PDO (PHP Data Objects)

$mensajeId = $dbConexion -> lastInsertId();

📌 **Explicación:**

lastInsertId() es un método de **PDO (PHP Data Objects)** que se usa para obtener el ID del último registro insertado en una tabla que tenga una columna con **auto-incremento**.

🔎 **Cómo funciona**

1. **Después de una inserción (INSERT INTO) en una tabla con una clave primaria AUTO_INCREMENT**, este método devuelve el ID generado en esa última inserción.
2. **Si no se ha hecho una inserción**, devolverá 0 o una cadena vacía ("").
3. **Si la tabla no tiene AUTO_INCREMENT**, lastInsertId() no funcionará correctamente.

✅ **Ejemplo práctico**

🔹 **Escenario: Insertar un usuario y obtener su id**

```
try {
   // Conectar a la base de datos con PDO
   $dbConexion = new PDO("mysql:host=localhost;dbname=mi_base", "usuario", "contraseña");
   $dbConexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

   // Insertar un nuevo usuario
   $sql = "INSERT INTO usuarios (nombre, email) VALUES (:nombre, :email)";
   $stmt = $dbConexion->prepare($sql);
   $stmt->execute([
      ':nombre' => 'Juan Pérez',
      ':email'  => 'juan@example.com'
```

```
    ]);

    // Obtener el ID del último usuario insertado
    $usuarioId = $dbConexion->lastInsertId();

    echo "El ID del usuario recién insertado es: " . $usuarioId;

} catch (PDOException $e) {
    echo "Error: " . $e->getMessage();
}
```

🔥 **Casos de uso de lastInsertId()**

📌 **Escenario**	⚡ **¿lastInsertId() funciona?**
🚀 **Insertar un nuevo usuario en una tabla con AUTO_INCREMENT**	✅ Sí, devuelve el ID generado
🔄 **Insertar manualmente un id en una tabla sin AUTO_INCREMENT**	❌ No, devuelve una cadena vacía
🚫 **No se ha hecho ningún INSERT en la sesión actual**	❌ No, devuelve 0 o ""

⚠️ **Cosas a tener en cuenta**

1. **Solo funciona con columnas AUTO_INCREMENT**.
2. **No devuelve múltiples IDs** en caso de una inserción masiva (INSERT INTO ... VALUES (...), (...)).
3. **Si usas transacciones (BEGIN TRANSACTION y COMMIT)**, el ID solo se obtiene después de confirmar (COMMIT).

🎯 **Resumen**

- lastInsertId() **recupera el último ID generado por una inserción en una tabla con AUTO_INCREMENT**.

- Es útil para **relacionar datos** en otras tablas (ej. insertar un usuario y luego su perfil).
- Si no hay inserción previa o la tabla no tiene AUTO_INCREMENT, devolverá 0 o "".

¿Tienes alguna duda específica sobre su uso? 🚀

Tu código tiene un **error en la validación** al comprobar si un nombre ya existe en la tabla ofertas.

❌ Errores en tu código

1. **La comparación if($checkNombre > 0) es incorrecta**
    - $checkNombre es un objeto PDOStatement, no un número.
    - Para verificar si hay resultados, **debes usar rowCount() o fetch()**.
2. **Uso incorrecto de LIKE con bindParam()**
    - bindParam() no concatena automáticamente los comodines (%).
    - Debes usar **bindValue()** o concatenar manualmente.

✅ Corrección

```
// Comprobar si el nombre ya existe en la tabla 'ofertas'
$checkNombre = $conexion->prepare("SELECT COUNT(*) FROM ofertas WHERE nombre = ?");
$checkNombre->bindParam(1, $datos['nombre']); // Asegurar que el nombre es exacto
$checkNombre->execute();

// Obtener el número de filas encontradas
$nombreExiste = $checkNombre->fetchColumn();

if ($nombreExiste > 0) {
   salidaDatos('Ya existe una actividad con el nombre: ' . $datos['nombre'] . '.',
```

```
        array('Content-Type: application/json', 'HTTP/1.1 400 Bad Request'));
    exit;
}
```

🔥 **Explicación de los cambios**

Error Original	Corrección
if($checkNombre > 0) ❌	$checkNombre->fetchColumn() > 0 ✅
bindParam(1, $datos['nombre']) con LIKE ❌	Se usa COUNT(*) y = ? ✅
No se obtiene el resultado ❌	fetchColumn() obtiene la cantidad de registros ✅

🚀 **Opción con LIKE (si necesitas coincidencias parciales)**

Si quieres buscar nombres que **contengan** el valor (%nombre%), usa:

```
$checkNombre = $conexion->prepare("SELECT COUNT(*) FROM ofertas WHERE nombre LIKE ?");
$nombreLike = "%" . $datos['nombre'] . "%";
$checkNombre->bindValue(1, $nombreLike); // `bindValue()` permite concatenaciones
$checkNombre->execute();

if ($checkNombre->fetchColumn() > 0) {
    salidaDatos('Ya existe una actividad con un nombre similar a: ' . $datos['nombre'] . '.',
        array('Content-Type: application/json', 'HTTP/1.1 400 Bad Request'));
    exit;
}
```

🎯 **Resumen**

1. Usa fetchColumn() en vez de rowCount() o directamente en $checkNombre.

2. Si usas LIKE, **concatena % antes de ejecutar**.
3. Usa bindValue() en lugar de bindParam() si necesitas modificar el valor antes de ejecutar la consulta.

Así tu código validará correctamente si el nombre ya existe antes de insertar. 🚀