← PHP MySQL 创建表 PHP 图像处理 →

PHP MySQL 预处理语句

预处理语句对于防止 MySQL 注入是非常有用的。

预处理语句及绑定参数

预处理语句用于执行多个相同的 SQL 语句，并且执行效率更高。

预处理语句的工作原理如下：

1. 预处理：创建 SQL 语句模板并发送到数据库。预留的值使用参数 "?" 标记 。例如：

```
INSERT INTO MyGuests (firstname, lastname, email) VALUES(?, ?, ?)
```

2. 数据库解析，编译，对SQL语句模板执行查询优化，并存储结果不输出。
3. 执行：最后，将应用绑定的值传递给参数（"?" 标记），数据库执行语句。应用可以多次执行语句，如果参数的值不一样。

相比于直接执行SQL语句，预处理语句有两个主要优点：

- 预处理语句大大减少了分析时间，只做了一次查询（虽然语句多次执行）。
- 绑定参数减少了服务器带宽，你只需要发送查询的参数，而不是整个语句。
- 预处理语句针对SQL注入是非常有用的，因为参数值发送后使用不同的协议，保证了数据的合法性。

MySQLi 预处理语句

以下实例在 MySQLi 中使用了预处理语句，并绑定了相应的参数:

实例 (MySQLi 使用预处理语句)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// 创建连接
$conn = new mysqli($servername, $username, $password, $dbname);
// 检测连接
if ($conn->connect_error) {
die("连接失败: " . $conn->connect_error);
}
// 预处理及绑定
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $firstname, $lastname, $email);
```

```
// 设置参数并执行
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();
$firstname = "Mary";
$lastname = "Moe";
$email = "mary@example.com";
$stmt->execute();
$firstname = "Julie";
$lastname = "Dooley";
$email = "julie@example.com";
$stmt->execute();
echo "新记录插入成功";
$stmt->close();
$conn->close();
?>
```

解析以下实例的每行代码:

```
"INSERT INTO MyGuests (firstname, lastname, email) VALUES(?, ?, ?)"
```

在 SQL 语句中，我们使用了问号 (?)，在此我们可以将问号替换为整型，字符串，双精度浮点型和布尔值。

接下来，让我们来看下 bind_param() 函数：

```
$stmt->bind_param("sss", $firstname, $lastname, $email);
```

该函数绑定了 SQL 的参数，且告诉数据库参数的值。 "sss" 参数列处理其余参数的数据类型。s 字符告诉数据库该参数为字符串。

参数有以下四种类型:

- i - integer（整型）
- d - double（双精度浮点型）
- s - string（字符串）
- b - BLOB（binary large object:二进制大对象）

每个参数都需要指定类型。

通过告诉数据库参数的数据类型，可以降低 SQL 注入的风险。

注意： 如果你想插入其他数据（用户输入），对数据的验证是非常重要的。

PDO 中的预处理语句

以下实例我们在 PDO 中使用了预处理语句并绑定参数:

实例 (PDO 使用预处理语句)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";
try {
$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
// 设置 PDO 错误模式为异常
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
// 预处理 SQL 并绑定参数
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email)
VALUES (:firstname, :lastname, :email)");
$stmt->bindParam(':firstname', $firstname);
$stmt->bindParam(':lastname', $lastname);
$stmt->bindParam(':email', $email);
// 插入行
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();
// 插入其他行
$firstname = "Mary";
$lastname = "Moe";
$email = "mary@example.com";
$stmt->execute();
// 插入其他行
$firstname = "Julie";
$lastname = "Dooley";
$email = "julie@example.com";
$stmt->execute();
echo "新记录插入成功";
}
catch(PDOException $e)
{
echo "Error: " . $e->getMessage();
}
$conn = null;
?>
```

← PHP MySQL 创建表 PHP 图像处理 →

点我分享笔记