

MiDaS

Millenium Nucleus  
Center for the Discovery  
of Structures in Complex  
Data



FACULTAD DE MATEMÁTICAS  
PONTIFICIA UNIVERSIDAD  
CATÓLICA DE CHILE



# Aprendiendo $\mathbb{R}$

Natalie Julian

Olimpiada del Big Data

# ¿Cómo instalo RStudio en mi computador?

Instalaremos `RStudio`, un entorno muy amigable para aprender R:

- 1 En el siguiente link busque la versión de acuerdo a las características de su computador:

`Link`

Descargar y seguir pasos de instalación.

- 2 Una vez instalado el software anterior, en el siguiente link busque la versión de acuerdo a las características de su computador:

`Link`

Descargar y seguir pasos de instalación.

- 3 Finalmente abrir `RStudio`.

# ¿Y si no es posible instalar RStudio en mi computador?

En caso de que no pueda instalar los softwares necesarios, favor escribirme a najulian@mat.uc.cl. También en caso de tener problemas de internet u otro inconveniente para tenerlo en consideración.

Es posible utilizar RStudio de manera online:

- 1 Ingresa al siguiente link y crea tu cuenta:

Link

- 2 Al iniciar haz click en New project esto abrirá RStudio de manera online.

# Vista inicial

Una vez abierto RStudio, verificar se tiene la siguiente ventana:



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Console

Terminal x

Jobs x

R version 3.6.2 (2019-12-12) -- "Dark and Stormy Night"  
Copyright (C) 2019 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.  
Usted puede redistribuirlo bajo ciertas circunstancias.  
Escriba 'license()' o 'licence()' para detalles de distribución.

R es un proyecto colaborativo con muchos contribuyentes.  
Escriba 'contributors()' para obtener más información y  
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,  
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.  
Escriba 'q()' para salir de R.

> |

Environment

History

Connections

Import Dataset

Global Environment

Environment

Files

Plots

Packages

Help

New Folder

Delete Rename

Home > R

Name

..  
win-library

# La Ciencia: En búsqueda de la verdad

La ciencia, está hecha de errores, pero de errores útiles de cometer, pues poco a poco, conducen a la verdad.

**Julio Verne**

¿Cómo determinar científicamente *lo verdadero*?

- Realizar una transcripción de los fenómenos y sucesos del mundo de manera compacta: Es necesario cuantificar la información y almacenarla en **Datos**.
- Aplicar la lógica en los datos: Desarrollar análisis, usar metodología y procedimientos para obtener **Inferencias** basadas en **Evidencia** presente en los datos.

# La Estadística: Minera de evidencia

La estadística es el único tribunal de apelación para juzgar el nuevo conocimiento.

**Prasanta Chandra Mahalanobis**

La base del **Método Científico** es estudiar una **Hipótesis**, la cual responde una pregunta de investigación sobre la **Naturaleza**. En base a evidencia, esta hipótesis puede rechazarse o no rechazarse.

El rol de la Estadística en el procedimiento científico es:

- Determinar la fuerza de la evidencia presente en los datos, ya sea para rechazar o no rechazar la hipótesis de estudio
- Cuantificar dicha evidencia utilizando metodologías adecuadas en el contexto del problema
- Considerar e incorporar la incertidumbre presente en los datos
- Entregar un consenso en base a los datos

MiDaS

# La era de los datos

Big Data: ¿Qué hacer con tantos datos sin morir en el intento?

La transformación digital se convirtió en un tsunami

**Panamá usa equipo de inteligencia artificial para detección de Covid-19**

Machine learning could improve lung cancer screening

**Deep Learning en Gmail: un avance para combatir archivos adjuntos maliciosos**

# Lenguaje de programación R

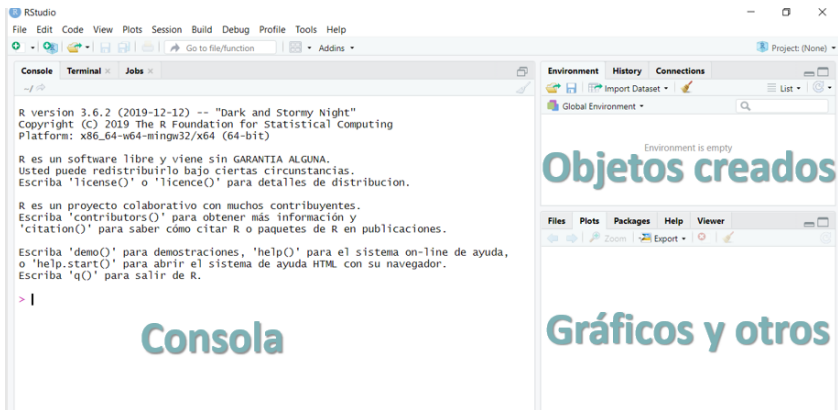
En el curso aprenderás metodología para la toma de decisiones. Dichas decisiones nacerán luego de desentrañar la evidencia estadística presente en los datos.

Aprenderemos a utilizar una de las herramientas más fuertes e imprescindibles a la hora de analizar datos: **manejo de software**.

El software que aprenderemos será **RStudio**, una plataforma de despliegue del lenguaje de programación **R**. Este lenguaje es eficaz para la manipulación de bases de datos y desarrollo de análisis estadísticos.



