Bellabeat Case using R , SQL Server and Tableu

Alonso Zenobio

2024-04-18

Fase Ask

Business Task

Analizar los datos de uso de dispositivos inteligentes no pertenecientes a Bellabeat para conocer el comportamiento y tendencias del usuario común. Los resultados serán compartidos tanto por informe final escrito y en una presentación donde se expondrán las recomendaciones para mejorar las estrategias de marketing o incentivar el desarrollo de nuevos productos.

Stakeholders

La empresa contratista es Bellabeat , manufacturera de productos de salud para mujeres. Los stakeholders clave para el proyecto son:

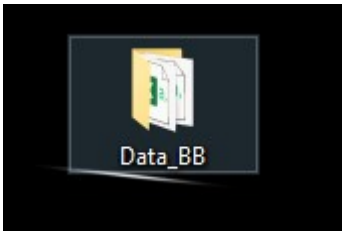
- Urška Sršen: Cofundadora de Bellabeat , es la principal stakeholder ya que plantea los requerimientos del proyecto.
- Equipo de data analytics: Es el area propia de bellabeat , encargada de tareas relativas a recolección , manipulación y análisis de datos.

Fase Prepare

Sobre los datos recolectados

El equipo de data analytics brindo el conjunto de datos Fitness Tracker Data proveniente de Kaggle que contiene datos de 30 usuarios de Fitbit en el periodo de Marzo - Mayo del 2016.

Fueron descargados un total de 18 conjuntos de datos distintos en formatos CSV , almacenados en la carpeta local Data_BB



Cada conjunto de datos fue nombrado bajo la nomenclatura *clasededatos_period(# periodo)* siendo el que hay 2 periodos de recoleccion.

dailyActivity_period1	17/04/2024 15:02	Archivo de valores...	51 KB
dailyActivity_period2	02/03/2024 20:47	Archivo de valores...	109 KB
heartrate_seconds_period1	02/03/2024 20:47	Archivo de valores...	40.108 KB
heartrate_seconds_period2	02/03/2024 20:47	Archivo de valores...	87.489 KB
hourlyCalories_period1	02/03/2024 20:47	Archivo de valores...	853 KB
hourlyCalories_period2	02/03/2024 20:47	Archivo de valores...	783 KB
hourlyIntensities_period1	02/03/2024 20:47	Archivo de valores...	949 KB
hourlyIntensities_period2	02/03/2024 20:47	Archivo de valores...	878 KB
hourlySteps_period1	02/03/2024 20:47	Archivo de valores...	846 KB
hourlySteps_period2	02/03/2024 20:47	Archivo de valores...	778 KB
minute_MET_Narrow_period1	02/03/2024 20:47	Archivo de valores...	50.753 KB
minute_MET_Narrow_period2	02/03/2024 20:47	Archivo de valores...	46.570 KB
minute_Steps_Narrow_period1	02/03/2024 20:47	Archivo de valores...	49.505 KB
minute_Steps_Narrow_period2	02/03/2024 20:47	Archivo de valores...	45.442 KB
minuteCalories_Narrow_period1	02/03/2024 20:47	Archivo de valores...	70.764 KB
minuteCalories_Narrow_period2	02/03/2024 20:47	Archivo de valores...	64.887 KB
minuteIntensities_Narrow_period1	02/03/2024 20:47	Archivo de valores...	49.340 KB
minuteIntensities_Narrow_period2	02/03/2024 20:47	Archivo de valores...	45.273 KB
minuteSleep_period1	02/03/2024 20:47	Archivo de valores...	9.098 KB
minuteSleep_period2	02/03/2024 20:47	Archivo de valores...	8.641 KB
weight_Log_Info_period1	02/03/2024 20:47	Archivo de valores...	4 KB
weight_Log_Info_period2	02/03/2024 20:47	Archivo de valores...	7 KB

Caracteristicas de los conjuntos:

- Han sido recolectados 22 conjuntos de datos en total , siendo en realidad 11 conjuntos que fueron divididos en 2 periodos de recoleccion distintos
- Los conjuntos de datos estan en formato Long
- Fueron descartados 7 conjuntos de datos pertenecientes al periodo 12-04-2016 al 12-05-2016 debido a la falta de datos por parte del periodo anterior
- Debido a limitaciones temporales , solo se estaran usando un maximo de 2 conjuntos de datos finales para etapas de analisis y visualizacion.

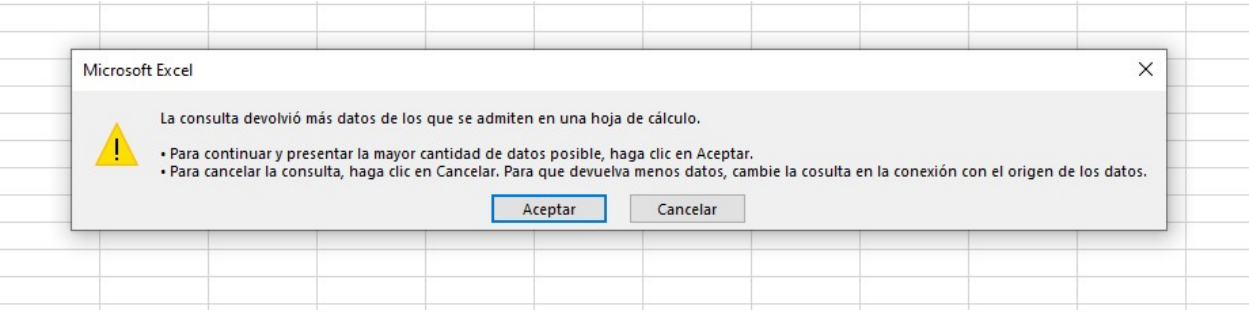
Descripcion de los conjuntos utilizados:

Nombre del conjunto de datos	Descripcion
dailyActivity_period[X]	Conjunto de datos que recolecta actividad diaria del usuario segun Id
heartrate_seconds_period[X]	Conjunto de datos que recolecta ritmo cardiaco por segundos
hourlyCalories_period[X]	Conjunto de datos que registra Calorias quemadas por hora
hourlyIntensities_period[X]	Conjunto de datos que registra la intensidad del ejercicio por hora
hourlySteps_period[X]	Conjunto de datos que registra pasos realizados por hora
minute_MET_Narrow_period[X]	Conjunto de datos que registra radio de ritmo metabolico angosto por minuto
minute_Steps_Narrow_period[x]	Conjunto de datos que registra pasos angostos por minuto
minuteCalories_Narrow_period[x]	Conjunto de datos que registra calorias quemadas por minuto
minute_Intensities_Narrow_period[X]	Conjunto de datos que registra intensidad del ejercicio angosto por minuto
minuteSleep_period[X]	Conjunto de datos de registra minutos de sueño
weight_Log_Info_period[X]	Conjunto de datos que registra peso del usuario por sesion

Fase Process

Para empezar con la fase de limpieza se hizo uso de la herramienta Excel para una rapida revision de los datos. Se notaron las siguientes caracteristicas:

- Conjuntos de datos como DailyActivity no conllevan tantas instancias y pueden ser cargadas en Excel.
- Gran parte de los conjuntos de datos contienen una cantidad de registros mayor al limite de excel dando el siguiente mensaje:



- No se detectan datos nulos en gran parte de los conjuntos , en caso de que no haya registros simplemente el valor es dejado en 0 , lo que no representa realmente falta de datos sino que no hubo actividad registrada en esa fecha.

Considerando lo mencionado , se indica lo siguiente:

- Se hara uso del Lenguaje R para la limpieza de datos , ya que se requiere juntar archivos CSV de forma sencilla y rapida.
- Debido al limite de Excel no es adecuado hacer uso de esta herramienta para un proceso de limpieza , en el caso de SQL este no admite el formato en el cual se registran las fechas en la importacion.
- El proceso de limpieza busca asegurar la integridad de los datos , en este caso , se requieren acciones para combinar datos , cambios de formatos y ordenamiento.
- Posterior al proceso de limpieza en R , se debe de llevar la carga de archivos a una base de datos en SQL Server

Limpieza en R

Para empezar el proceso de limpieza se debe empezar con la instalacion de librerias en el equipo

```
options(repos = "https://cloud.r-project.org")

install.packages("tidyverse")
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\alonx\AppData\Local\Temp\RtmpEPttrA\downloaded_packages
```

```
install.packages("here")
```

```
## package 'here' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\alonx\AppData\Local\Temp\RtmpEPttrA\downloaded_packages
```

```
install.packages("skimr")
```

```
## package 'skimr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\alonx\AppData\Local\Temp\RtmpEPttrA\downloaded_packages
```

```
install.packages("janitor")
```

```
## package 'janitor' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\alonx\AppData\Local\Temp\RtmpEPttrA\downloaded_packages
```

Posterior a ello se cargan en el entorno

Luego se debe realizar la carga de datos , con fines practicos se asigna la variable global ruta que contiene la ruta de almacenamiento como texto , no se mostrara su valor por razones de privacidad

```r
#Se reemplaza la ruta por la local
# ruta <- "ruta local o publica"

# Carga de todos los conjuntos de datos csv
da1 <- read_csv(paste(ruta,"dailyActivity_period1.csv",sep = ""))
da2  <- read_csv(paste(ruta,"dailyActivity_period2.csv",sep = ""))
hs1 <- read_csv(paste(ruta,"heartrate_seconds_period1.csv",sep = ""))
hs2 <- read_csv(paste(ruta,"heartrate_seconds_period2.csv",sep = ""))
hc1 <- read_csv(paste(ruta,"hourlyCalories_period1.csv",sep = ""))
hc2 <- read_csv(paste(ruta,"hourlyCalories_period2.csv",sep = ""))
hi1 <- read_csv(paste(ruta,"hourlyIntensities_period1.csv",sep = ""))
hi2 <- read_csv(paste(ruta,"hourlyIntensities_period1.csv",sep = ""))
hst1 <- read_csv(paste(ruta,"hourlySteps_period1.csv",sep = ""))
hst2 <- read_csv(paste(ruta,"hourlySteps_period2.csv",sep = ""))
mmn1 <- read_csv(paste(ruta,"minute_MET_Narrow_period1.csv",sep = ""))
mmn2 <- read_csv(paste(ruta,"minute_MET_Narrow_period2.csv",sep = ""))
mcn1 <- read_csv(paste(ruta,"minuteCalories_Narrow_period1.csv",sep = ""))
mcn2 <- read_csv(paste(ruta,"minuteCalories_Narrow_period2.csv",sep = ""))
min1 <- read_csv(paste(ruta,"minuteIntensities_Narrow_period1.csv",sep = ""))
min2 <- read_csv(paste(ruta,"minuteIntensities_Narrow_period2.csv",sep = ""))
ms1 <- read_csv(paste(ruta,"minuteSleep_period1.csv",sep = ""))
ms2 <- read_csv(paste(ruta,"minuteSleep_period2.csv",sep = ""))
wl1 <- read_csv(paste(ruta,"weight_Log_Info_period1.csv",sep = ""))
wl2 <- read_csv(paste(ruta,"weight_Log_Info_period2.csv",sep = ""))
mst1 <- read_csv(paste(ruta,"minute_Steps_Narrow_period1.csv",sep = ""))
mst2 <- read_csv(paste(ruta,"minute_Steps_Narrow_period2.csv",sep = ""))
```

Una vez realizada la carga de archivos , se procede a crear archivos ya unidos

```r
dailyActivity <- rbind(da1,da2)
heartrateSeconds <- rbind(hs1,hs2)
hourlyCalories <- rbind(hc1,hc2)
hourlyIntensities <- rbind(hi1,hi2)
hourlySteps <- rbind(hst1,hst2)
minuteMET <- rbind(mmn1,mmn2)
minuteSteps <- rbind(mst1,mst2)
minuteCalories <- rbind(mcn1,mcn2)
minuteIntensities <- rbind(min1,min2)
minuteSleep <- rbind(ms1,ms2)
weightInfo <- rbind(wl1,wl2)
```

