## Resumen operaciones CRUD

### Listar Datos (SELECT)

```
<?php
include 'db_connection.php';

$sql = "SELECT * FROM Actividades";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
   while($row = $result->fetch_assoc()) {
      echo "id: " . $row["id_actividad"]. " - Actividad: " . $row["actividad"]. " - Tipo: " .
$row["tipo"]. " - Precio: " . $row["precio"]. "<br>";
   }
} else {
   echo "0 resultados";
}
$conn->close();
?>
```

### Insertar Datos (INSERT)

```
<?php
include 'db_connection.php';

$actividad = "Buceo";
$tipo = "Aventura";
$precio = 120.00;
$lugar = "Mar";
$imagen = "img/actividades/buceo.jpg";
$comentario = "Comentario sobre buceo";
$fecha_inicio = "2024-08-01";
$duracion = 1;
$hora_inicio = "08:00:00";
$hora_fin = "12:00:00";
$duracion_horas = 4;
$id_monitores = 1;
$id_equipamiento = 1;

$sql = "INSERT INTO Actividades (actividad, tipo, precio, lugar, imagen, comentario,
fecha_inicio, duracion, hora_inicio, hora_fin, duracion_horas, id_monitores, id_equipamiento)
VALUES ('$actividad', '$tipo', $precio, '$lugar', '$imagen', '$comentario', '$fecha_inicio',
$duracion, '$hora_inicio', '$hora_fin', $duracion_horas, $id_monitores, $id_equipamiento)";

if ($conn->query($sql) === TRUE) {
   echo "Nuevo registro creado con éxito";
} else {
   echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

## Editar/Actualizar Datos (UPDATE)

```
<?php
include 'db_connection.php';

$id_actividad = 1; // ID de la actividad a actualizar
$precio = 60.00; // Nuevo precio

$sql = "UPDATE Actividades SET precio=$precio WHERE id_actividad=$id_actividad";

if ($conn->query($sql) === TRUE) {
   echo "Registro actualizado con éxito";
} else {
   echo "Error actualizando el registro: " . $conn->error;
}

$conn->close();
?>
```

## Borrar Datos (DELETE)

```
<?php
include 'db_connection.php';

$id_actividad = 1; // ID de la actividad a borrar

$sql = "DELETE FROM Actividades WHERE id_actividad=$id_actividad";

if ($conn->query($sql) === TRUE) {
    echo "Registro borrado con éxito";
} else {
    echo "Error borrando el registro: " . $conn->error;
}

$conn->close();
?>
```

## LISTAR

```
<!DOCTYPE html>
<html>
<head>
   <title>Listado de Camiones</title>
</head>
<body>
   <h1>Listado de Camiones</h1>
<?php
include 'config.php'; // Inclusión de Configuración y Consulta SQL

// Define una consulta SQL que selecciona todos los registros de la tabla 'vehiculos' donde 'subclase' es igual a 'Camion'.
$sql = "SELECT * FROM vehiculos WHERE subclase = 'Camion'";
$result = $conn->query($sql); // Ejecuta la consulta SQL y almacena el resultado en la variable $result.
esta parte del código en PHP representa una consulta SQL junto con su ejecución:
```

```
$sql = "SELECT * FROM vehiculos WHERE subclase = 'Camion'";
$result = $conn->query($sql);
```

## Comentarios del Código

1. **`$sql = "SELECT * FROM vehiculos WHERE subclase = 'Camion'";`**
    - Aquí se está creando una cadena de texto `$sql` que contiene una declaración SQL.
    - `SELECT * FROM vehiculos` selecciona todos los campos (`*`) de la tabla `vehiculos` en la base de datos.
    - `WHERE subclase = 'Camion'` es la condición que deben cumplir los registros seleccionados. Solo se seleccionarán los registros donde el campo `subclase` sea igual a `'Camion'`.
2. **`$result = $conn->query($sql);`**
    - Aquí se está ejecutando la consulta SQL definida previamente en `$sql`.
    - `$conn` es presumiblemente un objeto de conexión a la base de datos, posiblemente proveniente del archivo `config.php` incluido anteriormente en el código.
    - `->query($sql)` es un método que ejecuta la consulta SQL proporcionada.
    - El resultado de la consulta se almacena en la variable `$result`.
    - Si la consulta es exitosa, `$result` será un objeto que representa el conjunto de resultados obtenidos.
    - Si hay un error en la consulta, `$result` será `FALSE`.

En resumen, este fragmento de código está preparando y ejecutando una consulta SQL para seleccionar todos los registros de la tabla `vehiculos` donde el campo `subclase` sea igual a `'Camion'`, y está almacenando los resultados (o un valor `FALSE` en caso de error) en la variable `$result`.

```
// Verificación de Resultados y Creación de la Tabla
if ($result->num_rows > 0) { // Verifica si hay más de 0 registros obtenidos.
   // Inicia la construcción de una tabla HTML para mostrar los resultados.
   echo "<table border='1'>
      <tr>
         <th>Marca</th>
         <th>Modelo</th>
         <th>Cilindrada</th>
         <th>Fecha de Fabricación</th>
      </tr>";

   // Itera sobre cada registro obtenido.
   while ($row = $result->fetch_assoc()) {
      // Por cada registro, crea una nueva fila en la tabla y muestra los datos del registro en diferentes celdas.
      echo "<tr>
            <td>" . $row["marca"] . "</td>
            <td>" . $row["modelo"] . "</td>
            <td>" . $row["cilindrada"] . "</td>
            <td>" . $row["fechaFabricacion"] . "</td>
         </tr>";
```

```
    }
    echo "</table>"; // Finaliza la construcción de la tabla HTML.
} else {
    // Si no se obtuvieron registros, muestra un mensaje indicando que no se encontraron camiones.
    echo "No se encontraron camiones.";
}

$conn->close(); // Cierra la conexión a la base de datos.
?>
    <br>
    <a href="index.php">Volver al Listado General</a>
</body>
</html>
```

## AGREGAR

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Agregar Vehículo</title>
</head>
<body>
    <h1>Agregar Vehículo</h1>
    <?php
    include 'config.php'; // Incluir el archivo de configuración de la base de datos.
    // Verificar si el formulario fue enviado mediante el método POST.
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        // Obtener los valores del formulario.
        $marca = $_POST["marca"];
        $modelo = $_POST["modelo"];
        $cilindrada = $_POST["cilindrada"];
        $fechaFabricacion = $_POST["fechaFabricacion"];
        $subclase = $_POST["subclase"];

        // Preparar la consulta SQL para insertar el nuevo vehículo en la base de datos.
        $sql = "INSERT INTO vehiculos (marca, modelo, cilindrada, fechaFabricacion, subclase)
                VALUES (?, ?, ?, ?, ?)";

        // Preparar el statement
        if ($stmt = $conn->prepare($sql)) {
            // Vincular los parámetros.
            $stmt->bind_param("sssss", $marca, $modelo, $cilindrada, $fechaFabricacion, $subclase);

            // Ejecutar la consulta.
            if ($stmt->execute()) {
                echo "Vehículo agregado con éxito."; // Mensaje de éxito.
            } else {
                echo "Error al agregar el vehículo: " . $stmt->error; // Mensaje de error.
            }

            $stmt->close(); // Cerrar el statement.
        } else {
            echo "Error de preparación de la consulta: " . $conn->error; // Mensaje de error en la preparación.
        }
    } // Fin del bloque de PHP
    ?>
```