# Una mirada general a RStudio



MiDaS

# La consola como calculadora

Podemos realizar operaciones matemáticas en la consola. Algunas operaciones matemáticas que podemos realizar:

Función	Ejemplo 1	Ejemplo 2
Suma	<code>6+1</code>	<code>sum(2, 5)</code>
Multiplicación	<code>90*11</code>	<code>prod(4, 3)</code>
Valor absoluto	<code>abs(-2)</code>	<code>abs(-0.005)</code>
Potencia	<code>4**8</code>	<code>8^10</code>
Raíz cuadrada	<code>sqrt(4)</code>	<code>sqrt(0.05)</code>
División	<code>8/9</code>	<code>5/3</code>
Exponencial	<code>exp(-1/2)</code>	<code>exp(1)</code>
Logaritmo	<code>log(5, exp(2))</code>	<code>log(5, 3)</code>
Seno	<code>sin(10)</code>	<code>sin(pi)</code>

# Probando la consola

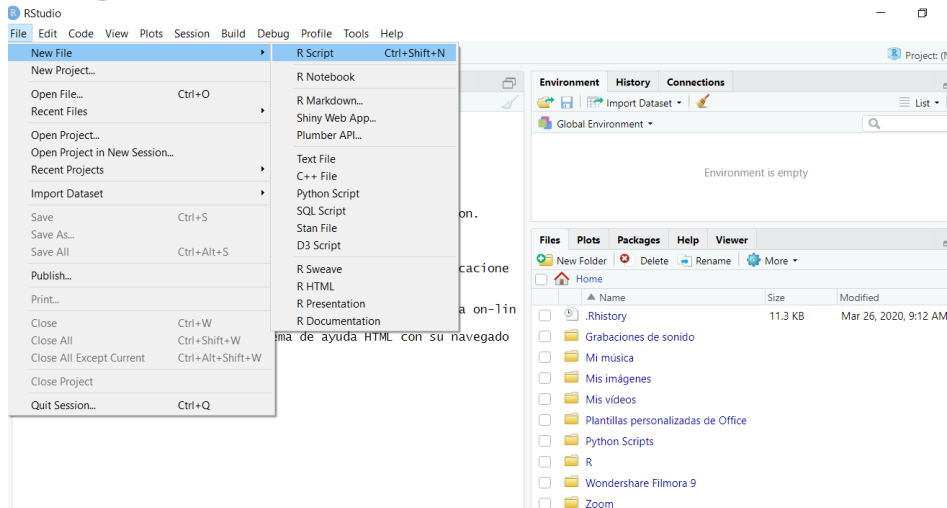
Copia las siguientes líneas de código y correlas en la consola. Observa el resultado.

```
2*2  
3/40  
sqrt(16)  
log(2,3)  
2*(3**5)
```

*Correr* una línea de código significa hacerla efectiva, generar una acción y entregar su resultado.

MiDaS

# Creación de un script para guardar los códigos





RStudio

Go to file/function



Untitled1 x

Source on Save

1

1:1 (Top Level) ↕

R Script ↕


Console Terminal  Jobs 

R es un proyecto colaborativo con muchos contribuyentes. Escriba 'contributors()' para obtener más información y 'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda, o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.

Escriba 'q()' para salir de R.

Environment History Connectivity



Global Environment ▾









Fi

Files Plots Packages Help

 New Folder  Delete  Ren

 Home

▲ Nam

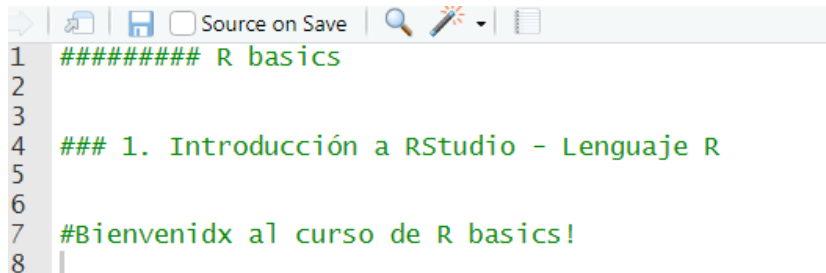
- ☐  .Rhistory
- ☐  Grabaciones de sonido
- ☐  Mi música
- ☐  Mis imágenes
- ☐  Mis videos
- ☐  Plantillas personalizadas d
- ☐  Python Scripts
- ☐  R

# MiDaS

## Primeros pasos en el script

Podemos escribir comentarios, anteponiendo el signo # antes del comentario.

Ejemplo:



The image shows a screenshot of an R script editor. The editor has a toolbar at the top with icons for navigation, saving, and searching. Below the toolbar, the script content is displayed with line numbers on the left. The script consists of four lines of text, all in green, which are comments. The first line is a multi-line comment '##### R basics'. The second line is empty. The third line is a section header '### 1. Introducción a RStudio - Lenguaje R'. The fourth line is a welcome message '#Bienvenidx al curso de R basics!'. The cursor is at the end of the fourth line.

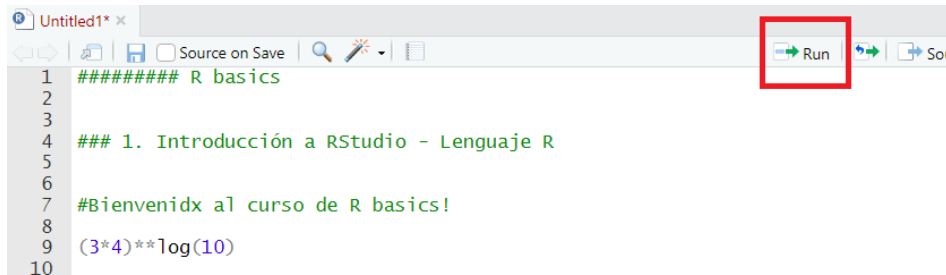
```
1 ##### R basics
2
3
4 ### 1. Introducción a RStudio - Lenguaje R
5
6
7 #Bienvenidx al curso de R basics!
8 |
```

# Correr una línea de código desde el script a la consola

Escribamos la siguiente línea de código en el script:

```
(3*4)**log(10)
```

Para correr la línea de código (esto es, generar una acción en la consola, generar el resultado de dicha línea de código), debemos hacer click en Run:



# Definiendo objetos en RStudio

Supongamos que queremos conocer el resultado de la siguiente operación:

```
(10000000000000*20) / (20**exp(1)) * log(20) / 10000000000000
```

Se ve un poco extenso, ¿verdad? Podemos escribir la expresión anterior de una forma análoga pero visualmente más sencilla y compacta. Asignemosle el valor 10000000000000 a "a" y el valor 20 a "b". Para eso, corra las siguientes líneas de código en el script:

```
a<-1000000000000000  
b<-20
```

Y podemos reescribir la enorme expresión anterior, de la siguiente forma:

```
(a*b) / (b**exp(1)) * log(b) / a
```

