MySQL - це система управління реляційними базами даних. У реляційній базі даних дані зберігаються в окремих таблицях, завдяки чому досягається виграш у швидкості й гнучкості. Таблиці зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість поєднувати при виконанні запиту дані з декількох таблиць. SQL як частина системи MySQL можна охарактеризувати як мова структурованих запитів, що використовується для доступу до баз даних.

MySQL - це ПЗ з відкритим кодом. Застосовувати його і модифікувати може будь-хто. Таке ПЗ можна отримувати за допомогою Internet і використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його у відповідності зі своїми потребами.

MySQL складається з двох частин: серверної і клієнтської.

Сервер MySQL постійно працює на комп'ютері. Клієнтські програми (наприклад, скрипти PHP) посилають серверу MySQL SQL-запити через механізм сокетів (тобто за допомогою мережевих засобів), сервер їх обробляє і запам'ятовує результат. Тобто скрипт (клієнт) вказує, яку інформацію він хоче отримати від сервера баз даних. Потім сервер баз даних посилає відповідь (результат) клієнтові (скрипту).

Структура MySQL трирівнева: бази даних - таблиці - записи. Бази даних і таблиці MySQL фізично представляються файлами з розширеннями frm, MYD, MYI. Логічно таблиця являє собою сукупність записів. А запису - це сукупність полів різного типу. Ім'я бази даних MySQL унікально в межах системи, а таблиці - в межах бази даних, поля - в межах таблиці. Один сервер MySQL може підтримувати одразу декілька баз даних, доступ до яких може розмежовуватись логіном і паролем.

Поля та їх типи в MySQL

База даних з точки зору MySQL - це звичайний каталог, що містить файли певного формату - таблиці. Таблиці складаються із записів, а записи, у свою чергу, складаються з полів. Поле має два атрибути - ім'я і тип.

Тип поля може бути:

- ціле число
- дійсним
- рядок
- бінарний
- Дата і час
- Перерахування і множини

Функції PHP для роботи з MySQL

Розглянемо основні функції PHP, призначені для роботи з MySQL сервером.

З'єднання з сервером MySQL

Основною функцією для з'єднання з сервером MySQL є mysql_connect (), яка підключає скрипт до сервера баз даних MySQL та виконується авторизацію користувача базою даних. Синтаксис у даної функції такий:

```
mysql_connect ([string $hostname] [, string $user] [, sting $password]);
```

Всі параметри даної функції є необов'язковими, оскільки значення за замовчуванням можна прописати у файлі конфігурації php.ini. Якщо ви хочете вказати інші імена MySQL-хоста, користувача і пароль, ви завжди можете це зробити. Параметр $hostname може бути вказаний у вигляді: хост: порт. Функція повертає ідентифікатор (типу int) з'єднання, вся дальнейшая робота здійснюється тільки через цей ідентифікатор. При наступному виконанні функції mysql_connect () з тими ж параметрами нове з'єднання не буде відкрито, а функція поверне ідентифікатор існуючого з'єднання.

Для закриття з'єднання призначена функція mysql_close (int $ connection_id). Взагалі, підключення можна і не закривати - воно буде закрито автоматично по завершенні роботи PHP скрипта. Однак, це поганий стиль. Якщо кількість з'єднань більше одного, вказується ідентифікатор $ connection_id того з'єднання, яке необхідно закрити.

Функція mysql_connect () встановлює звичайне з'єднання з MySQL. Однак, PHP підтримує постійні з'єднання - для цього використовують функцію mysql_pconnect (). Аргументи цієї функції такі ж, як і у mysql_connect (). Різниця між постійним і простим з'єднанням в тому, що постійне з'єднання не закривається після завершення роботи скрипта, навіть якщо скрипт викликав функцію mysql_close ().

Функція вибору бази даних

Функція mysql_select_db (string $ db [, int $ id]) обирає базу даних, з якою буде працювати PHP скрипт. Якщо відкрито не більше одного з'єднання, можна не вказувати параметр $ id.

