**DESARROLLO DE UN SISTEMA DE BASE DE DATOS DE LA BIBLIOTECA DE LA UNIVERSIDAD NACIONAL DE MOQUEGUA**

Marlon Raul Lovon Ticona 100%

Plinio Oscar Choquesa Mamani 100%

Docente: Luis Manuel Torres Carpio



UNIVERSIDAD NACIONAL DE MOQUEGUA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

BASE DE DATOS I

**ILO – PERU**

**2024**

**INDICE**

INTRODUCCIÓN 3

JUSTIFICACIÓN 4

OBJETIVOS 5

ANTECEDENTES 6

FUENTES A TOMAR 6

METODO A USAR 8

MODELO LOGICO 8

CREACION DE BASE DE DATOS 11

CONDUCTA\_USUARIO 11

USUARIOS 12

PRESTAMOS 13

LIBROS 14

CUBICULOS 15

RENOVACIONES 16

INFRACCIONES 17

PROCEDIMIENTO ALMACERADOS Y TRIGGERS 18

PUBLICACION EN LA WEB 22

DESARROLLO EN EDITOR VISUAL STUDIO CODE 25

Lista de personas 29

Libros 32

Cubículos 35

Lista de préstamos 37

Lista de usuarios con falta 39

Registro de nuevo prestamos 42

Registro de devolución 44

Reserva o devolución de cubículo 47

Registrar faltas de los usuarios 50

Registro de nuevo usuario 52

CONCLUSIONES 56

BIBLIOGRA 58

**INTRODUCCIÓN**

En la era digital, la gestión eficiente de la información es esencial para optimizar los procesos y mejorar la experiencia del usuario. En el contexto de una biblioteca, la organización y el acceso a los recursos son fundamentales para facilitar el préstamo de los libros que ofrecen, la devolución y la consulta de libros. Sin embargo, muchos sistemas tradicionales de gestión de bibliotecas dependen de métodos manuales que no permiten un control eficiente ni una actualización precisa de los registros.

Este informe tiene como objetivo desarrollar una base de datos estructurada para la biblioteca, enfocada en organizar y simplificar la administración de su catálogo de libros y otros recursos. Mediante la implementación de una base de datos relacional, se busca mejorar el almacenamiento, la introducción y la actualización de datos, optimizando así la gestión de préstamos, devoluciones y el seguimiento del inventario de manera integral. Además, se abordarán las consideraciones técnicas, el diseño de las tablas, las relaciones entre entidades, y las consultas que facilitarán el acceso a la información de manera rápida y precisa.

**JUSTIFICACIÓN**

La administración de bibliotecas implica el manejo de grandes volúmenes de información, que incluyen el registro de libros, el historial de préstamos, la gestión de usuarios y la actualización constante del inventario. En muchas bibliotecas, estos procesos son complejos y, en algunos casos, se gestionan de manera manual o mediante sistemas obsoletos, lo cual puede llevar a ineficiencias y dificultades en la consulta y organización de los recursos.

La creación de un sistema de base de datos para la biblioteca se justifica como una solución para mejorar la eficiencia y precisión en el manejo de información bibliográfica. Una base de datos relacional permite centralizar y estructurar todos los datos relevantes, brindando la posibilidad de realizar búsquedas rápidas, establecer relaciones claras entre los diferentes elementos (como usuarios, libros y préstamos), y mantener un control actualizado del inventario. Esta estructura reduce el margen de error, facilita la toma de decisiones, y contribuye a una experiencia de usuario mejorada tanto para los administradores como para los usuarios de la biblioteca.

Además, será una oportunidad académica para aplicar y profundizar en los conceptos de diseño y administración de bases de datos, y responde a la creciente necesidad de sistemas de información eficientes en instituciones educativas y centros de documentación. En resumen, la implementación de esta base de datos no solo facilitará los procesos internos de la biblioteca, sino que también contribuirá al desarrollo de competencias técnicas en bases de datos, indispensables para la formación en el ámbito de las tecnologías de la información.

**OBJETIVOS**

1. **Gestión de Inventario**: Crear un sistema centralizado que permita registrar, organizar y actualizar la información de los libros y otros materiales disponibles en la biblioteca universitaria. Este objetivo implica el desarrollo de una estructura de datos que permita almacenar información detallada sobre cada recurso, incluyendo el título, el autor, la fecha de publicación, el género, año. De esta forma, se garantiza que el inventario esté actualizado y accesible, lo que reduce el riesgo de pérdida de información y facilita el control de los recursos disponibles para los usuarios.
2. **Optimización del Préstamo y la Devolución**: Implementar un sistema que gestione de manera eficiente los procesos de préstamo y devolución de material bibliográfico. Este objetivo incluye el registro de las fechas de préstamo y devolución de cada recurso, así como la posibilidad de gestionar reservas y aplicar penalizaciones por retrasos en las devoluciones. De esta forma, se asegura un mejor control sobre el movimiento de los materiales, lo que contribuye a reducir la acumulación de libros en préstamo y a mejorar la disponibilidad de recursos para los estudiantes y el personal universitario.
3. **Búsqueda Eficiente**: Diseñar un sistema de consulta que permita realizar búsquedas rápidas y precisas de los materiales disponibles en la biblioteca. Este objetivo tiene como finalidad optimizar el tiempo que los usuarios dedican a localizar información, al proporcionar detalles específicos sobre los recursos, como el autor, el tema, la fecha de publicación y año.
4. **Consultas y Reportes**: Desarrollar una funcionalidad de generación de reportes estadísticos para analizar el comportamiento de los usuarios y el uso de los recursos de la biblioteca. Este objetivo permite recopilar datos sobre los patrones de préstamo, los materiales más solicitados, los usuarios más activos, y el número de reservas y devoluciones. Estos reportes no solo facilitan la toma de decisiones informadas sobre la gestión de la biblioteca, sino que también ofrecen una visión general del impacto de los servicios bibliotecarios en la comunidad universitaria y ayudan a identificar oportunidades de mejora en la gestión de los recursos.

**ANTECEDENTES**

En las últimas décadas, diversas universidades han implementado sistemas de bases de datos para mejorar la administración de sus bibliotecas y ofrecer un mejor servicio a su comunidad académica. Un ejemplo destacado es el caso de la Universidad Nacional de Moquegua (UNAM), que, ante el crecimiento de su colección y el aumento en la demanda de sus servicios, implementó un sistema de base de datos digital en su biblioteca. Este sistema fue desarrollado para mejorar la organización de sus recursos, facilitar los procesos de préstamo y devolución, y optimizar el acceso a su amplia colección de material académico.

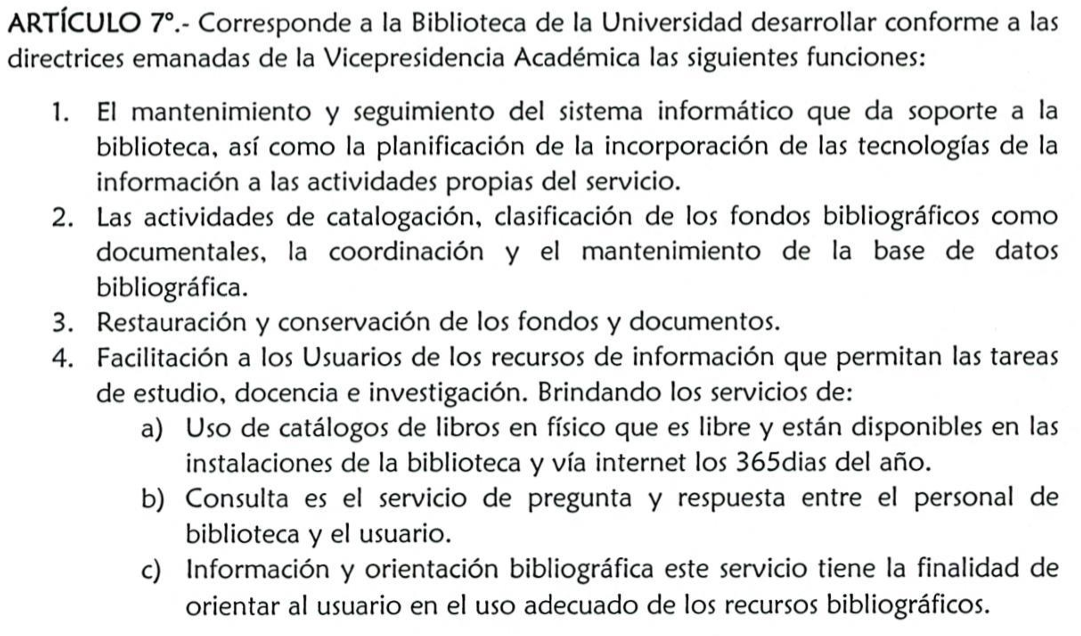
La base de datos permitió a la biblioteca registrar digitalmente todos los detalles de su inventario, como el título, el autor, la fecha de publicación, el número de copias disponibles y año. Además, se implementó un sistema para gestionar los préstamos y devoluciones. La digitalización también permitió al personal de la biblioteca monitorear los recursos en tiempo real, gestionando eficientemente las reservas y aplicando sanciones automáticas por devoluciones tardías.

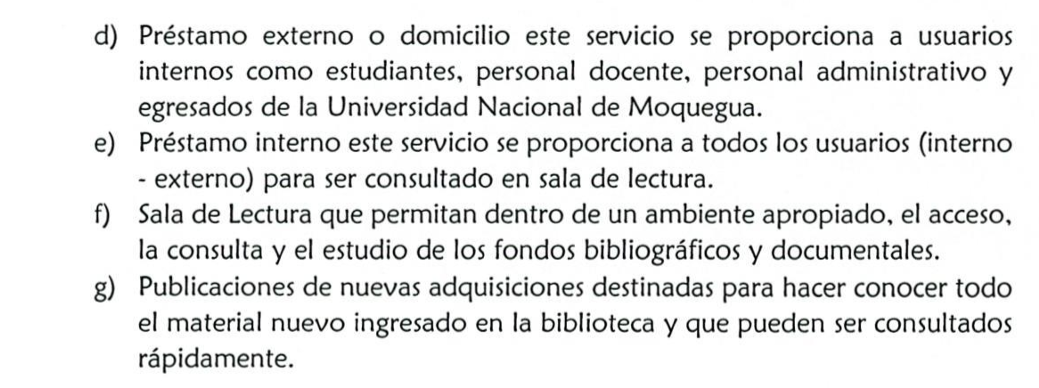
Este avance no solo mejoró la organización y la accesibilidad de los libros, sino que también permitió generar reportes detallados sobre el comportamiento de los usuarios, identificando los libros y temas más solicitados, lo cual fue de gran utilidad para la toma de decisiones en la adquisición de nuevos materiales.

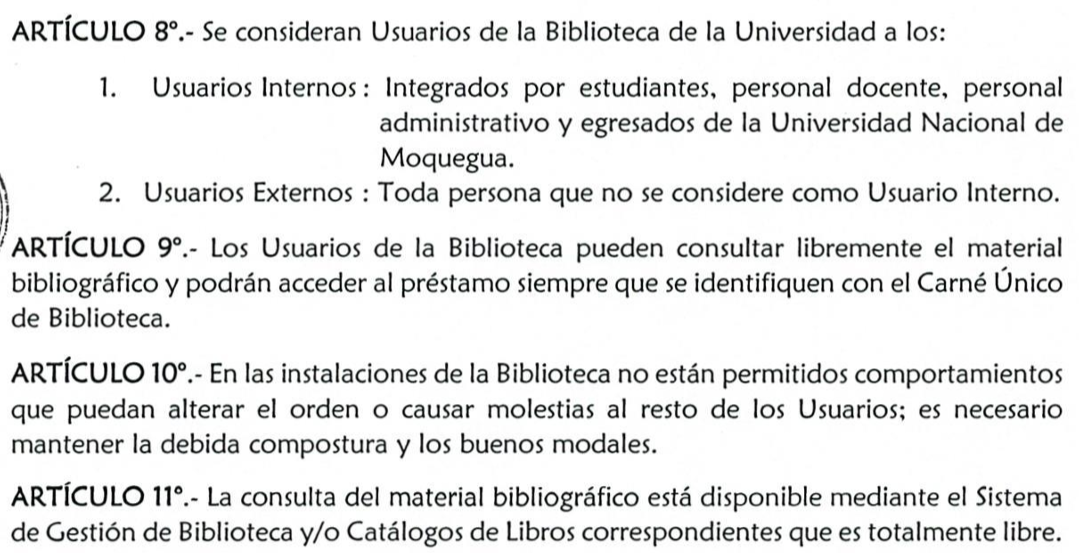
Este caso de la UNAM muestra cómo la implementación de una base de datos puede transformar la administración y operación de una biblioteca universitaria, facilitando el acceso a la información, mejorando la experiencia de los usuarios, y permitiendo una administración más eficiente y estratégica de los recursos

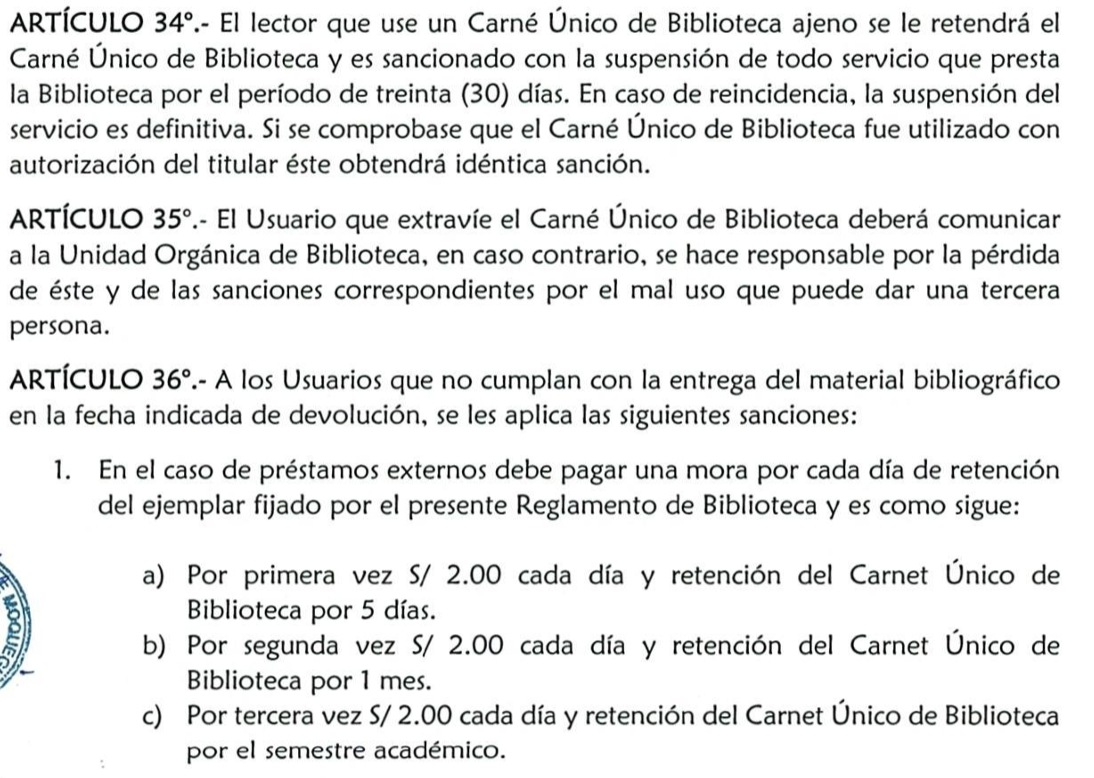
**FUENTES A TOMAR**

Las fuentes que tomamos primordial fue la resolución emitida de la Universidad Nacional de Moquegua para la biblioteca tomamos ciertos artículos importantes para ver un panorama más amplio de cuáles serían nuestras entidades principales para la generación de nuestra base de datos









**METODO A USAR**

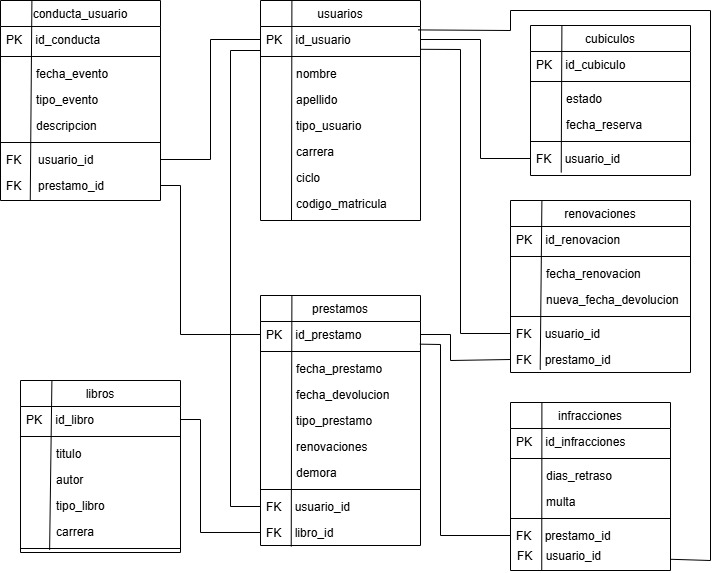
- El Método XP (Extreme Programming) es una metodología de Desarrollo Ágil que se enfoca en mejorar la calidad del software y la capacidad de respuesta a los requisitos cambiantes del cliente.

- Promueve la colaboración constante entre los miembros del equipo, la retroalimentación continua y el desarrollo en ciclos iterativos.

