

## Trabajo integrador nº 2

### Seminario de Lenguajes opción .NET 2022

Se desea desarrollar una aplicación para una Institución educativa que se dedica a dictar cursos. Se debe gestionar la siguiente información respecto de los cursos, estudiantes e inscripciones.

- **Curso:** título, descripción, fecha de inicio y fecha de finalización
- **Estudiante:** DNI, nombre, apellido y email
- **Inscripción:** estudiante, curso y fecha de inscripción

Se debe proveer la siguiente funcionalidad:

- Altas, bajas, modificaciones y consultas de los cursos
- Altas, bajas, modificaciones y consultas de los estudiantes
- Altas, bajas, modificaciones y consultas de las inscripciones

Para realizar las altas de las inscripciones, debe poder elegirse un estudiante y un curso previamente dados de alta.

Además debe ser posible obtener los siguientes listados:

- Listado de todos los estudiantes que están realizando algún curso actualmente, que aún no haya finalizado. Visualizar el nombre y apellido del estudiante junto con el título del curso.
- Listado de todos los estudiantes que han finalizado algún curso. Visualizar el nombre y apellido del estudiante junto con el título del curso.
- Dado un curso (debe poder elegirse de alguna manera), listar todos los alumnos inscriptos en ese curso
- Dado un estudiante (debe poder elegirse de alguna manera), listar todos los cursos en los que se ha inscripto.

Para desarrollar esta aplicación se debe utilizar una arquitectura limpia del estilo presentado en la teoría 13 (3 proyectos: Aplicación, Repositorios y UI). Definir los repositorios y casos de uso que considere convenientes. En el proyecto Repositorios, utilizar Entity Framework Core para persistir la información en una base de datos SQLite. Utilizar *code first*.

La interfaz de usuario debe realizarse utilizando Blazor Server. Diseñar esta interfaz libremente.

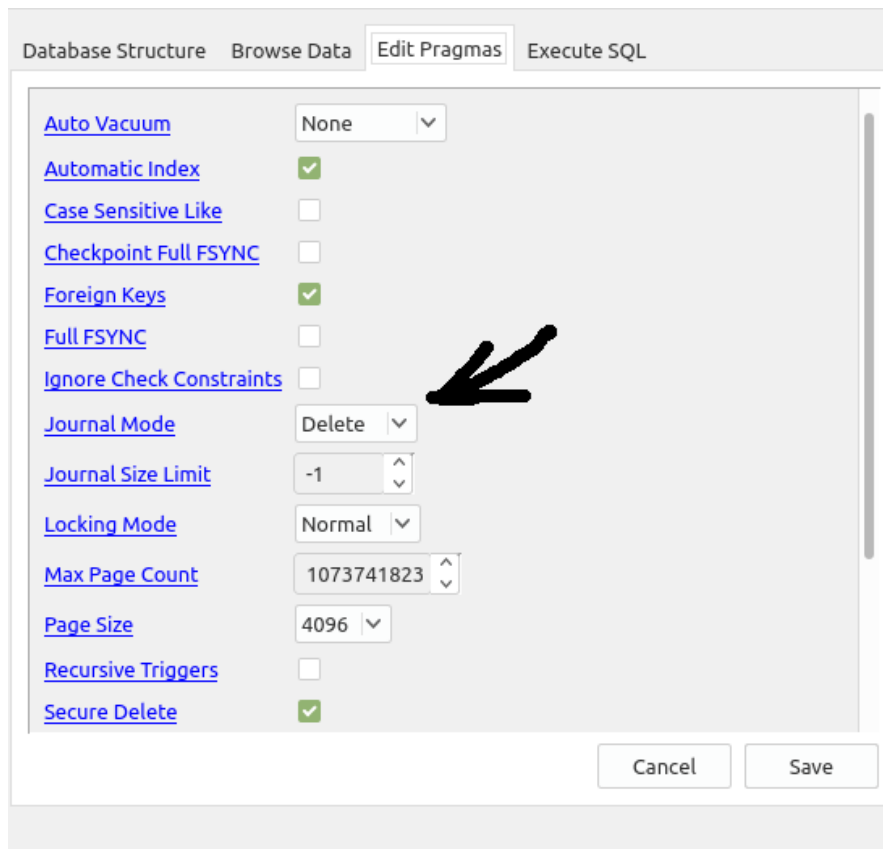
Nota: Es conveniente establecer la propiedad *journal mode* de la base de datos sqlite en **DELETE**

Se puede hacer con la aplicación DB Browser for SQLite

Database Structure   Browse Data   **Edit Pragmas**   Execute SQL

<a href="#">Auto Vacuum</a>	None ▾
<a href="#">Automatic Index</a>	<input checked="" type="checkbox"/>
<a href="#">Case Sensitive Like</a>	<input type="checkbox"/>
<a href="#">Checkpoint Full FSYNC</a>	<input type="checkbox"/>
<a href="#">Foreign Keys</a>	<input checked="" type="checkbox"/>
<a href="#">Full FSYNC</a>	<input type="checkbox"/>
<a href="#">Ignore Check Constraints</a>	<input type="checkbox"/>
<a href="#">Journal Mode</a>	Delete ▾
<a href="#">Journal Size Limit</a>	-1 ▴ ▾
<a href="#">Locking Mode</a>	Normal ▾
<a href="#">Max Page Count</a>	1073741823 ▴ ▾
<a href="#">Page Size</a>	4096 ▾
<a href="#">Recursive Triggers</a>	<input type="checkbox"/>
<a href="#">Secure Delete</a>	<input checked="" type="checkbox"/>

Cancel   Save



También se puede hacer por código:

```
context.Database.EnsureCreated();
var connection = context.Database.GetDbConnection();
connection.Open();
using (var command = connection.CreateCommand())
{
    command.CommandText = "PRAGMA journal_mode=DELETE;";
    command.ExecuteNonQuery();
}
```