Conexión y manejo de resultados en MySQL con PHP

En aplicaciones web dinámicas, el servidor PHP se comunica con una base de datos para almacenar y recuperar información de manera persistente. Como estudiamos a principio de curso, MySQL es uno de los sistemas de gestión de bases de datos más usados, y PHP ofrece varias formas de conectarse, entre ellas mysqli (MySQL Improved).

El fragmento de código que vamos a analizar, que deberemos entender para reproducirlo en nuestros desarrollos, realiza lo siguiente:

1. Conecta PHP con la base de datos MySQL.
2. Ejecuta una consulta SQL.
3. Recupera los resultados fila por fila y muestra los datos en HTML.
4. Cierra la conexión al terminar.

1.Conexión a la base de datos

```
$conn = new mysqli($servername, $username, $password, $dbname);


if ($conn->connect_error) {
   die("Conexión fallida: " . $conn->connect_error);
}
```

I. **new mysqli()**:

Crea un objeto de conexión a MySQL, utilizando la siguiente parametría:

- $servername: servidor donde está la base de datos, por ejemplo "localhost".
- $username: usuario de MySQL, por ejemplo "root".
- $password: contraseña del usuario.
- $dbname: nombre de la base de datos a usar.

Este objeto ($conn) permite ejecutar consultas y manejar resultados.

II. **Comprobación de errores de conexión:**

- connect_error es una propiedad del objeto $conn. Si ocurre algún error al conectarse, se usa:
- die("Conexión fallida: " . $conn->connect_error);
  - die() detiene la ejecución del script y muestra el mensaje.
  - Evita que el script intente ejecutar consultas si no hay conexión válida.

Nota: Nunca mostrar directamente errores de conexión en producción; podría revelar información sensible. En este diseño lo mostramos para aprender.

En lugar de die($conn->connect_error), es recomendable registrar el error en un log y mostrar un mensaje genérico al usuario. -> ESTUDIAREMOS CÓMO

2.Ejecución de una consulta SQL

```
$resultado = $conn->query($sql);
```

- $sql es una cadena de texto con la sentencia SQL que queremos ejecutar, por ejemplo:
- $sql = "SELECT id, task FROM tareas";
- $conn->query($sql) envía esa consulta al servidor de bases de datos.
- El resultado se guarda en $resultado, que es un objeto de tipo mysqli_result si la consulta devuelve filas (SELECT).

3.Recorrer los resultados y mostrar los datos

```
for ($i = 0; $i < $resultado->num_rows; $i++) {
    $row = $resultado->fetch_assoc();
    echo "<li>" . $row['id'] . " " . $row['task'] . "</li>";
}
```

I. **Número de filas**:
   - $resultado->num_rows devuelve cuántas filas devolvió la consulta.
   - Esto nos permite recorrer los resultados usando un bucle.

II. **fetch_assoc()**:
   - Devuelve la **siguiente fila** como un **array asociativo**, donde las claves son los nombres de columnas (id, task, etc.).
   - En cada iteración, $row contiene los datos de una fila.

III. **echo dentro del bucle**:
   - Genera contenido HTML dinámicamente.
   - En este caso, se crea una lista <li> con el id y la task de cada registro.
   - Resultado visual (en HTML):
   - 1 Comprar leche
   - 2 Llamar al cliente

También se puede usar un bucle while, que suele ser más elegante:

```
while($row = $resultado->fetch_assoc()){
    echo "<li>" . $row['id'] . " " . $row['task'] . "</li>";
```

```
}
```

4.Cierre de la conexión

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

- Libera recursos y cierra la conexión con la base de datos.
- Aunque PHP cierra automáticamente la conexión al terminar el script, es buena práctica hacerlo explícitamente.

¿Y si deseamos ejecutar consultas que modifican la base de datos: INSERT, UPDATE, DELETE?

Cuando ejecutamos consultas que no devuelven filas, como INSERT, UPDATE o DELETE, no necesitamos recorrer resultados. Lo importante es comprobar si la ejecución fue exitosa y, opcionalmente, cuántas filas fueron afectadas.

1) Ejemplo básico: UPDATE

```
$sql = "UPDATE tareas SET task='Comprar pan' WHERE id=1";

if ($conn->query($sql) === TRUE) {
    echo "Registro actualizado correctamente";
} else {
    echo "Error al actualizar: " . $conn->error;
}
```

1. $conn->query($sql) ejecuta la sentencia SQL.
2. Devuelve:
    - TRUE si la consulta se ejecutó correctamente (aunque no haya filas afectadas).
    - FALSE si hubo un error de ejecución.
3. $conn->error contiene el mensaje de error detallado (útil en desarrollo, pero como hemos dicho antes, mejor no mostrarlo en producción).

---

2) Ejemplo básico de INSERT

```
$sql = "INSERT INTO tareas (task) VALUES ('Comprar leche')";


if ($conn->query($sql) === TRUE) {
    echo "Registro insertado correctamente.”;
} else {
```

```
    echo "Error al insertar: " . $conn->error;
}
```

$conn->query($sql)

- Devuelve TRUE si la consulta se ejecutó correctamente.
- Devuelve FALSE si hubo algún error (por ejemplo, violación de clave primaria o error de sintaxis).

$conn->error

- Ya comentado en anteriores ejemplos

3) Ejemplo con DELETE

```
$sql = "DELETE FROM tareas WHERE id=2";
if ($conn->query($sql) === TRUE) {
    echo "Registro eliminado";
} else {
    echo "Error al eliminar: " . $conn->error;
}
```

- Igual que con UPDATE, solo necesitamos comprobar éxito o fallo.
- No hay fetch_assoc() ni bucles, porque no hay resultados que recorrer.

Aunque de momento tenemos suficiente con el conocimiento arriba desarrollado, a continuación se presenta un enlace a la documentación referente a la clase mysqli:

https://www.php.net/manual/es/book.mysqli.php