

# PROBABILIDAD Y ESTADÍSTICA

## Curso de R

**Gustavo A. Colmenares**

[gcolmenares@yachaytech.edu.ec](mailto:gcolmenares@yachaytech.edu.ec)

[gcolmena@gmail.com](mailto:gcolmena@gmail.com)

### 7- Importar y exportar datos

## Importar y exportar datos

Hasta ahora, hemos trabajado con datos ya existentes en **R** base o que hemos generado nosotros mismos, sin embargo, es frecuente que usemos datos almacenados en archivos fuera de **R**.

Se puede importar datos de una amplia variedad de tipos de archivo con las funciones de **R** base y además esta capacidad puede ser ampliada con el uso de paquetes específicos.

Cuando importamos un archivo, estamos guardando su contenido en nuestra sesión como un objeto. Dependiendo del procedimiento que usemos será el tipo de objeto creado.

De manera análoga, podemos exportar nuestros objetos de **R** a archivos en nuestra computadora.

## Importar y exportar datos

En primer lugar, vale la pena señalar que podemos descargar archivos de internet usando **R** con la función **download.file()**. De esta manera tendremos acceso a una vasta diversidad de fuentes de datos. Entre otras, podrás descargar los archivos

La función **download.file()** nos pide como argumento la dirección url, es decir, la dirección de internet del archivo que queremos descargar. En esta función es importante el argumento **destfile** que representa el nombre que tendrá el archivo en nuestra computadora. Ambos argumentos como cadenas de texto, es decir, entre comillas.

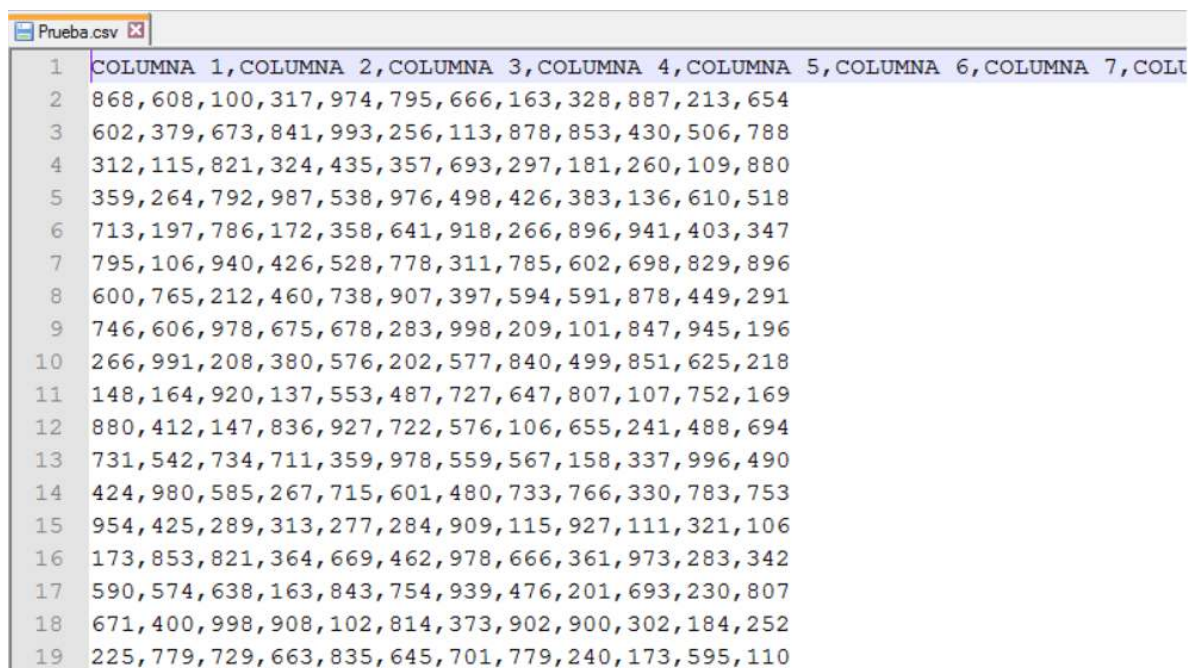


## Importar tablas (datos rectangulares)

Las estructuras rectangulares, en filas y columnas, son bastante comunes como fuentes de datos.

