

# SQL Injection

---



**Christian Wenz**

@chwenz



# Demo



Logging into the shop, without a login



# Famous SQL Injection Incidents



```
SELECT * FROM users  
WHERE username='admin'  
AND password='s€0R3T'=''
```

SQL, We Have a Problem!



```
$sql = "SELECT * FROM news  
WHERE id = '{$_GET[id]}'";
```

```
$sql = "SELECT * FROM news  
WHERE id = {$_GET[id]}";
```

```
$sql = sprintf('SELECT * FROM  
news WHERE id=%s',  
    $_GET['id']);
```

◀ Classical SQL injection

◀ Even worse – escaping quotes  
does not help here

◀ String concatenation in disguise





Error message facilitate SQL injection

Absence of error messages does not prevent SQL injection



# Preventing SQL Injection

**Validate Input**

**Escape Output**

**Prepared Statements**

**Least-Privileged Accounts**



```
mysqli_real_escape_string(  
    $db, $value);  
$db->real_escape_string(  
    $value);
```

```
pg_escape_string(  
    $db, $value);
```

```
SQLITE3::escapeString(  
    $value);
```

◀ MySQL

◀ PostgreSQL

◀ SQLite





```
$db = new PDO(...);  
$sql = 'INSERT INTO searches (term) VALUES (:term)';  
$cmd = $db->prepare($sql);  
$cmd->bindParam(':term', $term);  
$cmd->execute();
```

---

## Prepared Statements

### PDO (PHP Data Objects)



```
$db = new mysqli(...);           //mysqli_connect()
$sql = 'INSERT INTO searches (term) VALUES (?)';
$stmt = $db->prepare($sql);       //mysqli_prepare()
$stmt->bind_param('s', $term);    //mysqli_stmt_bind_param()
$stmt->execute();                  //mysqli_stmt_execute()
```

# Prepared Statements

## MySQL



```
$db = pg_connect(...);  
$sql = 'INSERT INTO searches (term) VALUES ($1)';  
pg_prepare($db, 'QueryName', $sql);  
pg_execute($db, 'QueryName', [$term]);
```

---

# Prepared Statements

## PostgreSQL



```
$db = new SQLite3(...);  
$sql = 'INSERT INTO searches (term) VALUES (:term)';  
$cmd = $db->prepare($sql);  
$cmd->bindValue(':term', $term);  
$cmd->execute();
```

## Prepared Statements

### SQLite



```
$db = oci_connect(...);  
$sql = 'INSERT INTO searches (term) VALUES (:term)';  
$cmd = oci_parse($db, $sql);  
oci_bind_by_name($cmd, ':term', $term);  
oci_execute($cmd);
```

---

## Prepared Statements

### Oracle



```
$db = sqlsrv_connect(...);  
$sql = 'INSERT INTO searches (term) VALUES (?)';  
$cmd = sqlsrv_prepare($db, $sql, [&$term]);  
sqlsrv_execute($cmd);
```

## Prepared Statements

### Microsoft SQL Server

ext/sqlsrv from <https://www.microsoft.com/en-us/download/details.aspx?id=20098>



# Summary



SQL mixes data and commands – you shouldn't

Use prepared statements, or (as a last resort) escaping functions

