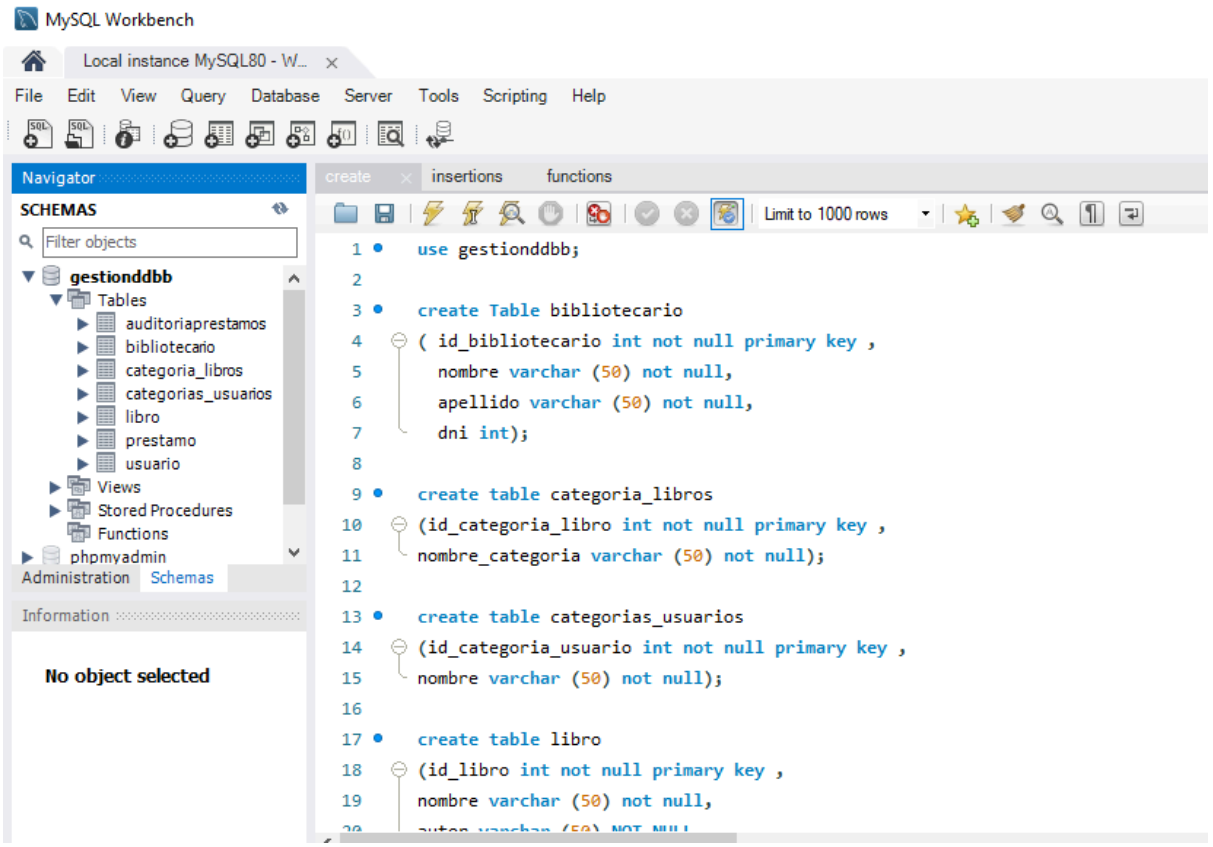
# Trabajo Práctico: Base Biblioteca




**MATERIA: Gestión de Base de Datos**

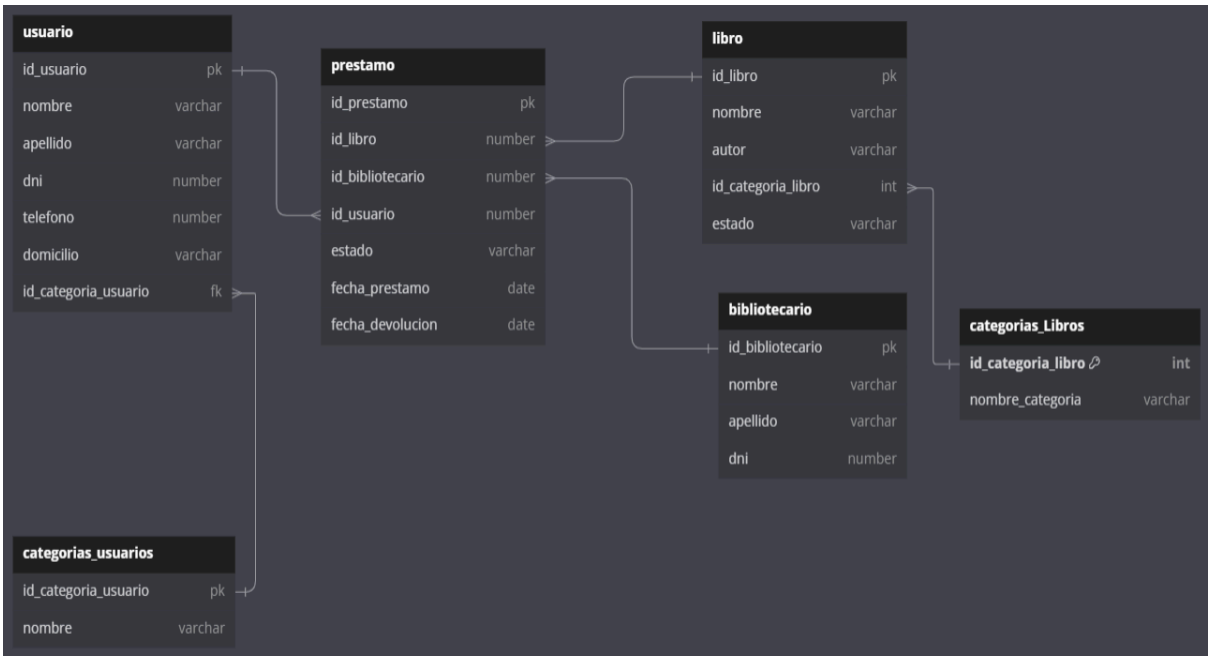
**DOCENTE: Salvatori, Lucas**

**ALUMNOS: Achaga Perez, Agustin**
**Cifarelli, Virginia.**

**JUNIO 2024**

## Introducción:

### 1. Gráfico digital del DER (Diagrama de Entidad Relación)




El diagrama de entidad relación está compuesto en un inicio por seis (6) tablas, siendo la tabla central la tabla “prestamos”, la cual obtiene los datos de tres tablas que son tabla de: “usuario”, “bibliotecario” y “libro”, a su vez, la tabla “libro” se relaciona con la entidad “categorias_libros”; y la entidad “usuario” se relaciona con la tabla “categorias_usuarios”.

Bibliotecario: esta entidad describe los datos de cada bibliotecario, está compuesta por:

- id_bibliotecario→ es nuestra clave primaria, no puede ser null;
- nombre → nombre de pila de la persona,
- apellido → apellido del bibliotecario,
- dni → dni de la persona.

Libro: esta entidad describe los datos de cada libro, está compuesta por:

- id_libro→ es nuestra clave primaria, no puede ser null;
- nombre → nombre del libro,
- autor → nombre y apellido del autor,
- id_categoria_libro → clave foránea,
- estado → define si el libro se encuentra prestado o disponible.

Categorias_libros: esta entidad describe el género de cada libro, está compuesta por:

- id_categoria_libros → es nuestra clave primaria, no puede ser null;
- nombre_categoria → categoría /género del libro.

Usuario: esta entidad describe los datos de cada usuario, está compuesta por:

- id_usuario→ es nuestra clave primaria, no puede ser null;
- nombre → nombre de pila de la persona,
- apellido → apellido del bibliotecario,
- dni → dni de la persona.
- telefono → num para poder comunicarse,
- domicilio → dirección de las personas que concurren a la biblioteca
- id_categoria_usuario → clave foránea, para ver qué tipo de usuario es la persona.

Categorias_usuarios: esta entidad describe los datos de cada categoría de usuarios, compuesta por dos campos:

- id_categoria_usuario → es nuestra clave primaria
- nombre → describe el tipo de usuario, socio, público en general.