- En resumen, el método XP aplicado al desarrollo de bases de datos promueve un enfoque iterativo, colaborativo y de mejora constante. Facilita la creación de bases de datos que evolucionan de forma ágil junto con las necesidades del proyecto.

**MODELO LOGICO**

Teniendo estos datos generamos nuestras entidades con sus atributos y respectivas relaciones



Descripción del Modelo de Base de Datos para la Biblioteca Universitaria

El modelo de base de datos diseñado para la biblioteca universitaria se organiza en varias tablas que representan las entidades y sus relaciones. A continuación, se describen cada una de las entidades y sus atributos principales:

1. **Usuarios:** Esta tabla almacena la información de los usuarios de la biblioteca, tales como id\_usuario (clave primaria), nombre, apellido, tipo\_usuario (por ejemplo, estudiante, profesor), carrera, ciclo, y codigo\_matricula. Estos datos permiten identificar a cada usuario y agruparlos según su rol en la universidad.
2. **Libros:** La tabla de libros contiene detalles sobre cada recurso disponible en la biblioteca. Sus atributos incluyen id\_libro (clave primaria), titulo, autor, año\_publicación, tipo\_libro y carrera, fecha\_adquisición, estado, num\_ejemplares, prestamos\_totales, lo cual permite organizar los libros según su temática o la carrera a la que están orientados.
3. **Préstamos:** Esta tabla gestiona el proceso de préstamo de libros, con atributos como id\_prestamo (clave primaria), fecha\_prestamo, fecha\_devolucion, tipo\_prestamo, renovaciones (número de veces que se ha renovado el préstamo) y demora. Las claves foráneas usuario\_id y libro\_id vinculan a esta tabla con las de Usuarios y Libros, permitiendo el registro del préstamo de un libro específico a un usuario.
4. **Renovaciones**: Aquí se registran las renovaciones de préstamos realizados por los usuarios. La tabla cuenta con atributos como id\_renovacion (clave primaria), fecha\_renovacion y nueva\_fecha\_devolucion, nueva\_fecha\_devolución, además de las claves foráneas usuario\_id y prestamo\_id que enlazan esta tabla con las tablas de Usuarios y Préstamos. Esta estructura permite un seguimiento detallado de las extensiones de los préstamos.
5. **Cubículos:** Esta tabla se refiere a la reserva de cubículos en la biblioteca. Sus atributos son id\_cubiculo (clave primaria), estado (disponible u ocupado), y fecha\_reserva. La clave foránea usuario\_id permite asociar una reserva específica a un usuario, lo que facilita el control de los espacios físicos de la biblioteca.
6. **Conducta\_usuario:** Esta entidad registra los eventos relacionados con la conducta de los usuarios, como puede ser una infracción. Los atributos incluyen id\_conducta (clave primaria), fecha\_evento, tipo\_evento, y descripcion. Esta tabla está vinculada con las tablas de Usuarios y Préstamos a través de las claves foráneas usuario\_id y prestamo\_id, permitiendo registrar y categorizar los eventos de conducta de los usuarios.
7. **Infracciones:** En esta tabla se registran las infracciones cometidas por los usuarios en relación con los préstamos. Los atributos incluyen id\_infracciones (clave primaria), dias\_retraso y multa. Las claves foráneas prestamo\_id y usuario\_id permiten asociar cada infracción con el usuario y el préstamo correspondiente, facilitando el cálculo de multas y el registro de sanciones.

Relaciones entre Entidades

Las relaciones entre las tablas están representadas mediante claves foráneas (FK), que permiten una correcta vinculación entre las entidades:

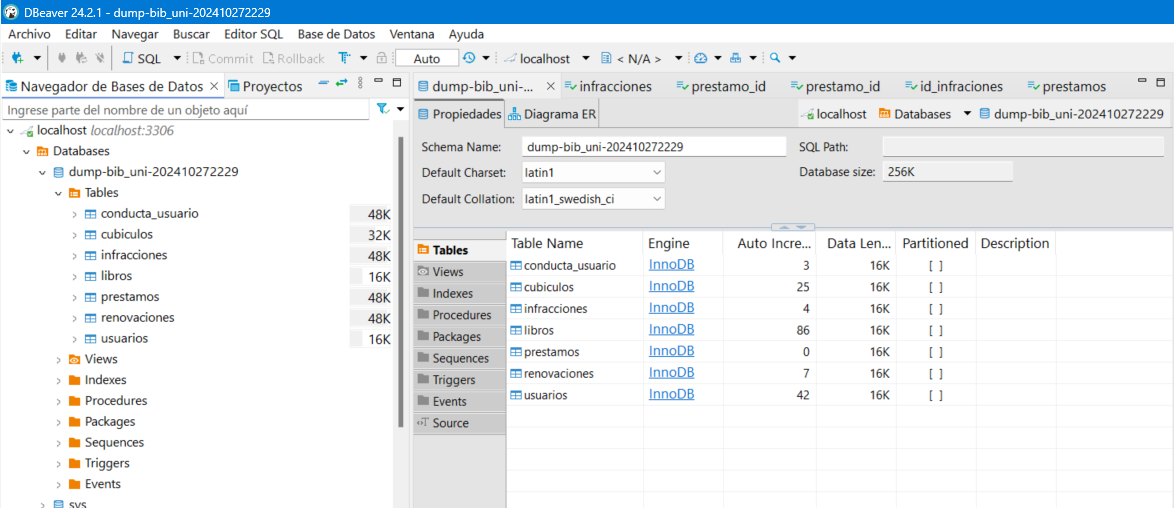
* La tabla de Préstamos está vinculada a Usuarios y Libros a través de las claves foráneas usuario\_id y libro\_id.
* Las tablas Conducta\_usuario, Renovaciones, y Infracciones están relacionadas con la tabla Préstamos y la tabla Usuarios, lo cual permite un seguimiento detallado de las acciones y sanciones de cada usuario.
* La tabla Cubículos se relaciona con Usuarios, permitiendo registrar las reservas de espacio hechas por los usuarios.

Este modelo facilita una gestión integral de los procesos de la biblioteca universitaria, permitiendo registrar, actualizar y analizar la información de manera eficiente. Además, ofrece soporte para la generación de reportes de conducta, préstamos, renovaciones y el estado de los recursos de la biblioteca.



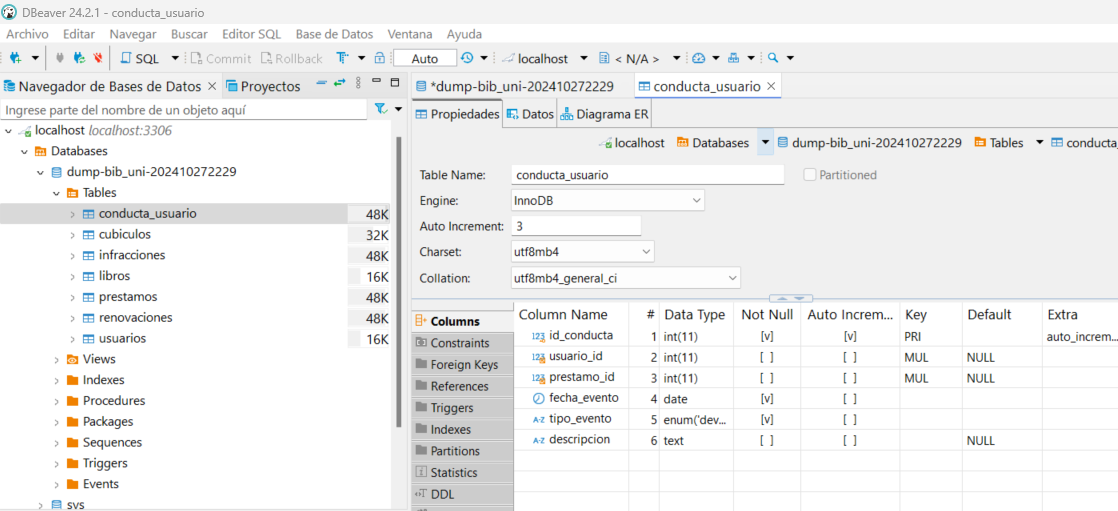
**CREACION DE BASE DE DATOS**

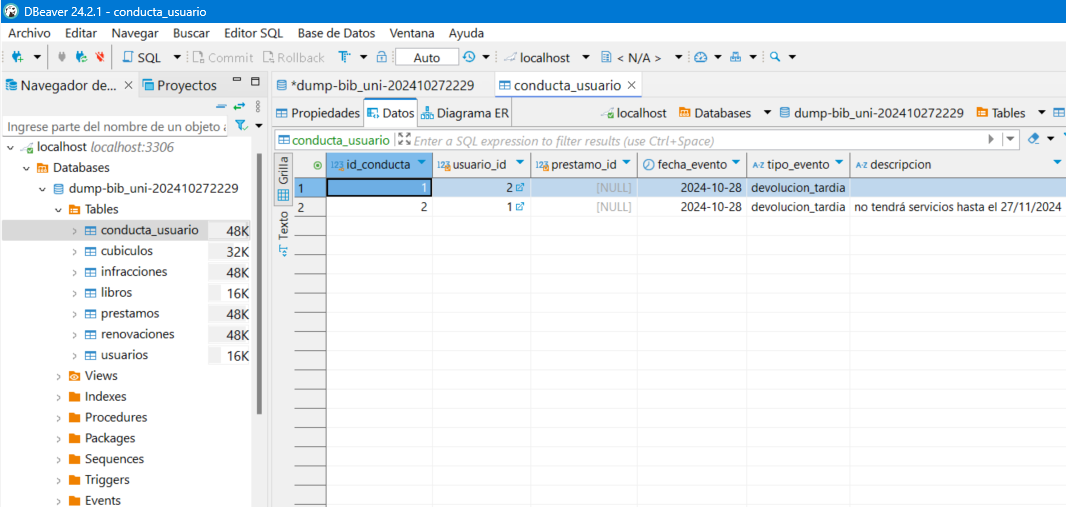
* Se crea todas las siguientes tablas:



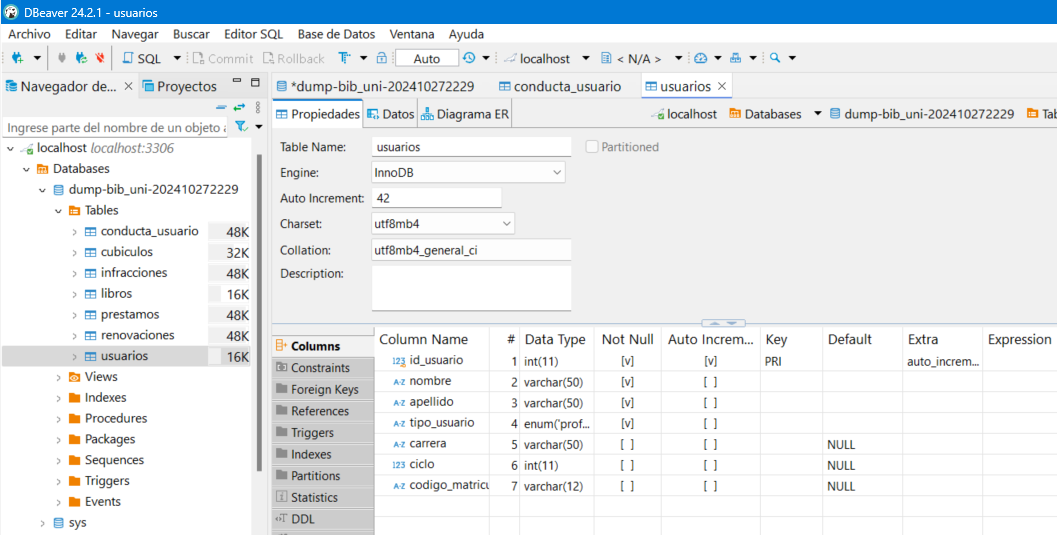
* Cada tabla tendrá las siguientes propiedades y sus respectivos datos

