    diaryn == "Diary11" |
                                                    diaryn == "Diary12" |
                                                    diaryn == "Diary13" |
                                                    diaryn == "Diary14" |
                                                    diaryn == "Diary15"), "invalid", invalid))

rm(datediff)
```

This chunk of syntax selects only the valid assessment diaries - i.e., diaries that will be used to compute outcomes - and then downloads the dataset to your working directory.

```
assessment.data.valid<- assessment.data %>%
  filter(invalid == "valid")

write.csv(assessment.data.valid, "Assessment_Sleep_Diaries.csv", row.names = FALSE)

rm(assessment.data, data.valid, grouped.assessment)
```

##----------

Compute Outcome Variables

##----------

There are five variables typically reported from the Sleep Diaries:

(1) SOL: Sleep Onset Latency - amount of time that the participant reports that it took them to fall asleep after starting to try to fall asleep. This is a self-reported variable.

(2) WASOEMA: Wake After Sleep Onset plus Early Morning Awakening - amount of time during mid-sleep awakenings reported by the user PLUS amount of time a person wakes before their intended rise time. WASO is self-reported by the participant; EMA is computed based on the CORRECTED sleep diary times.

(3) TST: Total Sleep Time - computed time that user would have been asleep based on their reported diary data. Equals (CORRECTED wake time) - (CORRECTED time trying to fall asleep) - (self-reported SOL) - (self-reported WASO)

(4) TIB: Time in Bed - computed time that a user is in bed attempting to sleep. Equals (CORRECTED rise time) - (CORRECTED time trying to fall asleep)

(5) SE: Sleep Efficiency - computed as the percentage of time that a user is asleep while in bed attempting to sleep. Equals TST / TIB * 100.

(6) NumAwake: Number of times awakening during the night - number of times that the participant reports they awoke between initially falling asleep and their final waking. This is a self-reported variable.

(7) SleepQ: Sleep Quality - the participant's rating of their perceived sleep quality during the past night. 5-point Likert scale where: 'very_poor' = '1', 'poor' = '2', 'fair' = '3', 'good' = '4', 'very_good' = '5'

Below, these variables are computed and summarized

(1) SOL (in minutes)

This code creates an average SOL for each participant within each assessment block and stores info into a new data frame named "sol"

```
grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)

sol<- grouped.assessment %>%
  summarise(sol = mean(sol))
```

(2) WASOEMA (in minutes)

This code creates new variable called "ema.fix" by subtracting utctime.wake.fix from utctime.rise.fix and then dividing that value by 60 to scale in units of minutes. NOTE: because of earlier fixes, ema.fix should never be a negative value (because you can't rise before you wake). The next part of the code creates a new variable called "wasoema", which is the sum of ema.fix + waso. Finally, the last part of the code creates an average wasoema for each participant within each assessment block and stores info into a new data frame named "wasoema".

```
assessment.data.valid<- assessment.data.valid %>%
  mutate(ema.fix = as.numeric(utctime.rise.fix - utctime.wake.fix) / 60)

assessment.data.valid<- assessment.data.valid %>%
  mutate(wasoema = as.numeric(ema.fix + waso))

grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)

wasoema<- grouped.assessment %>%
  summarise(wasoema = mean(wasoema))
```

(3) TST (in hours)

This code first overwrites the variable called "tst.fix" by subtracting utctime.sleep.fix from utctime.wake.fix The next part of the code overwrites the tst.fix that was just solved for by taking the value and further subtracting out the waso and sol values. Then, it divides that solution by 60 to scale from minutes to hours. Finally, the code creates an average tst for each participant within each assessment block and stores that info into a new data frame named "tst". NOTE: because of earlier fixes, tst.fix should never be a negative value (because you can't be asleep for negative minutes).

```
assessment.data.valid<- assessment.data.valid %>%
  mutate(tst.fix = as.numeric((utctime.wake.fix - utctime.sleep.fix)))

assessment.data.valid<- assessment.data.valid %>%
  mutate(tst.fix = as.numeric((tst.fix - waso - sol) / 60))


grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)
```

```
tst<- grouped.assessment %>%
  summarise(tst = mean(tst.fix))
```

(4) TIB (in hours)

This code creates a new variable called "tib.fix" by subtracting utctime.sleep.fix from utctime.rise.fix. Then this value is scaled from minutes to hours by dividing by 60. Finally, the code creates an average tib for each participant within each assessment block and stores info into a new data frame named "tib". NOTE: because of earlier fixes, tib.fix should never be a negative value (because you can't be in bed for negative minutes).