<!-- La preparación del "statement" (o declaración) se refiere al proceso de crear un "prepared statement" en PHP, el cual es un feature que permite ejecutar consultas SQL de manera segura, reduciendo el riesgo de inyecciones SQL.  Aquí está el código en cuestión:

// Preparar el statement.
**if ($stmt = $conn->prepare($sql)) {**
¿Qué hace este código?
Utiliza el método prepare del objeto de conexión a la base de datos $conn, pasándole la cadena $sql que contiene una consulta SQL con "placeholders" (simbolizados por ?) donde irán los valores.
Si la preparación es exitosa, el método prepare devuelve un objeto de tipo mysqli_stmt que se almacena en la variable $stmt.

Siguiente Paso: Vinculación de Parámetros
Después de preparar el statement, se vinculan los parámetros, es decir, se sustituyen los "placeholders" por los valores reales que el usuario ha ingresado en el formulario:

**$stmt->bind_param("sssss", $marca, $modelo, $cilindrada, $fechaFabricacion, $subclase);**
Aquí, "sssss" indica que se pasarán cinco strings como parámetros.
Luego, se listan las variables que contienen los valores que se sustituirán en los "placeholders" de la consulta SQL.
Beneficios
El uso de "prepared statements" tiene varias ventajas, entre ellas:

Seguridad: Ayuda a prevenir inyecciones SQL, ya que los valores se envían de forma separada de la consulta y no se interpretan como SQL.
Eficiencia: Si necesitas ejecutar una misma consulta múltiples veces con diferentes valores, preparar el statement una vez y ejecutarlo múltiples veces con diferentes valores es más eficiente.
Ejecución
Finalmente, una vez preparado y vinculado, el statement se ejecuta mediante:

**$stmt->execute();**
Si la ejecución es exitosa, se lleva a cabo la acción de la consulta SQL, en este caso, insertando un nuevo registro en la base de datos. Si ocurre un error, se maneja apropiadamente mostrando un mensaje de error. -->
 <br>
    <a href="index.php">Volver al Listado Principal</a> <!-- Enlace para volver a la página principal. -->
    <br><br>
    <form method="POST" action=""> <!-- Inicio del formulario, utilizando el método POST. -->
        <label for="marca">Marca:</label> <!-- Etiqueta para el campo marca. -->
        <input type="text" name="marca" required><br><br> <!-- Campo de texto para ingresar la marca. -->

        <label for="modelo">Modelo:</label> <!-- Etiqueta para el campo modelo. -->
        <input type="text" name="modelo" required><br><br> <!-- Campo de texto para ingresar el modelo. -->

        <label for="cilindrada">Cilindrada:</label> <!-- Etiqueta para el campo cilindrada. -->
        <input type="text" name="cilindrada" required><br><br> <!-- Campo de texto para ingresar la cilindrada. -->

        <label for="fechaFabricacion">Fecha de Fabricación:</label> <!-- Etiqueta para el campo fecha de fabricación. -->
        <input type="date" name="fechaFabricacion" required><br><br> <!-- Campo de fecha para ingresar la fecha de fabricación. -->

        <label for="subclase">Subclase:</label> <!-- Etiqueta para el campo subclase. -->
        <select name="subclase" required> <!-- Menú desplegable para seleccionar la subclase. -->

```
        <option value="Moto">Moto</option> <!-- Opción Moto. -->
        <option value="Coche">Coche</option> <!-- Opción Coche. -->
        <option value="Camión">Camión</option> <!-- Opción Camión. -->
    </select><br><br>

    <input type="submit" value="Agregar"> <!-- Botón para enviar el formulario. -->
  </form>
</body>
</html>
```

**BORRAR**

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Borrar Vehículo</title>
</head>
<body>
  <h1>Borrar Vehículo</h1>

  <?php
  include 'config.php'; // Inclusión del archivo de configuración de la base de datos.

  // Si el método de la solicitud es GET y se ha establecido el parámetro "id"...
  if ($_SERVER["REQUEST_METHOD"] == "GET" && isset($_GET["id"])) {
```

<!-- Explicación Detallada:

1. `$_SERVER["REQUEST_METHOD"] == "GET":`
    - **`$_SERVER["REQUEST_METHOD"]`** es una variable superglobal de PHP que contiene el método de la solicitud HTTP. En este caso, se está comparando con el string `"GET"` para ver si la solicitud usó el método GET. Los métodos HTTP típicos son `GET`, `POST`, `PUT`, `DELETE`, etc.
    - El método `GET` se utiliza generalmente para solicitar datos de un recurso.
2. `isset($_GET["id"])`
    - `$_GET` es otra variable superglobal de PHP y contiene los datos que son enviados al script mediante el método HTTP `GET`. Los datos se envían en la URL de la solicitud, por ejemplo: `http://dominio.com/archivo.php?id=123`.
    - `isset($_GET["id"])` verifica si existe un valor asociado a la clave `"id"` dentro del array `$_GET`, es decir, si el parámetro `id` ha sido enviado en la URL.

## En conjunto:

Si la condición `if ($_SERVER["REQUEST_METHOD"] == "GET" && isset($_GET["id"])) {` se cumple, significa que se ha hecho una solicitud `GET` a la página y se ha enviado un valor para el parámetro `id` en la URL. En este caso, el código dentro del bloque `if` se ejecutará, lo que implica que se tratará de buscar y mostrar los detalles del vehículo con el `id` especificado antes de proceder a borrarlo.

```
    $id = $_GET["id"]; // ...se asigna el valor de "id" a la variable $id.