Наприклад: Спроба встановити з'єднання з MySQL:

```
if (!mysql_connect($server, $user, $ password)) {
echo "Ошибка подключения к серверу MySQL";
exit;
}
// Если соединились, выбираем базу данных:
mysql_select_db($db);
```

Вибір кодування символів для з'єднання

Кодування символьних полів у базі MySQL даних повинна збігатися з кодуванням з'єднання. Тому для надійності слід відразу після підключення до MySQL виконати запит

```
mysql_query("SET NAMES cp1251");
```

або

```
mysql_query("SET CHARACTER SET utf8");
```

Для кирилиці придатними кодуваннями є utf8, koi8, cp1251, cp886.

Використовуйте кодування, яка підходить для мови даних, щоб забезпечити правильний пошук і сортування рядків.

детальніше - на http://www.linux.by/wiki/index.php/FAQ_PHP_MySQL_charset
Документація на англійською мовою

Функції обробки помилок

Якщо відбудеться помилка з'єднання з MySQL, то ви отримаєте відповідне повідомлення і скрипт завершить свою роботу. Це не завжди буває зручно, перш за все, при налагодженні скриптів. Тому, в PHP є наступні дві функції:

```
mysql_errno(int $id);           mysql_error(int $id);
```

Перша функція повертає номер помилки, а друга - повідомлення про помилку. У результаті ми можемо використовувати наступне:

```
echo "ERROR ".mysql_errno()." ".mysql_error()."\n";
```

Тепер буде ясно, через що сталася помилка - ми побачимо відповідним чином оформлене повідомлення.

Функція виконання запитів до сервера баз даних

Всі запити до поточної базі даних відправляються функцією mysql_query (). Цієї функції потрібно передати один параметр - текст запиту. Текст запиту модет містити пробільні символи і символи нового рядка (\ n). Текст повинен бути складений за правилами синтаксису SQL.

Приклади запитів:

Створення власної таблиці:

```
mysql_query("CREATE TABLE my_table(word VARCHAR(50), qid INT))"
```

Наступний запит повертає запис з таблиці mytable.

```
$q = mysql_query("SELECT * FROM my_table");
```

Результат запиту присвоюється змінної $ q. Результат - це набір даних, який після виконання запиту потрібно обробити певним чином.

Також використовують mysql_query () з SQL-запитом створення бази даних SQL CREATE DATABASE. Це краще, ніж використання функції створення бази даних mysql_create_db (), яка взагалі не рекомендується до використання і недоступна в бібліотеці для MySQL версій 4.x.

Функції обробки результатів запиту

Якщо запит, виконаний за допомогою функції mysql_query () успішно виконався, то в результаті клієнт отримає набір записів, який може бути оброблений функціями PHP. Розглянемо деякі з них.

Функція mysql_num_rows () дозволяє дізнатися, скільки записів містить результат запиту:

```
$q = mysql_query("SELECT * FROM mytable");
echo "В таблице mytable ".mysql_num_rows($q)." записей";
```

3

апис складається з полів (колонок). За допомогою функції mysql_num_fields () можна дізнатися, скільки полів містить кожна запис результату:

```
$q = mysql_query("SELECT * FROM mytable");
echo "В таблице mytable ".mysql_num_fields($q)." полей ";
```

Також є можливість дізнатися значення однієї клітинки результату запиту. Це можна зробити функцією mysql_result (resourse $ result, int $ row [, mixed $ field]).

Параметр функції $ row задає номер запису, а параметр $ field - ім'я або порядковий номер поля. Аргументом поля може бути зсув, ім'я поля, або ім'я поля й ім'я таблиці через крапку (tablename.fieldname). Якщо до імені колонки, в запиті, був використаний аліас ('select foo as bar from ...'), використовуйте його замість реального імені колонки.

Працюючи з великими результатами запитів, слід використовувати одну з функцій, які обробляють відразу цілий рядок результату. Так як ці функції повертають значення кількох осередків відразу, вони НАБАГАТО швидше mysql_result (). Крім того, вказівка чисельного зміщення працює набагато швидше, ніж вказівку колонки, або колонки і таблиці через крапку. Виклики функції mysql_result () не повинні змішуватися з іншими функціями, які працюють з результатом запиту.

```
<?php
$link = mysql_connect("localhost", "mysql_user", "mysql_password")
or die("Could not connect: " . mysql_error());

$result = mysql_query("SELECT name FROM work.employee")
or die("Could not query: . mysql_error());

echo mysql_result($result,2); // outputs third employee's name

mysql_close($link);
?>
```

Для обробки великих наборів записів рекомендується використовувати функції mysql_fecth_row (), mysql_fecth_array (), і.т.д.

