Desarrollo de Sistemas de Información

Segundo Examen Parcial Docente: Rildo M. Tapia Pacheco

🧩 Caso de estudio: Sistema de gestión de biblioteca

Implemente la documentación y la base de datos para el presente caso de estudio, incluye captura de pantallas para demostrar que logró ejecutar los scripts y otros componentes.

# 1️⃣ Recolección de requerimientos

Supongamos que el cliente necesita:

* Registrar libros, autores y editoriales.
* Controlar préstamos de libros a usuarios.
* Registrar devoluciones y calcular retrasos.
* Consultar estadísticas de uso.

# 2️⃣ Diseño conceptual (modelo entidad-relación)

## Entidades principales:

* Libro (id\_libro, título, año, id\_editorial)
* Autor (id\_autor, nombre)
* Editorial (id\_editorial, nombre)
* Usuario (id\_usuario, nombre, correo)
* Préstamo (id\_prestamo, id\_usuario, fecha\_prestamo, fecha\_devolución)
* DetallePréstamo (id\_prestamo, id\_libro)

## Relaciones:

* Un libro puede tener varios autores (relación N:M)
* Un préstamo puede incluir varios libros (relación N:M)
* Un usuario puede tener varios préstamos

# 3️⃣ Diseño lógico (tablas y claves)

sql

CREATE TABLE Editorial (

id\_editorial INT PRIMARY KEY AUTO\_INCREMENT, nombre VARCHAR(100) NOT NULL

);

CREATE TABLE Autor (

id\_autor INT PRIMARY KEY AUTO\_INCREMENT, nombre VARCHAR(100) NOT NULL

);

CREATE TABLE Libro (

id\_libro INT PRIMARY KEY AUTO\_INCREMENT, titulo VARCHAR(200) NOT NULL,

año INT, id\_editorial INT,

FOREIGN KEY (id\_editorial) REFERENCES Editorial(id\_editorial)

);

CREATE TABLE LibroAutor ( id\_libro INT,

id\_autor INT,

PRIMARY KEY (id\_libro, id\_autor),

FOREIGN KEY (id\_libro) REFERENCES Libro(id\_libro), FOREIGN KEY (id\_autor) REFERENCES Autor(id\_autor)

);

CREATE TABLE Usuario (

id\_usuario INT PRIMARY KEY AUTO\_INCREMENT,

nombre VARCHAR(100), correo VARCHAR(100)

);

CREATE TABLE Prestamo (

id\_prestamo INT PRIMARY KEY AUTO\_INCREMENT, id\_usuario INT,

fecha\_prestamo DATE, fecha\_devolucion DATE,

FOREIGN KEY (id\_usuario) REFERENCES Usuario(id\_usuario)

);

CREATE TABLE DetallePrestamo ( id\_prestamo INT,

id\_libro INT,

PRIMARY KEY (id\_prestamo, id\_libro),

FOREIGN KEY (id\_prestamo) REFERENCES Prestamo(id\_prestamo), FOREIGN KEY (id\_libro) REFERENCES Libro(id\_libro)

);

# 4️⃣ Diseño físico (índices, optimización)

sql

CREATE INDEX idx\_libro\_titulo ON Libro(titulo); CREATE INDEX idx\_usuario\_correo ON Usuario(correo);

# 5️⃣ Implementación de funciones

sql

DELIMITER //

CREATE FUNCTION calcular\_dias\_retraso(fecha\_devolucion DATE, fecha\_real DATE)

RETURNS INT DETERMINISTIC BEGIN

RETURN DATEDIFF(fecha\_real, fecha\_devolucion);

END;

// DELIMITER ;

# 6️⃣ Procedimientos almacenados

sql

DELIMITER //

CREATE PROCEDURE registrar\_prestamo( IN p\_id\_usuario INT,

IN p\_fecha DATE

) BEGIN

INSERT INTO Prestamo(id\_usuario, fecha\_prestamo) VALUES (p\_id\_usuario, p\_fecha);

END;

// DELIMITER ;

# 7️⃣ Triggers

sql

DELIMITER //

CREATE TRIGGER validar\_fecha\_devolucion BEFORE UPDATE ON Prestamo

FOR EACH ROW BEGIN

IF NEW.fecha\_devolucion < NEW.fecha\_prestamo THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'La fecha de devolución no puede ser anterior a la de préstamo';

END IF;

END;

// DELIMITER ;

# 8️⃣ Vistas

sql

CREATE VIEW vista\_prestamos\_usuario AS

SELECT u.nombre, p.fecha\_prestamo, p.fecha\_devolucion FROM Usuario u

JOIN Prestamo p ON u.id\_usuario = p.id\_usuario;