



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

" **Московский технологический университет** "

МИРЭА

Институт информационных технологий

Подлежит возврату

№

ИНТЕРНЕТ-ТЕХНОЛОГИИ И СИСТЕМЫ

Методические указания
по выполнению лабораторных работ

Часть 4

(Работы № 7-12)

для студентов

Направление подготовки

09.03.04 Программная инженерия

Профиль подготовки

Интеллектуальные программные системы и комплексы

МОСКВА 2017

Составитель И.И. Холкин
Редактор В.М. Панченко

Методические указания к лабораторным и практическим работам по дисциплине «Интернет-технологии и системы», Часть 4, предназначены для студентов 4-го курса очной формы обучения (квалификация Бакалавр. Направление подготовки 09.03.04 Программная инженерия. Профиль подготовки Интеллектуальные программные системы и комплексы).

Предполагается, что студенты выполнили лабораторные работы по дисциплине Интернет- технологии и системы [1, 8, 9] и особенно лабораторные работы № 5,6, где рассмотрена установка Apache, PHP, MySQL, phpMyAdmin и разобраны примеры создания серверных скриптов и динамических страниц на PHP. Установлены Денвер и среда разработки, например, WeBuilder 2015 v13.3.0.165 Final. В данных лабораторных и практических работах (№ 7 -12) рассматривается использование PHP и MySQL на примере реализации форума [2]. Работы закрепляют и расширяют полученные знания в области Интернет-технологий. Печатается по решению редакционно-издательского совета университета.

Рецензенты: Б.Б. Чумак,
А.Н. Райков

©МИРЭА, 2017

Лабораторная работа № 7

Практическое занятие № 7

Разработка структуры базы данных. Создание базы данных в phpMyAdmin

Цель работы

Ознакомление с принципами, построения **базы данных**. Формирование навыков создания базы данных в **phpMyAdmin**.

Методические указания

1. Проектирование интернет-ориентированных баз данных

Разработка современного Web-ресурса связано, как правило, со встраиванием в него базы данных (БД) и системы управления ею (СУБД). При этом на стороне Web-клиента в среде браузера должен быть спроектирован интерфейс работы с БД, а на стороне Web- сервера — Web-модули, обрабатывающие запросы клиента и создающие динамические страницы, соответствующие этим запросам, и возвращаемые клиенту. Таким образом, сама БД и большая часть СУБД располагаются на стороне Web- сервера, а браузер обеспечивает лишь интерфейс работы с БД.

Часто с целью разгрузки Web-сервера в рамках СУБД на браузер возлагают небольшие задачи, связанные с проверкой вводимых пользователем данных, улучшением средств навигации по БД, выводом сообщений и т. д. Подобные задачи решаются с помощью клиентских скриптов. Имеется ряд средств, обеспечивающих встраивание БД и СУБД в Web-ресурс. Ниже рассматриваются некоторые из них.

Важно помнить, что при решении задач встраивания БД и СУБД в Web-ресурс на локальном компьютере, на него должен быть установлен Web-сервер, совместимый с указанными средствами.

1.1. Создание Интернет-приложений с использованием PHP и MySQL

PHP (Hypertext Preprocessor — Препроцессор гипертекста). — скрипт-язык, который непосредственно встраивается в HTML-код и выполняется Web-сервером. Создателем PHP (1994 г.) является Расмус Лердорф.

Сейчас PHP — это быстро развивающееся средство программирования, работающее на очень многих серверах в Интернете.

MySQL— это надежная и простая в администрировании СУБД с открытым кодом, предлагаемая бесплатно. Богатство возможностей MySQL объясняет, почему с ней работают такие крупные организации, как **Yahoo!**, **Бюро переписей США** и **NASA**.

В [10] даются необходимые сведения о PHP/MySQL для

начинающих, рассматривается, как создать соединение с базой данных MySQL из PHP, посылать к ней запросы, анализировать результаты их выполнения, проверять данные на наличие ошибок, создавать HTML-страницы, основываясь на полученных данных.

В [5] рассматривается установка PHP/MySQL и Apache-сервера для различных платформ, основные функции PHP, примеры практического применения.

В [6] подробно излагаются вопросы установки MySQL, язык SQL, проектирование БД и ее эксплуатация.

В [7] особое место занимает рассмотрение взаимодействия PHP с базами данных. В книге рассмотрены, как MySQL, так и SQLite, поддерживаемая PHP 5. Книга ориентирована на достижение реальных практических результатов. В деталях описано, как создать на PHP свою гостевую книгу, чат, форум, почтовую рассылку на сайте, новостную ленту и даже Интернет-магазин. Приведены готовые скрипты. Отдельно рассмотрена методика создания системы автоматического управления содержимым сайта («движка»). Такая система позволяет максимально упростить поддержку и обновление своего сайта.

PHP можно разделить на язык и библиотеку функций. Существует большое количество инструментальных средств для PHP: интерфейсы ко всем популярным СУБД, почтовым протоколам, разделяемой памяти, графическим файлам, архивам и множество других инструментов, с которыми можно познакомиться в [5].

PHP можно установить в двух вариантах [5]: как отдельный интерпретатор, работающий через интерфейс CGI, или как модуль Web-сервера, встроенный в сам сервер. В последнем случае проявляются все преимущества PHP.

Если пакет устанавливается в виде модуля Web-сервера, то функциональные возможности PHP объединяются с функциональными возможностями Web-сервера в одной программе. При этом выполнение происходит намного быстрее, чем при помощи обычных программ CGI, так как CGI-программа не запускается, а скрипт-код в HTML-файлах выполняется непосредственно процессом Web-сервера.

Создание скриптов PHP значительно проще, чем создание скриптов на других языках. Для решения разных специфических задач не нужно писать и отлаживать многочисленные маленькие CGI-программы, что сводит к минимуму продолжительность разработки страниц и сайта в целом. Вместе с тем PHP обладает огромным набором функций, которые могут быть значительно расширены с помощью дополнительных внешних библиотек.

Хотя существует достаточно много различных СУБД (например, MS

SQL Server, Firebird, ORACLE), наибольшую распространенность в среде Web-программирования получила MySQL, т. к. она обладает оптимальным соотношением цены, скорости работы и устойчивости к ошибкам среди аналогов, к тому же MySQL очень проста в изучении. Официальный сайт MySQL [http:// www.mysql.com](http://www.mysql.com), русскоязычная документация доступна по адресу www.mysql.com/doc/ru/index.html.

1.2. Разработка структуры базы данных

Рассмотрим работу с phpMyAdmin на примере реализации форума. Но прежде чем приступать к практическим действиям, составим план будущего проекта, потому что именно от грамотного составленного плана зависит важнейшая составляющая успеха реализации качественного проекта, с минимальным количеством ошибок и внесений изменений на этапе кодирования.

В заранее определенных разделах форума пользователи могут общаться на различные темы (рис. 1).

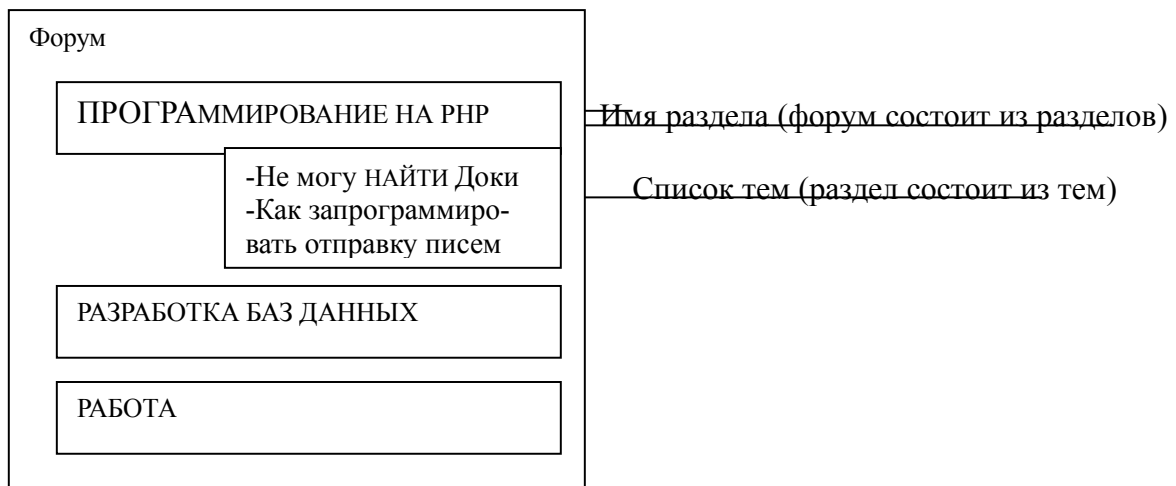


Рис. 1. Форум состоит из разделов, которые состоят из тем

Соответственно каждая тема состоит из набора сообщений пользователей (или посетителей), которые участвуют в ее обсуждении (рис. 2).

Каждая тема состоит из набора сообщений

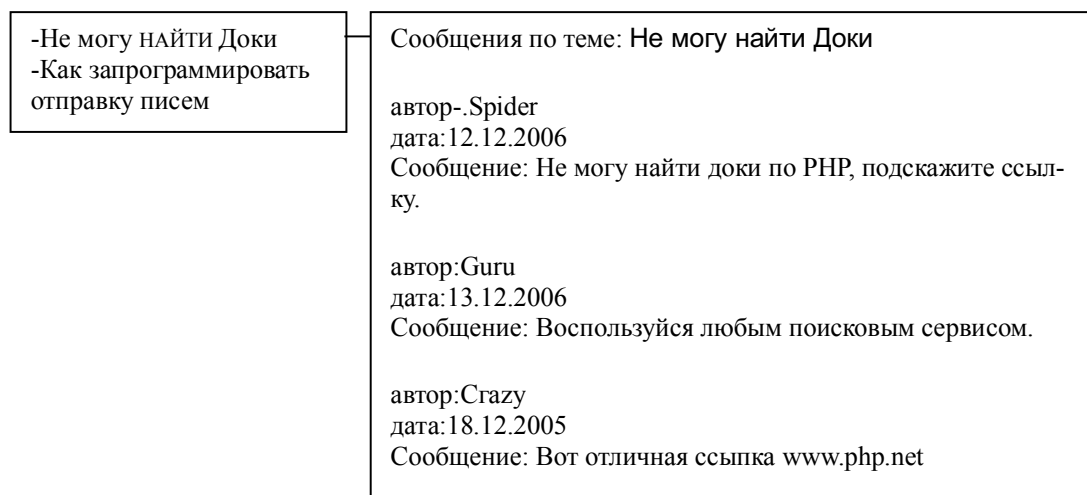


Рис. 2.. Каждая тема состоит из набора сообщений

Если пользователь зарегистрирован, то у него есть имя и пароль, которые он использует при авторизации на форуме. В противном случае его сообщения размещаются под ником Гость.

Информация об учетных записях посетителей форума будет храниться в таблице **users**, ее структура представлена в табл. 1.

Таблица 1. Структура таблицы **USERS**

№	Название поля	Описание
1	id	Поле-счетчик
2	name	Имя пользователя
3	pass	Пароль пользователя
4	role	Роль пользователя, возможен один из двух вариантов: admin — администратор форума или user — обычный пользователь

У каждого пользователя будет свой идентификационный номер, хранящийся в поле **id**, причем, нам не нужно беспокоиться о его уникальности, т. к. эту заботу можно смело возложить на **MySQL** (о том, как это сделать, будет сказано далее).

Также нам понадобится таблица **topic**, которая будет хранить информацию о разделах и темах форума, ее структура представлена в табл. 2.

Рассмотрим, как будут выглядеть записи (или строки) в таблице **topic** описывающие раздел (табл. 3).

Комментарии:

- в **id** будет храниться уникальный номер, который характеризует эту запись в пределах всей таблицы;
- **kodofrazdel** будет содержать 0;
- **name** будет содержать название раздела;
- **name_creator** будет пустым;
- **name_last_answer** будет пустым;
- **date_last_answer** будет пустым.

Таблица 2. Структура таблицы **TOPIC**

№	Название поля	Описание
1	id	Поле-счетчик
2	kodofrazdel	Это поле предназначено для темы, в нем будет храниться id раздела, которому принадлежит тема. Для раздела в этом поле будет храниться 0
3	name	Название раздела или темы
4	name_creator	Поле предназначено для темы, в нем будет храниться имя ее автора. Для раздела поле будет пустым

5	name_last_answer	Поле предназначено для темы, в нем будет храниться имя пользователя, ответ которого для данной темы был последним. Для раздела поле будет пустым
6	date_last_answer	Поле предназначено для темы, в нем будет храниться дата последнего ответа по теме. Для раздела поле будет пустым

Таблица 3. Записи для 2 разделов

id	kodofrazdel	name	name_creator	name_last_answer	date_last_answer
1	0	Для программистов	пустое	пустое	пустое
2	0	Работа	пустое	пустое	пустое

Рассмотрим, как будут выглядеть записи (или строки) в таблице **topic**, описывающие темы (табл. 4).

Таблица 4. Записи для 2 тем, раздела "Для программистов"

id	kodofrazdel	name	name_creator	name_last_answer	date_last_answer
3	1	Нужна помощь в поиске мануала по MySQL	Zero	MegaMan	2016-02-06
4	1	Подскажите, как установить PHP	Haos	Lion	2016-12-06

Комментарии:

id будет хранить уникальный номер, который характеризует эту запись в пределах всей таблицы;

kodof razdel будет содержать **id** раздела, которому принадлежит данная тема, в нашем случае темы относятся к разделу "Для программистов" (видите, как легко можно делить все записи таблицы по группам (раздел, тема) с помощью уникального номера);

name — название темы;

name_creator — имя автора, который создал эту тему;

name_last_answer — имя автора, чей ответ был последним по данной теме;

date_last_answer — дата последнего ответа по данной теме.

Также в базе данных **forum** будет присутствовать таблица **message**, предназначение которой состоит в хранении сообщений для всех тем, имеющих на форуме (табл. 5).

Таблица 5. Структура таблицы **message**

№	Название поля	Описание
1	id	Поле-счетчик
2	kodoftopic	Будет хранить id темы, которой принадлежит данное сообщение
3	textmessage	Текст сообщения
4	name_man	Имя автора, разместившего сообщение
5	date_answer	Дата размещения сообщения

Рассмотрим, как будут выглядеть записи в таблице **message**, описывающие сообщения (табл. 6).

Таблица 6. Записи для 2 сообщений, темы "Нужна помощь в поиске мануала по MySQL"

id	kodoftopic	text_message	name_man	date_answer
1	3	Где можно найти мануал по MySQL?	Zero	2006-01-05
2	3	Ты пробовал смотреть на официальном сайте?	MegaMan	2006-02-01

Следует отметить, что наличие в таблицах поля с уникальным номером позволяет связывать между собой информацию, содержащуюся в этих таблицах. Яркий пример этому связь между таблицами **message** и **topic** по полю **kodoftopic** и **id**.

1.3. Создание базы данных в phpMyAdmin

Дело в том, что у MySQL нет удобного графического интерфейса, поэтому для работы с этой СУБД чаще всего используют специальный инструмент, который позволяет до максимума упростить этот процесс. Его название- **phpMyAdmin**. **phpMyAdmin** полностью написан на PHP. Официальный сайт www.phpmyadmin.net, русскоязычный сайт доступен по адресу <http://www.php-myadmin.ru>.

Запустите **Денвер.и** наберите в строке браузера **localhost**, а затем выберите пункт **Утилиты** > **phpMyAdmin**, вы увидите следующее окно — рис. 1.

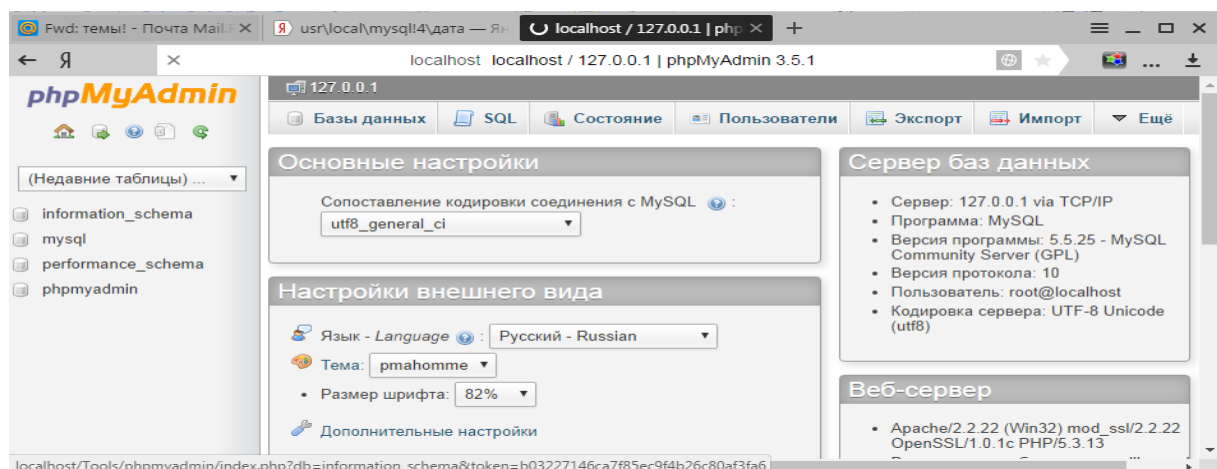


Рис. 1. Окно phpMyAdmin

Нажмите кнопку Базы данных. Вы получите окно Базы данных (Рис. 2).

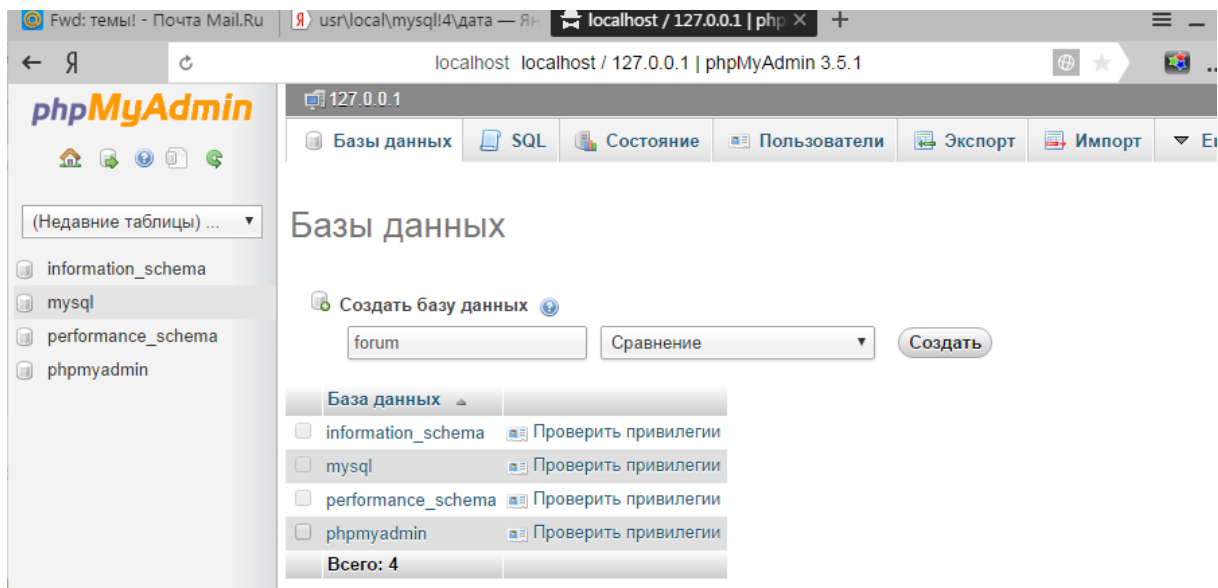


Рис. 2. Окно Базы данных

Введите **forum** (это имя будущей базы данных) в поле **Создать базу данных**. Далее нажмите кнопку **Создать**. Вы увидите следующее — рис. 3.

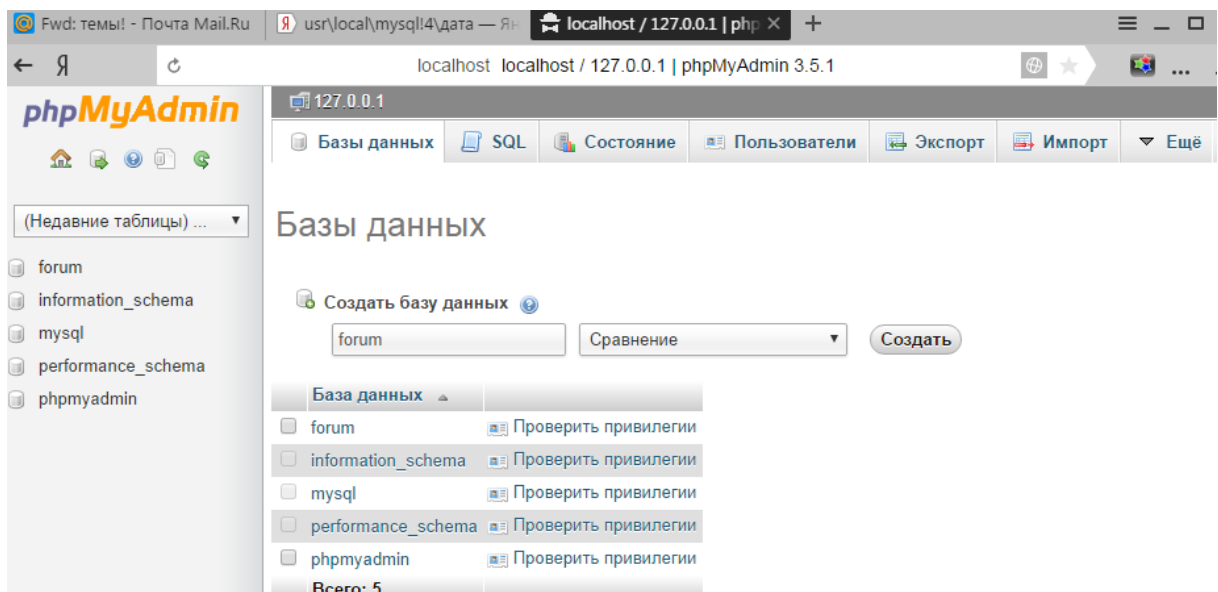


Рис. 3. Результат создания базы данных

База создана. В левой части и в центре phpMyAdmin вы можете видеть имя **forum** созданной базы данных.

Следует отметить, что язык SQL является связующим звеном между разработчиком и СУБД, т. к. последняя понимает именно этот язык. Поэтому, чтобы совершить какое-то действие, необходимо составить SQL-запрос и отправить его СУБД. Система phpMyAdmin берет эту заботу на себя, предоставляя разработчику визуальный интерфейс для работы с MySQL

Теперь создадим таблицу **users** (ее структура рассмотрена в табл. 1), для этого нажмите на кнопку **forum** и перейдите в окно **Создать таблицу** (Рис.4).