```
assessment.data.valid<- assessment.data.valid %>%
  mutate(tib.fix = as.numeric(utctime.rise.fix - utctime.sleep.fix) / 60)

grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)

tib<- grouped.assessment %>%
  summarise(tib = mean(tib.fix))
```

(5) SE (percentage)

This code creates a new variable called "se.fix" by dividing tst.fix by tib.fix and then multiplying the product by 100 to interpret as a percentage between 0 and 100. Finally, it creates an average sleep efficiency score for each participant within each assessment block and stores info into a new data frame named "se".

```
assessment.data.valid<- assessment.data.valid %>%
  mutate(se.fix = (tst.fix / tib.fix) * 100)

grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)

se<- grouped.assessment %>%
  summarise(se = mean(se.fix))
```

##V2: Added computation for NumAwake and SleepQ variables (6) NumAwake (integer)

This code creates new variable called "NumAwake" by averaging the number of times a participant reports that they have awoken each evening after initially falling asleep over the block of sleep diaries.

```
grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)

NumAwake<- grouped.assessment %>%
  summarise(NumAwake = mean(wakecount))
```

(7) SleepQ (Likert score)

This code creates new variable called "SleepQ" by averaging participants' sleep quality ratings over the the block of sleep diaires.

```
assessment.data.valid$SleepQ<- recode(assessment.data.valid$sleepqual,
                                      'very_poor' = '1', 'poor' = '2', 'fair' = '3',
                                      'good' = '4', 'very_good' = '5')
assessment.data.valid$SleepQ<- as.numeric(assessment.data.valid$SleepQ)

grouped.assessment<- assessment.data.valid %>%
  group_by(assessment, PID)

SleepQ<- grouped.assessment %>%
  summarise(SleepQ = mean(SleepQ))
```

##---

Compute Additional Variables of Interest Describing Users' Interactions with the SHUTi Program and Patient Education Control

##---

In this section we will build a data frame, called "indat", which will contain information about how a user interacted with the program during the 9-week intervention period (i.e., between Day 0 and Day 62). The variables that we will calculate in this section are: Int_Diary_Count; Int_Login_Count; Int_Cores_Comp; Tot_Cores_Comp; PE_initiated; and PE_completed. Descriptions for each of these variables will be provided before each relevant code chunk. After building the "intdat" data frame, we will merge this information with all the other outcomes data above to arrive at our final assessment outcomes data set.

First, we will calculate "Int_Diary_Count", which is a count variable that reflects the number of diaries that a participant entered during the 9-week intervention period. Only participants assiged to the SHUTi experimental condition will have >0 intervention diaries, as diary-logging is exclusively a feature of SHUTi and not available as part of PE. These diaries are separate of those completed as part of assessment periods.

```
#Step 1: Load in data/data management
rawDat <- read.csv("Diaries data_export.csv")
uniqueIDs <- unique(assessment.data.valid$PID)
rawDat <- rawDat[which(rawDat$PID %in% uniqueIDs), ]

dat_rawEvents <- read.csv("Events data_export.csv")
dat_rawEvents <- dat_rawEvents[which(dat_rawEvents$PID %in% uniqueIDs), ]

intdat <- uniqueIDs
intdat <- as.data.frame(intdat)
names(intdat) <- "PID"

#Step 2: Calculate Int_Diary_Count variable
day0Marker <-"Event::Day0"
dat_rawEvents$day0data <- ifelse(str_detect(dat_rawEvents$Event.Type, day0Marker),1,0)
day0data <- dat_rawEvents[which(dat_rawEvents$day0data==1),]
day0data <- day0data[c(4,9)]
names(day0data) <- c("PID", "Day0Date")

rawDat <- left_join(rawDat, day0data, by = "PID")
dat_rawEvents$day0data <- NULL
```

