**Практическое задание № 7**

**РАЗРАБОТКА НА PHP СРЕДСТВ АДМИНИСТРИРОВАНИЯ, ОБЕСПЕЧИВАЮЩИХ РЕГИСТРАЦИЮ И ВХОД ПОЛЬЗОВАТЕЛЕЙ НА САЙТ.**

**Цель работы:** Научиться использовать функции доступа к данным СУБД MySQL и сессии для организации, хранения и обработки информации на сайте.

**Теоретические сведения**

**Работа с данными в СУБД MySQL**

Предположим, что имеется текстовый файл, содержащий список товаров, которые продаются организацией. Можно написать PHP-сценарий, который для построчного считывания списка товаров многократно вызывает функцию fgets(). Затем, после каждого очередного считывания, вызывается инструкция echo или встроенная функцию print(). В результате будет создана веб-страница, содержащая список предлагаемых товаров. Таким образом, РНР и текстовые файлы можно использовать для создания содержимого веб-страницы на лету.

База данных SQL, как и текстовый файл, представляет собой всего лишь набор данных. Тем не менее, в отличие от текстовых файлов, в которых приложения хранят данные в виде неструктурированных фрагментов, СУБД управляет всеми операциями с данными в базе данных SQL и организует данные оптимальным образом для обеспечения быстрого и легкого доступа к данным и манипулирования ими.

Поиск по текстовому файлу, содержащему несколько тысяч пар имен пользователей и паролей, занимает недопустимо длительное время. В то же время СУБД позволяет выполнить поиск такого же объема (среди десятков тысяч записей) в течение долей секунды. Поэтому СУБД дает возможность вывести на экран список товаров или одного товара (выбранного из длинного списка) гораздо быстрее, чем при использовании текстового файла со списком товаров.

Практически вся работа по созданию баз данных на сервере базы данных, созданию и модификации таблиц в них, наполнению, редактированию и удалению данных из таблиц баз данных и, наконец, считыванию, фильтрации и упорядочиванию получаемых данных осуществляется с помощью описанной ранее функции mysql\_query(), обеспечивающей интерфейс для обращения с запросами к базам данных:

int mysql\_query(string запрос[, int ид\_соединения])

Аргумент запрос содержит текст запроса на языке SQL. Если обработка запроса завершилась неудачей, функция возвращает FALSE. Ниже описывается применение этой PHP-функции для выполнения различных команд языка структурированных запросов SQL:

CREATE – создать базу данных или таблицу;

INSERT – добавить запись данных в таблицу;

SELECT – получить запись данных из таблицы;

UPDATE – обновить запись данных в таблице;

DELETE – удалить запись данных из таблицы.

**Создание базы данных**

Для создания базы данных нужно воспользоваться SQL-командой CREATE DATABASE, синтаксис которой следующий:

CREATE DATABASE имя\_базы\_данных

На языке PHP создание новой базы данных с именем «database1» для уже открытого соединения $link1будет выглядеть следующим образом:

$query = "CREATE DATABASE database1"

$result = mysql\_query($query, $link1);

Если работа с сервером СУБД ведется только через одно соединение, то при вызове функции mysql\_query() ссылку на него можно опустить.

**Создание таблицы базы данных**

После создания базы данных необходимо создать в ней таблицы, в которых будут храниться данные. Каждая таблица состоит из строк связанных данных, организованных в виде столбцов. Каждая строка, называемая записью, содержит связанные между собой элементы данных. Синтаксис команды языка SQL для создания таблицы следующий:

CREATE TABLE имя\_таблицы (определение столбца1,

определение столбца2,

…

)

Для каждого из столбцов, разделенных запятой, задается имя столбца и тип столбца. Наиболее часто используемыми типами являются TEXT, VARCHAR (переменное число символов), DATETIME и INT (целое). Настоятельно рекомендуется создавать в каждой таблице столбец, который действует как первичный ключ (столбец, все значения которого являются уникальными, и поэтому могут использоваться для выбора каждой строки). С учетом этого простая команда CREATE TABLE может выглядеть так:

CREATE TABLE my\_table (id INT PRIMARY KEY,

information TEXT

)

Например, при необходимости создания в базе данных таблицы регистрации пользователей последовательность инструкций языка PHP может быть следующей:

$query = "CREATE TABLE database1.register(

SN VARCHAR(255) DEFAULT NULL,

FN VARCHAR(255) DEFAULT NULL,

PN VARCHAR(255) DEFAULT NULL,

Email VARCHAR(255) DEFAULT NULL,

Phone DECIMAL(15, 0) DEFAULT NULL,

Login VARCHAR(255) NOT NULL,

Password VARCHAR(255) DEFAULT NULL

)";

$result = mysql\_query($query, $link1);

**Включение данных в базу данных**

Очевидно, что таблица должна содержать данные, которые в нее можно записать с помощью SQL-команды INSERT, синтаксис которой имеет два варианта:

INSERT INTO имя\_таблицы (имя\_столбца1, имя\_столбца2, …)

VALUES (значение1, значение2, …)

INSERT INTO имя\_таблицы

VALUES (значение1, значение2, значение3, …)

Первый вариант команды добавляет в указанную таблицу новую запись и присваивает заданные в скобках после VALUES значения указанным в скобках после имени таблицы полям записи. Второй вариант присваивает заданные значения полям подряд, начиная с первого поля (имеющего индекс 0).

