Clase 2

Proyecto de trabajo

Paquetes

- Introducen nuevas funciones, bases de datos y objetos de R en general, complementarios a lo contenido en R Base.

Paquetes

- Podemos descargarlos e instalarlos con el siguiente comando:

  ```
  install.packages(“nombre_del_paquete”)
  ```

- Sólo es necesario instalarlo una vez por computadora
- Cada vez que abrimos una nueva sesión de R y queremos usar una función de determinado paquete, hay que llamarlo de la siguiente manera:

  ```
  library(nombre_del_paquete)
  ```

Directorio de trabajo

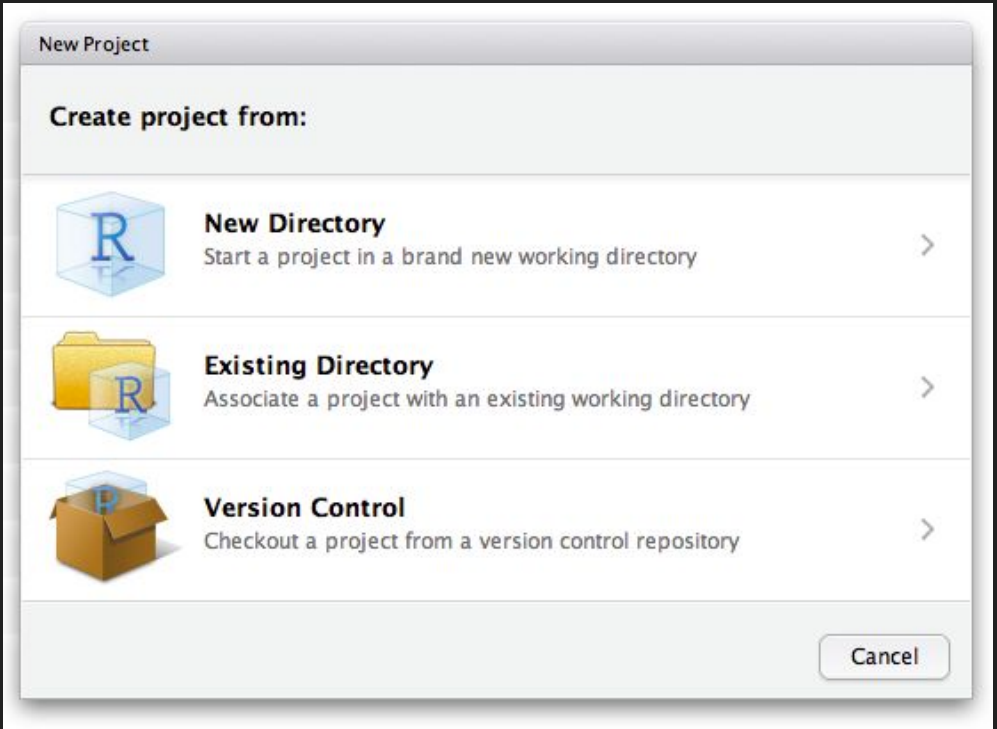
- El trabajo que hacemos en R “vive” en un directorio de trabajo.
- Allí es donde R busca los archivos que le pedimos que lea y colocará todos los archivos que le pidamos que guarde.
- Si queremos ver desde qué ruta estamos trabajando, hay que correr el comando:

```
getwd()
```

Directorio de trabajo - Proyectos

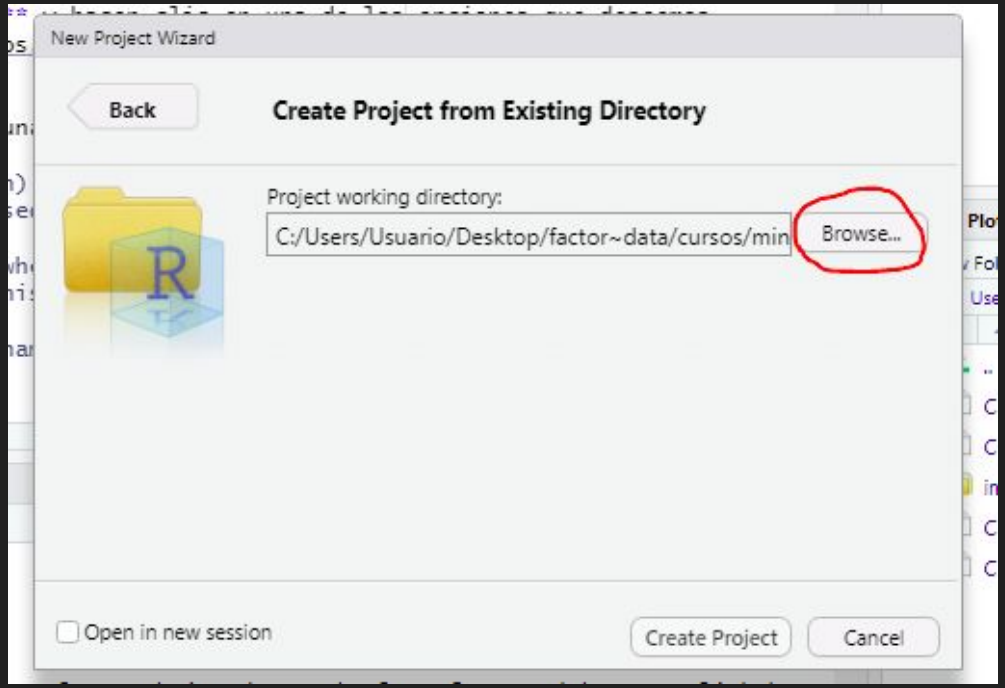
- Rstudio tiene una herramienta muy útil de trabajo, que son los proyectos. Un proyecto es un archivo que identifica toda una carpeta de trabajo.
- Para crearlo, vamos al logo de nuevo proyecto y elegimos la carpeta de trabajo.