```
rawDat$Created.At_date <- str_replace(rawDat$Created.At, "UTC", "")
rawDat$Day0Date <- str_replace(rawDat$Day0Date, "UTC", "")
rawDat$Created.At_date <- ymd_hms(rawDat$Created.At_date, tz = "UTC")
rawDat$Day0Date <- ymd_hms(rawDat$Day0Date, tz = "UTC")
rawDat$Created.At_date <- as.Date(rawDat$Created.At)
rawDat$Day0Date_date <- as.Date(rawDat$Day0Date)
rawDat$IntDay <- as.Date(as.character(rawDat$Created.At_date), format="%Y-%m-%d") -
  as.Date(as.character(rawDat$Day0Date_date), format="%Y-%m-%d")

rawDat$inWindow <- ifelse(rawDat$IntDay >= 1 & rawDat$IntDay <= 62,
                          "Retain", "Remove")
rawDat_reduced <- rawDat[which(rawDat$inWindow=="Retain"),]


intCount_dat <- aggregate(Diary.ID ~ PID, data = rawDat_reduced, FUN = length)

names(intCount_dat) <- c("PID", "Int_Diary_Count")


intdat <- left_join(intdat, intCount_dat, by = "PID")

intdat[is.na(intdat)] <- 0
```

Next, we will calculate "Int_Login_Count", which is a count variable that reflects the number of "meaningful logins" that a participant made to their assigned online intervention program during the 9-week intervention period. We define a "meaningful login" as all unique logins that took place at least 5 minutes apart from another login. The below code removes logins that occurred within 5 minutes of a previously counted login to avoid inflating a user's "Int_Login_Count" due to frequent clicking into and out of the intervention program without having time sufficient time to engage with the program. Logins are counted both for those assigned to the experimental SHUTi condition, as well as to the online Patient Education control condition.

```
#Step 1: Data management
dat_rawEvents <- left_join(dat_rawEvents, day0data, by = "PID")
dat_rawEvents$day0data <- NULL

dat_rawEvents$Created.At_date <- str_replace(dat_rawEvents$Created.At, "UTC", "")
dat_rawEvents$Day0Date <- str_replace(dat_rawEvents$Day0Date, "UTC", "")
dat_rawEvents$Created.At_date <- ymd_hms(dat_rawEvents$Created.At_date, tz = "UTC")
dat_rawEvents$Day0Date <- ymd_hms(dat_rawEvents$Day0Date, tz = "UTC")
dat_rawEvents$Created.At_date <- as.Date(dat_rawEvents$Created.At)
dat_rawEvents$Day0Date_date <- as.Date(dat_rawEvents$Day0Date)
dat_rawEvents$IntDay <- as.Date(as.character(dat_rawEvents$Created.At_date),
                        format="%Y-%m-%d") - as.Date(as.character(dat_rawEvents$Day0Date_date),
                        format="%Y-%m-%d")

dat_rawEvents$inWindow <- ifelse(dat_rawEvents$IntDay >= 0 & dat_rawEvents$IntDay <= 62,
                                 "Retain", "Remove")
dat_rawEvents_reduced<- dat_rawEvents[which(dat_rawEvents$inWindow=="Retain"),]


startsWith <- "Login by participant"
dat_rawEvents_reduced$login <- ifelse(str_detect(dat_rawEvents_reduced$Description, startsWith),
                                      "Yes", "No")
```

```r
dat_rawEvents_reduced_2 <- dat_rawEvents_reduced[which(dat_rawEvents_reduced$login=="Yes"),]
dat_rawEvents_reduced_2$Created.At <-str_replace(dat_rawEvents_reduced_2$Created.At, "UTC", "")
dat_rawEvents_reduced_2$Created.At <-ymd_hms(dat_rawEvents_reduced_2$Created.At, tz = "UTC")
dat_rawEvents_reduced_2$Day0Date <-str_replace(dat_rawEvents_reduced_2$Day0Date, "UTC", "")
dat_rawEvents_reduced_2$Day0Date <-ymd_hms(dat_rawEvents_reduced_2$Day0Date, tz = "UTC")
dat_rawEvents_reduced_2$timeDiff <- as.numeric(dat_rawEvents_reduced_2$Created.At
                                               - dat_rawEvents_reduced_2$Day0Date,
                                               units = "secs")

dat_rawEvents_reduced_2$inWindow <- ifelse((dat_rawEvents_reduced_2$IntDay == 0
                                          & dat_rawEvents_reduced_2$timeDiff > 300)
                                          | dat_rawEvents_reduced_2$IntDay > 0,
                                          "Retain", "Remove")

dat_rawEvents_reduced_2 <- dat_rawEvents_reduced_2[which(dat_rawEvents_reduced_2$inWindow==
                                                         "Retain"),]


