МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО - КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Пятигорский институт (филиал) СКФУ

Методические указания

по выполнению лабораторных работ
по дисциплине
«БЕЗОПАСНОСТЬ БАЗ ДАННЫХ»
для направления подготовки **10.03.01 Информационная безопасность**
направленность (профиль) **Безопасность компьютерных систем**

Пятигорск
2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1. Цель и задачи изучения дисциплины

Целью дисциплины «Безопасность баз данных» является формирование у обучающихся понимания основ информационной безопасности систем баз данных для последующего практического использования в науке и образовании, а также приобретение набора общекультурных и общепрофессиональных компетенций будущего бакалавра по направлению подготовки 10.03.01 «Информационная безопасность»

2. Оборудование и материалы

Для проведения лабораторных работ необходимо следующее материально-техническое обеспечение: персональный компьютер; проектор; возможность выхода в сеть Интернет для поиска по образовательным сайтам и порталам; интерактивная доска.

3. Наименование лабораторных работ

Тема 4. Общие понятия реляционного подхода к организации БД

Тема 5. Базисные средства манипулирования реляционными данными

Тема 1. Базы данных и файловые системы

Тема 6. Проектирование реляционных БД

Тема 2. Функций СУБД

Тема 7. Структуры внешней памяти в реляционных СУБД

Тема 3. Подходы к организации баз данных

Тема 8. Управление параллельностью работы транзакций

Тема 9. Методы сериализации транзакции

№ Темы дисциплины	Наименование тем дисциплины, их краткое содержание	Объем часов	Из них практическая подготовка, часов
4	Управление параллельностью работы транзакций	4	4
	5 семестр		
5	Методы сериализации транзакций	4	4
1	Подходы к организации баз данных	4	4
6	Язык SQL. Функции и основные возможности	4	4
7	Язык SQL. Функции и основные возможности	4	4
2	Общие понятия реляционного подхода к организации БД	4	4
8	Безопасность SQL при прикладном программировании	4	4
3	Базисные средства манипулирования реляционными данными	4	4
9	Безопасность архитектуры «клиент-сервер»	4	4
Итого за 5 семестр		**36**	36
	Итого	**36**	36

4. Содержание лабораторных работ
Лабораторная работа 1
Тема 1. Базы данных и файловые системы
Подходы к организации баз данных

В системах автоматизации обработки информации значительная часть затрат на разработку падает на проектирование базы данных как основного средства структурного хранения информации и организации доступа к данным. База данных в Visual FoxPro – это совокупность *таблиц* (Table), *представлений* (View) и *отношений* между ними. В БД входят также *хранимые процедуры* (Stored Procedures), которые используются для описания сложных правил проверки целостности данных в БД, а также *соединения* (Connections) для связи с внешними источниками данных. Файлы БД имеют расширение *dbc* и хранят информацию о входящих в нее компонентах.

Создание базы данных в Visual FoxPro осуществляется в интерактивном режиме с помощью визуального инструментария создания базы данных - *конструктора БД* (Database Designer), позволяющего:

- создавать и модифицировать таблицы, хранимые процедуры, представления данных;
- добавлять свободные таблицы;
- определять для таблиц индексы;
- устанавливать отношения между таблицами, которые будут поддерживаться при создании форм и отчётов.

Для создания базы данных необходимо открыть окно конструктора БД воспользовавшись системным меню Visual FoxPro. Для этого в меню *File* (Файл) выбирается команда *New* (Новый). В открывшемся диалоговом окне New необходимо установить опцию *Database* (База данных) и нажать кнопку *New file* (Новый файл). В окне *Create* в поле *Enter* задать имя создаваемой базы данных. Убедившись, что в поле Тип файла установлен тип сохраняемого файла *dbc*, а в раскрывающемся списке Сохранить в правильно указана папка, в которой будет располагаться создаваемая база данных, нажать кнопку Сохранить. Откроется пустое окно базы данных *Database Designer* (Конструктор базы данных) (Рисунок 1). Используя панель инструментов *Database Designer*, команды меню *Database* и контекстного меню, в окне конструктора базы данных можно добавлять имеющиеся свободные таблицы, создавать новые таблицы и представления, модифицировать и удалять существующие, создавать для них индексы, устанавливать отношения между таблицами.