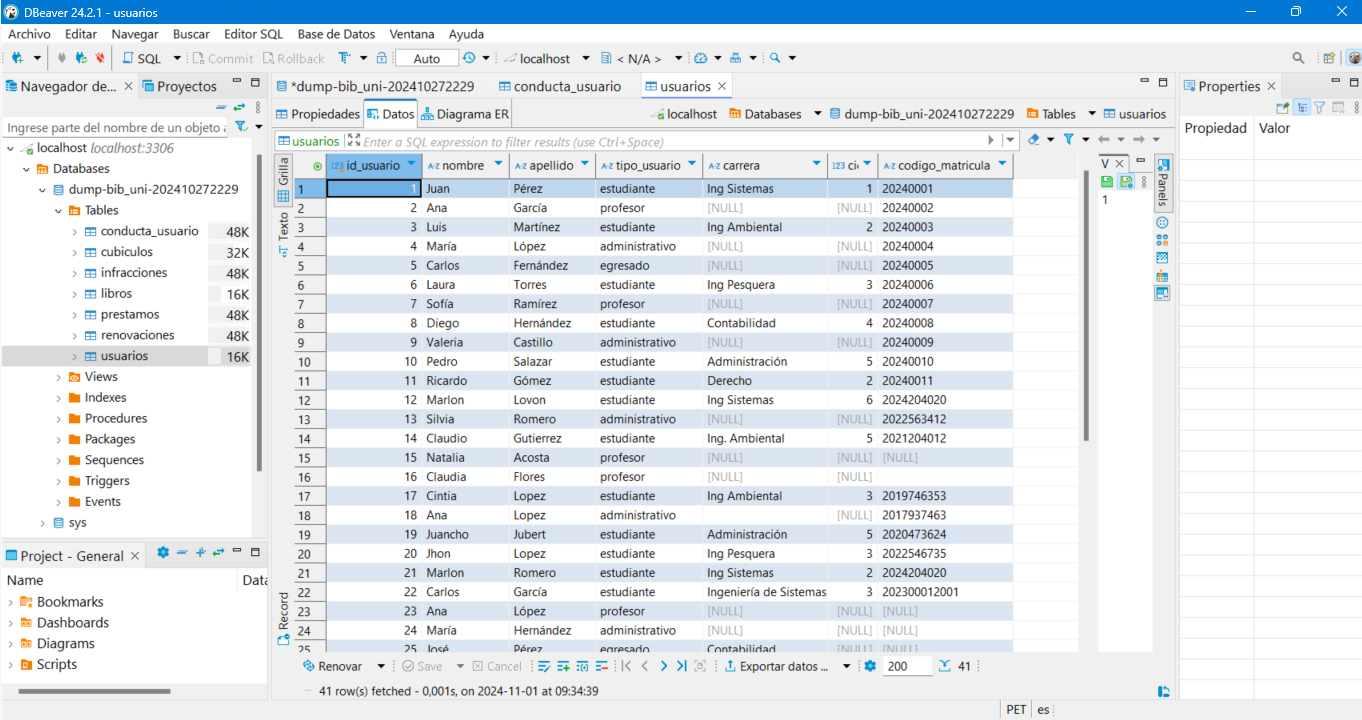
**CONDUCTA\_USUARIO**



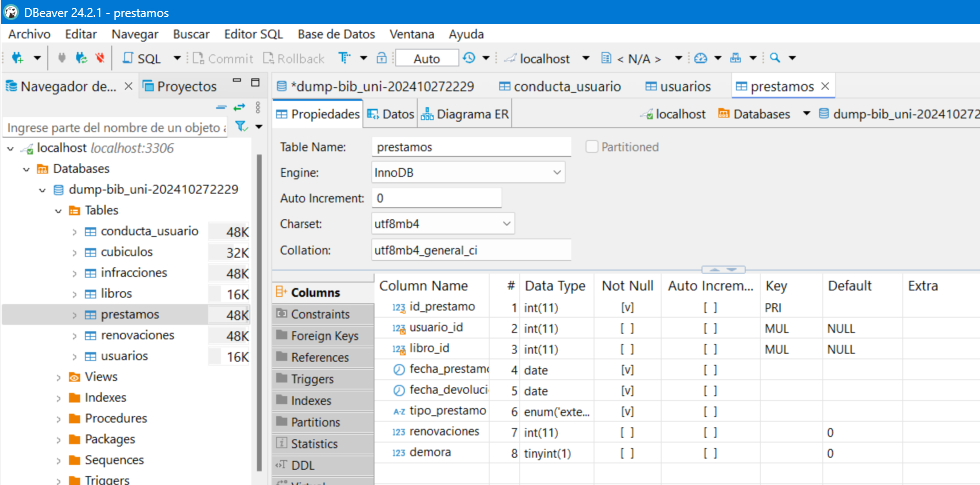


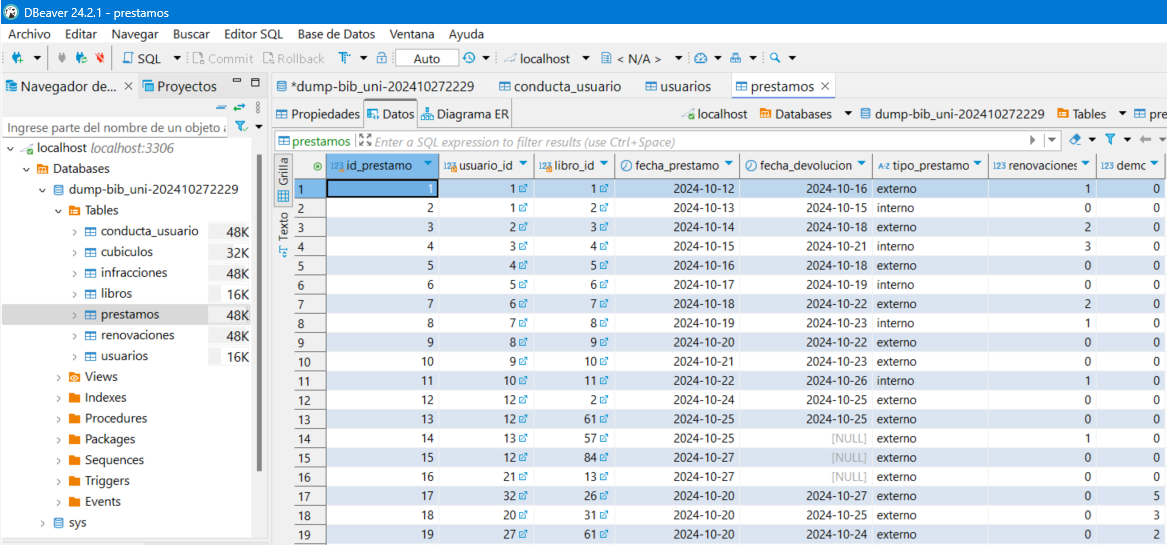
**USUARIOS**



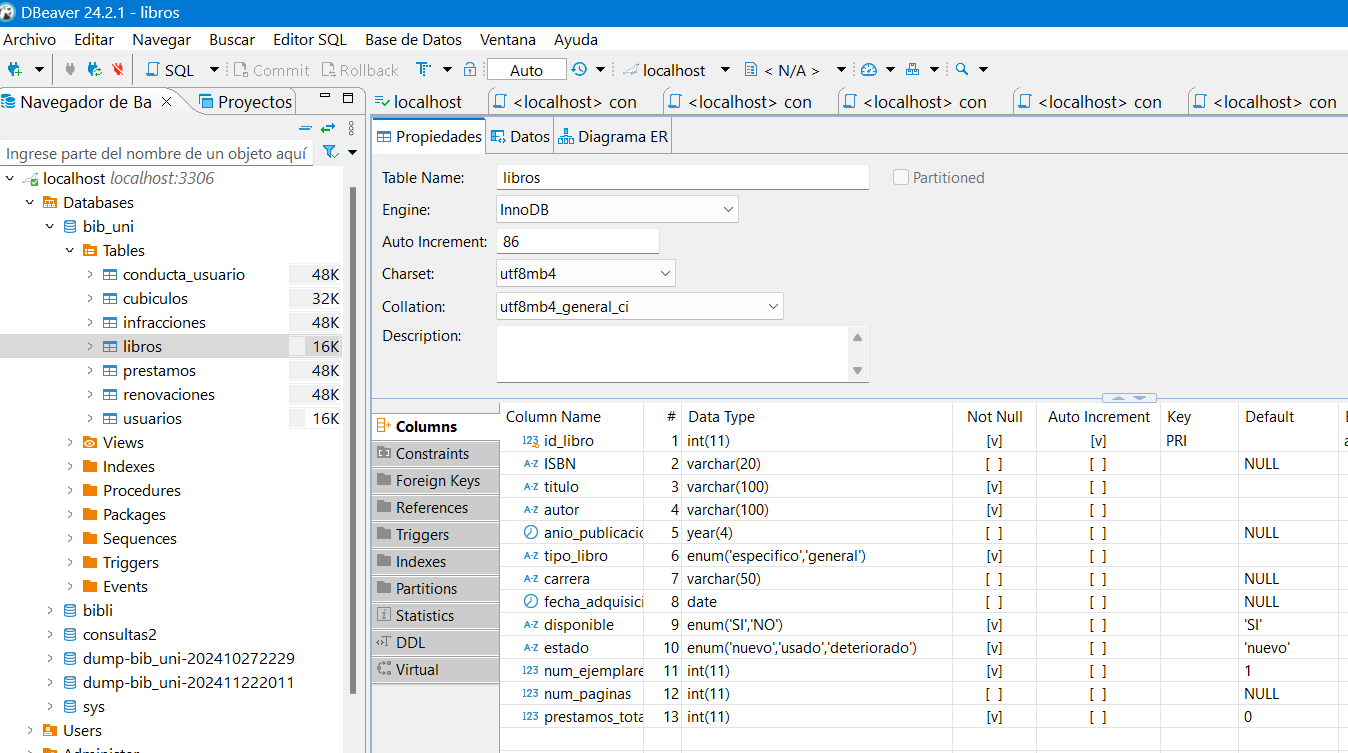


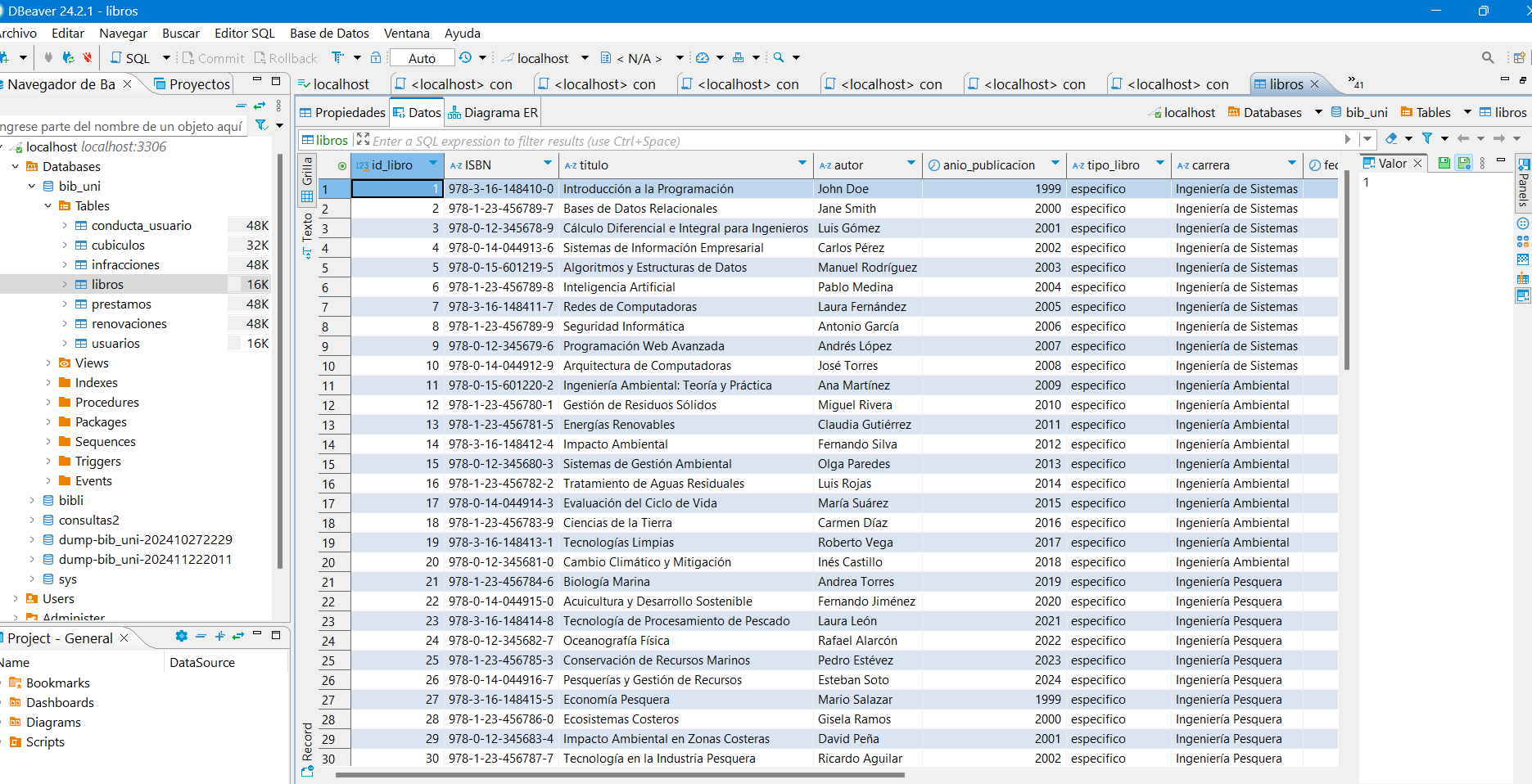
**PRESTAMOS**



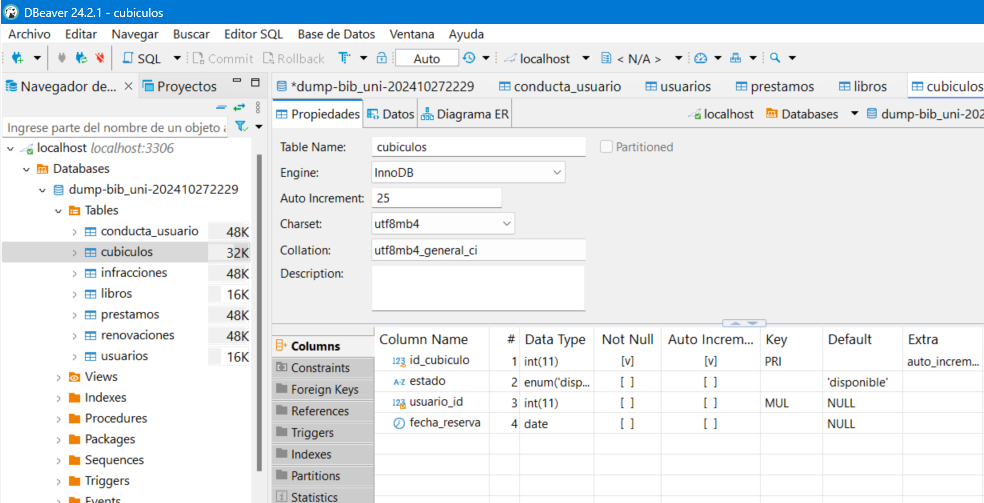


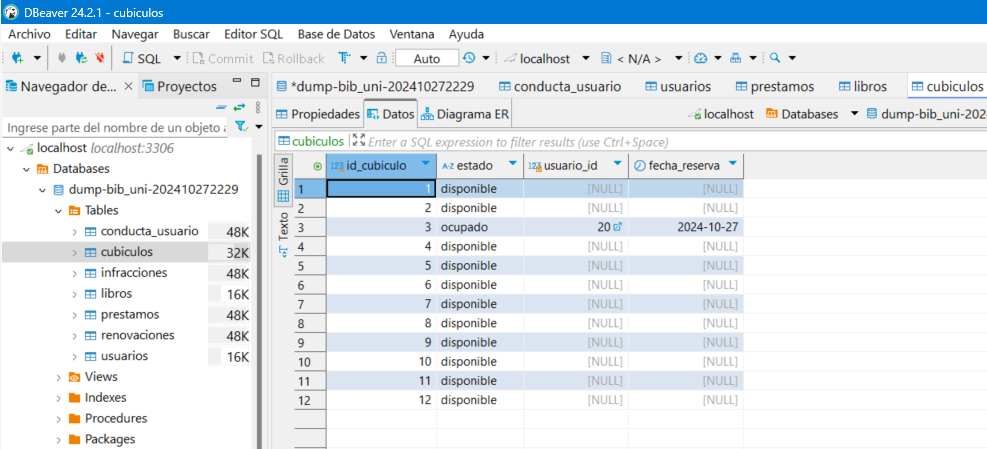
**LIBROS**

****

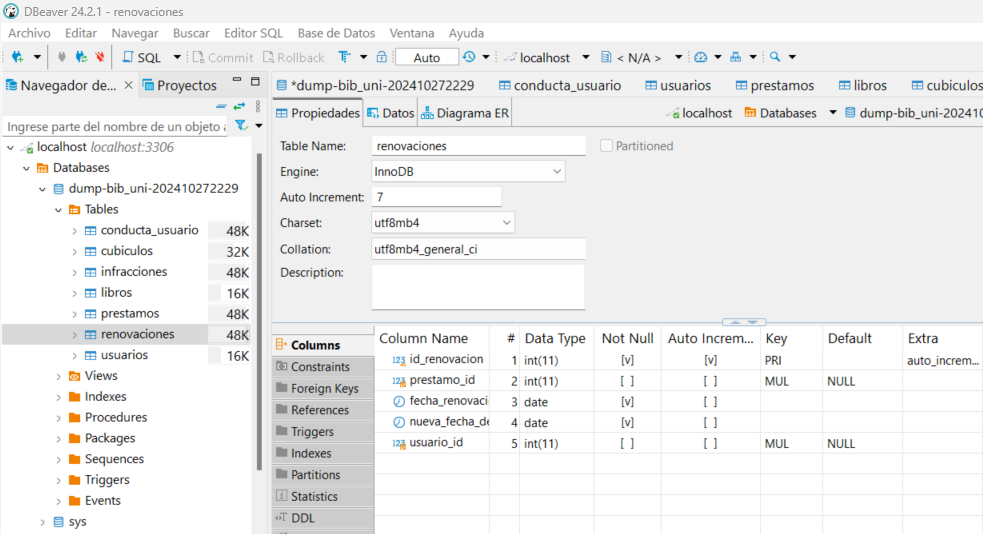
****

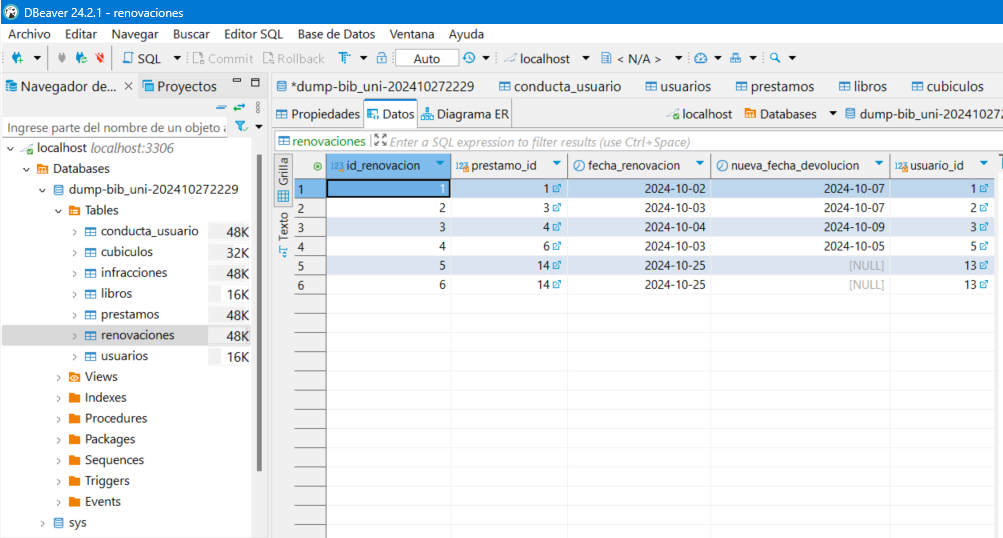
**CUBICULOS**

****

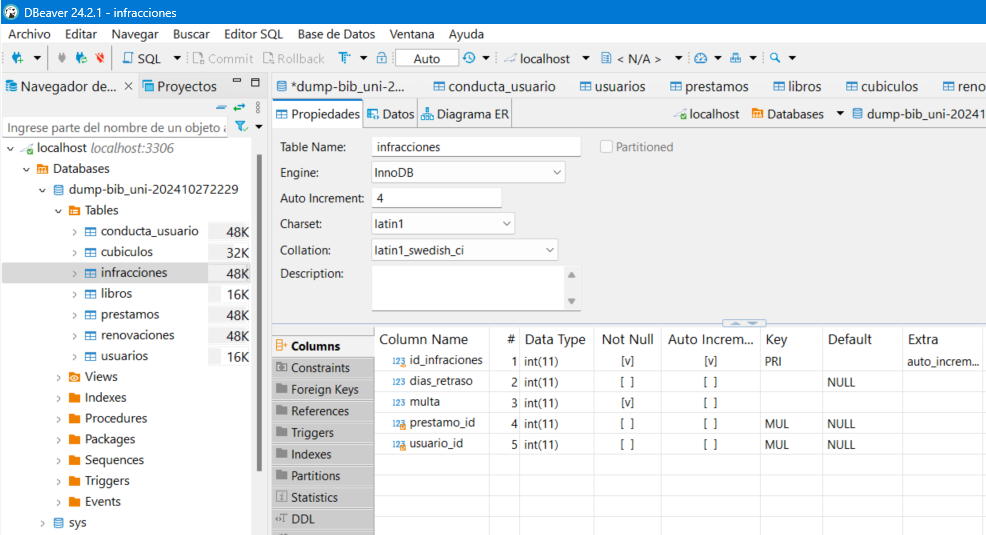
****

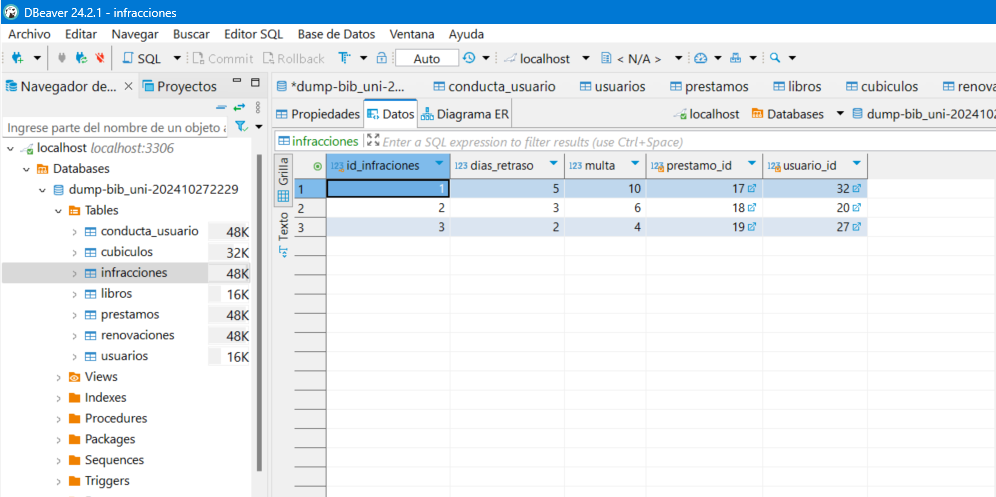
**RENOVACIONES**

****

****

**INFRACCIONES**

****

****

# PROCEDIMIENTOS ALMACENADOS Y TRIGGERS EN LA BASE DE DATOS DE BIBLIOTECA

## PROCEDIMIENTO: REGISTRARPRESTAMO

### Utilidad

Este procedimiento verifica si un libro está disponible antes de registrar un préstamo. Si el libro está disponible, se actualiza el estado del libro a "NO" y se registra el préstamo. Si no está disponible, devuelve un error indicando que no es posible realizar el préstamo.