Préstamo: esta entidad describe los datos de cada préstamo:

- id_prestamo → clave primaria, identificador de cada préstamo
- id_libro → clave foránea, para identificar el libro
- id_bibliotecario → clave foránea para identificar que bibliotecario lo registro,
- id_usuario → clave foránea, para identificar los datos del usuario,
- estado → "prestado",
- fecha_prestamo → fecha de registro del préstamo,
- fecha_devolucion → fecha de devolución del libro.

## 2. Crear las tablas en mysql (archivo create.sql)

```
create table bibliotecario
( id_bibliotecario int not null primary key ,
  nombre varchar (50) not null,
  apellido varchar (50) not null,
  dni int);

create table categoria_libros
(id_categoria_libro int not null primary key ,
nombre_categoria varchar (50) not null);

create table categorias_usuarios
(id_categoria_usuario int not null primary key ,
nombre varchar (50) not null);

create table libro
(id_libro int not null primary key ,
nombre varchar (50) not null,
autor varchar (50) NOT NULL,
id_categoria_libro INT NOT NULL,
```

```
estado VARCHAR(45) NOT NULL,
CONSTRAINT fk_categoria_libro FOREIGN KEY (id_categoria_libro) REFERENCES
categorias_libros(id_categoria_libro)
);

create table prestamo
(id_prestamo INT NOT NULL PRIMARY KEY,
id_libro INT NOT NULL,
id_bibliotecario INT NOT NULL,
id_usuario INT NOT NULL,
estado VARCHAR(45) NOT NULL,
fecha_prestamo DATE,
fecha_devolucion DATE,
CONSTRAINT fk_prestamo_libro FOREIGN KEY (id_libro) REFERENCES libro(id_libro),
CONSTRAINT fk_prestamo_bibliotecario FOREIGN KEY (id_bibliotecario) REFERENCES
bibliotecario(id_bibliotecario),
CONSTRAINT fk_prestamo_usuario FOREIGN KEY (id_usuario) REFERENCES
usuario(id_usuario)
);

create table usuario
(id_usuario int not null primary key,
nombre varchar (50) not null,
apellido varchar (50) not null,
dni int,
telefono int,
domicilio varchar (50),
id_categoria_usuario int not null,
CONSTRAINT fk_categorias_usuarios FOREIGN KEY (id_categoria_usuario)
REFERENCES categorias_usuarios(id_categoria_usuario));
```

## 3. Insercion datos a las tablas (archivo insertions.sql)

```
/*inserción datos bibliotecarios*/
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (1, 'Juan', 'González',
12345678);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (2, 'María', 'López',
87654321);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (3, 'Pedro', 'Martínez',
23456789);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (4, 'Ana', 'Rodríguez',
98765432);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (5, 'Carlos', 'Pérez',
34567890);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (6, 'Laura', 'Sánchez',
45678901);
```

```
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (7, 'José', 'García', 56789012);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (8, 'Sofía', 'Fernández', 67890123);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (9, 'Miguel', 'Ruiz', 78901234);
INSERT INTO bibliotecario (id_bibliotecario, nombre, apellido, dni) VALUES (10, 'Elena', 'Díaz', 89012345);

/*inserción datos categoria_libros*/
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (1, 'Novela');
INSERT INTO categoria_libros(id_categoria_libro, nombre_categoria) VALUES (2, 'Cuento');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (3, 'Clásico');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (4, 'Historia');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (5, 'Ciencia Ficción');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (6, 'Fantasía');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (7, 'Poesía');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (8, 'Teatro');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (9, 'Biografía');
INSERT INTO categoria_libros (id_categoria_libro, nombre_categoria) VALUES (10, 'Ensayo');

/*inserción datos categorias_usuarios*/
INSERT INTO categorias_usuarios (id_categoria_usuario, nombre) VALUES (1, 'Estudiante');
INSERT INTO categorias_usuarios (id_categoria_usuario, nombre) VALUES (2, 'Socio');
INSERT INTO categorias_usuarios (id_categoria usuario, nombre) VALUES (3, 'Público en General');

/*inserción datos libros*/
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (1, 'Cien años de soledad', 'Gabriel García Márquez', 1, 'Disponible');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (2, 'El principito', 'Antoine de Saint-Exupéry', 2, 'Prestado');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (3, 'Harry Potter y la piedra filosofal', 'J.K. Rowling', 1, 'Disponible');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (4, 'Don Quijote de la Mancha', 'Miguel de Cervantes', 3, 'Disponible');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (5, '1984', 'George Orwell', 2, 'Prestado');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (6, 'Orgullo y prejuicio', 'Jane Austen', 3, 'Disponible');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (7, 'Rayuela', 'Julio Cortázar', 1, 'Prestado');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (8, 'La sombra del viento', 'Carlos Ruiz Zafón', 1, 'Disponible');
INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (9, 'Crónica de una muerte anunciada', 'Gabriel García Márquez', 1, 'Disponible');
```