Y al correr dicha línea de código, obtenemos nuestro resultado en la consola:



## Resultado

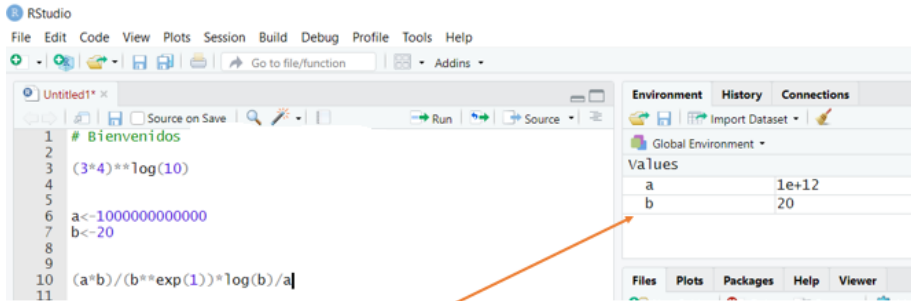
```
> a<-1000000000000000000
> b<-20
> (a*b)/(b**exp(1))*log(b)/a
[1] 0.01741674
> |
```

# Definiendo objetos en RStudio

Siempre que queramos definir un objeto debemos utilizar la flecha de asignación "<—" con la siguiente estructura:

```
nombreobjeto<-valor del objeto
```

Note que, cada vez que define objetos, estos objetos aparecen en la ventana derecha superior de RStudio:



# Ejercicios

- Asigne a un objeto llamado `impar` el valor raíz cuadrada de 625.
- Asigne a un objeto llamado `par` el valor 4 veces el seno de  $\frac{3\pi}{2}$ .
- Asigne a un objeto llamado `productoabs` el valor absoluto de la división  $\frac{par}{impar}$ .
- Consulte a la consola cuál es el valor de los objetos `impar`, `par` y `productoabs`.
- ¿Aparecieron los objetos creados en la ventana superior derecha?

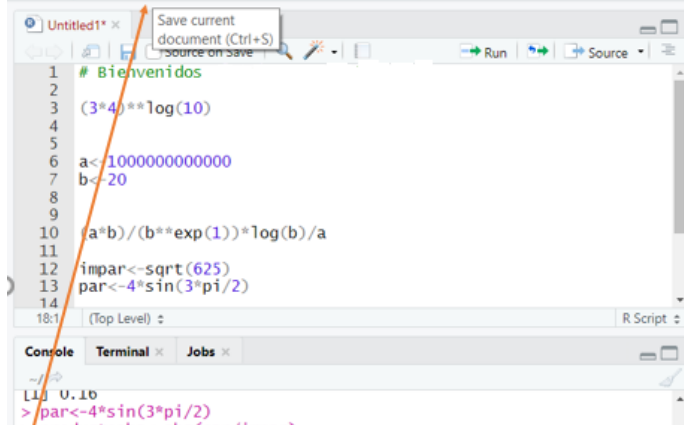
# Guardando mi script

Para guardar tu script sólo basta en hacer click en:

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins



```
1 # Bienvenidos
2
3 (3*4)**log(10)
4
5
6 a<-1000000000000
7 b<-20
8
9
10 (a*b)/(b**exp(1))*log(b)/a
11
12 impar<-sqrt(625)
13 par<-4*sin(3*pi/2)
14
```

Console

```
[1] 0.16
> par<-4*sin(3*pi/2)
```

Environment History Connection

Import Dataset

Global Environment

a	1e+12
b	20
impar	25
par	-4
productoabs	0.16
vector1	num [1:4

Files Plots Packages Help

New Folder Delete Rename

Home

Name

- .Rhistory
- Grabaciones de sonido
- Mi música
- Mis imágenes

MIDAS

## Próximamente...

En la próxima sesión trabajaremos con vectores. Para definir un vector en RStudio, se debe utilizar una estructura particular.

Por ejemplo, asignemos al objeto `vector1` el vector  $(2, -1, \frac{7}{3}, \frac{\pi}{2})$ .

En RStudio, esto se realiza de la siguiente forma:

```
vector1<-c(2,-1,7/3,pi/2)
```

**¡Profundizaremos más en la siguiente clase!**

# Vectores

Para definir vectores es necesario anteponer la letra `c` antes del paréntesis y separar los elementos del vector con coma.

Defina los siguientes vectores. ¿Qué tipo de vector es cada uno? (*Utilice el comando `class()`*).

```
cantidades<-c(-1,1.6,0,0,0,0,7)
nombres<-c("Miguel","José","Ricardo","María")
fracciones<-c(4/5,7,0,-1/3,8)
valorlogico<-c(TRUE,FALSE,TRUE,TRUE,TRUE,FALSE,FALSE,FALSE)
```

¿Qué tipo de vector es `c(1,2,"José","Ariel","B")`?

¿Conclusión?

# Secuencias

Pruebe las siguientes líneas de código en la Consola de RStudio:

```
0:4  
seq(0, 4)  
seq(0, 4, by=0.5)  
seq(0, 4, len=4)
```

¿Qué diferencias se producen al cambiar los argumentos en la función `seq()`?

# Vector de repeticiones

Pruebe las siguientes líneas de código en la Consola de RStudio:

```
rep(1, 5)  
rep(1:5, each=2)  
rep(c(1, 5), c(3, 7))  
rep(c(1, 5), len=9)
```

¿Qué diferencias se producen al cambiar los argumentos en la función `rep()`?



# Vector de vectores

Es posible crear un vector utilizando otros vectores, y así alternar el uso de los comandos `seq()` y `rep()`.

