SQLite бази от данни

Виолета Божикова

Какво е SQLite?

- SQLite е разширение (малка библиотека на C), което добави към PHP5 вградена функционалност на SQL машина за БД.
- Разширението е разработено през 2000г от Richard Hipp.
- От PHP 5.3.0 нататък е в сила SQLite3.

Предимства и недостатъци

Предимства:

- Самостоятелност: SQLite не използва модела клиент-сървър; вгражда се в PHP приложението и изисква само права за запис и четене за файловете от БД.
- Не е необходимо администриране, БД се създава с изпълнение на функцията от библиотеката.
- Високо бързодействие. За повечето заявки SQLite има бързодествие съизмеримо с това на MySQL.
- Наличие едновременно на процедурен и ОО интерфейс.
- Разширението е безплатно дори и при използването в комерсиални продукти.

Предимства и недостатъци

Недостатъци:

- Липсва сървърен процес. Кодът на SQLite се изпълнява в същия процес както интернет приложението. По правило сървърите за БД работят в отделен поцес и това дава възможност за изпълнение на асинхронни заявки.
- Липсва вградена възможност за обработка на двоични данни. При запис двоичните данни се прекодират в символен низ, а при четене се преобразуват отново в двоични.
- Транзакциите заключват цялата БД, което прави едновременния достъп на няколко потребителя много бавен.

Най-подходяща област за:

- Интернет приложения, които изпълняват много заявки за четене от БД, но по-рядко се прави запис, следователно - рядко се прави заключване на БД. В такива случаи SQLite операциите се изпълняват с висока скорост.

Основни функции на PHP за работа с SQLite3 за ОС Windows

- Следните функции са достатъчни за работа на PHP с SQLite3:

1. **$db_handle = new SQLite3($filename);**
2. **$db_handle->exec($query_string);**
3. **$result = $db_handle->query($query_string);**
4. **$row= $result->fetchArray();**

Функция за създаване на SQLite3 БД

- Създаване на SQLite3 БД, на практика представлява създаване на специално форматиран файл.

- За създаване на БД, SQLite3 предоставя ОО конструктор (функция **SQLite3()**), ч/з който се създава обект на базата данни и едновременно с това БД се отваря:

$db_handle = new SQLite3($filename);

- Параметърът **$filename** е string, който указва името на вашия базов файл (файлова спецификация), а **$db_handle** е обектът от типа на БД, който се създава и FALSE - при неуспех.

Създаване на таблица в SQLite БД

- **Таблица се създава с командата:**

CREATE TABLE tb_name (column_def[, column_def]…)

Където:

- **tb_name – име на таблица,**
- **column_def - име на поле [тип][ограничения].**

Ограничения: NOT NULL, PRIMARY KEY, UNIQUE, DEFAULT value, CHECK(exp) и др.

Типове данни в SQLite БД

- В I-ва версия на SQLITE има 2 типа данни:
  - INTEGER;
  - VARCHAR - символен низ с динамична дължина.
- Във версия 2 и 3 на SQLITE са налични следните типове данни:
  - INTEGER, VARCHAR, CHAR(n) - (низ с фиксирана дължина),
  - REAL, TEXT (низ с произволна дължина), BLOB – двоични с голям обем, CLOB - текстови данни, с голям обем.

Пример 1.

И така: Нека създадем SQLite3 БД, с име my_test_db:

```
$dbhandle  = new SQLite3('my_test_db') or die('Unable
to open database');
if (!$dbhandle) die ($sqliteerror);
$stm = "CREATE TABLE Friends(Id integer PRIMARY KEY,"
      "Name text UNIQUE NOT NULL, Sex text CHECK(Sex
IN ('M', 'F')))";
```

- **В примера имаме команда за създаване на таблица Friends (с SQL командата CREATE TABLE).**
- **Поле, Id е първичен ключ (PRIMARY KEY), Name е с уникална, различна от NULL стойност, а поле Sex е текстово, като с клаузата CHECK(exp) - се валидират данните за него.**

Целия пример 1. Създаване на SQLite БД име 'my_test_db и таблица Friends в нея.

```
<?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to
open database');
                //или if (!$dbhandle) die ($sqliteerror);
$stm = "CREATE TABLE Friends(Id integer PRIMARY KEY," .
      "Name text UNIQUE NOT NULL, Sex text CHECK(Sex IN
('M', 'F')))";
$ok = $dbhandle->exec($stm);
if (!$ok) die("Cannot execute query. $error");
echo "Database Friends created successfully";
$dbhandle->close();
?>              //Database Friends created successfully
```

Запълване с данни на таблица в SQLite3 БД

```
... //Запълване с данни на таблица Friends - INSERT INTO …
$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 = $dbhandle->exec($stm1);
if (!$ok1) die("Cannot execute statement.");

$ok2 = $dbhandle->exec($stm2);
if (!$ok2) die("Cannot execute statement.");

$ok3 = $dbhandle->exec($stm3);
if (!$ok3) die("Cannot execute statement.");
```

Функции exec() и query() в SQLite3 БД

- И двете функции - exec() и query(), изпращат string съдържащ SQL оператор към базата от данни за изпълнение.