INSERT INTO libro (id_libro, nombre, autor, id_categoria_libro, estado) VALUES (10, 'La casa de los espíritus', 'Isabel Allende', 3, 'Prestado');

/*inserción datos prestamos*/
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (1, 2, 3, 1, 'Prestado', '2024-01-01', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (2, 5, 6, 2, 'Prestado', '2024-01-02', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (3, 7, 1, 3, 'Prestado', '2024-01-03', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (4, 10, 4, 4, 'Prestado', '2024-01-04', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (5, 3, 8, 5, 'Prestado', '2024-01-05', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (6, 1, 10, 6, 'Prestado', '2024-01-06', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (7, 4, 2, 7, 'Prestado', '2024-01-07', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (8, 8, 9, 8, 'Prestado', '2024-01-08', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (9, 6, 5, 9, 'Prestado', '2024-01-09', null);
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo, fecha_devolucion) VALUES (10, 9, 7, 10, 'Prestado', '2024-01-10', null);

/*inserción datos usuarios*/
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (1, 'Luis', 'Pérez', 11111111, 1122334455, 'Av. Libertador 123', 2);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (2, 'Ana', 'Gómez', 22222222, 9988776655, 'Calle 10 de Octubre 456', 1);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (3, 'Jorge', 'Fernández', 33333333, 3344556677, 'Av. San Martín 789', 2);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (4, 'María', 'López', 44444444, 7766554433, 'Calle Mayor 567', 3);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (5, 'Carolina', 'García', 55555555, 5566778899, 'Av. Rivadavia 890', 1);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (6, 'Pablo', 'Martínez', 66666666, 3344667788, 'Calle Primera 234', 2);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (7, 'Lucía', 'González', 77777777, 9900112233, 'Av. Bolívar 901', 3);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (8, 'Diego', 'Díaz', 88888888, 1122334455, 'Calle Belgrano 345', 1);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (9, 'Sofía', 'Ruiz', 99999999, 6677889900, 'Av. Independencia 678', 2);
INSERT INTO usuario (id_usuario, nombre, apellido, dni, telefono, domicilio, id_categoria_usuario) VALUES (10, 'Mateo', 'Sánchez', 10101010, 3344556677, 'Calle Sarmiento 1234', 3);

## 4. Funciones.sql

## Para crear el Procedimiento Almacenado

El procedimiento registrar_prestamo tiene como objetivo registrar un nuevo préstamo de un libro en la biblioteca, asegurándose de que el libro no esté ya prestado. Si el libro ya está prestado, el procedimiento arroja un error.

**Parámetros de Entrada:**

- p_id_libro INT: ID del libro que se quiere prestar.
- p_id_usuario INT: ID del usuario que solicita el préstamo.
- p_fecha_prestamo DATE: Fecha en la que se realiza el préstamo.
- p_bibliotecario INT: ID del bibliotecario que gestiona el préstamo.

**Declaraciones de Variables:**

- v_estado VARCHAR(45): Variable para almacenar el estado del libro.
- v_id_prestamo INT: Variable para almacenar el nuevo ID del préstamo.

**Procedimiento:**

**Verificación del Estado del Libro:**

```
SELECT estado INTO v_estado
FROM prestamo
WHERE id_libro = p_id_libro
  AND estado = 'Prestado';
```

Esta sección verifica si el libro especificado (p_id_libro) ya está prestado. Si el estado del libro es 'Prestado', se almacena en la variable v_estado.

