***Curso:*** *SQL*

***Tema:*** *Proyecto Final*

***Alumno:*** *Roberto Agustín Mejía Collazos*

BASES DE DATOS

**INDICE**

Contenido

[PROYECTO CODERHOUSE: BIBLIOTECA 3](#_Toc179724774)

[1. Introducción 3](#_Toc179724775)

[1. Objetivos 3](#_Toc179724776)

[2. Situación problemática 3](#_Toc179724777)

[3. Modelo de negocio 4](#_Toc179724778)

[4. Diagrama de Entidad Relación 5](#_Toc179724779)

[5.1. Boceto 5](#_Toc179724780)

[5.3. Diagrama 2 (DrawSQL) 6](#_Toc179724781)

[5.4. Diagrama 3 (EER Diagram) 7](#_Toc179724782)

[5. Listado de tablas con descripción de estructuras 8](#_Toc179724783)

[5. Scripts de creación de cada objeto de la base de datos 8](#_Toc179724784)

[7.1 CREACIÓN DE BASE DE DATOS 8](#_Toc179724785)

[7.2 CREACIÓN DE TABLAS 9](#_Toc179724786)

[6. Scripts de inserción de datos 10](#_Toc179724787)

[9. Lista de Vistas 14](#_Toc179724788)

[10. Lista de Funciones 14](#_Toc179724789)

[11. Lista de Stored procedures 14](#_Toc179724790)

[12. Lista Triggers 14](#_Toc179724791)

[13. Scripts de Vistas 15](#_Toc179724792)

[14. Script de Funciones 16](#_Toc179724793)

[15. Scripts de Stored Procedures 17](#_Toc179724794)

[16. Scripts de Triggers 18](#_Toc179724795)

[17. Tablas, Views, Stored Procedures y funciones en Workbench 19](#_Toc179724796)

[18. Herramientas y tecnologías usadas 20](#_Toc179724797)

[16. Futuras líneas 20](#_Toc179724798)

# PROYECTO CODERHOUSE: BIBLIOTECA

# Introducción

El presente informe detalla el desarrollo de un proyecto de gestión de bibliotecas implementado en MySQL. El objetivo principal del sistema es facilitar la administración de los recursos bibliográficos, como libros, autores, géneros, editoriales, miembros y préstamos, mediante un diseño de base de datos relacional que optimiza la gestión y consulta de información. La base de datos está estructurada para soportar la compleja relación entre los diferentes elementos, garantizando así una gestión eficiente y coherente de la biblioteca.

# Objetivos

**Desarrollo de un sistema de gestión:** Crear una base de datos que permita administrar la información relacionada con los libros, autores, géneros, editoriales, miembros y préstamos en una biblioteca de manera eficiente.

**Optimización de consultas:** Diseñar un modelo de base de datos que soporte consultas rápidas y precisas, permitiendo a los administradores recuperar y manipular información de manera efectiva.

**Integridad de los datos:** Implementar restricciones y claves foráneas para asegurar la integridad referencial de los datos y evitar inconsistencias.

**Facilitar la escalabilidad:** Estructurar la base de datos de manera que pueda ser escalable y adaptarse a futuras expansiones de la biblioteca.

# Situación problemática

En muchas bibliotecas tradicionales, la gestión de los recursos bibliográficos y el control de préstamos se realizan manualmente o con sistemas obsoletos, lo que genera ineficiencias, pérdida de tiempo, y errores humanos. Estas ineficiencias pueden llevar a la pérdida de libros, errores en el seguimiento de los préstamos, y dificultades en la recuperación de información. Ante esta problemática, surge la necesidad de implementar un sistema de gestión automatizado que permita llevar un control más preciso y eficiente de los recursos bibliográficos, mejorando así la calidad del servicio que se brinda a los usuarios.

# Modelo de negocio

El modelo de negocio del sistema de gestión de bibliotecas se basa en ofrecer una herramienta digital robusta que permita a las bibliotecas optimizar sus operaciones diarias. El sistema permite:

**Gestión de inventarios:** Administrar la información relacionada con los libros, incluyendo su título, año de publicación, género y editorial.

**Registro de autores y editoriales:** Mantener una base de datos actualizada de autores y editoriales, lo que facilita la búsqueda y organización de los libros.