    // Se prepara una consulta SQL para seleccionar toda la información del vehículo con el id específico.
    $sql = "SELECT * FROM vehiculos WHERE id = $id";
```

<!-- esta línea de código es una cadena que contiene una instrucción SQL para seleccionar todos los registros (`SELECT *`) de la tabla `vehiculos` donde el valor de la columna `id` es igual al valor de la variable `$id`.

## Detalles:

1. `SELECT * FROM vehiculos`
    - Esta parte de la instrucción SQL selecciona todos los campos (`*`) de todos los registros de la tabla `vehiculos`. Si se quisiera seleccionar campos específicos, se podría especificar el nombre de los campos en lugar de usar `*`, por ejemplo: `SELECT marca, modelo FROM vehiculos`.
2. `WHERE id = $id`
    - La cláusula `WHERE` filtra los resultados de la consulta para incluir solo aquellos registros donde el valor del campo `id` es igual al valor contenido en la variable PHP `$id`.
    - `$id` es una variable PHP que, en este contexto, ha sido previamente definida como el valor del parámetro `id` recibido por el método `GET`.

## Seguridad:

En el contexto de este código, sería importante mencionar que incorporar directamente variables PHP dentro de las consultas SQL, como se hace aquí, puede presentar riesgos de seguridad, específicamente inyecciones SQL, donde un usuario malicioso podría manipular la consulta SQL para acceder, modificar, o eliminar datos en la base de datos.

Para prevenir esto, es recomendable utilizar consultas preparadas (prepared statements) con mysqli o PDO, que aseguran que los datos proporcionados sean seguros antes de ejecutar la consulta SQL.

Ejemplo de cómo podría verse con una consulta preparada en mysqli:

```
$sql = "SELECT * FROM vehiculos WHERE id = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("i", $id); // "i" indica que el tipo de dato es un entero
$stmt->execute();
$result = $stmt->get_result(); -->

    $result = $conn->query($sql); // Se ejecuta la consulta.

    // Si se encuentra un registro...
    if ($result->num_rows == 1) {
        // ...se obtienen y asignan los datos del vehículo a variables.
        $row = $result->fetch_assoc();
        $marca = $row["marca"];
        $modelo = $row["modelo"];
        $cilindrada = $row["cilindrada"];
        $fechaFabricacion = $row["fechaFabricacion"];
        $subclase = $row["subclase"];
    } else { // Si no se encuentra ningún registro...
        echo "No se encontraron registros."; // ...se muestra un mensaje de error.
    }
}
```

<!-- Este bloque de código verifica si la consulta SQL anterior obtuvo exactamente un registro de la base de datos y, si es así, asigna los valores de los campos de ese registro a variables PHP correspondientes. Si no se encuentra ningún registro, se muestra un mensaje de error.
Aquí está el código con comentarios detallados:

```
if ($result->num_rows == 1) { // Si el número de filas (registros) obtenidos es exactamente 1...
    // ...se obtiene la fila como un array asociativo y se asigna a $row.
    $row = $result->fetch_assoc();

    // ...se obtienen y asignan los datos del vehículo a variables.
    $marca = $row["marca"]; // Asigna el valor del campo 'marca' a la variable $marca.
    $modelo = $row["modelo"]; // Asigna el valor del campo 'modelo' a la variable $modelo.
    $cilindrada = $row["cilindrada"]; // Asigna el valor del campo 'cilindrada' a la variable $cilindrada.
    $fechaFabricacion = $row["fechaFabricacion"]; // Asigna el valor del campo 'fechaFabricacion' a la variable $fechaFabricacion.
    $subclase = $row["subclase"]; // Asigna el valor del campo 'subclase' a la variable $subclase.
} else { // Si no se encuentra ningún registro...
    echo "No se encontraron registros."; // ...se muestra un mensaje indicando que no se encontraron registros.
}
```

## Detalles:

- **`$result->num_rows`:** Obtiene el número de filas (registros) que la consulta SQL retornó.
- **`fetch_assoc()`**: Esta función de PHP obtiene una fila del resultado de la consulta SQL como un array asociativo. Los nombres de las claves en este array son los nombres de las columnas de la tabla de la base de datos.
- Si la consulta SQL retorna exactamente un registro, los valores de los campos de este registro se asignan a las variables PHP correspondientes.
- Si no se encuentra ningún registro (si `$result->num_rows` no es 1), se muestra un mensaje `"No se encontraron registros."`, indicando que no hay registros que coincidan con el criterio de búsqueda especificado en la consulta SQL.

```
// Si el método de la solicitud es POST y se ha enviado el formulario de confirmación...
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST["confirmar"])) {
    $id = $_POST["id"]; // ...se asigna el valor de "id" a la variable $id.
```

<!-- esta condición verifica si el método de la solicitud HTTP es `POST` y si se ha enviado el valor de `"confirmar"` en el formulario. Si ambas condiciones son verdaderas, entonces se procede a ejecutar el código dentro del bloque `if`.

```
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST["confirmar"])) {
    $id = $_POST["id"]; // ...se asigna el valor de "id" a la variable $id.
```

## Detalles:

1. `$_SERVER["REQUEST_METHOD"] == "POST"`:
    - `$_SERVER["REQUEST_METHOD"]` es una variable superglobal que contiene el método de la solicitud HTTP. En este caso, se está comparando con el string `"POST"` para ver si la solicitud usó el método POST. El método `POST` se utiliza típicamente para enviar datos para ser procesados a un recurso en el servidor.

2. `isset($_POST["confirmar"])`

    - `$_POST` es otra variable superglobal que contiene los datos que son enviados al script mediante el método HTTP `POST`.
    - `isset($_POST["confirmar"])` verifica si se ha enviado un valor asociado a la clave `"confirmar"` dentro del array `$_POST`, que sería un botón o campo en el formulario enviado con método `POST`.

El parámetro **`confirmar`** se refiere a un campo o botón en un formulario HTML que, cuando se presenta o se presiona, envía una señal de confirmación para proceder con una acción específica, en este caso, podría ser la eliminación de un registro.

verifica si el método de la solicitud es `POST` y si el parámetro `confirmar` está presente en los datos del formulario enviados. Si ambas condiciones son verdaderas, el código dentro del bloque `if` se ejecuta.

En la parte HTML del código, podrías tener algo así para el botón o campo `confirmar`:

```
<input type="submit" name="confirmar" value="Confirmar Eliminación">
```

- Aquí, `name="confirmar"` define el parámetro `confirmar` que se revisa en el PHP con `isset($_POST["confirmar"])`.
- Cuando el usuario presiona el botón `"Confirmar Eliminación"`, el formulario se envía usando el método `POST`, y el código PHP posteriormente revisa si el parámetro `confirmar` está presente.

Este parámetro sirve como una señal o indicador para que el script PHP sepa que el usuario ha confirmado la acción deseada, y en respuesta, el script puede proceder con la lógica de negocio asociada, como podría ser borrar un registro de la base de datos.

3. `$id = $_POST["id"];`

    - Si se cumplen ambas condiciones del `if`, se asigna el valor de `"id"` enviado en el formulario (accesible a través de `$_POST["id"]`) a la variable `$id` para su posterior uso dentro del bloque.

## Resumen:

En resumen, este bloque de código maneja el caso cuando el usuario ha enviado un formulario mediante el método `POST`, y el botón o campo `"confirmar"` está presente en los datos enviados. En este escenario, se captura y asigna el valor de `"id"` a la variable `$id` para su uso posterior, posiblemente para procesar la eliminación del registro correspondiente a dicho `id` en la base de datos.