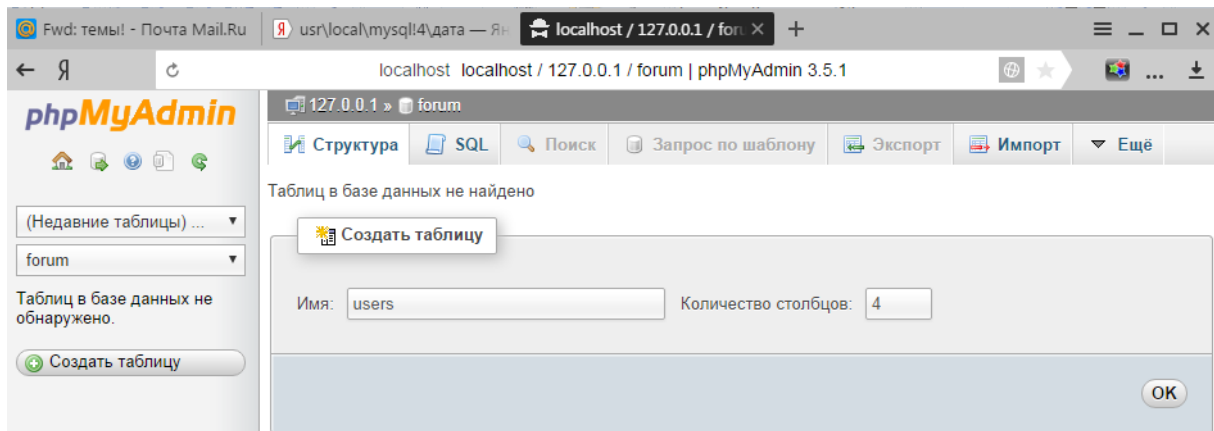


Рис. 4. Первый этап создания таблицы: ввод ее имени и числа столбцов, из которых она будет состоять

В поле **Имя**, расположенном под надписью **Создать таблицу**, введите **users**, а в поле **Количество столбцов** число 4.

Нажмите кнопку **ОК**, вы увидите набор из различных секций, каждая из которых состоит из однотипных элементов, предназначенных для ввода информации о будущих столбцах таблицы (рис. 5).

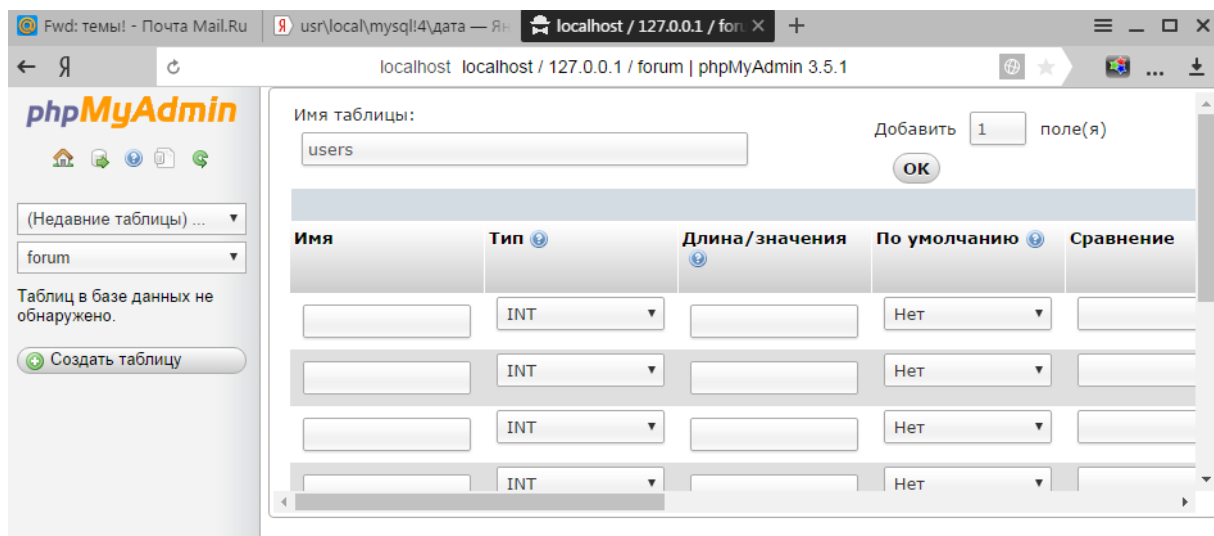


Рис. 5. Форма для ввода информации о столбцах для создаваемой таблицы

В секцию **Имя**, предназначенную для имени столбца, введите **id**, в секции **Тип** выберите **INT** — это тип данных для столбца.

Рассмотрим более подробно наиболее часто используемые типы данных:

varchar — текстовый тип, предназначенный для хранения до 255 символов;

text — текстовый тип данных, предназначенный для хранения до 65 535 символов;

int — числовой тип, предназначенный для хранения целых чисел в диапазоне от -2 147 483 648 до 2 147 483 647;

date — тип данных, предназначенный для хранения даты, как правило, она преобразуется к формату гggг-мм-дд.

Добавьте столбец **name** типа **varchar** с максимально возможным количеством символов, равным 20, а также столбец **pass** типа **varchar** с максимальным количеством символов, равным 32. Это нужно, т. к. в нем будет храниться хеш— 32-символьное 16-ричное число пользовательского пароля, возвращенный хеш-функцией **md5()**.

Хэширование — одностороннее математическое преобразование (на основе алгоритма хэширования), создающее дайджест (обзор, профиль, хэш-код или что-то вроде "отпечатка пальцев") сообщения или данных. Преобразование называется односторонним, так как обратное преобразование, т. е. получение исходной информации из дайджеста, невозможно. Подробнее об алгоритме шифрования md5 см. [3].

И последним добавьте столбец **role** типа **VARCHAR** с максимальным количеством символов, равным 10 (рис. 6).

The screenshot shows the phpMyAdmin 3.5.1 interface. On the left, there's a sidebar with the phpMyAdmin logo and navigation links. The main area displays the 'Add new column' dialog for a table named 'users'. The dialog has a table with the following columns:

Имя	Тип	Длина/значения	По умолчанию	Сравнение
id	INT		Нет	
name	VARCHAR	20	Нет	
pass	VARCHAR	32	Нет	
role	VARCHAR	10	Нет	

At the bottom of the dialog, there are fields for 'Комментарий к таблице:', 'Тип таблиц:', and 'Сравнение:'.

Рис. 6. Добавление столбцов в будущую таблицу с именем USERS

Теперь займемся настройкой таблицы **users**.

Сначала сделаем так, чтобы столбцу **id** значения присваивались автоматически, для этого прокрутите окно браузера с помощью горизонтального скроллинга, пока не будет виден **A_I (autoincrement)**. В нем необходимо проставить галочку. Далее в выпадающем списке **Индекс** установим **primary** (первичный ключ), назначение которого — указать СУБД, что значения столбца будут уникальными. Это обязательное условие для работы режима **auto_increment**, смысл данного действия сводится к следующему: столбцу невозможно будет присвоить значение, которое уже в нем имеется, а если будет совершена попытка сделать это, то СУБД выдаст ошибку.

Нажмите кнопку **Сохранить**, вы увидите следующее — рис. 7.

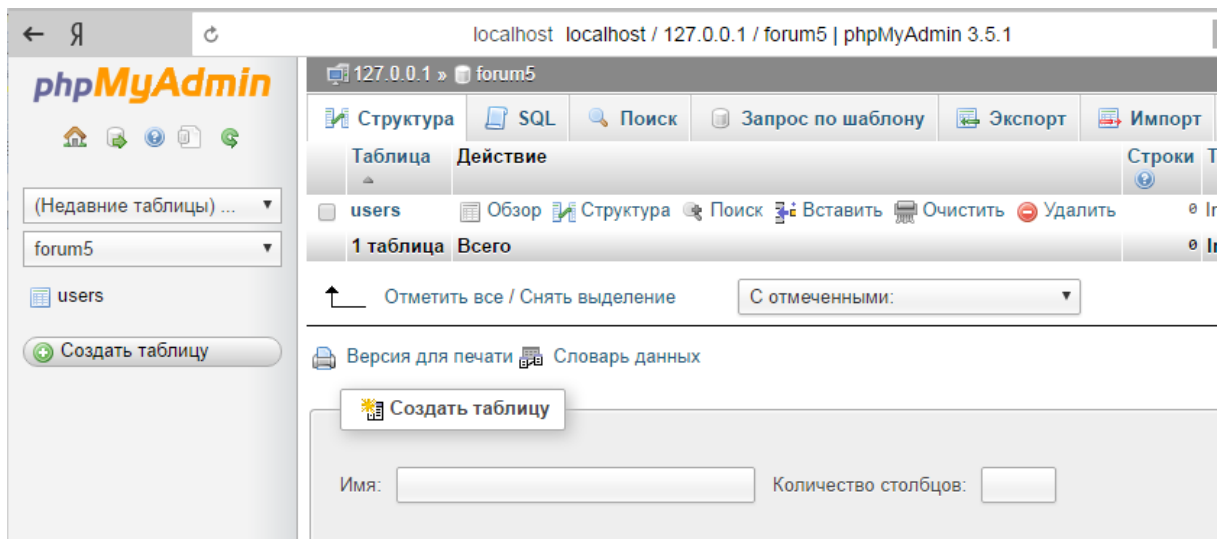


Рис. 7. Результат создания таблицы USERS

Слева вы можете увидеть имя созданной таблицы.

Добавим 2 записи, одна из которых будет характеризовать администратора, а другая —пользователя. Для этого щелкните левой кнопкой мыши на вкладке **Вставить**, расположенной в верхней части окна phpMyAdmin (рис. 8).

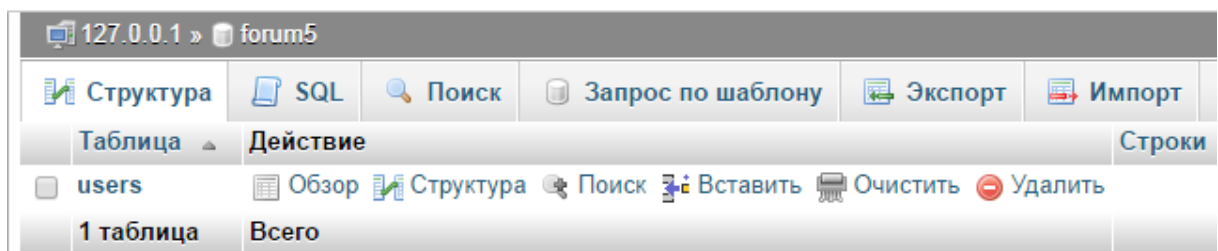


Рис. 8. Набор вкладок, посредством которых можно выполнять различные действия с таблицей

Это приведет к появлению формы, состоящей из двух абсолютно идентичных частей, для вставки новой записи в таблицу, которая является в данный момент активной, т. е. таблицы **users** (рис. 9).

Вставить таблицу

Поле	Тип	Функция	Null	Значение
id	int(11)			
name	varchar(20)			
pass	varchar(32)			
role	varchar(10)			

☒ Игнорировать

Поле	Тип	Функция	Null	Значение
id	int(11)			
name	varchar(20)			
pass	varchar(32)			
role	varchar(10)			

OK

Рис. 9. Форма для вставки новых записей

Введите в верхнюю часть, в секцию **Значение**, следующую информацию:

- id — вводить не надо, т. к. MySQL его поставит автоматически;
- name — administrator;
- pass — 12345;
- role — admin.

Заметим, что пароль не хранится в открытом виде, вместо этого используется его хеш, полученный как результат функции `md5()`. phpMyAdmin предоставляет нам удобный способ сделать это — в разделе **Функция** выберите в выпадающем списке, расположенном напротив слова **pass**, пункт **MD5** (рис. 10). В результате в таблицу `users` в поле **pass** будет записано значение, полученное от выполнения функции `md5()` с параметром `12345`, т. е. хеш от введенного вами значения, в чем вы убедитесь чуть позже.

Вставить таблицу

Поле	Тип	Функция	Null	Значение
id	int(11)			
name	varchar(20)			administrator
pass	varchar(32)	MD5		12345
role	varchar(10)			admin

OK

Рис. 10. Определение хеша пароля средствами phpMyAdmin

Теперь в нижнюю часть формы для добавления записи введите в секцию **Значение** следующую информацию о пользователе:

id — вводить не надо, т. к. MySQL его поставит автоматически;

name — guest;

pass — user (не забудьте в выпадающем списке, расположенном слева от поля, куда вы ввели значение, выбрать пункт MD5);

role — user.

Результат заполнения формы для добавления двух записей изображен на рис. 11.

Вставить таблицу

Поле	Тип	Функция	Null	Значение
id	int(11)			
name	varchar(20)			administrator
pass	varchar(32)	MD5		12345
role	varchar(10)			admin

☐ Игнорировать

Поле	Тип	Функция	Null	Значение
id	int(11)			
name	varchar(20)			guest
pass	varchar(32)	MD5		user
role	varchar(10)			user

Рис. 11. Ввод данных для добавления двух записей в таблицу **USERS**

Таким образом, с помощью phpMyAdmin можно не только добавлять данные, но и сразу обрабатывать их стандартными функциями.

Нажмите кнопку **OK**.

Теперь нажмите левой кнопкой мыши на вкладку **Обзор**, чтобы убедиться в том, что записи были добавлены (рис. 12).

SELECT *
FROM `users`
LIMIT 0 , 30

☐ Профилирование [Быстрая правка](#)

Показать : Начальная строка: Количество строк: Заголовки каждые строк

Сортировать по индексу:

+ Параметры

	id	name	pass	role
<input type="checkbox"/>	1	administrator	827ccb0eea8a706c4c34a16891f84e7b	admin
<input type="checkbox"/>	2	guest	ee11cbb19052e40b07aac0ca060c23ee	user

↑ Отметить все / Снять выделение С отмеченными:

Рис. 12. Просмотр содержимого таблицы USERS

Обратите внимание, что pass= 12345 соответствует зашифрованное 32-значение md5()=827ccb0eea8a706c4c34a16891f84e7b.

Проведем небольшой эксперимент — попробуем добавить нового пользователя, идентификационный номер которого будет равен уже имеющемуся в базе. Нажмите левой кнопкой мыши на вкладку **Вставить** и в верхнюю форму введите следующую информацию:

id — 1;

name — sss;

pass — aaa;

role — user.

Нажмите **ОК**, в результате чего вы увидите сообщение об ошибке (рис. 13).

phpMyAdmin

127.0.0.1 » forum5 » users

(Недавние таблицы) ...
forum5
users
[Создать таблицу](#)

Ошибка

SQL-запрос:

```
INSERT INTO `forum5`.`users` (
  `id`,
  `name`,
  `pass`,
  `role`
)
VALUES (
  '1', 'sss', 'aaa', 'user'
)
```

Ответ MySQL:

#1062 - Duplicate entry '1' for key 'PRIMARY'

Рис. 13. Сообщение об ошибке, возникшее в результате попытки вставить запись с уже существующим значением столбца id

Переходим к созданию таблицы **topic**, нажмите левой кнопкой мыши на ссылку **forum**, расположенную вверху (рис. 13).

Далее в поле **Имя**, расположенном под надписью **Создать таблицу**, введите **topic**, а в поле **Поля** — число **6** и нажмите кнопку **ОК**. После этого заполните информацию о будущих столбцах таблицы **topic**, представленную далее :

- имя — **id**, тип данных **INT**, значения присваиваются автоматически, первичный ключ;
- имя — **kodofrazdel**, тип данных **INT**;
- имя — **name**, тип данных **varchar**, максимально возможное количество символов — **100**;
- имя — **name_creator**, тип данных **varchar**, максимально возможное количество символов — **20**;
- имя — **name_last_answer**, тип данных **varchar**, максимальное возможное количество символов — **20**;
- имя — **date_last_answer**, тип данных **DATE**.

Нажмите кнопку **Сохранить**, после чего добавьте в только что созданную таблицу две записи, которые будут характеризовать разделы (см. табл. 3), и две записи, которые будут характеризовать темы (см. табл. 4). Результат представлен на рис. 14.

+ Параметры

	id	kodofrazdel	name	name_creator	name_last_answer	date_last_answer
<input type="checkbox"/> Изменить <input type="checkbox"/> Копировать <input type="checkbox"/> Удалить	1	0	Для программистов			0000-00-00
<input type="checkbox"/> Изменить <input type="checkbox"/> Копировать <input type="checkbox"/> Удалить	2	0	Работа			0000-00-00
<input type="checkbox"/> Изменить <input type="checkbox"/> Копировать <input type="checkbox"/> Удалить	3	1	Нужна помощь в поиске мануала по MySQL	Zero	MegaMan	2016-02-06
<input type="checkbox"/> Изменить <input type="checkbox"/> Копировать <input type="checkbox"/> Удалить	4	1	Подскажите, как установить PHP	Haos	Lion	2016-12-06

↑ Отметить все / Снять выделение С отмеченными: ☐ Изменить ☐ Удалить ☐ Экспорт

Рис. 14. Записи, добавленные в таблицу TOPIC

Создайте таблицу **message**, информация о ее столбцах:

- имя — **id**, тип данных **int**, значения присваиваются автоматически, первичный ключ;
- имя — **kodoftopic**, тип данных **int**;
- имя — **textmessage**, тип данных **TEXT**;
- имя — **name_man**, тип данных **varchar**, максимально возможное количество символов — **20**;
- имя — **date_answer**, тип данных **DATE**..

После этого добавьте в нее две записи, согласно табл. 6, результат представлен на рис. 15.

+ Параметры

	id	kodoftopic	textmessage	name_man	date_answer
<input type="checkbox"/> Изменить <input type="checkbox"/> Копировать <input type="checkbox"/> Удалить	1	3	Где можно найти мануал по MySQL?	Zero	2016-01-05
<input type="checkbox"/> Изменить <input type="checkbox"/> Копировать <input type="checkbox"/> Удалить	2	3	Ты пробовал смотреть на официальном сайте?	MegaMan	2016-02-01

↑ Отметить все / Снять выделение С отмеченными: ☐ Изменить ☐ Удалить ☐ Экспорт

Рис. 15. Записи, добавленные в таблицу MESSAGE

Денвер хранит все базы, созданные в MySQL, по следующему пути:

Z:\usr\local\mysql-5.5\data или, что тоже самое

C: WebServers\usr\local\mysql-5.5\data

Если вы сделаете копию папки ...data\forum, и переименуете ее в TEST_BASA, то в MySQL у вас будет доступна новая база данных с именем test_basa, содержимое которой полностью аналогично базе данных forum.

Задание

1. Изучите Методические указания или соответствующие разделы пособий по указанной тематике [1-10].
2. Выполните примеры, разобранные в п.п. 1.2, 1.3 методических указаний.
3. По аналогии с указанными примерами разработайте план вашего проекта реализации вашего форума и структуру базы данных.
4. Создайте базу данных для вашего форума в phpMyAdmin и заполните её таблицы.

Содержание отчета

1. Задание к работе.
2. Листинги, скриншоты и пояснения примеров, разобранных в п.п. 1.2, 1.3 методических указаний.
3. План вашего проекта реализации вашего форума и структура базы данных с указанием связей между таблицами.
3. Скриншоты, поясняющие создание базы данных вашего форума в **phpMyAdmin** и заполнение её таблиц.

Лабораторная работа № 8 **Практическое занятие № 8**

Разработка плана кодирования. Составление SQL-запросов. Подключение к базе данных.

Цель работы

Ознакомление с принципами, построения **плана кодирования**.
Формирование навыков составления и использования **SQL-запросов**. Ознакомление с методикой подключения к базе данных.
Создание модуля connect.php.

Методические указания

1. Разработка плана кодирования

Весь форум будет состоять из модулей.

Первый — это **Авторизация** — описывает механизм входа пользователя на форум под своей учетной записью.

Второй — один из основных, это модуль, отвечающий за **вывод** информации на форуме (формирование списка разделов, вывод списка тем для выбранного раздела, а также вывод сообщений для заданной темы).

Третий модуль — **действие**, в нем будет реализована возможность создания новой темы, ответа в тему и т. д.

На рис. 1 представлена схема функционирования трех описанных ранее модулей.

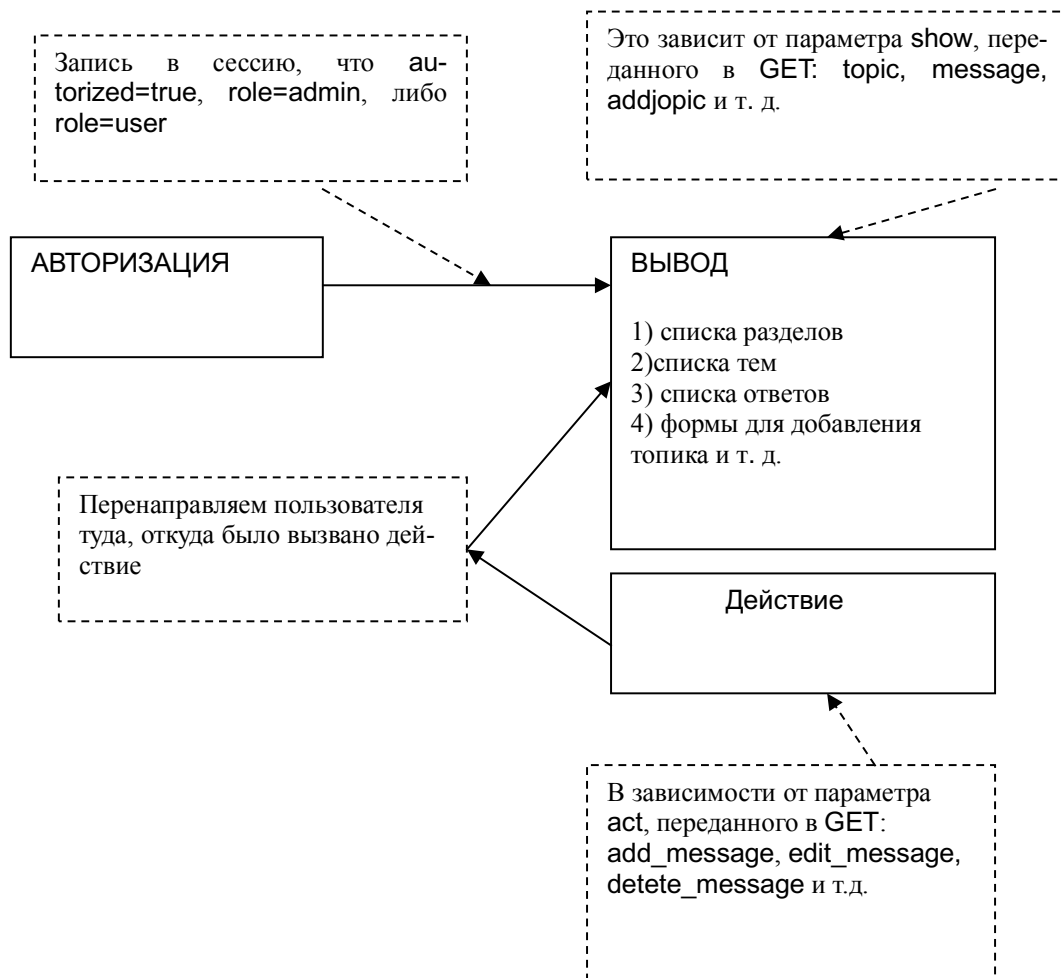


Рис. 1. Схема функционирования форума

2. Составление SQL-запросов

Прежде чем приступить непосредственно к процессу кодирования, рассмотрим методику составления SQL-запросов.

