Практическое занятие № 4 – Поддержка баз данных в РНР

Цель работы:

Изучение технологии и получение практических навыков создания динамических web-страниц на основе данных, расположенных на сервере MySQL.

**Теоретические сведения**

Несомненно, одним из важных аспектов РНР является поддержка баз данных. В РНР реализована обширная поддержка практически всех существующих серверов баз данных, в том числе:

|  |  |  |
| --- | --- | --- |
| Adabas D | Informix | PostgreSQL |
| Dbase | Ingres | Solid |
| Direct MS-SQL | InterBase | Sybase |
| Empress | mSQL | UNIX dbm |
| File-Pro (read-only) | MySQL | Velods |
| FrontBase | ODBC |  |
| IBM DB2 | Oracle (OCI7 и OC18) |  |

Как показывает этот список, поддержка баз данных в РНР простирается от совместимости с базами данных, известных всем (например, Oracle), до тех, о которых многие даже не слышали. Поддержка базы данных в РНР представлена набором стандартных функций для соединения с базой, обработки запросов и разрыва связи.

Сервер MySQL дает неплохое представление об общих возможностях поддержки баз данных в РНР и, к тому же, является на данный момент наиболее распространенным в Web. В принципе, независимо от того, с каким сервером баз данных вы будете работать, адаптация примеров не вызовет особых сложностей.

MySQL (http://www.mysql.com/) — надежная СУБД на базе SQL, разработанная и сопровождаемая фирмой Т.с.Х DataKonsultAB (Стокгольм, Швеция). Начиная с 1995 года, MySQL стала одной из самых распространенных СУБД в мире, что отчасти обусловлено ее скоростью, надежностью и гибкой лицензионной политикой.

Благодаря хорошим характеристикам и обширному набору стандартных интерфейсных функций, очень простых в использовании, MySQL стала самым популярным средством для работы с базами данных в РНР.

MySQL распространяется на условиях общей лицензии GNU (GPL, GNU Public License). Полное описание текущей лицензионной политики MySQL приведено на сайте MySQL (http://www.mysql.com/).

**Работа с сервером MySQL в PHP**

Общая последовательность действий при взаимодействии с сервером MySQL выглядит так:

1. установить соединение с сервером MySQL. Если попытка завершается неудачей, вывести соответствующее сообщение и завершить процесс.
2. выбрать базу данных сервера MySQL. Если попытка выбора завершается неудачей, вывести соответствующее сообщение и завершить процесс. Допускается одновременное открытие нескольких баз данных для обработки запросов.
3. обработать запросы к выбранной базе (или базам).
4. после завершения обработки запросов закрыть соединение с сервером баз данных.

В примерах используются таблицы products, customers и orders (см. рис. 1).

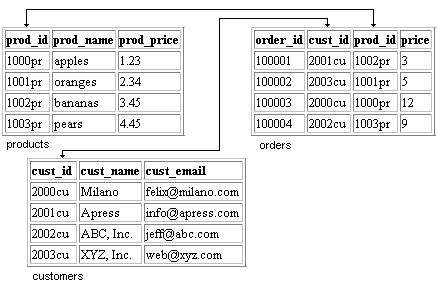


Рисунок 1

**Основные функции для работы с сервером MySQL**:

**mysql\_connect()**

Функция mysql\_connect( ) устанавливает связь с сервером MySQL После успешного подключения к MySQL можно переходить к выбору баз данных, обслуживаемых этим сервером. Синтаксис функции mysql\_connect( ):

resource mysql\_connect ([string хост [:порт] [:/путь//к/сокету] [, string имя пользователя] [, string пароль])

Возвращает: идентификатор связи MySQL при успешном выполнении, или false при ошибке.

В параметре хост передается имя хостового компьютера, указанное в таблицах привилегий сервера MySQL. Конечно, оно же используется для перенаправления запросов на web-сервер, на котором работает MySQL, поскольку к серверу MySQL можно подключаться в удаленном режиме. Наряду с именем хоста могут указываться необязательные параметры — номер порта, а также путь к сокету (для локального хоста). Параметры имя\_пользователя и пароль должны соответствовать имени пользователя и паролю, заданным в таблицах привилегий MySQL. Обратите внимание: все параметры являются необязательными, поскольку таблицы привилегий можно настроить таким образом, чтобы они допускали соединение без проверки. Если параметр хост не задан, mysql\_connect( ) пытается установить связь с локальным хостом.

