

# Reporte de Vulnerabilidad: SQL Injection

## Nombre de la vulnerabilidad:

SQL Injection (Inyección SQL)

## ID de prueba:

1' OR '1'='1

## Descripción del problema:

Se ha detectado una vulnerabilidad de tipo **inyección SQL** en el sistema. Esta falla ocurre cuando los datos ingresados por el usuario (como el "ID") no se validan correctamente y se usan directamente en una consulta a la base de datos.

En esta prueba, se utilizó el siguiente valor como ID:

bash

CopiarEditar

1' OR '1'='1

Este valor **modifica la consulta SQL** original, haciendo que se devuelvan **todos los registros de la base de datos**, en lugar de solo uno.

## Resultados obtenidos:

La siguiente información fue revelada, lo cual **no debería ser accesible solo con un ID**:

ID ingresado	Nombre	Apellido
1' OR '1'='1	admin	admin
1' OR '1'='1	Gordon	Brown
1' OR '1'='1	Hack	Me
1' OR '1'='1	Pablo	Picasso

1' OR '1'='1      Bob      Smith

Esto indica que la aplicación está ejecutando el siguiente tipo de consulta insegura:

sql

CopiarEditar

```
SELECT * FROM users WHERE id = '1' OR '1'='1';
```

La condición '1'='1' **siempre es verdadera**, por lo tanto se devuelven **todos los usuarios**.

The screenshot shows the DVWA web application interface. At the top is the DVWA logo. On the left is a sidebar menu with various security challenges. The main content area is titled 'Vulnerability: SQL Injection'. It features a 'User ID' input field and a 'Submit' button. Below the input field, the application displays the results of the query in red text, showing five user records: admin, Gordon Brown, Hack Me, Pablo Picasso, and Bob Smith. At the bottom, there is a 'More Information' section with links to external resources about SQL injection.

**Vulnerability: SQL Injection**

User ID:

ID: 1' OR '1'='1  
First name: admin  
Surname: admin

ID: 1' OR '1'='1  
First name: Gordon  
Surname: Brown

ID: 1' OR '1'='1  
First name: Hack  
Surname: Me

ID: 1' OR '1'='1  
First name: Pablo  
Surname: Picasso

ID: 1' OR '1'='1  
First name: Bob  
Surname: Smith

**More Information**

- [https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- [https://owasp.org/www-community/attacks/SQL\\_injection](https://owasp.org/www-community/attacks/SQL_injection)
- <https://bobby-tables.com/>

## Riesgos:

- Acceso no autorizado a datos personales.
- Robo o manipulación de datos.
- Posible compromiso total de la base de datos.

- Incumplimiento de leyes de protección de datos (como GDPR o Ley Federal de Protección de Datos Personales en México).

## Recomendaciones:

1. **Usar consultas preparadas (prepared statements)** en lugar de concatenar texto directamente en SQL.
2. **Validar y sanitizar todas las entradas del usuario.**
3. **Limitar los permisos del usuario de base de datos** utilizado por la aplicación.
4. **Implementar un firewall de aplicaciones web (WAF).**
5. **Realizar pruebas de seguridad regularmente** (pentesting y análisis estático).

## Conclusión:

Esta prueba demuestra que el sistema es **vulnerable a inyecciones SQL**. Urge tomar medidas correctivas para proteger los datos y evitar posibles ataques.