Directorio de trabajo - Proyectos



- Nos interesan las primeras dos opciones, que:
  - crean el proyecto en un directorio nuevo
  - asocian el proyecto a un directorio existente

Directorio de trabajo - Proyectos



- Busco en “Browse” la ruta donde quiero almacenar el proyecto.

Directorio de trabajo - Proyectos

- Esto crea un archivo .RProj en la carpeta que elegimos. Este archivo sirve para abrir el proyecto en R Studio haciendo directamente clic en él.
- Crea un directorio escondido (que se llama .Rroj.user) donde se almacenan los archivos temporales específicos del proyecto (por ejemplo, puedo cerrar la sesión de R y mantener ahí los objetos del Enviroment de manera que vuelvan a aparecer cuando abra de nuevo el programa).
- Carga el proyecto en RStudio y muestra su nombre en la barra de opciones de R Studio.

Directorio de trabajo - Rutas

- De esta manera, se simplifica el proceso para leer y exportar archivos.
- Si quiero leer el archivo “base_de_datos.csv” que está en el directorio principal del proyecto, puedo hacer:
  - `read.csv(“base_de_datos.csv”)`
- Si el archivo está en el subdirectorio “data”, lo hago de la siguiente forma:
  - `read.csv(“data/base_de_datos.csv”)`

Organización del proyecto

- Algunas buenas prácticas a la hora de organizar la carpeta de trabajo son:
  - Utilizar minúsculas para nombrar nuestros archivos

    ✅ procesamiento.r

    ❌ Procesamiento.r

    ✅ analisis.r

Organización del proyecto

- Algunas buenas prácticas a la hora de organizar la carpeta de trabajo son:
  - Evitar caracteres especiales y espacios en blanco

    ❌ base procesada.r

    ❌ base_procesada_año_2021.r

    ✅ base_procesada_2021.r

Organización del proyecto

- Algunas buenas prácticas a la hora de organizar la carpeta de trabajo son:
  - Si tengo muchos archivos en una carpeta: utilizar números para ordenar y palabras clave para documentos similares

    ✅ 01_procesamiento-bases.r

    ✅ 02_analisis.r

    ✅ 03_graficos.r

Lectura y escritura de datos

- Las bases de datos pueden presentarse en archivos de distinto formato. En R contamos con múltiples funciones para importar archivos según los distintos formatos que tienen:

Lectura y escritura de datos

Tipo de archivo	Extensión	Paquete	Función
Texto plano	.csv	`readr`	`read_csv()`
Texto plano	.txt	`readr`	`read_txt()`
Texto plano	.tsv	`readr`	`read_tsv()`
Extensión de R	.RData	`RBase`	`open()`
Extensión de R	.RDS	`RBase`	`readRDS()`
Excel	.xlsx	`openxlsx`	`read.xlsx()`
Excel	.xl	`readxl`	`read_excel()`
Otro software	.dta	`haven`	`read_dta()`
Otro software	.sav	`haven`	`read_spss()`

Lectura y escritura de datos

- Un parámetro común en todas esas funciones para la lectura de datos es `file`. Allí debemos especificar la ruta hasta el archivo, incluyendo la extensión del mismo.

  ```
  df <- read.csv("../data/snic-pais-series.csv")
  ```

- Si estamos trabajando desde un proyecto, el punto de partida para la ruta a especificar es la carpeta del proyecto. Si queremos ir hacia atrás en las carpetas, agregamos `../`

Lectura y escritura de datos

- Cada una de las funciones que vimos para la importación de datos tiene su contracara en una función para exportar datos.
- Dependiendo el formato en que queramos exportar, podemos usar:
    - write_csv() para exportar .csv
    - write.xlsx() para exportar un .xlsx
    - saveRDS() para exportar un archivo .RDS

Lectura y escritura de datos

- Estos archivos tienen dos parámetros importantes:
    - el objeto donde está almacenado el dataframe que exportaremos
    - la ruta donde lo vamos a exportar

```
write_csv(df_juguete, '../data/df_ejemplo.csv')

saveRDS(df_juguete, '../data/df_ejemplo.RDS')
```

Vamos al RMD…