How can I prevent SQL injection in PHP?

If user input is inserted without modification into an SQL query, then the application becomes vulnerable to SQL injection, like in the following example:

```
$unsafe_variable = $_POST['user_input'];

mysql_query("INSERT INTO `table` (`column`) VALUES ('$unsafe_variable')");
```

That's because the user can input something like `value'); DROP TABLE table;--`, and the query becomes:

```
INSERT INTO `table` (`column`) VALUES('value'); DROP TABLE table;--')
```

What can be done to prevent this from happening?

Use prepared statements and parameterized queries. These are SQL statements that are sent to and parsed by the database server separately from any parameters. This way it is impossible for an attacker to inject malicious SQL.
You basically have two options to achieve this:

1. Using PDO (for any supported database driver):

```
$stmt = $pdo->prepare('SELECT * FROM employees WHERE name = :name');

$stmt->execute(array('name' => $name));

foreach ($stmt as $row) {
    // do something with $row
}
```

2. Using MySQLi (for MySQL):

```
$stmt = $dbConnection->prepare('SELECT * FROM employees WHERE name = ?');
$stmt->bind_param('s', $name); // 's' specifies the variable type => 'string'

$stmt->execute();

$result = $stmt->get_result();
while ($row = $result->fetch_assoc()) {
    // do something with $row
}
```

If you're connecting to a database other than MySQL, there is a driver-specific second option that you can refer to (e.g. `pg_prepare()` and `pg_execute()` for PostgreSQL). PDO is the universal option.

Correctly setting up the connection

Note that when using `PDO` to access a MySQL database *real* prepared statements are **not used by default**. To fix this you have to disable the emulation of prepared statements. An example of creating a connection using PDO is:

```
$dbConnection = new PDO('mysql:dbname=dbtest;host=127.0.0.1;charset=utf8', 'user', 'pass');

$dbConnection->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
$dbConnection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```