В основе построения запроса лежит задание ключевого слова (команды), которое указывает на то, что мы хотим сделать, а каждая команда состоит из блоков, в которых необходимо указать дополнительные параметры.

Самая часто используемая команда **SQL — select** — предназначена для выборки данных. Общая схема ее построения представлена ниже:

```
SELECT имя_столбца1, имя_столбца2, ..., имя_столбцаN
FROM таблица1, таблица 2, ..., таблицаN
WHERE столбец1=условие1, столбец2=условие2, ..., столбецN= условиеN
ORDER BY столбец1, столбец2, ..., столбецN
```

Команда **select** состоит из следующих блоков:

- **select** указывает серверу, что клиент хочет получить данные;
- слово **from** указывает источник получения данных, им может быть таблица, представление и т. д.;
- слово **where** позволяет указать условия выбора данных;
- слово **order by** указывает атрибуты, по которым будет произведена сортировка.

В **phpMyAdmin** выберите таблицу **message** и нажмите левой кнопкой мыши на вкладку **SQL**, в поле для ввода многострочного текста **Выполнить SQL запрос на БД**, введите следующий запрос:

```
SELECT id, kodoftopic, textmessage, name_man, date_answer FROM `message`
```

Который должен выбрать все записи из таблицы **message** (рис. 2)

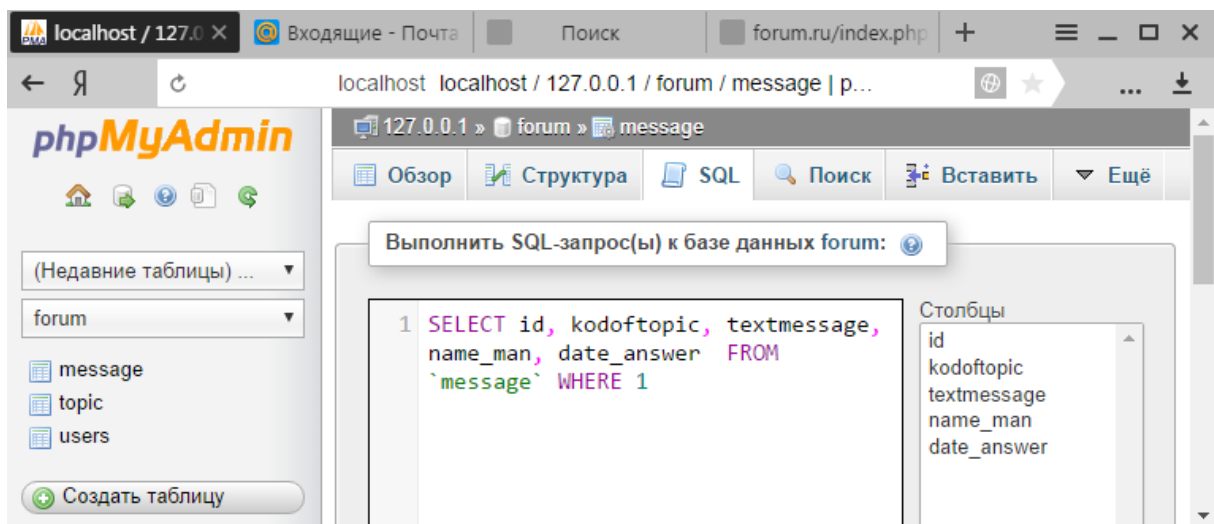


Рис. 2.. Ввод SQL-запроса в phpMyAdmin

Нажмите кнопку **ОК**, вы увидите следующий результат — рис. 3.

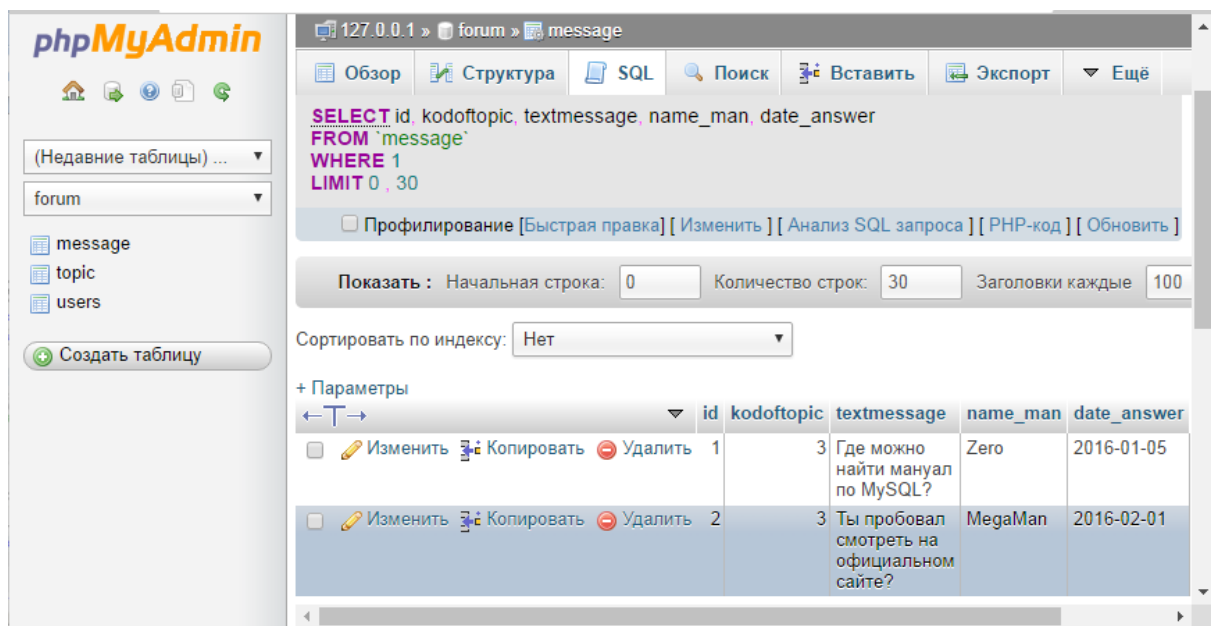
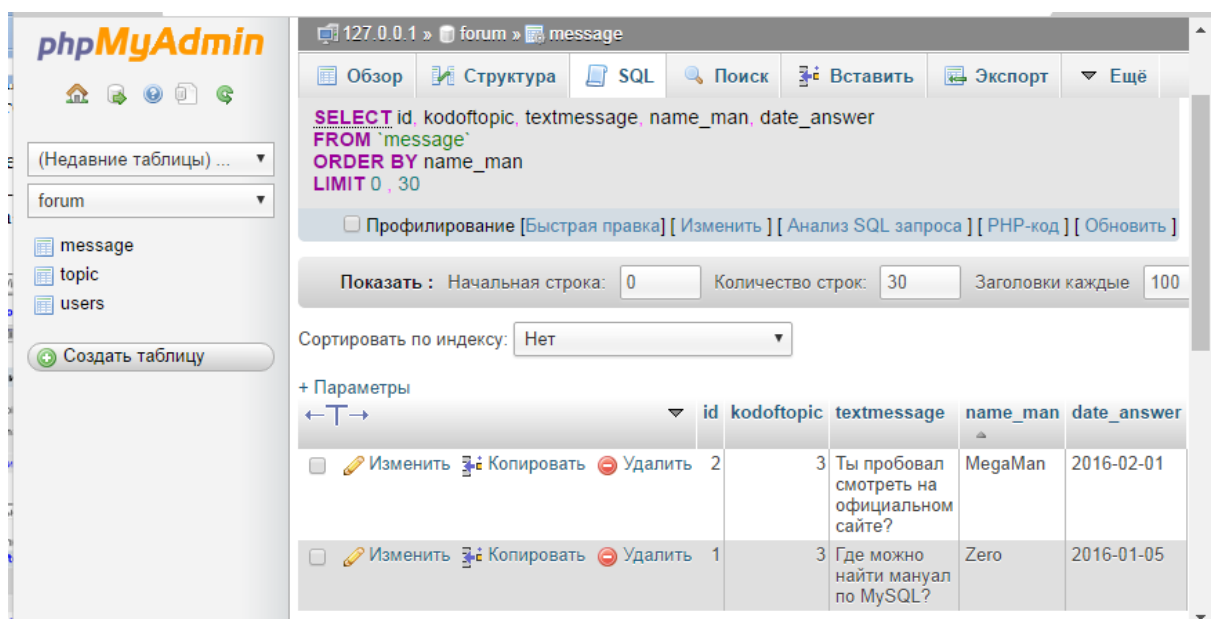


Рис. 3. Результат выполнения запроса

Теперь опять выберем все записи из таблицы **message**, но еще и отсортируем их по столбцу **name_man**, для этого необходимо выполнить следующий запрос (рис. 4.):

```
SELECT id, kodoftopic, textmessage, name_man, date_answer FROM
MESSAGE ORDER BY name_man
```

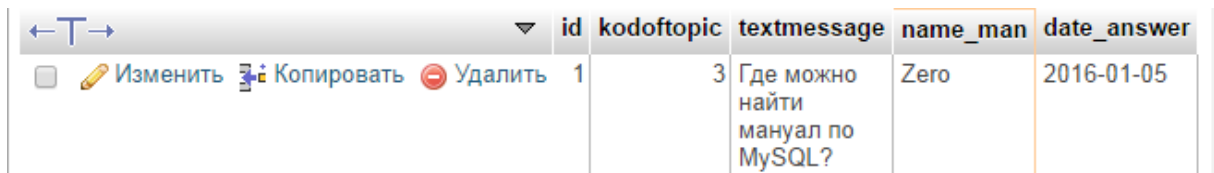
Обратите внимание, теперь на первом месте стоит запись с идентификационным номером, равным 2.

Рис. 4. Результат выполнения запроса с сортировкой по столбцу **name_man**

Для того чтобы выбрать записи, соответствующие определенному условию, используется специальное слово **where**, после которого идет само условие или условия, если их несколько. Например, запрос:

```
SELECT id, kodoftopic, textmessage, name_man, date_answer FROM
MESSAGE WHERE name_man='ZERO'
```

выведет следующий результат — рис. 5.



	id	kodoftopic	textmessage	name_man	date_answer
<input type="checkbox"/> Изменить <input type="button" value="Копировать"/> <input type="button" value="Удалить"/>	1		3 Где можно найти мануал по MySQL?	Zero	2016-01-05

Рис. 5. Результат выполнения запроса с условием

Когда необходимо добавить информацию в таблицу, используется команда **insert**, общая схема построения которой следующая:

```
INSERT INTO имя_таблицы SET
имя_столбца1=значение1,
имя_столбца2=значение2,
имя_столбцаN=значениеN
```

Давайте добавим в таблицу **topic** информацию о новом разделе с именем "**Все о работе с MySQL**", для этого достаточно выполнить следующий запрос:

```
INSERT INTO TOPIC SET name= 'Все о работе с MySQL'
```

Обратите внимание на то, что текст в условии взят в кавычки, это является обязательным действием

А теперь добавим пустую тему для этого раздела, для чего выполните следующий запрос, только не забудьте посмотреть идентификационный номер только что добавленного раздела, (он равняется 5):

```
INSERT INTO TOPIC
SET
Kodofrazdel=5,
name= 'Как добавить новую запись в таблицу?', name_creator = 'Dragon'
```

Когда необходимо изменить уже существующую информацию в таблице, используется команда **update**, общая схема построения которой следующая:

```
UPDATE имя таблицы
SET
имя_столбца1=новое_значение1 , имя_столбца2=новое_значение2,
...
WHERE Условие, при котором меняется значение записей таблицы
```

Давайте изменим имя пользователя **guest** на **ManOfHaos** и его роль с **user** на **admin**, для этого выполните следующий запрос:

```
UPDATE USERS SET name='ManOfHaos', role='admin' WHERE id=2
```

Стоит отметить, что пароль **user**, преобразованный функцией **md5(user)** к виду: **ee11cbb19052e40b07aac0ca060c23ee** при этом будет не затронут и останется старым. А теперь вернем пользователю его старую роль **user**, для этого выполните следующий запрос:

```
UPDATE USERS SET role='user' WHERE id=2
```

Когда необходимо удалить информацию из таблицы, используется команда **delete**, общая схема построения которой представлена ниже:

```
DELETE FROM имя_таблицы
WHERE Условие, по которому будет осуществлено удаление
```

Давайте удалим тему с именем **"Как добавить новую запись в таблицу?"**. Для этого выполните следующий запрос, только не забудьте посмотреть идентификационный номер данной темы, который должен равняться 6:

```
DELETE FROM TOPIC WHERE id=6
```

Если у вас возникло желание более углубленно изучить **SQL**, введите в любом поисковике фразу **"Учебник по SQL"**, и сделайте соответствующий выбор. Можно также посетить сайт www.sql-ex.ru, где вы не только познакомитесь с тонкостями языка **SQL**, но и проверите свои знания с помощью специальных упражнений.

3. Подключение к базе данных

Создайте виртуальный хост **forum.ru**, и в нем папку **www**, далее перезапустите **Денвер** — только после этого новый хост станет доступен. Напомним (см. [9 л.р. №7]), что все файлы, разработанные в данном разделе, необходимо сохранять по адресу:

Z:\home\forum.ru\www или, что то же самое C:\WebServers\home\forum.ru\www

Одно из первых действий, которое нужно совершить в программе, работающей с данной СУБД — это подключение к **MySQL**. Для этого используется следующая функция:

```
mysql_connect(hostname, username, password)
```

где

- **hostname** — имя хоста, чаще всего указывается **localhost**, т. к. обычно **MySQL** и **PHP**-скрипт находятся на одном и том же сервере;
- **username** — имя пользователя, под которым будет осуществлено подключение к серверу;
- **password** — пароль пользователя.

Если произвести подключение не удалось, то функция вернет **false**, поэтому очень часто **mysql_connect()** используется следующим образом:

```
mysql_connect(hostname, username, password)
or die("Не удалось подключиться к базе")
```

connect.php будет модулем подключения к базе данных **FORUM**, именно он будет использоваться во всех скриптах, входящих в данный проект. Главное преимущество в его применении заключается в следующем: чтобы изменить параметры работы с **MySQL**, например, после заочки на реальный хост, достаточно будет изменить пару строчек в **connect.php**, и форум будет успешно функционировать.

После того как подключение к MySQL осуществлено, необходимо выбрать базу данных, с которой будет производиться работа, для этого используется следующая функция:

```
mysql_db(database_name)
```

где `database_name` — имя базы данных.

Если `mysql_db ()` была выполнена успешно, то эта функция вернет `true`, в противном случае — `false`. Для `mysql_db ()` также часто используют защиту в виде `die`.

4. Кодирование

Приступаем к кодированию.

При работе с PHP удобно использовать CodeLobster PHP Edition Pro. CodeLobster PHP Edition - очень мощный и многофункциональный редактор PHP, HTML, CSS, javascript файлов со встроенным дебаггером и HTML инспектором аналогичным FireBug'у. Данная версия редактора является профессиональным выпуском со всеми дополнительными плагинами поддерживающие Drupal, Joomla, Smarty, WordPress, jQuery, CodeIgniter, CakePhp, Facebook, Symfony и MySQL.

Создайте файл `connect.php` (листинг 1).

Листинг 1. Файл `connect.php`

```

1  <?php
2  //хостинг1
3  $sqlhost="localhost";
4  //имя пользователя
5  $sqluser="root";
6  //пароль
7  $sqlpass="";
8  //имя базы данных
9  $db="forum";

11 //Подключаемся к MySQL
12 mysql_connect($sqlhost, $sqluser,$sqlpass)or die("MySQL не
   доступен! ".mysql_error());
13 //подключаемся к базе данных
14 mysql_select_db($db)or die("Нет соединения с базой данных
   ".mysql_error());
15 ?>
```

Все параметры подключения (имя хоста, имя пользователя, его пароль и имя базы данных) сохранены в специальных переменных: `$sqlhost`, `$sqluser`, `$sqlpass`, `$db`. Для Денвера имя хоста будет `localhost`, имя пользователя — `root`, а пароль будет пустым. В строке 12 осуществляется подключение к MySQL.

Запустите Денвер. Наберите в адресной строке браузера

`http://www.forum.ru/connect.php`

На рис.1 представлен результат просмотра браузера

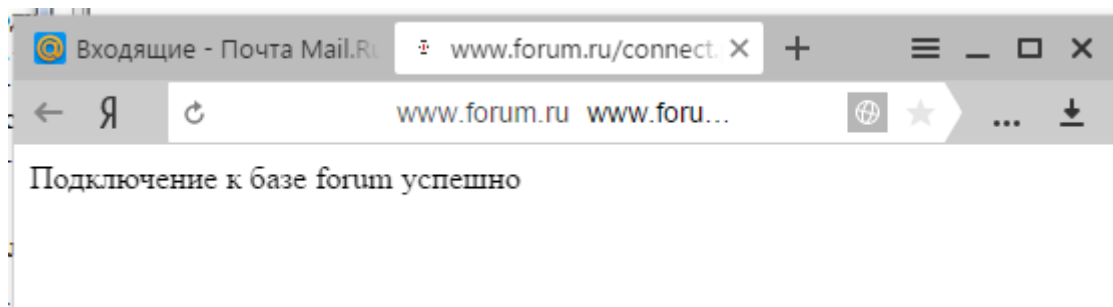


Рис.1 Результат просмотра файла connect.php

В строке 14 выбирается база данных, с которой будет осуществлена работа. В обоих случаях используется защитная конструкция `die`, которой в качестве параметра передается как текст, так и результат выполнения функции `mysql_error()`. Синтаксис `mysql_error()`:

```
mysql_error()
```

Данная функция возвращает информацию об ошибке для последней совершенной операции с MySQL. Например, если строка 12:

```
mysql_select_db($db) or die("MySQL не доступен! ".mysql_error());
```

выполнится с отрицательным результатом (`false`), т. е. подключиться к базе данных не удастся, то вы увидите как текст: MySQL не доступен!, так и сообщение об ошибке.

Измените в файле `connect.php` (строка 9) имя базы данных `$db="forum1"`. Перезагрузите страницу. На рис.2 представлен результат просмотра браузера при неправильном имени базы данных.

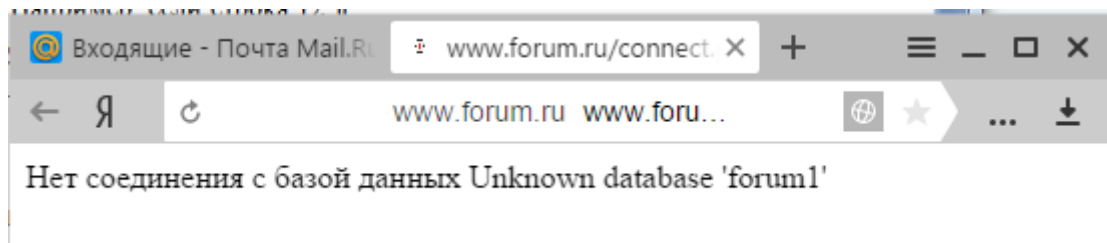


Рис.2 Результат просмотра файла connect.php при неправильном имени БД.

Задание

1. Изучите Методические указания или соответствующие разделы пособий по указанной тематике [2,3,4,5,6,7,10].
2. Выполните примеры, разобранные в п.п. 1-4 методических указаний.
3. По аналогии с указанными примерами разработайте **план кодирования** и составьте **SQL-запросы**, модуль `connect.php` для вашего форума.

Содержание отчета

1. Задание к работе
2. Листинги, скриншоты и пояснения примеров, разобранных в п. п. 1-4 методических указаний.
3. **План кодирования, Схема функционирования форума, SQL-запросы** с соответствующими скриншотами и пояснениями к ним; Модуль `connect.php` подключения к базе данных с соответствующими скриншотами и пояснениями к ним для **вашего форума**.

Лабораторная работа № 9 Практическое занятие № 9

Авторизация, общение на форуме и выход

Цель работы

Ознакомление с методикой разработки модуля авторизации `login.php` и модуля выхода `logout.php` из форума

Методические указания

1. Модуль авторизации

Логика **модуля авторизации** состоит в следующем.

- Пользователю выводится форма авторизации.
 - Далее введенные им данные обрабатываются.
 - Если авторизация успешна, то в сессию пользователя записывается информация об этом факте, а также о логине и роле пользователя.
 - В противном случае авторизация считается не пройденной.
- Создайте файл `login.php` (листинг 1)

Листинг 1. Файл **login.php**

```

1  <?php
2  //Данный модуль возвращает в $_SESSION['authorized'] значение
   TRUE,
3  //если авторизация пройдена
4
   5//Начинаем сессию
6 session_start();
```

```

7    //Проверяем, как запущен скрипт - обработчик? или как форма для
авторизации?
8    if(!isset($_POST['enter']))
9    {
10   //Выводим форму авторизации
11   ?>
12   <form method='post' action="">
13   Авторизация на форуме<BR>
14   Имя:<input type='text' name='name' value=""><BR>
15   Пароль:<input type='password' name='pass'><BR>
16   <input name='enter' type='submit' value='Войти'>
17   <?php
18   }
19   //Если как обработчик, то пытаемся авторизовать пользователя
20   else
21   {
22   //Проверяем, ввел ли пользователь имя и пароль
23   if ($_POST['name']!=" and $_POST[ 'pass']!=")
24   {
25   //Защита от взлома
26   $safe_name=mysql_escape_string($_POST['name']);
27   $safe_pass=mysql_escape_string($_POST['pass']);
28   //Преобразуем пароль в хеш
29   $safe_pass=md5($safe_pass);
30   //Подключаемся к MySQL и базе данных
31   require_once('connect.php');
32   //Формируем запрос
33   $sql="SELECT name,pass,role FROM USERS WHERE name='
".$safe_name." ' and pass=' ".$safe_pass." ' ";
34   //Получаем результат запроса в переменную $result
35   $result=mysql_query($sql);
36   //Проверяем, есть ли такой пользователь
37   if (!mysql_num_rows($result))
38   //Если такого пользователя нет, то отказываем ему в доступе
39   die("Неверный логин или пароль <a href='index.php'>Назад!</a>");
40   //Иначе записываем факт авторизации в сессию
41   else
42   {
43   //Получаем результат запроса
44   $line=mysql_fetch_row($result);
45   //Записываем факт авторизации в сессию
46   $_SESSION ['authorized'] =true;
47   //Сохраняем имя пользователя
48   $_SESSION ['name'] =$_POST ['name'] ;
49   //Сохраняем роль пользователя
50   $_SESSION['role']=$line[2];
51   //Выводим пользователю информацию, что он был авторизован
52   echo "Авторизация прошла успешно!

```

```

    <a href=index.php>Вернуться в форум</a>";
53  }
54
55  //Если пользователь не ввел данные
56  else
57  {
58  //Отказываем ему в доступе
59  die("Неправильный логин или пароль <a
href='index.php'>Назад!</a>");
60  }
61  }
62  ?>

```

В строке 6 происходит запуск сессии, т. к. информация об авторизации будет записываться в нее. В строке 8 проверяется условие

```
(isset($_post['enter'])),
```

т. к. сам скрипт работает в двух режимах:

Он просто выводит форму авторизации, когда условие

```
(isset($_post['enter'])) верно.
```

Осуществляет авторизацию пользователя, если он ввел данные в форму и нажал кнопку **Войти**.

