Atitit 防注入 sql参数编码法

目录

[1.2. 提升可读性pg\_escape\_literal — 转义文字以插入文本字段 1](#_Toc27695)

[1.2.1. 说明 1](#_Toc24585)

[1.3. 推荐pg\_escape\_string() 转义 text/char 数据类型的字符串 2](#_Toc2748)

[1.3.1. 说明 2](#_Toc15314)

[1.4. 2. mysql\_real\_escape\_string() (推荐指数4) 2](#_Toc2354)

[1.5. 魔术引用 (推荐指数3)addslashes 3](#_Toc25244)

预先处理法固然好，但是毕竟麻烦，调试不方便

#### Escaping

通过使用mysqli 扩展，你可以利用 mysqli\_real\_escape\_string() 函数来 escape包含在 SQL 查询中的所有数据。PostgresSQL 的 pgsql 扩展提供 pg\_escape\_bytea()、 pg\_escape\_identifier()、 pg\_escape\_literal() 和 pg\_escape\_string() 函数

基于以上原因，并不推荐使用 escaping。它可以用来救急，如果你用来抽象的数据库程序库不强制参数绑定就能进行 SQL 查询，可能就需要使用它。否则你应该避免使用 escape。它很混乱，容易出错，而且因数据库扩展不同而存在差异。

## 提升可读性pg\_escape\_literal — 转义文字以插入文本字段

缺点，必须要conn才可。。

pg\_escape\_literal — 转义文字以插入文本字段

### 说明

****pg\_escape\_literal**** （[ resource $connection ]， string $data ）：字符串

**pg\_escape\_literal（）**转义用于查询PostgreSQL数据库的文字。它以PostgreSQL格式返回转义文字。**pg\_escape\_literal（）**在数据前后添加引号。用户不应添加引号。建议使用此函数而不是[pg\_escape\_string（）](https://php.golaravel.com/function.pg-escape-string.html)。如果列的类型是bytea，[则](https://php.golaravel.com/function.pg-escape-bytea.html)必须[改用pg\_escape\_bytea（）](https://php.golaravel.com/function.pg-escape-bytea.html)。对于转义标识符（例如表，字段名称），必须使用[pg\_escape\_identifier（）](https://php.golaravel.com/function.pg-escape-identifier.html)。

## 推荐pg\_escape\_string() 转义 text/char 数据类型的字符串

### 说明

****pg\_escape\_string**** ( string $data ) : string

****pg\_escape\_string()**** 转义 text/char 数据类型的字符串，返回转义后的字符串。建议用此函数替代 [addslashes()](https://php.golaravel.com/function.addslashes.html)。

****Note****:

本函数需要 PostgreSQL 7.2 或以上版本。

## ****2. mysql\_real\_escape\_string() (推荐指数4)****

由于addslashes()不检测字符集，所以有宽字节注入风险，所以php中添加了这个函数。  
这个函数本来是mysql的扩展，但是由于存在宽字节的问题，php基于mysql的扩展开发了此函数。

gbk宽字符漏洞导致的sql注入  
[https://www.91ri.org/8611.html](https://www.91ri.org/8611.html" \t "https://segmentfault.com/a/_blank)  
[http://www.cnblogs.com/suihui...](http://www.cnblogs.com/suihui/archive/2012/09/20/2694751.html" \t "https://segmentfault.com/a/_blank)

mysql\_real\_escape\_chars()是mysql\_escape\_chars()的替代用法。  
与addslashes()相比，不仅会将' " NOL(ascii的0)转义，还会把r n进行转义。同时会检测数据编码。  
按php官方的描述，此函数可以安全的用于mysql。

此函数在使用时会使用于数据库连接(因为要检测字符集)，并根据不同的字符集做不同的操作。如果当前连接不存在，刚会使用上一次的连接。

mysql\_real\_escape\_string()防注入详解  
此方法在php5.5后不被建议使用，在php7中废除。  
参考：[https://segmentfault.com/q/10...](https://segmentfault.com/q/1010000006015688)

## ****魔术引用 (推荐指数3)****addslashes

1. 在防注入方面，addslashes()可以防止掉大多数的注入，但是此函数并不会检查变量的编码，当使用例如中文gbk的时候，由于长度比较长，会将某些gbk编码解释成两 .  
   addslashes()与stripslashes()是功能相反的函数。

addslashes()用于对变量中的' " 和NULL添加斜杠，用于避免传入sql语句的参数格式错误，同时如果有人注入子查询，通过加可以将参数解释为内容，而非执行语句，避免被mysql执行。

不过，addslashes()添加的只在php中使用，并不会写入mysql中。  
那么，tripslashes()的作用是将加了的php变量去掉，由于不会写入mysql中，所以从mysql查询出来的内容不需要再tripslashes()。

在防注入方面，addslashes()可以防止掉大多数的注入，但是此函数并不会检查变量的编码，当使用例如中文gbk的时候，由于长度比较长 ，会将某些gbk编码解释成两个ascii编码，造成新的注入风险(俗称宽字节注入)。见下面2。

如果从网页表单、php、mysql都使用utf8编码，则没有这个问题。  
基于此函数的风险，并不建议使用，推荐使用下面3中的方法。

## 为什么不用魔术引号

**Warning**

本特性已自 PHP 5.3.0 起废弃并将自 PHP 5.4.0 起移除。

* 可移植性 编程时认为其打开或并闭都会影响到移植性。可以用 [get\_magic\_quotes\_gpc()](mk:@MSITStore:C:\\迅雷下载\\offline chm\\php_manual_zh.chm::/res/function.get-magic-quotes-gpc.html) 来检查是否打开，并据此编程。
* 性能 由于并不是每一段被转义的数据都要插入数据库的，如果所有进入 PHP 的数据都被转义的话，那么会对程序的执行效率产生一定的影响。在运行时调用转义函数（如 [addslashes()](mk:@MSITStore:C:\\迅雷下载\\offline chm\\php_manual_zh.chm::/res/function.addslashes.html)）更有效率。 尽管 php.ini-dist 默认打开了这个选项，但是 php.ini-recommended 默认却关闭了它，主要是出于性能的考虑。
* 不便 由于不是所有数据都需要转义，在不需要转义的地方看到转义的数据就很烦。比如说通过表单发送邮件，结果看到一大堆的 \'。针对这个问题，可以使用 [stripslashes()](mk:@MSITStore:C:\\迅雷下载\\offline chm\\php_manual_zh.chm::/res/function.stripslashes.html) 函数处理。

## 为什么要用魔术引号

**Warning**

本特性已自 PHP 5.3.0 起废弃并将自 PHP 5.4.0 起移除。

* 没有理由再使用魔术引号，因为它不再是 PHP 支持的一部分。 不过它帮助了新手在不知不觉中写出了更好（更安全）的代码。 但是在处理代码的时候，最好是更改你的代码而不是依赖于魔术引号的开启。 为什么这个功能存在？是为了阻止[SQL 注入](mk:@MSITStore:C:\\迅雷下载\\offline chm\\php_manual_zh.chm::/res/security.database.sql-injection.html)。 在今天，开发者能够更好得意识到了安全问题，并最终使用数据库转移机制或者 prepared 语句来取代魔术引号功能。

PostgreSQL 函数 « PHP Manual \_ PHP 中文手册.html