Vea los siguientes ejemplos:

```
c(rep(c(2, 3), 3), 0, 1, -2, seq(15, 17, by=0.5))  
c(seq(0, 5, by=1), rep(c(6, 7, 8), len=8), 1)
```

# Funciones aplicables a vectores

Función	Descripción
<code>length()</code>	Largo de un vector
<code>sum()</code>	Suma de los elementos de un vector
<code>min()</code>	Mínimo de los elementos de un vector
<code>which.min()</code>	En qué elemento(s) del vector se alcanza el mínimo
<code>max()</code>	Máximo de los elementos de un vector
<code>which.max()</code>	En qué elemento(s) del vector se alcanza el máximo
<code>mean()</code>	Media de los elementos de un vector
<code>median()</code>	Mediana de los elementos de un vector
<code>sd()</code>	Desviación estándar de los elementos de un vector
<code>var()</code>	Varianza de los elementos de un vector
<code>order()</code>	Entrega un vector de índices para ordenar los elementos de un vector
<code>class()</code>	Indica el tipo de elementos que posee un vector ( <i>numeric, character o logical</i> )

También es posible aplicar operaciones matemáticas a los vectores, como las funciones que ya vimos, `log()`, `exp()`, `**`, `sqrt()`, `sin()`, entre otras.

MiDaS

## Ejercicios

Se posee información sobre el porcentaje de obesidad en la población de distintos países:

País	Índice de obesidad (porcentaje)
Hungría	19,5
Reino Unido	25
Australia	27
Canadá	25
Irlanda	23
Luxemburgo	23
Estados Unidos	35

- Escriba la información en dos vectores: `País` y `Porcentaje`.  
¿Cuántos registros hay?
- Obtenga la media, la mediana, mínimo y máximo de los índices de obesidad. Comente.
- ¿Cuál es el mayor índice de obesidad en los datos? ¿y el menor?  
¿A qué países corresponden?

MiDaS

## Obteniendo verdades de los datos

Sea  $x$  la edad de Bastián y  $z$  la edad de Lorena. Se definen las edades a continuación:

$$x < -5$$

$$z < -10$$

Podemos obtener mucha información a partir de objetos creados, y para obtener esta información, debemos hacernos algunas preguntas:

- ¿Es Bastián menor que Lorena?
- ¿Se cumple que la edad de Bastián es menor que 10 y que la edad de Lorena es menor a 25?
- ¿Es cierto que la edad de Bastián es mayor a 5 o que la edad de Lorena es menor a 3?

# Lenguaje lógico en R

Operador	Significado	Aseveración lógica	Valor lógico
<	Menor que	$x < z$	TRUE
>	Mayor que	$x > z$	FALSE
$\leq$	Menor o igual que	$x \leq z$	TRUE
$\geq$	Mayor o igual que	$x \geq z$	FALSE
==	Igual que	$x == z$	FALSE
!=	Distinto a	$x != z$	TRUE
&	Y	$(x < 10) \& (z < 25)$	TRUE
	O (inclusivo)	$(x > 5)   (z < 3)$	FALSE

*El comando `which()` selecciona solo los valores TRUE de una prueba lógica, ¡este comando es muy útil cuando queremos saber qué elementos cumplen una condición!*

*Si también queremos conocer cuántos registros no cumplen la condición, podemos utilizar `table()`*

MiDaS

## Próximamente...

En la próxima sesión trabajaremos con matrices y tablas de datos.

¡Aprender sobre vectores es importante para comprender la estructura de las tablas de datos!

**¡Profundizaremos más en la siguiente clase!**

# Ejercicios

Continuemos trabajando con los porcentajes de obesidad:

- ¿Existen índices de obesidad inferiores a 19? ¿Cuáles?
- ¿Qué índices cumplen que son mayores a 20 y menores que 25?
- ¿Existe algún índice de obesidad que sea exactamente 24? ¿Y 27?
- ¿Existe algún índice de obesidad que se repita? ¿Cuál?
- ¿Cuántos países poseen un índice de obesidad superior a 30?

# Creando matrices con vectores

Para crear matrices utilizando vectores podemos utilizar `cbind()`, `rbind()` o `data.frame()`.

Si utilizamos `cbind()`, la información se guardará verticalmente:

v	v	....
e	e	....
c	c	....
t	t	....
o	o	....
r	r	....
1	2	....

Si utilizamos `rbind()`, la información se guardará horizontalmente:

vector1
vector2
.
.
.
.
.



## `data.frame()`

Cuando tengamos variables de distinto tipo, es decir, variables de tipo `character`, de tipo `numeric` o de tipo `logical`, a estas bases de datos las llamaremos bases de datos mixtas.

`data.frame()` opera de manera similar al comando `cbind()`, verticalmente.

¿Qué comando deberíamos utilizar para crear una matriz con la información de las variables `País` y `Porcentaje`?

Utilice el comando correspondiente y cree la base de datos con estas dos variables.

# Definiendo una matriz de datos con vectores

Supongamos que tenemos las mediciones de peso (en kilogramos) y altura (en centímetros) de 5 pacientes:

Peso (kg)	Altura (cms)
55	163
90	174
80	163
45	150
59	147

Como usted sabe, esto se puede definir en  $\mathbb{R}$  utilizando ciertos comandos.

## Usando cbind()

`cbind()` ordena la información por columnas (variables) de manera vertical:

```
peso<-c(55,90,80,45,59)    #Se define la información por columna
altura<-c(163,174,163,150,147)
```

```
datos<-cbind(peso,altura)
datos
```

```
      peso altura
[1,]  55  163
[2,]  90  174
[3,]  80  163
[4,]  45  150
[5,]  59  147
```

## Usando rbind()

`rbind()` ordena la información por filas (registros) de manera horizontal:

```
paciente1<-c(55,163)      #Se define la información por fila
paciente2<-c(90,174)
paciente3<-c(80,163)
paciente4<-c(45,150)
paciente5<-c(59,147)
```

```
(datosf<-rbind(paciente1,paciente2,paciente3,paciente4,paciente5))
```

```
      [,1] [,2]
paciente1 55 163
paciente2 90 174
paciente3 80 163
paciente4 45 150
paciente5 59 147
```

## Otros comandos para definir una matriz de datos

El comando `matrix()` al igual que los comandos `cbind()` o `rbind()` se utiliza cuando tenemos todas las variables del mismo tipo (todas `character` o todas `numeric`). En este caso, es necesario entregarle el argumento `nrow=5` y `ncol=2` para indicar que la matriz es tal que tiene 5 filas y 2 columnas:

```
datosm<-matrix(c(peso,altura),nrow=5,ncol=2)
datosm
```

```
  [,1] [,2]
[1,] 55 163
[2,] 90 174
[3,] 80 163
[4,] 45 150
[5,] 59 147
```