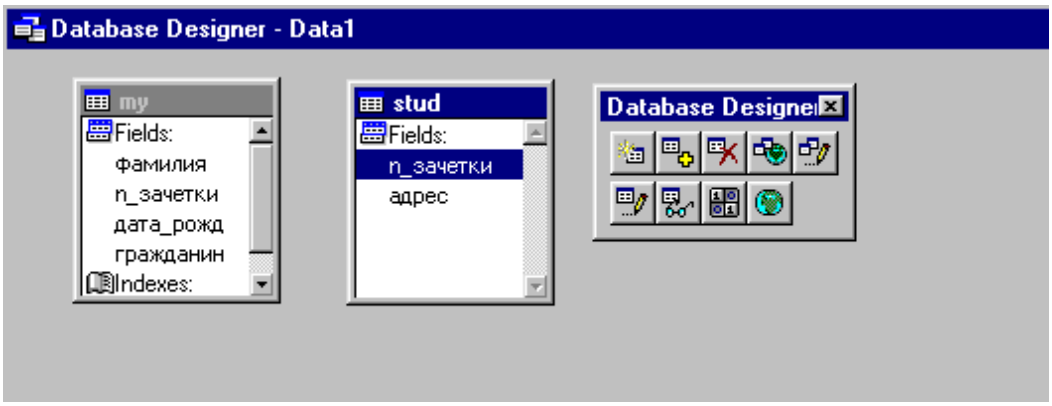


Рисунок 1 - Окно Database Designer

В начале создания базы данных окно конструктора базы данных пусто. Для создания в конструкторе базы данных новых таблиц и модификации существующих, можно использовать команды меню *Database*:
− New Table (Новая таблица) – создаёт новую таблицу;
− Add Table (Добавить таблицу) – добавляет созданную ранее свободную таблицу в базу данных;
− New Remote View (Новое удаленное представление) – создаёт удалённое представление данных;
− New Local View (Новое локальное представление) – создаёт локальное представление данных;
− Modify (Модифицировать) – открывает таблицу в конструкторе таблиц;
− Browse (Обзор таблицы) – показывает содержимое таблицы в режиме Browse;

− Remove (Удалить) – удаляет таблицу из базы данных;
− Find Object (Найти объект) – находит указанный объект в окне конструктора базы данных;
− Rebuild Table Indexes (Перестроить индексы) – перестраивает индексы;
− Remove Deleted Records (Удалить помеченные записи) – физически удаляет из таблицы помеченные для удаления записи;
− Edit Relationship (Редактирование отношений) – редактирует отношения между таблицами;
− Edit Referential Integrity (Редактирование условия целостности) – определяет условия целостности данных;
− Edit Stored Procedures (Редактирование хранимых процедур) – открывает окно редактирования хранимых процедур;
− Connections (Соединения) – выводит на экран диалоговое окно Connections, в котором создаются или модифицируются соединения с удалёнными данными;
− Arrange (Упорядочить) – упорядочивает объекты по имени или типу и выравнивает их по горизонтали или вертикали;
− Refresh (Обновить) – обновляет информацию в окне конструктора базы данных;
− Properties (Свойства) – выводит на экран диалоговое окно Database Properties;
− Clean Up Database (Очистка базы данных) – очищает базу данных от помеченных на удаление объектов.
Для работы в окне конструктора базы данных можно использовать контекстное меню, вызываемое нажатием правой кнопки мыши. Оно содержит наиболее часто используемые команды меню *Database*, команду вызова справочной системы, а также команды *Expand All* (Развернуть все) и *Collapse All* (Свернуть все), предназначенные для раскрытия и свёртывания уровней вложенности объектов в окне конструктора базы данных.
Панель инструментов *Database Designer* содержит кнопки для выполнения наиболее часто используемых операций над базой данных. Если панель инструментов *Database Designer* не видна на экране, в меню *View* (Вид) нужно выбрать команду *Toolbars* (Панель инструментов). Откроется диалоговое окно *Toolbars*, в котором необходимо установить флажок напротив пункта *Database Designer*. Вид панели инструментов *Database Designer* приведён на рисунке 2.

Рисунок 2 - Панель инструментов Database Designer

Ниже приведено описание кнопок панели *Database Designer* (слева направо):

− New Table (Новая таблица) - создаёт новую таблицу;
− Add Table (Добавить таблицу) – добавляет ранее созданную таблицу в базу данных;
− Remove Table (Удалить таблицу) – удаляет таблицу из базы данных;
− New Remote View (Новое удалённое представление) создаёт удалённое представление данных;
− New Local View (Новое локальное представление) – создаёт локальное представление данных;
− Modify Table (Модифицировать таблицу) – открывает таблицу в конструкторе таблиц;
− Browse Table (Обзор таблицы) – показывает содержимое таблицы в режиме Browse;
− Edit Stored Procedures (Редактирование хранимых процедур) – открывает окно для редактирования хранимых процедур;
− Connections (Соединение) – создает связь с удалёнными данными.

Для создания таблицы из конструктора базы данных нужно выполнить одно из следующих действий:

- выбрать команду *New Table* (Новая таблица) из меню *Database* (База данных);
- выбрать команду *New Table* контекстного меню;
- нажать кнопку *New Table* на панели инструментов *Database Designer* (Конструктор базы данных).

В появившемся диалоговом окне *New Table* (Новая таблица) нажать кнопку *Table Wizard* (для запуска Мастера таблиц) или *New File* (для запуска Конструктора таблиц).

При создании таблиц в базе данных есть дополнительные возможности для придания таблице большей функциональности. Окно конструктора таблиц в этом случае содержит дополнительные области для задания параметров, свойств, функций таблицы.

