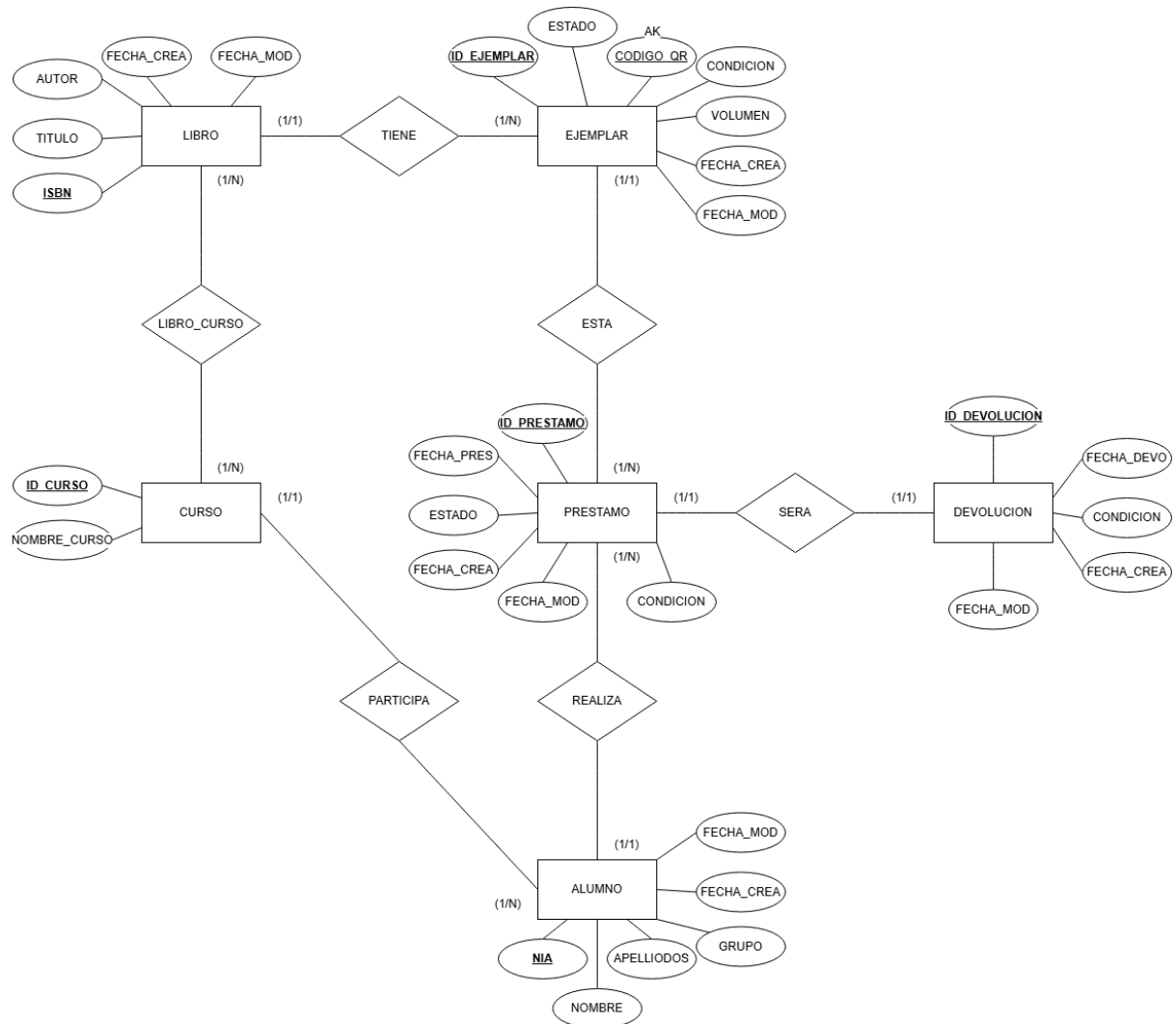


ENTIDAD RELACION

CURSO (id_curso PK, nombre_curso) 	LIBRO_CURSO (ISBN FK, id_curso FK, PK (ISBN, id_curso))
LIBRO (ISBN PK, titulo, autor, fecha_creacion, fecha_modificacion) 	PRESTAMO (id_prestamo PK, id_ejemplar FK, NIA FK, fecha_prestamo, estado, fecha_creacion, fecha_modificacion)
EJEMPLAR (id_ejemplar PK, ISBN FK, codigo_qr UNIQUE, volumen, estado, fecha_creacion, fecha_modificacion) 	DEVOLUCION (id_devolucion PK, id_prestamo FK UNIQUE, fecha_devolucion, estado, fecha_creacion, fecha_modificacion)
ALUMNO (NIA PK, nombre, apellidos, id_curso FK, grupo, fecha_creacion, fecha_modificacion) 	