Пример открытия соединения с MySQL:

@mysql\_connect(" localhost", "web", "4tf9zzzf") or

die("Could not connect to MySQL server!");

В данном примере значение, возвращаемое при вызове rnysql\_connect() не используется. Если в программе используется всего одно соединение с сервером MySQL, это вполне нормально. Но если программа устанавливает соединения с несколькими серверами MySQL на разных хостах, следует сохранить идентификатор соединения, возвращаемый при вызове mysql\_connect( ), чтобы адресовать последующие команды нужному серверу MySQL.

Пример:

<?

$link1 = @mysql\_connect("www.somehost.com", "web", "abcde")

or die("Could not connect to MySQL server!");

$linkl = @mysql\_connect("www.someotherhost.com", "usr","secret")

or die("Could not connect to MySQL server!");

?>

Идентификаторы $link1 и $link2 передаются при последующих обращениях к базам данных с запросами. Вскоре вы узнаете, как именно адресовать запрос нужному серверу при помощи идентификатора соединения.

Функция **mysql\_pconnect( )** обеспечивает поддержку восстанавливаемых (persistent) соединений. В многопользовательских средах рекомендуется использовать mysql\_pconnect( ) вместо mysql\_connect( ) для экономии системных ресурсов. По типам параметров и возвращаемого значения функция mysql\_pconnect( ) в точности совпадает c mysql\_connect( ).

**mysql\_select\_db( )**

После успешного соединения с MySQL необходимо выбрать базу данных, находящуюся на сервере. Для этого используется функция mysql\_select\_db( ). Синтаксис функции mysql\_select\_db( ):

bool mysql\_select\_db (string имя\_базы\_данных [,resource идентификатор\_соединения])

Возвращает: true при успешном выполнении, false при ошибке Параметр имя\_базы\_данных определяет выбираемую базу данных, идентификатор которой возвращается функцией mysql\_select\_db( ). Обратите внимание: параметр идентификатор\_соединения необязателен лишь при одном открытом соединении с сервером MySQL. При наличии нескольких открытых соединений этот параметр должен указываться.

Пример выбора базы данных функцией mysql\_select\_db( ):

<?

@mysql\_connect("localhost", "web". "4tf9zzzf")

or die("Could not connect to MySQL server!");

@mysql\_select\_db("company")

or die("Could not select company database!");

?>

**mysql\_close( )**

После завершения работы с сервером MySQL соединение необходимо закрыть. Функция mysql\_close( ) закрывает соединение, определяемое необязательным параметром. Если параметр не задан, функция mysql\_close( ) закрывает последнее открытое соединение.

Синтаксис функции mysql\_close( ):

bool mysql\_close ([resource идентификатор\_соединения])

Соединения, открытые функцией mysql\_pconnect( ), закрывать не обязательно.

**mysql\_query( )**

Функция mysql\_query( ) обеспечивает интерфейс для обращения с запросами к базам данных. Синтаксис функции mysql\_query( ):

resource mysql\_query (string запрос [,resource идентификатор\_соединения])

Параметр запрос содержит текст запроса на языке SQL. Запрос передается либо соединению, определяемому необязательным параметром идентификатор\_соединения, либо, при отсутствии параметра, последнему открытому соединению.

При успешном выполнении команды SQL SELECT возвращается идентификатор результата, который впоследствии передается функции mysql\_result( ) (только для SQL оператора SELECT) для последующего форматирования и отображения результатов запроса. Если обработка запроса завершилась неудачей, функция возвращает FALSE. Количество записей, участвующих в запросе, определяется при помощи функции mysql\_num\_rows( ). Эта функция также описана далее.

**mysql\_affected\_rows ( )**

Во многих ситуациях требуется узнать количество записей, участвующих в запросе SQL с командами INSERT, UPDATE, REPLACE или DELETE. Задача решается функцией mysql\_affected\_rows( ).

