php

- Downloads
- Documentation
- Get Involved
- Help

Search

php[tek] 2018

Keyboard Shortcuts

?

This help

j	Next menu item
k	Previous menu item
g p	Previous man page
g n	Next man page
G	Scroll to bottom
g g	Scroll to top
g h	Goto homepage
g s	Goto search (current page)
/	Focus search box

mysql_create_db »
« mysql_close

- PHP Manual
- Function Reference
- Database Extensions
- Vendor Specific Database Extensions
- MySQL
- MySQL (Original)
- MySQL Functions

Change language: English

Edit Report a Bug

mysql_connect

(PHP 4, PHP 5)

mysql_connect — Open a connection to a MySQL Server

Warning

This extension was deprecated in PHP 5.5.0, and it was removed in PHP 7.0.0. Instead, the MySQLi or PDO_MySQL extension should be used. See also MySQL: choosing an API guide and related FAQ for more information. Alternatives to this function include:

- mysqli_connect()

- PDO::__construct()

Description ¶

resource **mysql_connect** ([string `$server` = ini_get("mysql.default_host") [, string `$username` = ini_get("mysql.default_user") [, string `$password` = ini_get("mysql.default_password") [, bool `$new_link` = **`FALSE`** [, int `$client_flags` = 0]]]]])

Opens or reuses a connection to a MySQL server.

Parameters ¶

`server`

The MySQL server. It can also include a port number. e.g. "hostname:port" or a path to a local socket e.g. ":/path/to/socket" for the localhost.

If the PHP directive mysql.default_host is undefined (default), then the default value is 'localhost:3306'. In SQL safe mode, this parameter is ignored and value 'localhost:3306' is always used.

`username`

The username. Default value is defined by mysql.default_user. In SQL safe mode, this parameter is ignored and the name of the user that owns the server process is used.

`password`

The password. Default value is defined by mysql.default_password. In SQL safe mode, this parameter is ignored and empty password is used.

`new_link`

If a second call is made to **mysql_connect()** with the same arguments, no new link will be established, but instead, the link identifier of the already opened link will be returned. The `new_link` parameter modifies this behavior and makes **mysql_connect()** always open a new link, even if **mysql_connect()** was called before with the same parameters. In SQL safe mode, this parameter is ignored.

`client_flags`

The `client_flags` parameter can be a combination of the following constants: 128 (enable *LOAD DATA LOCAL* handling), **`MYSQL_CLIENT_SSL`**, **`MYSQL_CLIENT_COMPRESS`**, **`MYSQL_CLIENT_IGNORE_SPACE`** or **`MYSQL_CLIENT_INTERACTIVE`**. Read the section about MySQL client constants for further information. In SQL safe mode, this parameter is ignored.

Return Values ¶

Returns a MySQL link identifier on success or **`FALSE`** on failure.

Changelog ¶

Version	Description
5.5.0	This function will generate an **E_DEPRECATED** error.

Examples ¶

Example #1 mysql_connect() example

```
<?php
$link = mysql_connect('localhost', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
echo 'Connected successfully';
mysql_close($link);
?>
```

Example #2 mysql_connect() example using *hostname:port* syntax

```
<?php
// we connect to example.com and port 3307
$link = mysql_connect('example.com:3307', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
echo 'Connected successfully';
mysql_close($link);

// we connect to localhost at port 3307
$link = mysql_connect('127.0.0.1:3307', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
echo 'Connected successfully';
mysql_close($link);
?>
```

Example #3 mysql_connect() example using ":/path/to/socket" syntax

```
<?php
// we connect to localhost and socket e.g. /tmp/mysql.sock

// variant 1: omit localhost
$link = mysql_connect(':/tmp/mysql', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
echo 'Connected successfully';
mysql_close($link);
```

```
// variant 2: with localhost
$link = mysql_connect('localhost:/tmp/mysql.sock', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
echo 'Connected successfully';
mysql_close($link);
?>
```

Notes ¶

Note:

Whenever you specify "localhost" or "localhost:port" as server, the MySQL client library will override this and try to connect to a local socket (named pipe on Windows). If you want to use TCP/IP, use "127.0.0.1" instead of "localhost". If the MySQL client library tries to connect to the wrong local socket, you should set the correct path as in your PHP configuration and leave the server field blank.

Note:

The link to the server will be closed as soon as the execution of the script ends, unless it's closed earlier by explicitly calling mysql_close().

Note:

You can suppress the error message on failure by prepending a @ to the function name.

Note:

Error "Can't create TCP/IP socket (10106)" usually means that the variables_order configure directive doesn't contain character *E*. On Windows, if the environment is not copied the *SYSTEMROOT* environment variable won't be available and PHP will have problems loading Winsock.

See Also ¶

- mysql_pconnect() - Open a persistent connection to a MySQL server
- mysql_close() - Close MySQL connection

add a note

User Contributed Notes 36 notes

up
down
3

***Steve* ¶**

10 years ago

```
The too many connections issue can be due to several problems.

1. you are using pconnect. This can tie up many connections and is not really needed for
MySQL as new connections are really fast.

2. Apache children are hanging around for too long - combine this with pconnect and you
have recipe for disaster.

Suggestions: reduce the amount of time apache child processes stay connected to the client
and how many connections before they are killed off. And don't use pconnect.
```

up
down
1

***Harouk* ¶**

7 years ago

```
If you encounter speed problems using this command to a distant server, you can add the
line "skip-name-resolve" in your my.cnf to fix it.
```