**R** cuenta con la función genérica `read.table()`, que puede leer cualquier tipo de archivo que contenga una tabla.

La condición para que **R** interprete un archivo como una tabla es que tenga filas y en cada fila, los datos estén separados por comas, o algún otro carácter, indicando columnas.



```
1 COLUMNA 1,COLUMNA 2,COLUMNA 3,COLUMNA 4,COLUMNA 5,COLUMNA 6,COLUMNA 7,COLUMNA 8
2 868,608,100,317,974,795,666,163,328,887,213,654
3 602,379,673,841,993,256,113,878,853,430,506,788
4 312,115,821,324,435,357,693,297,181,260,109,880
5 359,264,792,987,538,976,498,426,383,136,610,518
6 713,197,786,172,358,641,918,266,896,941,403,347
7 795,106,940,426,528,778,311,785,602,698,829,896
8 600,765,212,460,738,907,397,594,591,878,449,291
9 746,606,978,675,678,283,998,209,101,847,945,196
10 266,991,208,380,576,202,577,840,499,851,625,218
11 148,164,920,137,553,487,727,647,807,107,752,169
12 880,412,147,836,927,722,576,106,655,241,488,694
13 731,542,734,711,359,978,559,567,158,337,996,490
14 424,980,585,267,715,601,480,733,766,330,783,753
15 954,425,289,313,277,284,909,115,927,111,321,106
16 173,853,821,364,669,462,978,666,361,973,283,342
17 590,574,638,163,843,754,939,476,201,693,230,807
18 671,400,998,908,102,814,373,902,900,302,184,252
19 225,779,729,663,835,645,701,779,240,173,595,110
```

Archivo delimitado por comas (,)  
Extensión .csv

## Importar tablas (datos rectangulares)

Por supuesto, en lugar de comas podemos tener puntos y coma, dos puntos, tabuladores o cualquier otro signo de puntuación como **separador** de columnas.

1	COLUMNA 1	COLUMNA 2	COLUMNA 3	COLUMNA 4	COLUMNA 5	COLUMNA 6	COLUMNA 7	COLUMNA 8	COLUMNA 9			
2	868	608	100	317	974	795	666	163	328	887	213	654
3	602	379	673	841	993	256	113	878	853	430	506	788
4	312	115	821	324	435	357	693	297	181	260	109	880
5	359	264	792	987	538	976	498	426	383	136	610	518
6	713	197	786	172	358	641	918	266	896	941	403	347
7	795	106	940	426	528	778	311	785	602	698	829	896
8	600	765	212	460	738	907	397	594	591	878	449	291
9	746	606	978	675	678	283	998	209	101	847	945	196
10	266	991	208	380	576	202	577	840	499	851	625	218
11	148	164	920	137	553	487	727	647	807	107	752	169
12	880	412	147	836	927	722	576	106	655	241	488	694
13	731	542	734	711	359	978	559	567	158	337	996	490
14	424	980	585	267	715	601	480	733	766	330	783	753
15	954	425	289	313	277	284	909	115	927	111	321	106
16	173	853	821	364	669	462	978	666	361	973	283	342
17	590	574	638	163	843	754	939	476	201	693	230	807
18	671	400	998	908	102	814	373	902	900	302	184	252
19	225	779	729	663	835	645	701	779	240	173	595	110
20	250	449	131	703	349	759	970	353	814	207	305	456
21	577	311	1198	559	200	655	1884	785	751	1239	184	1108

Delimitado por el carácter '|'

