**Лабораторная работа №2**

**Взаимодействие PHP и MySQL**

**Пример:**

Нам потребуются две таблицы: сотрудники и отделы, назовем их emps и depts соответственно.

Столбцы таблицы отделов (depts):

* id\_dept(первичный ключ)
* name(название отдела)

Столбцы таблицы сотрудников (emps):

* id\_emp(первичный ключ)
* id\_dept(внешний ключ, ссылающийся на таблицу depts)
* first\_name(имя)
* middle\_name(отчество)
* last\_name(фамилия)

Таблицы реляционной базы данных характерны тем, что каждая должна содержать первичный ключ. Каждый сотрудник работает в каком-то отделе. Таблица emps ссылается на таблицу depts с помощью столбца id\_dept. Столбец id\_dept в данном случае называется внешним ключом.

Пусть в нашей фирме два отдела: «Бухгалтерия» и «Маркетинг». В бухгалтерии работают Иванов Иван Иванович, Петров Петр Петрович и Сидорова Елена Николаевна. Маркетингом занимаются Ушаков Павел Павлович и Ефремов Илья Викторович.

**Таблица depts:**

|  |  |
| --- | --- |
| **id\_dept** | **name** |
| **1** | Бухгалтерия |
| **2** | Маркетинг |

**Таблица emps:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id\_emp** | **id\_dept** | **first\_name** | **middle\_name** | **last\_name** |
| **1** | 1 | Иван | Иванович | Иванов |
| **2** | 1 | Петр | Петрович | Петров |
| **3** | 2 | Павел | Павлович | Ушаков |
| **4** | 2 | Илья | Викторович | Ефремов |
| **5** | 1 | Елена | Николаевна | Сидорова |

Если вы тестируете сайт на локальном компьютере, то параметры конфигурации должны быть следующими (если вы, конечно, не меняли конфигурацию):

$server = 'localhost';

$username = 'root';

$password = ''';

Когда сайт работает на удаленном сервере, параметр $server остается «localhost», $username и $password определяются при создании пользователя базы данных.

**Подключение к базе данных возможно с использованием расширений MySQLi и PDO.**

**MySQLi**

**MySQLi** (MySQL Improved) — расширение драйвера реляционных СУБД, используемого в языке программирования PHP для предоставления доступа к базам данных MySQL. **MySQLi** является обновлённой версией драйвера PHP MySQL, и даёт различные улучшения в работе с базами данных.

Перед тем, как работать с базой данных, необходимо установить с ней соединение. Для этого служит функция **mysqli\_connect().** Результат выполнения функции – дескриптор соединения, который пригодится, только если вы собираетесь работать сразу с несколькими подключениями. В большинстве случаем это не требуется, и результат выполнения функции проверяют лишь на неравенство false (что означает, что подключение прошло успешно).

int mysqli\_connect (

[ string $server = ini\_get("mysqli.default\_host") [,

string $username = ini\_get("mysqli.default\_user") [,

string$password = ini\_get("mysqli.default\_password") [,

bool $new\_link = false [,

int $client\_flags = 0]]]]])

Функция принимает множество параметров и все они необязательны. Подробнее о их назначении можно прочитать в документации. Чаще всего функция вызывается с тремя параметрами:

mysqli\_connect($server, $username, $password, $dbname);

Если вы тестируете сайт на локальном компьютере и у вас установлен пакет Денвер, то параметры должны быть следующими (если вы, конечно, не меняли конфигурацию):

$server = 'localhost';

$username = 'root';

$password = ''';

$dbname = 'test';

Когда сайт работает на удаленном сервере, параметр $server остается «localhost», $username и $password определяются при создании пользователя базы данных.

int mysqli\_select\_db(string $dbname [, int $link\_identifier])