В строке 23 происходит проверка, ввел ли пользователь логин и пароль с использованием логического оператора **and**, позволяющего строить логическое выражение, обе части которого должны быть равны **true** — именно в этом случае условие считается верным.

В строках 26 и 27 данные, введенные пользователем, обрабатываются с помощью защитной функции `mysql_escape_string()`, которая позволяет обезопасить программу от взлома, ее синтаксис представлен ниже:

```
mysql_escape_string(строка);
```

Эта функция экранирует специальные символы (добавляет перед ними слэши), используемые в SQL-запросе, т. е. заменяет их на альтернативные — безопасные варианты (табл. 1).

Таблица 1. Список опасных символов для SQL-запроса

Специальный символ	После обработки функцией <code>mysql_escape_string</code>
<code>\n</code>	<code>\\n</code>
<code>\r</code>	<code>\\r</code>
<code>\</code>	<code>\\</code>
<code>'</code>	<code>'\'</code>
<code>"</code>	<code>\"</code>

Такая замена необходима для предотвращения атак типа **SQL injection** (дословно "SQL-инъекция"), которые заключаются в том, что злоумышленник пытается изменить SQL-запрос, используемый в PHP-скрипте, таким образом, чтобы получить доступ к базе данных и возможность совершить над ней нежелательные действия.

Рассмотрим небольшой пример. Запустите phpMyAdmin и выполните следующий запрос:

```
SELECT name,pass,role FROM USERS WHERE name = " OR '1' = '1' AND
pass = " OR '1' = '1'
```

Результат вы можете видеть на рис. 1.

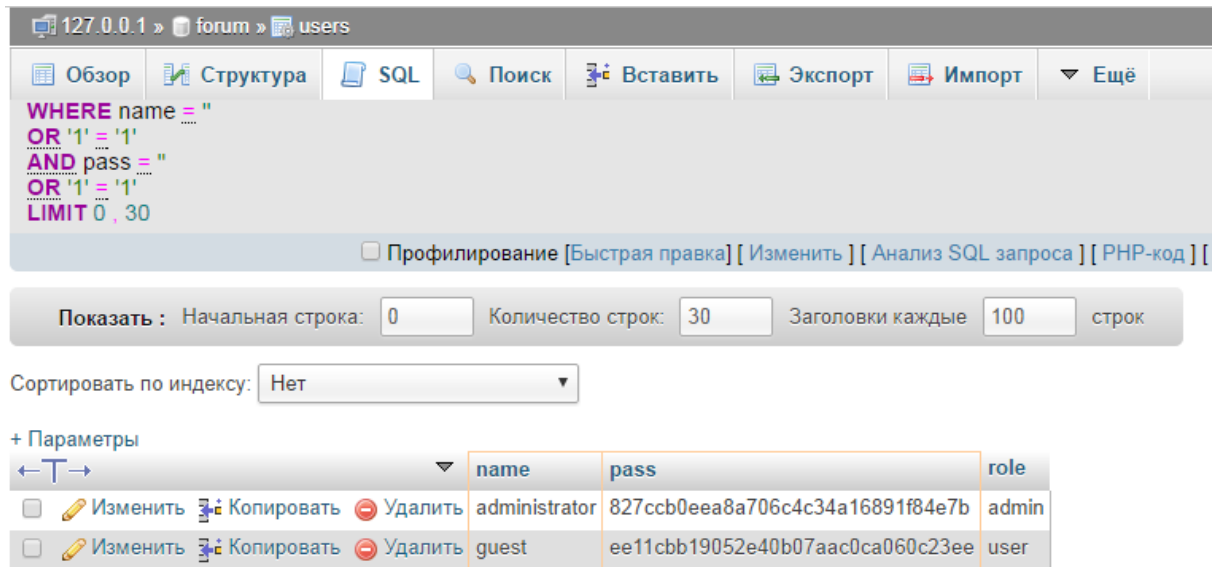


Рис. 1. Пример SQL injection

Таким образом, не зная ни логина, ни пароля, нам стало доступно все содержимое таблицы users, поэтому старайтесь всегда использовать функцию

`mysql_escape_string()`, иначе злоумышленнику не составит труда подвергнуть вас атаки типа SQL injection.

Возвращаясь к `login.php`, в строке 29 пароль, введенный пользователем, преобразуется в хеш. Затем, в строке 31, подключается модуль соединения с базой данных `connect.php`. В строке 33 формируется SQL-запрос:

```
"SELECT name,pass,role FROM USERS WHERE name=' ".$safe_name." '
and pass=' ".$safe_pass." ' "
```

который выберет имя пользователя и его роль из таблицы users для записи, у которой совпадает пароль и логин, введенные пользователем. В запросе используются два условия, объединенных логическим оператором `and`, это означает, что они оба должны быть верными.

В строке 35 происходит выполнение запроса с помощью функции `mysql_query()`, синтаксис которой представлен ниже:

```
mysql_query(SQL-запрос)
```

Данная функция отправляет MySQL-запрос, переданный в качестве параметра. Если это запрос типа `select` и его выполнение успешно, то функция возвращает указатель на результат, в противном случае — `false`. Когда работа осуществляется с запросами типа `insert`, `update` или `delete`, то функция в случае успеха возвращает `true`, в случае неудачи — `false`.

Важно отметить, что если запрос составлен неправильно, то выполнение `mysql_query()` не вызовет ошибку, поэтому данная функция часто используется вместе с защитной конструкцией `die`. В противном случае, когда вы не получите желаемого результата, понять причину этого будет сложно, т. к. никаких поясняющих сообщений в браузере вы не увидите. Рассмотрим пример, в котором используется некорректный SQL-запрос, т. к. в нем указано имя несуществующего столбца (`prog15.php`):

```
prog15.php
<?php
require_once("connect.php");
$sql="SELECT dragon FROM MESSAGE";
mysql_query($sql);
echo "Все ОК";
?>
```

Запустите Денвер. Наберите в адресной строке браузера

`http://www.forum.ru/prog15.php`

На рис.2 представлен результат просмотра браузера

Результат работы скрипта, полученный с использованием Codelobster PHP Edition Pro v5.3, изображен на рис. 2.

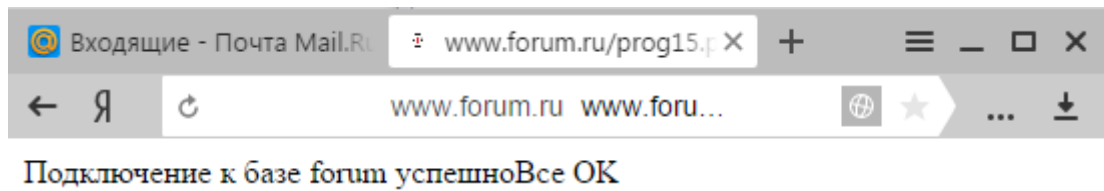


Рис. 2. Выполнение неправильно составленного SQL-запроса без использования защитной конструкции `die`.

Как видите, сообщение об ошибке не возникло. А теперь выполните следующий скрипт (`prog16.php`):

```
prog16.php
<?php
require_once("connect.php");
$sql="SELECT dragon FROM MESSAGE";
mysql_query($sql) or die (" Возникла ошибка: ".mysql_error());
echo "Все ОК";
?>
```

Его результат работы представлен на рис. 3.

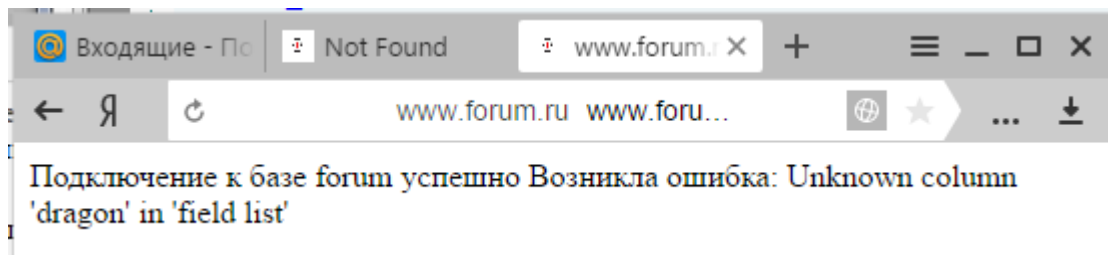


Рис. 3. Выполнение неправильно составленного SQL-запроса с использованием защитной конструкции **die**.

В данном случае мы можем видеть сообщение о некорректном запросе, что позволит более быстро исправить возникшую в программе ошибку.

Чтобы в дальнейшем использовать результат выполнения SQL-запроса, необходимо сохранить указатель на результат, возвращенный функцией `mysql_query()`, как правило, это делается следующие образом:

```
Имя_переменной=mysql_query(SQL-запрос);
```

После этого уже переменная будет ссылаться на возвращенный результат. Для того чтобы получить его, надо воспользоваться одной из специальных функций, наиболее часто используемой из которых является `mysql_fetch_row()`, ее синтаксис:

```
mysql_fetch_row(ссылка на результат SQL-запроса);
```

Функция возвращает одну запись из результата SQL-запроса в виде массива, ключи которого пронумерованы по порядку, начиная с нуля. Например, в результате выполнения приведенной далее программы (`prog17.php`):

```
prog17.php
<?php
<?php
require_once("connect.php") ;
$sql="SELECT id, name FROM TOPIC";
$data=mysql_query($sql);
$line=mysql_fetch_row($data);
echo "<br>Идентификационный номер: ".$line[0];
echo "<br>Название: ".$line[1];
?>
```

вы увидите следующее — рис. 4.

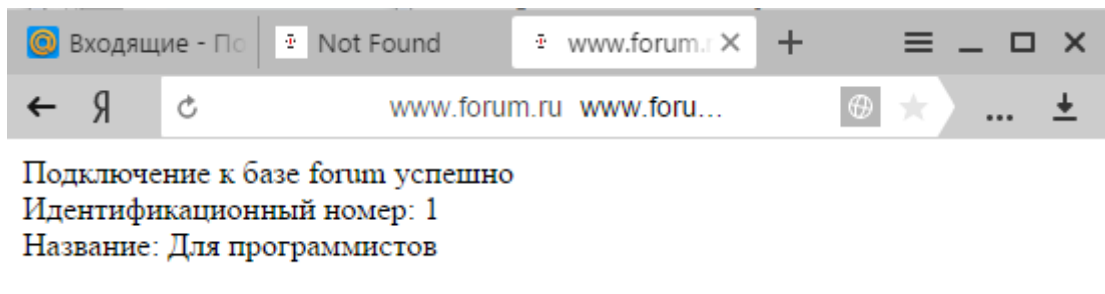


Рис. 4. Хотя запрос возвращает 4 записи, однократное обращение к функции `mysql_fetch_row()` позволяет отобразить только 1 запись

Как правило, для получения результатов SQL-запроса, когда их количество больше 1 используют цикл `while` по следующей схеме:

```
while ($line=mysql_fetch_row (ссылка на результат SQL-запроса))
{
    обработка записей, полученных в результате выполнения запроса
}
```

Смысл условия цикла заключается в следующем: пока функция `mysql_fetch_row()` возвращает записи, условие верно, как только все записи возвращены, функция вернет `false`, соответственно цикл прекратит работу. Теперь измените программу `prog17.php`, введя в нее рассмотренный цикл (см. код далее), и сохраните ее с именем `prog19.php`:

```
prog19.php
<?php
require_once("connect.php");
$sql="SELECT id,name FROM TOPIC";
$data=mysql_query($sql);
while ($line=mysql_fetch_row($data))
{
    echo "<br>Идентификационный номер: ".$line[0];
    echo "<br>Название: ".$line[1];
}
?>
```

Результат работы программы `prog19.php` приведен на рис. 5.

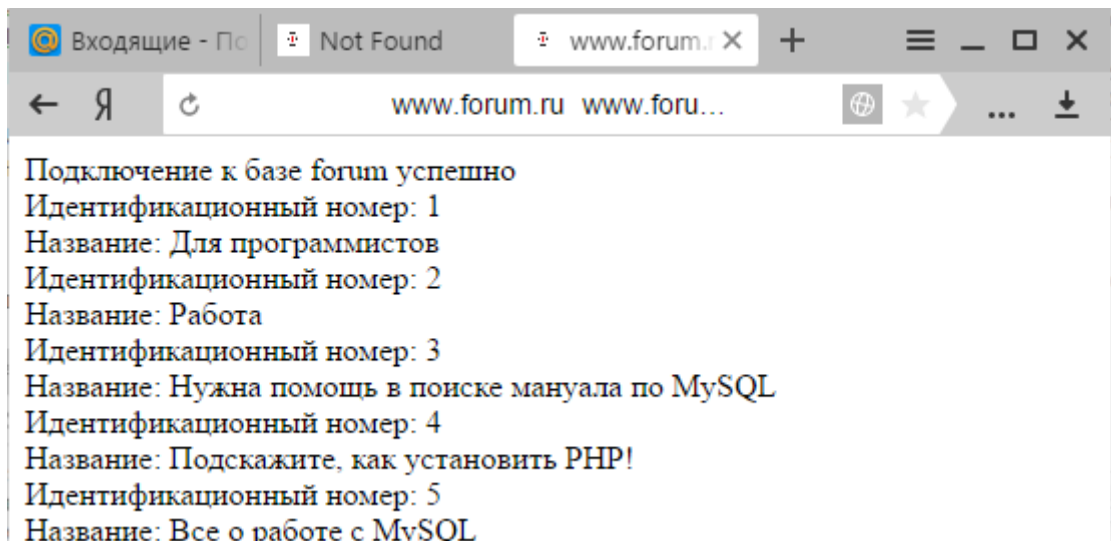


Рис. 5. Результат работы программы prog19.php

Рассмотрим еще одну полезную функцию `mysql_num_rows()`, синтаксис:

`mysql_num_rows(ссылка на результат SQL-запроса)`

Эта функция возвращает количество записей, которое получилось в результате выполнения запроса.

Например, следующий скрипт (prog20.php):

prog20.php

```
<?php
require_once("connect.php") ;
$sql="SELECT id,name FROM TOPIC";
$data=mysql_query($sql);
echo "В результате выполнения Запроса<BR>";
echo "получилось ".mysql_num_rows($data) . " записей";
?>
```

выведет на экран текст **В результате выполнения запроса получилось 5 записей** (рис. 6).

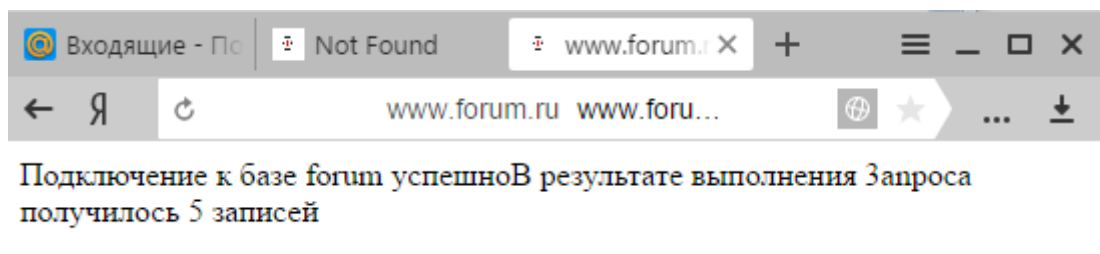


Рис. 6. Результат работы программы prog20.php

Запустите Денвер. Наберите в адресной строке браузера

`http://www.forum.ru/login.php`

На рис 7 представлен результат просмотра браузера

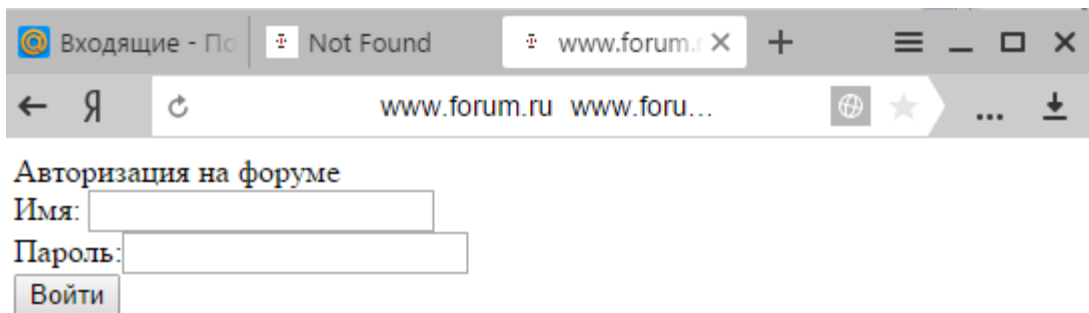


Рис. 7. Результат работы модуля login.php.

В строке 37 листинга login.php с помощью функции `mysql_num_rows()` осуществляется проверка, вернул ли что-то SQL-запрос, выполненный в строке 35, если нет, то, значит пользователь не зарегистрирован в системе и ему выводится сообщение "Неправильный логин или пароль". Нажмите Войти (см. рис. 8).

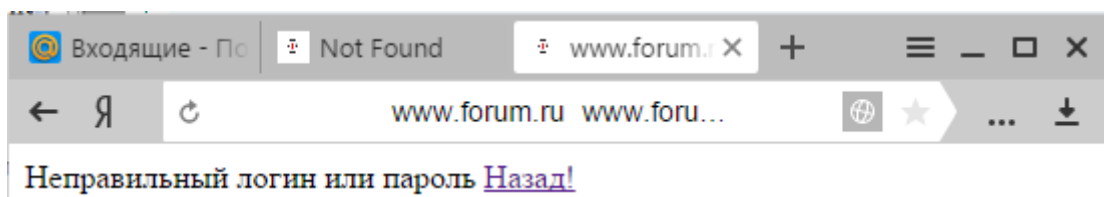


Рис. 8. Результат работы модуля login.php при неправильном вводе логина или пароля.

Если запрос вернул результат, то значит это зарегистрированный пользователь — его имя и роль записываются в сессию. Также в сессию записывается элемент `authorized` со значением `true`.

Введите: Имя — administrator; Пароль — 12345. Результат на рис. 9.

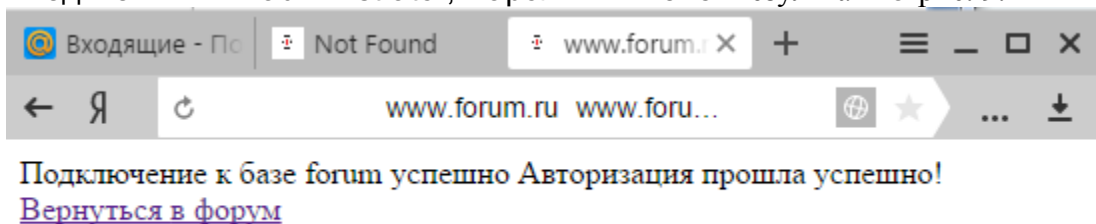


Рис. 9. Результат работы модуля login.php при правильном вводе логина или пароля.

Проверьте правильность работы модуля login.php при: Введите: Имя — guest; Пароль — user.

Сессия — это механизм, позволяющий серверу идентифицировать пользователя посредством уникального номера, который назначается сеансу работы пользователя с сервером.

Для того чтобы инициализировать сессию, необходимо использовать функцию `session_start()`, после чего посетителю будет присвоен уникальный номер — идентификатор, который сохранится как на сервере, так и на компьютере пользователя.

При следующем обращении пользователя к тому же серверу незаметно для него вместе с его запросом будет отправлен и идентификатор сессии. Сервер, получив его, сверяет с теми, которые у него имеются, и соответственно может определить, обращался этот пользователь к серверу раньше или нет.

Самое важное свойство сессий заключается в том, что они могут хранить данные на сервере, время жизни которых равно времени жизни сессии. Для этого используется специальный массив `$_session`, если необходимо что-то сохранить, достаточно объявить элемент этого массива с нужным именем и присвоить ему сохраняемое значение. При следующем обращении пользователя к серверу ваш скрипт просто может обратиться к этому элементу массива `$_session`.

Как отмечалось ранее, идея авторизации с использованием сессии состоит в том, чтобы после авторизации пользователь создает элемент `$_session['authorized']` и присваивает ему значение `true` (см. строку 46). А дальше в каждом скрипте, который входит в админку, просто проверяется наличие `$_session['authorized']`. Если данный элемент массива создан и равен `true`, то значит это авторизованный администратор, в противном случае это злоумышленник, который хочет обойти авторизацию. Более подробно о механизме см. [2 с. 168-164].

2. Модуль `logout.php`

После того как пользователь авторизовался и пообщался на форуме, он может выйти. За этот процесс будет отвечать модуль `logout.php` (листинг 2).

Листинг 2.3. Файл `logout.php`

```
<?php
//Уничтожаем сессию
session_start();
session_unset();
session_destroy();
//Перенаправляем пользователя на index.php
header('location:index.php');
?>
```

Как это ни парадоксально, но перед тем как начать удалять сессию, ее сначала нужно начать, иначе удалять будет нечего, потому что сведения о сессии будут потеряны.

Задание

1. Изучите Методические указания или соответствующие разделы пособий по указанной тематике [2,3,4,5,6,7,10].
2. Выполните примеры, разобранные в п.п. 1, 2 методических указаний.
3. По аналогии с указанными примерами разработайте модуль авторизации **`login.php`** и модуль выхода **`logout.php`** для **вашего форума**.

Содержание отчета

1. Задание к работе
2. Листинги, скриншоты и пояснения примеров, разобранных в п. п. 1,2 методических указаний.
3. Модуль авторизации **login.php** и модуль выхода **logout.php**, а также рабочие программы **prog15.php** - **prog20.php** для **вашего форума** с соответствующими скриншотами и пояснениями к ним.

Лабораторная работа № 10 Практическое занятие № 10

Разработка основного модуля Форума **index.php**

Цель работы

Ознакомление с методикой разработки основного модуля Форума **index.php**

Методические указания

1. Основной модуль Форума **index.php**

Центральным файлом форума является **index.php** (листинг 1). В нем будет объединена единой логикой работа всех модулей.

Листинг 1. Файл **index.php**

```
<?php
//Запускаем сессию
session_start();
//Подключаемся к MySQL и базе данных FORUM
require_once ('connect.php');

//Если пользователь не авторизован,
//то выводим ссылку на login. php
if(!isset($_SESSION['authorized']))
{
?>
<p align='right'>
<a href='login.php'>Авторизация</a><br>
</p>
<?php
```

```

16  $_SESSION['name']='guest';
17  $_SESSION['role'] = 'user';
    }
    else
    {
        //Если пользователь авторизован,то сообщаем ему об этом
        //и выводим ссылку на logout.php
        echo "<p align='right'>Вы авторизованы под ником:
        ".$_SESSION['name']. "<BR>";
        echo "<a href='logout.php'>Выход</a></p>";
        //Если выполняется действие,
28  if(isset($_GET['act']))
    {
        //то подключаем модуль, отвечающий за это,
        //и совершаем действие
        require_once ('action.php');
    }
    //Иначе осуществляем простой вывод информации
    else
        require_once ('show.php');
    }
    ?>

```

Работа скрипта начинается с проверки, авторизован ли пользователь. Если нет, то выводится ссылка **Авторизация**, нажатие на которую приведет к запуску `login.php`, пользователь в этом случае будет иметь имя `guest` и роль `user` (см. строки 16 и 17). Если же пользователь авторизован, то выводится текст **Вы авторизованы под ником** и **ник** пользователя, а также ссылка **Выход**, нажатие на которую приведет к запуску `logout.php`.