Например, при включении данных об очередном регистрирующемя пользователе инструкций языка может такой:

$query = "INSERT INTO register

VALUES('$lN', '$fN', '$pN', '$eA', '$ma', '$lI', '$pW')";

$result = mysql\_query($query, $link1);

Переменные, имена которых указаны в списке значений команды INSERT INTO, содержат значения, которые очередной пользователь ввел в форме регистрации.

**Получение данных из базы данных**

Как уже говорилось ранее, получение данных из одной или нескольких таблиц базы данных производится с помощью команды (запроса) SELECT языка SQL. Основной синтаксис запроса SELECT для получения данных следующий:

SELECT список столбцов FROM список таблиц

Наиболее простой запрос для считывания данных одной таблицы:

SELECT \* FROM имя\_таблицы

Звездочка эквивалентна перечислению имен всех столбцов. Если требуется, чтобы были получены только некоторые столбцы, можно ограничить запрос примерно так:

SELECT Login, Password FROM register

Этот запрос требует, чтобы была возвращена информация только двух столбцов с логинами и паролями пользователей. Нужно понимать, что такой запрос не ограничивает число получаемых записей – они получаются все, а ограничивается только получаемые столбцы для этих записей – полученная результирующая таблица будет состоять только из двух столбцов, но изо всех записей.

Ограничение количества получаемых записей достигается добавлением в запрос предложения WHERE:

SELECT \* FROM register WHERE Login='Eugen'

Этот запрос получает все поля записи, в которой значение столбца с именем Login равно «Eugen».

Как уже говорилось ранее, доступ к данным, полученным с помощью запроса SELECT, в программе на PHP осуществляется с помощью идентификатора результата, возвращаемого функцией mysql\_query(), например.

$query = "SELECT Login, Password FROM register

WHERE Login='Eugen'";

$result = mysql\_query($query, $link1);

print "Ваш пароль = ".mysql\_result($result, 0, 1).\n";

**Удаление данных из базы данных**

Команда DELETE языка SQL обеспечивает удаление записей таблицы базы данных. Основной синтаксис команды DELETE такой:

DELETE FROM имя\_таблицы WHERE имя\_столбца=значение

Предложение WHERE не является обязательным, но если оно опущено, будут удалены из таблицы все записи. Необходимо также понимать, что удаленные записи никоим образом нельзя восстановить, если не имеется резервной копии базы данных.

В качестве предохранительной меры, если хочется удалить только одну запись из таблицы, лучше добавить в запрос предложение LIMIT:

DELETE FROM имя\_таблицы WHERE имя\_столбца=значение LIMIT 1

Это предложение гарантирует, что будет удалено не более одной записи. После определения запроса на удаление, он выполняется, используя функцию mysql\_query() function, так же как и для других запросов, например:

$query = "DELETE FROM register WHERE Login='Eugen'";

$result = mysql\_query($query, $link1);

Чтобы проверить, как сработал запрос DELETE, можно использовать функцию mysql\_affected\_rows(). Эта функция возвращает число записей, затронутых запросами INSERT, DELETE или UPDATE.

**Обновление данных в базе данных**

Для изменения значений полей записи используется команда UPDATE языка SQL. Синтаксис команды следующий:

UPDATE имя\_таблицы SET имя\_столбца1= значение,

имя\_столбца2= значение,

…

WHERE имя\_столбца=значение

Как и с любым другим запросом, если значение является строкой, оно должно заключаться в одиночные кавычки:

UPDATE register SET FN = 'Алексей', SN='Иваненко'

WHERE Login='Alex'

Как и с запросом DELETE, предложение WHERE нужно использовать, чтобы ограничить определить записи, значения которых изменяются. Если этого не сделать, все записи в таблице базы данных будут обновлены. Чтобы проверить как работает обновление, можно воспользоваться функцией mysql\_affected\_rows(), которая возвратит число обновленных записей, например:

$query = "UPDATE register SET FN = 'Алексей', SN='Иваненко'

WHERE Login='Alex'";

$result = mysql\_query($query, $link1);

print "Обновилось записей: ".mysql\_affected\_rows($link1).\n";

**Сессии (сеансы) в PHP**

Сессии и cookies предназначены для хранения сведений о пользователях при переходах между несколькими страницами. При использовании сессий данные сохраняются во временных файлах на сервере. Файлы с cookies хранятся на компьютере пользователя, и по запросу отсылаются броузером серверу.