Функція mysql_fecth_row (int $ res) повертає масив, що містить дані обробленого рядка, або FALSE, якщо рядів більше немає. Вона обробляє один ряд результату, на який посилається переданий покажчик. Ряд повертається в масиві. Кожна колонка розташований в наступній комірці масиву. Масив починається з індексу 0. Наступні виклики функції mysql_fetch_row () повернуть наступні ряди або FALSE, якщо рядів не залишилося.

Зауваження: Імена полів, що повертаються цією функцією, чутливі до регістру. Приклад:

```
$q = mysql_query("SELECT * FROM mytable WHERE month=\"$db_m\" AND day=\"$db_d\");
for ($c=0; $c<mysql_num_rows($q); $c++)
{
$f = mysql_fetch_row($q);
echo $f;
}
```

Функція mysql_fecth_array (int $ res [, int $ result_type]) повертає асоціативний масив, а масив, заданий необов'язковим параметром $ result_type, який може приймати наступні значення:

- MYSQL_ASSOC - повертає асоціативний масив;
- MYSQL_NUM - повертає масив з числовими індексами, як у функції mysql_fecth_row ();
- MYSQL_BOTH - повертає масив з подвійними індексами, тобто ви можете працювати з ним, як з асоціативним масивом і як зі списком (MYSQL_BOTH - це значення за замовчуванням для параметра $ result_type.

У PHP є функція, яка повертає асоціативний масив з одним індексом mysql_fetch_assoc (int $ res). Фактично, ця функція є синонімом для

mysql_fetch_array ($ res, MYSQL_ASSOC);

Приклади використання функції mysql_fecth_array ():

```
$q = mysql_query("SELECT * FROM mytable WHERE month=\"$db_m\" AND day=\"$db_d\");
for ($c=0; $c<mysql_num_rows($q); $c++)
{
$f = mysql_fetch_array($q);
echo "$f[email] $f[name] $f[month] $f[day] <br>";
}
```

З допомогою циклу while можна заповнити масив результатів:

```
$q = mysql_query("SELECT * FROM mytable WHERE month=\"$db_m\" AND day=\"$db_d\");
$res = Array();
while ($f = mysql_fetch_array($q)) $res[] = $f;
mysql_free_result($q);
```

mysql_free_result (resource result) вивільнить всю пам'ять, займану результатом, на який посилається переданий функції покажчик result. Може бути необхідна у випадку, якщо запит до бази даних повертає велику кількість даних. Опції отримання інформації про результати SQL-запитів

PHP надає ще кілька корисних функцій, які дозволяють дізнатися інформацію про результами SQL-запитів. Функція mysql_field_name (int $ result, int $ offset) повертає ім'я поля, що знаходиться в результаті $ result з номером $ offset (нумерація починається з 0). Тобто, простіше, функція повертає ім'я поля з номером $ offset.

Функція mysql_field_type (int $ result, int $ offset) повертає тип поля з номером $ offset в результаті $ result (номер задається щодо результату, а не таблиці).

Функція mysql_field_flags (int $ result, int $ offset) повертає перелічені через пробіл прапори (модифікатори), які є у поля з номером $offset.

Закриття з'єднання з сервером MySql

Функція mysql_close ([resource link_identifier]) - закриває з'єднання з сервером MySQL. Повертає TRUE в разі успішного завершення або FALSE в разі виникнення помилки.

```
<?php
$link = mysql_connect("localhost", "mysql_user", "mysql_password")
or die("Could not connect: " . mysql_error());
print ("Connected successfully");
mysql_close($link);
?>
```

mysql_close () закриває з'єднання з базою даних MySQL, на яке вказує переданий покажчик. Якщо параметр link_identifier не вказано, закривається останнє відкрите (поточне) з'єднання. Використання цієї функції не обов'язково для непостійних з'єднань. Вони автоматично закриваються в кінці роботи скрипта.

Зауваження: mysql_close () не може закривати постійні з'єднання, відкриті функцією mysql_pconnect ().

SQL Injection

SQL Injection - це спосіб атаки на сайти, які взаємодіють з базою даних.