Analisis exploratorio de datos

Es necesario tener una vista previa de los conjuntos de datos antes de realizar algun cambio:

```r
# Conjunto dailyActivity
head(dailyActivity)
```

```
## # A tibble: 6 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <chr>             <dbl>         <dbl>           <dbl>
## 1 1503960366 3/25/2016         11004          7.11            7.11
```

```
## 2 1503960366 3/26/2016         17609         11.6            11.6
## 3 1503960366 3/27/2016         12736          8.53            8.53
## 4 1503960366 3/28/2016         13231          8.93            8.93
## 5 1503960366 3/29/2016         12041          7.85            7.85
## 6 1503960366 3/30/2016         10970          7.16            7.16
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
#Conjunto de datos HeartrateSeconds
head(heartrateSeconds)
```

```
## # A tibble: 6 x 3
##           Id Time                 Value
##        <dbl> <chr>                <dbl>
## 1 2022484408 4/1/2016 7:54:00 AM    93
## 2 2022484408 4/1/2016 7:54:05 AM    91
## 3 2022484408 4/1/2016 7:54:10 AM    96
## 4 2022484408 4/1/2016 7:54:15 AM    98
## 5 2022484408 4/1/2016 7:54:20 AM   100
## 6 2022484408 4/1/2016 7:54:25 AM   101
```

```
#Conjunto de datos HourlyCalories
head(hourlyCalories)
```

```
## # A tibble: 6 x 3
##           Id ActivityHour          Calories
##        <dbl> <chr>                    <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM       48
## 2 1503960366 3/12/2016 1:00:00 AM        48
## 3 1503960366 3/12/2016 2:00:00 AM        48
## 4 1503960366 3/12/2016 3:00:00 AM        48
## 5 1503960366 3/12/2016 4:00:00 AM        48
## 6 1503960366 3/12/2016 5:00:00 AM        48
```

```
#Conjunto de datos HourlyIntensities
head(hourlyIntensities)
```

```
## # A tibble: 6 x 4
##           Id ActivityHour          TotalIntensity AverageIntensity
##        <dbl> <chr>                          <dbl>             <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM             0                 0
## 2 1503960366 3/12/2016 1:00:00 AM              0                 0
## 3 1503960366 3/12/2016 2:00:00 AM              0                 0
## 4 1503960366 3/12/2016 3:00:00 AM              0                 0
## 5 1503960366 3/12/2016 4:00:00 AM              0                 0
## 6 1503960366 3/12/2016 5:00:00 AM              0                 0
```

```r
#Conjunto de datos HourlySteps
head(hourlySteps)
```

```
## # A tibble: 6 x 3
##           Id ActivityHour          StepTotal
##        <dbl> <chr>                     <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM         0
## 2 1503960366 3/12/2016 1:00:00 AM          0
## 3 1503960366 3/12/2016 2:00:00 AM          0
## 4 1503960366 3/12/2016 3:00:00 AM          0
## 5 1503960366 3/12/2016 4:00:00 AM          0
## 6 1503960366 3/12/2016 5:00:00 AM          0
```

```r
#Conjunto de datos MinuteMET
head(minuteMET)
```

```
## # A tibble: 6 x 3
##           Id ActivityMinute         METs
##        <dbl> <chr>                 <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM    10
## 2 1503960366 3/12/2016 12:01:00 AM    10
## 3 1503960366 3/12/2016 12:02:00 AM    10
## 4 1503960366 3/12/2016 12:03:00 AM    10
## 5 1503960366 3/12/2016 12:04:00 AM    10
## 6 1503960366 3/12/2016 12:05:00 AM    10
```

```r
#Conjunto de datos MinuteSteps
head(minuteSteps)
```

```
## # A tibble: 6 x 3
##           Id ActivityMinute        Steps
##        <dbl> <chr>                 <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM     0
## 2 1503960366 3/12/2016 12:01:00 AM     0
## 3 1503960366 3/12/2016 12:02:00 AM     0
## 4 1503960366 3/12/2016 12:03:00 AM     0
## 5 1503960366 3/12/2016 12:04:00 AM     0
## 6 1503960366 3/12/2016 12:05:00 AM     0
```

```r
#Conjunto de datos MinuteCalories
head(minuteCalories)
```

```
## # A tibble: 6 x 3
##           Id ActivityMinute        Calories
##        <dbl> <chr>                    <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM    0.797
## 2 1503960366 3/12/2016 12:01:00 AM    0.797
## 3 1503960366 3/12/2016 12:02:00 AM    0.797
## 4 1503960366 3/12/2016 12:03:00 AM    0.797
## 5 1503960366 3/12/2016 12:04:00 AM    0.797
## 6 1503960366 3/12/2016 12:05:00 AM    0.797
```

```r
#Conjunto de datos MinuteIntensities
head(minuteIntensities)
```

```
## # A tibble: 6 x 3
##           Id ActivityMinute        Intensity
##        <dbl> <chr>                     <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM         0
## 2 1503960366 3/12/2016 12:01:00 AM         0
## 3 1503960366 3/12/2016 12:02:00 AM         0
## 4 1503960366 3/12/2016 12:03:00 AM         0
## 5 1503960366 3/12/2016 12:04:00 AM         0
## 6 1503960366 3/12/2016 12:05:00 AM         0
```

```r
#Conjunto de datos MinuteSleep
head(minuteSleep)
```

```
## # A tibble: 6 x 4
##           Id date                 value       logId
##        <dbl> <chr>                <dbl>       <dbl>
## 1 1503960366 3/13/2016 2:39:30 AM     1 11114919637
## 2 1503960366 3/13/2016 2:40:30 AM     1 11114919637
## 3 1503960366 3/13/2016 2:41:30 AM     1 11114919637
## 4 1503960366 3/13/2016 2:42:30 AM     1 11114919637
## 5 1503960366 3/13/2016 2:43:30 AM     1 11114919637
## 6 1503960366 3/13/2016 2:44:30 AM     1 11114919637
```

```r
#Conjunto de datos WeightLog
head(weightInfo)
```

```
## # A tibble: 6 x 8
##           Id Date       WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##        <dbl> <chr>         <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1 1503960366 4/5/2016 ~     53.3         118.    22  23.0 TRUE           1.46e12
## 2 1927972279 4/10/2016~    130.          286.    NA  46.2 FALSE          1.46e12
## 3 2347167796 4/3/2016 ~     63.4         140.    10  24.8 TRUE           1.46e12
## 4 2873212765 4/6/2016 ~     56.7         125.    NA  21.5 TRUE           1.46e12
## 5 2873212765 4/7/2016 ~     57.2         126.    NA  21.6 TRUE           1.46e12
## 6 2891001357 4/5/2016 ~     88.4         195.    NA  25.0 TRUE           1.46e12
```

Revisamos la estructura de cada conjunto de datos

```r
str(dailyActivity)
```

```
## spc_tbl_ [1,397 x 15] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id                       : num [1:1397] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityDate             : chr [1:1397] "3/25/2016" "3/26/2016" "3/27/2016" "3/28/2016" ...
##  $ TotalSteps               : num [1:1397] 11004 17609 12736 13231 12041 ...
##  $ TotalDistance            : num [1:1397] 7.11 11.55 8.53 8.93 7.85 ...
##  $ TrackerDistance          : num [1:1397] 7.11 11.55 8.53 8.93 7.85 ...
##  $ LoggedActivitiesDistance: num [1:1397] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance       : num [1:1397] 2.57 6.92 4.66 3.19 2.16 ...
```

```
##  $ ModeratelyActiveDistance: num [1:1397] 0.46 0.73 0.16 0.79 1.09 ...
##  $ LightActiveDistance     : num [1:1397] 4.07 3.91 3.71 4.95 4.61 ...
##  $ SedentaryActiveDistance : num [1:1397] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes       : num [1:1397] 33 89 56 39 28 30 33 47 40 15 ...
##  $ FairlyActiveMinutes     : num [1:1397] 12 17 5 20 28 13 12 21 11 30 ...
##  $ LightlyActiveMinutes    : num [1:1397] 205 274 268 224 243 223 239 200 244 314 ...
##  $ SedentaryMinutes        : num [1:1397] 804 588 605 1080 763 ...
##  $ Calories                : num [1:1397] 1819 2154 1944 1932 1886 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityDate = col_character(),
##   ..   TotalSteps = col_double(),
##   ..   TotalDistance = col_double(),
##   ..   TrackerDistance = col_double(),
##   ..   LoggedActivitiesDistance = col_double(),
##   ..   VeryActiveDistance = col_double(),
##   ..   ModeratelyActiveDistance = col_double(),
##   ..   LightActiveDistance = col_double(),
##   ..   SedentaryActiveDistance = col_double(),
##   ..   VeryActiveMinutes = col_double(),
##   ..   FairlyActiveMinutes = col_double(),
##   ..   LightlyActiveMinutes = col_double(),
##   ..   SedentaryMinutes = col_double(),
##   ..   Calories = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(heartrateSeconds)
```

```
## spc_tbl_ [3,638,339 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id   : num [1:3638339] 2.02e+09 2.02e+09 2.02e+09 2.02e+09 2.02e+09 ...
##  $ Time : chr [1:3638339] "4/1/2016 7:54:00 AM" "4/1/2016 7:54:05 AM" "4/1/2016 7:54:10 AM" "4/1/201
##  $ Value: num [1:3638339] 93 91 96 98 100 101 104 105 102 106 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   Time = col_character(),
##   ..   Value = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(hourlyCalories)
```

```
## spc_tbl_ [46,183 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id          : num [1:46183] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityHour: chr [1:46183] "3/12/2016 12:00:00 AM" "3/12/2016 1:00:00 AM" "3/12/2016 2:00:00 AM"
##  $ Calories    : num [1:46183] 48 48 48 48 48 48 48 48 48 49 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityHour = col_character(),
##   ..   Calories = col_double()
```

```
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(hourlyIntensities)
```

```
## spc_tbl_ [48,168 x 4] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id              : num [1:48168] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityHour    : chr [1:48168] "3/12/2016 12:00:00 AM" "3/12/2016 1:00:00 AM" "3/12/2016 2:00:00
##  $ TotalIntensity  : num [1:48168] 0 0 0 0 0 0 0 0 0 1 ...
##  $ AverageIntensity: num [1:48168] 0 0 0 0 0 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityHour = col_character(),
##   ..   TotalIntensity = col_double(),
##   ..   AverageIntensity = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(hourlySteps)
```

```
## spc_tbl_ [46,183 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id          : num [1:46183] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityHour: chr [1:46183] "3/12/2016 12:00:00 AM" "3/12/2016 1:00:00 AM" "3/12/2016 2:00:00 AM"
##  $ StepTotal   : num [1:46183] 0 0 0 0 0 0 0 0 0 8 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityHour = col_character(),
##   ..   StepTotal = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(minuteMET)
```

```
## spc_tbl_ [2,770,620 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id            : num [1:2770620] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityMinute: chr [1:2770620] "3/12/2016 12:00:00 AM" "3/12/2016 12:01:00 AM" "3/12/2016 12:02:
##  $ METs          : num [1:2770620] 10 10 10 10 10 10 10 10 10 10 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityMinute = col_character(),
##   ..   METs = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(minuteSteps)
```

```
## spc_tbl_ [2,770,620 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id            : num [1:2770620] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
```

```
##  $ ActivityMinute: chr [1:2770620] "3/12/2016 12:00:00 AM" "3/12/2016 12:01:00 AM" "3/12/2016 12:02:0
##  $ Steps         : num [1:2770620] 0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityMinute = col_character(),
##   ..   Steps = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(minuteCalories)
```

```
## spc_tbl_ [2,770,620 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id            : num [1:2770620] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityMinute: chr [1:2770620] "3/12/2016 12:00:00 AM" "3/12/2016 12:01:00 AM" "3/12/2016 12:02:0
##  $ Calories      : num [1:2770620] 0.797 0.797 0.797 0.797 0.797 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityMinute = col_character(),
##   ..   Calories = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(minuteIntensities)
```

```
## spc_tbl_ [2,770,620 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id            : num [1:2770620] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityMinute: chr [1:2770620] "3/12/2016 12:00:00 AM" "3/12/2016 12:01:00 AM" "3/12/2016 12:02:0
##  $ Intensity     : num [1:2770620] 0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityMinute = col_character(),
##   ..   Intensity = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(minuteSleep)
```

```
## spc_tbl_ [387,080 x 4] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id   : num [1:387080] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ date : chr [1:387080] "3/13/2016 2:39:30 AM" "3/13/2016 2:40:30 AM" "3/13/2016 2:41:30 AM" "3/13/1
##  $ value: num [1:387080] 1 1 1 1 1 1 2 2 1 1 ...
##  $ logId: num [1:387080] 1.11e+10 1.11e+10 1.11e+10 1.11e+10 1.11e+10 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   date = col_character(),
##   ..   value = col_double(),
##   ..   logId = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(weightInfo)
```

```
## spc_tbl_ [100 x 8] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id            : num [1:100] 1.50e+09 1.93e+09 2.35e+09 2.87e+09 2.87e+09 ...
##  $ Date          : chr [1:100] "4/5/2016 11:59:59 PM" "4/10/2016 6:33:26 PM" "4/3/2016 11:59:59 PM" "
##  $ WeightKg      : num [1:100] 53.3 129.6 63.4 56.7 57.2 ...
##  $ WeightPounds  : num [1:100] 118 286 140 125 126 ...
##  $ Fat           : num [1:100] 22 NA 10 NA NA NA NA NA NA NA ...
##  $ BMI           : num [1:100] 23 46.2 24.8 21.5 21.6 ...
##  $ IsManualReport: logi [1:100] TRUE FALSE TRUE TRUE TRUE TRUE ...
##  $ LogId         : num [1:100] 1.46e+12 1.46e+12 1.46e+12 1.46e+12 1.46e+12 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   Date = col_character(),
##   ..   WeightKg = col_double(),
##   ..   WeightPounds = col_double(),
##   ..   Fat = col_double(),
##   ..   BMI = col_double(),
##   ..   IsManualReport = col_logical(),
##   ..   LogId = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Notamos que gran parte de los datos concuerdan con los tipos que representan , por lo que no se vera incluido algun proceso de transformacion de tipo. Antes de iniciar con ajustes de formato , veremos si los conjuntos de datos presentan gran cantidad de nulos.