**Administración de miembros:** Gestionar la información de los miembros de la biblioteca, permitiendo un control preciso sobre el registro de nuevos miembros y el seguimiento de los actuales.

**Control de préstamos:** Facilitar la administración de préstamos y devoluciones, garantizando que los libros se manejen de manera adecuada y reduciendo el riesgo de pérdidas.

El sistema está diseñado para ser escalable, permitiendo la inclusión de nuevas funcionalidades conforme las necesidades de la biblioteca evolucionen. Este enfoque asegura que la inversión en la implementación del sistema sea sostenible a largo plazo.

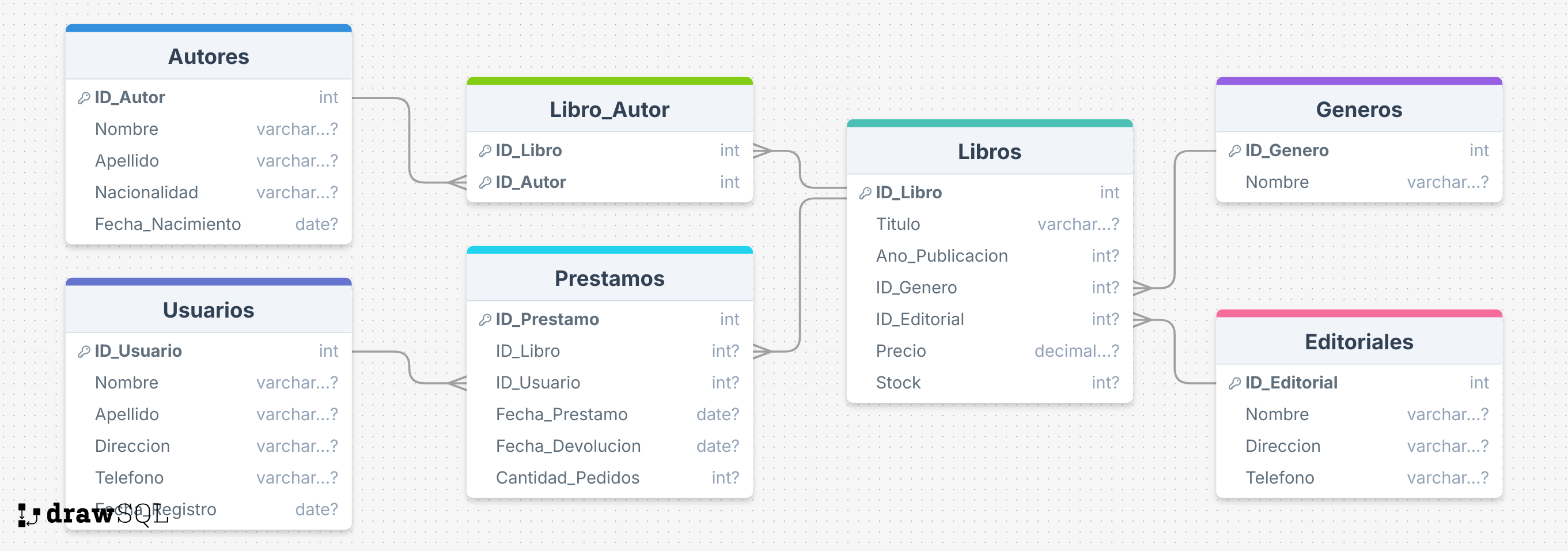
**ENLACE DE REPOSITORIO:**

[**https://github.com/robermejia/coderhouse\_sql\_59410/tree/main/3\_Entrega\_Final**](https://github.com/robermejia/coderhouse_sql_59410/tree/main/3_Entrega_Final)

# Diagrama de Entidad Relación

## 5.1. Boceto

## 5.3. Diagrama 2 (DrawSQL)



## Imagen que contiene Interfaz de usuario gráfica Descripción generada automáticamente5.4. Diagrama 3 (EER Diagram)

# Tabla Descripción generada automáticamente5. Listado de tablas con descripción de estructuras

# Scripts de creación de cada objeto de la base de datos

## 7.1 CREACIÓN DE BASE DE DATOS

*-- ELIMINACIÓN DE BASE DE DATOS*

DROP DATABASE IF EXISTS proyecto\_biblioteca;

