Lab No.5

```
library(readxl)
library(nycflights13)
library(lubridate)
library(dplyr)
library(tidyverse)
```

Parte 1:

```
inicio_eclipse <- ymd_hms("2017-08-21 18:26:40")
synodic_month <- ddays(29) + dhours(12) + dminutes(44) + dseconds(3)
saros <- 223*synodic_month
next_eclipse <- inicio_eclipse + saros
print(paste("El siguiente eclipse sera en:",next_eclipse))
```

```
## [1] "El siguiente eclipse sera en: 2035-09-02 02:09:49"
```

Parte 2:

```
dat <- read_xlsx("data.xlsx") %>% filter(Call == 1)# 5,725
parse_date <- function(df,columna){
  t <- df[columna]
  parsed_date <- "empty"
  if (grepl("-",t,fixed=TRUE)){
    parsed_date <- as.character(dmy(t))
  } else {
    t_int <- as.integer(t)
    parsed_date <- as.character(as.Date(t_int,origin="1899-12-30")) # Origen fecha Excel
  }
  return(parsed_date)
}

dat["Fecha Final"] <- apply(dat,1,parse_date,"Fecha Final")
dat["Fecha Final"] <- as.Date(dat$`Fecha Final`)

dat["Fecha Creación"] <- apply(dat,1,parse_date,"Fecha Creación")
dat["Fecha Creación"] <- as.Date(dat$`Fecha Creación`)

dat["mes"] <- lapply(dat["Fecha Creación"], month)
dat["dow"] <- lapply(dat["Fecha Creación"], wday) # Day of Week
dat["hour"] <- lapply(dat["Hora Creación"], hour)
dat["length"] <- as.integer((dat$`Hora Final` - dat$`Hora Creación`)/60)
```

Pregunta 1:

```
dat %>%
  group_by(`Caller ID`,mes) %>%
  summarise(total_caller_mes = n()) %>%
  group_by(mes) %>%
  summarise(total_mes = n()) %>%
  arrange(desc(total_mes))
```

```
## # A tibble: 12 x 2
##      mes total_mes
##    <dbl>     <int>
##  1     7       493
##  2     5       489
##  3    11       489
##  4     3       486
##  5    10       480
##  6     6       471
##  7    12       470
##  8     8       469
##  9     9       462
## 10     1       460
## 11     4       457
## 12     2       434
```

Tanto Julio como Mayo son meses donde más usuarios hicieron llamadas.

Pregunta 2:

```
dat %>%
  group_by(dow) %>%
  summarise(total_dia = n()) %>%
  arrange(desc(total_dia))
```

```
## # A tibble: 7 x 2
##     dow total_dia
##   <dbl>     <int>
## 1     4       887
## 2     3       849
## 3     2       811
## 4     6       799
## 5     1       796
## 6     7       795
## 7     5       788
```

El día miercoles resultó ser el día con más actividad.

Pregunta 3:

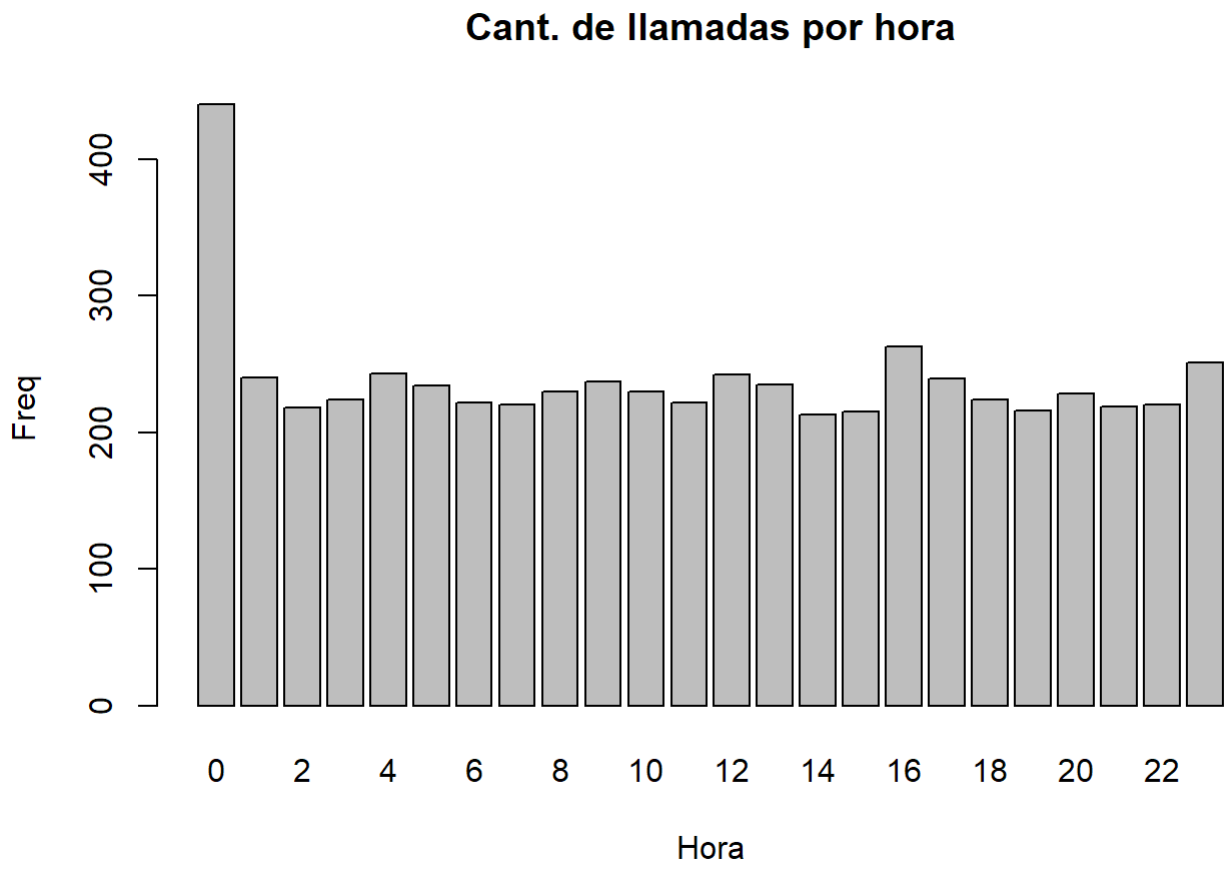
```
dat %>%
  group_by(mes) %>%
  summarise(total_dia = n()) %>%
  arrange(desc(total_dia))
```

```
## # A tibble: 12 x 2
##      mes total_dia
##    <dbl>     <int>
##  1     3       497
##  2     7       496
##  3     5       494
##  4    11       493
##  5    10       487
##  6    12       478
##  7     8       474
##  8     6       471
##  9     1       465
## 10     9       465
## 11     4       462
## 12     2       443
```

Marzo y Julio fueron los meses con más llamadas totales. No es sorpresa que Julio haya sido alto puesto que ya lo habiamos visto en nuestro analisis por llamada de codigos, sin embargo, Marzo resulta interesante ya que no estaba dentro del top 3 de la 1era pregunta. Esto nos hace pensar que si bien no tantos usuarios hicieron llamadas en Marzo, a comparación de otros meses, los usuarios que hicieron llamadas en este mes hicieron multiples llamadas.

Pregunta 4:

```
barplot(table(dat$hour),xlab="Hora",main="Cant. de llamadas por hora",ylab="Freq")
```



Los resultados pasados muestran que no existe una diferencia evidente entre meses y día de la semana. Sin embargo, al analizar las llamadas por hora podemos ver que existe un gran pico a las 12 am, esto puede ser un comportamiento de los usuarios o un error dentro de nuestro dataset.

Pregunta 5:

```
print(mean(dat$length))
```

```
## [1] 14.5579
```

```
print(median(dat$length))
```

```
## [1] 14
```

Pregunta 6:

```
dat %>%
  group_by(length) %>%
  summarise(freq = n())
```

```
## # A tibble: 31 x 2
##    length  freq
##     <int> <int>
##  1      0   221
##  2      1   211
##  3      2   173
##  4      3   195
##  5      4   193
##  6      5   184
##  7      6   194
##  8      7   197
##  9      8   212
## 10      9   166
## # ... with 21 more rows
```

Parte 3:

```r
your_sign <- function(str_date){
  birthday <- ymd(str_date)
  yrs_to_normalize <- 2020-year(birthday)
  sign <- ""

  birthday <- add_with_rollback(birthday,years(yrs_to_normalize),roll_to_first = FALSE)
  # Aries
  if (birthday %within% interval(ymd("2020-03-21"),ymd("2020-04-20"))){
    sign <- "Aries"
  }
  # Taurus
  else if(birthday %within% interval(ymd("2020-04-21"),ymd("2020-05-21"))){
    sign <- "Taurus"
  }
  # Gemini
  else if(birthday %within% interval(ymd("2020-05-22"),ymd("2020-06-21"))){
    sign <- "Gemini"
  }
  # Cancer
  else if(birthday %within% interval(ymd("2020-06-20"),ymd("2020-07-23"))){
    sign <- "Cancer"
  }
  # Leo
  else if(birthday %within% interval(ymd("2020-07-24"),ymd("2020-08-23"))){
    sign <- "Leo"
  }
  # Virgo
  else if(birthday %within% interval(ymd("2020-08-24"),ymd("2020-09-23"))){
    sign <- "Virgo"
  }
  # Libra
  else if(birthday %within% interval(ymd("2020-09-24"),ymd("2020-10-23"))){
    sign <- "Libra"
  }
  # Scorpio
  else if(birthday %within% interval(ymd("2020-10-24"),ymd("2020-11-22"))){
    sign <- "Scorpio"
  }
  # Sagitarius
  else if(birthday %within% interval(ymd("2020-11-23"),ymd("2020-12-21"))){
    sign <- "Sagitarius"
  }
  # Capricorn
  else if(birthday %within% interval(ymd("2020-12-22"),ymd("2020-12-31"))){
    sign <- "Capricorn"
  }
  # Capricorn 2no intervalo
  else if(birthday %within% interval(ymd("2020-01-01"),ymd("2020-01-20"))){
    sign <- "Capricorn"
  }
  # Aquarius
  else if(birthday %within% interval(ymd("2020-01-21"),ymd("2020-02-19"))){
    sign <- "Aquarius"
```

```
  }
  # Pisces
  else if(birthday %within% interval(ymd("2020-02-20"),ymd("2020-03-20"))){
    sign <- "Pisces"
  }
  return(sign)
}
```