```
sum(is.na(dailyActivity))
```

```
## [1] 0
```

```
sapply(dailyActivity, function(x) sum(is.na(x)))
```

```
##                       Id             ActivityDate               TotalSteps
##                        0                        0                        0
##            TotalDistance          TrackerDistance LoggedActivitiesDistance
##                        0                        0                        0
##       VeryActiveDistance ModeratelyActiveDistance      LightActiveDistance
##                        0                        0                        0
##  SedentaryActiveDistance        VeryActiveMinutes      FairlyActiveMinutes
##                        0                        0                        0
##     LightlyActiveMinutes         SedentaryMinutes                 Calories
##                        0                        0                        0
```

```
sum(is.na(heartrateSeconds))
```

```
## [1] 0
```

```r
sapply(heartrateSeconds, function(x) sum(is.na(x)))
```

```
##    Id  Time Value
##     0     0     0
```

```r
sum(is.na(hourlyCalories))
```

```
## [1] 0
```

```r
sapply(hourlyCalories, function(x) sum(is.na(x)))
```

```
##           Id ActivityHour     Calories
##            0            0            0
```

```r
sum(is.na(hourlyIntensities))
```

```
## [1] 0
```

```r
sapply(hourlyIntensities, function(x) sum(is.na(x)))
```

```
##               Id     ActivityHour   TotalIntensity AverageIntensity
##                0                0                0                0
```

```r
sum(is.na(hourlySteps))
```

```
## [1] 0
```

```r
sapply(hourlySteps, function(x) sum(is.na(x)))
```

```
##           Id ActivityHour    StepTotal
##            0            0            0
```

```r
sum(is.na(minuteMET))
```

```
## [1] 0
```

```r
sapply(minuteMET, function(x) sum(is.na(x)))
```

```
##             Id ActivityMinute           METs
##              0              0              0
```

```r
sum(is.na(minuteSteps))
```

```
## [1] 0
```

```r
sapply(minuteSteps, function(x) sum(is.na(x)))
```

```
##             Id ActivityMinute          Steps
##              0              0              0
```

```r
sum(is.na(minuteCalories))
```

```
## [1] 0
```

```r
sapply(minuteCalories, function(x) sum(is.na(x)))
```

```
##             Id ActivityMinute       Calories
##              0              0              0
```

```r
sum(is.na(minuteIntensities))
```

```
## [1] 0
```

```r
sapply(minuteIntensities, function(x) sum(is.na(x)))
```

```
##             Id ActivityMinute      Intensity
##              0              0              0
```

```r
sum(is.na(minuteSleep))
```

```
## [1] 0
```

```r
sapply(minuteSleep, function(x) sum(is.na(x)))
```

```
##    Id  date value logId
##     0     0     0     0
```

```r
sum(is.na(weightInfo))
```

```
## [1] 96
```

```r
sapply(weightInfo, function(x) sum(is.na(x)))
```

```
##             Id           Date       WeightKg   WeightPounds            Fat
##              0              0              0              0             96
##            BMI IsManualReport          LogId
##              0              0              0
```

La mayoria de los conjuntos de datos no contienen columnas con valores nulos , a excepcion del conjunto weightInfo , que recolecta informacion del peso de los usuarios. Este conjunto presenta 96 valores nulos en la columna Fat , por lo tanto se decide:

- Eliminar la columna Fat debido a que el total de registros es de 100 siendo un 96% de datos faltantes
- Eliminar la columna ManualRecord , debido a que no aporta datos relevantes al analisis

```
#Con esto elegimos mantener el conjunto de datos a excepcion de las columnas Fat y IsManualReport
weightInfo <- select(weightInfo,-Fat, -IsManualReport)
```

Otro ajuste necesario es el de las fechas , ya que este se encuentra en un formato dd/mm/yyyy hh:mm:ss PM-AM . Lo mas comun seria ponerlo en un formato de 24 horas generico

```
# Conjunto de datos dailyActivity

# En esta linea se indica que la columna ActivityDate se interprete como fecha y el formato actual
dailyActivity$ActivityDate <- as_date(dailyActivity$ActivityDate,format="%m/%d/%Y")
#Esta linea modifica el formato a una mas generico
dailyActivity$ActivityDate <- format(dailyActivity$ActivityDate,"%d-%m-%Y")

#Esto se aplica para el resto de conjuntos , excepto que algunos seran de tipo datetime

# Conjunto de datos heartrateSeconds
heartrateSeconds$Time <- strptime(heartrateSeconds$Time,"%m/%d/%Y %I:%M:%S %p")
heartrateSeconds$Time <- format(heartrateSeconds$Time,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos hourlyCalories
hourlyCalories$ActivityHour<- strptime(hourlyCalories$ActivityHour,"%m/%d/%Y %I:%M:%S %p")
hourlyCalories$ActivityHour <- format(hourlyCalories$ActivityHour,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos hourlyIntensities
hourlyIntensities$ActivityHour<- strptime(hourlyIntensities$ActivityHour,"%m/%d/%Y %I:%M:%S %p")
hourlyIntensities$ActivityHour <- format(hourlyIntensities$ActivityHour,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos hourlySteps
hourlySteps$ActivityHour<- strptime(hourlySteps$ActivityHour,"%m/%d/%Y %I:%M:%S %p")
hourlySteps$ActivityHour <- format(hourlySteps$ActivityHour,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos minuteMET
minuteMET$ActivityMinute <- strptime(minuteMET$ActivityMinute,"%m/%d/%Y %I:%M:%S %p")
minuteMET$ActivityMinute <- format(minuteMET$ActivityMinute,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos minuteSteps
minuteSteps$ActivityMinute <- strptime(minuteSteps$ActivityMinute ,"%m/%d/%Y %I:%M:%S %p")
minuteSteps$ActivityMinute <- format(minuteSteps$ActivityMinute,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos minuteCalories
minuteCalories$ActivityMinute <- strptime(minuteCalories$ActivityMinute ,"%m/%d/%Y %I:%M:%S %p")
minuteCalories$ActivityMinute <- format(minuteCalories$ActivityMinute,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos minuteIntensities
minuteIntensities$ActivityMinute <- strptime(minuteIntensities$ActivityMinute ,"%m/%d/%Y %I:%M:%S %p")
minuteIntensities$ActivityMinute <- format(minuteIntensities$ActivityMinute,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos minuteSleep
minuteSleep$date <- strptime(minuteSleep$date ,"%m/%d/%Y %I:%M:%S %p")
minuteSleep$date <- format(minuteSleep$date,"%Y-%m-%d %H:%M:%S")

# Conjunto de datos weightInfo
weightInfo$Date <- strptime(weightInfo$Date,"%m/%d/%Y %I:%M:%S %p")
```

```
weightInfo$Date <- format(weightInfo$Date,"%Y-%m-%d %H:%M:%S")
```

El ultimo cambio por parte de R seria el cambio de nombre en ciertas columnas , para brindar mejor formato y claridad en cada conjunto de datos

```
# Conjunto de datos dailyActivity
colnames(dailyActivity) #No hay cambios , formatos aptos
```

```
##  [1] "Id"                       "ActivityDate"
##  [3] "TotalSteps"               "TotalDistance"
##  [5] "TrackerDistance"          "LoggedActivitiesDistance"
##  [7] "VeryActiveDistance"       "ModeratelyActiveDistance"
##  [9] "LightActiveDistance"      "SedentaryActiveDistance"
## [11] "VeryActiveMinutes"        "FairlyActiveMinutes"
## [13] "LightlyActiveMinutes"     "SedentaryMinutes"
## [15] "Calories"
```

```
# Conjunto de datos heartrateSeconds
colnames(heartrateSeconds)
```

```
## [1] "Id"    "Time"  "Value"
```

```
names(heartrateSeconds) <- c("Id","Record","Heartrate")

# Conjunto de datos hourlyCalories
colnames(hourlyCalories)
```

```
## [1] "Id"           "ActivityHour" "Calories"
```

```
names(hourlyCalories) <- c("Id","Record","Calories")

# Conjunto de datos hourlyIntensities
colnames(hourlyIntensities)
```

```
## [1] "Id"               "ActivityHour"     "TotalIntensity"   "AverageIntensity"
```

```
names(hourlyIntensities) <- c("Id","Record","Total_Intensity" , "Avg_Intensity")

# Conjunto de datos hourlySteps
colnames(hourlySteps)
```

```
## [1] "Id"           "ActivityHour" "StepTotal"
```

```
names(hourlySteps) <- c("Id","Record","Total Steps")

# Conjunto de datos minuteMET
colnames(minuteMET)
```

```
## [1] "Id"             "ActivityMinute" "METs"
```

```r
names(minuteMET) <- c("Id","Record","METs")

# Conjunto de datos minuteSteps
colnames(minuteSteps)
```

```
## [1] "Id"             "ActivityMinute" "Steps"
```

```r
names(minuteSteps) <- c("Id","Record","Steps")

# Conjunto de datos minuteCalories
colnames(minuteCalories)
```

```
## [1] "Id"             "ActivityMinute" "Calories"
```

```r
names(minuteCalories) <- c("Id","Record","Calories")

# Conjunto de datos minuteIntensities
colnames(minuteIntensities)
```

```
## [1] "Id"             "ActivityMinute" "Intensity"
```

```r
names(minuteIntensities) <- c("Id","Record","Intensity")

# Conjunto de datos minuteSleep
colnames(minuteSleep)
```

```
## [1] "Id"    "date"  "value" "logId"
```

```r
names(minuteSleep) <- c("Id","Record","Value","LogId")

# Conjunto de datos weightInfo
colnames(weightInfo)
```

```
## [1] "Id"           "Date"         "WeightKg"     "WeightPounds" "BMI"
## [6] "LogId"
```

```r
names(weightInfo) <- c("Id","Record","Weight Kg","Weight Pounds","BMI")
```

El cambio mas relevante es el de colocar cada registro de fecha como Record , esto debido a que colocar Date no es algo correcto , pues cada dato representa una instancia de registro que puede contener varias veces una misma fecha en distintos periodos de tiempo.

Finalmente , se cargan los conjuntos de datos a la ruta de almacenamiento local:

```r
write.csv(dailyActivity,paste("../Data/","DailyActivity.csv",sep = ""),row.names=FALSE)
write.csv(heartrateSeconds,paste("../Data/","HeartrateSeconds.csv",sep = ""),row.names=FALSE)
write.csv(hourlyCalories,paste("../Data/","HourlyCalories.csv",sep = ""),row.names=FALSE)
write.csv(hourlyIntensities,paste("../Data/","HourlyIntensities.csv",sep = ""),row.names=FALSE)
write.csv(hourlySteps,paste("../Data/","HourlySteps.csv",sep = ""),row.names=FALSE)
write.csv(minuteMET,paste("../Data/","MinuteMET.csv",sep = ""),row.names=FALSE)
```