#Step 2: Remove logins that occurred within 5 minutes of an eligible login

KeepDrop <- rep(NA, nrow(dat_rawEvents_reduced_2))
k <- 1
for(i in 1:length(unique(dat_rawEvents_reduced_2$PID))){

  personDat <- dat_rawEvents_reduced_2$Created.At[dat_rawEvents_reduced_2$PID ==
                                                    unique(dat_rawEvents_reduced_2$PID)[i]]

  firstLogin <- personDat[1]

  KeepDrop[k] <- "Keep"

  k <- k + 1

  if (length(personDat) == 1){
    next
  }

  for(j in 2:length(personDat)){

    if (as.numeric(personDat[j] - firstLogin, units = "secs") > 300){
      KeepDrop[k] <- "Keep"
      firstLogin <- personDat[j]
    } else {
      KeepDrop[k] <- "Drop"
    }
    k <- k + 1
  }

}

dat_rawEvents_reduced_2$KeepDrop <- KeepDrop

dat_rawEvents_rm<- dat_rawEvents_reduced_2[which(dat_rawEvents_reduced_2$KeepDrop=="Keep"),]
```

```
#Step 3: Calculate number of intervention logins submitted between Day 0
#        (not including first 5 minutes) and Day 62
intLoginCount_dat <- aggregate(Event.ID ~ PID, data = dat_rawEvents_rm, FUN = length)
names(intLoginCount_dat) <- c("PID", "Int_Login_Count")

intdat <- left_join(intdat, intLoginCount_dat, by = "PID")
intdat[is.na(intdat)] <- 0
```

Now, we will compute "Int_Cores_Comp", which is a count of the number of SHUTi Cores completed during the 9-week intervention window. This variable can range from 0 (no SHUTi program logins) to 6 (all SHUTi Cores completed).

```
completed <- "transitioned to Completed"
dat_rawEvents_reduced3<-dat_rawEvents_reduced
dat_rawEvents_reduced3$CoresComp <- ifelse(str_detect(dat_rawEvents_reduced3$Description, completed),
                                           "Yes", "No")
dat_rawEvents_reduced3 <- dat_rawEvents_reduced3[which(dat_rawEvents_reduced3$CoresComp=="Yes"),]

core <- "Activity: Core"
dat_rawEvents_reduced3$Core <- ifelse(str_detect(dat_rawEvents_reduced3$Description, core),
                                      "Yes", "No")
dat_rawEvents_reduced3 <- dat_rawEvents_reduced3[which(dat_rawEvents_reduced3$Core=="Yes"),]


rawDat_corecomp<- aggregate(data = dat_rawEvents_reduced3,
                        Description ~ PID,
                        function(Description) length(unique(Description)))

names(rawDat_corecomp) <- c("PID", "Int_Cores_Comp")

intdat <- left_join(intdat, rawDat_corecomp, by = "PID")
intdat[is.na(intdat)] <- 0
```

Now, we will compute "Tot_Cores_Comp", which is a count of the total number of SHUTi Cores completed. This variable captures the total number of Cores completed, both within and outside the traditional 9-week intervention window. This variable can range from 0 (no SHUTi program logins) to 6 (all SHUTi Cores completed).

```
completed <- "transitioned to Completed"
dat_rawEvents$CoresComp <- ifelse(str_detect(dat_rawEvents$Description, completed),
                                  "Yes", "No")
dat_rawEvents <- dat_rawEvents[which(dat_rawEvents$CoresComp=="Yes"),]

core <- "Activity: Core"
dat_rawEvents$Core <- ifelse(str_detect(dat_rawEvents$Description, core),
                             "Yes", "No")
dat_rawEvents <- dat_rawEvents[which(dat_rawEvents$Core=="Yes"),]


rawDat_corecomptot<- aggregate(data = dat_rawEvents,
                        Description ~ PID,
                        function(Description) length(unique(Description)))
```

```
names(rawDat_corecomptot) <- c("PID", "Tot_Cores_Comp")

intdat <- left_join(intdat, rawDat_corecomptot, by = "PID")
intdat[is.na(intdat)] <- 0
```

Next, we will calculate "PE_initiated", which is a binary variable that reflects whether or not a user ever initiated the Patient Education (PE) control condition program. This variable is scored 1 if the user, at any point during the 9-week intervention period, initiated PE (i.e., logged in and clicked into the program). It is scored 0 if the user never initiated PE during the intervention period.