Наберите `http://www.forum.ru/`. Результат - на рис.1. Здесь наличие предупреждающих сообщений связано с отсутствием служебных модулей, которые будут рассмотрены в следующих лабораторных работах.

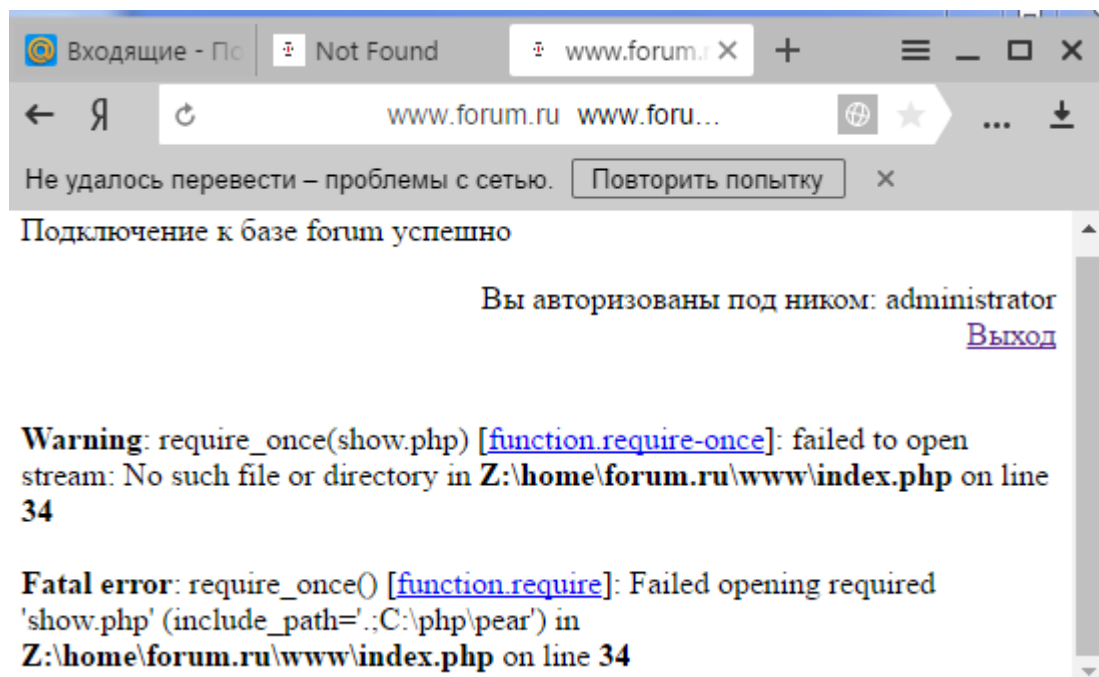


Рис. 1. Результат работы модуля index.php

Далее в строке 28 осуществляется проверка, какой из модулей нужно подключать.

Если пользователь совершает какое-то действие, например, создает тему, то в index.php методом `get` будет передан параметр `act`, и осуществится подключение `action.php`, в противном случае, будет подключен модуль `show.php`.

Задание

1. Изучите Методические указания или соответствующие разделы пособий по указанной тематике [2,3,4,5,6,7,10].
2. Выполните примеры, разобранные в п. 1 методических указаний.
3. По аналогии с указанными примерами разработайте Основной модуль для **вашего форума index.php**.

Содержание отчета

1. Задание к работе
2. Листинги, скриншоты и пояснения примеров, разобранных в п.1 методических указаний.
3. Основной модуль **index.php** для **вашего форума** с соответствующими скриншотами и пояснениями к ним.

Лабораторная работа № 11 Практическое занятие № 11

Организация вывода информации в форуме

Цель работы

Ознакомление с методикой разработки модуля вывода Информации show.php форума.

Методические указания

1. Модуль вывода Информации show.php

Создайте файл show.php, его содержимое представлено в листинге 1.

Листинг 1. Файл show.php

```

<?php
//Запуск данного скрипта без параметра show, переданного
//в GET-строке, приведет к выводу разделов
if (!isset($_GET['show']))
{
//Задаем SQL-запрос
$sql="SELECT id,name FROM TOPIC WHERE kodofrazdel=0";
//Выполняем его
$data=mysql_query($sql) ;
//Надпись "Список разделов"
echo "<big><b>Список разделов</b></big><BR><BR>";
//Выводим список всех разделов
while($line=mysql_fetch_row($data))
{
?>
<table BORDER=1 cellpadding=20 width=100%>
<tr>
<td>
<?php
//Ссылка на index.php только с параметром show-topic
echo '<a href="?show=topic&numrazdel=
21  '.$line[0]."'>'.$line[1]."'</a>';
?>
</td>
</tr>
</table>

```

```

<?php
}
//Больше ничего выполнять не стоит
exit;
}
// end - if (!isset($_GET['show']))

//Если задан параметр show, то в зависимости от него
//выводим соответствующую информацию
switch ($_GET ['show'] )
{
//Если нужно вывести темы для выбранного раздела

38 case 'topic' :
   require_once ('SHOW_MODULE/show_topic.php');
40 break;

//Если нужно вывести сообщения для выбранной темы
case 'message':
   require_once ('SHOW_MODULE/show_message.php');
   break;

//Если нужно вывести форму создания темы
case 'add_topic' :
   require_once ('SHOW_MODULE/show_addtopic.php');
   break;

//Если нужно вывести форму редактирования темы
case 'edit_topic':
   require_once('SHOW_MODULE/show_edittopic.php');
   break;
//Если нужно удалить тему
case 'del_topic':
   require_once('SHOW_MODULE/show_deltopic.php');
   break;
//Если нужно вывести форму редактирования сообщения
case 'edit_message':
   require_once('SHOW_MODULE/show_editmessage.php');
   break;
//Если нужно удалить сообщение
case 'del_message':
   require_once('SHOW_MODULE/show_delmessage.php');
   break;
}
//end - case
?>

```

Как видите, разбив всю задачу на небольшие модули, мы добились того, что код каждого из них стал очень компактным, простым и понятным.

Вся работа данного модуля строится в зависимости от параметра `show`, переданного методом `get`. Если его нет, то значит будет выводиться просто список разделов, для чего используется следующий SQL-запрос:

```
SELECT id, name FROM TOPIC WHERE kodofrazdel=0
```

который можно описать как:

выбрать все идентификационные номера и имена для записей, у которых столбец `kodofrazdel` имеет значение 0, из таблицы `topic`. То есть, другими словами, выбрать все разделы.

Затем каждый раздел, возвращенный SQL-запросом, выводится как ссылка (см. строку 21 кода) следующего содержания:

`?show=topic&numrazdel=идент. номер раздела.`

Таким образом, нажатие на эту ссылку приведет к вызову скрипта `show.php` с параметром `show`, равным `topic`, и параметром `numrazdel`, равным идентификационному номеру выбранного раздела, в этом случае будет выполнен участок кода, заключенный между строками 38—40, который принадлежит конструкции `switch ()` (или, другими словами, будет вызван еще не описанный нами мини-модуль `show_topic.php`).

Запустите Денвер. Наберите в адресной строке браузера

`http://www.forum.ru/`

На рис.1 представлен результат просмотра браузера

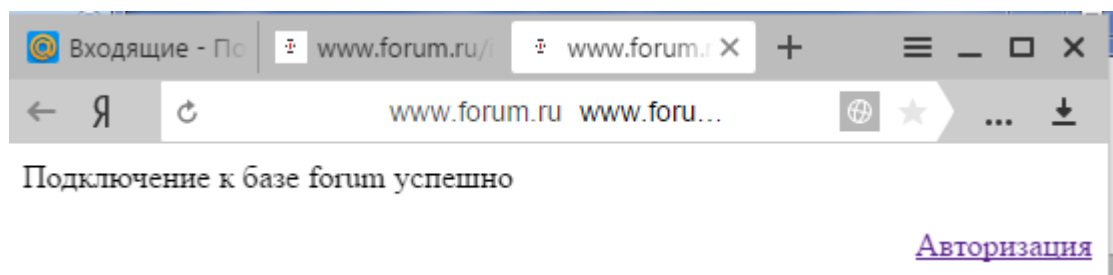


Рис. 1. Результат выполнения запроса **`http://www.forum.ru/`**.

Выполните авторизацию, введя: Имя — `administrator`; Пароль — `12345`.

Результат- на рис. 2.

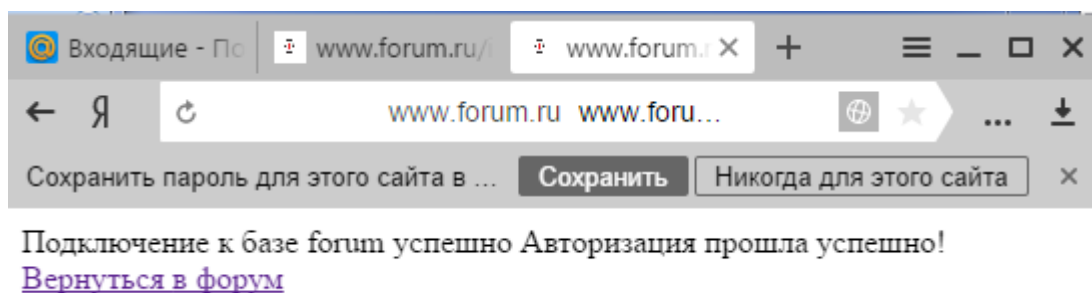


Рис. 2. Результат выполнения Авторизации

Вернитесь в форум. Результат- на рис. 3.

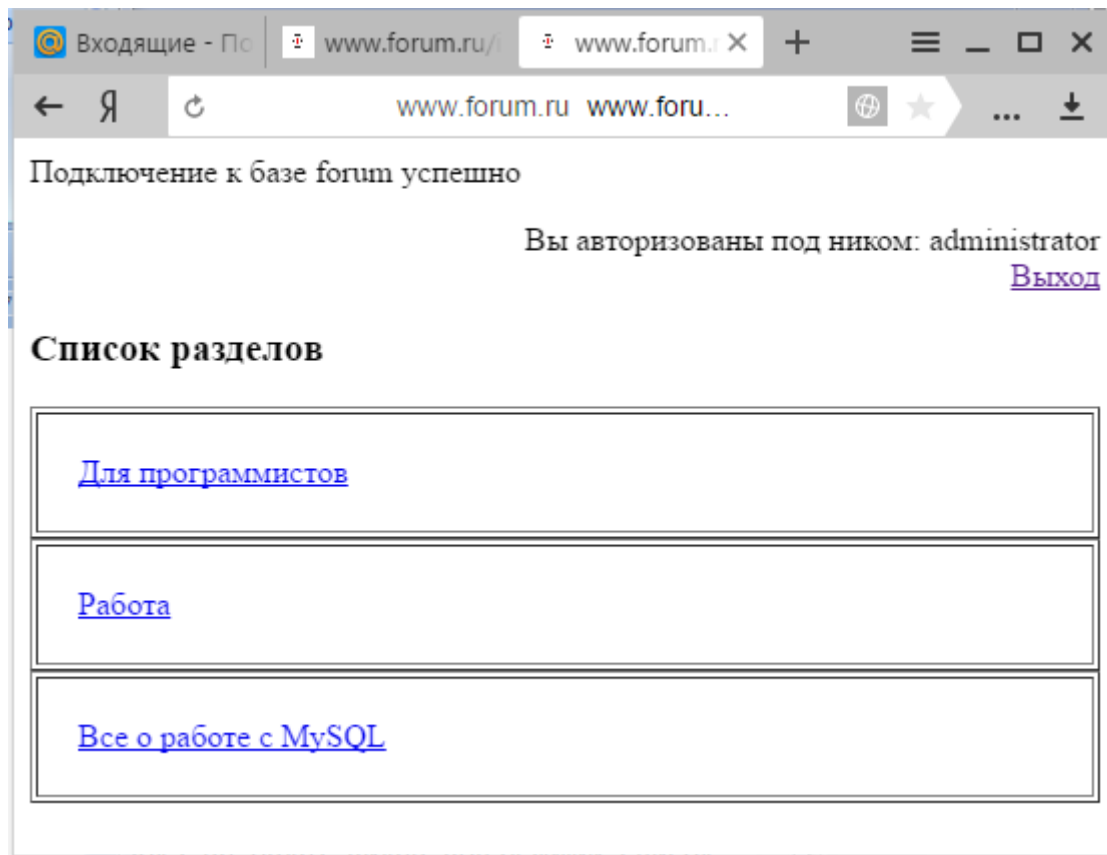


Рис. 2. Результат выполнения Авторизации administrator

Аналогичные операции проделайте при Авторизации: Имя — guest; Пароль — user.

Создайте папку **SHOW_MODULE** и в ней файл **show_topic.php** (листинг 2).

Листинг 2. Файл SHOW_MODULE/show_topic.php

```

...J
<?php
//Задаем SQL-запрос
3  $sql="SELECT id, kodofrazdel, name, name_creator, name_last_answer,
    date_last_answer FROM TOPIC WHERE kodofrazdel=".$_GET
    ['numrazdel']."ORDER BY date_last_answer";
    //Выполняем его
5  $data=mysql_query ($sql);
    //Задаем SQL-запрос, который вернет имя выбранного
    //пользователем раздела
8  $sql2="SELECT name FROM TOPIC WHERE id=" .$_GET['numrazdel'];
    //Выполняем его
10 $data2=mysql_query($sql2);
    //Получаем результат - одна запись
12 $line2=mysql_fetch_row($data2);

```

```

//Выводим надпись
15 echo "<BIG><B>Список тем для ";
16 echo "раздела: ".$line2 [0] . "</B></BIG><BR><BR>";
//Кнопка для создания новой темы
?>
<p align='right'>
<form action="?show=add_topic&numrazdel=<?php echo
$_GET ['numrazdel'] ; ?>" method="post">
22 <input type="submit" value="Создать новую тему">
</form>
<?php
//Выводим заголовок для таблицы
?>
<table BORDER=1 cellpadding=3 width=100%>
<tr>
<td width=60%>
Название темы
</td>
<td width=10%>
<font size=2>Автор</font>
</td>
<td width=30%>
<font size=2>Последнее сообщение (Кто|Дата)</font>
</td>
</tr>
</table>
<?php
//Выводим список всех тем для выбранного раздела
while($line=mysql_fetch_row($data))
{
?>
<table BORDER=1 cellpadding=20 width=100%>
<tr>
<td width=60%>
<?php
//Это в виде ссылки, она на index.php
//только с параметром message
echo '<a href="?show=message&numtopic='.$line[0].'">'
.$line[2]. '</a>';
//Если это админ, то он может редактировать название темы
//и удалять ее
56 if ($_SESSION['role']=='admin')
{
?>
<form action="?show=edit_topic&numtopic=<?echo $line[0]?>"
method="post">
<input type="submit" value="Изменить название">
</form>

```

```

<form action="?show=del_topic&numtopic=<? echo $line[0]?>"
  method="post">
<input type="submit" value="Удалить тему">
</form>
<?php
}
//end - if
?>
</td>
<td width=10%>
<?php
//Имя создавшего тему
echo $line[3];
?>
</td>
<td width=10%>
<?php
//Имя последнего ответившего
echo $line[4];
?>
</td>
<td width=20%>
<?php
//Дата последнего ответа
echo $line[5];
?>
</td>
</tr>
</table>
<?php
}
//end - while
?>

```

В строке 3 задается SQL-запрос на выбор всех тем для выбранного раздела, его идентификатор доступен через `$_get['numrazdel']`. Сам запрос представлен далее:

```

SELECT id, kodofrazdel, name, name_creator,
name_last_answer, date_last_answer FROM TOPIC
WHERE kodofrazdel=идент.номеру выбранного пользователем раздела
ORDER BY date_last_answer

```

Листинг `show_topic.php` получился довольно большим, но в основном это из-за HTML-кода

В строке 5 происходит выполнение этого запроса и ссылка на результат сохраняется в переменной `$data`. В строке 8 формируется еще один SQL-запрос, предназначение которого — получить имя выбранного пользователем раздела, т. к. оно тоже понадобится. Сам запрос представлен далее:

```

SELECT name
FROM TOPIC

```

WHERE id=идентификационный номер выбранного пользователем раздела

В строке 10 происходит выполнение этого запроса, ссылка на результат сохраняется в переменной `$data2`, затем в строке 12 она сразу обрабатывается функцией `mysql_fetch_row()` и результат этого действия — имя выбранного пользователем раздела — сохраняется в переменной `$line2`.

В строках 15 и 16 выводится текст "Список тем для раздела" и значение переменной `$line2`. А в строке 22 создается кнопка Создать новую тему, нажатие которой приведет к вызову файла `index.php` со следующей строкой параметров: `?show=add_topic&numrazdel=<?php echo $_GET['numrazdel'];`

то есть это означает, что необходимо будет вывести форму создания новой темы для раздела с идентификационным номером, хранящимся в `$_get['numrazdel']`.

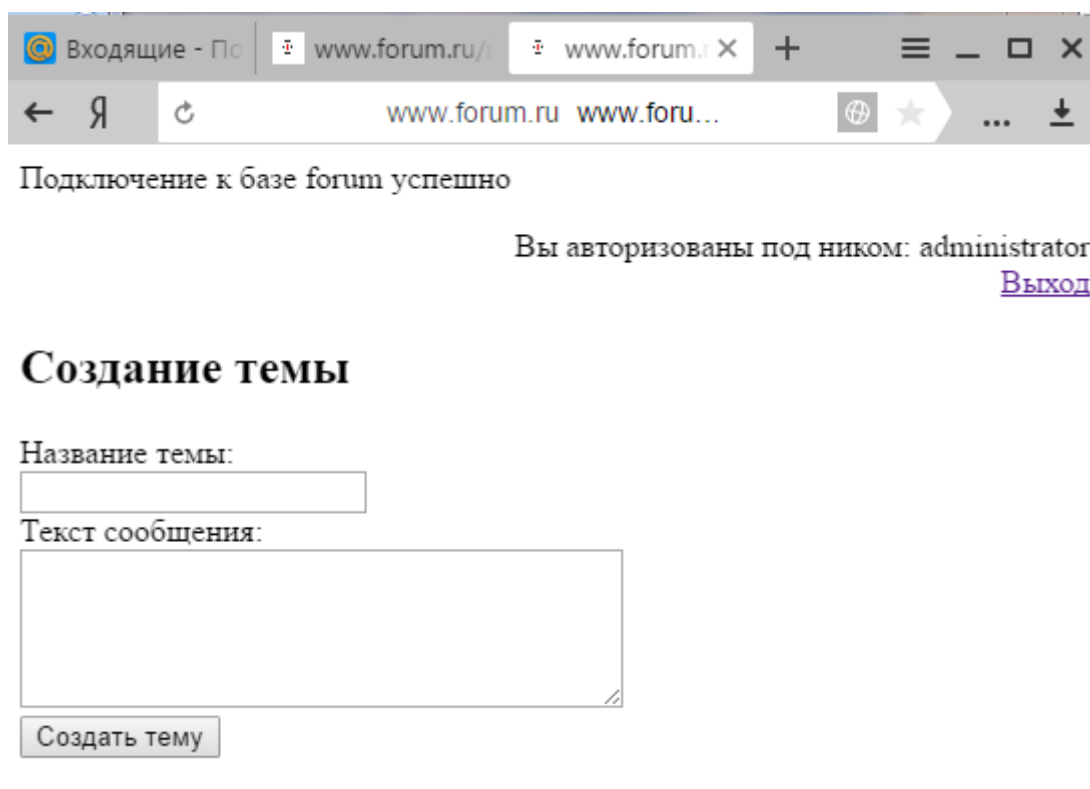
В строке 56 осуществляется проверка роли пользователя, и если он имеет права администратора, то ему будет доступен дополнительный инструмент для работы с форумом, реализованный в виде двух кнопок: Изменить название и Удалить тему.

Создайте файл **SHOW_MODULE/show_addtopic.php** (листинг 3).

Листинг 3. Файл `SHOW_MODULE/show_addtopic.php`

```
<?php
echo "<H2>Создание темы</H2>";
?>
<form action="?act=add_topic&numrazdel=<?php echo
$_GET['numrazdel']?>" method="post">
Название темы:<BR>
<input name="name_topic" type="text" value="">
<BR>
Текст сообщения:<BR>
<textarea name="message" cols=40 rows=5></textarea>
<BR>
<input type="submit" value="Создать тему">
</form>
```

Как видите (рис. 3), это обычная форма с полем ввода, именуемым `name_topic` и предназначенным для ввода названия темы, и полем для ввода многострочного текста с именем `message`, предназначенным для ввода сообщения для создаваемой темы. Также имеется кнопка Ответить, нажатие которой приведет к вызову файла `index.php` со строкой параметров: `?act=add_topic&numrazdel=` номер выбранного пользователем раздела.



Подключение к базе forum успешно

Вы авторизованы под ником: administrator
[Выход](#)

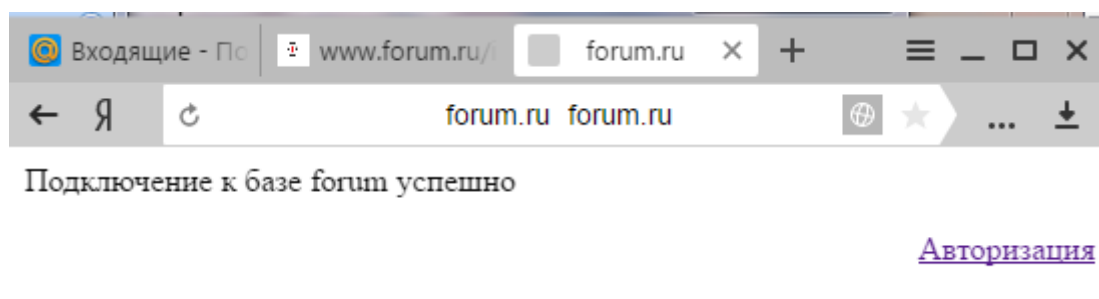
Создание темы

Название темы:

Текст сообщения:

Рис. 1. Форма для создания темы

Запустите форум (введите в браузере <http://forum.ru>), увидите следующее — рис. 2.