# Definiendo los datos manualmente en una matriz

Es posible ir definiendo los datos directamente en la función a utilizar:

```
matrix(c(55, 90, 80, 45, 59, 163, 174, 163, 150, 147), ncol=2, nrow=5)
      [,1] [,2]
[1,]  55 163
[2,]  90 174
[3,]  80 163
[4,]  45 150
[5,]  59 147
```

```
matrix(c(55, 163, 90, 174, 80, 163, 45, 150, 59, 147), byrow=TRUE, ncol=2, nrow=5)
      [,1] [,2]
[1,]  55 163
[2,]  90 174
[3,]  80 163
[4,]  45 150
[5,]  59 147
```

# Definiendo un conjunto de datos usando `data.frame()`

```
#Aquí se definen las variables peso y altura dentro del comando  
data.frame()
```

```
(DF<-data.frame("peso"=c(55,90,80,45,59),  
"altura"=c(163,173,163,150,147)))
```

```
  peso altura  
1  55    163  
2  90    173  
3  80    163  
4  45    150  
5  59    147
```

## Cambiando los nombres de las columnas de una tabla de datos

Muchas veces, los nombres de las columnas de una base de datos están en inglés o no son muy específicos. En este caso, considere que es relevante indicar que el peso se encuentra en kilogramos y que la altura se encuentra en centímetros. Para modificar el nombre de las columnas de un objeto de del tipo `data.frame` como el objeto `DF` definido anteriormente se utiliza el comando `colnames()`:

```
colnames(DF)<-c("peso (kg)", "altura (cms)")
```

DF

	peso (kg)	altura (cms)
1	55	163
2	90	174
3	80	163
4	45	150
5	59	147



# Cambiando los nombres de las filas de una planilla de datos

En este caso, tenemos que cada fila corresponde a la información de un paciente diferente, en este caso, al ser pocos pacientes, la idea de añadirle un nombre a cada fila de forma manual es factible, y se realiza de la siguiente forma:

```
rownames(DF) <- c("Paciente 1", "Paciente 2", "Paciente 3", "Paciente 4", "Paciente 5")
```

DF

	peso (kg)	altura (cms)
Paciente 1	55	163
Paciente 2	90	174
Paciente 3	80	163
Paciente 4	45	150
Paciente 5	59	147

# Funciones aplicables a dataframes

Función	Descripción
<code>dim()</code>	Dimensión de una planilla
<code>nrow()</code>	Número de filas (registros)
<code>ncol()</code>	Número de columnas (variables)
<code>names()</code>	Nombre de las columnas (variables)
<code>summary()</code>	Resumen estadístico de las variables
<code>head()</code>	Muestra los primeros registros de una planilla
<code>tail()</code>	Muestra los últimos registros de una planilla
<code>View()</code>	Muestra en una ventana aparte la planilla
<code>str()</code>	Muestra el tipo de variables que contiene la planilla
<code>class()</code>	Indica el formato de la planilla

*También es posible aplicar operaciones matemáticas a las matrices, como las funciones que ya vimos, `log()`, `exp()`, `**`, `sqrt()`, `sin()`, entre otras.*

## Próximamente...

En la próxima sesión aprenderemos a crear nuevas variables y añadirlas a la matriz DF.

**¡Profundizaremos más en la siguiente clase!**

# Extraer elementos de una planilla de datos

## Extraer una variable completa

Por ejemplo, si quisiéramos extraer del objeto `DF` la variable `peso` (kg) utilizamos:

```
DF$'peso (kg)' #Entrega el vector de pesos (kg)
```

También, note que la variable `peso` (kg) corresponde a la primera columna de la planilla `DF`, por lo que, es posible también extraer dicha columna de la siguiente forma:

```
DF[,1] #Entrega el vector de pesos (kg)
```

Análogamente se puede extraer la información de la variable `altura` (cms) de las siguientes dos formas:

```
DF$'altura (cms)' # Entrega el vector de alturas (cms)
```

```
DF[,2] #Entrega el vector de alturas (cms)
```

# Extraer elementos de una planilla de datos

## Extraer una fila completa

Por ejemplo, suponga que al nutricionista a cargo, le interesa conocer la información del Paciente 4, que corresponde a la cuarta fila del objeto `DF`. Esto es posible utilizando la siguiente línea de código:

```
DF [ 4, ]
```

## Extraer una posición en particular

Si quisiéramos extraer solo la altura del tercer paciente del objeto `DF` se utiliza lo siguiente:

```
DF [ 3, 2 ]
```

# Extraer elementos de una planilla

## Extraer varias filas o columnas

Para extraer varias filas o columnas es análogo al procedimiento anterior, pero se debe especificar un vector. Por ejemplo, suponga que interesa conocer el peso y altura de los pacientes 1, 3 y 5, esto se puede realizar de la siguiente forma:

```
DF[seq(1, 5, by=2), 1:2]
```

	peso (kg)	altura (cms)
Paciente 1	55	163
Paciente 3	80	163
Paciente 5	59	147

## Ejercicio

La fórmula del IMC es peso (kg) dividido en la altura (mt) al cuadrado.  
Calcule el IMC para todos los pacientes.

# Solución

```
DF$`peso (kg)`/(DF$`altura (cms)`/100)**2
```

Extrae variable peso en kilogramos

Extrae variable altura en centímetros

Se eleva al cuadrado la altura en metros por fórmula del IMC

Se le divide en 100 para convertirla a metros

```
DF$`peso (kg)`/(DF$`altura (cms)`/100)**2  
[1] 20.70082 30.07117 30.11028 20.00000 27.30344
```



## Añadiendo el IMC al objeto DF

Es posible añadir la información recién calculada a la tabla DF en una nueva columna:

```
DF[,3]<-DF$`peso (kg)`/(DF$`altura (cms)`/100)**2
```

DF

	peso (kg)	altura (cms)	V3
Paciente 1	55	163	20.70082
Paciente 2	90	173	30.07117
Paciente 3	80	163	30.11028
Paciente 4	45	150	20.00000
Paciente 5	59	147	27.30344

Y si queremos añadir el nombre de IMC a la tercera columna, utilizamos:

```
colnames(DF)[3]<-"IMC"
```

# Resultado

DF

	peso (kg)	altura (cms)	IMC
Paciente 1	55	163	20.70082
Paciente 2	90	173	30.07117
Paciente 3	80	163	30.11028
Paciente 4	45	150	20.00000
Paciente 5	59	147	27.30344

Una tabla de datos bien lograda siempre es autoexplicativa por sí misma.

