SQLite бази от данни. Функции на PHP за операции с SQLite бази от данни

SQLite е библиотека (разширение), която се добави към PHP5 вградена функционалност за БД. Представлява малка библиотека на C, разработена през 2000г от Richard Hipp.

Създаване на SQLite БД

Създаване на SQLite БД, на практика е създаване на специално форматиран файл. За създаване на БД се изпълнява функцията за отваряне на БД. Ако БД не съществува, то тя се създава със sqlite_open.

Синтаксис:
resource sqlite_open (string filename[,int mode[,string &error_mess]]);
filename е спецификация на името на базата
mode – указва режима на отваряне, права за достъп (0666 – за четене и запис);
&error_mess - променлива, в която се записва инфото за грешката, ако възникне такава.
Пример:

```
$dbhandle = sqlite_open('my_test_db', 0666, $sqliteerror);
if (!$dbhandle) die ($sqliteerror);
```

Създава се файл за SQLite БД име 'my_test_db' в същата директория, където е php приложението с права за четене/запис.
SQLite предоставя и ОО Конструктор, ч/з който се създава обект за базата данни и едновременно с това я и отваря.

```
$db = new SQLite Database ( string filename[ ,int mode[ ,string &error_mess]]);
$db = new SQLite Database (“mydb”, 0666, $err_mess);
```

Пример 1. Създаване на SQLite БД име 'my_test_db', създаване на таблица Friends в нея (CREATE TABLE ...).
Ето го целия пример:

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $sqliteerror);
if (!$dbhandle) die ($sqliteerror);
$stm = "CREATE TABLE Friends(Id integer PRIMARY KEY," .
       "Name text UNIQUE NOT NULL, Sex text CHECK(Sex IN ('M', 'F')))";
$ok = sqlite_exec($dbhandle, $stm, $sqliteerror);

if (!$ok)
   die("Cannot execute query. $error");
echo "Database Friends created successfully";
sqlite_close($dbhandle);
?>
```

//Database Friends created successfully
Забележка: За изпълнение на SQL команди, които не връщат извадка, SQLite предоставя функция sqlite_exec, която ние мнократно използвахме.

Пример 2. Запълване с данни на таблица Friends (INSERT INTO Friends...):

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $sqliteerror);
if (!$dbhandle) die ($sqliteerror);
$stm1 = "INSERT INTO Friends VALUES(1,'Jane', 'F')";
$stm2 = "INSERT INTO Friends VALUES(2,'Thomas', 'M')";
$stm3 = "INSERT INTO Friends VALUES(3,'Franklin', 'M')";

$ok1 = sqlite_exec($dbhandle, $stm1);
if (!$ok1) die("Cannot execute statement.");

$ok2 = sqlite_exec($dbhandle, $stm2);
if (!$ok2) die("Cannot execute statement.");

$ok3 = sqlite_exec($dbhandle, $stm3);
if (!$ok3) die("Cannot execute statement.");

echo "Data inserted successfully";

sqlite_close($dbhandle);
//Data inserted successfully
?>
```

Пример 3: Получаване на извадка от БД: подава се команда SELECT като параметър на функция sqlite_query() или sqlite_unbuffered_query(). Използва се функция sqlite_fetch_array - за извличане на данни от прочетен запис в масив (SQLITE_ ASSOC, SQLITE_NUM, SQLITE_BOTH).

resource sqlite_query(db_handle,string query[,result_type][&string error_msg]])

Където:

db_handle - манипулатор на БД, който получаваме от sqlite_open

query – SELECT заявка

int result_type – указва как ще бъде индексиран масивът, който ще получи като резултат от заявката. Има три Възможни стойности:

SQLITE_NUM – указва че ще се използва целочислени изрази за индекси на извлечените стойности (от 0 нататък).

SQLITE_ASSOC - елементите на масива имат като ключова стойност имената на полетата

SQLITE_BOTH – указва, че за всяко поле ще имаме по 2 елемента – един с ключ и един с целочислен израз.

error_msg - съобщение за грешка.