*-- CREACIÓN DE BASE DE DATOS*

CREATE DATABASE proyecto\_biblioteca;

*-- POSICIONARSE EN LA BD*

USE proyecto\_biblioteca;

## 7.2 CREACIÓN DE TABLAS

*-- CREACIÓN DE TABLAS*

CREATE TABLE IF NOT EXISTS Editoriales (

    ID\_Editorial INT AUTO\_INCREMENT PRIMARY KEY,

    Nombre VARCHAR(255),

    Direccion VARCHAR(255),

    Telefono VARCHAR(20)

);

CREATE TABLE IF NOT EXISTS Generos (

    ID\_Genero INT AUTO\_INCREMENT PRIMARY KEY,

    Nombre VARCHAR(255)

);

CREATE TABLE IF NOT EXISTS Usuarios (

    ID\_Usuario INT AUTO\_INCREMENT PRIMARY KEY,

    Nombre VARCHAR(255),

    Apellido VARCHAR(255),

    Direccion VARCHAR(255),

    Telefono VARCHAR(20),

    Fecha\_Registro DATE

);

CREATE TABLE IF NOT EXISTS Autores (

    ID\_Autor INT PRIMARY KEY,

    Nombre VARCHAR(255),

    Apellido VARCHAR(255),

    Nacionalidad VARCHAR(100),

    Fecha\_Nacimiento DATE

);

CREATE TABLE IF NOT EXISTS Libros (

    ID\_Libro INT PRIMARY KEY,

    Titulo VARCHAR(255),

    Ano\_Publicacion INT,

    ID\_Genero INT,

    ID\_Editorial INT,

    Precio DECIMAL(10, 2),

    Stock INT,

    FOREIGN KEY (ID\_Genero) REFERENCES Generos(ID\_Genero),

    FOREIGN KEY (ID\_Editorial) REFERENCES Editoriales(ID\_Editorial)

);

CREATE TABLE IF NOT EXISTS Prestamos (

    ID\_Prestamo INT AUTO\_INCREMENT PRIMARY KEY,

    ID\_Libro INT,

    ID\_Usuario INT,

    Fecha\_Prestamo DATE,

    Fecha\_Devolucion DATE,

    Cantidad\_Pedidos INT,

    FOREIGN KEY (ID\_Libro) REFERENCES Libros(ID\_Libro),

    FOREIGN KEY (ID\_Usuario) REFERENCES Usuarios(ID\_Usuario)

);

CREATE TABLE IF NOT EXISTS Libro\_Autor (

    ID\_Libro INT,

    ID\_Autor INT,

    PRIMARY KEY (ID\_Libro, ID\_Autor),

    FOREIGN KEY (ID\_Libro) REFERENCES Libros(ID\_Libro),

    FOREIGN KEY (ID\_Autor) REFERENCES Autores(ID\_Autor)

);

*-- VISUALIZAR TODAS LAS TABLAS DE LA BASE DE DATOS*

SHOW TABLES;