Detalles importantes:

- Especificar el nombre de cada una de las variables
- Especificar el formato de las variables con cierta ambigüedad

## Ejercicio

¿Qué sintaxis hubiera utilizado si solo hubiera querido calcular el IMC para aquellos pacientes que presentaban un peso mayor a 60 kilogramos?

# Solución 1

Extraemos del objeto DF

Seleccionamos la primera columna que corresponde al peso (kg)

Se eleva al cuadrado la altura en metros por fórmula del IMC

```
DF[which(DF$'peso (kg) '>60),1]/(DF[which(DF$'peso (kg) '>60),2]/100)**2
```

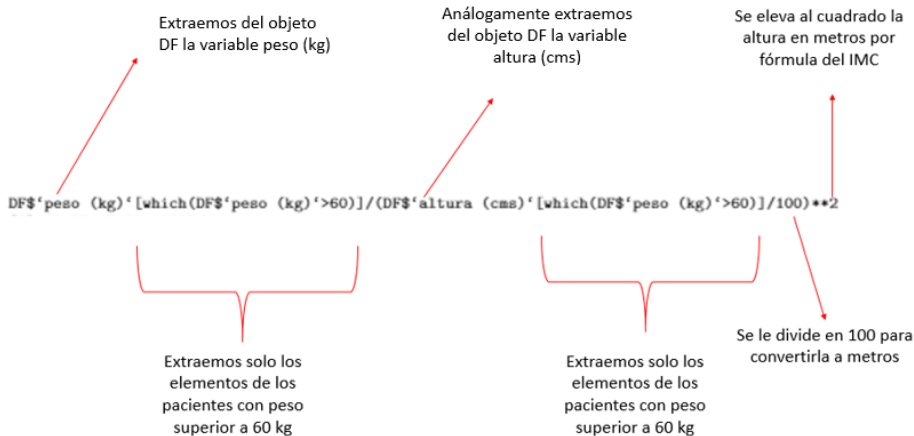
Extraemos sólo los registros asociados a pacientes con peso superior a 60 kg

Análogamente, extraemos la variable altura (cms) de sólo los pacientes con peso superior a 60 kg

Se le divide en 100 para convertirla a metros

```
DF[which(DF$'peso (kg) '>60),1]/(DF[which(DF$'peso (kg) '>60),2]/100)**2  
[1] 30.07117 30.11028
```

## Solución 2



```
DF$`peso (kg)`[which(DF$`peso (kg)`>60)]/(DF$`altura  
(cms)`[which(DF$`peso (kg)`>60)]/100)**2  
[1] 30.07117 30.11028
```

## Importando Bases de Datos en R

Ya aprendimos cómo definir manualmente planillas de datos y matrices en R. Sin embargo, no podemos definir manualmente una matriz de datos de miles de registros.

Usualmente, las bases de datos son creadas de forma automática y guardadas en un formato que sea legible en los softwares de análisis estadísticos.

Algunos formatos de bases de datos son:

- Archivo de valores de Microsoft Excel `.csv`
- Hoja de cálculo de Microsoft Excel `.xlsx`
- Documento de texto `.txt`
- Bases de datos de otros softwares, por ejemplo: Stata, SPSS, SAS.

# Bases de datos que trabajaremos

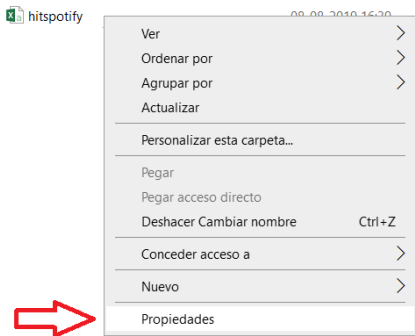
Cargaremos y analizaremos distintas bases de datos en R:

- `hitspotify`: Información de las 50 canciones más escuchadas de Spotify en el año 2019. El fin es concluir sobre la popularidad de las canciones y las distintas variables que podrían influir en ésta.
- `breast-cancer`: Información de núcleos celulares obtenidos a partir de extracciones de masa del seno con presencia de tumor. El fin es estudiar y determinar características asociadas a tumores benignos o malignos (radio promedio del núcleo de las células, perímetro promedio, etcétera).
- `pokedex`: Información de pokemones, su clasificación y rendimiento bajo distintas métricas (Attack, Speed, Hit Points, y otras). Interesa analizar las diferencias que puedan presentarse al estar en una clasificación u otra.

# ¿Cómo saber en qué formato están guardados los datos?

Antes de cargar una base de datos es muy importante conocer algunos detalles de ésta. Por ejemplo, el formato, el tamaño, la ruta.

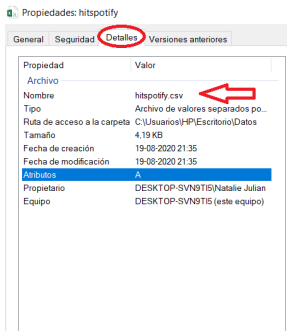
Para saber qué formato tiene el archivo, hacemos click derecho en el archivo y seleccionamos Propiedades.





# ¿Cómo saber en qué formato están guardados los datos?

Se abrirá una ventana y hacemos click en Detalles:



Podemos reconocer que los datos `hitspotify` en formato `.csv`, tiene un tamaño de 4.19 KB y la ubicación donde se encuentra guardado.