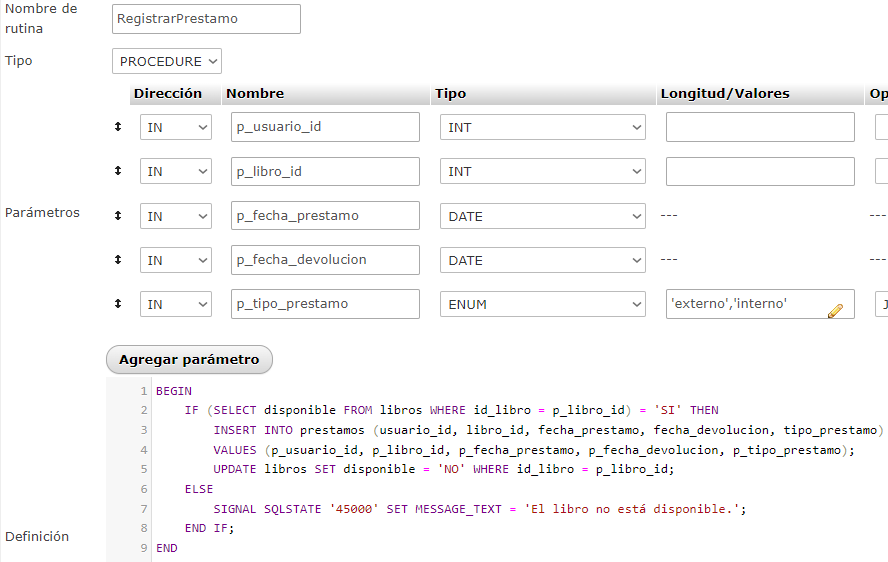
### Cómo Ejecutarlo

CALL RegistrarPrestamo(p\_usuario\_id, p\_libro\_id, p\_fecha\_prestamo, p\_fecha\_devolucion, p\_tipo\_prestamo);

### Parámetros

- `p\_usuario\_id`: ID del usuario que realiza el préstamo.  
- `p\_libro\_id`: ID del libro a prestar.  
- `p\_fecha\_prestamo`: Fecha del préstamo.  
- `p\_fecha\_devolucion`: Fecha de devolución esperada.  
- `p\_tipo\_prestamo`: Tipo de préstamo (`externo` o `interno`).

### Código:



## PROCEDIMIENTO: REGISTRARDEVOLUCION

### Utilidad

Actualiza el estado del libro a "SI" al devolverlo y registra si hubo una demora. Si la devolución es tardía, registra una infracción en la tabla correspondiente con los días de retraso y la multa correspondiente.

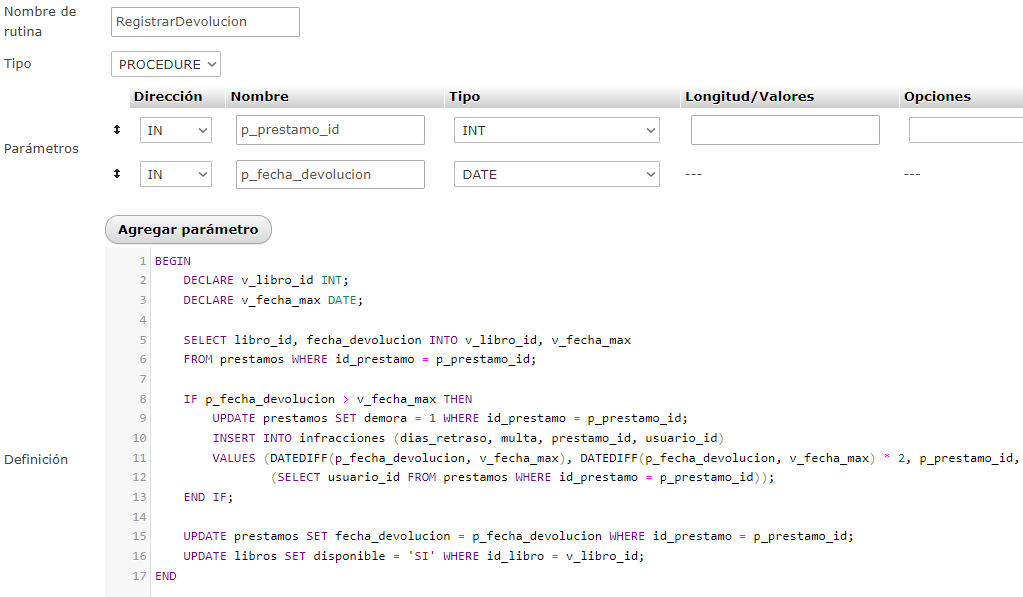
### Cómo Ejecutarlo

CALL RegistrarDevolucion(p\_prestamo\_id, p\_fecha\_devolucion);

### Parámetros

- `p\_prestamo\_id`: ID del préstamo a devolver.  
- `p\_fecha\_devolucion`: Fecha efectiva de devolución.

**Código:**

****

## PROCEDIMIENTO: REGISTRARRENOVACION

### Utilidad

Este procedimiento extiende la fecha de devolución de un préstamo y registra la renovación en una tabla específica para mantener un historial.

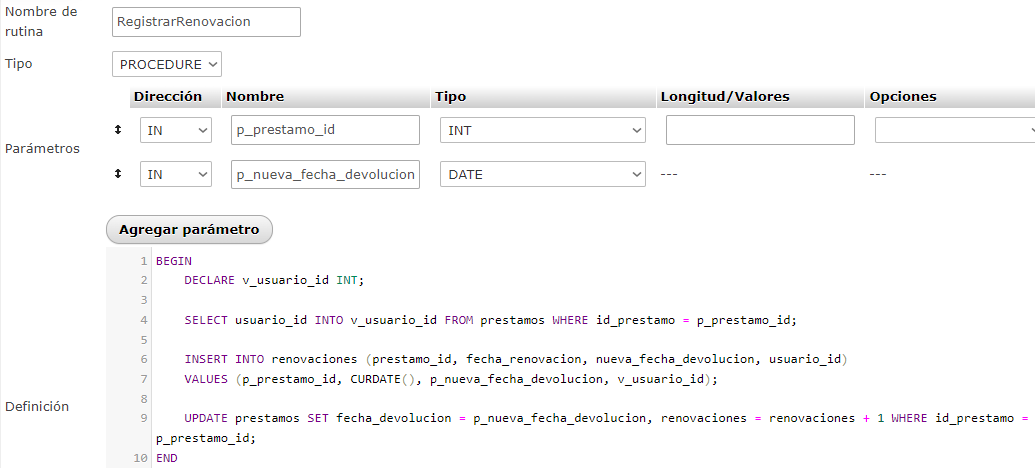
### Cómo Ejecutarlo

CALL RegistrarRenovacion(p\_prestamo\_id, p\_nueva\_fecha\_devolucion);

### Parámetros

- `p\_prestamo\_id`: ID del préstamo a renovar.  
- `p\_nueva\_fecha\_devolucion`: Nueva fecha de devolución.

**Código:**



## TRIGGER: ACTUALIZARESTADOCUBICULO

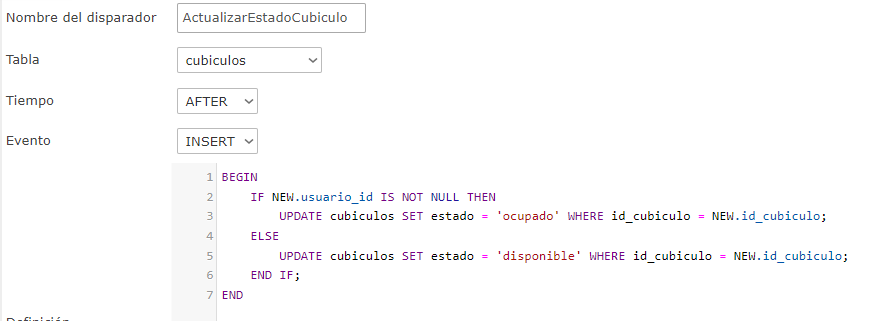
### Utilidad

Este trigger actualiza el estado de un cubículo a "ocupado" o "disponible" dependiendo de si hay un usuario asociado al registro.

### Cómo Ejecutarlo

Este trigger se activa automáticamente al insertar o actualizar un registro en la tabla `cubiculos`. No requiere una llamada manual.

**Código:**

****

## TRIGGER: REGISTRARCONDUCTAUSUARIO

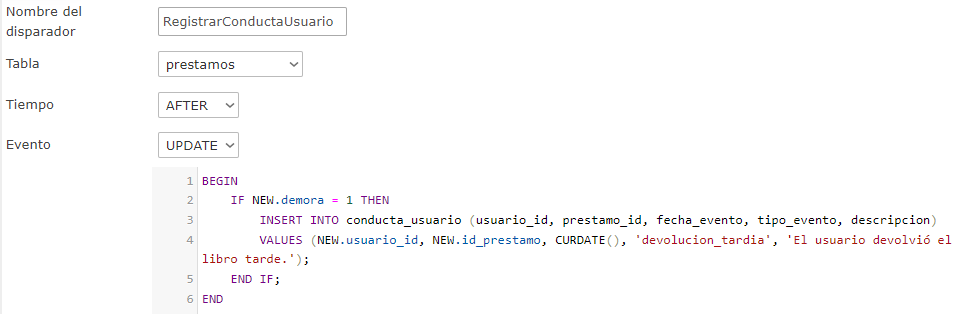
### Utilidad

Inserta un registro en la tabla `conducta\_usuario` si un usuario devuelve un libro tarde. Registra el evento con detalles como la fecha y descripción.

### Cómo Ejecutarlo

Este trigger se activa automáticamente al actualizar un registro en la tabla `prestamos`. No requiere una llamada manual.

**Código:**

****

## TRIGGER: INCREMENTARPRESTAMOSTOTALES

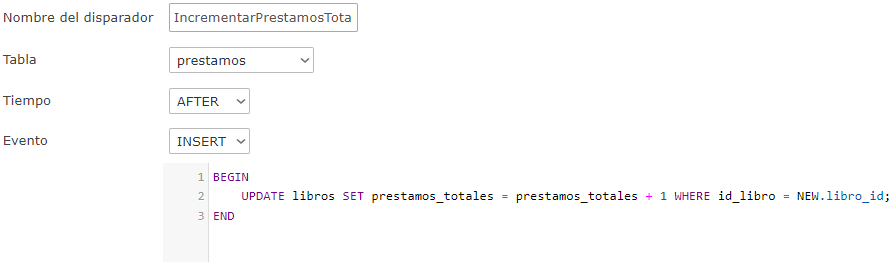
### Utilidad

Incrementa el contador de préstamos totales de un libro cada vez que se registra un nuevo préstamo.

### Cómo Ejecutarlo

Este trigger se activa automáticamente al insertar un registro en la tabla `prestamos`. No requiere una llamada manual.

**Código:**

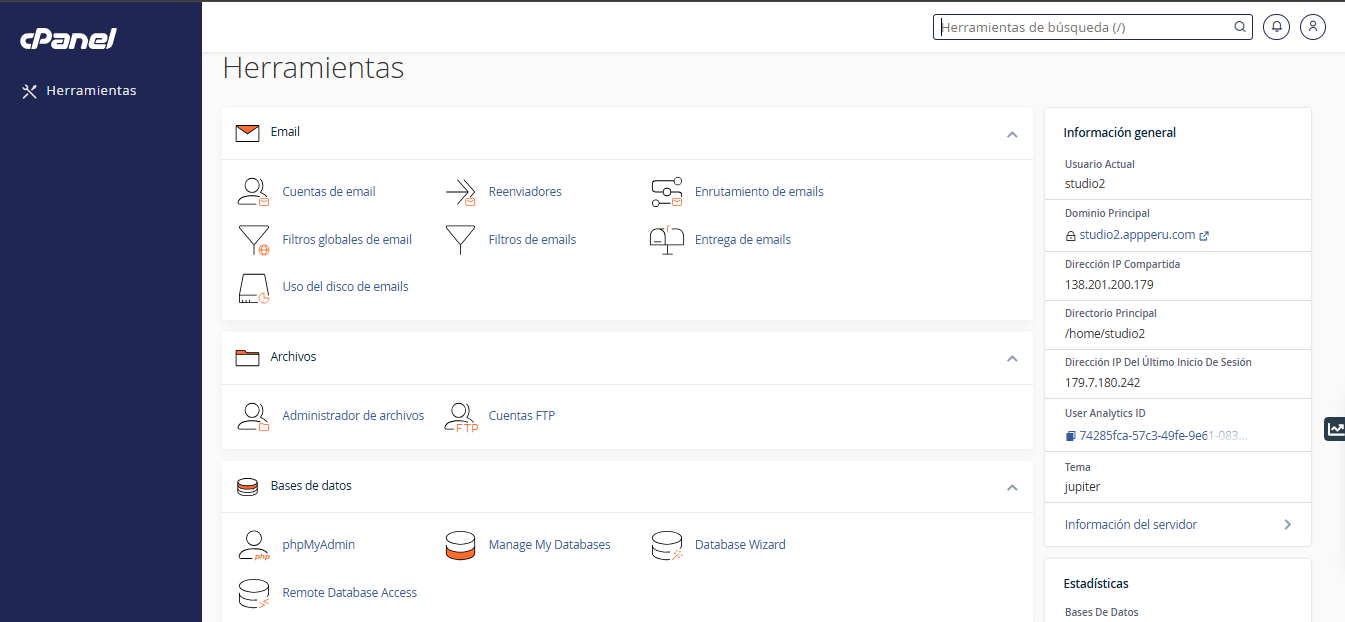
****

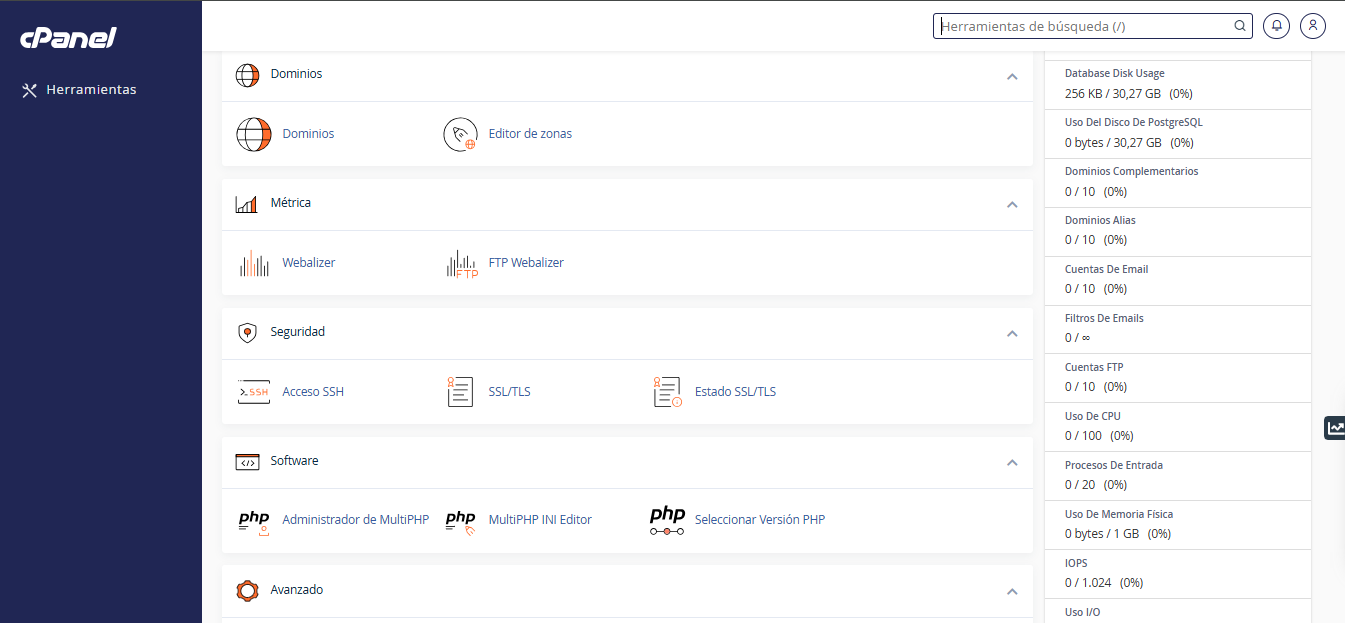
**PUBLICACION EN LA WEB**

**CPANEL:**

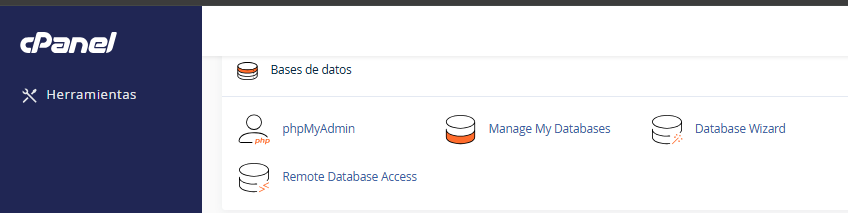
CPanel es un software para la gestión de [servidores](https://www.arsys.es/servidores). Consiste en una aplicación web que puedes instalar en tus propios servidores y que permite la creación y configuración de todos los recursos que debes gestionar en una máquina. Entre muchas de sus posibilidades, permite la creación de [dominios](https://www.arsys.es/dominios), la gestión de los recursos asociados a los dominios, las [bases de datos](https://www.arsys.es/blog/tipos-de-bases-de-datos-que-existen), los accesos por [FTP](https://www.arsys.es/soporte/hosting-web/ftp/que-es-ftp), etc.