up
down
0

***nicodenboer at yahoo dot com* ¶**

5 years ago

```
Be carefull here if you use utf8.

The file db.opt of your database should contain the following lines:
default-character-set=utf8
default-collation=utf8_general_ci

It means that your database is created to use the utf8 characterset.
One way to accomplish this is:
CREATE DATABASE my_database DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;

Then, after connecting to it from PHP you should use:
mysql_set_charset("UTF8", $connection);

If you don't do this, you will get ugly problems in case other software is reading and
writing to the same database!!!!!!
```

up
down
-2

***cory dot mawhorter gmail.com* ¶**

8 years ago

```
Hopefully this saves someone some grief.

My dev computer is windows and runs wampserver.  I have frequent problems with PHP being
unable to connect to MySQL after periods of extreme DB activity.
```

```
Long story short, it was because I was not running mysql via named-pipes and Windows was
running out of available ports to serve PHP.  Apparently, on windows, you have 5000 ports
to work with and once they are opened, they remain so for 120 seconds before being
released.  This causes problems with mysql/networking because a new port is requested for
each connection.

You can read more about the problem at:
(Link too long and had to be broken up)
http://dev.mysql.com/doc/refman/5.0/en
/can-not-connect-to-server.html#can-not-connect-to-server-on-windows
?>

Since mysql is on localhost, I can just enable named-pipes (which is how you should have
mysql setup if you don't need networking) to get around the problem instead of the
workaround listed on that page.

For details, see:
http://dev.mysql.com/tech-resources
/articles/securing_mysql_windows.html
```

up
down
-3
abelcheung at gmail dot com ¶
7 years ago

```
Note that named pipe on Windows is unusable since PHP 5.3, and TCP connection shall be
used even in localhost.
```

up
down
-3
rui dot batista at netcabo dot pt ¶
11 years ago

```
Ever wonder what "default username" is?
<?php
$link = mysql_connect() or die(mysql_error());
$result = mysql_query("SELECT SESSION_USER(), CURRENT_USER();");
$row = mysql_fetch_row($result);
echo "SESSION USER: ", $row[0], "<br>\n";
echo "CURRENT USER: ", $row[1], "<br>\n";
?>
Both are ODBC@localhost in my win2k install, so my advice for windows is:
- create a MySQL user named ODBC with no password
- add localhost to ODBC user [right-click ODBC]
- set schema previleges to ODBC@localhost
- use mysql_connect() with no parms, or do not use ;)
This turns to work also with odbc_connect:
odbc_connect("myDSN", "", "")
```

up

down
-3
sky dot sama dot remove dot dots at Gmail dot com ¶
11 years ago

```
In case anyone else is getting "Client does not support authentication protocol requested
by server; consider upgrading MySQL client" error. The problem is the new password hashing
method used by MySQL >= 4.1 mentioned below.

Either update your PHP to v5 where the new password hashing is supported or use
old_password() in MySQL 4.1.

FROM: http://www.digitalpeer.com/id/mysql

UPDATE mysql.user SET password=old_password("youroldhashpassword") WHERE user
='youruserid' and host ='yourhost'

then do

FLUSH PRIVILEGES
```

up
down
-4
Peter Robinett ¶
10 years ago

```
The use of mysql connections can become tricky with objects. I am using mysql_connect() in
a database class I wrote and the class destructor calls mysql_close. Because I have
several of these database objects, mysql_connect reuses existing connections. This is fine
except when the script reaches the end of execution and PHP's garbage collection calls
all the objects' __destruct() functions. mysql_close() throws a warning that the
connection is invalid, in my case for one object. This is happening with objects which use
an existing connection, as the connection has already been closed. I solved the problem by
forcing mysql_connect() to create a new connection each time. This is not efficient but is
sufficient for my purposes for now.

I wouldn't say this is a bug per-se, but it's something to look out for. I imagine using
mysqli is the ultimate solution...
```

up
down
-4
trev at dedicate dot co dot uk ¶
5 years ago

```
A little note if your scripts sleep a lot, you want to run exactly the same SQL statement
2+ times and you have the "MySQL has gone away" error a lot.

Try setting the 4th parameter to TRUE as it seems sometimes PHP doesn't spot that resource
ID x which it used for the last identical lookup is now dud and so tries to use it, thus
bypassing tests such as is_resource() and causing a failure.
```

```
This is for when mysql_ping() doesn't work for your situation of course.
```

up
down
-5
martinnitram at excite dot com ¶
14 years ago

```
to use load data local infile function from mysql (at mysql 4.0.16, php 4.3.3), set fifth
parameter of mysql_connect() to CLIENT_LOCAL_FILES(128), which based on MYSQL C API ( also
mysql server support load file, check by "show variables like 'local_infile' ")

Thank  'phpweb at eden2 dot com' to point this out
```

up
down
-5
Graham_Rule at ed dot ac dot uk ¶
11 years ago

```
The addition of entries to httpd.conf to stop .inc files being served by Apache is
certainly useful and to be recommended.

But it doesn't change the fact that these files have to be readable by Apache so that the
PHP processor can get at them.

As long as your don't have multiple, possibly untrusted, users on your machine then that's
OK.  But when you are running a large multi-user service with thousands of users its
always possible that one of them will look at your .inc files and take a note of the
passwords you have in them.  They could even copy them into their own scripts and modify
your databases!

Even if local users are trusted, there is always the possibility of a rogue script (PHP or
some nastier language) being installed by an ignorant user.  That script might then read
your .inc files (whether or not they are in the web publishing tree) and expose your
password.
```

up
down
-5
brinca at substancia dot com ¶
11 years ago