Подключение к базе forum успешно

[Авторизация](#)

Рис. 2. Вход в Форум

Нажмите **Авторизация** и наберите
 Имя administrator
 Пароль 12345
 (рис. 3)

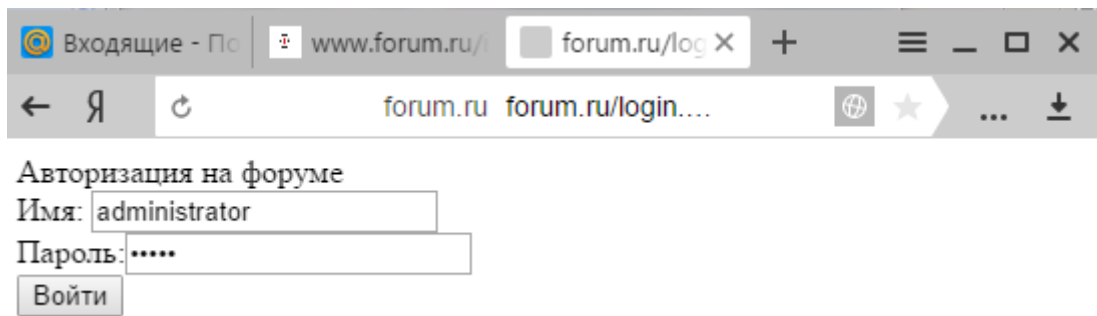


Рис. 3. Авторизация на Форуме

Нажмите Войти. Вы увидите (рис. 4)

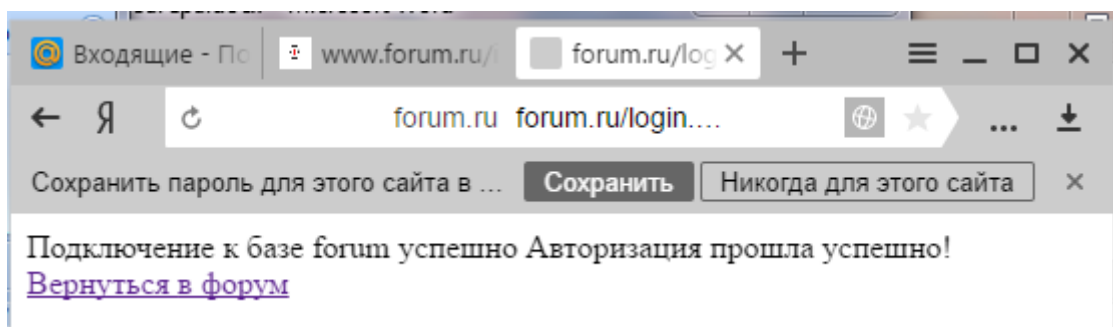


Рис. 4. Успешная авторизация

Нажмите Вернуться в форум. Результат смотрите на рис. 5.

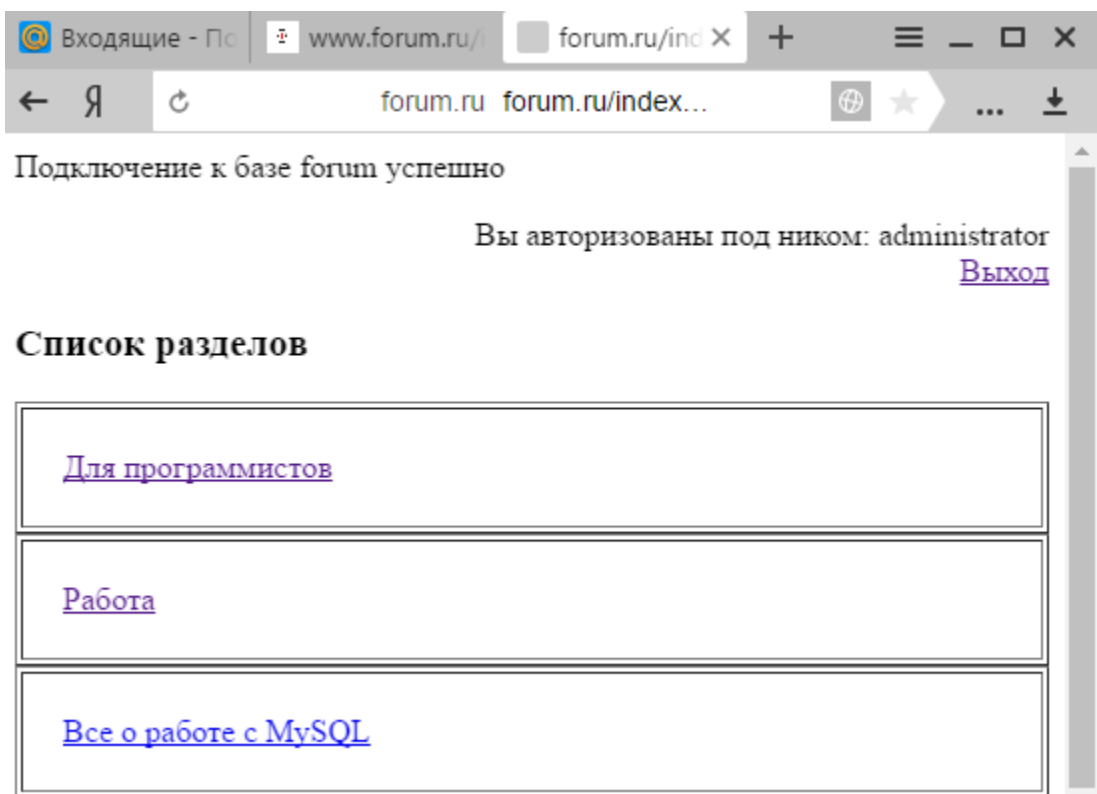


Рис. 5. Списки разделов

Выберите любой раздел (например, Для программистов). Вы увидите следующее — рис. 6. Обратите внимание, что если вы пройдете авторизацию, как **user** (имя **guest** пароль **user**), то вам не будет доступна часть режимов администратора. В частности, вы не можете удалять темы или изменять их **название** (см. рис. 6-а).

Подключение к базе forum успешно

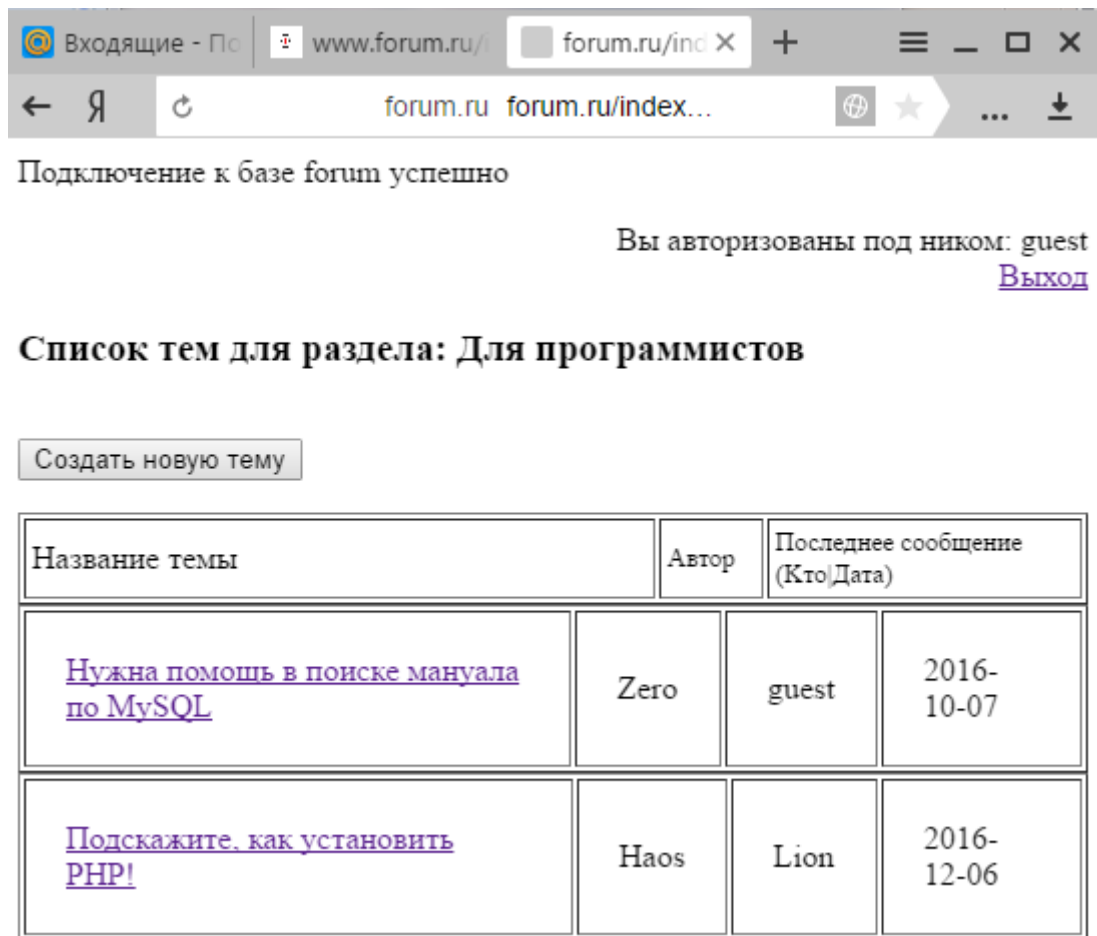
Вы авторизованы под ником: administrator [Выход](#)

Список тем для раздела: Для программистов

Создать новую тему

Название темы	Автор	Последнее сообщение (Кто Дата)	
Нужна помощь в поиске мануала по MySQL Изменить название Удалить тему	Zero	guest	2016-10-07
Подскажите, как установить PHP! Изменить название Удалить тему	Haos	Lion	2016-12-06

Рис. 6. Просмотр списка тем в режиме администратора



Подключение к базе forum успешно

Вы авторизованы под ником: guest
[Выход](#)

Список тем для раздела: Для программистов

Создать новую тему

Название темы	Автор	Последнее сообщение (Кто Дата)	
Нужна помощь в поиске мануала по MySQL	Zero	guest	2016-10-07
Подскажите, как установить PHP!	Haos	Lion	2016-12-06

Рис. 6-а. Просмотр списка тем в режиме user

Нажмите кнопку Создать новую тему. Вы увидите следующее — рис. 7.

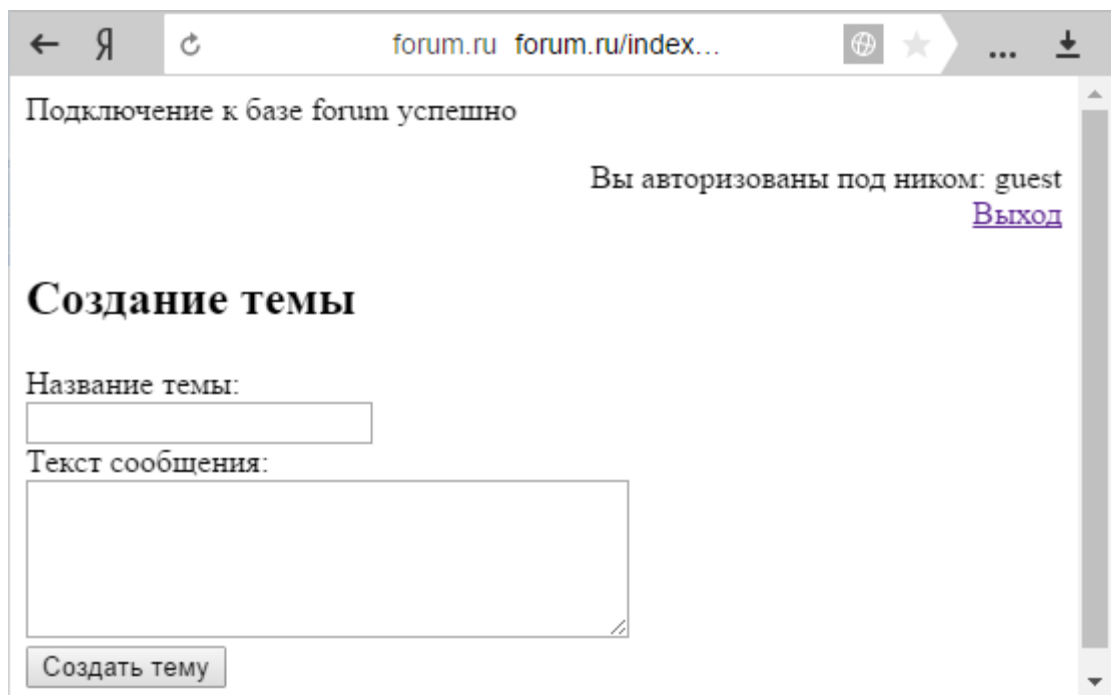


Рис. 7. Создание новой темы

Создайте файл **SHOW_MODULE/show_edittopic.php** (листинг 4).

Листинг 4. Файл **SHOW_MODULE/show_edittopic.php**

```
<?php
echo "<H2>Редактирование темы</H2>";
//Задаем SQL-запрос, который выберет данные по теме
$sql="SELECT id,name FROM TOPIC WHERE id= " . $_GET['numtopic'];
//Выполняем его
$data=mysql_query($sql)or die(mysql_error());
//Получаем результат запроса - одна запись
$line=mysql_fetch_row($data);
?>
<!--Создаем форму для редактирования темы-->
<form action="?act=edit_topic&numtopic=<?php echo $line[0];?>" method="post">
Название темы:<BR>
<input name="name_topic" type="text" value="<?=$line[1]?>" size=40>
<BR>
<input type="submit" value="Изменить">
</form>
```

Этот мини-модуль выводит обычную форму с единственным элементом — полем ввода, в котором уже будет находиться название темы, выбранное пользователем для редактирования. После нажатия кнопки **Изменить** произойдет обращение к файлу **index.php** с передачей ему следующей строки параметров:

?act=edit_topic&numtopic=идент.номер темы, которая подверглась редактированию

На рис. 6 показан форум, в котором отображается список тем в режиме администратора, а на рис. 8. изображена форма редактирования названия темы.

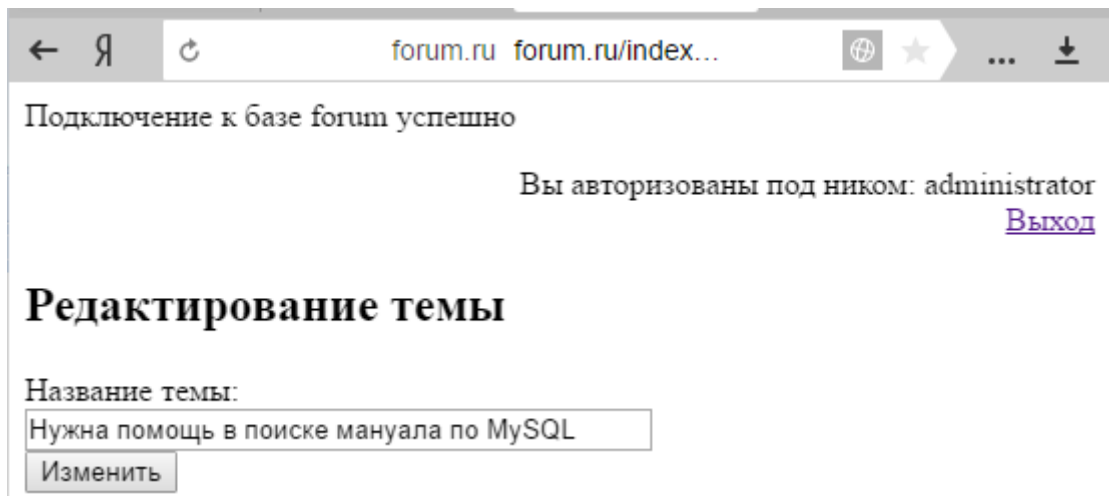


Рис. 8. Форма для редактирования названия темы

Создайте файл SHOW_MODULE/show_deltopic.php (листинг 5).

Листинг 5. Файл SHOW_MODULE/show_deltopic.php

```
<?php
echo "<H2>Удаление темы</H2>";
//Задаем SQL-запрос, который выберет данные по удаляемой теме
$sql="SELECT id, name FROM TOPIC WHERE id=".$_GET['numtopic'];
//Выполняем его
$data=mysql_query($sql) or die(mysql_error());
//Получаем результат - одна запись
$line=mysql_fetch_row($data);
//Выводим надпись
echo "Вы действительно хотите удалить тему: <B>".$line[1]."</B>, и все ее
сообщения?";
?>
<!--Создаем форму для удаления темы-->
<form action="?act=del_topic&numtopic=<?php echo $line[0]?>"
method="post">
<input type="submit" value="Да ">
</form>
<form action="index.php" method="post">
<input type="submit" value="Отмена">
</form>
```

Этот мини-модуль выводит запрос на подтверждение удаления темы, а также две кнопки **Да** и **Отмена** каждая из них расположена в отдельной форме (рис. 9.).

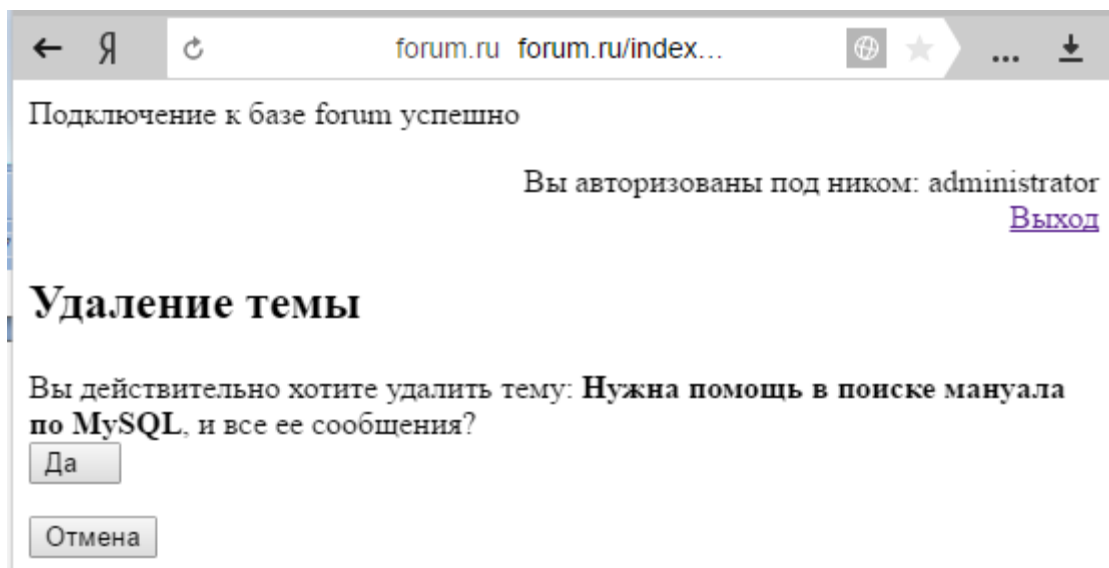


Рис. 9. Форма подтверждения удаления выбранной темы

Нажатие на кнопку **Да** приведет к вызову файла `index.php` с передачей ему следующей строки параметров:

`?act=del_topic&numtopic=` идент. номер темы, которая подверглась редактированию

Нажатие же кнопки **Отмена** осуществит отмену действия.

Создайте файл **SHOW_MODULE/show_message.php** (листинг 6).

Он будет отвечать за вывод сообщений для выбранной пользователем темы.

Листинг 6. Файл **SHOW_MODULE/show_message.php**

```
<?php
//Задаем SQL-запрос, который выберет все сообщения для
//заданной темы
$sql="SELECT id, textmessage, name_man, date_answer".
" FROM MESSAGE WHERE kodoftopic=".$_GET['numtopic'].
" ORDER BY date_answer";
//Выполняем его
$data=mysql_query($sql);
//Задаем SQL-запрос, который вернет имя выбранной пользователем
//темы
$sql2="SELECT name FROM TOPIC WHERE id=".$_GET['numtopic'];
//Выполняем его
$data2=mysql_query ($sql2) ;
//Получаем результат - одна запись
$line2=mysql_fetch_row($data2);
//Выводим надпись
echo "<BIG><B>Список сообщений для ";
echo "темы: ". $line2[0]."</B></BIG><BR><BR>";
//Выводим заголовок для таблицы
?>
<table BORDER=1 cellpadding=3 width=100%>
```

```

<tr>
<td width=70%>
Сообщение
</td>
<td width=10%>
<font size=2>Автор</font>
</td>
<td width=20%>
<font size=2>Дата</font>
</td>
</tr>
</table>
<?php
//Выводим список всех сообщений для выбранной темы
while($line=mysql_fetch_row($data))
{
?>
<table BORDER=1 cellpadding=20 width=100%>
<tr>
<td width=70%>
<?php
echo $line[1];
//Если это админ, то он может редактировать сообщение и
//удалять его
43 if($_SESSION['role']=='admin')
{
?>
<form action="?show=edit_message&nummessage=<?=$line[0]?>"
method="post">
<input type="submit" value="Редактировать сообщение">
</form>
<form action="?show=del_message&nummessage=<?=$line[0]?>"
method="post">
<input type="submit" value="Удалить сообщение">
</form>
<?php
}
//end -if
?>
</td>
<td width=10%>
<?php
//имя пользователя, создавшего сообщение
echo $line[2];
?>
</td>
<td width=20%>
<?php

```

```

//Дата размещения сообщения
echo $line[3];
?>
</td>
</tr>
</table>
<?php

//end - while
?>
<form action="?act=add_message&numtopic=<?php echo
$_GET['numtopic']?>" method="post">
Текст сообщения:<BR>
<textarea name="message" cols=40 rows=5></textarea>
<BR>
<input type="submit" value="Ответить">
</form>

```

Этот мини-модуль выбирает список всех сообщений для темы, идентификатор которой передан в `$_get ['numtopic']`, сообщения сортируются по дате размещения, т. е. последние будут снизу. Под последним сообщением будет также размещена форма для добавления нового сообщения с кнопкой **Ответить**, нажатие которой приведет к вызову файла `index.php` с передачей ему следующей строки параметров:

`?act=add_message&numtopic=` идент. номер темы, для которой добавляется сообщение

В строке 43 осуществляется проверка роли пользователя, и если он имеет права администратора, то ему будет доступен дополнительный инструмент для работы с сообщениями, реализованный в виде двух кнопок: **Редактировать сообщение** и **Удалить сообщение** (рис. 10).

Вы авторизованы под ником: administrator
[Выход](#)

Список сообщений для темы: Нужна помощь в поиске мануала по MySQL

Сообщение	Автор	Дата
<p>Где можно найти мануал по MySQL?</p> <p><input type="button" value="Редактировать сообщение"/></p> <p><input type="button" value="Удалить сообщение"/></p>	Zero	2016-01-05
<p>Ты пробовал смотреть на официальном сайте?</p> <p><input type="button" value="Редактировать сообщение"/></p> <p><input type="button" value="Удалить сообщение"/></p>	MegaMan	2016-02-01
<p>Посмотри Внимательней!</p> <p><input type="button" value="Редактировать сообщение"/></p> <p><input type="button" value="Удалить сообщение"/></p>	guest	2016-10-07

Текст сообщения:

Рис. 10. Работа с сообщениями

Создайте файл **SHOW_MODULE/show_editmessage.php** (листинг 7).

Листинг 7. Файл SHOW_MODULE/show_editmessage.php

```
<?php
echo "<H2>Редактирование сообщения</H2>";
//SQL-запрос, который выберет данные по теме
$sql="SELECT id, textmessage FROM MESSAGE WHERE id="
.$_GET['nummessage'];
//Выполняем запрос
```

```

$data=mysql_query($sql) or die(mysql_error());
//Получаем результат - одна запись
$line=mysql_fetch_row($data);
?>
<form action="?act=edit_message&nummessage=<?php echo $line[0]?>"
method="post">
Текст сообщения : <BR>
<textarea name="message" cols=40 rows=5><?=$line[1]?>
</textarea>
<BR>
<input type="submit" value="Изменить">
</form>

```

Этот мини-модуль представляет собой полную аналогию с SHOW_MODULE/show_edittopic, только сообщение редактируется в поле для ввода многострочного текста и обработчиком формы является:

?act=edit_message&nummessage=идент.номер редактируемого сообщения

На рис. 11. вы можете увидеть только что созданный модуль в работе.