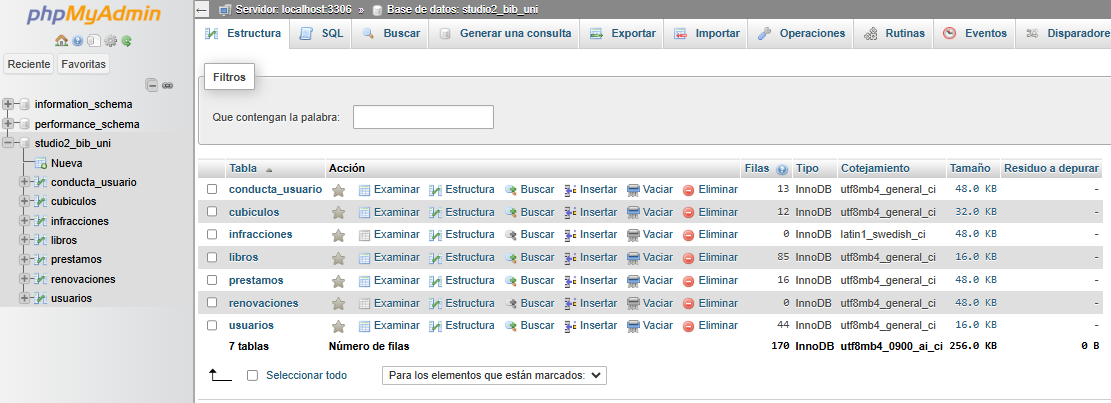
Además de los recursos, el software permite la gestión asociada a distintos perfiles de usuario, de modo que podríamos montar literalmente un servicio de [hosting](https://www.arsys.es/hosting) o de dominios donde los clientes también tendrían la posibilidad de administrar sus propios servicios.



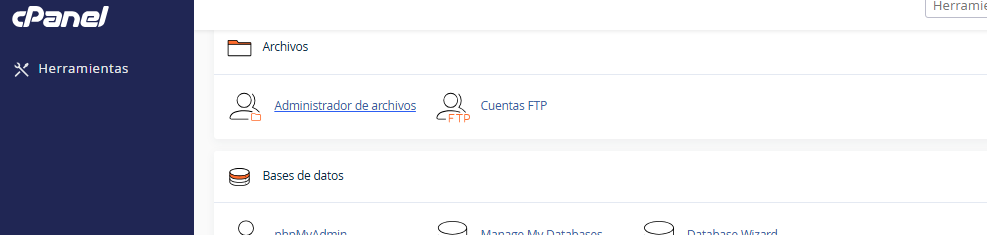


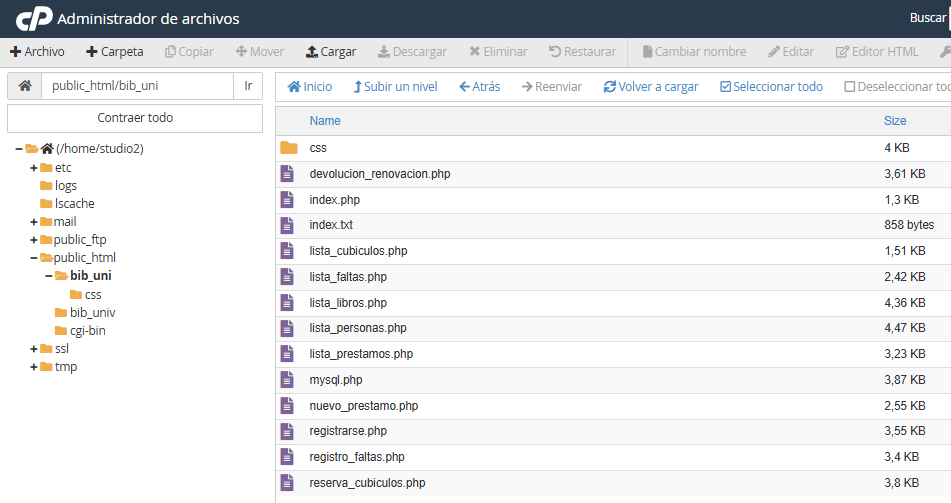
* Subiremos la base de datos nos dirigimos phpMyAdmin de Cpanel

****

****

* Luego nos entramos a la opción de administración de archivos y en la carpeta public\_html subiremos los códigos php y css para su publicación.

****

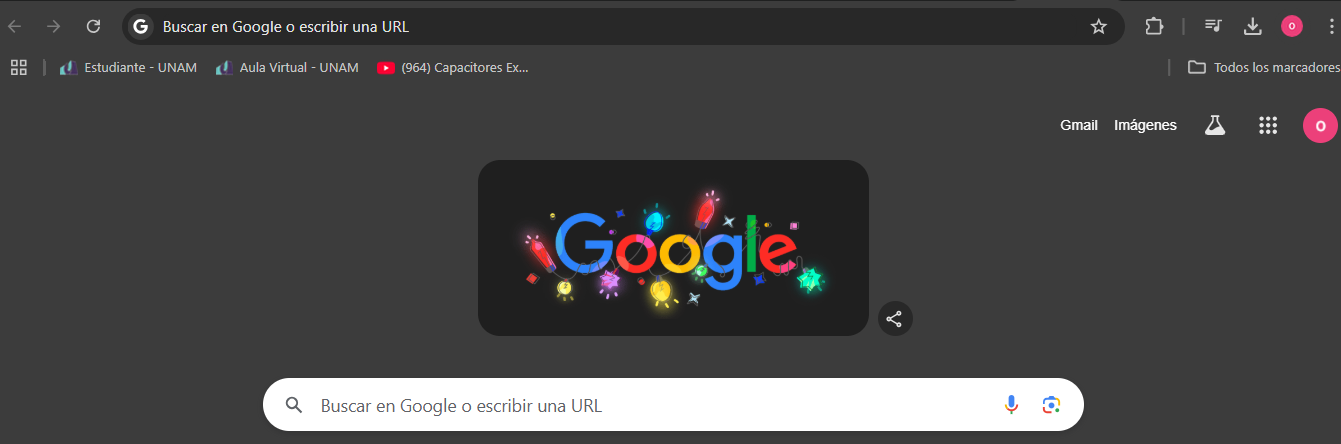
****

* La conexión entre la base de datos y la publicación hecha tendrá la siguiente codificación.



* Ahora nos dirigimos a una pagina web vacía y ingresamos la siguiente dirección.

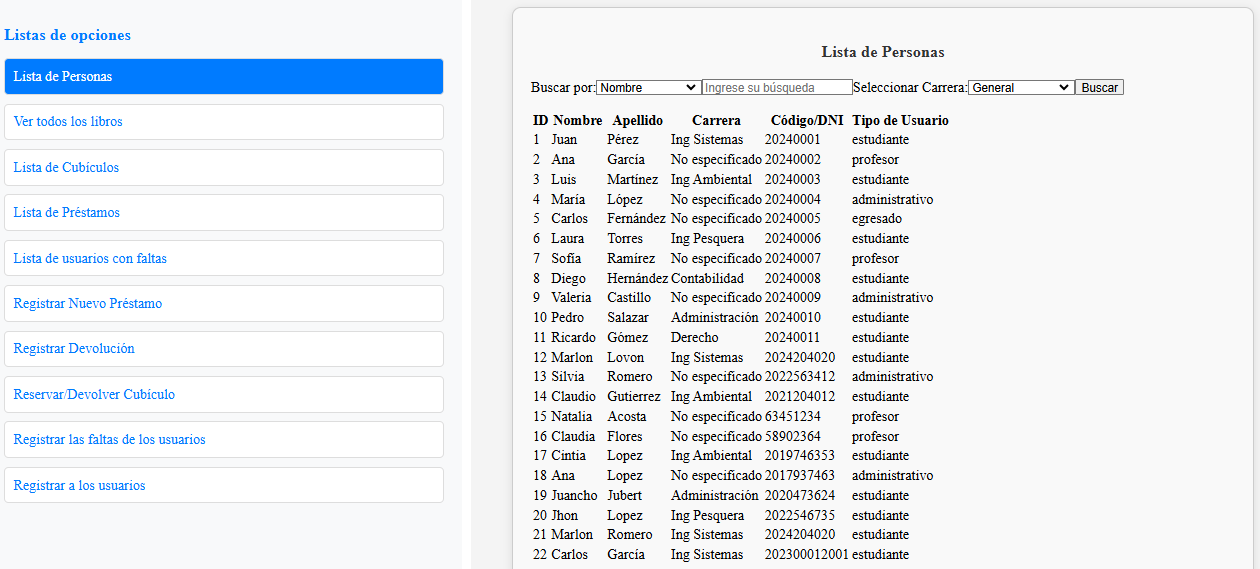
[**https://server99.intecdev.com:2083//bib\_uni/**](https://server99.intecdev.com:2083/bib_uni/)

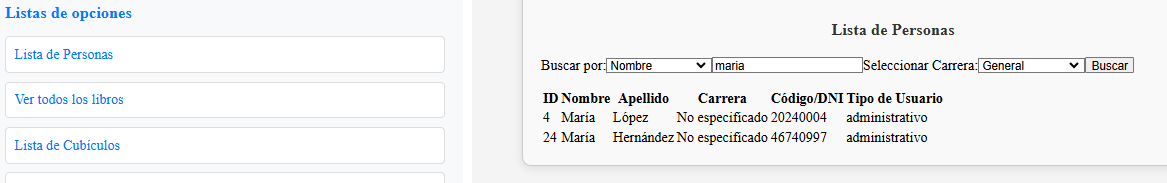
****

**DESARROLLO EN EDITOR VISUAL STUDIO CODE**

* Desarrollo completo del entorno mysql.php para la conexión.
* <?php
* ini\_set('display\_errors', 1);
* /\*
* Nombre: heasy\_mysql
* Versión: 1.2
* Autor del Script: Luis Torres
* Correo: luis.torres.carpio1@gmail.com
* Descripción:
* Script para facilitar las querys en Mysql desde PHP
* Pensado para el desarrollo agil.
* \*/
* class conexion{
* private $server;
* private $user;
* private $clave;
* private $db;
* private $port;
* public $conex;
* function \_\_construct()
* {
* $this->server="127.0.0.1";
* $this->user="studio2";
* $this->clave=" HA3hdvG12G";
* $this->db="studio2\_bib\_uni";
* }
* public function conectar()
* {
* $this->conex=new mysqli($this->server, $this->user, $this->clave, $this->db);
* if ($this->conex->connect\_error) {
* die('Error de conexión (' .$this->conex->connect\_errno . ') '
* .$this->conex->connect\_error);
* }
* $this->conex->set\_charset("utf8");//save accent db
* }
* public function cerrar()
* {
* $this->conex->close();
* }
* }
* $GLOBALS['mc']=new conexion();
* function query\_multiple($query){
* $mc=new conexion();
* $mc->conectar();
* $ok\_affected = [];
* if( $rsta = $mc->conex->multi\_query($consulta)  === TRUE ){
* do {
* $row=$mc->conex->affected\_rows;
* array\_push($ok\_affected, $row);
* if (!$mc->conex->more\_results()) {
* break;
* }
* } while ($mc->conex->next\_result());
* $mc->cerrar();
* }else{
* echo "Error: " . $mc->conex->error;
* }
* return $ok\_affected;
* }
* function query\_exec($query){
* $mc = new conexion();
* $mc->conectar();
* $fecth\_rows = 0;
* $mc->conex->multi\_query($query);
* do {
* if ($result = $mc->conex->store\_result()) {
* $fecth\_rows =  $result->num\_rows;
* }
* }while ($mc->conex->next\_result()); // flush multi\_queries
* $mc->cerrar();
* return $fecth\_rows;
* }
* function query\_start($query){
* $GLOBALS['mc']->conectar();
* return $GLOBALS['mc']->conex->query($query);
* }
* function query($query){
* $type = array(
* "INSERT INTO",
* "INSERT",
* "UPDATE",
* "SELECT \* FROM",
* "SELECT",
* "DELETE FROM",
* "CREATE TABLE",
* "CALL",
* );
* $running\_case = false;
* for($i=0;$i<count($type);$i++){
* if (strpos($query, $type[$i]) !== false && $running\_case == false){
* switch ($type[$i]) {
* case 'CALL':
* $running\_case = true;
* if(!$results = query\_start($query)){
* return ["error"=>$GLOBALS['mc']->conex->error];
* }
* if(strrpos($query, "select\_")){
* return select($results);
* }else if(strrpos($query, "update\_")){
* return 1;
* }else if(strrpos($query, "delete\_")){
* return 1;
* }else if(strrpos($query, "insert\_")){
* $id=$GLOBALS['mc']->conex->insert\_id;
* return $id;
* }
* break;
* case 'INSERT INTO':
* $running\_case = true;
* if( !query\_start($query)){
* return ["error"=>$GLOBALS['mc']->conex->error." - Query: " . $query];
* }
* $id=$GLOBALS['mc']->conex->insert\_id;
* return $id;
* break;
* case 'UPDATE':
* $running\_case = true;
* if( !query\_start($query))
* return ["error"=>$GLOBALS['mc']->conex->error];
* else
* return 1;
* break;
* case 'SELECT \* FROM':
* $running\_case = true;
* if(!$results = query\_start($query))
* return ["error"=>$GLOBALS['mc']->conex->error];
* return select($results);
* break;
* case 'SELECT':
* $running\_case = true;
* if(!$results = query\_start($query))
* return ["error"=>$GLOBALS['mc']->conex->error];
* return select($results);
* break;
* case 'DELETE FROM':
* $running\_case = true;
* if( !query\_start($query))
* return ["error"=>$GLOBALS['mc']->conex->error];
* else
* return 1;
* break;
* }
* $GLOBALS['mc']->cerrar();
* }
* }
* }
* function select($results){
* if(!$results) return [];
* if($results->num\_rows>0){
* while( $rr = mysqli\_fetch\_assoc($results) ) $rows[] = $rr; //return array
* return $rows;
* }else
* return [];
* }
* ?>
* Veremos como se presenta la ventana de la pagina web asi como el desarrollo de el código php utilizado

**LISTA DE PERSONAS**

****

****

<?php

require("mysql.php");

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/lis\_personas.css">';

echo "<div class='container'>";

echo "<h3>Lista de Personas</h3>";

// Formulario de búsqueda

echo "<form method='GET' action=''>";

echo "<label for='criterio'>Buscar por:</label>";

echo "<select name='criterio' id='criterio'>";

echo "<option value='nombre' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'nombre' ? 'selected' : '') . ">Nombre</option>";

echo "<option value='apellido' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'apellido' ? 'selected' : '') . ">Apellido</option>";

echo "<option value='carrera' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'carrera' ? 'selected' : '') . ">Carrera</option>";

echo "<option value='codigo\_o\_dni' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'codigo\_o\_dni' ? 'selected' : '') . ">Código/DNI</option>";

echo "<option value='tipo\_usuario' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'tipo\_usuario' ? 'selected' : '') . ">Tipo de Usuario</option>";

echo "</select>";

// Campo de texto para buscar

echo "<input type='text' name='buscar' placeholder='Ingrese su búsqueda' value='" . (isset($\_GET['buscar']) ? $\_GET['buscar'] : '') . "' />";

// Lista de carreras para filtrar por carrera específica

echo "<label for='carrera'>Seleccionar Carrera:</label>";

echo "<select name='carrera' id='carrera'>";

echo "<option value=''>General</option>";

echo "<option value='Ing Sistemas' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'Ing Sistemas' ? 'selected' : '') . ">Ing Sistemas</option>";

echo "<option value='Ing Ambiental' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'Ing Ambiental' ? 'selected' : '') . ">Ing Ambiental</option>";

echo "<option value='Ing Pesquera' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'Ing Pesquera' ? 'selected' : '') . ">Ing Pesquera</option>";

echo "<option value='Contabilidad' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'Contabilidad' ? 'selected' : '') . ">Contabilidad</option>";

echo "<option value='Administración' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'Administración' ? 'selected' : '') . ">Administración</option>";

echo "<option value='Derecho' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'Derecho' ? 'selected' : '') . ">Derecho</option>";

echo "<option value='No especificado' " . (isset($\_GET['carrera']) && $\_GET['carrera'] == 'No especificado' ? 'selected' : '') . ">No especificado</option>";

echo "</select>";

