# Estadística 2.0

R es un software colaborativo de uso libre, que como lo expone el manual de introducción al programa (https://cran.r-project.org/doc/contrib/R-intro-1.1.0-espanol.1.pdf), tiene con un conjunto integrado de programas para manipulación de datos, cálculo y gráficos. Para el uso de la estadística, sus características principales pueden ser categorizadas en tres:

1. Almacenamiento y manipulación efectiva de datos,
2. Una amplia, coherente e integrada colección de herramientas para análisis de datos,
3. Posibilidades gráficas para análisis de datos, que funcionan directamente sobre pantalla o impresora.

Para el uso del software es necesario descargar e instalar el programa en nuestro computador, para ello podemos recurrir a los siguientes enlaces:

En caso de contar con un sistema operativo Windows

https://cran.r-project.org/bin/windows/base/

En caso de contar con un sistema operativo MAC

https://cran.r-project.org/bin/macosx/

## *Crear una Bases de Datos*

Si queremos disponer de información en R, el software nos permite crear o importar datos. Para crear una base de datos utilizaremos el comando data.frame.

# Supongamos que queremos crear una base de datos con dos características de los últimos cinco presidentes del Perú,

# Crearemos tres objetos, el apellido del presidente, edad del presidente cuando fue elegido y su partido político

```
presidente <- c ( 'Humala','Garcia','Toledo','Paniagua','Fujimori')
edad <- c ('49','57','55','64','52')
partido.politico <- c('GanaPeru','Apra','PeruPosible','AccionPopular','Cambio90')
```

# Ahora utilizaremos el comando **data.frame** para juntar nuestros tres objetos en una sola base de datos y lo llamaremos **pol.peru**

```
pol.peru<- data.frame(presidente, edad, partido.politico)
edit(pol.peru)
```

## *Importar Bases de Datos*

R nos permite exportar datos de diferentes tipos de  extensiones como xls o csv de Excel, dat de STATA, sav de SPSS, archivos txt, dbf, entre otros, o incluso importar base de datos desde la web. Para ello, la importación de estos archivos utiliza distintos comandos para cada extensión, en las siguientes líneas podrás encontrar algunos scripts para cargar tus bases datos en los softwares más comunes:

**1. De Excel**

**a. Archivos CSV**

```
data <- read.csv("midata1.csv")
```

**b. Archivos XLS**

```
# para importar archivos xls es necesario instalar y cargar el paquete gdata

install.packages("gdata")

library(gdata)

data <- read.xls ("pavimentando_con_votos.xls")
```

**2. De Texto**

```
data<- read.table("IDH.txt", fill = TRUE , header = TRUE )

#el argumento fill=TRUE permite que se mantenga el formato de la base de datos

#el header=TRUE permite que se mantenga la primera fila horizontal de títulos
```

**3. De SPSS**

```
#para importar archivos sav, primero se debe instalar y cargar el paquete foreign

install.packages("foreign")
library(foreign)

data <- read.spss("LAPOP2010.sav", use.value.labels=True, to.data.frame=True)

# el argumento use.value.label permitirá que se mantengan las etiquetas de las variables
# el argumento  to.data.frame es necesario para mantener el marco de base de datos
```

**4. De STATA**

#para importar archivos dta, primero se debe instalar y cargar el paquete foreign

install.packages(“foreign”)
library(foreign)

data <- read.dta("lima.dta")

**5. De DBF**

#para importar archivos dbf, primero debemos instalar la librería y cargar el paquete foreign
install.packages(“foreign”)
library(foreign)

data<-read.dbf("urba1990.dbf")

**6. Desde la Web**

#para poder cargar datas desde la Web es necesario disponer del URL desde donde se hará la descarga. En este caso trabajaremos con el siguiente URL:
http://s3.amazonaws.com/assets.datacamp.com/course/dasi/cdc.Rdata

#para descargar y cargar la data se utiliza el comando “load”

load(url("http://s3.amazonaws.com/assets.datacamp.com/course/dasi/cdc.Rdata"))

#puesto que el nombre del archivo de descarga es “cdc”, el R creará automáticamente un objeto con el mismo nombre, a partir del cual se hará el resto del trabajo

## *Merge en R*

#esta función se utiliza para combinar dos bases de datos que tengan por lo menos una variable en común, es necesario que en ambas bases de datos la variable en común sea idéntica en todos los casos y en el nombre de la variable.

