# Accessing SQLite Databases in PHP using PDO

**This is an article on *Accessing SQLite Databases in PHP using PDO* in *PHP*.**

PDO or PHP Data Objects is a data abstraction layer, i.e. it provides uniform methods to access different types of databases, as a result switching between or moving to a different database system is relatively easier. So, in simple language the code to access a SQLite db will also work for MySQL db with minor changes. PDO provides a plethora of database drivers like MySQL, Oracle, Postgre SQL, SQLite, Firebird, etc. In this article we will look at accessing a SQLite db using PDO.

## Installing PDO

PDO & driver for SQLite is enabled by default from PHP version 5.1.0 onwards, if not so use the following command to install PDO:

Code:

```
$ yum install php-pdo
```

Windows users please refer to the official PDO installation documentation athttp://www.php.net/manual/en/pdo.installation.php.

## Access SQLite DB

Let's get our hands dirty with a little code, first let's try to list all the different database drivers available to PDO, see the code snippet below:

Code: PHP

```
<?
print_r(PDO::getAvailableDrivers());
?>
```

PDO needs a DSN (Data Source Name) to connect to, for SQLite the DSN is just the path to the database file. Follow the code snippet below:

Code: PHP

```php
<?php

// DSN
$dsn = 'sqlite:/home/pradeep/test.db';
// Connect to SQLite database
$sqlite_db = new PDO($dsn);

// Create table
$sqlite_db->exec("CREATE TABLE IF NOT EXISTS books (
                    id INTEGER PRIMARY KEY,
                    title TEXT,
                    author TEXT,
                    entry_time INTEGER)");
?>
```

Now, we have connected to the database and created a table, next let's look at a few more basic things required to work with databases, like preparing queries, using placeholders, etc.

Prepared queries help optimizing code when the same query is executed multiple times with changes in the values, similarly placeholder help you prevent SQL injection attacks on your code by automatically add quotes to values. Let's look at the code snippet below:

Code: PHP

```php
<?
// normal prepared query & placeholder
$prepare1 = $sqlite_db->prepare('INSERT INTO books(id, title, author,
entry_time) VALUES(?,?,?,?)');

// execute
$prepare1->execute(1,'Oath Of The Vayuputras','Amish','2013-04-22 00:00:00');

// named placeholder prepared query
$prepare2 = $sqlite_db->prepare('INSERT INTO books(id, title, author,
entry_time) VALUES(:id,:title,:author,:time)');

// execute
$prepare2->execute(array(':id' => 2, ':title' => 'Twenties Girl', ':author'
=> 'Sophie Kinsella', ':time' => '2013-04-25 00:00:00'));
```

```
// fetching data
$sth = $dbh->prepare( 'SELECT * FROM books', array( PDO::ATTR_CURSOR =>
PDO::CURSOR_FWDONLY ) );
$sth->execute();
$books = $sth->fetchAll();
?>
```

That's all you need to get started.

**Source: http://www.go4expert.com/articles/accessing-sqlite-databases-php-using-pdo-t29644/**