```
write.csv(minuteSteps,paste("../Data/","MinuteSteps.csv",sep = ""),row.names=FALSE)
write.csv(minuteCalories,paste("../Data/","MinuteCalories.csv",sep = ""),row.names=FALSE)
write.csv(minuteIntensities,paste("../Data/","MinuteIntensities.csv",sep = ""),row.names=FALSE)
write.csv(minuteSleep,paste("../Data/","MinuteSleep.csv",sep = ""),row.names=FALSE)
write.csv(weightInfo,paste("../Data/","WeightInfo.csv",sep = ""),row.names=FALSE)
```

Carga en SQL Server

Para este caso de estudio , se eligio realizar la fase Analize en SQL Server , un sistema gestor bastante comun y de uso empresarial , haciendo uso de SQL Server Management Studio se hizo la carga de los archivos CSV considerando:

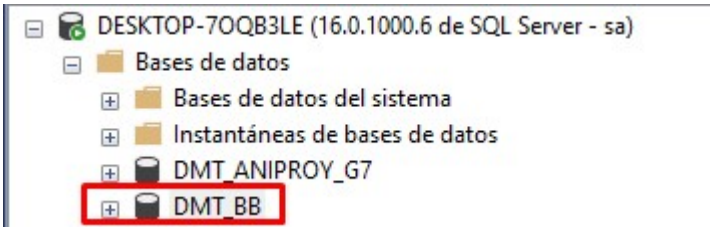
- El uso del Wizard para archivos planos CSV
- El uso del wizard para carga de datos general

La razon por la que no fue usada solo una opcion fue debido a ciertos inconvenientes:

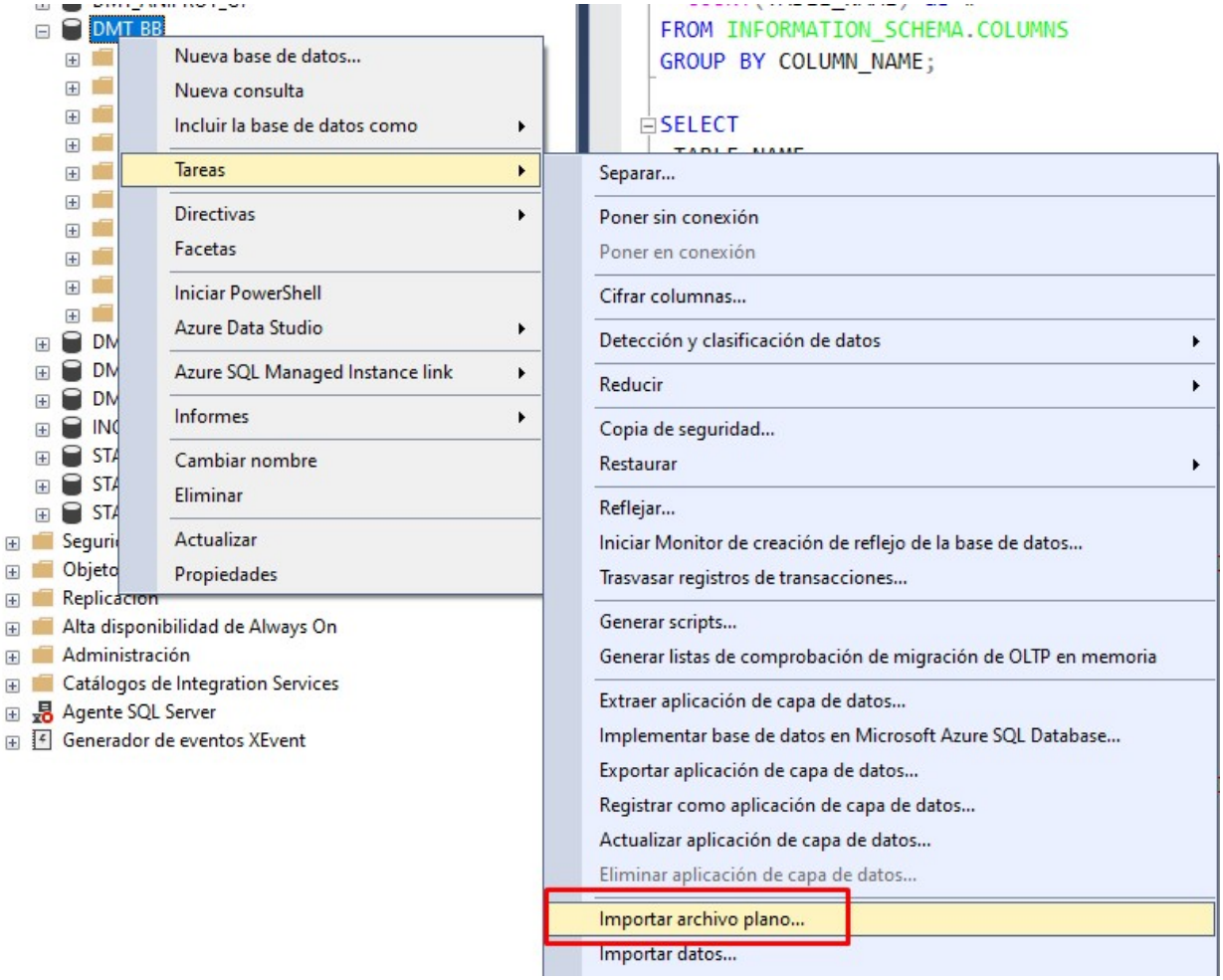
- En la carga de archivos CSV , se reconocen los datos datetime como datetime2 sin ningum problema , pero no reconoce los datos con punto decimal (.) correctamente.
- En la carga de datos general si se reconocen los tipos de datos con punto decimal de forma correcta , pero los datos de tipo fecha no son reconocidos.

A continuacion , se detalla el proceso a seguir para cada archivo CSV:

1. Crear una base de datos , para nuestro caso se crea una base de datos llamada DMT_BB (DataMart Bellabeat)



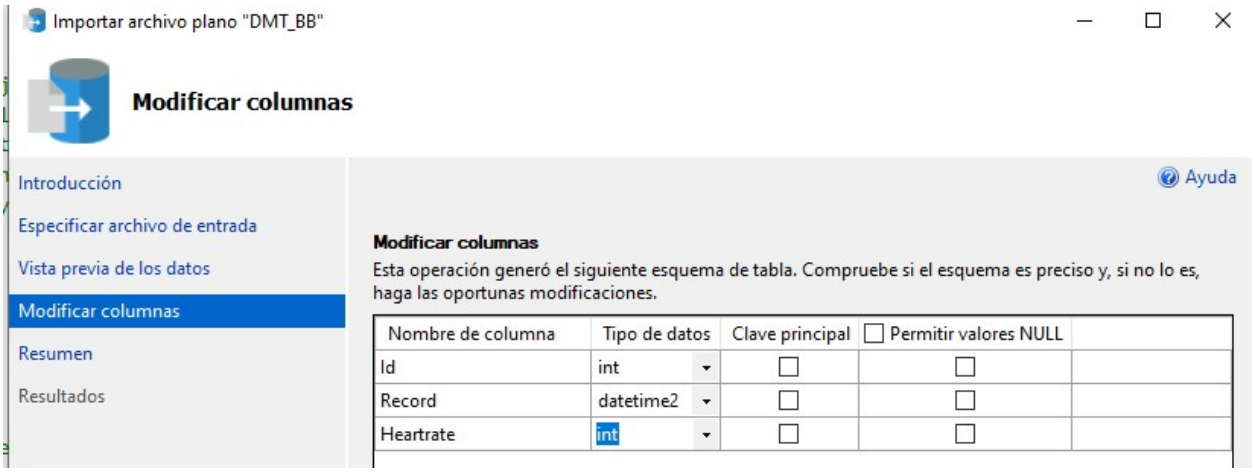
2. Usaremos la opcion para carga de archivos CSV:



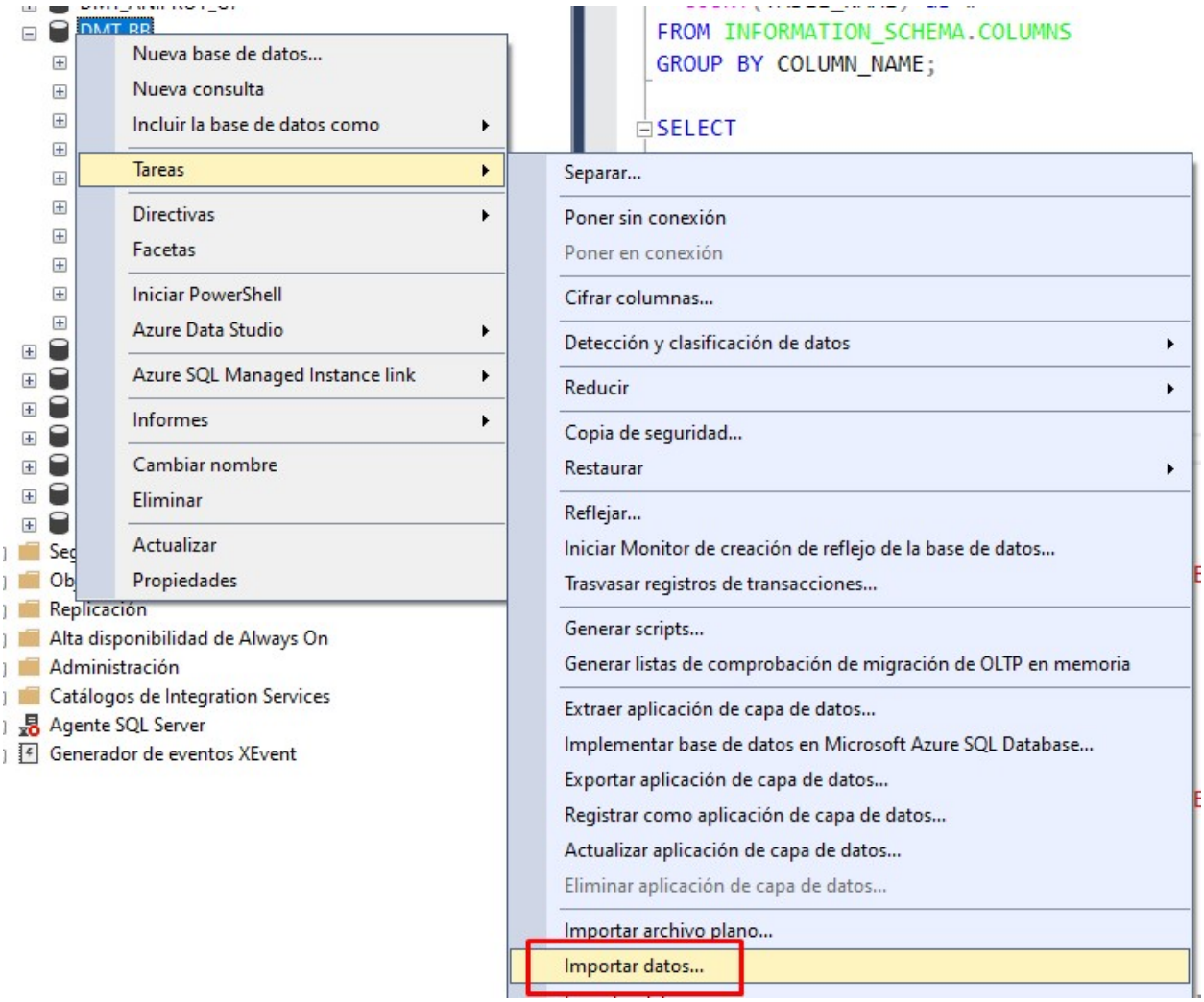
Esta opcion se usara para los siguientes archivos:

- HeartrateSeconds.csv
- HourlyCalories.csv
- HourlySteps.csv
- MinuteIntensities.csv
- MinuteMET.csv
- MinuteSteps.csv
- MinuteSleep.csv

3. Configurar la clase de datos , solamente es seleccionar el tipo de dato en cada columna usualmente el mayor cambio sera pasar de un INT a Bigint o de un tinyint a INT dependiendo del caso



4. Usar la opcion para carga de datos generica:



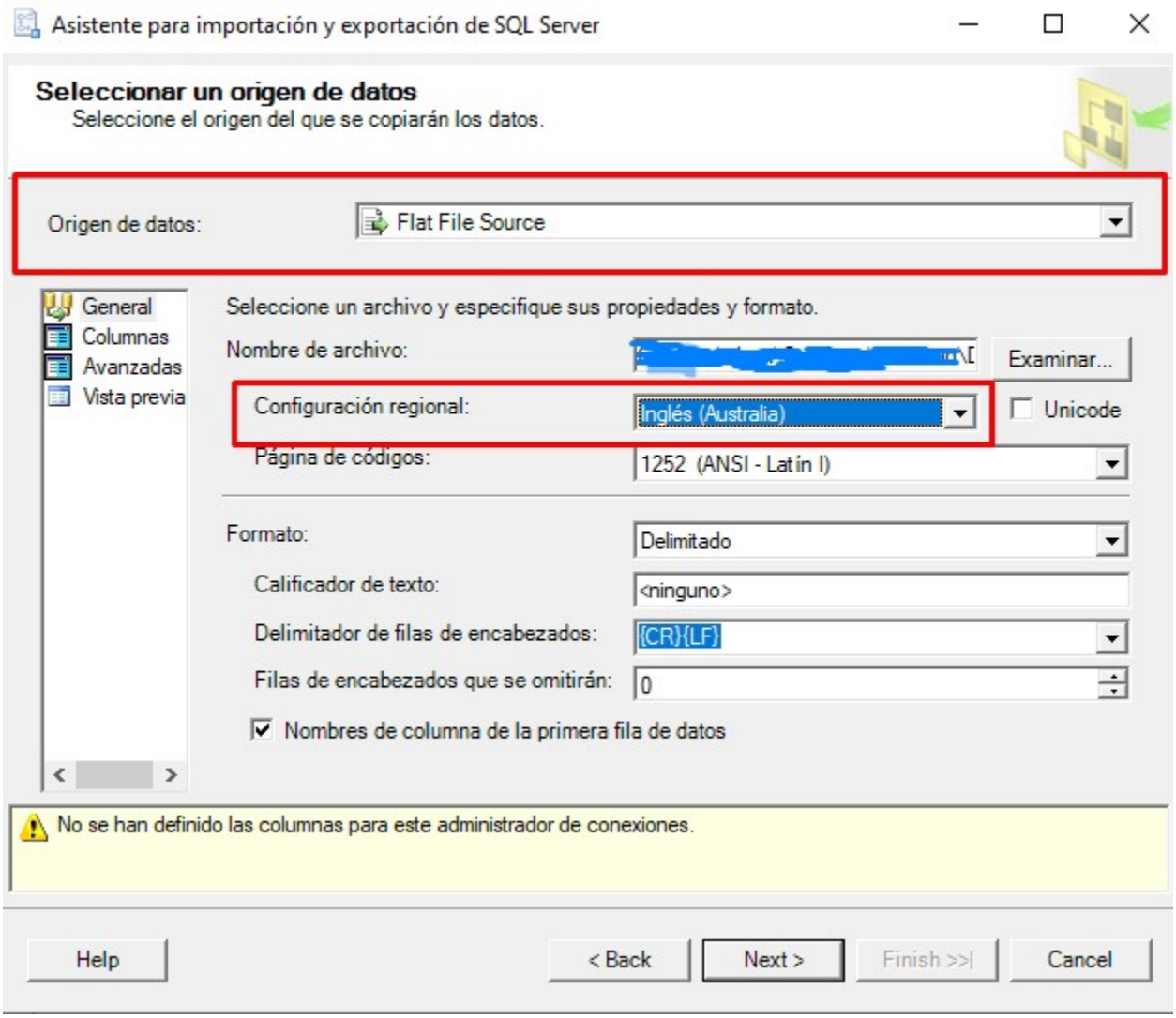
Esta opcion se usara para los siguientes archivos:

- HourlyIntensities.csv

- DailyActivity.csv
- MinuteCalories.csv
- WeightInfo.csv

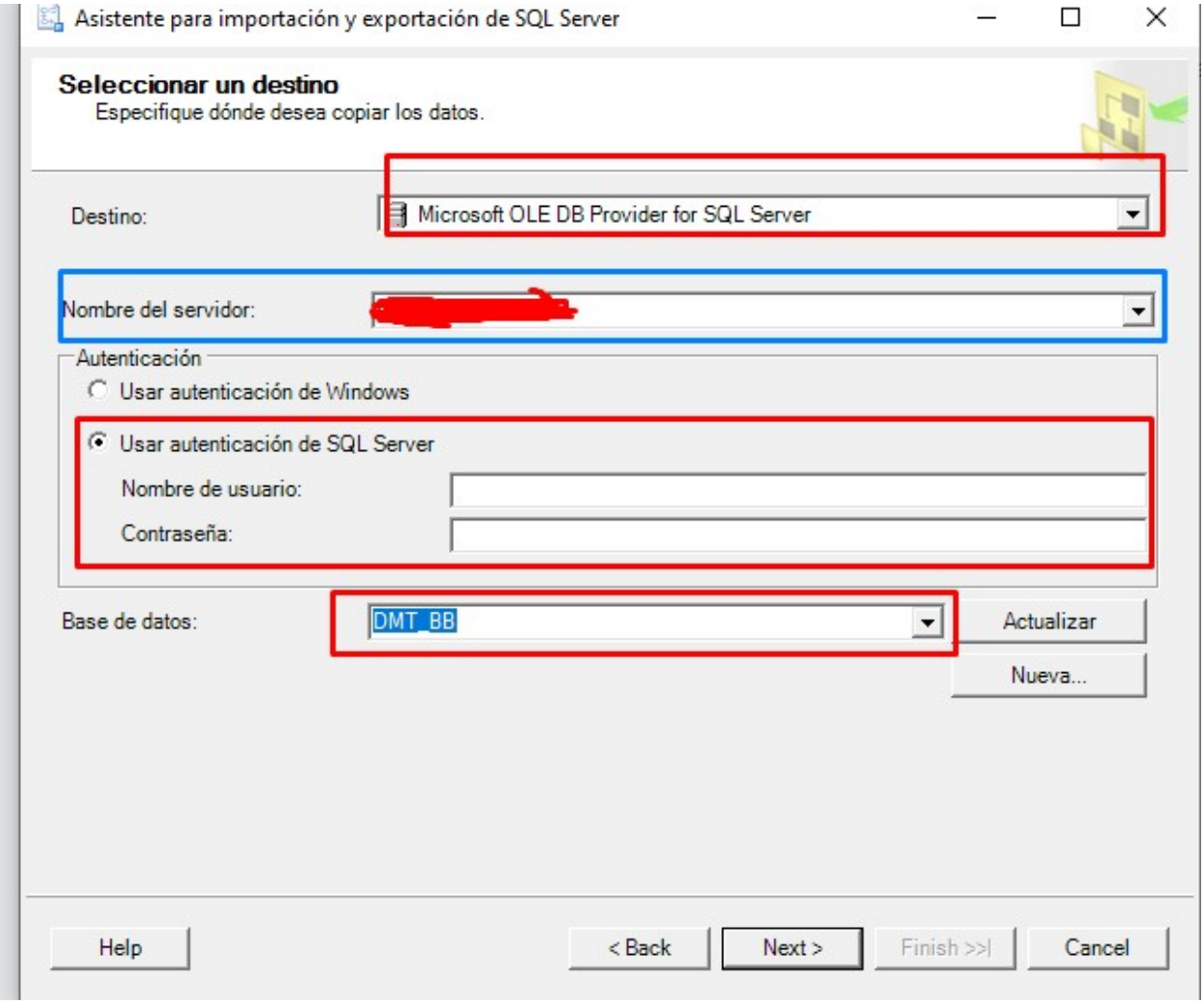
5. Configuracion de origen , aqui deben configurarse:

- Origen de los datos , que en este caso es “Archivo plano”
- Region , para reconocer datos con punto decimal (.) elegir Ingles(Australia)



6. Configurar conexion a servidor SQL Server:

Seleccionar la opcion Microsoft OLE DB Provider for SQL Server , marcar “Usar autenticacion SQL Server” e ingresar credenciales y la base de datos creada



Luego de ello se debera finalizar las opciones del wizard , es necesario verificar los tipos de datos como en el paso (3)

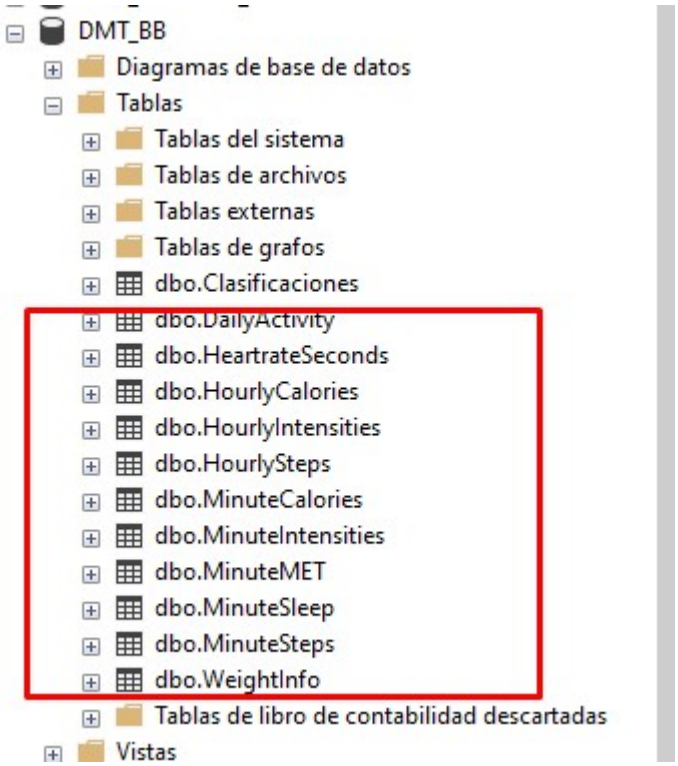
7. Modificar las columnas de fecha con un script SQL , esto netamente debe de ejecutarse una vez ya que es para la conversion a fechas solo de 4 tablas:

```
UPDATE HourlyIntensities SET Record = CONVERT(datetime2,REPLACE(Record,'"',''))
UPDATE MinuteCalories SET Record = CONVERT(datetime2,REPLACE(Record,'"',''))
UPDATE WeightInfo SET Record = CONVERT(datetime2,REPLACE(Record,'"',''))
UPDATE DailyActivity SET ActivityDate = CONVERT(date,REPLACE(ActivityDate,'"',''))

ALTER TABLE HourlyIntensities ALTER COLUMN Record datetime2;
ALTER TABLE MinuteCalories ALTER COLUMN Record datetime2;
ALTER TABLE WeightInfo ALTER COLUMN Record datetime2;
ALTER TABLE DailyActivity ALTER COLUMN ActivityDate date;
```

Con esos comandos , el tipo de dato sera cambiado a tipo date/datetime2 en la tabla original

8. Verificar que la base de datos cuente con todas las tablas cargadas



Con esta ultima operacion , la fase Process ha sido concluida con exito y se puede proceder a un analisis mediante SQL Server.

Fase Analyze

Para el analisis se hizo uso de SQL Server como Sistema Gestor de Base de Datos , es importante considerar que:

- Se debe de configurar una coneccion a la base de datos en caso se desee proceder en un mismo script de R
- Solo se realizo un analisis en las tablas HeartRate Seconds y HourlyCalories por limitaciones temporales

```
# De seguir con R , se debe configurar la conexion a la propia base de datos:
# Instalacion de librerias para conexion SQL
# install.packages('DBI')
# install.packages('odbc')
#
# #Carga de las librerias
# library(DBI)
# library(odbc)
#
# # Establecer la conexión a la base de datos SQL Server
# sql_connection <- dbConnect(
#   odbc::odbc(),
#   Driver = "SQL Server",
#   Server = "nombre_del_servidor",
#   Database = "nombre_de_la_base_de_datos",
#   UID = "usuario",
```

```
#   PWD = "contraseña"
# )
```

El analisis empieza con la revision del nombre de columnas en toda la base de datos:

```
SELECT
  COLUMN_NAME,
  COUNT(TABLE_NAME) as #
FROM INFORMATION_SCHEMA.COLUMNS
GROUP BY COLUMN_NAME;
```

Table 2: Displaying records 1 - 10

COLUMN_NAME	#
ActivityDate	1
Avg_Intensity	1
BMI	1
Calories	3
definition	1
diagram_id	1
FairlyActiveMinutes	1
Heartrate	1
Id	11
Intensity	1

La columna que mas se presenta en la base de datos es Id , lo que significa que la carga ha sido igual para todos los conjuntos de datos , otro dato a resaltar es el Record , presente en todas las tablas como el registro de tiempo a excepcion de la tabla ActivityDate

```
SELECT
 TABLE_NAME,
 SUM(CASE
     WHEN COLUMN_NAME = 'Id' THEN 1
     ELSE
     0
     END ) AS has_id_column
FROM INFORMATION_SCHEMA.COLUMNS
GROUP BY TABLE_NAME
ORDER BY TABLE_NAME ASC;
```

Table 3: Displaying records 1 - 10

TABLE_NAME	has_id_column
DailyActivity	1
HeartrateSeconds	1
HourlyCalories	1
HourlyIntensities	1
HourlySteps	1
MinuteCalories	1
MinuteIntensities	1

TABLE_NAME	has_id_column
MinuteMET	1
MinuteSleep	1
MinuteSteps	1

Verificar si todas las tablas contienen algun dato de fecha o tiempo

```
SELECT
 TABLE_NAME,
 SUM(CASE
     WHEN data_type IN ('TIMESTAMP','DATETIME','DATETIME2','TIME','DATE') THEN 1
     ELSE
     0
     END) AS has_time_info
FROM INFORMATION_SCHEMA.COLUMNS
GROUP BY TABLE_NAME
HAVING SUM(CASE
     WHEN data_type IN ('TIMESTAMP','DATETIME','DATETIME2','TIME','DATE') THEN 1
     ELSE
     0
     END) = 0;
```

Table 4: 1 records

TABLE_NAME	has_time_info
sysdiagrams	0

En este caso , sysdiagrams es solo una tabla propia del sistema por lo que todas nuestras tablas cuentan con datos temporales adecuadamente. Para mayor detalle se obtiene:

```
SELECT
 CONCAT(TABLE_CATALOG,'.', TABLE_SCHEMA ,'.', TABLE_NAME) AS table_path,
 TABLE_NAME,
 COLUMN_NAME
FROM INFORMATION_SCHEMA.COLUMNS
WHERE
    DATA_TYPE IN ('TIMESTAMP','DATETIME','DATETIME2','DATE');
```

Table 5: Displaying records 1 - 10

table_path	TABLE_NAME	COLUMN_NAME
DMT_BB.dbo.HeartrateSeconds	HeartrateSeconds	Record
DMT_BB.dbo.HourlyCalories	HourlyCalories	Record
DMT_BB.dbo.HourlySteps	HourlySteps	Record
DMT_BB.dbo.MinuteMET	MinuteMET	Record
DMT_BB.dbo.MinuteSteps	MinuteSteps	Record
DMT_BB.dbo.MinuteIntensities	MinuteIntensities	Record
DMT_BB.dbo.MinuteSleep	MinuteSleep	Record
DMT_BB.dbo.HourlyIntensities	HourlyIntensities	Record

table_path	TABLE_NAME	COLUMN_NAME
DMT_BB.dbo.MinuteCalories	MinuteCalories	Record
DMT_BB.dbo.WeightInfo	WeightInfo	Record

Habiendo comprobado la integridad de los datos , procedemos con el analisis limitado a las tablas:

- HeartrateSeconds: Registra el ritmo cardiaco de usuario por segundo
- HourlyCalories: Registra la quema de calorias de usuario por hora

Analisis de HeartrateSeconds

Inicia con la vista previa de la tabla

```
SELECT
 *
FROM
 HeartrateSeconds
```

Table 6: Displaying records 1 - 10

Id	Record	Heartrate
2022484408	2016-04-01 07:54:00.0000000	93
2022484408	2016-04-01 07:54:05.0000000	91
2022484408	2016-04-01 07:54:10.0000000	96
2022484408	2016-04-01 07:54:15.0000000	98
2022484408	2016-04-01 07:54:20.0000000	100
2022484408	2016-04-01 07:54:25.0000000	101
2022484408	2016-04-01 07:54:30.0000000	104
2022484408	2016-04-01 07:54:35.0000000	105
2022484408	2016-04-01 07:54:45.0000000	102
2022484408	2016-04-01 07:54:55.0000000	106

Descubrir el promedio de ritmo cardiaco por usuario en todo el periodo de recoleccion:

```
SELECT
    Id,
    AVG(Heartrate) as Avg_Heartrate
FROM HeartrateSeconds
GROUP BY Id
```

Table 7: Displaying records 1 - 10

Id	Avg_Heartrate
5577150313	68
8792009665	73
5553957443	68
2022484408	80
4558609924	81

Id	Avg_Heartrate
4020332650	82
2026352035	89
6391747486	84
7007744171	90
8877689391	84

Notamos que gran parte de los usuarios ha mantenido un ritmo cardiaco habitual para adultos en todo el tiempo de recoleccion de datos , siendo el minimo un ritmo de 66 y uno maximo de 94.

Hay que resaltar que segun estandares medicos , los adultos deben contemplar ritmos cardiacos entre 60 a 100 , de no contemplarse en estos limites puede representar complicaciones cardiacas.