```
    // Se prepara una consulta SQL para borrar el vehículo con el id específico.
    $sql = "DELETE FROM vehiculos WHERE id = $id";
    if ($conn->query($sql) === TRUE) { // Si la consulta se ejecuta correctamente...
        echo "Vehículo eliminado con éxito."; // ...se muestra un mensaje de éxito.
    } else { // Si hay un error en la ejecución de la consulta...
        echo "Error al eliminar el vehículo: " . $conn->error; // ...se muestra un mensaje de error.
    }
}
?>// Fin del bloque de código PHP.
```

```
    <br>
    <a href="index.php">Volver al Listado</a> <!-- Enlace para volver al listado principal. -->
    <br><br>

    <?php if (isset($marca)) { ?> <!-- Si se ha establecido la variable $marca, se muestra el formulario de confirmación. -->
        <h2>Confirmar Eliminación</h2>
        <p>¿Estás seguro de que deseas eliminar el siguiente vehículo?</p> <!-- Mensaje de confirmación. -->
        <ul> <!-- Lista con los detalles del vehículo a eliminar. -->
            <li>Marca: <?php echo $marca; ?></li>
            <li>Modelo: <?php echo $modelo; ?></li>
            <li>Cilindrada: <?php echo $cilindrada; ?></li>
            <li>Fecha de Fabricación: <?php echo $fechaFabricacion; ?></li>
            <li>Subclase: <?php echo $subclase; ?></li>
        </ul>
        <form method="POST" action=""> <!-- Formulario de confirmación de eliminación. -->
            <input type="hidden" name="id" value="<?php echo $id; ?>"> <!-- Campo oculto con el id del vehículo. -->
            <input type="submit" name="confirmar" value="Confirmar Eliminación"> <!-- Botón de confirmación. -->
        </form>
    <?php } ?>
</body>
</html>
```

## EDITAR

```
<!DOCTYPE html>
<html>
<head>
    <title>Editar Vehículo</title>
</head>
<body>
    <h1>Editar Vehículo</h1>

 <?php
    include 'config.php'; // Incluye el archivo de configuración que probablemente contiene la conexión a la base de datos.

    // Si el método de solicitud es GET y se proporciona un ID, entonces busca el vehículo en la base de datos.
    if ($_SERVER["REQUEST_METHOD"] == "GET" && isset($_GET["id"])) {
        $id = $_GET["id"]; // Toma el ID proporcionado en la URL.

<!-- Este bloque de código se ejecutará si el método de solicitud HTTP es GET
y si hay un parámetro "id" establecido en la URL (por ejemplo, editVehiculo.php?
id=123).
if ($_SERVER["REQUEST_METHOD"] == "GET" && isset($_GET["id"])) {

  $_GET["id"] obtiene el valor del parámetro "id" de la URL.
  $id almacena este valor para su uso posterior dentro del bloque de código.
    $id = $_GET["id"];

   El resto del código dentro de este bloque generalmente manipularía o recuperaría
    información basada en el valor de $id, por ejemplo, recuperar un vehículo
    específico de la base de datos para editarlo.
}
```

En resumen, este código verifica si la página fue accedida mediante un método `GET` y si se proporcionó un parámetro `id` en la URL. Si ambas condiciones son verdaderas, asigna el valor de este parámetro a la variable `$id`, y probablemente lo usaría para realizar operaciones adicionales, como recuperar detalles de un vehículo de la base de datos utilizando el `$id` proporcionado. -->

```
// Realiza una consulta SQL para obtener la información del vehículo con el ID proporcionado.
$sql = "SELECT * FROM vehiculos WHERE id = $id";
$result = $conn→query($sql);
```

<!-- Este fragmento de código en PHP está ejecutando una consulta SQL en la base de datos. Aquí te dejo el código comentado:

```
phpCopy code
// La variable $sql contiene una cadena que representa la consulta SQL que queremos
ejecutar.
// Esta consulta seleccionará todos los campos (*) de la tabla 'vehiculos' donde el
'id'
// coincida con el valor almacenado en la variable $id.
$sql = "SELECT * FROM vehiculos WHERE id = $id";

// $conn probablemente es un objeto de conexión a la base de datos previamente creado.
// Aquí se está utilizando el método ->query() de este objeto para ejecutar la consulta
SQL
// almacenada en la variable $sql.
// El resultado de la ejecución de la consulta (que podría ser un conjunto de
registros,
// un valor booleano TRUE en caso de éxito o FALSE en caso de error) se almacena en la
variable $result.
$result = $conn->query($sql);
```

Este código, por tanto, está preparando y ejecutando una consulta SQL para obtener todos los datos de un registro específico en la tabla `vehiculos`, basándose en el valor de `id`, y almacena el resultado en la variable `$result` para su posterior procesamiento. -->

```
// Si se encuentra el vehículo, se asignan los datos del vehículo a variables PHP.
if ($result->num_rows == 1) {
   $row = $result->fetch_assoc();

} else {
   // Si no se encuentra ningún vehículo con el ID proporcionado, muestra un mensaje de error.
   echo "No se encontraron registros.";
}
}
```

<!-- Este fragmento de código está verificando si la consulta SQL ha devuelto exactamente un registro y luego procesa ese registro, o en caso contrario, muestra un mensaje de error. Aquí está el código con comentarios adicionales:

```
// Verifica si el número de filas (registros) devueltas por la consulta SQL es igual a
1.
// $result->num_rows accede al número de filas devueltas por la consulta.
if ($result->num_rows == 1) {
    // Si hay exactamente un registro, entonces se procede a recuperar los datos de ese
registro.
    // $result->fetch_assoc() extrae la siguiente fila de resultado como un array
asociativo,
    // donde las claves son los nombres de los campos y los valores son los datos
asociados a esos campos.
```

```
    // El array asociativo se almacena en la variable $row.
    $row = $result→fetch_assoc();

    // En este punto podrías procesar los datos almacenados en $row si así lo
necesitas.
    // Por ejemplo, asignar los valores a diferentes variables o mostrarlos en la
página.