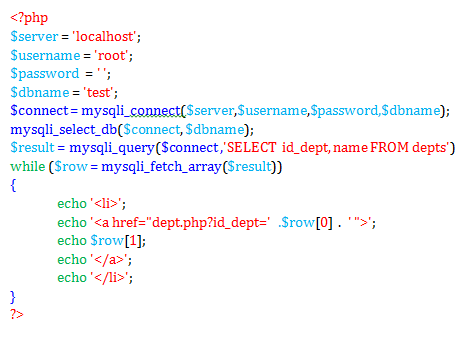
До того как послать первый запрос серверу MySQL, необходимо указать, с какой базой данных мы собираемся работать. Для этого и предназначена функция mysqli\_select\_db(). Она уведомляет PHP, что в дальнейших операциях с соединением $link\_identifier (или с последним открытым соединением, если указанный параметр не задан) будет использоваться база данных $dbname. Функция возвращает true в случае успеха, иначе false.

int mysqli\_query ( mysqli $link , string $query [, int $resultmode = MYSQLI\_STORE\_RESULT ] )

Запрос к базе данных. Текст запроса формулируется на языке SQL. Для запросов на выборку данных функция возвращает идентификатор результата в случае успеха и false в случае ошибки. Для запросов, не подразумевающих результат (INSERT, UPDATE, DELETE) функция в случае успеха возвращает true.

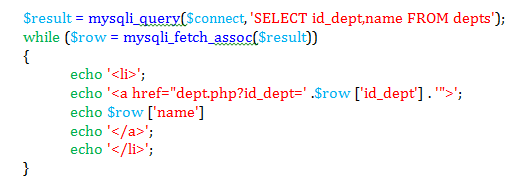
array mysqli\_fetch\_array(int $result)

Функция извлекает очередную строку результата выборки данных. В качестве параметра принимает идентификатор, полученной вызовом функции mysqli\_query(). Возвращает массив, состоящий из значений каждого столбца текущей строки. В качестве ключа – порядковый номер столбца (начиная с нуля). Когда извлекать больше нечего, функция возвращает false.



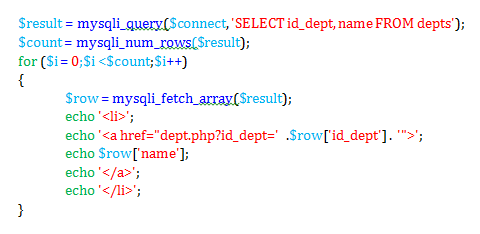
array mysqli\_fetch\_assoc(int $result)

Функция аналогична mysqli\_fetch\_array(), но возвращает ассоциативный массив, в котором в качестве ключа выступает имя столбца.



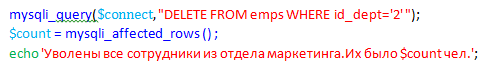
int mysqli\_num\_rows(int $result)

Функция возвращает число строк, содержащееся в результате выборки данных.



int mysqli\_affected\_rows([resource $link\_identifier])

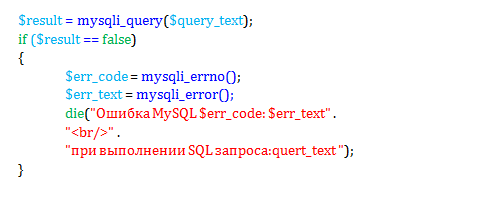
Функция возвращает число строк, затронутых последним запросом INSERT, UPDATE или DELETE.



int mysqli\_errno([ int $link\_identifier])

string mysqli\_error([ int $link\_identifier])

Если в процессе работы с MySQL возникают ошибки, то сообщение об ошибке и ее номер можно получить с помощью этих двух функций. Первая возвращает номер последней зарегистрированной ошибки. Вторая - строку, содержащую текст сообщения об ошибке. Ее удобно применять в отладочных целях.