# Scripts de inserción de datos

1. *-- INSERCIONES MULTIPLES*
2. INSERT INTO Editoriales (Nombre, Direccion, Telefono) VALUES
3. ('Editorial Planeta', 'Av. Javier Prado 123, Lima', '01-2345678'),
4. ('Penguin Random House', 'Calle de las Letras 45, Madrid', '+34 910 123 456'),
5. ('HarperCollins', '195 Broadway, New York', '+1 212-207-7000'),
6. ('Simon & Schuster', '1230 Avenue of the Americas, New York', '+1 212-698-7000'),
7. ('Alfaguara', 'Calle Alfonso XII, 62, Madrid', '+34 917 595 300'),
8. ('Anagrama', 'Calle Pedró Martell, 19, Barcelona', '+34 933 687 850'),
9. ('Random House Mondadori', 'Av. Diagonal, 662-664, Barcelona', '+34 934 928 840'),
10. ('Editorial SM', 'Calle de Impresores, 2, Boadilla del Monte', '+34 917 596 400'),
11. ('Santillana', 'Calle del Tambre, 50, Madrid', '+34 913 984 500'),
12. ('Edebé', 'Calle de Provença, 386, Barcelona', '+34 934 535 500'),
13. ('Minotauro', 'Calle de Espronceda, 32, Barcelona', '+34 933 663 600'),
14. ('Roca Editorial', 'Calle de Roger de Llúria, 118, Barcelona', '+34 932 082 040');
15. INSERT INTO Generos (Nombre) VALUES
16. ('Novela'),
17. ('Ciencia Ficción'),
18. ('Historia'),
19. ('Fantasía'),
20. ('Biografía'),
21. ('Romance'),
22. ('Misterio'),
23. ('Terror'),
24. ('Aventura'),
25. ('Poesía'),
26. ('Ensayo'),
27. ('Drama');
28. INSERT INTO Usuarios (Nombre, Apellido, Direccion, Telefono, Fecha\_Registro) VALUES
29. ('Carlos', 'López', 'Calle Falsa 123, Lima', '987654321', '2024-01-15'),
30. ('María', 'Pérez', 'Av. Universitaria 678, Lima', '912345678', '2024-02-20'),
31. ('José', 'Martínez', 'Av. Pardo y Aliaga 120, Lima', '987654322', '2024-03-10'),
32. ('Lucía', 'Gómez', 'Jirón Miraflores 567, Lima', '987654323', '2024-04-05'),
33. ('Andrés', 'Ramírez', 'Calle Las Lomas 876, Lima', '987654324', '2024-05-12'),
34. ('Ana', 'Torres', 'Av. Brasil 450, Lima', '987654325', '2024-06-18'),
35. ('Jorge', 'Rodríguez', 'Calle Los Sauces 34, Lima', '987654326', '2024-07-25'),
36. ('Elena', 'Gutiérrez', 'Av. Benavides 223, Lima', '987654327', '2024-08-01'),
37. ('Ricardo', 'Méndez', 'Calle Los Olivos 789, Lima', '987654328', '2024-08-15'),
38. ('Isabel', 'Fernández', 'Jirón Lampa 234, Lima', '987654329', '2024-09-20'),
39. ('Rosa', 'García', 'Av. San Felipe 453, Lima', '987654330', '2024-10-05'),
40. ('Diego', 'Herrera', 'Calle Las Flores 123, Lima', '987654331', '2024-11-10');
41. INSERT INTO Autores (ID\_Autor, Nombre, Apellido, Nacionalidad, Fecha\_Nacimiento) VALUES
42. (1, 'Gabriel', 'García Márquez', 'Colombiana', '1927-03-06'),
43. (2, 'Isaac', 'Asimov', 'Rusa-Americana', '1920-01-02'),
44. (3, 'Mario', 'Vargas Llosa', 'Peruana', '1936-03-28'),
45. (4, 'J.K.', 'Rowling', 'Británica', '1965-07-31'),
46. (5, 'George', 'Orwell', 'Británica', '1903-06-25'),
47. (6, 'Julio', 'Verne', 'Francesa', '1828-02-08'),
48. (7, 'Jane', 'Austen', 'Británica', '1775-12-16'),
49. (8, 'Ernest', 'Hemingway', 'Americana', '1899-07-21'),
50. (9, 'Agatha', 'Christie', 'Británica', '1890-09-15'),
51. (10, 'H.P.', 'Lovecraft', 'Americana', '1890-08-20'),
52. (11, 'Miguel de', 'Cervantes', 'Española', '1547-09-29'),
53. (12, 'Virginia', 'Woolf', 'Británica', '1882-01-25');
54. INSERT INTO Libros (ID\_Libro, Titulo, Ano\_Publicacion, ID\_Genero, ID\_Editorial, Precio, Stock) VALUES
55. (1, 'Cien Años de Soledad', 1967, 1, 1, 150.50, 10),
56. (2, 'Fundación', 1951, 2, 2, 120.75, 15),
57. (3, 'La Casa Verde', 1966, 1, 3, 95.25, 8),
58. (4, 'Harry Potter y la Piedra Filosofal', 1997, 4, 4, 180.30, 20),
59. (5, '1984', 1949, 6, 5, 88.90, 12),
60. (6, 'Viaje al Centro de la Tierra', 1864, 4, 6, 75.60, 5),
61. (7, 'Orgullo y Prejuicio', 1813, 5, 7, 68.40, 7),
62. (8, 'El Viejo y el Mar', 1952, 1, 8, 105.80, 18),
63. (9, 'Asesinato en el Orient Express', 1934, 7, 9, 110.20, 14),
64. (10, 'La Llamada de Cthulhu', 1928, 8, 10, 95.75, 9),
65. (11, 'Don Quijote de la Mancha', 1605, 1, 11, 65.90, 25),
66. (12, 'Al Faro', 1927, 12, 12, 82.45, 6);