#Supongamos que las bases de datos con la que vamos a trabajar ya se encuentra importada y hemos creado dos objeto para cada una “**mydata1**” y “**mydata2**”**.**

#Las bases de datos cuenta con una variable en común “**names**”

**newdata** <- **merge**(mydata1, mydata2, by=c(“names”))

# En el objeto **newdata** hemos creado una nueva base de datos con la información de ambas datas

# Si tenemos más de una variable en común, podemos agregar después de una coma el nombre de las otras variables. Por ejemplo:  c(“names”, **“year”, “city”**)

## *Append en R*

#Al igual que merge esta función se utiliza para combinar dos bases de datos pero la diferencia es que en append buscamos adherir casos a una base de datos inicial.

#Se requiere que ambas bases de datos tengas **variables exactamente iguales**

**newdata2**<- **rbind**(mydata3, mydata4})

#En el objeto **newdata2** hemos creado una nueva base de datos con la información de ambas datas. El comando **rbind** es el que realiza el proceso de juntarlos.

## *Filtrar datos en R*

#Supongamos que la base de datos con la que vamos a trabajar ya se encuentra importada y la hemos llamado “**data1**”,

# **data1** tiene las siguiente variables. P1, P2, P3, P4 y P5; y nosotros queremos filtrar una de las cuatro categorías de fila P1, que tiene cuatro categorías: **Nada Importante, Poco Importante, Importante y Muy Importante**.

**table** (data1$P1)

# Solicitamos la frecuencia de datos de nuestra variable P1

**data2** <- data1[data1$P1==”Nada importante”, ]

**#** Estamos creando un nuevo objeto (data2) que está filtrando la categoría “Nada importante” de P1 de la base de datos “data1”

**table** (data2$P1)

# Ahora estoy pidiendo la frecuencia de la variable P1 con el nuevo filtro y sólo otorgará los datos relativos de esa categoría.

# **Importante,** sólo hemos realizado el filtro para esa variable, en el resto de variables, data2 sigue conteniendo los variables relativos originales.

## *Eliminar una variable*

#Supongamos que hemos creado el objeto “datA” para importar una base de datos.

#Podemos eliminar la variable por su nombre o su posición en la base de datos.

#Primero eliminaremos la variable por su nombre, en este caso supondremos que existe una variable que se llama **Var1** y que queremos eliminar

data<-datA [ ,!colnames(datA)=="Var1"]

#Tambièn, podemos eliminar la variable por su posiciòn en el nùmero de columnas, es decir, si lo que queremos es eliminar una variable que está en la tercera columna utilizaremos el -3

data1<- datA[, -3]

## *Eliminar una fila*

#Supongamos que hemos creado la base de datos “datan1” y queremos eliminar la tercera fila para ello crearemos un nuevo objeto que se llamará “datan2”

datan2 <- datan1[-c(3),]

#Si queremos eliminar más casos sólo necesitamos poner el nùmero de la fila consecutivamente, por ejemplo : -c(3, 4, 5)

## *Split en R*

#Con esta función queremos separar una columna en dos, para ello tenemos que ubicar un delimitador común en toda la columna para tener en una columna la parte anterior al delimitador y en la otra columna la parte posterior

#Supongamos que tenemos una base de datos y la hemos llamado map1; además, tenemos una variable llamada “**ubicacion**” que contiene el punto de longitud y latitud de una ciudad en una sola variable y separado por **“:”** , por ejemplo: Huancayo [12º09'37”¨**:** 75º14'03”], ahora nosotros queremos separar la columna, en dos, una para longitud y otra para latitud.

#El comando que usamos en **rbind**

```
map2<- within(data=map1, ubicacion<-data.frame
        (do.call('rbind', strsplit (as.character (ubicacion), ":", fixed=TRUE))))
```