# ¿Por qué es necesario conocer el formato del archivo de datos?

En R existe un apartado `Import Dataset` donde podemos importar datos dependiendo del formato de estos:

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Console Terminal x Jobs x

~/

R version 3.6.2 (2019-12-12) -- "Dark and Stormy Night"  
Copyright (C) 2019 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.  
Usted puede redistribuirlo bajo ciertas circunstancias.  
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.  
Escriba 'contributors()' para obtener más información y  
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,  
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.  
Escriba 'q()' para salir de R.

[Workspace loaded from ~/.RData]

Environment History Connect

Import Dataset

Global Environment

Environment is e

Files Plots Packages Help

Home Find in Topic

MiDaS

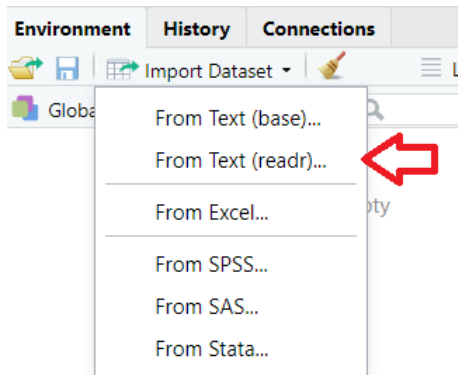
## Próximamente...

En la próxima sesión importaremos las bases de datos y realizaremos análisis de cada una de ellas.

**¡Profundizaremos más en la siguiente clase!**

## Import Dataset

Ya sabemos que los datos *hitspotify* están en formato *.csv*. Para cargar bases de datos cuyo formato sea *.csv* o *.txt* debemos seleccionar `From Text (readr)`.



# Instalando un paquete

Al hacer click en From Text (`readr`) se abrió un mensaje en RStudio. Hacer click en Yes (R comenzará a instalar el paquete `readr` que necesitamos para cargar nuestros datos).

Studio

Edit Code View Plots Session Build Debug Profile Tools Help

Console Terminal Jobs

version 3.6.2 (2019-10-10)  
copyright (C) 2019 The R Foundation for Statistical Computing  
platform: x86\_64-w64-mingw32

es un software libre  
usted puede redistribuirlo y/o modificarlo  
scriba 'license()' o 'help(license())' para saber más

es un proyecto colaborativo  
scriba 'contributors()' para ver los contribuyentes  
'citation()' para saber cómo citar R

scriba 'demo()' para ejecutar ejemplos  
'help.start()' para abrir el sitio web de R  
scriba 'q()' para salir de R.

workspace loaded from ~/.RData]

Installing Packages

Stop

Installing package into 'C:/Users/HP/Documents/R/win-library/3.6'  
(as 'lib' is unspecified)

Conn

aset

t (base)

t (readr

el...

S...

S...

ta...

s He

Home Find in Topic

MiDaS

# Paquetes en R

¿Qué es un paquete en R?

RStudio trae funciones base como todas las que hemos utilizado hasta ahora, por ejemplo, *seq()*, *cbind()*, *which()*, *data.frame()*, son funciones que se encuentran por defecto en R, pero muchas veces necesitamos utilizar funciones adicionales, por lo cual, necesitamos instalar paquetes que posean las funciones que necesitamos utilizar.

## library()

Cada vez que se quiera utilizar un paquete, éste debe instalarse y para cargarlo se utiliza `library()`. En este caso, estamos instalando el paquete `readr()` que posee una función para cargar datos en formato `.csv` o `.txt`. Cuando la instalación se haya completado, se abrirá una ventana:

Import Text Data

File/URL:

Browse...

Data Preview:

Import Options:

Name:  ☒ First Row as Names Delimiter:  Escape:

Skip:  ☒ Trim Spaces Quotes:  Comment:

☒ Open Data Viewer Locale:  NA:

Code Preview:

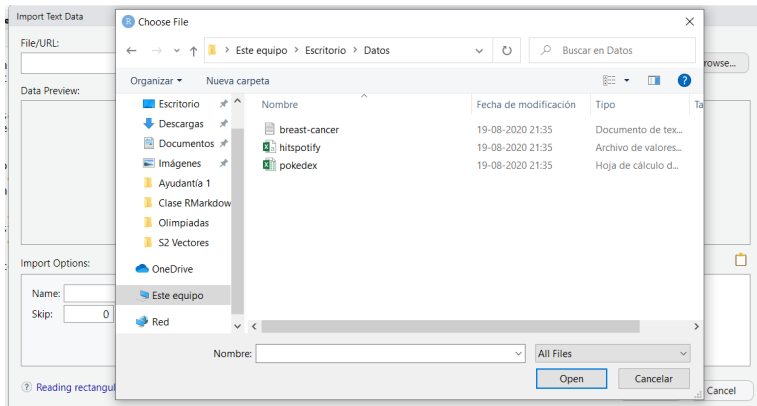
```
library(readr)
dataset <- read_csv
(NULL)
View(dataset)
```

? Reading rectangular data using readr

Import Cancel

# Importando datos en R

Para cargar los datos necesitamos hacer click en `Browse`, se abrirá una ventana para que busquemos la ubicación del archivo *hitspotify* y lo seleccionamos.



Al hacer click en `Open` se abrirá una vista de la base de datos en la ventana de importación de datos.



# Importando datos en R

Import Text Data

File/URL:  
C:/Users/HP/Desktop/Datos/hitspotify.csv Browse...

Data Preview:

X1 (double)	Track.Name (character)	Artist.Name (character)	Genre (character)	Beats.Per.Minute (double)	Energy (double)	Da
1	Seorita	Shawn Mendes	canadian pop	117	55	
2	China	Anuel AA	reggaeton flow	105	81	
3	boyfriend (with Social House)	Ariana Grande	dance pop	190	80	
4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93	65	

Previewing first 50 entries.