INSERT INTO Prestamos (ID\_Libro, ID\_Usuario, Fecha\_Prestamo, Fecha\_Devolucion, Cantidad\_Pedidos) VALUES

(1, 1, '2024-08-10', '2024-08-20', 2),

(2, 2, '2024-08-15', '2024-08-25', 1),

(3, 3, '2024-07-01', '2024-07-15', 3),

(4, 4, '2024-07-20', '2024-08-05', 0),

(5, 5, '2024-06-25', '2024-07-05', 5),

(6, 6, '2024-05-15', '2024-05-25', 2),

(7, 7, '2024-04-10', '2024-04-20', 1),

(8, 8, '2024-03-18', '2024-03-28', 4),

(9, 9, '2024-02-20', '2024-03-01', 0),

(10, 10, '2024-01-12', '2024-01-22', 2),

(11, 11, '2024-12-05', '2024-12-15', 3),

(12, 12, '2024-11-01', '2024-11-11', 1);

INSERT INTO Libro\_Autor (ID\_Libro, ID\_Autor) VALUES

(1, 1),

(2, 2),

(3, 3),

(4, 4),

(5, 5),

(6, 6),

(7, 7),

(8, 8),

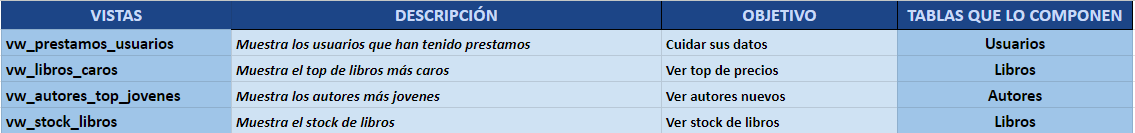
(9, 9),

(10, 10),

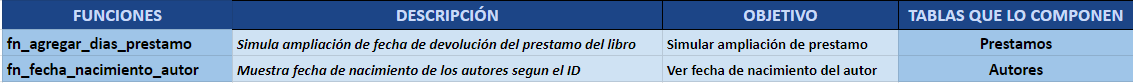
(11, 11),

(12, 12);

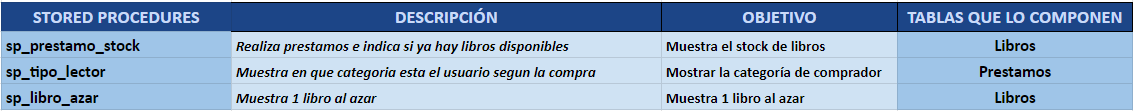
# 9. Lista de Vistas



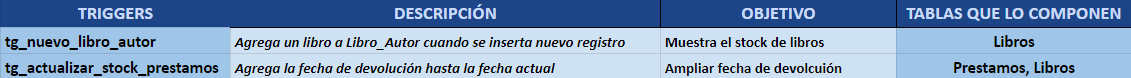
# 10. Lista de Funciones



# 11. Lista de Stored procedures



# 12. Lista Triggers



# 13. Scripts de Vistas

*-- ######################## VISTAS #######################*

*-- ========= VISTA 1 - PRESTAMOS DE LIBROS =========*

DROP VIEW IF EXISTS vw\_prestamos\_usuarios; *-- Eliminando la vista si es que existe*

*-- CREANDO LA VISTA "prestamos\_usuarios\_vw"*

CREATE VIEW vw\_prestamos\_usuarios AS

SELECT usuarios.ID\_Usuario, usuarios.Nombre,

MAX(prestamos.Fecha\_Devolucion) AS Ultima\_Fecha\_Devolucion

FROM usuarios INNER JOIN prestamos ON usuarios.ID\_Usuario = prestamos.ID\_Usuario

GROUP BY usuarios.ID\_Usuario;