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $sqliteerror);
if (!$dbhandle) die ($sqliteerror);
$query = "SELECT Name, Sex FROM Friends";
$result = sqlite_query($dbhandle, $query);
if (!$result) die("Cannot execute query.");
while ($row = sqlite_fetch_array($result, SQLITE_ASSOC)) {
    echo $row['Name']  . " : " . $row['Sex'];
    echo "<br>";
}
sqlite_close($dbhandle);
?>
```

Jane : F
Thomas : M
Franklin : M

Пример 4: Още един пример: За извличане на данни:

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $sqliteerror);
if (!$dbhandle) die ($sqliteerror);
$query = "SELECT Name, Sex FROM Friends";
$result = sqlite_query($dbhandle, $query);
if (!$result) die("Cannot execute query.");
$row = sqlite_fetch_array($result, SQLITE_ASSOC);
print_r($row);
echo "<br>";
sqlite_rewind($result);
$row = sqlite_fetch_array($result, SQLITE_NUM);
print_r($row);
echo "<br>";
sqlite_rewind($result);
$row = sqlite_fetch_array($result, SQLITE_BOTH);
print_r($row);
echo "<br>";
sqlite_close($dbhandle);
?>
```

Резултат:
Array ([Name] => Jane [Sex] => F)
Array ([0] => Jane [1] => F)
Array ([0] => Jane [Name] => Jane [1] => F [Sex] => F)

Пример 5: Извличане на данни в таблица:

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $sqliteerror);
if (!$dbhandle) die ($sqliteerror);
$query = "SELECT Name, Sex FROM Friends";
$result = sqlite_query($dbhandle, $query);
if (!$result) die("Cannot execute query.");
$rows = sqlite_num_rows($result);
$field1 = sqlite_field_name($result, 0);
$field2 = sqlite_field_name($result, 1);
echo "<table style='font-size:12;font-family:verdana'>";
echo "<thead><tr>";
echo "<th align='left'>$field1</th>";
echo "<th align='left'>$field2</th>";
echo "</tr></thead>";
for ($i = 0; $i < $rows; $i++) {
    $row = sqlite_fetch_array($result, SQLITE_NUM);
    echo "<tr>";
    echo "<td>$row[0]</td>";
    echo "<td>$row[1]</td>";
    echo "</tr>";
}
echo "</table>";
sqlite_close($dbhandle);
?>
```

Name	Sex
Jane	F
Thomas	M
Franklin	M

Получаване на броя на извлечените записи и броя на извлечените колони. Пример 6: Команда sqlite_num_rows() връща броя на редовете в нашата извадка (result set) .Команда sqlite_num_fields() връща броя на КОЛОНИТЕ в нашата извадка.

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $error);
if (!$dbhandle) die ($error);

$query = "SELECT * FROM Friends LIMIT 2";
$result = sqlite_query($dbhandle, $query);
if (!$result) die("Cannot execute query.");


$rows = sqlite_num_rows($result);
$cols = sqlite_num_fields($result);

echo "The result set has $rows rows and
      $cols columns";

sqlite_close($dbhandle);

?>
```

Резултат:

```
The result set has 2 rows and 3 columns
```

Пример 7: Промяна на данни, например Jane да стане Nora (UPDATE Friends SET name ...):

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $error);
if (!$dbhandle) die ($error);
$stm1 = "UPDATE Friends SET name  = 'Nora' Where name='Jane'";
$ok1 = sqlite_exec($dbhandle, $stm1);
sqlite_close($dbhandle);
?>
```

Пример 8: Изтриване на данни, например да изтрием запис по зададено име (DELETE FROM Friends WHERE name= ...):

```
Пример: Изтриваме Nora
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $error);
if (!$dbhandle) die ($error);
$stm1 = "Delete FROM Friends Where name='Nora'";
$ok1 = sqlite_exec($dbhandle, $stm1);
```