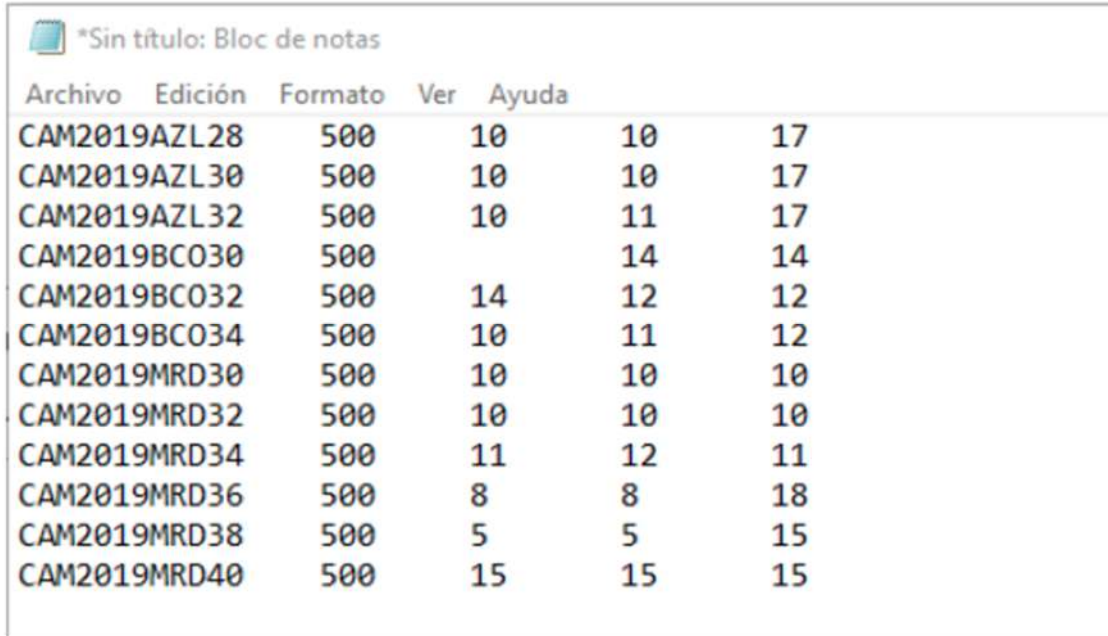


# Importar tablas (datos rectangulares)

```
Open;High;Low;Close;Adj Close;Volume
56,040001;56,189999;55,419998;55,48;52,345066;27334100
54,32;54,799999;53,389999;54,799999;51,703487;53778000
54,93;55,389999;54,540001;55,049999;51,939362;34079700
54,32;54,400002;53,639999;54,049999;50,995865;39518900
52,700001;53,490002;52,07;52,169998;49,222095;56564900
52,369999;53,279999;52,150002;52,330002;49,373062;48754000
52,509998;52,849998;51,459999;52,299999;49,344746;36663600
52,759998;53,099998;52,060001;52,779999;49,797626;36095500
53,799999;54,07;51,299999;51,639999;48,722046;66883600
52;53,419998;51,57;53,110001;50,108986;52381900
51,310001;51,970001;50,34;50,990002;48,108776;71820700
51,48;51,68;50,060001;50,560001;47,703075;43564500
49,98;51,380001;49,099998;50,790001;47,920078;63273000
51;51,580002;50,299999;50,48;47,627583;40191200
51,41;52,330002;51,259998;52,290001;49,33532;37555800
51,939999;52,650002;51,650002;51,790001;48,863567;34707700
51,790001;52,439999;51,549999;52,169998;49,222095;28699500
52,009998;52,200001;51,02;51,220001;48,325783;36775200
51,860001;52,209999;51,25;52,060001;49,118313;62513800
54,73;55,09;54,55,09;51,977104;83611700
54,880001;55,09;54,5;54,709999;51,618568;44208500
54,169998;54,259998;52,650002;53;50,005196;56313800
53,25;53,389999;51,259998;52,16;49,212662;57559800
52,099998;52,810001;51,369999;52;49,061703;46803400
51,939999;52;49,560001;50,16;47,325676;62009000
49,549999;49,57;48,189999;49,41;46,61805;59290500
49,02;50,240002;48,669998;49,279999;46,495396;45822200
49,889999;50,389999;49,52;49,709999;46,9011;38237000
48,68;50,110001;48,509998;49,689999;46,882233;48878600
50,25;50,68;49,75;50,5;47,646461;34243300
50,900002;51,09;50,130001;51,09;48,549217;37291200
51,100002;52,77;51,150001;52,110002;49,813076;40780000
```

Delimitado por el carácter ‘;’

# Importar tablas (datos rectangulares)



\*Sin título: Bloc de notas

Archivo	Edición	Formato	Ver	Ayuda
CAM2019AZL28	500	10	10	17
CAM2019AZL30	500	10	10	17
CAM2019AZL32	500	10	11	17
CAM2019BCO30	500		14	14
CAM2019BCO32	500	14	12	12
CAM2019BCO34	500	10	11	12
CAM2019MRD30	500	10	10	10
CAM2019MRD32	500	10	10	10
CAM2019MRD34	500	11	12	11
CAM2019MRD36	500	8	8	18
CAM2019MRD38	500	5	5	15
CAM2019MRD40	500	15	15	15