Синтаксис функции:

int mysql\_affected\_rows ([resource идентификатор\_соединения])

Обратите внимание: параметр идентификатор\_соединения не является обязательным. Если он не указывается, mysql\_affected\_rqws( ) пытается использовать последнее открытое соединение. Пример:

<?

// Подключиться к серверу и выбрать базу данных

@mysql\_connect("localhost", "web". "4tf9zzzf")

or die("Could not connect to MySQL server!");

@mysql\_select\_db("company")

or die("Could not select company database!");

// Создать запрос

$query = "UPDATE products SET prod\_name = \"cantaloupe\" WHERE prod\_id = \'10001pr\";

// Выполнить запрос

$result = mysql\_query($query);

// Определить количество обновленных записей

print "Total row updated; ". mysql\_affected\_rows( );

mysql\_close( );

?>

При выполнении этого фрагмента будет выведен следующий результат:

**Total row updated: 1**

Функция mysql\_affected\_rows( ) не работает с запросами, основанными на команде SELECT. Для определения количества записей, возвращенных при вызове SELECT, используется функция mysql\_num\_rows( ), описанная в следующем разделе.

В одной специфической ситуации функция mysql\_affected\_rows( ) работает с ошибкой. При выполнении команды DELETE без секции WHEREmysql\_affected\_rows( ) всегда возвращает 0.

**mysql\_num\_rows( )**

Функция mysql\_num\_rows( ) определяет количество записей, возвращаемых командой SELECT. Синтаксис функции mysql\_num\_rows( ):

int mysql\_num\_rows(resource результат)

**mysql\_result( )**

Функция mysql\_result() используется в сочетании с mysql\_query( ) (при выполнении запроса с командой SELECT) для получения набора данных. Синтаксис функции mysql\_result():

mixed mysql\_result (resource идентификатор\_результата, int запись [. mixed поле"]')

В параметре идентификатор\_результата передается значение, возвращенное функцией mysql\_query( ). Параметр запись ссылается на определенную запись набора данных, определяемого параметром идентификатор\_результата. Наконец, в необязательном параметре поле могут передаваться: смещение поля в таблице; имя поля; имя поля в формате имя\_поля\_имя\_таблицы.

Пример: выборка и форматирование данных в базе данных MySQL <?

@mysql\_connect("localhost", "web", "ffttss")

or die("Could not connect to MySQL server!");

@mysql\_select\_db("company")

or die("Could not select products database!");

// Выбрать все записи из таблицы products

$query = "SELECT \* FROM products"; $result = mysql\_query($query);

$x = 0;

print "<table>\n";

print "<tr>\n<th>Product ID</th><th>Product Name</th><th>Product Price</th>\n</tr>\n";

while ($x < mysql\_numrows($result)) :

$id = mysql\_result($result. $x. 'prod\_id');

$name = mysql\_result($result, $x, 'prod\_name');

$price = mysql\_result($result. $x, 'prod\_price'); print "<tr>\n";

print "<td>$id</td>\n<td>$name</td>\n<td>$price</td>\n";

print "</tr>\n";

$x++;

endwhile;

print "</table>";

mysql\_close();

?>

Функция mysql\_result() удобна для работы с относительно небольшими наборами данных, однако существуют и другие функции, работающие намного эффективнее, — а именно, функции mysql\_fetch\_row() и mysql\_fetch\_array().

**mysql\_fetch\_row()**

Обычно гораздо удобнее сразу присвоить значения всех полей записи элементам индексируемого массива (начиная с индекса 0), нежели многократно вызывать mysql\_result( ) для получения отдельных полей. Задача решается функцией mysql\_fetch\_row( ), имеющей следующий синтаксис:

array mysql\_fetch\_row (resource результат)

Использование функции list( ) в сочетании с mysql\_fetch\_row( ) позволяет сэкономить несколько команд, необходимых при использовании mysql\_result( ).

Пример: выборка данных функцией mysql\_fetch\_row( )

<?