Prueba

```
your_sign("1998-05-29")
```

```
## [1] "Gemini"
```

Parte 4:

```
# Quitamos filas con NAs
flights <- flights %>%
  drop_na(arr_time) %>%
  drop_na(dep_time)
```

```
# Para Departures podemos sumar durations
flights["sched_dep_time_hour"] <- flights["time_hour"] + flights["minute"]*60
flights["dep_time_hour"] <- flights["sched_dep_time_hour"] + flights["dep_delay"]*60
```

```
to_datetime <- function(df,type) {
  yr <- as.integer(df["year"])
  mes <- as.integer(df["month"])
  d <- as.integer(df["day"])


  if (type=="sched") {
    sched_arr <- as.integer(df["sched_arr_time"])
    sched_dep <- as.integer(df["sched_dep_time"])
    h <- as.integer(sched_arr/100)
    m <- as.integer(sched_arr%%100)
    if (sched_arr<sched_dep) {
      time_hour <- make_datetime(yr,mes,d+1,h,m,tz="UTC")
    } else {
      time_hour <- make_datetime(yr,mes,d,h,m,tz="UTC")
    }
  } else if (type=="real") {
    real_arr <- as.integer(df["arr_time"])
    real_dep <- as.integer(df["dep_time"])
    h <- as.integer(real_arr/100)
    m <- as.integer(real_arr%%100)
    if (real_arr<real_dep) {
      time_hour <- make_datetime(yr,mes,d+1,h,m,tz="UTC")
    } else {
      time_hour <- make_datetime(yr,mes,d,h,m,tz="UTC")
    }
  }
  return(time_hour)
}
```

```
flights["sched_arr_time_hour"] <- apply(flights,1,to_datetime,"sched")
flights["sched_arr_time_hour"] <- as.POSIXct(flights$sched_arr_time_hour,origin=origin,tz="UTC")

flights["arr_time_hour"] <- apply(flights,1,to_datetime,"real")
flights["arr_time_hour"] <- as.POSIXct(flights$arr_time_hour,origin=origin,tz="UTC")

flights["total_delay"] <- ((flights["arr_time_hour"]-flights["sched_arr_time_hour"]) + (flights[
"dep_time_hour"]-flights["sched_dep_time_hour"]))/60

flights["total_delay"] <- as.integer(flights$total_delay)
```

```
flights[0:5,c("sched_dep_time_hour","dep_time_hour")]
```

```
## # A tibble: 5 x 2
##   sched_dep_time_hour dep_time_hour
##   <dttm>              <dttm>
## 1 2013-01-01 05:15:00 2013-01-01 05:17:00
## 2 2013-01-01 05:29:00 2013-01-01 05:33:00
## 3 2013-01-01 05:40:00 2013-01-01 05:42:00
## 4 2013-01-01 05:45:00 2013-01-01 05:44:00
## 5 2013-01-01 06:00:00 2013-01-01 05:54:00
```

```
flights[0:5,c("sched_arr_time_hour","arr_time_hour")]
```

```
## # A tibble: 5 x 2
##   sched_arr_time_hour arr_time_hour
##   <dttm>              <dttm>
## 1 2013-01-01 08:19:00 2013-01-01 08:30:00
## 2 2013-01-01 08:30:00 2013-01-01 08:50:00
## 3 2013-01-01 08:50:00 2013-01-01 09:23:00
## 4 2013-01-01 10:22:00 2013-01-01 10:04:00
## 5 2013-01-01 08:37:00 2013-01-01 08:12:00
```

```
flights[0:5,c("total_delay")]
```

```
## # A tibble: 5 x 1
##   total_delay
##         <int>
## 1          13
## 2          24
## 3          35
## 4         -19
## 5         -31
```

```
#Delay en minutos
```