```
SELECT
    Id,
    AVG(Heartrate) as AVG_Heartrate,
    CASE
        WHEN AVG(Heartrate) < 60 THEN 'Low Heartrate'
        WHEN AVG(Heartrate) BETWEEN 60 AND 100 THEN 'Normal Heartrate'
        WHEN AVG(Heartrate) > 100 THEN 'High Heartrate'
        ELSE 'ERROR'
    END AS Class_by_Heartrate
FROM HeartrateSeconds
GROUP BY Id
ORDER BY AVG(Heartrate)
```

Table 8: Displaying records 1 - 10

Id	AVG_Heartrate	Class_by_Heartrate
4388161847	66	Normal Heartrate
5577150313	68	Normal Heartrate
5553957443	68	Normal Heartrate
8792009665	73	Normal Heartrate
2347167796	76	Normal Heartrate
6962181067	78	Normal Heartrate
2022484408	80	Normal Heartrate
4558609924	81	Normal Heartrate
4020332650	82	Normal Heartrate
6117666160	83	Normal Heartrate

En esta consulta , parece que durante todo el tiempo de recoleccion la mayoria de nuestros usuarios presenta un ritmo cardiaco normal , lo esperado de adultos con una rutina de ejercicio comun.

Sin embargo , no seria apropiado concluir con un periodo de tiempo tan amplio ; por lo que , seria mejor ver la tendencia en dias

```
SELECT
    Id,
    CAST(Record AS DATE) AS heartrate_day,
    AVG(Heartrate) AS avg_heartrate_per_day
FROM
```

```
    HeartrateSeconds
GROUP BY
    Id,
    CAST(Record AS DATE)
ORDER BY Id , CAST(Record AS DATE)
```

Table 9: Displaying records 1 - 10

Id	heartrate_day	avg_heartrate_per_day
2022484408	2016-04-01	88
2022484408	2016-04-02	72
2022484408	2016-04-03	74
2022484408	2016-04-04	78
2022484408	2016-04-05	83
2022484408	2016-04-06	82
2022484408	2016-04-07	90
2022484408	2016-04-08	81
2022484408	2016-04-09	84
2022484408	2016-04-10	80

Notamos que si un usuario ya presenta un rango de valores en su ritmo cardiaco , este tiende a mantenerse en el resto de dias. Ello es visible con usuarios como el **2022484408** que mantiene su rango de valores entre 70 - 90 o el usuario **5553957443** que presenta menores valores entre 60 - 75 . Esto no quiere decir que el usuario **2022484408** realiza mayor ejercicio que **5553957443** pero nos da es un indicador inicial del estilo de vida entre cada usuarios y de posibles condiciones fisicas.

Un analisis a detalle entre periodos de tiempo , podria otorgar una mejor aclaracion de los resultados previos:

```
DECLARE
    @MORNING_START NVARCHAR(12),
    @MORNING_END NVARCHAR(12),
    @AFTERNOON_END NVARCHAR(12),
    @EVENING_END NVARCHAR(12);

SET @MORNING_START = '06:00:00:000000';
SET @MORNING_END = '12:00:00:000000';
SET @AFTERNOON_END = '18:00:00:000000';
SET @EVENING_END = '21:00:00:000000';

WITH
dow_heartrate_summary AS (
    SELECT
        Id,
        DATEPART(WEEKDAY,Record) as dow_number,
        DATENAME(WEEKDAY,Record) as day_of_week,
        CASE
            WHEN DATENAME(WEEKDAY, Record) IN ('Domingo', 'Sábado') THEN 'Weekend'
            WHEN DATENAME(WEEKDAY, Record) NOT IN ('Domingo', 'Sábado') THEN 'Weekday'
            ELSE 'ERROR'
        END AS part_of_week,
        CASE
            WHEN CAST(Record AS TIME) BETWEEN CAST(STUFF(@MORNING_START,9,1,'.') AS TIME) AND CAST(STUF
```

```
            WHEN CAST(Record AS TIME) BETWEEN CAST(STUFF(@MORNING_END,9,1,'.') AS TIME) AND CAST(STUFF(@
            WHEN CAST(Record AS TIME) BETWEEN CAST(STUFF(@AFTERNOON_END,9,1,'.') AS TIME) AND CAST(STUFF
            WHEN CAST(Record AS TIME) >= CAST(STUFF(@EVENING_END,9,1,'.') AS TIME)
                 OR CAST(CAST(Record AS TIME) AS DATETIME) <= CAST(CAST(STUFF(@MORNING_START,9,1,'.') AS
            ELSE 'ERROR'
        END AS time_of_day ,
        AVG(Heartrate) as AVG_Heartrate_per_period
    FROM HeartrateSeconds
    GROUP BY
    Id,
    DATEPART(WEEKDAY,Record),
        DATENAME(WEEKDAY,Record),
        CASE
            WHEN DATENAME(WEEKDAY, Record) IN ('Domingo', 'Sábado') THEN 'Weekend'
            WHEN DATENAME(WEEKDAY, Record) NOT IN ('Domingo', 'Sábado') THEN 'Weekday'
            ELSE 'ERROR'
        END,
        CASE
            WHEN CAST(Record AS TIME) BETWEEN CAST(STUFF(@MORNING_START,9,1,'.') AS TIME) AND CAST(STUFF
            WHEN CAST(Record AS TIME) BETWEEN CAST(STUFF(@MORNING_END,9,1,'.') AS TIME) AND CAST(STUFF(@
            WHEN CAST(Record AS TIME) BETWEEN CAST(STUFF(@AFTERNOON_END,9,1,'.') AS TIME) AND CAST(STUFF
            WHEN CAST(Record AS TIME) >= CAST(STUFF(@EVENING_END,9,1,'.') AS TIME)
                 OR CAST(CAST(Record AS TIME) AS DATETIME) <= CAST(CAST(STUFF(@MORNING_START,9,1,'.') AS
            ELSE 'ERROR'
        END
)
/*Insert into temp Table*/
SELECT *
INTO #dow_summary_for_heartrate
FROM dow_heartrate_summary

/*Summary preview*/
SELECT * FROM #dow_summary_for_heartrate

/*Point 1*/
SELECT
    part_of_week,
    day_of_week,
    time_of_day,
    AVG(AVG_Heartrate_per_period) AS avg_in_period
FROM #dow_summary_for_heartrate
GROUP BY
    part_of_week,
    day_of_week,
    time_of_day
ORDER BY day_of_week

/*Point 2*/
SELECT
    day_of_week,
    AVG(AVG_Heartrate_per_period) AS avg_in_day
FROM
    #dow_summary_for_heartrate
```

```
GROUP BY
    day_of_week

/*Point 3*/
SELECT
    time_of_day,
    AVG(AVG_Heartrate_per_period) AS avg_per_time
FROM
    #dow_summary_for_heartrate
GROUP BY
    time_of_day

/*Point 4*/
SELECT
    part_of_week,
    AVG(AVG_Heartrate_per_period) AS avg_per_week_part
FROM
    #dow_summary_for_heartrate
GROUP BY
    part_of_week
```

Esta consulta permite desarrollar un esquema resumido que muestra los ritmos cardiacos entre periodos distintos del dia , todo ello sera almacenado en una tabla temporal para mas comodidad.

Consideramos algunos puntos:

1. En un principio , se nota como el promedio en todos los periodos tiende a valores entre 70-85 , valores normales considerando la rutina de un adulto activo. Pero notamos algo , el ritmo cardiaco suele ser mayor en las tardes y noches de cada dia , ademas los dias que presentan un mayor valor son justamente los que forman parte del fin de semana.
2. Si promediamos por dia , todos tienen un valor comun de 78 a excepcion del domingo y martes con promedios de 79. Para validar nuestra hipotesis anterior , seria mejor agrupar por periodos del dia:
3. Tal como se supuso , las tardes y noches presentan un mayor ritmo cardiaco promedio , esto puede representar que estos periodos son los de mayor actividad ritmica entre usuarios , verifiquemos en el caso de los periodos de la semana:
4. No hay mucha diferencia entre dias y fines de semana como se supuso , al menos en promedio.

Podemos denotar algunos puntos del analisis de Heartrate Seconds:

- El ritmo cardiaco de los usuarios no parece presentar anomalias cardiacas tanto bajas como altas , lo que supone que los usuarios llevan un estilo de vida saludable en la adultez.
- La tendencia de ritmo cardiaco entre usuarios se mantiene a lo largo de los dias , sin presentar muchas anomalias.
- El ritmo cardiaco no representa necesariamente una mayor realizacion de ejercicio pero sirve como un indicador potencial del estado de salud de los usuarios que puede ser relacionado.
- Los periodos de mayor aumento son en las tardes y noches con un pico algo mayor en fines de semana , si bien no representa obligatoriamente mayor ejercicio fisico en estos periodos , es un indicador potencial de mayor actividad.

Analisis de HourlyCalories

Este conjunto de datos representa el registro de calorias quemadas por hora , empezamos viendo cuantos usuarios fueron registrados:

```
SELECT
DISTINCT
    Id
FROM
    HourlyCalories
```

Table 10: Displaying records 1 - 10

Id
1644430081
7086361926
4702921684
3977333714
6391747486
6290855005
3372868164
8877689391
2026352035
5577150313

Hay por lo menos 35 usuarios que fueron registrados, sobre la quema de calorias se tiene:

- Segun estandares de salud , una quema de calorias para adultos activos es entre 2000 y 3000
- Una quema menor a 2000 podria representar un deficit de ejercicio o estilo sedentario
- Una quema mayor a 3000 podria representar un sobreesfuerzo que puede llevar a complicaciones de salud

Por ello , es conveniente agrupar la quema de calorias por dia en tres categorias

```
SELECT
    Id,
    CAST(RECORD AS Date) as fecha_cal,
    SUM(Calories) as calorias_por_dia,
    CASE
        WHEN SUM(Calories) < 2000 THEN 'Quema baja de calorias'
        WHEN SUM(Calories) BETWEEN 2000 AND 3000 THEN 'Quema moderada de calorias'
        WHEN SUM(Calories) > 3000 THEN 'Quema alta de calorias'
    END AS clasificacion
FROM
    HourlyCalories
GROUP  BY Id, CAST(Record as DATE)
ORDER BY Id , CAST(Record as DATE)
```

Table 11: Displaying records 1 - 10

Id	fecha_cal	calorias_por_dia	clasificacion
1503960366	2016-03-12	2228	Quema moderada de calorias
1503960366	2016-03-13	2100	Quema moderada de calorias
1503960366	2016-03-14	1830	Quema baja de calorias
1503960366	2016-03-15	2111	Quema moderada de calorias
1503960366	2016-03-16	1967	Quema baja de calorias
1503960366	2016-03-17	2039	Quema moderada de calorias
1503960366	2016-03-18	2002	Quema moderada de calorias
1503960366	2016-03-19	2057	Quema moderada de calorias
1503960366	2016-03-20	2096	Quema moderada de calorias
1503960366	2016-03-21	1846	Quema baja de calorias

A simple vista , se puede deducir que no se sigue una tendencia por usuario ya que se ve como el mismo usuario contempla dias donde hay quemas moderadas o bajas , lo que tiene sentido pues no se realiza ejercicio fisico al mismo nivel o de la misma forma en cada dia. Resaltar que el valor que parece repetirse menos es de 'Quema alta'.

Lo mas apropiado , seria ver cuantas veces cada usuario a ello una quema baja , moderada o alta y ello se puede lograr por medio de una subconsulta y colocar los resultados en una tabla temporal

```
SELECT
    Id,
    clasificacion,
    COUNT(*) as number_class
INTO #user_cal_classification
FROM (
    SELECT
        Id,
        CAST(RECORD AS Date) as fecha_cal,
        SUM(Calories) as calorias_por_dia,
        CASE
            WHEN SUM(Calories) < 2000 THEN 'Quema baja de calorias'
            WHEN SUM(Calories) BETWEEN 2000 AND 3000 THEN 'Quema moderada de calorias'
            WHEN SUM(Calories) > 3000 THEN 'Quema alta de calorias'
        END AS clasificacion
    FROM
        HourlyCalories
    GROUP  BY Id, CAST(Record as DATE)
) daily_cal_table
GROUP BY Id, clasificacion

/*Preview of summary*/
SELECT * FROM #user_cal_classification ORDER BY Id

/*Point 1 and 2*/
SELECT
    Id,
    clasificacion,
    number_class
FROM (
    SELECT
```

```
        Id,
        clasificacion,
        number_class,
        ROW_NUMBER() OVER (PARTITION BY Id ORDER BY number_class DESC) AS max_class
    FROM
        #user_cal_classification
) freq_table
WHERE max_class = 1
ORDER BY Id