// Botón de búsqueda

echo "<input type='submit' value='Buscar' />";

echo "</form>";

// Si se ha enviado el formulario

$criterio = isset($\_GET['criterio']) ? $\_GET['criterio'] : '';

$buscar = isset($\_GET['buscar']) ? $\_GET['buscar'] : '';

$carrera = isset($\_GET['carrera']) ? $\_GET['carrera'] : '';

// Condición base para la consulta SQL

$where = "WHERE 1";

// Si se proporciona un término de búsqueda y un criterio, se añade a la consulta

if ($criterio != '' && $buscar != '') {

    if ($criterio == 'nombre' || $criterio == 'apellido' || $criterio == 'codigo\_o\_dni' || $criterio == 'tipo\_usuario') {

        $where .= " AND $criterio LIKE '%$buscar%'";

    }

}

// Si se selecciona una carrera, se añade a la consulta

if ($carrera != '') {

    if ($carrera == 'No especificado') {

        // Si se selecciona "No especificado", buscar por carrera NULL

        $where .= " AND (carrera IS NULL OR carrera = '')";

    } else {

        $where .= " AND carrera = '$carrera'";

    }

}

// Consulta para obtener los usuarios con la búsqueda aplicada

$query = "SELECT \* FROM usuarios $where ORDER BY id\_usuario ASC";

$rows = query($query);

// Mostrar los resultados de la búsqueda

if (count($rows) > 0) {

    echo "<table class='tabla-personas'>";

    echo "<tr><th>ID</th><th>Nombre</th><th>Apellido</th><th>Carrera</th><th>Código/DNI</th><th>Tipo de Usuario</th></tr>";

    foreach ($rows as $row) {

        echo "<tr>";

        echo "<td>" . $row["id\_usuario"] . "</td>";

        echo "<td>" . $row["nombre"] . "</td>";

        echo "<td>" . $row["apellido"] . "</td>";

        echo "<td>" . ($row["carrera"] ? $row["carrera"] : "No especificado") . "</td>";

        echo "<td>" . $row["codigo\_o\_dni"] . "</td>";

        echo "<td>" . $row["tipo\_usuario"] . "</td>";

        echo "</tr>";

    }

    echo "</table>";

} else {

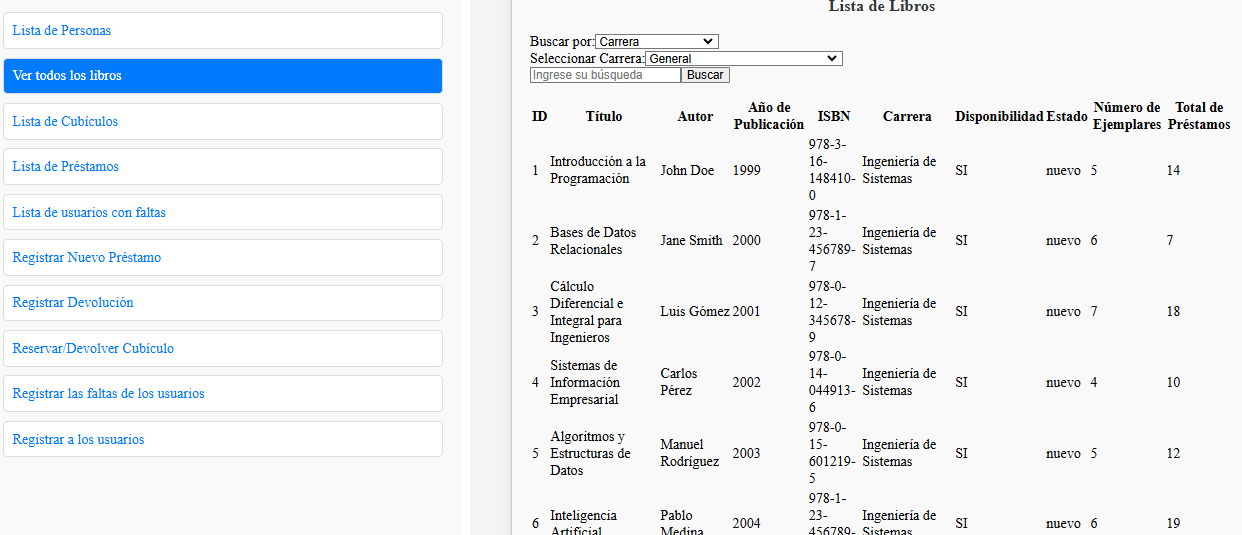
    echo "<p>No se encontraron resultados para su búsqueda.</p>";

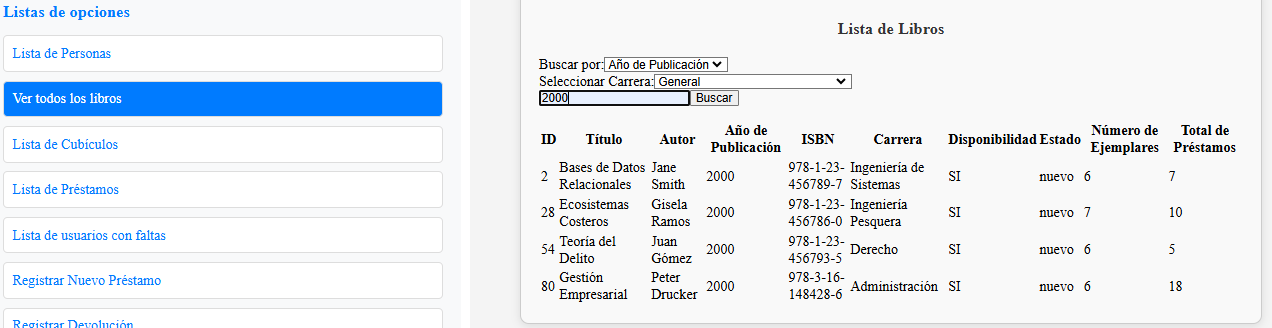
}

echo "</div>"; // Cierra el contenedor

?>

**LIBROS**

****

****

<?php

// Incluir el archivo mysql.php

require("mysql.php");

// Crear una instancia de la clase conexion

$db = new conexion();

$db->conectar(); // Establecer la conexión

$conn = $db->conex; // Acceder al objeto de conexión

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/lis\_libros.css">';

echo "<div class='container'>";

echo "<h3>Lista de Libros</h3>";

// Formulario de búsqueda

echo "<form method='GET' action=''>";

echo "<label for='criterio'>Buscar por:</label>";

echo "<select name='criterio' id='criterio'>";

echo "<option value='carrera'>Carrera</option>";

echo "<option value='titulo'>Título</option>";

echo "<option value='autor'>Autor</option>";

echo "<option value='anio\_publicacion'>Año de Publicación</option>";

echo "<option value='isbn'>ISBN</option>";

echo "</select>";

echo "<div id='carrera-div'>";

echo "<label for='carrera'>Seleccionar Carrera:</label>";

echo "<select name='carrera' id='carrera'>";

echo "<option value=''>General</option>";

echo "<option value='Ingeniería Pesquera'>Ingeniería Pesquera</option>";

echo "<option value='Contabilidad'>Contabilidad</option>";

echo "<option value='Administración'>Administración</option>";

echo "<option value='Derecho'>Derecho</option>";

echo "<option value='Ingeniería Sistemas e Informática'>Ingeniería Sistemas e Informática</option>";

echo "<option value='Ingeniería Ambiental'>Ingeniería Ambiental</option>";

echo "<option value='NULL'>Ninguna Específica</option>";

echo "</select>";

echo "</div>";

echo "<input type='text' name='buscar' placeholder='Ingrese su búsqueda' />";

echo "<input type='submit' value='Buscar' />";

echo "</form>";

// Obtener los valores de los parámetros de búsqueda

$criterio = isset($\_GET['criterio']) ? $\_GET['criterio'] : '';

$buscar = isset($\_GET['buscar']) ? $\_GET['buscar'] : '';

$carrera = isset($\_GET['carrera']) ? $\_GET['carrera'] : '';

// Condición base para la consulta SQL

$where = "WHERE 1";

// Si se proporciona un término de búsqueda y un criterio, se añade a la consulta

if ($criterio != '' && $buscar != '') {

    $where .= " AND $criterio LIKE ?";

}

// Si se selecciona una carrera, se añade a la consulta

if ($carrera != '') {

    if ($carrera == 'NULL') {

        $where .= " AND carrera IS NULL"; // Para los libros con carrera NULL

    } else {

        $where .= " AND carrera = ?"; // Para la carrera seleccionada

    }

}

// Consulta SQL utilizando parámetros preparados

$query = "SELECT \* FROM libros $where";

$stmt = $conn->prepare($query);

// Vincular los parámetros

if ($criterio != '' && $buscar != '') {

    $param = '%' . $buscar . '%';

    $stmt->bind\_param('s', $param); // Usamos 's' para indicar que es una cadena

}

if ($carrera != '') {

    if ($carrera != 'NULL') {

        $stmt->bind\_param('s', $carrera); // Bind para la carrera seleccionada

    }

}

// Ejecutar la consulta

$stmt->execute();

// Obtener los resultados

$result = $stmt->get\_result();

// Mostrar los resultados de la búsqueda

if ($result->num\_rows > 0) {

    echo "<table class='tabla-libros'>";

    echo "<thead>";

    echo "<tr><th>ID</th><th>Título</th><th>Autor</th><th>Año de Publicación</th><th>ISBN</th><th>Carrera</th><th>Disponibilidad</th><th>Estado</th><th>Número de Ejemplares</th><th>Total de Préstamos</th></tr>";

    echo "</thead><tbody>";

    while ($row = $result->fetch\_assoc()) {

        echo "<tr>";

        echo "<td>" . htmlspecialchars($row["id\_libro"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["titulo"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["autor"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["anio\_publicacion"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["ISBN"]) . "</td>";

        echo "<td>" . ($row["carrera"] ? htmlspecialchars($row["carrera"]) : "General") . "</td>";

        echo "<td>" . htmlspecialchars($row["disponible"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["estado"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["num\_ejemplares"]) . "</td>";

        echo "<td>" . htmlspecialchars($row["prestamos\_totales"]) . "</td>";

        echo "</tr>";

    }

    echo "</tbody></table>";

} else {

    echo "<p>No se encontraron resultados para su búsqueda.</p>";

}

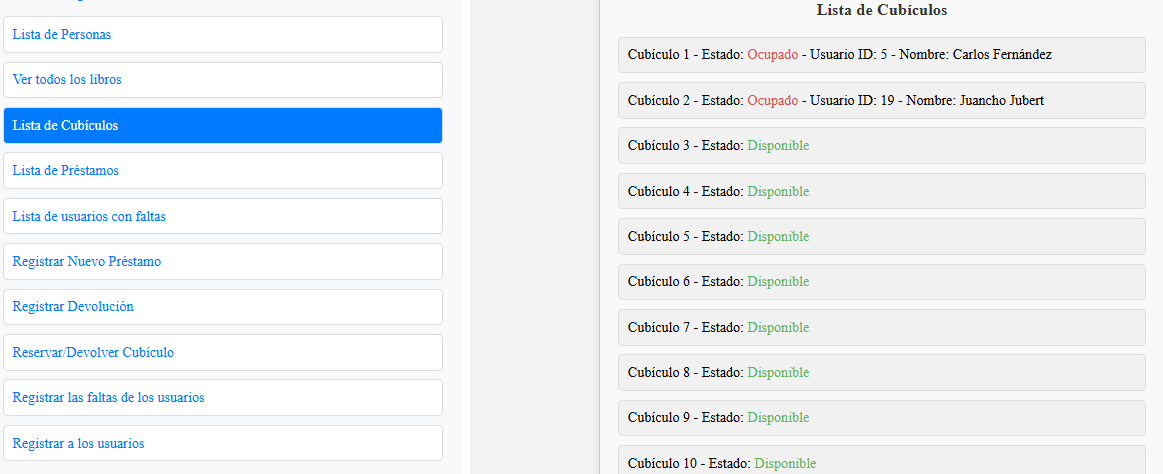
echo "</div>"; // Cierra el contenedor

// Cerrar la conexión

$db->cerrar();

?>

**CUBICULOS**

****

<?php

// Asegurarse de incluir mysql.php

require\_once("mysql.php");

echo "<div class='container'>";

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/lis\_cubiculos.css">';

echo "<h3>Lista de Cubículos</h3>";

// Consulta para obtener todos los cubículos junto con el usuario que los ocupa

$consulta = "SELECT c.id\_cubiculo, c.estado, u.id\_usuario, u.nombre, u.apellido

             FROM cubiculos c

             LEFT JOIN usuarios u ON c.usuario\_id = u.id\_usuario";

$rows = query($consulta); // Usar la función definida en mysql.php

// Verificar si la consulta devolvió resultados

if (is\_array($rows) && !isset($rows['error'])) {

    foreach ($rows as $row) {

        $estado = $row["estado"] === 'ocupado' ? 'Ocupado' : 'Disponible';

        $estadoClass = $row["estado"] === 'ocupado' ? 'estado-ocupado' : 'estado-disponible';

        echo "<div class='cubiculo'>";

        echo "Cubículo " . htmlspecialchars($row["id\_cubiculo"]) . " - Estado: <span class='" . htmlspecialchars($estadoClass) . "'>" . htmlspecialchars($estado) . "</span>";

        // Mostrar usuario solo si el cubículo está ocupado

        if ($row["estado"] === 'ocupado') {

            echo " - Usuario ID: " . htmlspecialchars($row["id\_usuario"]);

            echo " - Nombre: " . htmlspecialchars($row["nombre"]) . " " . htmlspecialchars($row["apellido"]);

        }

        echo "</div>";

    }

} else {

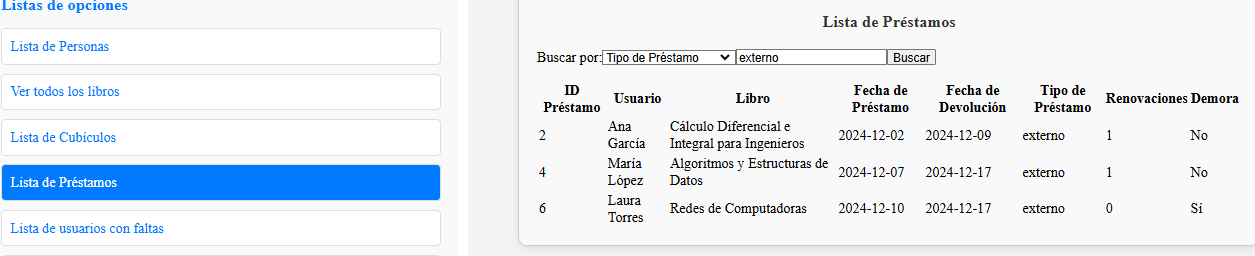
    echo "<p>No hay cubículos disponibles o no se encontraron datos.</p>";

}

?>

**LISTA DE PRESTAMOS**

****

****

<?php

require("mysql.php");

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/lis\_prestamos.css">';

echo "<div class='container'>";

echo "<h3>Lista de Préstamos</h3>";

// Formulario de búsqueda

echo "<form method='GET' action=''>";

echo "<label for='criterio'>Buscar por:</label>";

echo "<select name='criterio' id='criterio'>";

echo "<option value='fecha\_prestamo' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'fecha\_prestamo' ? 'selected' : '') . ">Fecha de Préstamo</option>";

echo "<option value='fecha\_devolucion' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'fecha\_devolucion' ? 'selected' : '') . ">Fecha de Devolución</option>";

echo "<option value='tipo\_prestamo' " . (isset($\_GET['criterio']) && $\_GET['criterio'] == 'tipo\_prestamo' ? 'selected' : '') . ">Tipo de Préstamo</option>";

echo "</select>";

// Campo de texto para la búsqueda

echo "<input type='text' name='buscar' placeholder='Ingrese su búsqueda' value='" . (isset($\_GET['buscar']) ? $\_GET['buscar'] : '') . "' />";

// Botón de búsqueda

echo "<input type='submit' value='Buscar' />";

echo "</form>";

// Filtro de tipo de préstamo

$tipo\_prestamo = isset($\_GET['tipo\_prestamo']) ? $\_GET['tipo\_prestamo'] : '';

// Condiciones para la consulta SQL

$criterio = isset($\_GET['criterio']) ? $\_GET['criterio'] : '';

$buscar = isset($\_GET['buscar']) ? $\_GET['buscar'] : '';

$where = "WHERE 1";

// Si se selecciona un criterio y un término de búsqueda

if ($criterio != '' && $buscar != '') {

    if ($criterio == 'fecha\_prestamo' || $criterio == 'fecha\_devolucion') {

        $where .= " AND $criterio LIKE '%$buscar%'";

    } elseif ($criterio == 'tipo\_prestamo') {

        $where .= " AND tipo\_prestamo LIKE '%$buscar%'";

    }

}

// Consulta para obtener los préstamos con la búsqueda aplicada

$query = "SELECT p.\*, u.nombre AS nombre\_usuario, u.apellido AS apellido\_usuario, l.titulo AS titulo\_libro

          FROM prestamos p

          JOIN usuarios u ON p.usuario\_id = u.id\_usuario

          JOIN libros l ON p.libro\_id = l.id\_libro

          $where

          ORDER BY p.fecha\_prestamo ASC";