} else {
    // Si el número de filas no es 1, significa que no se ha encontrado ningún registro
con el ID proporcionado.
    // En este caso, se muestra un mensaje de error al usuario indicando que no se
encontraron registros.
    echo "No se encontraron registros.";
}
// Fin del bloque de código encapsulado por las llaves, normalmente asociado a un
bloque if, loop o función.
```

Este bloque de código, por tanto, se encarga de manejar los resultados de una consulta SQL, recuperando los datos si se encuentra un registro, y manejando el caso en el que no se encuentre ningún registro con un mensaje de error.

El método `fetch_assoc()` en PHP se utiliza para obtener una fila de resultado de un conjunto de resultados asociado con una consulta MySQLi como un array asociativo.

Aquí tienes una breve explicación de cómo funciona:

**Uso Básico:**

Cuando ejecutas una consulta SQL para seleccionar datos de una base de datos, obtienes un objeto de resultado. Para acceder a los datos contenidos en este objeto de resultado, puedes utilizar diferentes métodos, y `fetch_assoc()` es uno de ellos.

**Funcionamiento:**

```
phpCopy code
$row = $result->fetch_assoc();
```

- `$result` es un objeto de resultado obtenido, por ejemplo, al ejecutar una consulta SQL usando métodos como `$conn->query($sql)`.
- `fetch_assoc()` extrae la siguiente fila disponible de este conjunto de resultados.
- La fila extraída se devuelve como un **array asociativo, donde las claves son los nombres de los campos (columnas) y los valores son los datos correspondientes de esos campos para esa fila particular.**
- Si no hay más filas en el conjunto de resultados, `fetch_assoc()` devuelve `null`, indicando que has alcanzado el final del conjunto de resultados.

**Ejemplo:**

Supón que tienes una tabla `usuarios` con dos columnas: `id` y `nombre`. Si ejecutas una consulta SQL para seleccionar todos los registros de esta tabla y luego usas `fetch_assoc()`, obtendrás algo así:

```
phpCopy code

$result = $conn->query("SELECT * FROM usuarios");
if($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        echo "ID: " . $row["id"] . " - Nombre: " . $row["nombre"] . "<br>";
    }
}
```

En este ejemplo, `$row["id"]` y `$row["nombre"]` acceden a los datos de las columnas `id` y `nombre` de la fila actual, respectivamente. Y el bucle `while` seguirá iterando y extrayendo filas del conjunto de resultados hasta que `fetch_assoc()` devuelva `null`, indicando que no hay más filas por procesar.

```
    if ($_SERVER["REQUEST_METHOD"] == "POST") { // Si el método de solicitud es POST, se toman los datos del formulario y se actualiza el vehículo en la base de datos.
        $id = $_POST["id"];
        $marca = $_POST["marca"];
        $modelo = $_POST["modelo"];
        $cilindrada = $_POST["cilindrada"];
        $fechaFabricacion = $_POST["fechaFabricacion"];
        $subclase = $_POST["subclase"];

        // Actualizar vehículo en la base de datos
        $sql = "UPDATE vehiculos
        SET marca = '$marca', modelo = '$modelo', cilindrada = '$cilindrada', fechaFabricacion = '$fechaFabricacion', subclase = '$subclase' WHERE id = $id";

        if ($conn->query($sql) === TRUE) {
            echo "Vehículo actualizado con éxito.";
        } else {
            echo "Error al actualizar el vehículo: " . $conn->error;
        }
    }
    ?>
// Si se han definido los datos del vehículo (es decir, si se ha encontrado un vehículo con el ID proporcionado), este fragmento mostrará un formulario precargado con los datos del vehículo para que el usuario los edite. Cuando se envía el formulario, se activa la lógica de actualización mencionada anteriormente.
    <br>
    <a href="index.php">Volver al Listado</a>
    <br><br>

    <?php if (isset($marca)) { ?>
        <h2>Editar Vehículo</h2>
        <!-- Se crea un formulario con los datos del vehículo que se pueden editar. -->
        <form method="POST" action="">
        <!-- Los campos están prellenados con los datos actuales del vehículo. -->
            <input type="hidden" name="id" value="<?php echo $id; ?>">
            <label for="marca">Marca:</label>
            <input type="text" name="marca" value="<?php echo $marca; ?>" required><br><br>

            <label for="modelo">Modelo:</label>
            <input type="text" name="modelo" value="<?php echo $modelo; ?>" required><br><br>

            <label for="cilindrada">Cilindrada:</label>
            <input type="text" name="cilindrada" value="<?php echo $cilindrada; ?>" required><br><br>

            <label for="fechaFabricacion">Fecha de Fabricación:</label>
            <input type="date" name="fechaFabricacion" value="<?php echo $fechaFabricacion; ?>" required><br><br>

            <label for="subclase">Subclase:</label>
            <select name="subclase" required>
```

```
            <option value="Moto" <?php if ($subclase == "Moto") echo "selected"; ?>>Moto</option>
            <option value="Coche" <?php if ($subclase == "Coche") echo "selected"; ?>>Coche</option>
            <option value="Camion" <?php if ($subclase == "Camion") echo "selected"; ?
>>Camion</option>
        </select><br><br>

        <input type="submit" value="Guardar Cambios">
    </form>
  <?php } ?>
</body>
</html>
```

<!-- `<input type="hidden" name="id" value="<?php echo $id; ?>">`

Este fragmento de HTML con PHP incrustado define un campo de entrada (`<input>`) que es utilizado para almacenar un valor que no será visible ni editable por el usuario, ya que su atributo `type` está configurado como `"hidden"`.

Vamos a desglosarlo:

1. `<input>`: Es la etiqueta HTML que se utiliza para crear un campo de entrada en un formulario.
2. `type="hidden"`: Este atributo y valor determinan que este campo de entrada será de tipo oculto, lo que significa que no se mostrará en la página web y será invisible para el usuario.
3. `name="id"`: El atributo `name` se utiliza para dar un nombre al campo de entrada, que será la clave cuando se envíen los datos del formulario. En este caso, la clave será `"id"`.
4. `value="<?php echo $id; ?>"`: Aquí se establece el valor del campo de entrada utilizando PHP. El código PHP `<?php echo $id; ?>` imprime el valor de la variable `$id` en el campo de entrada oculto. Es decir, asigna el valor de `$id` al atributo `value` del campo de entrada.

   **Utilidad:**

Este tipo de campos ocultos es útil cuando necesitas enviar información adicional con el formulario, como un identificador único (`id`), sin que el usuario lo vea o lo modifique. Cuando el formulario se envía, este valor oculto también se incluirá en los datos del formulario que se envían al servidor.

## Ejemplo de Uso:

Si estás editando un registro, como un vehículo en tu caso, necesitarías enviar el `id` del vehículo junto con los datos modificados del formulario, para que el servidor sepa qué registro debe actualizar en la base de datos. Como no quieres que el usuario modifique el `id` del vehículo, lo colocas en un campo oculto.

El código **`<?php } ?>`** es necesario en este contexto, ya que está cerrando un bloque condicional PHP que probablemente comenzó con algo similar a **`<?php if (isset($marca)) {`** ?>.

Este bloque condicional determina si se debe generar o no el formulario HTML, basándose en si la variable `$marca` está definida o no. Si `$marca` está definida, se asume que se ha recuperado con éxito un vehículo de la base de datos y se genera el formulario para editar dicho vehículo. El bloque de código entre `<?php if (isset($marca)) { ?>` y **`<?php } ?>`** solo se ejecutará si la condición del `if` es `true`, es decir, si `$marca` está definida.

Si omites el `<?php } ?>`, obtendrías un error de sintaxis PHP, ya que cada bloque `if` abierto con `{` debe ser cerrado con `}`, y en el contexto de un documento HTML, esto se hace dentro de un bloque PHP, denotado por `<?php ?>`.

## La función `isset` en PHP

Se utiliza para verificar si una variable o un elemento de un array o matriz está definido y no es nulo. Su sintaxis básica es la siguiente:

```
isset(variable1, variable2, ...);
```

Donde `variable1`, `variable2`, etc., son las variables o elementos que deseas verificar.

La **función `isset`** devuelve **`true`** si la variable o elemento está definido y no es nulo, y **`false`** si no lo está o es nulo.

**Ejemplos** :

```
$nombre = "Juan";
$edad = 30;

// Verificar si las variables están definidas
if (isset($nombre)) {
    echo "La variable nombre está definida.<br>";
}

if (isset($edad)) {
    echo "La variable edad está definida.<br>";
}

// Verificar una variable que no está definida
if (isset($direccion)) {
    echo "La variable dirección está definida.<br>";
} else {
    echo "La variable dirección no está definida.<br>";
}

// Verificar si un elemento de un array está definido
$frutas = array("manzana", "banana", "cereza");

if (isset($frutas[1])) {
    echo "El elemento 1 del array frutas está definido.<br>";
}
```

En este ejemplo:

1. **`isset($nombre)`** devuelve **`true`** porque la variable `$nombre` está definida y tiene un valor.
2. **`isset($edad)`** devuelve **`true`** porque la **variable `$edad`** está definida y tiene un valor.
3. **`isset($direccion)`** devuelve **`false`** porque la **variable `$direccion`** no está definida.
4. **`isset($frutas[1])`** devuelve **`true`** porque **el elemento 1 del array `$frutas`** (que contiene "banana") está definido.

La función `isset` es útil para evitar errores y comprobar si una variable o un elemento de un array está presente antes de intentar acceder a él, lo que ayuda a escribir código más robusto y evitar mensajes de error no deseados.

**Ejemplo: OBTENER LOS DATOS DE LAS TABLAS EN LA BASE DE DATOS**

```php
<?php
// Datos de conexión
$host = "localhost";        // Dirección del servidor de base de datos, usualmente es "localhost".
$db_user = "magux";         // Usuario para conectarse a la base de datos.
$db_password = "gAsDev=7A1B??";     // Contraseña del usuario.
$db_name = "db_magux";       // Nombre de la base de datos.
// Crear conexión
// Se instancia un nuevo objeto de la clase mysqli.
$conn = new mysqli($host, $db_user, $db_password, $db_name);
// Verificar la conexión
// $conn->connect_error es una propiedad del objeto $conn.
// Si hay un error en la conexión, esta propiedad contendrá detalles del error.
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
// Realizar la consulta SQL para obtener todas las noticias
$sql = "SELECT * FROM Noticia";
// $conn->query() es un método del objeto $conn que ejecuta la consulta SQL.
// El resultado se almacena en $result, que es otro objeto.
$result = $conn→query($sql);

// Inicializar el array para almacenar las noticias
$noticias = [];
// $result->num_rows es una propiedad del objeto $result.
// Nos dice cuántas filas fueron devueltas por la consulta SQL.
if ($result->num_rows > 0) {
 // Si hay resultados, recorremos cada fila.
 // $result->fetch_assoc() es un método que obtiene la siguiente fila como un array asociativo.
   while($row = $result->fetch_assoc()) {
      $noticias[] = $row;
   }
} else {
   echo "0 resultados";
}
// Cerrar la conexión.
// $conn->close() es un método que cierra la conexión a la base de datos.
$conn→close();
// Mostrar los títulos de las noticias
foreach ($noticias as $noticia) {
   echo $noticia["Titulo"] . "<br>";
}
?>
```

## Actividad-1: CONECTAR PHP CON UNA BASE DE DATOS Y MOSTRARLOS POR PANTALLA

**Paso 1:** Crear la Base de datos

```
CREATE DATABASE biblioteca;

USE biblioteca;
CREATE TABLE libros (
   id INT AUTO_INCREMENT PRIMARY KEY,
   titulo VARCHAR(255) NOT NULL,
   autor VARCHAR(255) NOT NULL,
   publicacion YEAR NOT NULL
);
INSERT INTO libros (titulo, autor, publicacion) VALUES
('El Gran Gatsby', 'F. Scott Fitzgerald', 1925),
('Matar a un Ruiseñor', 'Harper Lee', 1960),
('1984', 'George Orwell', 1949);
```

**Paso 2:** Conectar a la base de datos usando mysql **conectar.php:**

```
<?php
        $host = '127.0.0.1';
        $db = 'biblioteca';
        $user = 'root';
        $pass = '';
        $charset = 'utf8mb4';

        $dsn = "mysql:host=$host;dbname=$db;charset=$charset";
        $options = [
           PDO::ATTR_ERRMODE          => PDO::ERRMODE_EXCEPTION,
           PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
           PDO::ATTR_EMULATE_PREPARES   => false,
        ];
        try {
           $pdo = new PDO($dsn, $user, $pass, $options);
        } catch (\PDOException $e) {
           throw new \PDOException($e->getMessage(), (int)$e->getCode());
        }
?>
```

**Paso 3**: Realizar una consulta y mostrar los resultados **mostrar_libros.php:**

```
<?php
        require 'conectar.php';

        $stmt = $pdo->query('SELECT titulo, autor, publicacion FROM libros');
        $libros = $stmt->fetchAll();