Paso a Tablas y Entidades Finales

- **LIBRO** (ISBN PK, título, autor, fecha_creacion, fecha_modificacion)

PK = ISBN

- **EJEMPLAR** (id_ejemplar PK, ISBN FK, codigo_qr UNIQUE, condicion, estado, volumen, fecha_creacion, fecha_modificacion)

PK = id_ejemplar

FK = ISBN -> LIBRO(ISBN)

- **ALUMNO** (NIA PK, nombre, apellidos, id_curso FK, grupo, fecha_creacion, fecha_modificacion)

PK = NIA

FK = id_curso -> CURSO(id_curso)

- **CURSO** (id_curso PK, nombre_curso)

PK = id_curso

- **LIBRO_CURSO** (ISBN PK/FK, id_curso PK/FK)

PK = (ISBN, id_curso)

FK = ISBN -> LIBRO(ISBN)

FK = id_curso -> CURSO(id_curso)

- **PRESTAMO** (id_prestamo PK, id_ejemplar FK, NIA FK, fecha_prestamo, estado, condicion, fecha_creacion, fecha_modificacion)

PK = id_prestamo

FK = id_ejemplar -> EJEMPLAR(id_ejemplar)

FK = NIA -> ALUMNO(NIA)

- **DEVOLUCION** (id_devolucion PK, id_prestamo FK, fecha_devolucion, condicion, fecha_creacion, fecha_modificacion)

PK = id_devolucion

FK = id_prestamo -> PRESTAMO(id_prestamo)

Relaciones finales:

- Libro 1:N Ejemplar
- Ejemplar 1:N Prestamo
- Prestamo 1:1 Devolución
- Alumno 1:N Prestamo
- Alumno N:1 Curso
- Libro N:M Curso (libros permitidos por curso)

3. DDL – Script SQL MySQL (Versión Corregida v3)

```
CREATE DATABASE IF NOT EXISTS banco_libros;
```

```
USE banco_libros;
```

```
CREATE TABLE curso (
```

```
    id_curso INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nombre_curso VARCHAR(50) NOT NULL
```

```
);
```

```
CREATE TABLE libro (
```

```
    ISBN VARCHAR(13) PRIMARY KEY,
```

```
    titulo VARCHAR(255) NOT NULL,
```

```
    autor VARCHAR(255) NOT NULL,
```

```
    fecha_creacion DATETIME DEFAULT CURRENT_TIMESTAMP,
```

```
    fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP
```

```
    ON UPDATE CURRENT_TIMESTAMP
```

```
);
```

```
CREATE TABLE ejemplar (
```

```
    id_ejemplar INT AUTO_INCREMENT PRIMARY KEY,
```

```
    ISBN VARCHAR(13) NOT NULL,
```

```

codigo_qr VARCHAR(50) NOT NULL UNIQUE,
volumen VARCHAR(20) NOT NULL,
estado ENUM('disponible', 'prestado') DEFAULT 'disponible',
condicion ENUM('nuevo', 'usado', 'deteriorado') DEFAULT 'nuevo',
fecha_creacion DATETIME DEFAULT CURRENT_TIMESTAMP,
fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP
    ON UPDATE CURRENT_TIMESTAMP,
CONSTRAINT fk_ejemplar_libro
    FOREIGN KEY (ISBN)
    REFERENCES libro(ISBN)
    ON UPDATE CASCADE
    ON DELETE RESTRICT
);

CREATE TABLE alumno (
    NIA INT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(150) NOT NULL,
    id_curso INT NOT NULL,
    grupo VARCHAR(10),
    fecha_creacion DATETIME DEFAULT CURRENT_TIMESTAMP,
    fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP
    ON UPDATE CURRENT_TIMESTAMP,
CONSTRAINT fk_alumno_curso
    FOREIGN KEY (id_curso)
    REFERENCES curso(id_curso)
    ON UPDATE CASCADE
    ON DELETE RESTRICT
);