*-- ========= VISTA 2 - LIBROS MÁS CAROS =========*

DROP VIEW IF EXISTS vw\_libros\_caros; *-- Eliminando la vista si es que existe*

CREATE VIEW vw\_libros\_caros AS

SELECT ID\_Libro, Titulo, Precio

FROM libros ORDER BY Precio DESC LIMIT 5;

*-- ========= VISTA 3 - AUTOR MÁS JOVEN =========*

DROP VIEW IF EXISTS vw\_autores\_top\_jovenes; *-- Eliminando la vista si es que existe*

CREATE VIEW vw\_autores\_top\_jovenes AS

SELECT ID\_Autor, Nombre, Nacionalidad, Fecha\_Nacimiento

FROM autores ORDER BY ABS(DATEDIFF(Fecha\_Nacimiento, NOW())) LIMIT 5;

*-- ========= VISTA 4 - STOCK DE LIBROS =========*

DROP VIEW IF EXISTS vw\_stock\_libros; *-- Eliminando la vista si es que existe*

CREATE VIEW vw\_stock\_libros AS

SELECT COUNT(ID\_Libro) AS STOCK\_LIBROS FROM Libros;

# 14. Script de Funciones

*-- ####################### FUNCIONES #######################*

*-- FUNCIÓN SIMULACIÓN DE AMPLIACIÓN DE FECHA DE DEVOLUCIÓN*

DROP FUNCTION IF EXISTS fn\_agregar\_dias\_prestamo;

DELIMITER //

CREATE FUNCTION fn\_agregar\_dias\_prestamo (prestamo\_id INT, dias INT)

RETURNS VARCHAR(255)

READS SQL DATA

BEGIN

    DECLARE fecha\_prestamo DATE;

    DECLARE fecha\_limite DATE;

    SELECT Fecha\_Devolucion INTO fecha\_prestamo

    FROM prestamos

    WHERE ID\_Prestamo = prestamo\_id;

    SET fecha\_limite = fecha\_prestamo + INTERVAL dias DAY;

    RETURN CONCAT('Fecha de devolución: ', DATE\_FORMAT(fecha\_prestamo, '%Y-%m-%d'),

                ' - Nueva fecha límite: ', DATE\_FORMAT(fecha\_limite, '%Y-%m-%d'));

END;

//

DELIMITER ;

*-- FUNCIÓN VER FECHA DE NACIMIENTO DE AUTORES POR SU ID*

DROP FUNCTION IF EXISTS fn\_fecha\_nacimiento\_autor;

DELIMITER //

CREATE FUNCTION fn\_fecha\_nacimiento\_autor (autor\_id INT)

RETURNS DATE

READS SQL DATA

BEGIN

    DECLARE fecha\_autor DATE;

    SELECT Fecha\_Nacimiento INTO fecha\_autor

    FROM autores

    WHERE ID\_Autor = autor\_id;

    RETURN fecha\_autor;

END;

//

DELIMITER ;

# 15. Scripts de Stored Procedures

*-- ################# STORES PROCEDURES #####################*

*-- ======= PROCEDIMIENTO 1 - PRÉSTAMO DE STOCK =======*

DROP PROCEDURE IF EXISTS sp\_prestamo\_stock;

DELIMITER //

CREATE PROCEDURE sp\_prestamo\_stock(IN ID\_Miembro INT, IN ID\_Libro INT, OUT cantidad INT)

BEGIN

    DECLARE cantidad\_libros INT;

    SELECT Stock INTO cantidad\_libros FROM libros WHERE libros.ID\_Libro = ID\_Libro;

    IF cantidad\_libros > 0 THEN

*-- Actualizar el stock en la tabla libros*

        UPDATE libros SET Stock = Stock - 1 WHERE libros.ID\_Libro = ID\_Libro;

        SET cantidad = cantidad\_libros - 1;

        SELECT "Se ha realizado el préstamo exitosamente." AS Mensaje;

    ELSE

        SELECT "No es posible realizar préstamos." AS Mensaje;

    END IF;

END //

DELIMITER ;

*-- ======= PROCEDIMIENTO 2 - CATEGORÍA DE USUARIOS =======*

DROP PROCEDURE IF EXISTS sp\_tipo\_lector;

DELIMITER //

CREATE PROCEDURE sp\_tipo\_lector(IN ID\_Usuario INT)

BEGIN

    SET @cantidad = (SELECT Cantidad\_Pedidos FROM prestamos

                    WHERE prestamos.ID\_Usuario = ID\_Usuario);