**PDO**

Соединения устанавливаются автоматически при создании объекта PDO от его базового класса. Не имеет значения, какой драйвер вы хотите использовать; все что требуется, это имя базового класса. Конструктор класса принимает аргументы для задания источника данных (DSN), а также необязательные имя пользователя и пароль (если есть).

Чтобы установить подключение к базе достаточно создать экземпляр класса PDO и передать данные для подключения.

**Рассмотрим пример подключение к базе данных:**

<? Php

$user = 'root';

$pass = 'usbw';

try

{$dbh = new PDO ('MySQL: host=localhost; port=3307; dbname=test; charset=utf8', $user, $pass);

Foreach ($dbh->query ('SELECT \* from FOO') as $row)

print\_r ($row);

$dbh = null;

} catch (PDOException $e)

{print "Error!: " . $e->getMessage () . "<br/>";

Die () ;}

?>

PDO::query

Если в запрос не передаются никакие переменные, то можно воспользоваться функцией query(). Она выполнит запрос и вернёт специальный объект — PDO statement. Получить данные из этого объекта можно как традиционным образом, через while, так и через foreach().

$stmt = $pdo->query('SELECT name FROM depts');  
while ($row = $stmt->fetch())  
{  
    echo $row['name'] . "\n";  
}

PDO::fetch

Этот метод является аналогом функции mysq\_fetch\_array() и ей подобных, но действует по-другому: вместо множества функций здесь используется одна, но ее поведение задается переданным параметром.

$stmt = $pdo->prepare('SELECT name FROM users WHERE email = ?');  
$stmt->execute([$\_GET['email']]);  
while ($row = $stmt->fetch(PDO::FETCH\_LAZY))  
{  
    echo $row[0] . "\n";  
    echo $row['name'] . "\n";  
    echo $row->name . "\n";  
}

PDO::Fetch\_Assoc

Возвращает массив с названиями столбцов в виде ключей.

$STH = $DBH->query('SELECT id\_dept , name from depts');

$STH->setFetchMode(PDO::FETCH\_ASSOC);

while($row = $STH->fetch()) {

**echo** $row['id\_dept'] . "\n";

**echo** $row['name'] . "\n";

}

PDOStatement::rowCount

public int **PDOStatement::rowCount** ( void )

Функция возвращает число строк, содержащееся в результате выборки данных.

<?php  
$del = $dbh->prepare("DELETE FROM emps WHERE id\_dept='2'");  
$del->execute();  
print("Возвращает количество удаленных строк :\n");  
$count = $del->rowCount();  
print("Удалено $count строк.\n");  
?>

PDO: ErrorInfo

Если в процессе работы с MySQL возникают ошибки, то сообщение об ошибке и ее номер можно получить с помощью этой функции. Функция получает расширенную информацию об ошибке, произошедшей в ходе последнего обращения к базе данных.

public array **PDO::errorInfo** ( void )

**PDO::errorInfo()** возвращает массив с информацией об ошибке произошедшей в ходе выполнения последней операции с базой данных. Массив содержит следующие поля:

* 0-Код ошибки SQLSTATE (пятисимвольный идентификатор определенный в стандарте ANSI SQL).
* 1-Код ошибки, заданный драйвером.
* 2-Выданное драйвером сообщение об ошибке.

**Задания к лабораторной работе**

На языке PHP реализовать интерфейс для взаимодействия с БД, созданной в прошлой лабораторной работе, позволяющий выполнить операции добавления,удаления, редактирования данных всех таблиц, включающий:

* Формы добавления новой записи в таблицы. Поле id не вводится. При добавлении новой книги, автор и жанр должны выбираться из списка НЕ по ID.
* Интерфейс удаления записи таблицы;
* Формы редактирования записи таблицы. Редактирование – это не перезаписывание. При редактировании в форме должны отображаться старые данные с возможностью их корректировки.
* Произвольный запрос на выборку данных с полем фильтрации данных (данные для условия поиска задаются не статически в запросе, а выбираются пользователем в соответствующих полях формы). 