$rows = query($query);

// Mostrar los resultados de la búsqueda

if (count($rows) > 0) {

    echo "<table class='tabla-prestamos'>";

    echo "<tr><th>ID Préstamo</th><th>Usuario</th><th>Libro</th><th>Fecha de Préstamo</th><th>Fecha de Devolución</th><th>Tipo de Préstamo</th><th>Renovaciones</th><th>Demora</th></tr>";

    foreach ($rows as $row) {

        echo "<tr>";

        echo "<td>" . $row["id\_prestamo"] . "</td>";

        echo "<td>" . $row["nombre\_usuario"] . " " . $row["apellido\_usuario"] . "</td>";

        echo "<td>" . $row["titulo\_libro"] . "</td>";

        echo "<td>" . $row["fecha\_prestamo"] . "</td>";

        echo "<td>" . ($row["fecha\_devolucion"] != "0000-00-00" ? $row["fecha\_devolucion"] : "Pendiente") . "</td>";

        echo "<td>" . $row["tipo\_prestamo"] . "</td>";

        echo "<td>" . $row["renovaciones"] . "</td>";

        echo "<td>" . ($row["demora"] == 1 ? "Sí" : "No") . "</td>";

        echo "</tr>";

    }

    echo "</table>";

} else {

    echo "<p>No se encontraron resultados para su búsqueda.</p>";

}

echo "</div>"; // Cierra el contenedor

?>

LISTA DE USUARIOS CON FALTA



<?php

require("mysql.php");

?>

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Registro de Conducta</title>

    <link rel="stylesheet" type="text/css" href="css/lis\_faltas.css">

</head>

<body>

    <h3>Registro de Mala Conducta de Usuarios</h3>

    <!-- Tabla para mostrar los registros de conducta -->

    <table class="tabla-registros">

        <thead>

            <tr>

                <th>ID Conducta</th>

                <th>Nombre</th>

                <th>Apellido</th>

                <th>Tipo de Usuario</th>

                <th>Carrera</th>

                <th>Código o DNI</th>

                <th>ID Préstamo</th>

                <th>Fecha del Evento</th>

                <th>Tipo de Evento</th>

                <th>Descripción</th>

            </tr>

        </thead>

        <tbody>

            <?php

            // Consulta SQL para obtener los registros de mala conducta

            $query = "SELECT cu.id\_conducta, u.nombre, u.apellido, u.tipo\_usuario, u.carrera, u.codigo\_o\_dni,

                      cu.prestamo\_id, cu.fecha\_evento, cu.tipo\_evento, cu.descripcion

                      FROM conducta\_usuario cu

                      JOIN usuarios u ON cu.usuario\_id = u.id\_usuario";

            $rows = query($query);

            // Iteración de los resultados para mostrarlos en la tabla

            foreach ($rows as $row) {

                echo "<tr>";

                echo "<td>" . htmlspecialchars($row["id\_conducta"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["nombre"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["apellido"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["tipo\_usuario"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["carrera"] ?? 'N/A') . "</td>";

                echo "<td>" . htmlspecialchars($row["codigo\_o\_dni"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["prestamo\_id"] ?? 'N/A') . "</td>";

                echo "<td>" . htmlspecialchars($row["fecha\_evento"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["tipo\_evento"]) . "</td>";

                echo "<td>" . htmlspecialchars($row["descripcion"] ?? 'N/A') . "</td>";

                echo "</tr>";

            }

            ?>

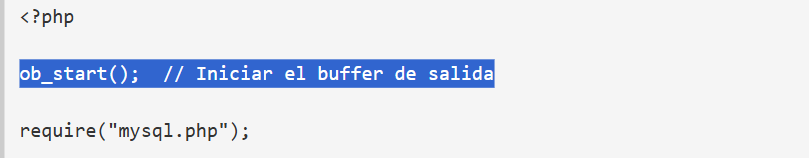
        </tbody>

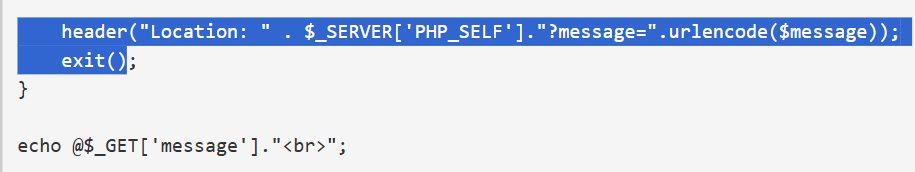
    </table>

</body>

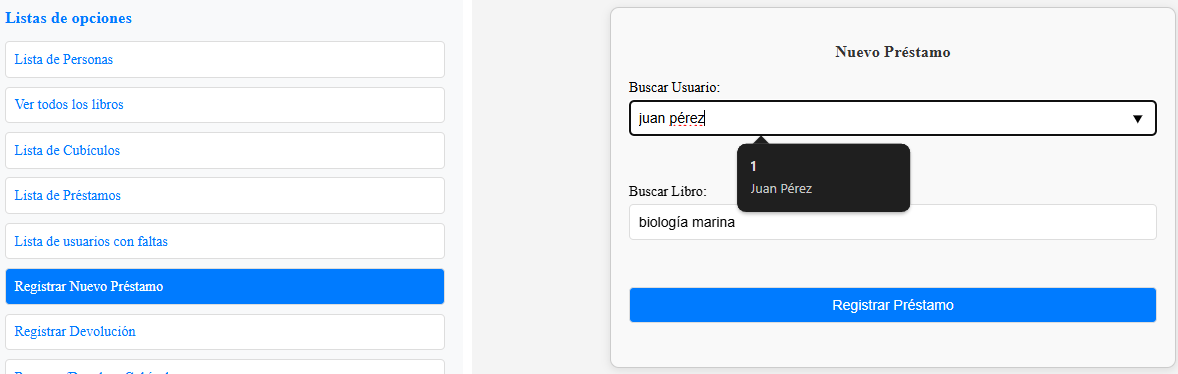
</html>

* Se agrego una nueva codificación para el buen funcionamiento del sistema.



****

REGISTRO DE NUEVO PRESTAMOS

****

<?php

require("mysql.php");

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/prestamo.css">';

// Función para ajustar el valor de AUTO\_INCREMENT

function ajustarAutoIncrement() {

    // Consulta para obtener el último id\_prestamo

    $ultimoId = query("SELECT MAX(id\_prestamo) AS ultimo\_id FROM prestamos")[0]['ultimo\_id'];

    // Si el último id\_prestamo es mayor o igual a 19, se ajusta el AUTO\_INCREMENT

    if ($ultimoId >= 19) {

        query("ALTER TABLE prestamos AUTO\_INCREMENT = " . ($ultimoId + 1));

    }

}

if (isset($\_POST['submit-registrar'])) {

    $usuario\_id = $\_POST['usuario\_id'];

    $libro\_id = $\_POST['libro\_id'];

    // Llamamos a la función para ajustar el AUTO\_INCREMENT antes de insertar un nuevo préstamo

    ajustarAutoIncrement();

    // Agregar el préstamo

    $r = query("INSERT INTO prestamos (usuario\_id, libro\_id, fecha\_prestamo, tipo\_prestamo)

                VALUES ('{$usuario\_id}', '{$libro\_id}', NOW(), 'interno')");

    if (isset($r['error'])) {

        $message = "Error al registrar préstamo: " . $r["error"];

    } else {

        $message = "Préstamo registrado exitosamente.";

    }

    header("Location: " . $\_SERVER['PHP\_SELF']."?message=".urlencode($message));

    exit();

}

echo @$\_GET['message']."<br>";

echo "<div class='container'>";

echo "<h3>Nuevo Préstamo</h3>";

echo "

    <form action='#' method='post'>

        <label for='usuario\_id'>Buscar Usuario:</label>

        <input list='usuarios' name='usuario\_id' id='usuario\_id' placeholder='Escribe el nombre del usuario'>

        <datalist id='usuarios'>

";

// Obtener usuarios existentes de la base de datos

$usuarios = query("SELECT id\_usuario, nombre, apellido FROM usuarios");

foreach ($usuarios as $usuario) {

    echo "<option value='{$usuario['id\_usuario']}'> {$usuario['nombre']} {$usuario['apellido']}</option>";

}

echo "

        </datalist>

        <br><br>

        <label for='libro\_id'>Buscar Libro:</label>

        <input list='libros' name='libro\_id' id='libro\_id' placeholder='Escribe el nombre del libro'>

        <datalist id='libros'>

";

// Obtener libros existentes de la base de datos

$libros = query("SELECT id\_libro, titulo FROM libros");

foreach ($libros as $libro) {

    echo "<option value='{$libro['id\_libro']}'> {$libro['titulo']}</option>";

}

echo "

        </datalist>

        <br><br>

        <button type='submit' name='submit-registrar'>Registrar Préstamo</button>

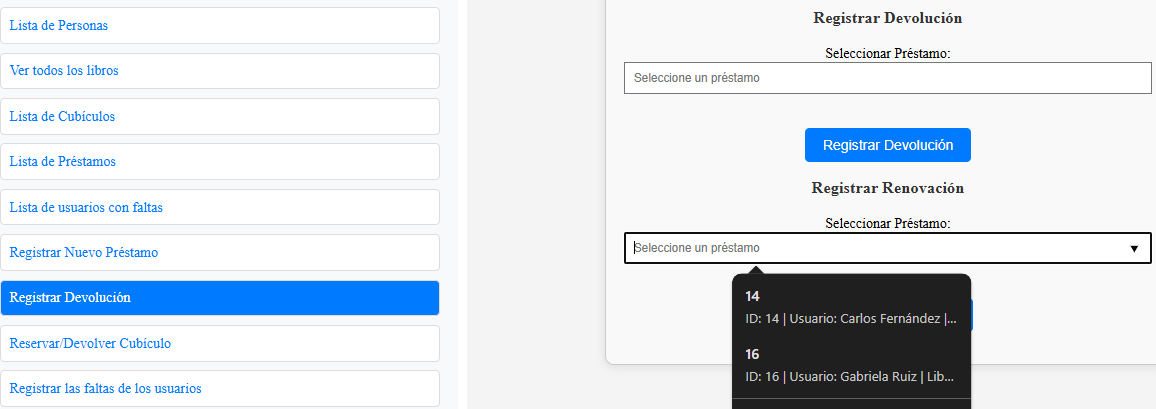
    </form>

</div>

";

?>

REGISTRO DE DEVOLUCION



<?php

// Enlace al CSS específico

echo "<link rel='stylesheet' type='text/css' href='css/devolucion.css'>";

require("mysql.php");

// Función para registrar devolución

function registrarDevolucion($prestamo\_id) {

    // Actualizar fecha\_devolucion con la fecha actual

    $r = query("UPDATE prestamos SET fecha\_devolucion = NOW() WHERE id\_prestamo='{$prestamo\_id}'");

    if (isset($r['error'])) {

        return "Error al registrar la devolución: " . $r["error"];

    }

    return "Devolución registrada exitosamente.";

}

// Función para registrar renovación

function registrarRenovacion($prestamo\_id) {

    // Incrementar el número de renovaciones y mantener fecha\_devolucion como NULL

    $r = query("UPDATE prestamos SET renovaciones = renovaciones + 1 WHERE id\_prestamo='{$prestamo\_id}'");

    if (isset($r['error'])) {

        return "Error al registrar la renovación: " . $r["error"];

    }

    return "Renovación registrada exitosamente.";

}

// Verificar si se ha enviado el formulario de devolución

if (isset($\_POST['submit-devolucion'])) {

    $prestamo\_id = $\_POST['prestamo\_id'];

    $message = registrarDevolucion($prestamo\_id);

    header("Location: " . $\_SERVER['PHP\_SELF'] . "?message=" . urlencode($message));

    exit();

}

// Verificar si se ha enviado el formulario de renovación

if (isset($\_POST['submit-renovacion'])) {

    $prestamo\_id = $\_POST['prestamo\_id'];

    $message = registrarRenovacion($prestamo\_id);

    header("Location: " . $\_SERVER['PHP\_SELF'] . "?message=" . urlencode($message));

    exit();

}

// Mostrar mensaje (si existe)

echo @$\_GET['message'] . "<br>";

// Consultar préstamos con `NULL` en `fecha\_devolucion`

$prestamos = query("

    SELECT p.id\_prestamo, u.nombre, u.apellido, l.titulo

    FROM prestamos p

    JOIN usuarios u ON p.usuario\_id = u.id\_usuario

    JOIN libros l ON p.libro\_id = l.id\_libro

    WHERE p.fecha\_devolucion IS NULL

");

echo "<div class='container'>"; // Añadir contenedor

// Formulario para registrar devolución

echo "<h3>Registrar Devolución</h3>";

echo "

    <form action='#' method='post'>

        <label for='prestamo\_id'>Seleccionar Préstamo:</label>

        <input list='prestamos-devolucion' name='prestamo\_id' id='prestamo\_id\_devolucion' placeholder='Seleccione un préstamo' style='width: 100%; padding: 10px;'>

        <datalist id='prestamos-devolucion'>

";

foreach ($prestamos as $prestamo) {

    echo "<option value='{$prestamo['id\_prestamo']}'>ID: {$prestamo['id\_prestamo']} | Usuario: {$prestamo['nombre']} {$prestamo['apellido']} | Libro: {$prestamo['titulo']}</option>";

}

echo "

        </datalist>

        <br><br>

        <button type='submit' name='submit-devolucion'>Registrar Devolución</button>

    </form>

";

// Formulario para registrar renovación

echo "<h3>Registrar Renovación</h3>";

echo "

    <form action='#' method='post'>

        <label for='prestamo\_id'>Seleccionar Préstamo:</label>

        <input list='prestamos-renovacion' name='prestamo\_id' id='prestamo\_id\_renovacion' placeholder='Seleccione un préstamo' style='width: 100%; padding: 10px;'>

        <datalist id='prestamos-renovacion'>

";

foreach ($prestamos as $prestamo) {

    echo "<option value='{$prestamo['id\_prestamo']}'>ID: {$prestamo['id\_prestamo']} | Usuario: {$prestamo['nombre']} {$prestamo['apellido']} | Libro: {$prestamo['titulo']}</option>";

}

echo "

        </datalist>

        <br><br>

        <button type='submit' name='submit-renovacion'>Registrar Renovación</button>

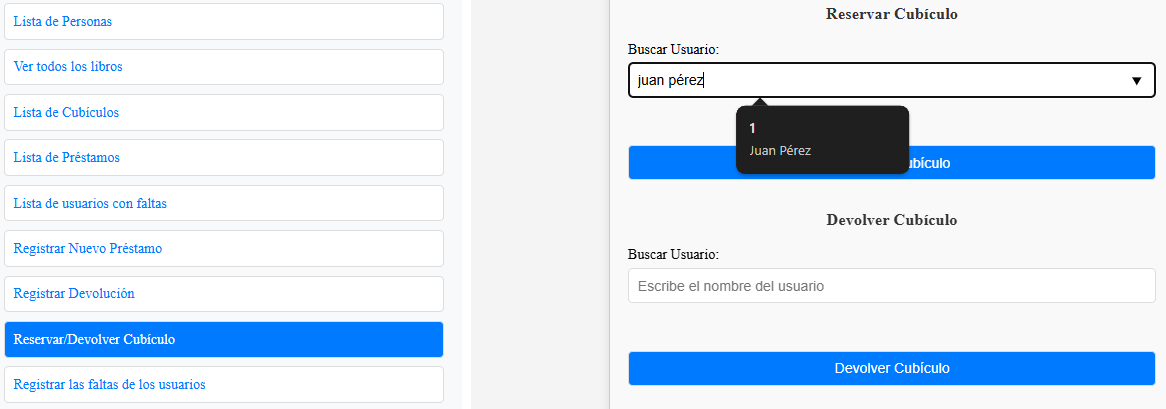
    </form>

";

echo "</div>"; // Cerrar contenedor

?>

RESERVA O DEVOLUCION DE CUBICULO



<?php

require("mysql.php");

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/prestamo.css">';

echo @$\_GET['message']."<br>";

echo "<div class='container'>";

/\* --- Reservar Cubículo --- \*/

echo "<h3>Reservar Cubículo</h3>";

echo "

    <form action='#' method='post'>

        <label for='usuario\_id\_reserva'>Buscar Usuario:</label>

        <input list='usuarios\_reserva' name='usuario\_id' id='usuario\_id\_reserva' placeholder='Escribe el nombre del usuario'>

        <datalist id='usuarios\_reserva'>

";

$usuarios = query("SELECT id\_usuario, nombre, apellido FROM usuarios");

foreach ($usuarios as $usuario) {

    echo "<option value='{$usuario['id\_usuario']}'> {$usuario['nombre']} {$usuario['apellido']}</option>";

}

echo "

        </datalist>

        <br><br>

        <button type='submit' name='submit-reservar'>Reservar Cubículo</button>

    </form>

";

if (isset($\_POST['submit-reservar'])) {

    $usuario\_id = $\_POST['usuario\_id'];

    if (empty($usuario\_id)) {

        $message = "Por favor, seleccione un usuario.";

    } else {

        // Verificar si el usuario ya tiene un cubículo reservado

        $cubiculo = query("SELECT id\_cubiculo FROM cubiculos WHERE usuario\_id = '{$usuario\_id}' AND estado = 'ocupado'");

        if (count($cubiculo) > 0) {

            $message = "Este usuario ya tiene un cubículo reservado.";