Использование сессий и cookies очень удобно и оправдано в таких приложениях как Интернет-магазины, форумы, доски объявлений, когда, во-первых, необходимо сохранять информацию о пользователях на протяжении нескольких станиц, а, во-вторых, своевременно предоставлять пользователю новую информацию.

Протокол HTTP является протоколом "без сохранения состояния". Это означает, что данный протокол не имеет встроенного способа сохранения состояния между двумя транзакциями. Т. е., когда пользователь открывает сначала одну страницу сайта, а затем переходит на другую страницу этого же сайта, то основываясь только на средствах, предоставляемых протоколом HTTP невозможно установить, что оба запроса относятся к одному пользователю. Т. о. необходим метод, при помощи которого было бы отслеживать информацию о пользователе в течение одного сеанса связи с Web-сайтов. Одним из таких методов является управление сеансами при помощи предназначенных для этого функций. Для нас важно то, что сеанс по сути, представляет собой группу переменных, которые, в отличие от обычных переменных, сохраняются и после завершения выполнения PHP-сценария.

При работе с сессиями различают следующие этапы:

• открытие сессии,

• регистрация переменных сессии и их использование,

• закрытие сессии.

**Открытие сессии**

Самый простой способ открытия сессии заключается в использовании функции

bool session\_start();

которая вызывается в начале PHP-сценария.

Эта функция проверяет, существует ли идентификатор сессии, и, если нет, то создает его. Если идентификатор текущей сессии уже существует, то загружаются зарегистрированные переменные сессии.

**Регистрация переменных сессии**

После инициализации сессии появляется возможность сохранять информацию в глобальном массиве $\_SESSION. Пусть имеется файл index.php в котором в массив $\_SESSION сохраняется переменная и массив.

<?php

// Инициализация сессии

session\_start();

// Сохранение значения в сессии

$\_SESSION['name'] = "value";

// Сохранение массива в сессии

$arr = array("first", "second", "third");

$\_SESSION['arr'] = $arr;

// Вывод гиперссылки на другую страницу

echo "<a href='other.php'>другая страница</a>";

?>

На страницах, где происходит вызов функции session\_start(), значения данных переменных можно извлечь из глобального массива $\_SESSION. В следующем листинге приводится содержимое страницы other.php, где извлекаются данные, ранее помещенные на странице index.php.

<?php

// Инициализация сессии

session\_start();

// Вывод содержимого глобального массива $\_SESSION

echo "<pre>";

print\_r($\_SESSION);

echo "</pre>";

?>

**Закрытие сессии**

После завершения работы с сессией сначала нужно разрегистрировать все переменные сессии, а затем вызвать функцию unset():

unset($\_SESSION["username"]);

**Пример простой сессии**

Рассмотрим пример простой сессии, работающей с тремя страницами. При посещении пользователем первой страницы открывается сессия и регистрируется переменная $username.:

<?

session\_start();

$\_SESSION['username'] = "maksim";

echo 'Привет, '.$\_SESSION['username']."<br>";

?>

<a href="page2.php">На следующую страницу </a>

После этого, пользователь maksim нажимает на ссылку и попадает на страницу page2.php, код которой приведен в листинге:

<?

session\_start();

echo $\_SESSION['username'].' , Вы пришли на другую страницу этого сайта!';

echo("<br>");

?>

<a href="page3.php">На следующую страницу </a>

При нажатии на ссылку, пользователь попадает на страницу page3.php, при этом происходит разрегистрация сеансовой переменной и уничтожение сессии. Соответствующий код реализации приведен в листинге:

<?

session\_start();

unset($\_SESSION['username']); // разрегистрация переменной

echo 'Привет, '.$\_SESSION['username'];

/\* теперь имя пользователя уже не выводится \*/

session\_destroy(); // разрушение сессии

?>

**Задание**

Разработать форму входа пользователей на сайт, регистрационную форму и форму приветствия пользователя в соответствии с вариантом задания (табл. 1) и примерами первых двух форм, приведенных ниже.

|  |
| --- |
|  |
| **Рисунок 1. Пример формы входа на сайт** |

На рисунке приведен минимум информации, которая должна содержаться на форме входа. Программа на PHP должна проверить правильность ввода информации в поля ввода и, если она введена правильно, найти запись с данными регистрации данного пользователя в таблице регистрации в СУБД, сравнить введенный пароль со значением пароля, хранящегося в таблице регистрации.