    CASE

        WHEN @cantidad = 5 THEN

            SELECT "Fanático" AS Mensaje;

        WHEN @cantidad = 4 THEN

            SELECT "Aficionado" AS Mensaje;

        WHEN @cantidad = 3 THEN

            SELECT "Promedio" AS Mensaje;

        ELSE

            SELECT "Nuevo" AS Mensaje;

    END CASE;

END //

DELIMITER ;

*-- ======= PROCEDIMIENTO 3 - LIBROS AL AZAR =======*

DROP PROCEDURE IF EXISTS sp\_libro\_azar; *-- Eliminando el procedimiento si es que existe*

DELIMITER //

CREATE PROCEDURE sp\_libro\_azar()

BEGIN

    SET @iterador = 0;

    WHILE @iterador < 3 DO

        SELECT ID\_Libro, Titulo FROM libros ORDER BY RAND() LIMIT 1;

        SET @iterador = @iterador + 1;

    END WHILE;

END //

DELIMITER ;

# 16. Scripts de Triggers

*-- ####################### TRIGGERS #######################*

*-- ###################### TRIGGERS 1 ########################*

DELIMITER //

CREATE TRIGGER tg\_nuevo\_libro\_autor

AFTER INSERT ON Libros

FOR EACH ROW

BEGIN

*-- Verificar si el libro ya existe en la tabla Libro\_Autor*

    IF NOT EXISTS (SELECT 1 FROM Libro\_Autor WHERE ID\_Libro = NEW.ID\_Libro) THEN

*-- Insertar un nuevo registro en la tabla Libro\_Autor*

    INSERT INTO Libro\_Autor (ID\_Libro, ID\_Autor) VALUES (NEW.ID\_Libro, 1); *-- Aquí puedes cambiar el ID\_Autor por el que consideres*

    END IF;

END //

DELIMITER ;

*-- ################################# TRIGGERS 2 ################################*

DELIMITER //

CREATE TRIGGER tg\_actualizar\_stock\_prestamo

AFTER INSERT ON Prestamos

FOR EACH ROW

BEGIN

*-- Actualizar el stock del libro en la tabla Libros*

    UPDATE Libros

    SET Stock = Stock - NEW.Cantidad\_Pedidos

    WHERE ID\_Libro = NEW.ID\_Libro;

END //

DELIMITER ;

# 17. Usuarios

*-- ################################# USUARIOS #################################*

*-- ========= USER 1 (SIN PERMISOS) ===============*

CREATE USER 'coderhouse'@'%'

IDENTIFIED BY 'coderhouse';

*-- ========= USER 2 (CON TODOS LOS DERECHOS) ===============*

CREATE USER 'coderhouse\_alumno'@'%'

IDENTIFIED BY 'coderhouse'

COMMENT 'ESTE USUARIO SOLO VA ACCEDER POR MEDIO DE WORKBENCH';

GRANT ALL ON \*.\* TO 'coderhouse\_alumno' @'%'; *-- Dar permisos*

*-- ========= USER 3 (CON TODOS LOS DERECHOS)===============*

CREATE USER 'coderhouse\_docente'@'%'

IDENTIFIED BY 'coderhouse'

COMMENT 'ESTE USUARIO SOLO VA ACCEDER POR MEDIO DE WORKBENCH';

GRANT ALL ON \*.\* TO 'coderhouse\_docente'@'%'; *-- Dar permisos*

*-- ========= USER 4 (CON ACCESO A UNA BASE DE DATOS) ===============*

CREATE USER 'coderhouse\_invitado'@'%'

IDENTIFIED BY 'coderhouse'

COMMENT 'ESTE USUARIO SOLO VA ACCEDER POR MEDIO DE WORKBENCH';

GRANT ALL ON proyecto\_biblioteca.\* TO 'coderhouse\_invitado'@'%'; *-- Dar permisos a una sola bd*

# 18. Tablas, Views, Stored Procedures y funciones en Workbench

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

# 19. Herramientas y tecnologías usadas

* + *MySQL Workbench*
  + *MySQL 8.0 Command Line Client*
  + *Visual Studio Code*
  + *Windows PowerShell*
  + *GitHub*

# 20. Futuras líneas