

Ejercicios Bases de Datos Relacionales

[Descargar estos ejercicios](#)

Índice

- [Ejercicio 1](#)
- [Ejercicio 2](#)

Ejercicio 1

A partir del [ejercicio 2 de anónimos y extensores](#) (si no lo hiciste en su día, descárgalo de Aules) completa el último punto con:

- Define la clase estática `BibliotecaJson` con un **método extensor de la clase `Biblioteca`** `Guarda` que reciba un `string` con el nombre del fichero y guarde la biblioteca en formato JSON. Deberás **añadir las anotaciones necesarias** a las propiedades de los `record` para que se genere correctamente el fichero `biblioteca.json` con el siguiente formato:


```
{
  "nombre": "EL RINCÓN DE LEER",
  "libros": [
    {
      "titulo": "Don Quijote de la Mancha",
      "autor": "Miguel de Cervantes",
      "editorial": "Editorial EDAF, S.A",
      "numPaginas": 765,
      "isbn": "9788441405298",
      "reseña": "El libro, sinopsis..."
    },
    ...
  ]
}
```

Crea en el programa principal las llamadas necesarias para probar este código.

Ejercicio 2

Crear una aplicación de consola en C# que permita gestionar eventos académicos (ponencias), incluyendo su almacenamiento en archivos JSON, recuperación de datos, y una visualización clara y estilizada mediante tablas en consola usando la biblioteca

Spectre.Console.

 **Aviso:** Deberemos seguir las especificaciones que se indican en el tema, para codificar de forma correcta la serialización con JSON, como utilizar clases `record` para representar los datos, y aplicar buenas prácticas de serialización con `System.Text.Json`.

Para ello crearemos los siguientes **record**:

- **DesglosePonencia**, formado por: **Momento** de tipo string que aportará información de en que momento se ha tratado un tema **Tema** tratado en ese momento (string) y **Resumen** del tema tratado (string).
- **Ponencia** que contenga: **Ponente** (string), **Temática** (string), **Fecha** (DateTime), **Horainicio** y **HoraFin** (float) y una lista de desgloses **DesglosePonencia**.

Además tendremos una clase de **utilidad** con dos métodos, uno para guardar una ponencia en un fichero (la ponencia y el fichero se pasarán como argumentos) y otro para recuperar la ponencia del fichero pasado como argumento

En la clase **Program** tendremos los métodos necesarios que nos permitan pedir los datos completos de una ponencia, teniendo en cuenta que una ponencia puede tener varios desgloses (hasta que el usuario presione ESC). Cuando se recoja toda la información, se pasará a Guardar la ponencia en un fichero localizado en la carpeta **Ponencias** (Si no existe se crea) el nombre del fichero tendrá el siguiente formato.

```
Pedro_Pietro_310320251503.json
```

Genera un nombre de archivo válido y lo guarda en la carpeta Ponencias.

Otro de los métodos de utilidad es el de recuperación de archivos, que mostrará todos los archivos JSON existentes en la carpeta Ponencias (Se explica más abajo).

Al ejecutar el programa, se mostrará un menú como este:

```
[1] Añadir ponencia
[2] Recuperar ponencia
[3] Salir
```

Si el usuario elige [1], se le pedirá introducir los datos, se guardará el archivo y confirmará el guardado.

Si elige [2], se mostrarán los archivos disponibles en la carpeta Ponencias (precedidos por un número)

```
Ponencias disponibles:
[1] Pedro_Pietro_31032025103.json
Selecciona la ponencia a recuperar (número):
```

al seleccionar uno de los números, se mostrará la ponencia con sus desgloses en una tabla como la siguiente, deberás usar [Spectre.Console](#) (puedes instalarla con: `dotnet add package Spectre.Console`):

Ponente	Pedro Pietro		
Temática	Aws		
Fecha	31/03/2025		
Hora	10:3 - 11:3		
Información desglosada	Momento	Tema Tratado	Resumen
	10,50	Automatizar despliegue	Como automatizar el despliegue en la nube con un solo click
	11,10	ultimas pruebas	Comprobación de que el despliegue ha sido correcto
	11,30	Fin de la conferencia	Con un resumen de los veneficios de DevOp



Se deberá controlar que las entradas por consola son válidas, sobre todo a la hora de seleccionar el archivo a mostrar. Además de controlar las posibles Excepciones que puedan ocurrir.