```
startsWith <- "Patient Education transitioned to In progress"
dat_rawEvents_reduced$PE_intitiation_status <- ifelse(str_detect(dat_rawEvents_reduced$Description,
                                                                startsWith), 1, 0)

PEint_dat <- aggregate(PE_intitiation_status ~ PID, data = dat_rawEvents_reduced, FUN = sum)
names(PEint_dat) <- c("PID", "PE_initiated")

PEint_dat$PE_initiated <- ifelse(PEint_dat$PE_initiated >= 1, 1, 0)

intdat <- left_join(intdat, PEint_dat, by = "PID")
```

Finally, we will calculate "PE_completed", which is a binary variable that reflects whether or not a user ever completed Patient Education (PE) control condition program. This variable is scored 1 if the user, at any point during the 9-week intervention period, completed PE (i.e., clicked through the program and clicked "complete" button at conclusion of program). It is scored 0 if the user never completed PE during the intervention period.

```
startsWith <- "Patient Education transitioned to Completed"
dat_rawEvents_reduced$PE_status <- ifelse(str_detect(dat_rawEvents_reduced$Description,
                                                     startsWith), 1, 0)

PEcomp_dat <- aggregate(PE_status ~ PID, data = dat_rawEvents_reduced, FUN = sum)
names(PEcomp_dat) <- c("PID", "PE_completed")

PEcomp_dat$PE_completed <- ifelse(PEcomp_dat$PE_completed >= 1, 1, 0)

intdat <- left_join(intdat, PEcomp_dat, by = "PID")
```

##----------

Creating Final Datasets for Analysis

##----------

Here, we create a new long (one row per participant per timepoint [e.g., 100 participants x 4 time points = 400 rows]) & wide (one row per participant [e.g., 100 participants x 4 time points = 100 rows with additional columns]) dataset. Datasets will include variables for participants' number of diaries used in the block, the five computed sleep diary metrics, and the four computed engagement metrics.

##EDITING MAY BE REQUIRED: The Wide dataset was written for studies that have 4 time points. If your study has fewer than 4 time points, delete the syntax lines for the additional time points. If you have more than 4 time points, copy and paste the syntax lines for a time point and edit as appropriate.

```
n.block<- grouped.assessment %>%
  summarise(n.block = n())

#Long
final.assessment<- merge(n.block, sol,
                         by = c("PID", "assessment"))
final.assessment<- plyr::join_all(list(final.assessment, wasoema, tst, tib, se, NumAwake, SleepQ),
                            by = c("PID", "assessment"))
final.assessment.long<- final.assessment
final.assessment.long <- left_join(final.assessment.long, intdat, by = "PID")

write.csv(final.assessment.long, "Final_Assessment_Diary_Outcomes_Long.csv", row.names = FALSE)


#Wide
final.t1<- final.assessment %>%
  filter(assessment == "time_1")
final.t2<- final.assessment %>%
  filter(assessment == "time_2")
final.t3<- final.assessment %>%
  filter(assessment == "time_3")
final.t4<- final.assessment %>%
  filter(assessment == "time_4")
final.assessment.wide<- plyr::join_all(list(final.t1, final.t2, final.t3, final.t4),
                                       by = "PID")
colnames(final.assessment.wide)<-c("PID", "assessment.1", "n.block.1", "sol.1",
                                   "wasoema.1", "tst.1", "tib.1", "se.1",
                                   "NumAwake.1", "SleepQ.1", "assessment.2",
                                   "n.block.2", "sol.2", "wasoema.2", "tst.2",
                                   "tib.2", "se.2", "NumAwake.2", "SleepQ.2",
                                   "assessment.3", "n.block.3", "sol.3",
                                   "wasoema.3", "tst.3", "tib.3", "se.3",
                                   "NumAwake.3", "SleepQ.3","assessment.4",
                                   "n.block.4", "sol.4", "wasoema.4", "tst.4",
                                   "tib.4", "se.4", "NumAwake.4", "SleepQ.4")


final.assessment.wide <- left_join(final.assessment.wide, intdat, by = "PID")

write.csv(final.assessment.wide, "Final_Assessment_Diary_Outcomes_Wide.csv", row.names = FALSE)

rm(assessment.data.valid, dat_rawEvents, dat_rawEvents_reduced, dat_rawEvents_reduced_2, dat_rawEvents_
```

You should now review your final data to ensure it meets your expectations (e.g., are there outcomes for participants at expected time points?).

Be sure to save this syntax so you can replicate these analyses as needed. You may then close out of RStudio, and you do not need to save your 'Workspace Image.'

CONGRATULATIONS! You've completed managing your SHUTi for Researchers sleep diary trial data!