$db_handle->exec($query_string);

$result = $db_handle->query($query_string);

- Разликата е, че **exec()** не връща извадка (резултат от БД) и по този начин е по-подходящ за SQL оператори като CREATE, INSERT, UPDATE, DELETE или DROP.
- А query() се използва за SELECT оператори, които връщат резултат ($result) от базата от данни. Върнатият резултат е PHP object, който вие ще използвате за достъп до данните, върнати от заявката.

Целия 2. Запълване на таблица Friends с данни.

```
<?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to open database');
$sql =<<<tralala
INSERT INTO Friends VALUES(7,'Jana', 'F');
INSERT INTO Friends VALUES(8,'Thom', 'M');
INSERT INTO Friends VALUES(9,'Franko', 'M');
tralala;
$ok =$dbhandle->exec($sql);
if (!$ok) die ("Cannot execute statement.");
else echo "Data inserted successfully";
$dbhandle->close();
//Data inserted successfully
?>
```

Връщане на извадка от БД – функция query()

- Получаване на извадка от БД: подава се команда **SELECT** като параметър на функция **query()**.
- Използва се функция **fetchArray()** - за извличане на един ред (**$row**) от извадката (**$result**). Върнатият ред $row е масив, всеки елемент на който е колона на базата.
- Ключът на всеки елемент от масива може да е името на колоната на базата, може да е цяло число от 0 нататък или да се състои от 2 елемента: името на колоната и цяло число, започващо от 0, т.е SQLITE3_ASSOC, SQLITE3_NUM, SQLITE3_BOTH.

Пример 3: Получаване на извадка от БД. Функции query() и fetchArray(SQLITE3_ASSOC)

Пример 3: команда SELECT се подава като параметър на функция query(), а функция **fetchArray(SQLITE3_ASSOC)** се използва за извличане на един ред от извадката.

```
<?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to open database');
$query = "SELECT Name, Sex FROM Friends";
$result = $dbhandle->query($query);
if (!$result) die("Cannot execute query.");
while ($row = $result->fetchArray(SQLITE3_ASSOC))
{ echo $row['Name']  . " : " . $row['Sex'];    echo "<br>";}
$dbhandle->close();
?>
```

Резултат:

Jane : F
Thomas : M
Franklin : M

Пример 4: Получаване на извадка от БД. Функции query() и fetchArray(SQLITE3_BOTH)

Пример 3: команда SELECT се подава като параметър на функция query(), а функция fetchArray(SQLITE3_BOTH) се използва за извличане на един ред от извадката.

```
<?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to open database');
if (!$dbhandle) die ($sqliteerror);
$query = "SELECT Name, Sex FROM Friends";
$result = $dbhandle->query($query);
if (!$result) die("Cannot execute query.");
while ($row = $result->fetchArray(SQLITE3_BOTH)) {
   echo $row[0]  . " : " . $row[1];
   echo "<br>";
}
$dbhandle->close();
?>
```

Резултат:

Jane : F
Thomas : M
Franklin : M

Пример 5: Получаване на извадка от БД и таблично представяне. Функции query() и fetchArray(SQLITE3_NUM)

Пример 3: команда SELECT се подава като параметър на функция query(), а функция fetchArray(SQLITE3_NUM) се използва за извличане на един ред от извадката.

```
<?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to open database');
if (!$dbhandle) die ($sqliteerror);
$query = "SELECT Name, Sex FROM Friends";
$result = $dbhandle->query($query);
if (!$result) die("Cannot execute query.");
echo "<table BORDER='3'>";
while ($row = $result->fetchArray(SQLITE3_NUM))
{ echo "<tr>"; echo "<td>$row[0]</td>"; echo "<td>$row[1]</td>";  echo "</tr>";}
echo "</table>";
$dbhandle->close();
?>
```

Резултат:

Jane	F
Thomas	M
Franklin	M
Jana	F
Thom	M
Franko	M

Пример 6. Заявки за промяна и изтриване в SQLite БД: UPDATE и DELETE

```
 <?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to
open database');
if (!$dbhandle) die ($sqliteerror);
$stm = "UPDATE Friends SET name  = 'Nina' Where
name='Jana'"; // Jane да стане Nina
$ok = $dbhandle->exec($stm);
if (!$ok) die ("Cannot execute UPDATE Friends.");
else echo "Data UPDATED successfully";
$dbhandle->close();
//Data UPDATED successfully
?>
```

Пример 7. Заявки за промяна и изтриване в SQLite БД: UPDATE и DELETE

```
<?php
$dbhandle  = new SQLite3('my_test_db') or die('Unable to
open database');
if (!$dbhandle) die ($sqliteerror);
$stm = "Delete FROM Friends Where name='Nora'";
// запис с име Nora ще бъде изтрит
$ok = $dbhandle->exec($stm);
if (!$ok) die ("Cannot execute UPDATE Friends.");
else echo "Data Deleted successfully";
$dbhandle->close();
//Data Deleted successfully
?>
```

Литература:

- http://php.net/manual/en/sqlite3.version.php
- http://babbage.cs.qc.cuny.edu/courses/cs903/2013_02/using_sqlite3.html
- http://www.tutorialspoint.com/sqlite/sqlite_php.htm