Import Options:

Name:  ☒ First Row as Names Delimiter:  Escape:

Skip:  ☒ Trim Spaces Quotes:  Comment:

☒ Open Data Viewer Local:  NA:

Code Preview:

```
library(readr)
hitspotify <-
read_csv("C:/Users/HP/Desktop/Datos/hitspotify.csv")
View(hitspotify)
```

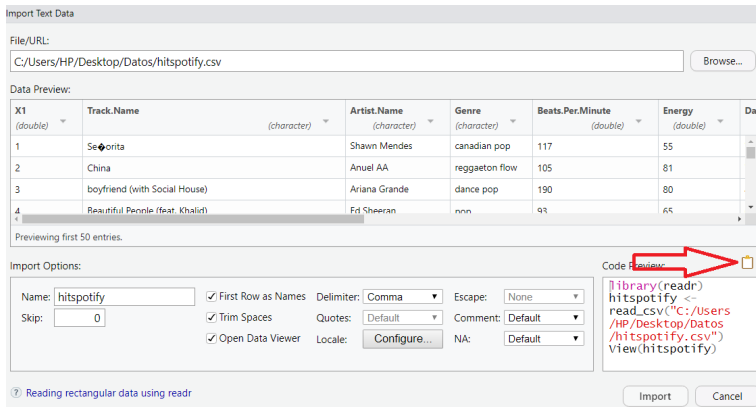
[? Reading rectangular data using readr](#)

En este paso es muy importante verificar que la base de datos se lea correctamente. En este caso, podemos observar que las columnas se separan de manera adecuada, que cada columna tiene su nombre especificado excepto la primera (que corresponde al ID).

MiDaS

# ¿Cómo obtener mi código para cargar datos?

Una vez que verificamos que la base de datos se lee correctamente, podemos proceder a importarla. Para obtener el código de la importación, debemos hacer click aquí:



Import Text Data

File/URL:

C:/Users/HP/Desktop/Datos/hitspotify.csv

Browse...

Data Preview:

X1	Track.Name	Artist.Name	Genre	Beats.Per.Minute	Energy	Da
1	Seorita	Shawn Mendes	canadian pop	117	55	
2	China	Anuel AA	reggaeton flow	105	81	
3	boyfriend (with Social House)	Ariana Grande	dance pop	190	80	
4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93	65	

Previewing first 50 entries.

Import Options:

Name: hitspotify

Skip: 0

☒ First Row as Names

☒ Trim Spaces

☒ Open Data Viewer

Delimiter: Comma

Quotes: Default

Locale: Configure...

Escape: None

Comment: Default

NA: Default

Code Preview:

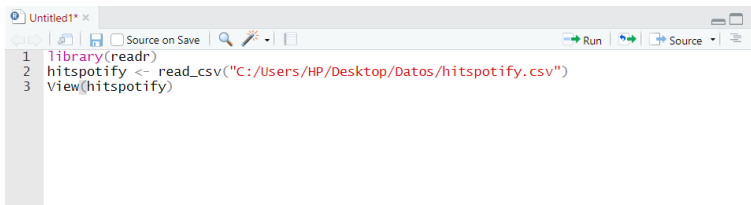
```
library(readr)
hitspotify <-
read_csv("C:/Users
/HP/Desktop/Datos
/hitspotify.csv")
View(hitspotify)
```

Import Cancel

Lo anterior copia automáticamente el código de importación de la base de datos. Hacemos click en **Import** (los datos se cargan en R).

MIPAs

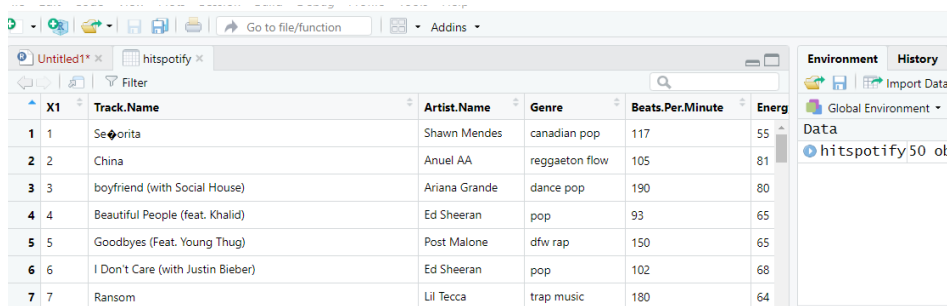
# Código de importación de los datos *hitspotify*



```
1 library(readr)
2 hitspotify <- read_csv("C:/Users/HP/Desktop/Datos/hitspotify.csv")
3 View(hitspotify)
```

Una vez pegado el código en el script, lo corremos y la base de datos se cargará con el nombre de `hitspotify` en R.

# Mirando una planilla de datos cargada en R



X1	Track.Name	Artist.Name	Genre	Beats.Per.Minute	Energy
1	Señorita	Shawn Mendes	canadian pop	117	55
2	China	Anuel AA	reggaeton flow	105	81
3	boyfriend (with Social House)	Ariana Grande	dance pop	190	80
4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93	65
5	Goodbyes (Feat. Young Thug)	Post Malone	dfw rap	150	65
6	I Don't Care (with Justin Bieber)	Ed Sheeran	pop	102	68
7	Ransom	Lil Tecca	trap music	180	64

Descripción de los datos hitspotify aquí

# Ejercicios

- Realice un análisis de las características de la base de datos `hitspotify`. Comente.
- ¿Existe alguna columna de la base de datos que no sea relevante analizar? ¿Cuál y por qué? Quite dicha variable de la base de datos. Verifique que efectivamente se quitó dicha variable.
- Entregue un análisis estadístico de las variables contenidas en la base de datos.
- ¿Cuál fue la canción más popular en Spotify el año 2019? ¿Cuáles son las otras 3 canciones más populares?
- A una industria musical le interesa obtener inferencias sobre las canciones más populares. Defina una base de datos `populars` que contenga solo las canciones cuya popularidad se encuentre en el 25% superior de *Popularity*. (Para obtener el cuantil  $q\%$  se utiliza `quantile(variable, q)`). ¿Cuántas canciones cumplen con la condición anterior? ¿Cuáles son?
- Utilizando la data `populars`, ¿cuáles son los generos más populares?

# Ejercicios

- Determine el tipo de formato de los datos `breast-cancer` y `pokedex`. Cárguelas en R como corresponda y realice inferencias de interés de estas tablas de datos.

Descripción de `breast-cancer` aquí

Descripción de `pokedex` aquí