Если пароль введен правильно, отобразить в браузере форму приветствия с контентом по усмотрению студента. Как минимум, эта форма должна содержать таблицу с данными из СУБД, вывод которой был предусмотрен в лабораторной работе №4.

Если логин и/или пароль введены неправильно, то нужно сообщить об этом пользователю и предложить ему повторно ввести данные или же предложить ему зарегистрироваться с помощью гиперссылки «Регистрация».

На рис. 2 показан пример формы регистрации. Если при регистрации пользователь ввел синтаксически правильную информацию, нужно проверить, что в СУБД отсутствует информация о пользователе с указанным логином. Если это так, добавить новую запись в таблицу регистрации пользователей СУБД.

|  |
| --- |
|  |
| **Рисунок 2. Пример регистрационной формы** |

Если же информация с указанным логином уже имеется в СУБД, нужно запросить пользователя о необходимости подтверждения коррекции информации, содержащейся в СУБД. После подтверждения пользователем такой коррекции, нужно обновить те данные, которые он ввел в форму регистрации. При этом обязательным условием коррекции информации пользователя является ввод правильного пароля. (Для этого необходимо предусмотреть возможность пересылки пользователю его пароля по электронной почте).

Если регистрация прошла успешно, отобразить в браузере форму приветствия с контентом по усмотрению студента.

Нужно обеспечить проверку, имеется ли на сервере MySQL нужная база данных. Если база данных отсутствует, программа на PHP должна ее создать. Должна быть также обеспечена проверка на наличии в базе данных таблицы для регистрации пользователей. Если она отсутствует, ее также нужно программно создать.

Требования к форме:

1. Во всех полях выполнять проверку на пустое поле.
2. Поля "Фамилия", "Имя", "Отчество" должны содержать только буквы.
3. Поля "Фамилия", "Имя", "Отчество" - автоматическая смена регистра букв по маске Ххххххх. Внимание! Строковые функции по автоматической смене регистра букв некорректно работают с кириллицей.
4. Поле "Адрес E-mail" - Автоматическая смена регистра на маленький. Проверка на присутствие пробела, символа @ и точки.
5. Поле "Телефон" должно содержать только цифры.
6. Поле "Логин" - количество символов в логине от 4-х до 8-ми. В логине должны использоваться только символы "a-z". Автоматическая смена регистра на маленький.
7. Поле "Пароль" - Введенные данные должны соответствовать полю "Повтор пароля". Количество введенных символов от 6-ти до 10-ти.
8. Проверить корректность заполнения не названных выше полей в соответствии с вариантом.
9. Вывести либо сообщение об ошибке, либо текст «Вы ввели такие данные:…»

Таблица 1 – Варианты заданий

|  |  |
| --- | --- |
| **№ варианта** | **Поля регистрационной формы** |
| 1 | Название: «Регистрационная форма студента, желающегопринять участие в курсе обучения, проводимом компанией такой-то*»*  Поля:  Фамилия  Имя  Отчество  Телефон контактный  Телефон мобильный  Адрес электронной почты  Страна (выпадающий список с вариантами)  Информация об учебном заведении:  Название  Город |
| 2 | Регистрационная форма участника международной конференции «…»  Поля формы:  Фамилия  Имя  Отчество  Организация  должность  ученая степень (выпадающий список с вариантами: к.т.н., д.т.н., нет)  Нужна ли гостиница? (радиокнопки: да, нет) |
| 3 | Регистрационная форма участника семинара «…»  Фамилия  Имя  Дата рождения (день, месяц, год)  Пол (варианты ответа с радиокнопками)  Номер паспорта  Место работы  Тема выступления на семинаре |
| 4 | Регистрационная форма  Поля:  ФИО  Должность  Телефон  Факс  E-mail  Почтовый адрес  Почтовый индекс |
| 5 | Семинар «Проблемы теоретической физики», посвященный 100-летию со дня рождения Л.Д. Ландау  Поля:  Фамилия  Имя  Отчество  Должность  Организация  Адрес организации  Электронный адрес  Дата приезда  Гостиница (нужна / не нужна - радиокнопки) |
| 6 | Регистрационная форма для студентов-участников проекта такого-то:  Группа  ФИО  Контактная информация:  e-mail  телефон (моб)  телефон (дом)  Сообщите, какие оценки Вы получили на экзаменах по следующим дисциплинам:  линейная алгебра  математический анализ  теория вероятностей  математическая статистика  (ответы либо в виде списка, либо радиокнопки) |
| 7 | Поля в регистрационной форме, приведенной на рис. 1 |

Содержание отчёта

Отчёт по практическому заданию должен содержать следующие сведения:·

1. название и цель работы;·
2. исходные тексты разработанной программы с комментариями;·
3. результат работы программы.