```

```
CREATE TABLE libro_curso (  
    ISBN VARCHAR(13) NOT NULL,  
    id_curso INT NOT NULL,  
    PRIMARY KEY (ISBN, id_curso),  
    CONSTRAINT fk_librocurso_libro  
        FOREIGN KEY (ISBN)  
        REFERENCES libro(ISBN)  
        ON UPDATE CASCADE  
        ON DELETE CASCADE,  
    CONSTRAINT fk_librocurso_curso  
        FOREIGN KEY (id_curso)  
        REFERENCES curso(id_curso)  
        ON UPDATE CASCADE  
        ON DELETE CASCADE  
);
```

```
CREATE TABLE prestamo (  
    id_prestamo INT AUTO_INCREMENT PRIMARY KEY,  
    id_ejemplar INT NOT NULL,  
    NIA INT NOT NULL,  
    fecha_prestamo DATETIME DEFAULT CURRENT_TIMESTAMP,  
    estado ENUM('activo', 'finalizado') DEFAULT 'activo',  
    condicion ENUM('nuevo', 'usado', 'deteriorado') DEFAULT 'nuevo',  
    fecha_creacion DATETIME DEFAULT CURRENT_TIMESTAMP,  
    fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP  
        ON UPDATE CURRENT_TIMESTAMP,  
    CONSTRAINT fk_prestamo_ejemplar  
        FOREIGN KEY (id_ejemplar)  
        REFERENCES ejemplar(id_ejemplar)
```

```

ON UPDATE CASCADE
ON DELETE RESTRICT,
CONSTRAINT fk_prestamo_alumno
FOREIGN KEY (NIA)
REFERENCES alumno(NIA)
ON UPDATE CASCADE
ON DELETE RESTRICT
);

```

```

CREATE TABLE devolucion (
    id_devolucion INT AUTO_INCREMENT PRIMARY KEY,
    id_prestamo INT NOT NULL UNIQUE,
    fecha_devolucion DATETIME DEFAULT CURRENT_TIMESTAMP,
    condicion ENUM('correcto', 'deteriorado') DEFAULT 'correcto',
    fecha_creacion DATETIME DEFAULT CURRENT_TIMESTAMP,
    fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP
    ON UPDATE CURRENT_TIMESTAMP,
    CONSTRAINT fk_devolucion_prestamo
    FOREIGN KEY (id_prestamo)
    REFERENCES prestamo(id_prestamo)
    ON UPDATE CASCADE
    ON DELETE RESTRICT
);

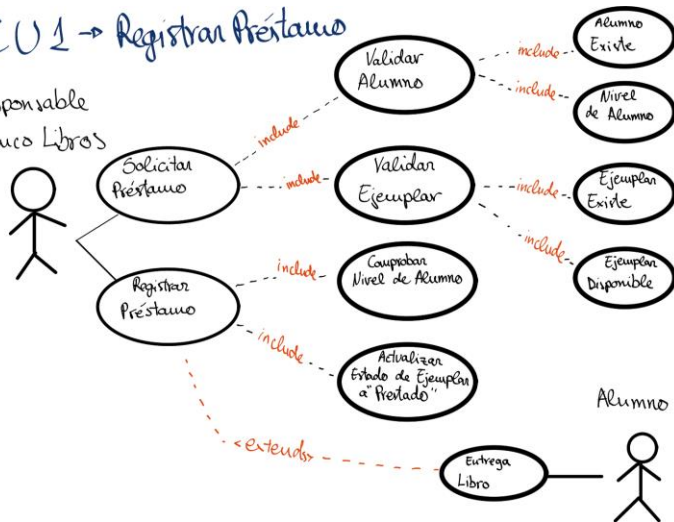
```

RELACIONES	
Libro — Ejemplar	1 : N
Ejemplar — Prestamo	1 : N
Alumno — Prestamo	1 : N
Curso — Alumno	1 : N