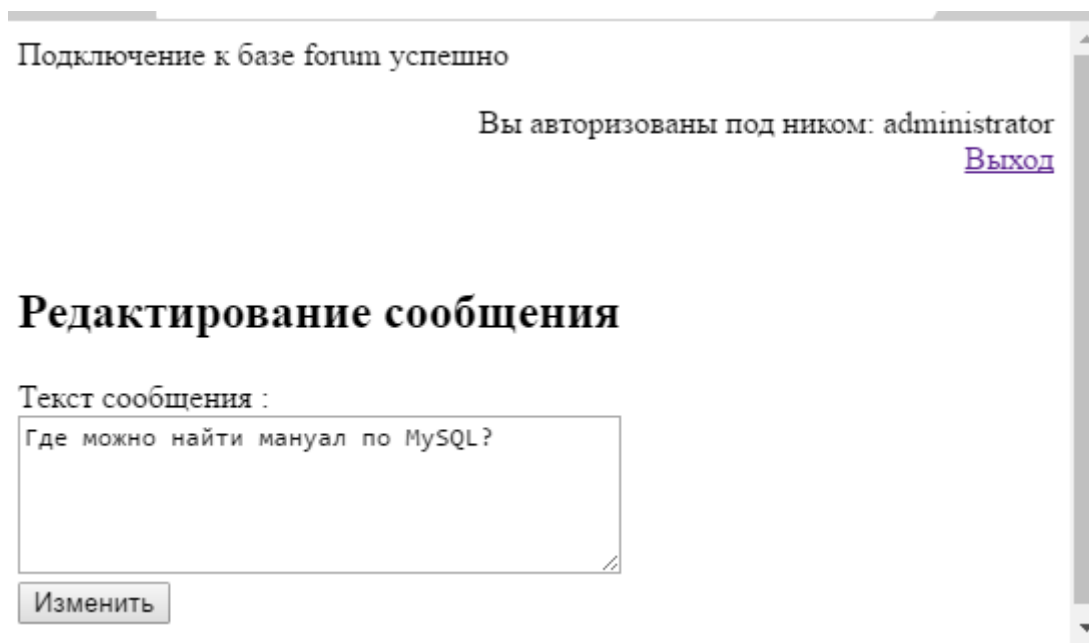


Рис. 11. Редактирование сообщений

И последний модуль из серии SHOW_MODULE — это show_delmessage.php (листинг 8). Он предназначен для удаления сообщения.

Листинг 8 Файл SHOW_MODULE/ show_delmessage.php

```

<?php
echo "<H2>Удаление сообщения</H2>";
//SQL-запрос, который выберет идент. номер удаляемого сообщения
$sql="SELECT id FROM MESSAGE WHERE id=" .$_GET['nummessage'];

```

```

//Выполняем запрос
$data=mysql_query ($sql) or die (mysql_error()) ;
//Получаем результат - одна запись
$line=mysql_fetch_row($data) ;
//Выводим надпись
echo"Вы действительно хотите удалить выбранное сообщение?";
?>
<form action="?act=del_message&nummessage=<?php echo $line[0]?>"
method="post">
<input type="submit" value="Да">
</form>
<form action="index.php" method="post">
<input type="submit" value="Отмена">
</form>

```

Этот мини-модуль представляет собой полную аналогию с SHOW_MODULE/show_deltopic, только обработчиком формы является: ?act=del_message&nummessage= идент. номер редактируемого сообщения

На рис. 12. вы можете увидеть только что созданный модуль в работе.

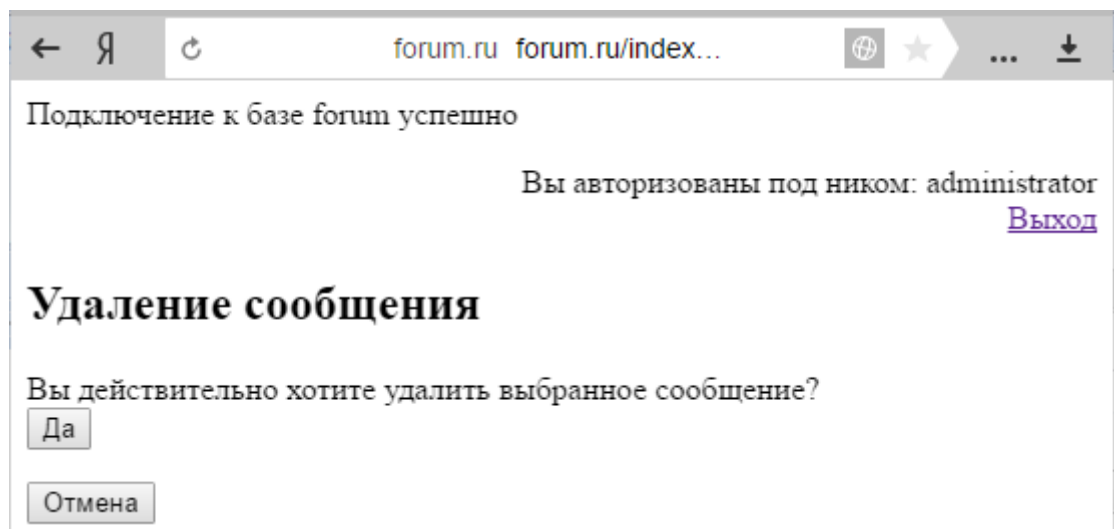


Рис. 12. Удаление сообщения

Задание

1. Изучите Методические указания или соответствующие разделы пособий по указанной тематике [2,3,4,5,6,7,10].
2. Выполните примеры, разобранные в п. 1 методических указаний.

3. По аналогии с указанными примерами разработайте **Модуль вывода информации show.php** с соответствующими подмодулями для **вашего форума**.

Содержание отчета

1. Задание к работе
2. Листинги, скриншоты и пояснения примеров, разобранных в п. 1 методических указаний.
3. **Модуль вывода информации show.php** с подмодулями для **вашего форума** с соответствующими скриншотами и пояснениями к ним.

Лабораторная работа № 12 Практическое занятие № 12

Организация действий в форуме

Цель работы

Ознакомление с методикой разработки **Модуля действий action.php** форума.

Методические указания

1. Модуль действий **action.php**

Создайте в forum.ru/www файл action.php, его содержимое представлено в листинге 1.

Листинг 1. Файл action.php

```
<?php

//Добавление темы
if($_GET['act']=='add_topic')
{
//Обрабатываем название темы в целях безопасности
$safe_topic=mysql_escape_string($_POST['name_topic']);
//SQL-запрос для добавления темы
$sql="INSERT INTO TOPIC SET
```

```

kodofrazdel="$_GET['numrazdel']." ,name= "" . $safe_topic.", name_creator=
"".$_SESSION['name'].", date_last_answer= ""
. date('Y-m-d')."";
//Выполняем запрос
mysql_query($sSQL)or die (mysql_error());
//Обрабатываем текст сообщения в целях безопасности
$safe_message=mysql_escape_string($_POST['message']);
//Определяем номер созданной темы
$id=mysql_insert_id();
//SQL-запрос, добавляющий сообщение для вновь созданной темы
$sSQL="INSERT INTO MESSAGE SET kodoftopic=".$id.",
textmessage="".$safe_message. "",name_man="" .$_SESSION['name']
. "" , date_answer= "".$date ( 'Y-m-d')."";
//Выполняем запрос
mysql_query ($sSQL) or die (mysql_error());
//Выводим надпись и ссылку на список тем для текущего раздела
echo "Тема Создана <BR>";
echo "<a href='index.php?show=topic&numrazdel=
".$_GET['numrazdel']."'>";
echo "Назад к списку тем</a>";

}

```

```

//Изменение названия темы
if($_GET['act']=='edit_topic')
{
//Обрабатываем название темы в целях безопасности
$safe_topic=mysql_escape_string($_POST['name_topic']);
//SQL-запрос, который изменит название темы
$sSQL="UPDATE TOPIC SET name="".$safe_topic."
WHERE id="".$_GET['numtopic'];
//Выполняем запрос
mysql_query($sSQL)or die(mysql_error()) ;
//Выбираем код раздела, чтобы можно было перенаправить
//пользователя на список тем для этого раздела
$sSQL="SELECT kodofrazdel FROM TOPIC
WHERE id="".$_GET['numtopic'];
//Выполняем запрос
$data=mysql_query($sSQL);
//Получаем результат - одна запись
$line=mysql_fetch_row($data);
//Выводим надпись и ссылку на список тем для текущего раздела
echo "Название темы изменено<BR>";
echo "<a href='index.php?show=topic&numrazdel=$line[0]'>";
echo "Назад к списку тем</a>";
}
//Удаление темы и всех ее сообщений

```

```

if($_GET['act'] == 'del_topic')
{
    //Выбираем код раздела, чтобы можно было вернуться в него
    $sSQL="SELECT kodofrazdel FROM TOPIC WHERE id=".$_GET['numtopic'];
    $data=mysql_query($sSQL);
    //Получаем результат - одна запись
    $line=mysql_fetch_row($data);
    //Удаляем все сообщения для выбранной темы
    $sSQL="DELETE FROM MESSAGE WHERE kodoftopic
   =".$_GET['numtopic'];
    mysql_query($sSQL) or die (mysql_error());
    //Удаляем саму тему
    $sSQL="DELETE FROM TOPIC WHERE id=".$_GET['numtopic'];
    mysql_query($sSQL) or die(mysql_error( ));
    //Выводим надпись и ссылку на список тем для текущего раздела
    echo "Тема удалена<BR>";
    echo "<a href='index.php?show=topic&numrazdel=$line[0]'">Назад к списку
    тем</a>";
}
//Добавление нового сообщения
if($_GET['act']=='add_message')
{
    //Обрабатываем текст в целях безопасности
    $safe_message=mysql_escape_string($_POST['message']);
    //Запрос для добавления сообщения
    $sSQL="INSERT INTO MESSAGE SET kodoftopic=".$_GET['numtopic'].",
    textmessage='".$safe_message."', name_man='".$_SESSION[
    'name']."', date_answer= '". date ('Y-m-d')."''";
    //Выполняем запрос
    mysql_query($sSQL) or die(mysql_error());
    //Теперь добавляем информацию об имени посетителя и дате
    //размещаемого сообщения для темы, которой принадлежит сообщение
    $sSQL="UPDATE TOPIC SET name_last_answer='".$_SESSION['name'].',
    date_last_answer= '".date('Y-m-d')."''WHERE id=".$_GET['numtopic'];
    mysql_query($sSQL) or die(mysql_error());
    //Выводим надпись и ссылку на список сообщений для текущей темы
    echo "Ответ принят<BR>";
    echo "<a href='index.php?show=message&numtopic=
    ".$_GET['numtopic']."'>";
    echo "Назад к обсуждению темы</a>";
}
//Изменение сообщения
if($_GET['act']=='edit_message')
{
    //Обрабатываем название в целях безопасности
    $safe_message=mysql_escape_string($_POST['message']);
    //Меняем текст сообщения
    $sSQL="UPDATE MESSAGE SET textmessage='".$safe_message.'"

```

```

WHERE id=".$_GET['nummessage'];
mysql_query($sSQL) or die(mysql_error());
//Выбираем код темы, чтобы можно было перенаправить
//пользователя на список сообщений для этой темы
$sSQL="SELECT kodoftopic FROM MESSAGE WHERE
id=".$_GET['nummessage'];
$data=mysql_query($sSQL);
//Получаем результат - одна запись
$line=mysql_fetch_row($data);
//Выводим надпись и ссылку на список сообщений для текущей темы
echo"Название сообщения изменено<BR>";
echo"<a href = 'index.php?show=message&numtopic=".$line[0]."'>";
echo"Назад к обсуждению темы </a>";
}
//Удаление сообщения
if ($_GET['act']=='del_message')
{
//Выбираем код темы, чтобы можно было вернуться
//в список сообщений для нее
$sSQL="SELECT kodoftopic FROM MESSAGE
WHERE id=".$_GET['nummessage'];
$data=mysql_query($sSQL);
//Получаем результат - одна запись
$line=mysql_fetch_row($data);
//Удаляем выбранное сообщение
$sSQL="DELETE FROM MESSAGE WHERE id=".$_GET['nummessage'];
//Выполняем запрос
mysql_query($sSQL) or die(mysql_error());
//Выводим -надпись и ссылку на список сообщений для текущей темы
echo"Тема удалена<BR>";
echo "<a href='index.php?show=message&numtopic=".$line[0]."'>"; echo
"Назад к обсуждению темы </a>";
}
?>

```

Весь модуль поделен на небольшие условные фрагменты, в которых проверяется значение `$_get['act']`.

Если происходит добавление темы, то `$_get ['act'] = 'add_topic'`, само действие осуществляется с помощью двух SQL-запросов. Первый запрос добавляет в таблицу `topic` информацию о новой теме.

```
INSERT INTO TOPIC
```

```
SET kodoftopic=номер раздела,
name=название создаваемой темы,
name_creator=имя автора,
date_last_answer=дата создания темы;
```

В качестве названия темы берется значение из `$safe_topic`, в качестве имени пользователя, создавшего тему, берется значение из `$_session ['name']`, дата создания темы формируется с помощью стандартной функции `date ()`. Создаваемая тема привязывается к разделу, идентификационный номер которого доступен в

`$_GET ['numrazdel']`.

Второй запрос добавляет в таблицу `message` первое сообщение для только что созданной темы.

```
INSERT INTO MESSAGE
```

```
SET      kodoftopic=номер созданной темы,
```

```
textmessage='текст сообщения',
```

```
name_man='имя автора', date_answer='дата создания сообщения';
```

Для привязки сообщения к теме используется его идентификационный номер, который запрашивается у MySQL с помощью специальной функции `mysql_insert_id()` т. к. он заранее не известен, потому что тема была недавно создана. Синтаксис функции:

```
mysql_insert_id()
```

Данная функция возвращает номер, сгенерированный MySQL при последнем INSERT-запросе, для столбца, у которого установлена автоматическая нумерация.

Когда происходит изменение названия темы, то `$_get ['act'] ='edit_topic'`, само действие осуществляется с помощью одного SQL-запроса:

```
update topic SET name='название темы'
```

```
WHERE id=номер изменяемой темы;
```

Когда происходит удаление темы, то `$_get['act'] ='del_topic'`, само действие осуществляется с помощью двух SQL-запросов.

Первый запрос удаляет все сообщения из таблицы `message`, которые принадлежат удаляемой теме.

```
DELETE FROM MESSAGE
```

```
WHERE kodoftopic=код темы, все сообщения которой будут удалены;
```

Второй запрос удаляет выбранную пользователем тему.

```
DELETE FROM TOPIC WHERE id=номер удаляемой темы;
```

Если происходит добавление сообщения, то `$_get['act'] ='add_message'`, само действие осуществляется с помощью двух SQL-запросов. Первый запрос добавляет в таблицу `message` новое сообщение.

```
INSERT INTO MESSAGE
```

```
SET kodoftopic= код темы, сообщение для которого будет добавлено,
```

```
textmessage='текст добавляемого сообщения', name_man='имя автора',
```

```
date_answer='дата добавления сообщения'
```

В качестве текста сообщения берется значение из `$safe_message`, в качестве имени пользователя, создавшего сообщение, берется значение из `$_session['name']`, дата создания сообщения формируется с помощью стандартной функции `date()`. Создаваемое сообщение привязывается к теме, идентификационный номер которой доступен в `$_GET['numtopic']`.

Второй запрос изменяет в таблице `topic` имя последнего ответившего автора и дату последнего ответа для темы, сообщение в которую было добавлено первым запросом.

```
UPDATE TOPIC
```

```
SET      name_last_answer='имя автора',
```

```
date_last_answer='дата добавления сообщения'
```

```
WHERE id=номер темы, для которой было добавлено сообщение;
```

Когда происходит изменение сообщения, то `$_get['act'] = 'edit_message'`, само действие осуществляется с помощью одного SQL-запроса:

```
UPDATE MESSAGE
```

```
SET textmessage='текст сообщения1'
```

```
WHERE id=номер изменяемого сообщения
```

Когда происходит удаление сообщения, то `$_get ['act'] = 'del_message'`, само действие осуществляется с помощью одного SQL-запроса:

```
DELETE FROM MESSAGE
```

```
WHERE id=номер удаляемого сообщения
```

Переместите модуль `action.php` из папки **www** в папку **РабМодули**. Запустите Денвер. Наберите в адресной строке браузера

`http://www.forum.ru/`

Авторизуйтесь под ником: `administrator`. Создайте новую тему для раздела: Для программистов (Рис.1)

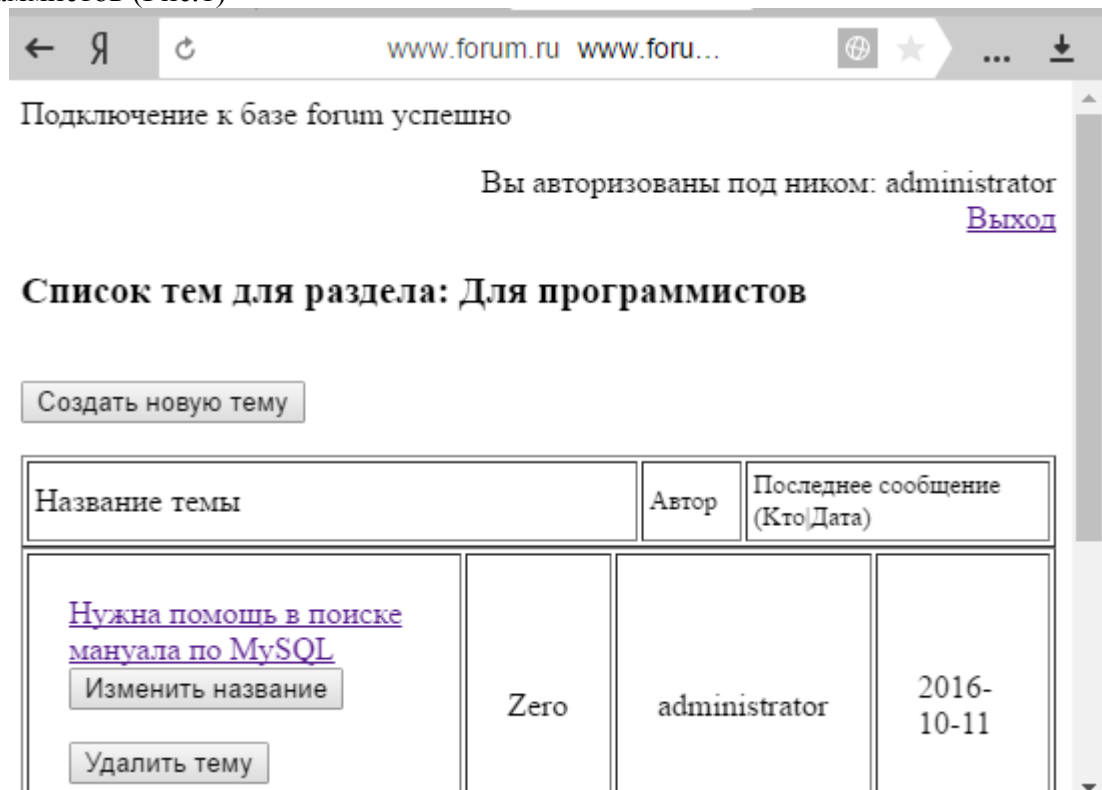
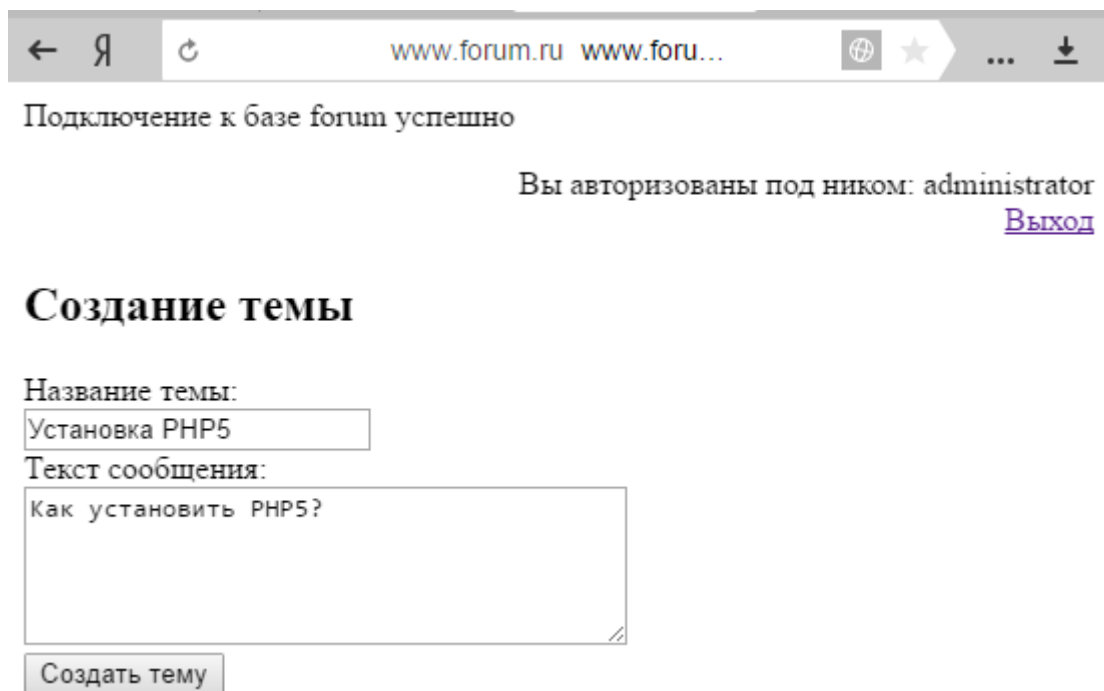


Рис. 1. Результат выполнения запроса Создайте новую тему.

Нажмите **Создать новую тему**, заполните поля формы (Рис.2).



← Я ↻ www.forum.ru www.foru...

Подключение к базе forum успешно

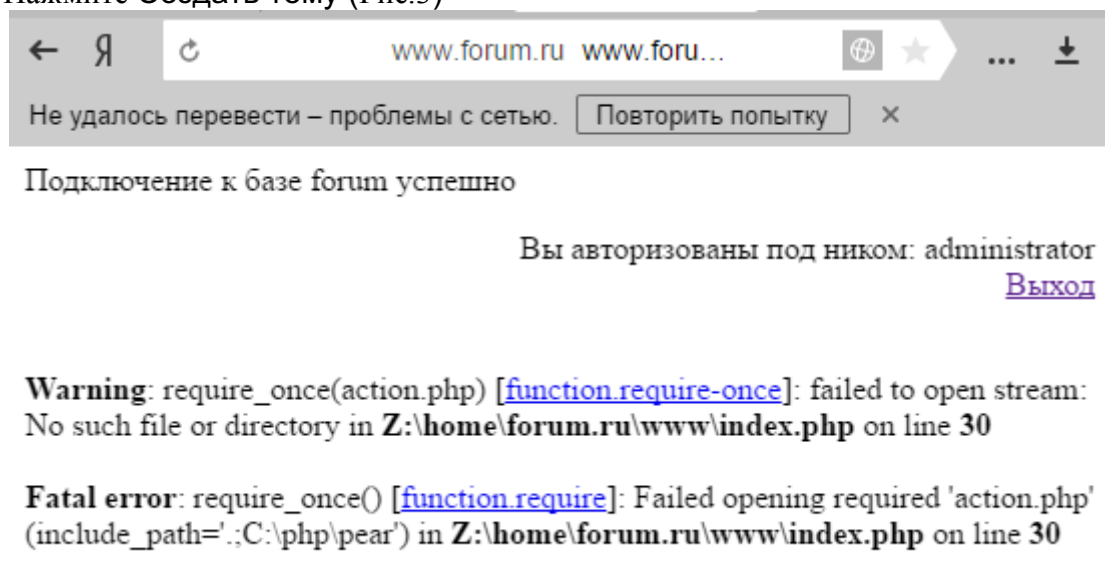
Вы авторизованы под ником: administrator [Выход](#)

Создание темы

Название темы:

Текст сообщения:

Нажмите Создать тему (Рис.3)



← Я ↻ www.forum.ru www.foru...