Delimitado por tabulaciones

## Importar tablas (datos rectangulares)

La función **read.table()** acepta un número considerable de argumentos. Los más importantes son los siguientes:

- **file**: La ruta del archivo que importaremos, como cadena de texto. Si el archivo se encuentra en nuestro directorio de trabajo, es suficiente dar el nombre del archivo, sin la ruta completa.
- **header**: Si nuestro archivo tiene encabezados, para ser interpretados como nombres de columna, definimos este argumento como TRUE.
- **sep**: El caracter que es usado como separador de columnas. Por defecto es “;”.

Es importante señalar que el objeto obtenido al usar esta función es siempre un **data frame**.





## Hojas de cálculo de Excel

Un formato usado con mucha frecuencia para almacenar archivos son las hojas de cálculo, en particular las generadas por el paquete Microsoft Excel.

**R** base no tiene una función para importar archivos almacenados en archivos con extensión .xls y .xlsx, creados con Excel.

Para importar datos desde este tipo de archivos, necesitamos instalar el paquete **readxl**, que contiene funciones específicas para realizar esta tarea. Para ello usamos la función **installpackages()**:



## Exportar datos

Un paso muy importante en el trabajo con **R** es exportar los datos que hemos generado, ya sea para que sean usados por otras personas o para almacenar información en nuestro disco duro en lugar de que queden en la memoria RAM. Esto dependerá del tipo de estructura de dato del objeto en cuestión.

Si nuestros datos se encuentran contenidos en una estructura de datos rectangular (**data frames**), podemos exportarlos con diferentes funciones. De manera análoga a **read.table()**, la función **write.table()** nos permite exportar matrices o data frames, como archivos de texto con distintas extensiones. También podemos exportar datos a archivos con extensión .csv con la función **write.csv()**.

En el caso de las listas, la manera más sencilla es guardarlas en archivos **RDS**. Este es un tipo de archivo nativo de **R** que puede almacenar cualquier objeto a un archivo en nuestro disco duro. Además, RDS comprime los datos que almacena, por lo que ocupa menos espacio en disco duro que otros tipos de archivos. Para exportar un objeto a un archivo RDS, usamos la función **saveRDS()**.



# Acceso y recopilación de datos con API en R

## ¿Qué es una API?

Una API (Application Programming Interface) es un intermediario entre un conjunto de datos (normalmente muy grandes) y el resto del mundo (¡como nosotros!)

Las APIs proporcionan una forma accesible de solicitar un conjunto de datos, lo que se conoce como hacer una "llamada" a la API. Una llamada se envía a la API abriendo una dirección web.

Como ejemplo, vamos a solicitar datos a la API de [COVID Act Now](#).

# Acceso y recopilación de datos con API en R

## Componentes de una URL

Esta petición solicitará datos COVID de series temporales para un solo condado de los Estados Unidos:

<https://api.covidactnow.org/v2/county/06037.timeseries.json?apiKey=xyxyxy>

Esta llamada a la API consta de varios elementos. Primero, la **URL base**: <https://api.covidactnow.org/v2/>

Esta parte de la URL será la misma para todas nuestras llamadas a esta API.

La parte del **county/** de la URL indica que sólo queremos los datos de COVID de un solo condado. Mirando la documentación de la API de **COVID Act Now**, puedo ver que **states** es una opción alternativa para esta parte de la URL.

**06037** es el identificador único de un solo condado. Si quiero obtener los mismos datos pero para un condado diferente, sólo tengo que cambiar este número.

**.timeseries** proporciona a la API más información sobre los datos que estoy solicitando, y **.json** indica a la API que formatee los datos como JSON (que convertiremos en un marco de datos).

Todo lo que sigue a **'apiKey='** es el token de autorización, que indica a los servidores de **COVID Act Now** que estoy autorizado a solicitar estos datos. 'xyxyxy' no es un token real.

Nota: se puede obtener un token en: <https://apidocs.covidactnow.org/>



FIN