Libro — Curso	N : M (Libro_Curso)
Prestamo(fuerte) — Devolucion(debil)	1 : 1

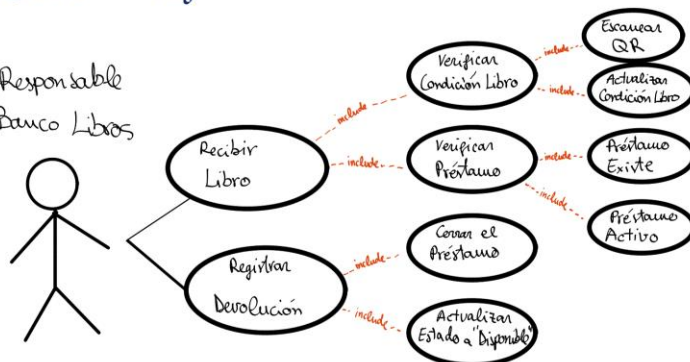
CU1 → Registrar Prestamo

Responsable
Banco Libros

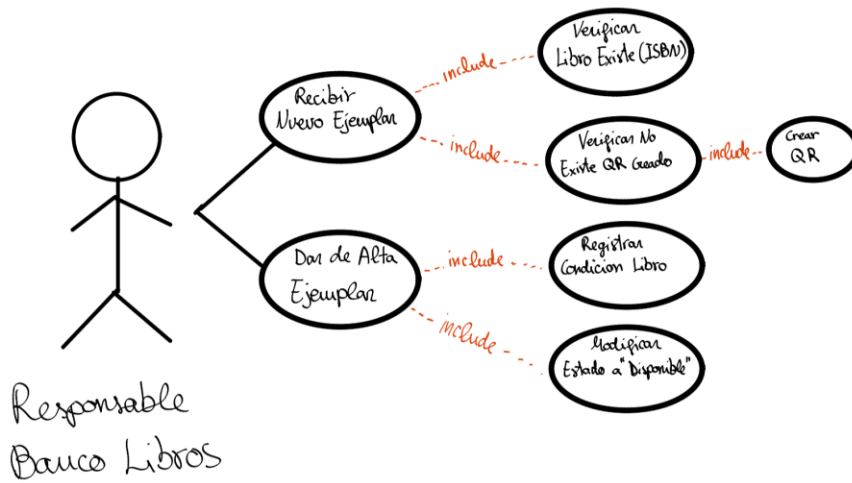


CU2 → Registrar Devolución

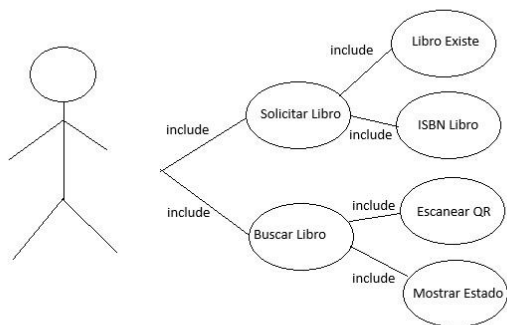
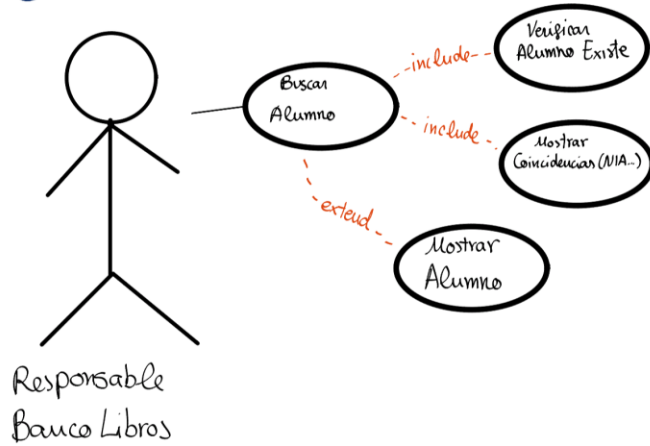
Responsable
Banco Libros

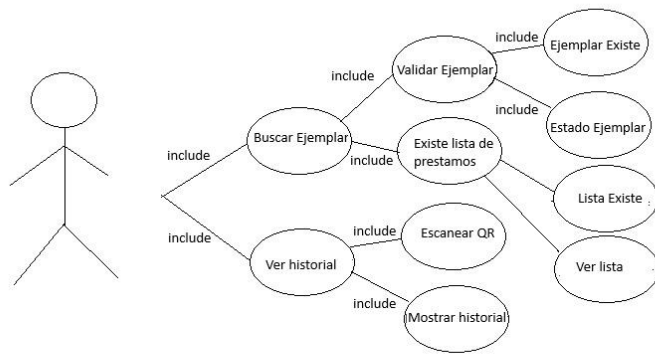


CU3 Dar Alta Ejemplar



CU4 Buscar Alumno





CASOS DE USO

1. Registrar préstamo
2. Registrar devolución
3. Dar de alta ejemplar
4. Buscar alumno
5. Buscar libro / ejemplar
6. Ver historial de ejemplar
7. Validaciones del sistema (*caso de uso incluido / extendido*)

