

# Re: PDO: Switch database user without reopening connection

---

*Source:* <http://coding.derkeiler.com/Archive/PHP/comp.lang.php/2008-05/msg01062.html>

---

- *From:* Erwin Moller <[Since\\_humans\\_read\\_this\\_I\\_am\\_spammed\\_too\\_much@xxxxxxxxxxxxxxxxxxxxx](mailto:Since_humans_read_this_I_am_spammed_too_much@xxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Mon, 19 May 2008 14:34:33 +0200
- 

Gordon schreef:

On May 19, 12:37 pm, Erwin Moller  
<[Since\\_humans\\_read\\_this\\_I\\_am\\_spammed\\_too\\_m...@xxxxxxxxxxxxxxxxxxxxx](mailto:Since_humans_read_this_I_am_spammed_too_m...@xxxxxxxxxxxxxxxxxxxxx)> wrote:

Gordon schreef:

I want to add a feature to a project I'm working on where i have multiple users set up on my Postgres database with varying levels of access. At the bare minimum there will be a login user who only has read access to the users table so that users can log in. Once a user has been logged in successfully I want to escalate that user's access level to one appropriate to their role, which will include switching the postgres user they are logged in as to one that can make modifications to the database as well (editors get update permission, supereditors get insert/delete permission for articles, admin get insert/delete access on the user database etc). The problem is the only way I can find of doing this is to close the open PDO and create a new one, in other words disconnect from the database and reconnect. As database connections are expensive to initialize I really want to avoid this and do the postgres of an su instead.  
Back when I was doing this the old fashioned way (php 4,

Re: PDO: Switch database user without reopening connection

MySQL, MySQL extension, no OOP) I could use `mysql_change_user ()` to switch DB users once a logging in user's credentials had been validated. PDO is a great new addition to PHP and has so many excellent new features that there's really little excuse not to use it, but one thing it apparently lacks is a PDO equivalent to the old `mysql_change_user` command. I'm pretty sure that user switching is supported in Postgres, but with no `change_user` function how do I go about doing it?

Hi Gordon,

Unless you are using persistent connections, there is not much use in changing. I stopped using persistent connections (PHP4.3 on Postgresql8.2) because I got into strange troubles that I didn't understand. Reading around in here confirmed my suspicions, a few more regulars in here don't like the persistent connections either. (ask Jerry) So I quit using them.

Each script I write makes a fresh connection. (I believe PHP does some pooling behind the scenes anyway, but that never gave me troubles.) I actually never had performance problems using a fresh connection on Postgresql.

In your case I would simply store the Postgresql-username you assign to a certain visitor in the Session, and use that value to start the right (and fresh) connection each invocation of your scripts.

just my 2 cent.

Regards,  
Erwin Moller

Thanks for the quick reply but I don't think I fully made myself clear. I'm not using persistent connections or any kind of witchcraft like that. :) I meant during the lifetime of a given script invocation I want to be able to switch users. Some quick pseudocode is given below: The first one is the approach where the user gets a new PDO

```
if ($_SESSION ['user'])  
{  
  // Connect as guest user  
  $db = new PDO ('guest user login credentials');
```

## Re: PDO: Switch database user without reopening connection

```
if (user successfully verified against database)
{
unset ($db);
$db = new PDO ('authenticated user login credentials');
// Do stuff
}
}
```

I don't like this approach because 1) it involves dropping and reinstating a connection to the database and the destruction and recreation of an object, both of which have overhead which I'd rather avoid, and 2) You have all the references to the \$db object to contend with.

I think a more elegant solution would be something like:

```
if ($_SESSION ['user'])
{
// Connect as guest user
$db = new PDO ('guest user login credentials');
if (user successfully verified against database)
{
$db -> switchUser ('authenticated user login credentials');
// Do stuff
}
}
```

The main reason for wanting to do this is of course security. I want a guest role and an authenticated role, where the guest role is limited to reading from a handful of approved table. I don't want everybody who runs my scripts to do so with a database user that has the Power of Greyskull for obvious reasons. :)

Hi Gordon,

OK, more clear now.

I don't work your way, hence the misinterpretation. ;-)

Isn't it possible to simply use the SESSION to hold the login as I suggested?

In my humble opinion, your first route to the database (as guestuser to check the login credentials) shouldn't be needed anyway.

I think it is easier to set the result of that query (succes or not) in the session AFTER the first login (or second, or whatever).

Why authenticate a user multiple times?

'my way' would be:

1) user logs in (login.php), POSTS username/password to login\_process.php

2) login\_process.php checks the passed username/password.

As a result it:

a) rejects the login (unknown username/password)

or

## Re: PDO: Switch database user without reopening connection

b) Sets the result (level) in the Session, eg:

```
$_SESSION["authenticated"] = "Y";  
$_SESSION["userid"] = row["userid"];  
$_SESSION["DB_user_connect"] = row["dblevel"];
```

The last 2 rows contain fantasynames of course, but the idea is that you store in the latter a username to connect to to the database.

eg:

```
$db = "";  
if ($_SESSION["DB_user_connect"] == "guest"){  
    $db = new PDO ('guest user login credentials');  
} else {  
    if ($_SESSION["DB_user_connect"] == "level1"){  
        $db = new PDO ('level1 user login credentials');  
    }  
}
```

etc.

(Use a switch statement if you have many instead of if-then-else mess)

In that way you always build 1 connection.

The above is only to show how I approach this, maybe it is not possible in your setup.

Regards,  
Erwin Moller

.