

# What are the differences between addslashes(), mysql\_escape\_string() and mysql\_real\_escape\_string()

addslashes() escapes single quote ('), double quote ("), backslash (\) and NUL (\x00).

mysql\_escape\_string() and mysql\_real\_escape\_string() escapes the characters above plus: CR (\r), LF (\n) and EOF (\x1a). Apparently (according to the manual), MySQL wants these characters escaped too, but my experiment shows otherwise (i.e. MySQL doesn't care if these characters are in a string).

Suppose:

```
$value = 'bar'; // 'ba' and then CR-LF and then 'r'
```

print "insert into pairs values ('foo', '" . addslashes(\$value) . "')" gives:

```
insert into pairs values ('foo', 'ba\r\nr')
```

print "insert into pairs values ('foo', '" . mysql\_real\_escape\_string(\$value) . "')" gives:

```
insert into pairs values ('foo', 'bar')
```

In this case, the execution result should be the same, but the statement itself is different.

For other EOF, the execution result and statement are identical for both functions.

mysql\_real\_escape\_string() is available on PHP 4.3.0 or above. mysql\_escape\_string() is deprecated and you should use mysql\_real\_escape\_string() instead, as it takes the current character set into account when escaping characters.

addslashes() should be enough for single-byte strings. For multi-byte strings though, mysql\_real\_escape\_string() does provide better security. See [this article](#) for details.

PHP manual on:

- [addslashes](#)
  - [mysql\\_escape\\_string](#)
  - [mysql\\_real\\_escape\\_string](#)
- 

Revision #1

Created Fri, Jun 22, 2007 4:38 PM by Chan, Wing Kai

Updated Fri, Jun 22, 2007 6:03 PM by Chan, Wing Kai