/*Point 3*/
SELECT
    clasificacion,
    COUNT(*) as class_freq
FROM (
    SELECT
        Id,
        clasificacion,
        number_class,
        ROW_NUMBER() OVER (PARTITION BY Id ORDER BY number_class DESC) AS max_class
    FROM
        #user_cal_classification
) freq_table
WHERE max_class = 1
GROUP BY clasificacion
```

Notamos como gran parte de los usuarios queman calorias de forma baja o moderada , aunque si hay usuarios presentes con una quema alta de calorias. Ademas si sumamos el numero de registros por clase el maximo sera 62 , concordando en que el periodo de recoleccion fue de 62 dias.

Algo notable , es que no todos los usuarios presentan necesariamente registros que suman 62 , como el caso del usuario **2891001357** que solo registro una 'quema baja' en todo el periodo. Esto nos obliga a recuperar el maximo de otra forma , extrayendo la frecuencia por usuario.

1. Si deseamos saber que clase se da con mas frecuencia , es decir que clase de quema es mas frecuente entre nuestros usuarios debemos recuperar aquella con frecuencia maxima por usuario:
2. Confirmamos algunos puntos al ver:

- La quema alta solo parece ser mas frecuente en 5 usuarios
- La quema baja y moderada parecen estar a la par , representando una rutina saludable en mayoria de los usuarios.

3. Con esta ultima consulta , podemos evaluar algunos puntos del analisis:

- No hay una tendencia clara entre las quemas de calorias por usuario , ya que una rutina de ejercicio puede no tener el mismo gasto entre dias.
- La mayoria de los usuarios realizan rutinas con una quema de calorias menor a 2000 , ello no representa necesariamente sedentarismo pero seria apropiado evaluar un aumento de intensidad
- No obstante, al menos el 40% (14 usuarios) presentan una quema de calorias moderada lo que da a entender que sus rutinas cumplen con estandares adecuados para la vida adulta.

- Muy pocos usuarios presentan quema alta de calorias , lo que podria reflejar un sobreesfuerzo de ejercicio , la advertencia de este mismo puede resultar en posibles caracteristicas a implementar.

Esta fase nos ha dado a entender datos interesantes para conocer el comportamiento del publico objetivo , en la siguiente fase Share es posible confirmar estas hipotesis.

Fase Share

Para esta fase se eligio usar Tableu como herramienta de visualizacion , el producto final se puede visualizar en este enlace:

Bellabeat Case

Visualizaciones de Heartrate

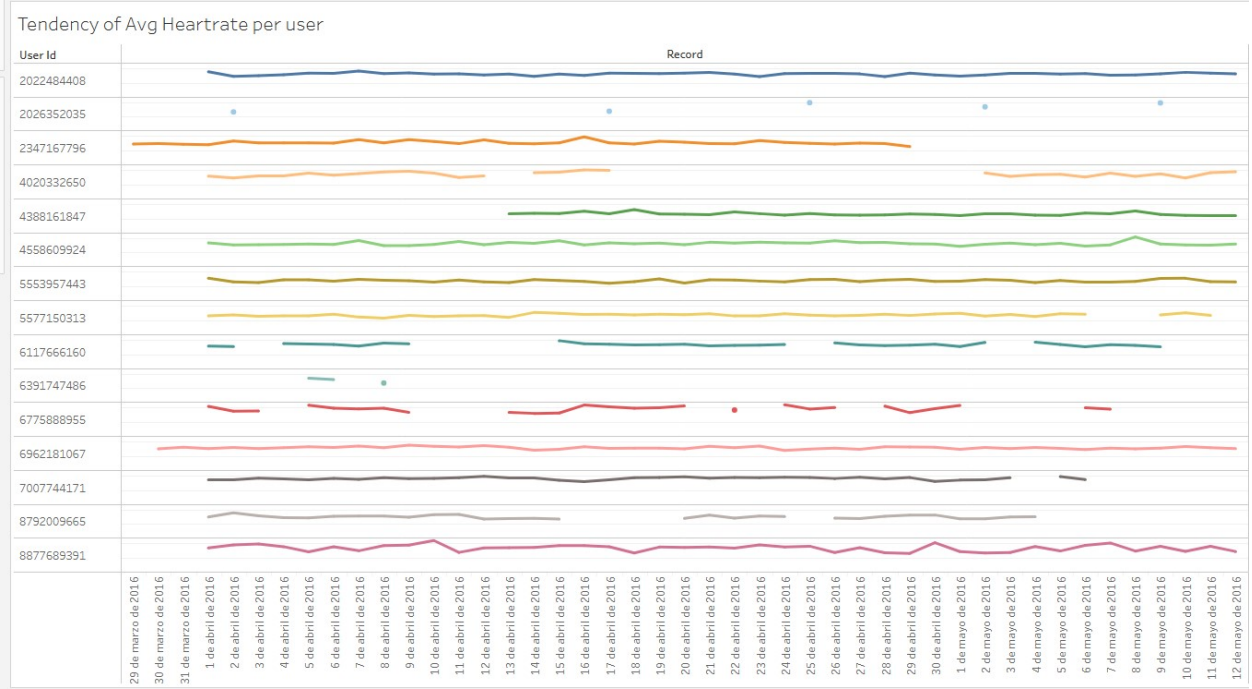
Uno de los primeros graficos realizados es una tabla de apoyo , que practicamente es un elemento para corroborar datos en futuras visualizaciones , un ejemplo seria la tabla de clasificaciones de ritmo cardiaco en todo el periodo:

AVG per user

User Id	Heartrate class	
6775888955	Normal Heartrate	94,98
7007744171	Normal Heartrate	90,73
2026352035	Normal Heartrate	89,58
8877689391	Normal Heartrate	84,69
6391747486	Normal Heartrate	84,12
6117666160	Normal Heartrate	83,69
4020332650	Normal Heartrate	82,10
4558609924	Normal Heartrate	81,11
2022484408	Normal Heartrate	80,63
6962181067	Normal Heartrate	78,68
2347167796	Normal Heartrate	76,43
8792009665	Normal Heartrate	73,88
5553957443	Normal Heartrate	68,85
5577150313	Normal Heartrate	68,38
4388161847	Normal Heartrate	66,13

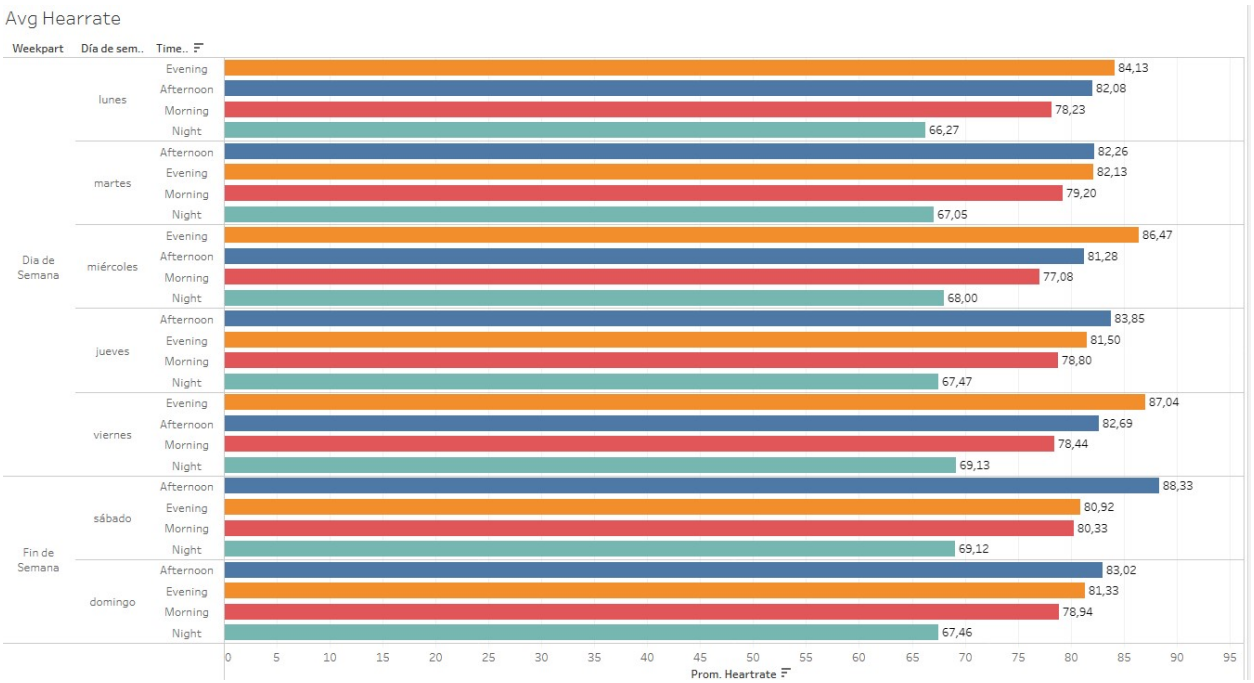
Esta tabla corrobora una deduccion previa , los usuarios presentan ritmos cardiacos normales en todo el periodo de recoleccion.

Lo ideal para corroborar la hipotesis anterior: *"La tendencia de ritmo cardiaco entre usuarios se mantiene a lo largo de los dias , sin presentar muchas anomalias"* es realizar un grafico de linea a lo largo del tiempo:



De inmediato , confirmamos que realmente en cuanto a ritmo cardiaco no hay picos ni caidas resaltantes por cada usuario ya que se ha mantenido un promedio a lo largo de todo el periodo de datos. Aunque ahora es mas notable que no todos los usuarios registraron adecuadamente este dato como el usuario **2026352035** y otros donde se nota una interrupcion ; sin embargo , es suficiente para corroborar nuestra suposición.

Tambien incluimos un grafico de barras por periodo para corrobar que las tardes y noches son periodos con mayor ritmo cardiaco:



Este grafico fue ordenado de forma descendente , confirmando que los periodos de tarde y noche son donde mayor promedio de ritmo cardiaco es presente. Tambien afirmamos que los periodos de madrugada son los mas bajos , algo de esperar ya que al ser personas adultas no se espera mucha actividad en esas horas.

Mediante estas visualizaciones hemos confirmado las hipotesis anteriores del analisis HeartrateSeconds ademas de encontrar algunas observaciones en nuestro datos no contempladas como la falta de registros en algunos usuarios.

Visualizaciones de HourlyCalories

Para este conjunto de datos es importante el seguimiento de quema de calorias a lo largo del tiempo , por lo que un mapa de calor para valores minimo y maximo es lo ideal:

Calorie Class

Record (HourlyCalories.csv)

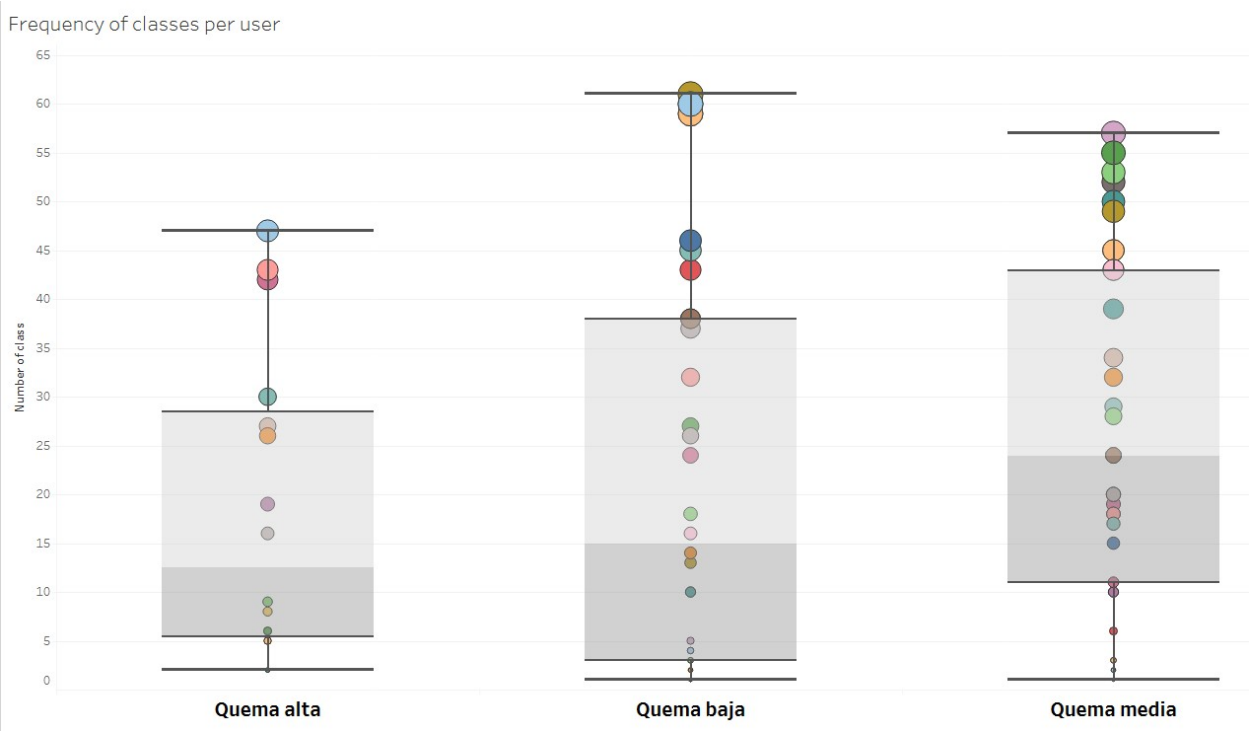
Id (HourlyC..	12 de ma..	13 de ma..	14 de ma..	15 de ma..	16 de ma..	17 de ma..	18 de ma..	19 de ma..	20 de ma..	21 de ma..	22 de ma..	23 de ma..	24 de ma..	25 de ma..	26 de ma..	27 de ma..	28 de ma..	29 de ma..	30 de ma..	31 de ma..
1844505072	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.344	1.576	1.962
1927972279	2.758	2.608	2.135	2.560	2.893	3.395	3.677	2.310	2.518	2.206	2.666	2.551	2.174	2.625	2.433	3.587	3.910	2.627	3.406	2.547
2022484408	2.334	2.252	2.631	2.653	2.641	2.313	2.459	2.716	2.447	2.885	2.398	2.657	3.139	2.782	2.613	2.338	2.219	3.111	2.869	3.066
2026352035	1.678	1.694	1.808	1.801	1.802	1.623	1.319	1.427	1.418	1.463	1.392	1.395	1.336	1.433	1.360	1.425	1.425	1.374	1.396	1.669
2320127002	1.553	1.320	1.320	1.320	1.320	1.322	1.320	1.320	1.320	1.320	1.814	1.792	1.618	1.351	1.320	1.320	1.321	1.320	1.320	1.320
2347167796	2.515	2.528	2.141	2.114	2.087	2.103	2.003	2.058	2.317	2.148	2.159	2.060	2.116	1.914	2.339	2.189	1.872	2.044	2.186	1.928
2873212765	2.019	2.296	2.002	1.866	1.978	2.227	2.433	1.926	1.811	1.981	1.849	2.241	1.989	1.533	1.772	1.254	1.807	1.987	1.819	2.022
2891001357																				
3372868164	1.900	2.002	1.993	1.790	1.998	1.793	1.887	1.812	2.034	2.015	1.779	1.861	1.940	1.900	1.840	2.074	1.841	2.005	1.872	2.076
3977333714	1.360	1.350	1.212	1.675	1.762	1.597	1.540	1.155	1.008	1.008	1.008	1.008	1.008	1.008	1.008	1.008	1.008	1.008	1.008	1.076
4020332650	2.991	2.484	2.565	3.015	3.825	2.157	1.992	1.992	1.992	1.992	1.992	1.992	1.992	1.992	1.992	1.992	2.968	2.973	3.416	3.441
4057192912	1.776	1.776	2.454	1.776	1.934	1.776	1.869	1.776	1.781	1.978	2.289	2.422	1.778	1.776	1.776	2.006	1.778	2.248	2.237	1.776