?>
```

```
<!DOCTYPE html>
<html>
<head>
  <title>Biblioteca</title>
</head>
<body>
  <h1>Lista de Libros</h1>
  <table border="1">
    <tr>
      <th>Título</th>
      <th>Autor</th>
      <th>Año de Publicación</th>
    </tr>
    <?php foreach ($libros as $libro): ?>
            <tr>
              <td><?php echo htmlspecialchars($libro['titulo']); ?></td>
              <td><?php echo htmlspecialchars($libro['autor']); ?></td>
              <td><?php echo htmlspecialchars($libro['publicacion']); ?></td>
            </tr>
    <?php endforeach; ?>
  </table>
</body>
</html>
```

**Donde:**

1. **conectar.php**:
   - **$host**: La dirección del servidor MySQL. 127.0.0.1 se refiere al servidor local.
   - **$db**: El nombre de la base de datos a la que queremos conectarnos.
   - **$user**: El nombre de usuario de MySQL. Por defecto, es root.
   - **$pass**: La contraseña de MySQL. Si no has configurado una contraseña, déjala vacía.
   - **$conn = new mysqli($host, $user, $pass, $db)**: Crea una nueva instancia de la clase mysqli para conectarse a la base de datos con los detalles proporcionados.
   - **if ($conn->connect_error)**: Verifica si ocurrió un error durante la conexión. Si es así, muestra un mensaje de error y termina el script con die().

2. **mostrar_libros.php**:
   - **require 'conectar.php'**: Incluye el archivo de conexión para establecer la conexión a la base de datos.
   - **$sql = "SELECT titulo, autor, publicacion FROM libros"**: Define una consulta SQL para seleccionar los campos titulo, autor y publicacion de la tabla libros.
   - **$result = $conn->query($sql)**: Ejecuta la consulta y almacena el resultado en $result.
   - **if ($result->num_rows > 0)**: Comprueba si hay filas en el resultado de la consulta.
   - **while ($row = $result->fetch_assoc())**: Recorre cada fila del resultado. fetch_assoc() devuelve una fila como un array asociativo.

- **htmlspecialchars()**: Escapa los caracteres especiales en una cadena para evitar ataques de Cross-Site Scripting (XSS).
- **$result->free()**: Libera la memoria asociada con el resultado.
- **$conn->close()**: Cierra la conexión a la base de datos.

Este ejemplo cubre la conexión a una base de datos MySQL usando mysqli, ejecuta una consulta para obtener datos y muestra los resultados en una tabla HTML. Es un enfoque básico pero efectivo para muchas aplicaciones web.

## PRÁCTICA: Listar, añadir, editar y eliminar registros de una base de datos MySQL

Para ello, seguiré estos pasos:

1. Crear la base de datos y la tabla con valores iniciales.
2. Crear el archivo de conexión.
3. Crear el archivo para listar registros.
4. Crear el archivo para añadir registros.
5. Crear el archivo para editar registros.
6. Crear el archivo para eliminar registros.

**Paso 1:** Crear la base de datos y la tabla con valores iniciales

**script.sql**

```
CREATE DATABASE biblioteca;
USE biblioteca;
CREATE TABLE libros (
  id INT AUTO_INCREMENT PRIMARY KEY,
  titulo VARCHAR(255) NOT NULL,
  autor VARCHAR(255) NOT NULL,
  publicacion YEAR NOT NULL
);
INSERT INTO libros (titulo, autor, publicacion) VALUES
('El Gran Gatsby', 'F. Scott Fitzgerald', 1925),
('Matar a un Ruiseñor', 'Harper Lee', 1960),
('1984', 'George Orwell', 1949);
```

**Paso 2:** Crear el archivo de conexión

**conectar.php**

```
<?php
    // Detalles de la conexión a la base de datos
    $host = '127.0.0.1';  // Dirección del servidor MySQL
    $db = 'biblioteca';   // Nombre de la base de datos
    $user = 'root';       // Nombre de usuario de MySQL
    $pass = '';           // Contraseña de MySQL
    // Crear una nueva conexión a la base de datos usando mysqli
    $conn = new mysqli($host, $user, $pass, $db);
```

```
        // Comprobar si la conexión fue exitosa
        if ($conn->connect_error) {
            // Si la conexión falla, muestra un mensaje de error y detiene el script
            die("Conexión fallida: " . $conn->connect_error);
        }
        // Si la conexión es exitosa, no se hace nada y el script continúa
    ?>
```

**Paso 3:** Crear el archivo para listar registros

**listar.php**

```
<?php
    // Incluir el archivo de conexión a la base de datos
    require 'conectar.php';

    // Preparar una consulta SQL para seleccionar todos los registros de la tabla 'libros'
    $sql = "SELECT id, titulo, autor, publicacion FROM libros";
    $result = $conn->query($sql);
?>
<!DOCTYPE html>
<html>
<head>
  <title>Biblioteca</title>
</head>
<body>
  <h1>Lista de Libros</h1>
  <a href="añadir.php">Añadir un nuevo libro</a>
  <table border="1">
    <tr>
      <th>ID</th>
      <th>Título</th>
      <th>Autor</th>
      <th>Año de Publicación</th>
      <th>Acciones</th>
    </tr>
        <?php
        // Verificar si la consulta devolvió algún resultado
        if ($result->num_rows > 0) {
            // Recorrer los resultados y mostrarlos en una tabla HTML
            while ($row = $result->fetch_assoc()) {
                echo "<tr>";
                echo "<td>" . htmlspecialchars($row['id']) . "</td>";
                echo "<td>" . htmlspecialchars($row['titulo']) . "</td>";
                echo "<td>" . htmlspecialchars($row['autor']) . "</td>";
                echo "<td>" . htmlspecialchars($row['publicacion']) . "</td>";
                echo "<td>";
                echo "<a href='editar.php?id=" . $row['id'] . "'>Editar</a> | ";
                echo "<a href='eliminar.php?id=" . $row['id'] . "' onclick='return confirm(\"¿Estás seguro de
    eliminar este libro?\");'>Eliminar</a>";
                echo "</td>";
                echo "</tr>";
            }
        } else {
            // Si no hay resultados, mostrar un mensaje en la tabla
            echo "<tr><td colspan='5'>No hay libros en la base de datos</td></tr>";
        }
```

```
            // Liberar el conjunto de resultados
            $result->free();
            ?>
  </table>
</body>
</html>
        <?php
        // Cerrar la conexión a la base de datos
        $conn->close();
        ?>
```

**Paso 4:** Crear el archivo para añadir registros

**añadir.php**

```
        <!DOCTYPE html>
        <html>
        <head>
          <title>Añadir Libro</title>
        </head>
        <body>
          <h1>Añadir un nuevo libro</h1>
          <form action="procesar_añadir.php" method="post">
            <label for="titulo">Título:</label><br>
            <input type="text" id="titulo" name="titulo" required><br><br>
            <label for="autor">Autor:</label><br>
            <input type="text" id="autor" name="autor" required><br><br>
            <label for="publicacion">Año de Publicación:</label><br>
            <input type="number" id="publicacion" name="publicacion" required><br><br>
            <input type="submit" value="Añadir Libro">
          </form>
          <a href="listar.php">Volver a la lista de libros</a>
        </body>
        </html>
```

**procesar_añadir.php**

```
        <?php
        // Incluir el archivo de conexión a la base de datos
        require 'conectar.php';
        // Comprobar si el formulario ha sido enviado y los datos están disponibles
        if ($_SERVER["REQUEST_METHOD"] == "POST") {
          // Recoger los datos del formulario
          $titulo = $_POST['titulo'];
          $autor = $_POST['autor'];
          $publicacion = $_POST['publicacion'];
          // Preparar una declaración SQL para insertar los datos
          $sql = "INSERT INTO libros (titulo, autor, publicacion) VALUES (?, ?, ?)";
          // Preparar la declaración usando mysqli
          if ($stmt = $conn->prepare($sql)) {
            // Vincular los parámetros a la declaración
            $stmt->bind_param("ssi", $titulo, $autor, $publicacion);
            // Ejecutar la declaración
            if ($stmt->execute()) {
              // Si la ejecución es exitosa, redirigir a la lista de libros
              header("Location: listar.php");
              exit();
            } else {
```