CU-01 – Registrar préstamo

Actor: Responsable banco libros

Descripción:

Permite registrar el préstamo de un ejemplar a un alumno, validando que cumple las normas del programa Xarxa Llibres.

Precondiciones:

- El alumno existe
- El ejemplar existe
- El ejemplar está disponible
- El libro está autorizado para el curso del alumno

Flujo principal:

1. El responsable busca al alumno por NIA, nombre o curso.
2. El sistema muestra los datos del alumno.
3. El responsable escanea el código QR del ejemplar.
4. El sistema verifica:
 - a. Que el ejemplar no esté prestado.
 - b. Que el alumno no tenga ya ese libro.
 - c. Que el libro corresponde a su curso.
5. El sistema registra el préstamo.
6. El sistema cambia el estado del ejemplar a *prestado*.
7. El sistema muestra confirmación.

Flujos alternativos:

- El ejemplar ya está prestado → se cancela el préstamo.
- El libro no corresponde al curso → se deniega.
- El alumno ya tiene ese libro → se deniega.

Postcondiciones:

- Préstamo registrado como *activo*
- Ejemplar marcado como *prestado*

CU-02 – Registrar devolución

Actor: Responsable banco libros

Descripción:

Permite registrar la devolución de un ejemplar prestado y evaluar su estado físico.

Precondiciones:

- El préstamo existe
- El préstamo está activo

Flujo principal:

1. El responsable escanea el QR del ejemplar.
2. El sistema localiza el préstamo activo.
3. El responsable introduce la condición del libro.
4. El sistema registra la devolución.
5. El sistema cierra el préstamo.
6. El sistema actualiza el estado del ejemplar a *disponible*.

Flujos alternativos:

- El ejemplar no tiene préstamo activo → error.
- El ejemplar no pertenece al alumno → error.

Postcondiciones:

- Préstamo marcado como *finalizado*
- Ejemplar disponible

CU-03 – Dar de alta ejemplar

Actor: Responsable banco libros

Descripción:

Permite registrar un nuevo ejemplar físico en el sistema.

Precondiciones:

- El libro existe (ISBN)
- El código QR no existe previamente

Flujo principal:

1. El responsable introduce ISBN, volumen y QR.
2. El sistema valida el ISBN.
3. El sistema registra el ejemplar.
4. El ejemplar queda disponible.

Flujos alternativos:

- QR duplicado → error.
- ISBN inexistente → error.

Postcondiciones:

- Ejemplar registrado y disponible

CU-04 – Buscar alumno

Actor: Responsable banco libros

Descripción:

Permite localizar alumnos para gestión de préstamos.

Precondiciones: Ninguna

Flujo principal:

1. Introducir criterio (NIA, nombre, curso).
2. El sistema muestra coincidencias.

Postcondiciones:

- Alumno seleccionado o no encontrado

CU-05 – Buscar libro / ejemplar

Actor: Responsable banco libros

Descripción:

Permite localizar libros o ejemplares mediante ISBN o QR.

Flujo principal:

1. Introducir ISBN o escanear QR.
2. Mostrar datos del libro y estado.

CU-06 – Ver historial de ejemplar

Actor: Responsable banco libros

Descripción:

Permite consultar el historial completo de préstamos de un ejemplar.

Precondiciones:

- El ejemplar existe

Flujo principal:

1. Escanear QR.
2. Mostrar lista de préstamos por curso académico.
3. Mostrar condición de entrega y devolución.

Postcondiciones:

- Historial visualizado

CU-07 – Validaciones del sistema (incluido)

Este **NO se dibuja solo**, se usa como:

- <<include>> en:
 - Registrar préstamo
 - Registrar devolución

Incluye:

- Validar disponibilidad
- Validar curso
- Validar duplicidad