```
sqlite_close($dbhandle);
?>
```

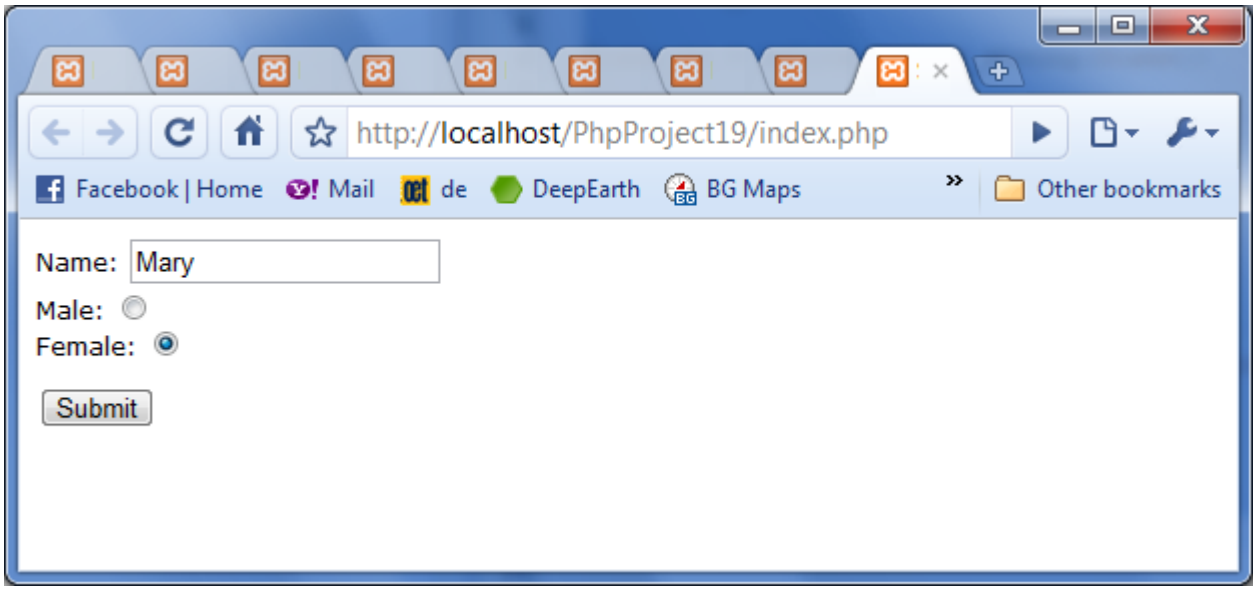
Пример 9 (Самостоятелно). Да се създаде форма за въвеждане на данни

```
<html>
<head>
<title>SQLite PHP form</title>
</head>
<body style="font-size:12;font-family:verdana">
<form action="add.php" method="post">
<p>
Name: <input type="text" name="name"><br>
Male: <input type="radio" value="M" name="gender"><br>
Female: <input type="radio" value="F" name="gender">
</p>
<p>
<input type="submit">
</p>
</form>
</body>
</html>
```

Add.php

```
<?php
$gender = $_POST['gender'];

$name = $_POST['name'];
$name_es = sqlite_escape_string($name);
if (!empty($name)) {
   $dbhandle = sqlite_open('my_test_db', 0666, $error);
   if (!$dbhandle) die ($error);
   $stm = "INSERT INTO Friends(Name, Sex) VALUES('$name_es', '$gender')";
   $ok = sqlite_exec($dbhandle, $stm, $error);
   if (!$ok) die("Error: $error");
   echo "Form submitted successfully";
}
```




```
?>
```

Form submitted successfully

Използваме sqlite_escape_string(). Чрез тази функция се избягва необходимостта от употреба на (\) пред апостроф, например ако името е да кажем O'Hara.

Пример 10. Този пример извежда всички таблици в базата от данни, в случая - my_test_db.

```
<?php
$dbhandle = sqlite_open('my_test_db', 0666, $error);
if (!$dbhandle) die ($error);
$query = "SELECT name, sql FROM sqlite_master WHERE type='table'";
$result = sqlite_query($dbhandle, $query, SQLITE_NUM);
if (!$result) die("Cannot execute query.");
while (sqlite_has_more($result)) {
    $row = sqlite_fetch_array($result);
    echo "table: $row[0], sql: $row[1]";
    echo "<br>";
}

sqlite_close($dbhandle);

?>
```

//Result:
table: Friends, sql: CREATE TABLE Friends(Id integer PRIMARY KEY,Name text UNIQUE NOT NULL, Sex text CHECK(Sex IN ('M', 'F')))

Използваме таблица **sqlite_master** за да получим списък всички таблици в базата от данни.

```
$query = "SELECT name, sql FROM sqlite_master WHERE type='table'";
```

Атрибут (колона) name на таблица **sqlite_master** съдържа имената на таблиците в базата от данни. Колоба sql на таблица sqlite_master дава SQL командата използвана за създаване на съответната таблица.

```
while (sqlite_has_more($result)) {
    $row = sqlite_fetch_array($result);
    echo "table: $row[0], sql: $row[1]";
    echo "<br>";
}
```

Използвана е функция **sqlite_has_more()**, която връща TRUE ако има още редове в извадката ($result), иначе FALSE.