        } else {

            // Reservar un cubículo disponible

            $r = query("UPDATE cubiculos SET estado = 'ocupado', usuario\_id = '{$usuario\_id}', fecha\_reserva = NOW() WHERE estado = 'disponible' LIMIT 1");

            if (isset($r["error"])) {

                $message = "Error: " . $r["error"];

            } else {

                $message = "Cubículo reservado correctamente.";

            }

        }

    }

}

/\* --- Devolver Cubículo --- \*/

echo "<h3>Devolver Cubículo</h3>";

echo "

    <form action='#' method='post'>

        <label for='usuario\_id\_devolucion'>Buscar Usuario:</label>

        <input list='usuarios\_devolucion' name='usuario\_id' id='usuario\_id\_devolucion' placeholder='Escribe el nombre del usuario'>

        <datalist id='usuarios\_devolucion'>

";

$usuarios\_ocupados = query("SELECT u.id\_usuario, u.nombre, u.apellido FROM usuarios u

                            JOIN cubiculos c ON u.id\_usuario = c.usuario\_id

                            WHERE c.estado = 'ocupado'");

foreach ($usuarios\_ocupados as $usuario) {

    echo "<option value='{$usuario['id\_usuario']}'> {$usuario['nombre']} {$usuario['apellido']}</option>";

}

echo "

        </datalist>

        <br><br>

        <button type='submit' name='submit-devolver'>Devolver Cubículo</button>

    </form>

";

if (isset($\_POST['submit-devolver'])) {

    $usuario\_id = $\_POST['usuario\_id'];

    if (empty($usuario\_id)) {

        $message = "Por favor, seleccione un usuario.";

    } else {

        // Verificar que el usuario tiene un cubículo reservado

        $cubiculo = query("SELECT id\_cubiculo FROM cubiculos WHERE usuario\_id = '{$usuario\_id}' AND estado = 'ocupado'");

        if (count($cubiculo) > 0) {

            $id\_cubiculo = $cubiculo[0]['id\_cubiculo'];

            // Actualizar el cubículo para marcarlo como disponible

            $r = query("UPDATE cubiculos SET estado = 'disponible', usuario\_id = NULL, fecha\_reserva = NULL WHERE id\_cubiculo = {$id\_cubiculo}");

            if (isset($r["error"])) {

                $message = "Error: " . $r["error"];

            } else {

                $message = "Cubículo devuelto correctamente.";

            }

        } else {

            $message = "Este usuario no tiene cubículo reservado.";

        }

    }

}

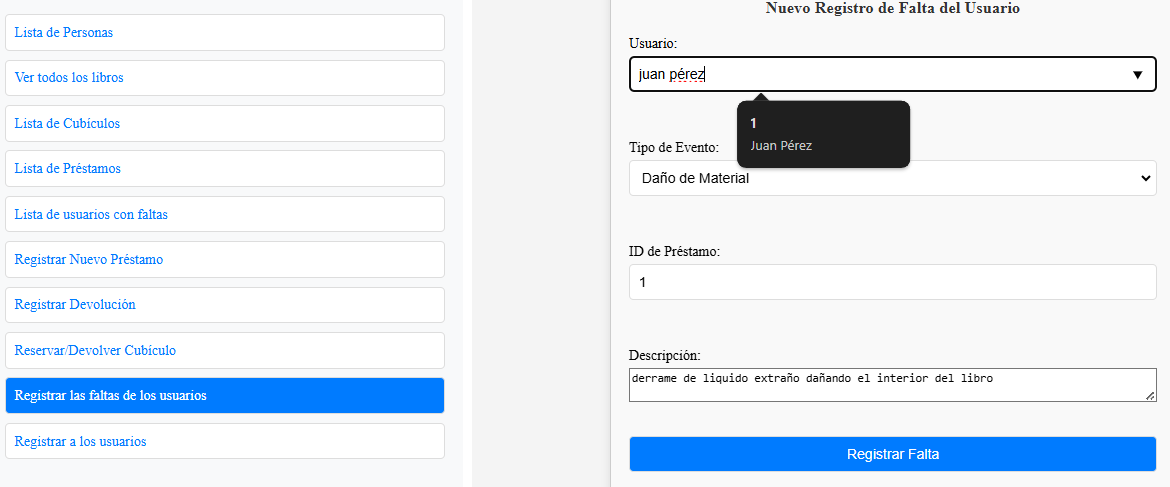
// Mostrar el mensaje de éxito o error si existe

echo isset($message) ? $message : '';

echo "</div>";

?>

REGISTRAR FALTAS DE LOS USUARIOS



<?php

require("mysql.php");

// Enlace al CSS

echo '<link rel="stylesheet" type="text/css" href="css/prestamo.css">';

echo "<div class='container'>";

echo "<h3>Nuevo Registro de Falta del Usuario</h3>";

echo "

    <form action='#' method='post'>

        <label for='usuario\_id'>Usuario:</label>

        <input list='usuarios' name='usuario\_id' id='usuario\_id' placeholder='Escribe el nombre del usuario'>

        <datalist id='usuarios'>

";

// Obtener usuarios existentes de la base de datos

$usuarios = query("SELECT id\_usuario, nombre, apellido FROM usuarios");

foreach ($usuarios as $usuario) {

    echo "<option value='{$usuario['id\_usuario']}'> {$usuario['nombre']} {$usuario['apellido']}</option>";

}

echo "

        </datalist>

        <br><br>

        <label for='tipo\_evento'>Tipo de Evento:</label>

        <select name='tipo\_evento' id='tipo\_evento'>

            <option value='devolucion\_tardia'>Devolución Tardía</option>

            <option value='danio de material'>Daño de Material</option>

            <option value='comentario'>Comentario</option>

        </select>

        <br><br>

        <label for='prestamo\_id'>ID de Préstamo:</label>

        <input list='prestamos' name='prestamo\_id' id='prestamo\_id' placeholder='Selecciona un préstamo'>

        <datalist id='prestamos'>

";

// Obtener préstamos existentes de la base de datos con nombre de usuario y libro

$prestamos = query("

    SELECT

        p.id\_prestamo,

        u.nombre AS usuario\_nombre,

        u.apellido AS usuario\_apellido,

        l.titulo AS libro\_titulo

    FROM prestamos p

    JOIN usuarios u ON p.usuario\_id = u.id\_usuario

    JOIN libros l ON p.libro\_id = l.id\_libro

");

foreach ($prestamos as $prestamo) {

    echo "<option value='{$prestamo['id\_prestamo']}'>ID: {$prestamo['id\_prestamo']} - Usuario: {$prestamo['usuario\_nombre']} {$prestamo['usuario\_apellido']} - Libro: {$prestamo['libro\_titulo']}</option>";

}

echo "

        </datalist>

        <br><br>

        <label for='descripcion'>Descripción:</label>

        <textarea name='descripcion' id='descripcion' placeholder='Escribe una descripción sobre el evento'></textarea>

        <br><br>

        <button type='submit' name='submit-registrar'>Registrar Falta</button>

    </form>

</div>

";

// Procesar el formulario al enviar

if (isset($\_POST['submit-registrar'])) {

    $usuario\_id = $\_POST['usuario\_id'];

    $prestamo\_id = $\_POST['prestamo\_id'] ?? null;

    $tipo\_evento = $\_POST['tipo\_evento'];

    $descripcion = $\_POST['descripcion'];

    // Validar si el usuario existe

    $usuario = query("SELECT id\_usuario FROM usuarios WHERE id\_usuario = '$usuario\_id'");

    if ($usuario) {

        // Insertar falta en la tabla conducta\_usuario

        query("

            INSERT INTO conducta\_usuario (usuario\_id, prestamo\_id, fecha\_evento, tipo\_evento, descripcion)

            VALUES ('$usuario\_id', '$prestamo\_id', NOW(), '$tipo\_evento', '$descripcion')

        ");

        // Si el evento es "devolución tardía", actualizar el campo demora en la tabla prestamos

        if ($tipo\_evento === 'devolucion\_tardia' && $prestamo\_id) {

            query("UPDATE prestamos SET demora = 1 WHERE id\_prestamo = '$prestamo\_id'");

        }

        echo "<p>Registro de falta realizado con éxito.</p>";

    } else {

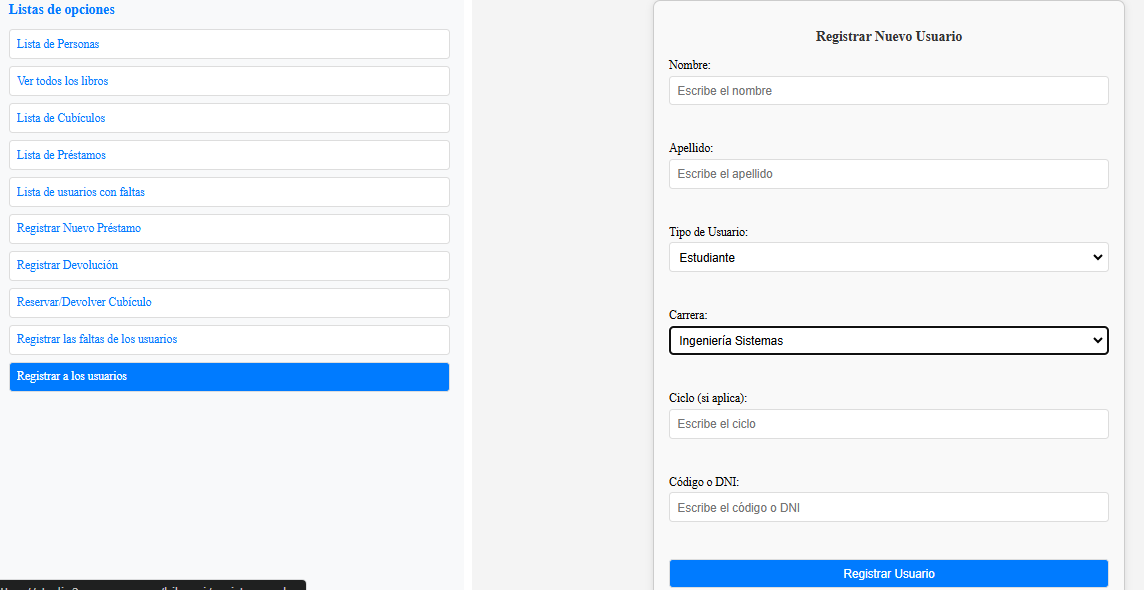
        echo "<p>No se encontró el usuario. Verifica el ID ingresado.</p>";

    }

}

?>

REGISTRO DE NUEVO USUARIO



<?php

require("mysql.php");

// Enlace al nuevo CSS

echo '<link rel="stylesheet" type="text/css" href="css/prestamo.css">';

echo @$\_GET['message']."<br>";

echo "<div class='container'>";

echo "<h3>Registrar Nuevo Usuario</h3>";

if (isset($\_POST['submit-registrar'])) {

    // Recuperar los datos del formulario

    $nombre = $\_POST['nombre'];

    $apellido = $\_POST['apellido'];

    $tipo\_usuario = $\_POST['tipo\_usuario'];

    $carrera = $\_POST['carrera'];

    $ciclo = $\_POST['ciclo'];

    $codigo\_o\_dni = $\_POST['codigo\_o\_dni'];

    // Validación simple (puedes añadir más validaciones si lo deseas)

    if (empty($nombre) || empty($apellido) || empty($tipo\_usuario) || empty($codigo\_o\_dni)) {

        $message = "Por favor, complete todos los campos obligatorios.";

    } else {

        // Consultar si ya existe un usuario con el mismo código o DNI

        $existe\_usuario = query("SELECT id\_usuario FROM usuarios WHERE codigo\_o\_dni = '{$codigo\_o\_dni}'");

        if (count($existe\_usuario) > 0) {

            $message = "Ya existe un usuario con este código o DNI.";

        } else {

            // Insertar el nuevo usuario

            $r = query("INSERT INTO usuarios (nombre, apellido, tipo\_usuario, carrera, ciclo, codigo\_o\_dni)

                        VALUES ('{$nombre}', '{$apellido}', '{$tipo\_usuario}', '{$carrera}', '{$ciclo}', '{$codigo\_o\_dni}')");

            if (isset($r["error"])) {

                $message = "Error al registrar el usuario: " . $r["error"];

            } else {

                $message = "Usuario registrado exitosamente.";

            }

        }

    }

}

echo isset($message) ? $message : '';

echo "

    <form action='#' method='post'>

        <label for='nombre'>Nombre:</label>

        <input type='text' name='nombre' id='nombre' placeholder='Escribe el nombre' required>

        <br><br>

        <label for='apellido'>Apellido:</label>

        <input type='text' name='apellido' id='apellido' placeholder='Escribe el apellido' required>

        <br><br>

        <label for='tipo\_usuario'>Tipo de Usuario:</label>

        <select name='tipo\_usuario' id='tipo\_usuario' required>

            <option value='profesor'>Profesor</option>

            <option value='estudiante'>Estudiante</option>

            <option value='administrativo'>Administrativo</option>

            <option value='egresado'>Egresado</option>

        </select>

        <br><br>

        <label for='carrera'>Carrera:</label>

        <select name='carrera' id='carrera' required>

            <option value='Ingeniería Pesquera'>Ingeniería Pesquera</option>

            <option value='Contabilidad'>Contabilidad</option>

            <option value='Administración'>Administración</option>

            <option value='Derecho'>Derecho</option>

            <option value='Ingeniería Sistemas'>Ingeniería Sistemas</option>

            <option value='Ingeniería Ambiental'>Ingeniería Ambiental</option>

            <option value='No especificado'>No especificado</option>

        </select>

        <br><br>

        <label for='ciclo'>Ciclo (si aplica):</label>

        <input type='number' name='ciclo' id='ciclo' placeholder='Escribe el ciclo'>

        <br><br>

        <label for='codigo\_o\_dni'>Código o DNI:</label>

        <input type='text' name='codigo\_o\_dni' id='codigo\_o\_dni' placeholder='Escribe el código o DNI' required>

        <br><br>

        <button type='submit' name='submit-registrar'>Registrar Usuario</button>

    </form>

</div>

";

?>

* Aquí podrás ingresar a la pagina web

[**https://studio2.appperu.com/bib\_uni/index.php**](https://studio2.appperu.com/bib_uni/index.php)

* Aquí puedes encontrar los archivos para el proyecto

[**https://github.com/oroagua/proyecto-biblioteca-base-de-datos.git**](https://github.com/oroagua/proyecto-biblioteca-base-de-datos.git)

**CONCLUSIONES**

1. **Mejora en la Eficiencia y Organización:** La implementación de una base de datos para la biblioteca universitaria permite optimizar la gestión de los recursos bibliográficos al facilitar el almacenamiento y actualización de datos sobre los libros, usuarios y préstamos. Esta estructura reduce el riesgo de errores en el inventario y asegura que la información esté organizada y accesible para su consulta en tiempo real.
2. **Automatización de Procesos de Préstamo y Devolución:** La base de datos desarrollada proporciona un sistema estructurado para gestionar de manera automatizada los préstamos, devoluciones y renovaciones, así como las sanciones por retraso. Esto contribuye a una administración más efectiva y precisa de los materiales, beneficiando tanto a los usuarios como al personal de la biblioteca.
3. **Facilitación de la Búsqueda de Recursos:** El modelo de base de datos optimiza el proceso de libros mediante Esto permite a los usuarios localizar de manera rápida y eficiente los libros u otros recursos que necesitan.
4. **Soporte para Toma de Decisiones y Planeación:** La funcionalidad de generación de reportes estadísticos brinda una visión detallada del uso de la biblioteca, como los materiales más solicitados y el comportamiento de los usuarios. Esta información es fundamental para la toma de decisiones, ayudando a identificar áreas de mejora y a realizar una planeación adecuada para futuras adquisiciones y mejoras en los servicios.
5. **Contribución al Aprendizaje Práctico de Bases de Datos:** Este proyecto proporciona una valiosa oportunidad de aprendizaje para los estudiantes al aplicar conceptos de diseño y administración de bases de datos en un entorno práctico y realista. La construcción de este sistema permite una comprensión más profunda de los fundamentos de bases de datos relacionales, los cuales son esenciales para el desarrollo profesional en el ámbito de la informática y la gestión de información.
6. **Impacto Positivo en la Comunidad Universitaria:** La implementación de esta base de datos no solo mejora la eficiencia operativa de la biblioteca, sino que también enriquece la experiencia educativa y académica de los estudiantes, profesores y personal administrativo. Con un acceso más organizado y ágil a los recursos, la base de datos contribuye al aprovechamiento óptimo de los materiales de estudio y a la creación de un ambiente de aprendizaje más dinámico y accesible.

En conclusión, la creación de una base de datos para la biblioteca universitaria representa un avance significativo en la administración de los recursos bibliográficos y el servicio a los usuarios. La automatización, organización y accesibilidad que brinda este sistema no solo optimiza los procesos internos de la biblioteca, sino que también tiene un impacto positivo en la comunidad universitaria en su conjunto

**BIBLIOGRAFIA**

 **Silberschatz, A., Korth, H. F., & Sudarshan, S.** (2011). Fundamentos de bases de datos (6.a ed.). McGraw-Hill.

 **Pérez Noriega, J.** (2013). Bases de datos relacionales: Diseño, implementación y administración. Alfaomega.

 **Resolucion de comision organizadora n° 0168\_2021\_UNAM** Universidad Nacional de Moquegua

 **Oracle MySQL Documentation**  
URL: <https://dev.mysql.com/doc/>

 **Manual de phpMyAdmin**  
URL: <https://www.phpmyadmin.net>