@mysql\_connect( "localhost", "web", "ffttss")

or die("Could not connect to MySQL server!");

@mysql\_select\_db("company")

or die("Could not select products database!");

$query = "SELECT \* FROM products";

$result = mysql\_query($query);

print "<table>\n";

print "<tr>\n<th>Product ID</th><th>Product Name</th><th> Product Price</th>\n</tr>\n";

while ($row = mysql\_fetch\_array($result)) :

print "<tr>\n":

print "<td>".$row["prod\_id"]."</td>\n<td>".$row["prod\_name"]."</td>\n<td>" .$row["prod\_price"]. "</td>\n";

print "</tr>\n";

endwhile;

print "</table>";

mysql\_close();

?>

**mysql\_fetch\_array ()**

Функция mysql\_fetch\_array( ) аналогична mysql\_fetch\_row( ), однако по умолчанию значения полей записи сохраняются в ассоциативном массиве. Можно выбрать тип индексации (ассоциативная, числовая или комбинированная). Синтаксис функции mysql\_fetch\_array( ):

array mysql\_fetch\_array (resource идентификатор результата [, тип\_индексации])

В параметре идентификатор\_результата передается значение,

возвращенное функцией mysql\_query( ). Необязательный параметр тип\_индексации принимает одно из следующих значений:

MYSQL\_ASSOC — функция mysql\_fetch\_array( ) возвращает

ассоциативный массив. Если параметр не указан, это значение используется по умолчанию;

MYSQL\_NUM — функция mysql\_fetch\_array( ) возвращает массив с числовой индексацией;

MYSQL\_BOTH — к полям возвращаемой записи можно обращаться как по числовым, так и по ассоциативным индексам.

**Задание**

На рисунке 2 изображена структура БД каталога продукции.

Здесь таблица TYPES – содержит информацию о типах продукции или услугах. Где ID\_TYPE – первичный ключ таблицы; NAME\_TYPE – наименование продукции или услуг.

Таблица PRODUCTS – содержит список продукций или услуг. Первичным ключем является поле ID\_PROD, вторичным ключем, т.е. ссылкой на тип продукции – ID\_TYPE. Наименование продукции или услуг содержится в поле NAME\_PROD, а в поле DESCR – ее описание. В поле COST – цена на ту или иную продукцию или вид услуг. Поля SMALL\_PICT и PICT содержат ссылки на маленькую и большую картинки соответственно.

Таблица PROPERT содержит характеристики той или Инной продукции или вида услуг. Здесь первичным и вторичным ключами являются поля ID\_PROP и ID\_PROD соответственно. В поле NAME\_PROP содержится информация о наименовании свойства, а в поле VAL\_PROP – значение свойства.

В соответствии с данной структурой базы данных создать каталог продукции, позволяющий просматривать типы продукции или видов услуг. В соответствии с выбранным пользователем типом выводить список продукции или видов услуг, соответствующий выбранному типу. По каждому наименованию продукции или услуг предусмотреть возможность просмотра его свойств.

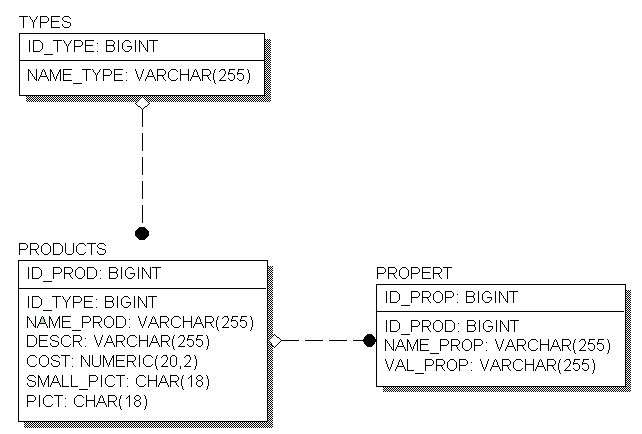


Рисунок 2

Контрольные вопросы

1. Назовите последовательность работы с БД в PHP?

2. В чем отличие методов mysql\_connect и mysql\_pconnect?

3. Каким образом узнать количество записей, затронутых командой UPDATE?

Содержание отчёта

Отчёт по практическому заданию должен содержать следующие сведения:

1. Ответ на контрольные вопросы;
2. название и цель работы;
3. исходные тексты разработанной программы с комментариями;
4. результат работы программы.