**Condición para el Préstamo:**

```
IF v_estado IS NOT NULL THEN
   SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'El libro ya está prestado.';
ELSE
   SELECT COALESCE(MAX(id_prestamo), 0) + 1 INTO v_id_prestamo FROM prestamo;
   INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado,
fecha_prestamo)
   VALUES (v_id_prestamo, p_id_libro, 1, p_id_usuario, 'Prestado', p_fecha_prestamo);
END IF;
```

**Condición de Error:** Si v_estado no es NULL, significa que el libro ya está prestado. En este caso, se lanza un error con el mensaje 'El libro ya está prestado.'.

```
SIGNAL SQLSTATE '45000'
```

SET MESSAGE_TEXT = 'El libro ya está prestado.';

**Inserción del Préstamo:** Si el libro no está prestado (v_estado es NULL), se procede a registrar el nuevo préstamo.

Se genera un nuevo id_prestamo automáticamente utilizando el mayor id_prestamo existente incrementado en uno.

SELECT COALESCE(MAX(id_prestamo), 0) + 1 INTO v_id_prestamo FROM prestamo;

**Se inserta el nuevo registro en la tabla prestamo con los datos proporcionados**.

INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado, fecha_prestamo)

VALUES (v_id_prestamo, p_id_libro, 1, p_id_usuario, 'Prestado', p_fecha_prestamo);

**Explicación Procedure/procedimiento almacenado:**

1. Declarar Variables:
   - v_estado: Variable para almacenar el estado del libro.
2. Verificar Disponibilidad:
   - Consulta en la tabla prestamo si el libro ya está prestado.
   - Si encuentra un registro con estado = 'prestado', el libro no está disponible.
3. Condicional:
   - Si el libro está prestado, lanza un error.
   - Si el libro está disponible, inserta un nuevo registro en la tabla prestamo con los parámetros proporcionados y marca el estado como prestado.

Este procedimiento asegura que no se puedan registrar préstamos de libros que ya están prestados.

## Para crear Vista: vista_prestamos_actuales

La vista: `vista_prestamos_actuales` muestra una lista de todos los libros que están actualmente prestados, junto con el nombre del usuario que los tiene y la fecha en que se prestaron.

A continuación, redactamos como deberia ingresar el usuario los parámetros para poder ver lo mencionado con anterioridad:

```
CREATE VIEW vista_prestamos_actuales AS
SELECT
   l.nombre AS nombre_libro,
   u.nombre AS nombre_usuario,
   p.fecha_prestamo
FROM
   prestamo p
   JOIN libro l ON p.id_libro = l.id_libro
   JOIN usuario u ON p.id_usuario = u.id_usuario
```

```
WHERE
  p.estado = 'prestado';
```

## Para crear Disparador: actualizar_fecha_devolucion

Este trigger se ejecutará antes de actualizar la tabla prestamo y registrará la fecha actual como fecha_devolucion cuando un préstamo se devuelve.

**Código del Trigger:**

```
DELIMITER $$
CREATE TRIGGER actualizar_fecha_devolucion
BEFORE UPDATE ON prestamo
FOR EACH ROW
BEGIN
  IF NEW.estado = 'Devuelto' AND OLD.estado <> 'Devuelto' THEN
    SET NEW.fecha_devolucion = CURDATE();
  END IF;
END$$
DELIMITER ;
```

**Explicación del Código:**

- **DELIMITER $$:** Cambia el delimitador de comandos para que MySQL interprete correctamente el cuerpo del trigger.
- **CREATE TRIGGER actualizar_fecha_devolucion:** Define el nombre del trigger**.**
- **BEFORE UPDATE ON prestamo:** Indica que el trigger se ejecutará antes de cualquier operación de actualización en la tabla prestamo.
- **FOR EACH ROW:** Especifica que el trigger se ejecutará una vez por cada fila que se actualice.
- **IF NEW.estado = 'Devuelto' AND OLD.estado <> 'Devuelto' THEN:** Comprueba si el nuevo estado del préstamo es 'Devuelto' y si el estado anterior no era 'Devuelto'. Esto asegura que solo se registre la fecha de devolución cuando se marca explícitamente un préstamo como devuelto.
- **SET NEW.fecha_devolucion = CURDATE();**: Establece la fecha de devolución a la fecha actual (CURDATE() devuelve la fecha actual).
- **END IF;:** Finaliza la condición IF.
- **END$$:** Finaliza el cuerpo del trigger.
- **DELIMITER ;:** Restaura el delimitador predeterminado.
- Se almacenan en el esquema de la base de datos y pueden ser visualizados utilizando la tabla information_schema.TRIGGERS.