```
            // Si hay un error en la ejecución, mostrar un mensaje de error
            echo "Error: " . $stmt->error;
        }
        // Cerrar la declaración
        $stmt->close();
    } else {
        // Si hay un error en la preparación, mostrar un mensaje de error
        echo "Error: " . $conn->error;
    }
}
// Cerrar la conexión a la base de datos
$conn->close();
?>
```

**Paso 5:** Crear el archivo para editar registros

**editar.php**

```
<?php
// Incluir el archivo de conexión a la base de datos
require 'conectar.php';
// Comprobar si se ha proporcionado un ID válido
if (isset($_GET['id']) && is_numeric($_GET['id'])) {
    // Recoger el ID del libro
    $id = $_GET['id'];
    // Preparar una declaración SQL para seleccionar el libro
    $sql = "SELECT titulo, autor, publicacion FROM libros WHERE id = ?";
    if ($stmt = $conn->prepare($sql)) {
        // Vincular el parámetro a la declaración
        $stmt->bind_param("i", $id);
        // Ejecutar la declaración
        $stmt->execute();
        // Vincular los resultados a variables
        $stmt->bind_result($titulo, $autor, $publicacion);
        // Obtener el resultado
        if ($stmt->fetch()) {
            // Si se encuentra el libro, mostrar el formulario con los datos del libro
?>
<!DOCTYPE html>
<html>
<head>
    <title>Editar Libro</title>
</head>
<body>
    <h1>Editar libro</h1>
    <form action="procesar_editar.php" method="post">
        <input type="hidden" name="id" value="<?php echo $id; ?>">
        <label for="titulo">Título:</label><br>
        <input type="text" id="titulo" name="titulo" value="<?php echo htmlspecialchars($titulo); ?>"
required><br><br>
        <label for="autor">Autor:</label><br>
        <input type="text" id="autor" name="autor" value="<?php echo htmlspecialchars($autor); ?>"
required><br><br>
```

```
    <label for="publicacion">Año de Publicación:</label><br>
    <input type="number" id="publicacion" name="publicacion" value="<?php echo
htmlspecialchars($publicacion); ?>" required><br><br>
    <input type="submit" value="Actualizar Libro">
  </form>
  <a href="listar.php">Volver a la lista de libros</a>
</body>
</html>
<?php
    } else {
      // Si no se encuentra el libro, mostrar un mensaje de error
      echo "Libro no encontrado.";
    }

    // Cerrar la declaración
    $stmt->close();
  } else {
    // Si hay un error en la preparación, mostrar un mensaje de error
    echo "Error: " . $conn->error;
  }
} else {
  // Si no se proporciona un ID válido, redirigir a la lista de libros
  header("Location: listar.php");
  exit();
}

// Cerrar la conexión a la base de datos
$conn->close();
?>
```

**procesar_editar.php**

```
<?php
// Incluir el archivo de conexión a la base de datos
require 'conectar.php';

// Comprobar si el formulario ha sido enviado y los datos están disponibles
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['id']) && is_numeric($_POST['id'])) {
  // Recoger los datos del formulario
  $id = $_POST['id'];
  $titulo = $_POST['titulo'];
  $autor = $_POST['autor'];
  $publicacion = $_POST['publicacion'];

  // Preparar una declaración SQL para actualizar los datos
  $sql = "UPDATE libros SET titulo = ?, autor = ?, publicacion = ? WHERE id = ?";

  // Preparar la declaración usando mysqli
  if ($stmt = $conn->prepare($sql)) {
    // Vincular los parámetros a la declaración
    $stmt->bind_param("ssii", $titulo, $autor, $publicacion, $id);
```

```
    // Ejecutar la declaración
    if ($stmt->execute()) {
        // Si la ejecución es exitosa, redirigir a la lista de libros
        header("Location: listar.php");
        exit();
    } else {
        // Si hay un error en la ejecución, mostrar un mensaje de error
        echo "Error: " . $stmt->error;
    }

    // Cerrar la declaración
    $stmt->close();
  } else {
    // Si hay un error en la preparación, mostrar un mensaje de error
    echo "Error: " . $conn->error;
  }
}

// Cerrar la conexión a la base de datos
$conn->close();
?>
```

**Paso 6:** Crear el archivo para eliminar registros

**eliminar.php**

```
<?php
// Incluir el archivo de conexión a la base de datos
require 'conectar.php';

// Comprobar si se ha proporcionado un ID válido
if (isset($_GET['id']) && is_numeric($_GET['id'])) {
  // Recoger el ID del libro
  $id = $_GET['id'];

  // Preparar una declaración SQL para eliminar el libro
  $sql = "DELETE FROM libros WHERE id = ?";

  // Preparar la declaración usando mysqli
  if ($stmt = $conn->prepare($sql)) {
    // Vincular el parámetro a la declaración
    $stmt->bind_param("i", $id);

    // Ejecutar la declaración
    if ($stmt->execute()) {
      // Si la ejecución es exitosa, redirigir a la lista de libros
      header("Location: listar.php");
      exit();
    } else {
      // Si hay un error en la ejecución, mostrar un mensaje de error
      echo "Error: " . $stmt->error;
    }

    // Cerrar la declaración
    $stmt->close();
  } else {
    // Si hay un error en la preparación, mostrar un mensaje de error
    echo "Error: " . $conn->error;
```

```
    }
} else {
    // Si no se proporciona un ID válido, redirigir a la lista de libros
    header("Location: listar.php");
    exit();
}

// Cerrar la conexión a la base de datos
$conn->close();
?>
```

**Explicación del Código:**

- **conectar.php**: Establece la conexión a la base de datos.
- **listar.php**: Muestra una tabla con los registros de la base de datos y enlaces para editar o eliminar cada registro.
- **añadir.php**: Muestra un formulario para añadir un nuevo libro.
- **procesar_añadir.php**: Procesa los datos enviados desde añadir.php y añade el nuevo libro a la base de datos.
- **editar.php**: Muestra un formulario con los datos del libro a editar.
- **procesar_editar.php**: Procesa los datos enviados desde editar.php y actualiza el libro en la base de datos.
- **eliminar.php**: Elimina el libro especificado por el ID en la base de datos.

Este conjunto de archivos permite gestionar una tabla libros en una base de datos MySQL, cubriendo las operaciones básicas de CRUD (Crear, Leer, Actualizar, Eliminar) con PHP y MySQLi.