4319703577	2.202	2.118	2.234	2.130	2.227	2.180	2.039	2.048	1.872	2.088	2.137	2.202	2.347	2.203	2.048	1.786	2.297	1.957	2.141	2.130
4388161847																				
4445114986	2.218	2.398	2.306	2.271	2.291	2.394	2.166	2.301	2.217	2.456	2.128	2.299	2.049	2.285	2.399	2.302	2.070	2.010	2.032	2.228
4558609924	1.997	1.656	1.460	1.707	1.665	1.894	1.682	1.344	1.677	1.344	1.344	1.344	1.836	1.641	1.713	2.148	2.134	2.091	1.884	1.963
4702921684	3.607	3.102	2.946	3.062	2.796	2.920	2.732	2.018	2.016	2.434	3.131	2.870	2.948	2.821	3.656	3.378	2.975	2.885	2.915	2.896
5553957443	1.490	1.709	2.132	2.039	2.062	1.763	1.569	1.320	1.320	1.615	1.419	1.605	1.860	1.684	1.787	1.554	1.853	2.204	1.925	2.098
5577150313	4.776	4.452	3.757	3.808	3.752	3.575	3.882	4.330	2.739	2.125	2.256	3.320	3.120	3.163	3.789	2.938	3.825	3.543	3.326	3.824
6117666160	1.847	2.242	2.169	2.112	2.324	1.992	2.260	2.334	2.404	2.287	2.082	2.148	2.443	2.454	2.550	2.050	1.987	2.292	2.411	2.500
6290855005	3.647	2.724	2.064	2.064	2.952	2.697	2.778	2.701	2.947	2.555	3.333	2.382	2.599	2.064	2.064	3.445	2.950	2.599	2.775	2.064
6391747486	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824	1.824
6775888955	2.229	2.775	2.454	2.414	2.071	2.081	3.009	2.512	2.898	2.452	1.848	1.848	1.848	2.426	4.564	4.066	3.921	2.063	2.146	2.273
6962181067	2.275	2.049	2.259	2.010	2.523	2.803	2.876	2.354	2.445	2.448	1.950	1.965	1.971	2.033	2.005	2.027	2.206	2.186	2.244	2.190
7007744171	1.560	2.829	2.665	2.635	1.560	3.746	2.594	2.201	3.041	3.404	2.890	2.702	2.800	2.891	2.398	2.462	2.282	2.228	2.755	2.501
7086361926	2.157	1.861	2.522	2.979	2.439	2.275	2.388	2.685	2.015	2.383	1.903	1.897	2.990	2.259	3.052	2.357	2.149	2.296	2.273	2.396
8053475328	3.259	4.528	2.755	3.042	2.911	3.253	2.842	3.319	3.381	2.759	2.911	2.906	2.744	3.028	2.666	2.720	2.651	2.990	2.089	2.783
8253242879	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440	1.440
8378563200	2.701	2.646	3.378	3.460	3.124	2.831	3.405	2.894	3.105	2.847	3.827	3.879	3.556	4.013	2.871	2.594	4.140	4.170	3.070	4.118
8583815059	2.439	2.390	2.336	2.408	2.389	2.016	2.016	2.016	2.016	2.016	2.016	2.016	2.016	2.016	2.016	2.016	2.407	2.781	2.798	2.875
8792009665	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	1.680	2.648	2.126	1.977	1.888	2.148
8877689391	3.889	2.800	3.677	2.814	3.486	3.675	3.723	3.782	3.401	2.874	3.914	3.817	2.942	2.788	2.821	4.731	2.832	3.520	4.202	3.000

De esta muestra del grafico original , se tiene la quema de calorias diaria por usuario a lo largo del tiempo , este cuenta con tres colores siendo:

- Naranja si la quema de calorias fue baja (<2000)
- Amarilla si la quema de calorias fue moderada (Entre 2000 y 3000)
- Roja si la quema de calorias fue alta (>3000)

En la muestra , se nota resalta como la gran mayoria presente una quema de calorias entre moderada y baja mientras que el color rojo no es dominante en la tabla. Ademas , notamos que hay registros en blanco , lo que quiere decir que no hubo registro en esa fecha.

Para confirmar la hipotesis de que predominan la quema moderada y baja sobre la alta se desarrollo un grafico de cajas:

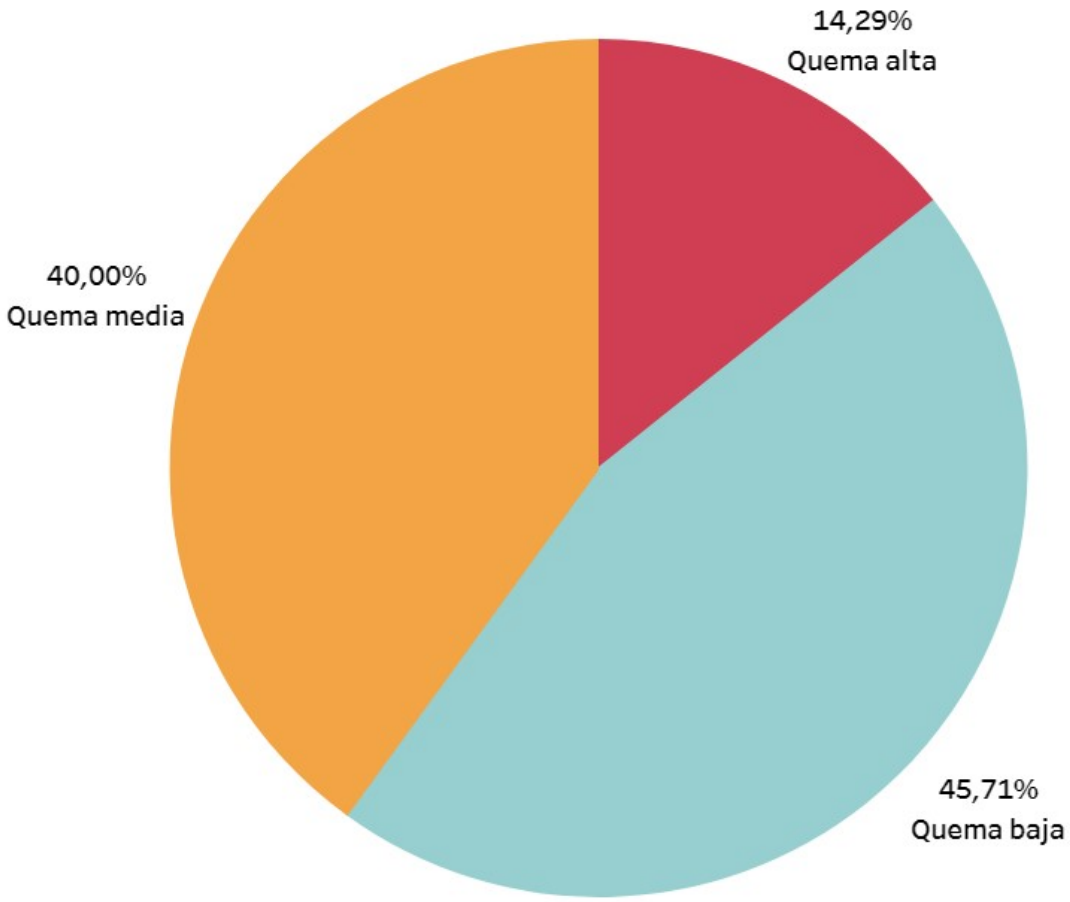


Este grafico considera el numero de veces a las que un usuario pertenecio a una clase siendo:

- El color de la bola , representa el id del usuario
- El tamaño representa cuantas veces estuvo en esa clase
- El eje Y representa el numero de veces , siendo el centro de bola alineado al valor.
- La caja representa los valores donde hay mayor distribucion

Notamos como la Quema baja contiene gran cantidad de valores de considerable tamaño pero fuera de la distribucion normal , contando con mayor afluencia en promedio entre 5 - 35 recuentos. De hecho , es notable como la Quema moderada es aquella que realmente cuenta con una frecuencia de valores altos que supera la quema baja , aunque claro un menor numero de usuarios en general.

Para finalizar , corroboramos la distribucion de mayor frecuencia de clase por usuario con un grafico torta:



Con este grafico , se confirma que la quema baja es una clase mas frecuente con 45.71% de repeticion siendo un 5.71% mayor a la Quema media . La quema alta no representa gran porcentaje de los usuarios.

Con estas visualizaciones se nos deja en claro lo siguiente:

- No todos los usuarios han registrado continuamente su quema de calorias , dejando cierta ambiguedad para el analisis.
- Si bien la mayoria de hipotesis anteriores fueron confirmadas , hay que denotar que la distribucion de datos refleja otro hecho. Pues , si bien la Quema baja puede ser mas frecuente la Quema media resulta repetirse mucho mas dias por un mayor numero de usuarios que la Quema baja.

Fase Act

Habiendo finalizado el caso de estudio , se tienen las siguiente conclusiones:

- El ritmo cardiaco de usuarios usualmente seguida una tendencia por usuario , sin presentar anomalias.
- Se denota un mayor promedio de ritmo cardiaco en las tardes y noches , representando una posibilidad de que el ejercicio es mayor en estos tiempos.

*La quema de calorias no tiene una tendencia exacta , pues los valores de quema diarios tienen gran variacion por usuario.

- Los usuarios suelen presentar mas frecuentemente una quema de calorias baja ; sin embargo , aquellos que suelen presentar una quema moderada de calorias suelen mantenerla durante mayores periodos de tiempo.

Se recomienda a la empresa Bellabeat y a la sstakeholder Urška Sršen lo siguiente:

- Realizar un estudio del ejercicio en periodos de dia y noche para corroborar que estos periodos sean los de mayor ejercicio.
- Es factible una linea de productos enfocada a periodos nocturnos para el publico objetivo.
- Implementar caracteristicas enfocadas a la quema de calorias en productos Bellabeat con el fin de advertir un sobreesfuerzo o de aumentar la intensidad de ejercicio , resultando en mayor adquisicion de otros productos bellabeat relacionados.
- Realizar un estudio enfocado a la quema de calorias , con posibilidad de iniciar una campaña en el cuidado fisico sobre la quema de calorias.

Se han contemplado las siguientes limitaciones:

- No todos los usuarios han registrado sus datos en todo el periodo de tiempo , lo que puede resultar en ambiguedad.
- No se tienen datos sobre la condicion fisica de los usuarios , limitando el analisis y sus hipotesis.
- El periodo de recoleccion fue de 62 dias calendario , este periodo puede ser considerado corto para objetivos de largo plazo.

Anexos

Repositorio del caso de estudio: https://github.com/ZDev-19/CaseStudyBellabeat.git Dashboard en Tableu Public:https://public.tableau.com/views/Bellabeat_case/Hearratedashboard?:language=es-ES&:sid=&:display_count=n&:origin=viz_share_link