**Ejemplo de Uso:**

**Imaginemos que tienes un préstamo en la tabla prestamo con el siguiente estado inicial:**

```
INSERT INTO prestamo (id_prestamo, id_libro, id_bibliotecario, id_usuario, estado,
fecha_prestamo, fecha_devolucion)

VALUES (1, 11, 1, 2, 'Prestado', '2024-06-01', NULL);
```

**Si deseas actualizar el estado del préstamo a 'Devuelto', la actualización se vería así:**

```
UPDATE prestamo
SET estado = 'Devuelto'
WHERE id_prestamo = 1;
```

**Después de esta actualización, el trigger actualizar_fecha_devolucion se ejecutará y establecerá fecha_devolucion a la fecha actual.**

**Resultados:**

**Antes de la actualización:**

(sql) SELECT * FROM prestamo WHERE id_prestamo = 1;

(textoplano)

```
+------------+----------+----------------+-----------+---------+--------------+----------------+
| id_prestamo | id_libro | id_bibliotecario | id_usuario | estado  | fecha_prestamo | fecha_devolucion |
+------------+----------+----------------+-----------+---------+--------------+----------------+
| 1          | 11       | 1              | 2         | Prestado | 2024-06-01   | NULL           |
+------------+----------+----------------+-----------+---------+--------------+----------------+
```

**Después de la actualización:**

(sql) SELECT * FROM prestamo WHERE id_prestamo = 1;

(textoplano)

```
+------------+----------+----------------+-----------+---------+--------------+----------------+
| id_prestamo | id_libro | id_bibliotecario | id_usuario | estado  | fecha_prestamo | fecha_devolucion |
+------------+----------+----------------+-----------+---------+--------------+----------------+
| 1          | 11       | 1              | 2         | Devuelto | 2024-06-01   | 2024-06-21     |
+------------+----------+----------------+-----------+---------+--------------+----------------+
```

Esto muestra cómo el trigger actualizar_fecha_devolucion actualiza automáticamente la fecha_devolucion cuando se cambia el estado del préstamo a 'Devuelto'.

## Para crear Transacción

La transacción registrará un nuevo préstamo, actualizará el estado del libro a "prestado" y registrará la operación en una tabla de auditoría llamada AuditoriaPrestamos.

**Primero, creamos la tabla de auditoría:**

```
CREATE TABLE AuditoriaPrestamos (
    id_auditoria INT AUTO_INCREMENT PRIMARY KEY,
    id_prestamo INT,
    accion VARCHAR(50),
    fecha DATETIME
);
```

**Luego, implementamos la transacción:**

```
DELIMITER //

CREATE PROCEDURE registrar_prestamo_con_transaccion(
    IN p_id_libro INT,
    IN p_id_usuario INT,
    IN p_fecha_prestamo DATE
)

BEGIN

    DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        ROLLBACK;
        RESIGNAL;
    END;
    START TRANSACTION;
    DECLARE libro_prestado INT;
```

**Verificar si el libro ya está prestado**

```
    SELECT COUNT(*) INTO libro_prestado
    FROM prestamo
    WHERE id_libro = p_id_libro AND estado = 'prestado';
    IF libro_prestado = 0 THEN
```

**Registrar el nuevo préstamo**

```
        INSERT INTO prestamo (id_libro, id_usuario, estado, fecha_prestamo)
        VALUES (p_id_libro, p_id_usuario, 'prestado', p_fecha_prestamo);
```

**Obtener el id del nuevo préstamo**

```
        DECLARE new_prestamo_id INT;
        SET new_prestamo_id = LAST_INSERT_ID();
```

**Actualizar el estado del libro a 'prestado'**

```
        UPDATE libro
        SET estado = 'prestado'
        WHERE id_libro = p_id_libro;
```

**Registrar la operación en la tabla de auditoría**

```
        INSERT INTO auditoriaprestamos (id_prestamo, accion, fecha)
        VALUES (new_prestamo_id, 'Registro de préstamo', NOW());
        COMMIT;
    ELSE
        ROLLBACK;
            SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'El libro ya está prestado.';
    END IF;
END //
DELIMITER ;
```