# R Programming – Comandos Importantes

## Leer un archivo CSV

library(readr)

my\_dataset <- read\_csv("C:/Users/n74197/Desktop/Escritorio/Mis Pagos/Coursera/1.R Programming/hw1\_dataNEW.csv")

View(my\_dataset)

## Subset – Subconjunto de datos de un dataset

calor <-subset(my\_dataset, Ozone >31 & Temp > 90)

mayo <- subset(my\_dataset, Month == 5)

subset(airquality, Temp > 80, select = c(Ozone, Temp))

subset(airquality, Day == 1, select = -Temp)

## Descartar registros con valores nulos

buenos = !is.na(mayo[,1])

mayo2 = mayo[buenos, ]

## Obtener máximo valor de una columna de un dataset

max(mayo2[,1])

## Ordenar dataframe por múltiples columnas

<https://stackoverflow.com/questions/1296646/how-to-sort-a-dataframe-by-multiple-columns>

## Obtener distinct de un vector

<https://stackoverflow.com/questions/7755240/list-distinct-values-in-a-vector-in-r>

## Crear dataframe vacío con nombres de columnas

<https://stackoverflow.com/questions/32712301/create-empty-data-frame-with-column-names-by-assigning-a-string-vector>

## initialize result dataframe

result <- data.frame(hospital=character(), state=character())

## Agregar row a un dataframe

<http://www.snippetexample.com/2014/09/add-new-row-dataframe-r/>

new\_row<- data.frame(hospital = NA, state = st)

result <- rbind(result, new\_row)

## Leer una planilla Excel

install.packages("readxl")

library(readxl)

ngap<- **read\_xlsx**("NGAP.xlsx")

## Leer un archivo XML

library("XML")

<http://productdesignjournal.blogspot.com/2014/11/r-code-for-reading-xml-file-and.html>

Let us assume that the file I am working with is the master data file of 100,000 employees. At any given point of time I want to find out how many employees live in a certain zip code.  
  
**Step 1 is to load the web address of the XML file in a vector.**fileURL < -"http://www.website.com/filename.xml"  
  
**Step 2 is to load all the content of the XML file in another vector.**documentcontent <- xmlTreeParse(fileURL, userInternalNodes=TRUE)  
  
**Step 3 is to parse the root node of the XML content and store it in another vector.**rootNode <- xmlRoot(documentcontent)  
  
**Step 4 is to extract all zip codes into a vector.**allzipcodes <- xpathSApply(rootNode, "//zipcode", xmlValue)  
  
**Step 5 is to count the number of people who have the zip code "90210".**sum(allzipcodes == "90210")  
  
In 5 simple steps you have performed meaningful data extraction from XML data, which normally requires very sophisticated and costly tools.  
  
To perform data extraction like this, you will need some basic understanding of XML and some logical thinking. If you are a cloud professional services or an SAP ERP HCM functional consultant, I believe you can perform basic data extraction like the one I described below using R, with a little bit of effort.

## Usar API de GitHub

<https://github.com/r-lib/httr/blob/master/demo/oauth2-github.r>

Install.packages(“httpuv”)

library(httr)

# 1. Find OAuth settings for github:

# http://developer.github.com/v3/oauth/

oauth\_endpoints("github")

# 2. To make your own application, register at

# https://github.com/settings/developers. Use any URL for the homepage URL

# (http://github.com is fine) and http://localhost:1410 as the callback url

#

# Replace your key and secret below.

myapp <- oauth\_app("github",

key = "56b637a5baffac62cad9",

secret = "8e107541ae1791259e9987d544ca568633da2ebf")

# 3. Get OAuth credentials

github\_token <- oauth2.0\_token(oauth\_endpoints("github"), myapp)

# 4. Use API

gtoken <- config(token = github\_token)

req <- GET("https://api.github.com/rate\_limit", gtoken)

stop\_for\_status(req)

content(req)

# OR:

req <- with\_config(gtoken, GET("https://api.github.com/rate\_limit"))

stop\_for\_status(req)

content(req)

## Ejecutar sentencias SQL sobre un dataframe

install.packages("sqldf")

library(sqldf)

acs<-read.csv("survey.csv")

sqldf("select pwgtp1 from acs where AGEP < 50")

## Leer archivo SST con columnas de ancho fijo

<https://stackoverflow.com/questions/14383710/read-fixed-width-text-file>

library(readr)

x <- read\_fwf(file="wksst8110.for", skip=4, fwf\_widths(c(12, 7, 4, 9, 4, 9, 4, 9, 4)))

## Merge de dos dataframes

<https://www.statmethods.net/management/merging.html>

Sirve para hacer un “join” entre ambos o para hacer un “union” de ambos.

## Multiples plots en una página

<https://cran.r-project.org/web/packages/egg/vignettes/Ecosystem.html>

<http://www.cookbook-r.com/Graphs/Facets_(ggplot2)/>

<http://environmentalcomputing.net/plotting-with-ggplot-adding-titles-and-axis-names/>