Не удалось перевести – проблемы с сетью.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator [Выход](#)

Warning: require_once(action.php) [function.require-once]: failed to open stream: No such file or directory in Z:\home\forum.ru\www\index.php on line 30

Fatal error: require_once() [function.require]: Failed opening required 'action.php' (include_path='.:C:\php\pear') in Z:\home\forum.ru\www\index.php on line 30

Рис.3. Неудачная попытка: Создать новую тему.

Аналогичные неудачные попытки возникнут при выполнении пунктов:

- Удалить тему.
- Изменить название темы.
- Изменить сообщение.
- Удалить сообщение.
- Ответить (Создать сообщение).

Верните модуль `action.php` из папки РабМодули в папку **www** и проделайте указанные действия (Рис. 4- Рис. 9):

Рис. 4. Результат выполнения запроса Создать новую тему.

Рис. 5. Результат выполнения запроса Удалить тему.

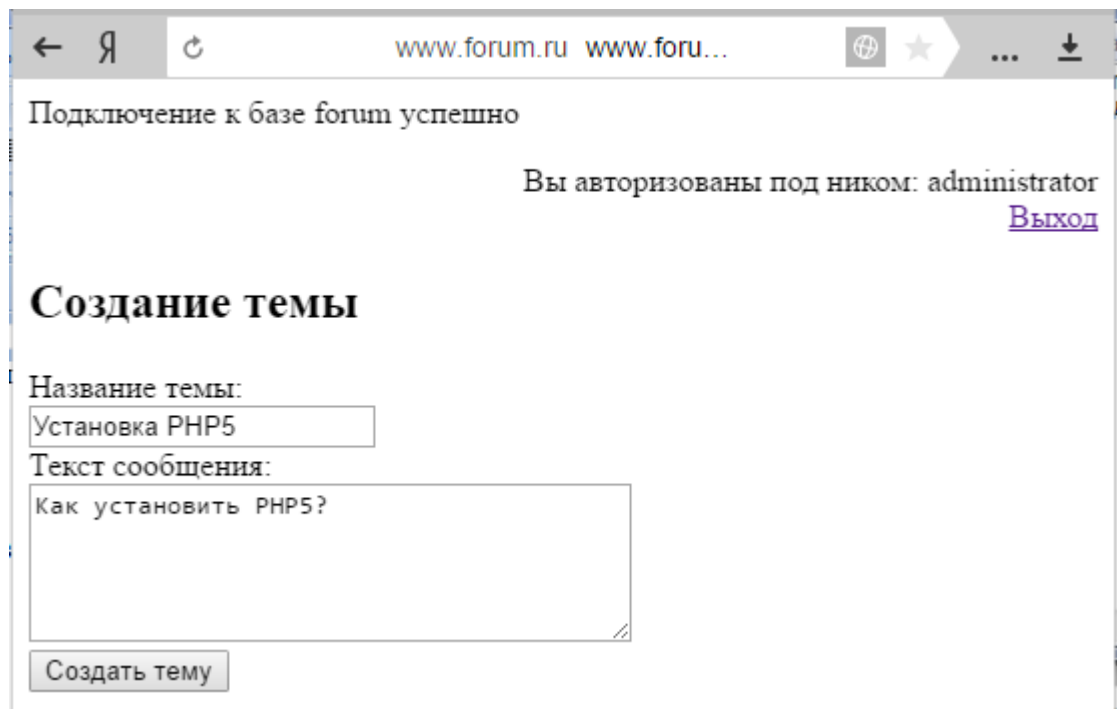
Рис. 6. Результат выполнения запроса Изменить название темы.

Рис. 7. Результат выполнения запроса Редактировать сообщение.

Рис. 8. Результат выполнения запроса Создать сообщение (Ответить).

Рис. 9. Результат выполнения запроса Удалить сообщение.

Создаём новую тему



← Я ↻ www.forum.ru www.foru...

Подключение к базе forum успешно

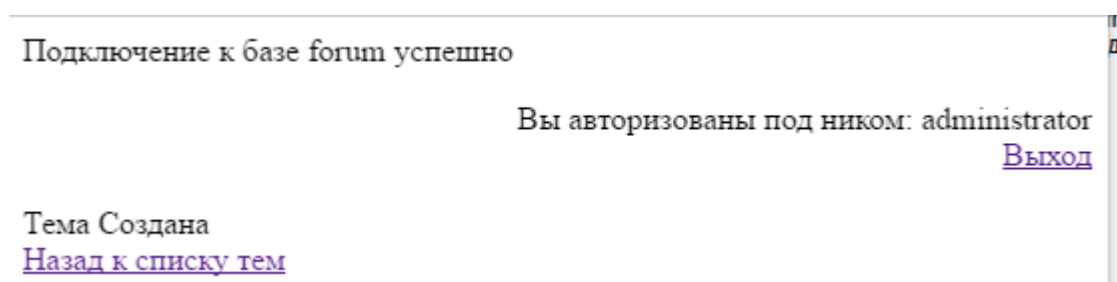
Вы авторизованы под ником: administrator [Выход](#)

Создание темы

Название темы:

Текст сообщения:

Рис. 4а. Результат выполнения запроса Создать новую тему



Подключение к базе forum успешно

Вы авторизованы под ником: administrator [Выход](#)

Тема Создана
[Назад к списку тем](#)

Рис. 4б. Результат выполнения запроса Создать новую тему.

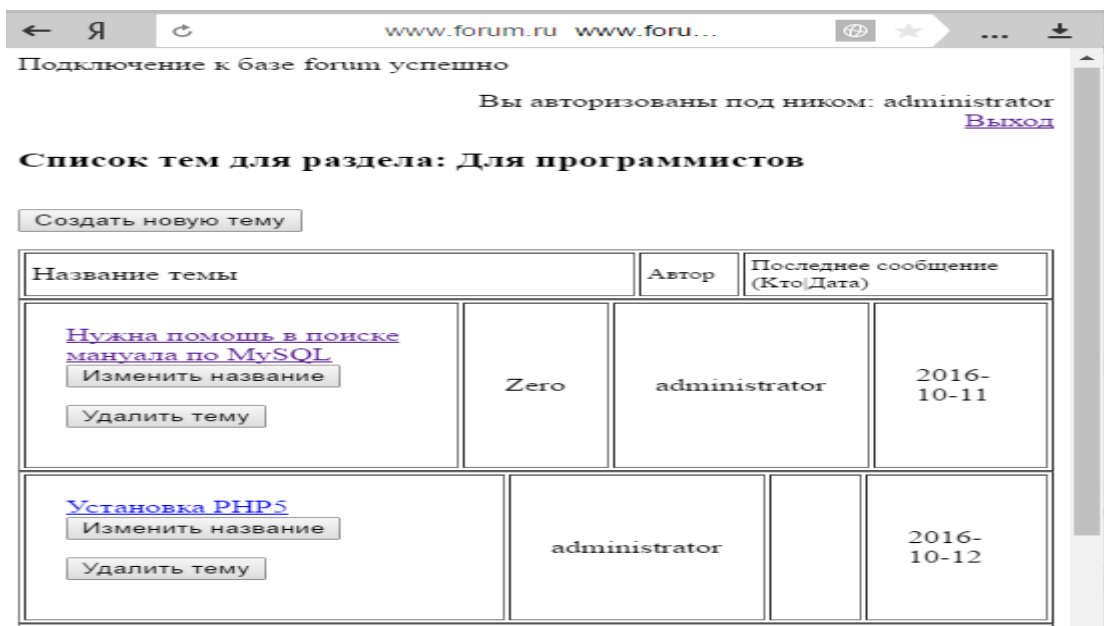


Рис. 4. Результат выполнения запроса Создать новую тему.

Удаляем тему.

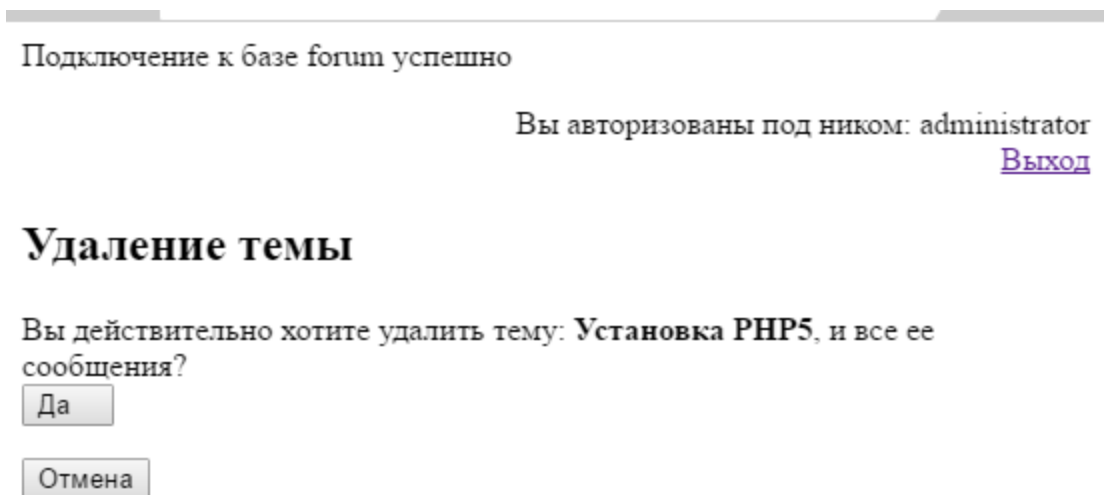


Рис. 5а. Результат выполнения запроса Удалить тему.

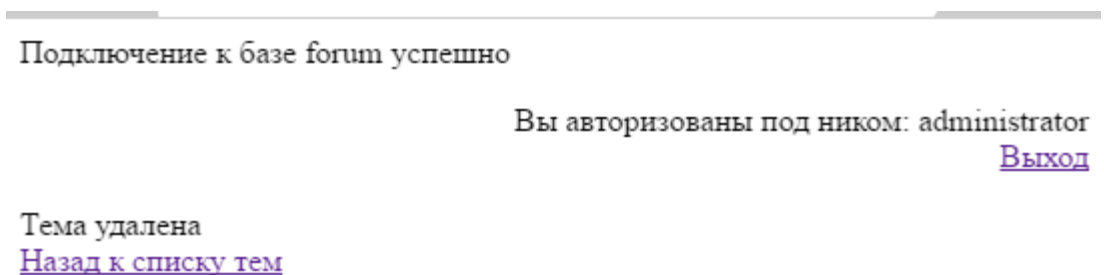


Рис. 5б. Результат выполнения запроса Удалить тему.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator
[Выход](#)

Список тем для раздела: Для программистов

[Создать новую тему](#)

Название темы	Автор	Последнее сообщение (Кто/Дата)	
Нужна помощь в поиске мануала по MySQL Изменить название Удалить тему	Zero	administrator	2016-10-11
Подскажите, как установить PHP! Изменить название Удалить тему	Haos	Lion	2016-12-06

Рис. 5. Результат выполнения запроса Удалить тему.

Изменяем название темы.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator
[Выход](#)

Список тем для раздела: Для программистов

[Создать новую тему](#)

Название темы	Автор	Последнее сообщение (Кто/Дата)	
Нужна помощь в поиске мануала по MySQL Изменить название Удалить тему	Zero	administrator	2016-10-11

Рис. 6а. Результат выполнения запроса Изменить название темы.

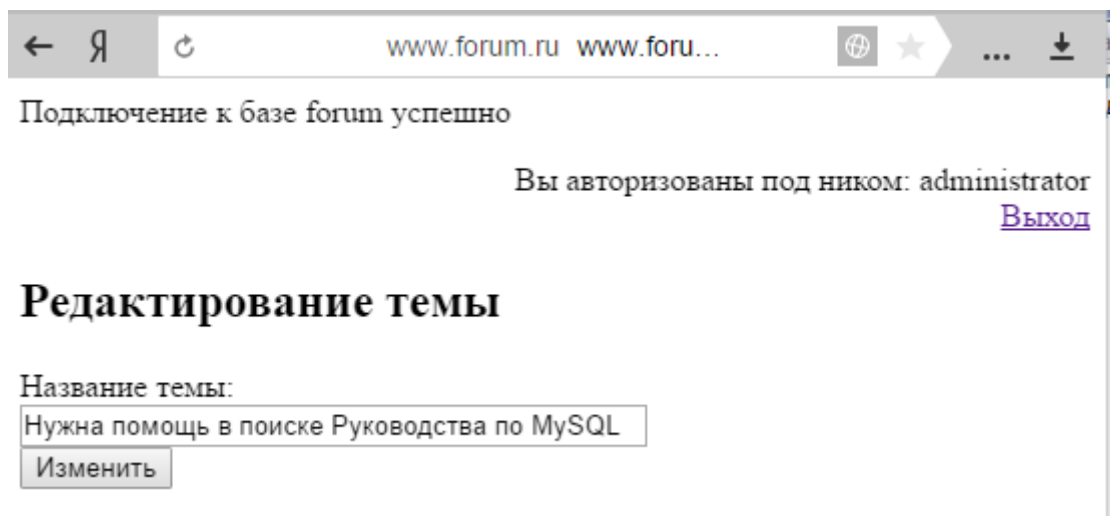


Рис. 6б. Результат выполнения запроса Изменить название темы.

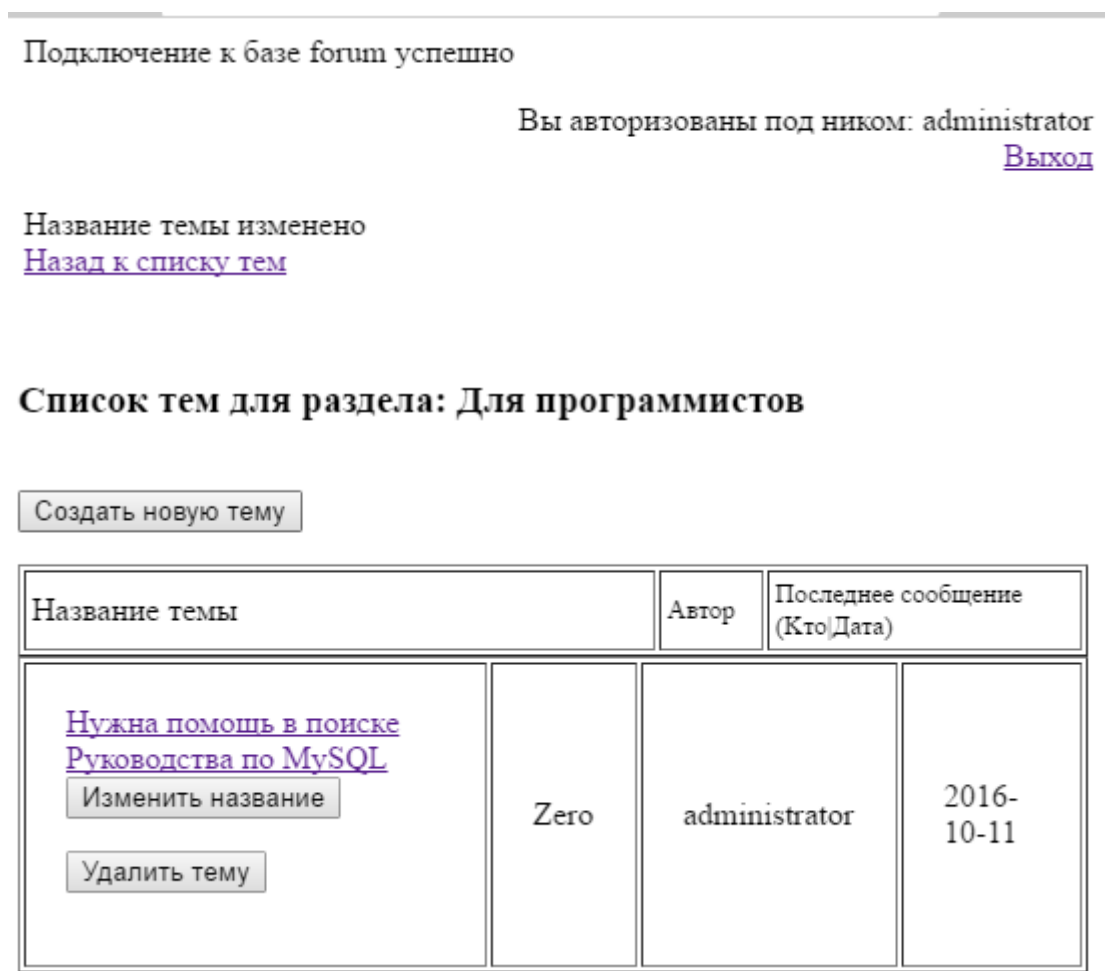


Рис. 6. Результат выполнения запроса Изменить название темы.

Редактирование сообщения

Список сообщений для темы: Нужна помощь в поиске Руководства по MySQL

Сообщение	Автор	Дата
<p>Где можно найти мануал по MySQL??</p> <p>Редактировать сообщение</p> <p>Удалить сообщение</p>	Zero	2016-01-05

Рис. 7а. Результат выполнения запроса Редактировать сообщение.

Редактирование сообщения

Текст сообщения :

Где можно найти Руководство по MySQL??

Изменить

Рис. 7б. Результат выполнения запроса Редактировать сообщение.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator

[Выход](#)

Название сообщения изменено

[Назад к обсуждению темы](#)

Рис. 7в. Результат выполнения запроса Редактировать сообщение.

Список сообщений для темы: Нужна помощь в поиске Руководства по MySQL

Сообщение	Автор	Дата
<p>Где можно найти Руководство по MySQL??</p> <p>Редактировать сообщение</p> <p>Удалить сообщение</p>	Zero	2016-01-05

Рис. 7. Результат выполнения запроса Редактировать сообщение.

Создаём сообщение

Подключение к базе forum успешно

Вы авторизованы под ником: administrator [Выход](#)

Список сообщений для темы: Подскажите, как установить PHP!

Сообщение	Автор	Дата
-----------	-------	------

Текст сообщения:

Ответить

Рис. 8а. Результат выполнения запроса Создать (Ответить) сообщение.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator

[Выход](#)

Список сообщений для темы: Подскажите, как установить РНР!

Сообщение	Автор	Дата
<p>Текст сообщения:</p> <p>Вы искали на официальном сайте?</p>		
<p>Ответить</p>		

Рис. 8б. Результат выполнения запроса Создать (Ответить) сообщение.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator

[Выход](#)

Ответ принят

[Назад к обсуждению темы](#)

Рис. 8в. Результат выполнения запроса Создать (Ответить) сообщение.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator

[Выход](#)

Список сообщений для темы: Подскажите, как установить РНР!

Сообщение	Автор	Дата
<p>Вы искали на официальном сайте?</p> <p>Редактировать сообщение</p> <p>Удалить сообщение</p>	administrator	2016-10-12

Текст сообщения:

[Ответить](#)

Рис. 8. Результат выполнения запроса Создать (Ответить) сообщение.

Удаляем сообщение

Список сообщений для темы: Подскажите, как установить РНР!

Сообщение	Автор	Дата
<p>Вы искали на официальном сайте?</p> <p><input type="button" value="Редактировать сообщение"/></p> <p><input type="button" value="Удалить сообщение"/></p>	administrator	2016-10-12

Рис. 9а. Результат выполнения запроса Удалить сообщение.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator

[Выход](#)

Удаление сообщения

Вы действительно хотите удалить выбранное сообщение?

Рис. 9б. Результат выполнения запроса Удалить сообщение.

Подключение к базе forum успешно

Вы авторизованы под ником: administrator [Выход](#)

Список сообщений для темы: Подскажите, как установить PHP!

Сообщение	Автор	Дата
Текст сообщения:		
<div></div>		
<input type="button" value="Ответить"/>		

Рис. 9. Результат выполнения запроса Удалить сообщение.

На этом закончим рассмотрение примера использования PHP и MySQL при проектировании Web-ресурса. Заметим, что данный пример является учебным. Ограничение на объем не позволили осветить такие вопросы, как задание пароля и логина, или работу с разделами Форума через Web-форму. Здесь это выполняется при генерации БД с помощью phpMyAdmin. Этот материал предлагается на самостоятельную проработку.

Задание

1. Изучите Методические указания или соответствующие разделы пособий по указанной тематике [2,3,4,5,6,7,10].
2. Выполните примеры, разобранные в п. 1 методических указаний.
3. По аналогии с указанными примерами разработайте **Модуль действий action.php** для **вашего форума**.

Содержание отчета

1. Задание к работе
2. Листинги, скриншоты и пояснения примеров, разобранных в п.1 методических указаний.

3. Модуль действий action.php для вашего форума с соответствующими скриншотами и пояснениями.

Библиографический список

1. Холкин И. И. Интернет- технологии и системы. Методические указания по выполнению лабораторных работ. № 0448, Москва, МИРЭА, 2005 , — 24 с.
2. Шкрыль А. А. PHP — это просто. Програмируем для Web-сайта. —СПб.: БХВ-Петербург, 2006. — 368 с.
3. Парижский С.М., Литвиненко Н.А. PHP. Теория и практика - К.: "МК-Пресс", 2006. — 384 с.
4. Харрис Э. PHP/MySQL для начинающих. / Пер. с англ. —М.: КУДИЦ-ОБРАЗ, 2005.—384 с.
5. Мазуркевич А.М., Еловой Д.С. PHP: настольная книга программиста. — М.: Новое знание, 2004. — 479 с.: ил.
6. Ульман Л. MySQL / Пер. с англ. — М.: ДМК Пресс; СПб.: Питер, 2004.-352 с.: ил.
7. Колисниченко Д.Н. Самоучитель PHP 5.—СПб.: Наука и Техника, 2004, —576 с: ил.
8. Холкин И. И. Интернет-технологии и системы. Методические указания по выполнению лабораторных работ. Часть 2, № 0939, Москва, МИРЭА, 2010, — 24 с.
9. Холкин И. И. Интернет-технологии и системы. Методические указания по выполнению лабораторных работ. Часть 3, № 1173, Москва, МИРЭА, 2012, — 16 с.
10. Кузнецов М.В. Самоучитель PHP 5. — СПб.: БХВ-Петербург, 2005. — 536 с. (шифр в библиотеке МИРЭА:004 К89).