При определении полей таблицы используется вкладка *Fields* (Поля), позволяющая помимо основных параметров указать для каждого поля дополнительные параметры, которые будут определять условия ввода данных в него, а также краткое описание, которое поможет разработчику при модификации таблицы в процессе создания приложения или его сопровождения (Рисунок 3).

В нижней части вкладки *Fields* конструктора таблиц расположены области, позволяющие задать для каждого поля таблицы свойства, которые будут использоваться при вводе данных в эти поля.

Область *Display* (Отображение) содержит поля, позволяющие задать форматы ввода и отображения данных:
− Format (Формат) – задаёт формат отображения данных в формах, отчётах и окне Browse;
− Input mask (Маска ввода) задаёт формат ввода данных;
− Caption (Надпись) – определяет выводимый заголовок поля.

Область *Map field type to classes* (Используемые типы полей для классов) предназначена для указания библиотеки и имени класса, который будет использоваться для создания объектов при размещении данного поля таблицы в форме:
− Display library (Показывать библиотеку) – задаёт местоположение и имя файла библиотеки классов;
− Display class (Показывать класс) – задаёт имя класса из выбранной библиотеки.

Область Field Validation (Проверка правильности ввода) позволяет задать следующие параметры:
− Rule (Условие) – условие правильности ввода данных;
− Message (Сообщение) – сообщение, выводимое при неправильном вводе данных в поле;
− Default Value (Значение по умолчанию) – значение, вводимое в поле по умолчанию.

В текстовом поле *Field comment* (Комментарий) можно ввести краткое описание поля, которое может потребоваться при последующих модификациях структуры таблицы и сопровождении проекта.

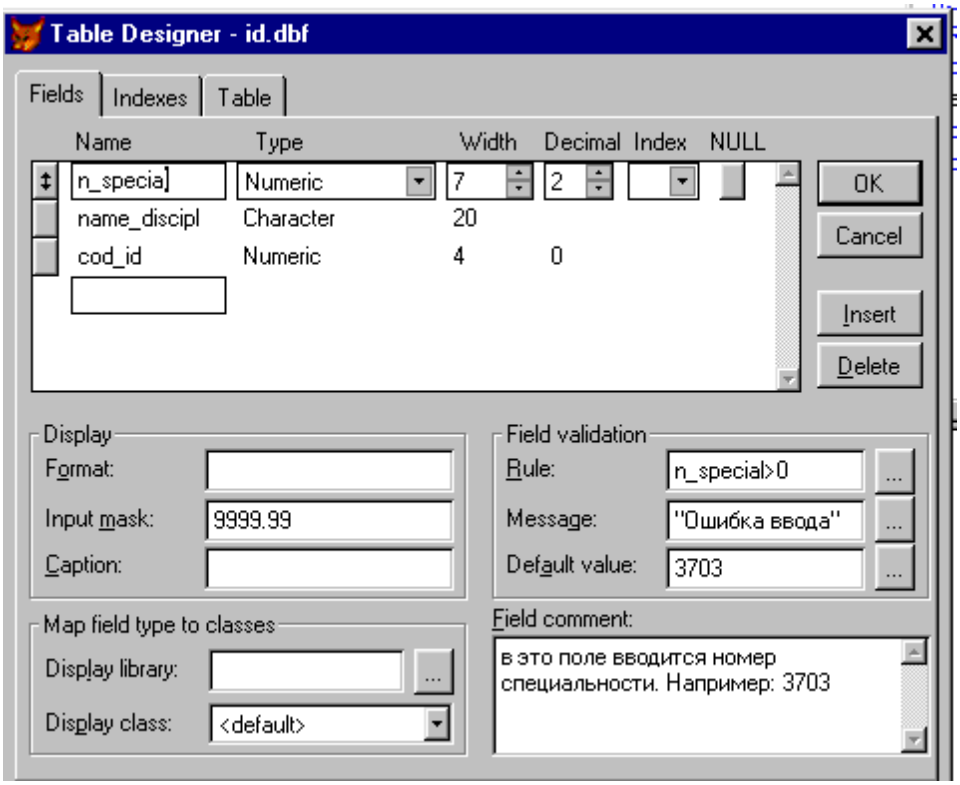


Рисунок 3 -Вкладка Fields Конструктора таблиц с дополнительными областями.

На вкладке *Indexes* (Индексы) можно задать новые и переопределить имеющиеся индексы в таблице. В таблице БД в отличие от свободной таблицы можно задать первичный ключ - *Primary Key*. При этом нужно помнить, что поле может использоваться для задания первичного ключа таблицы только в том случае, если оно содержит неповторяющиеся значения. С помощью кнопок *Insert* и *Delete* добавляют новые индексы в таблицу и удаляют существующие.

Для определения свойств самой таблицы предназначена вкладка *Table* (Таблица) конструктора. Для таблиц, входящих в базу данных, можно задать два имени. Одно вводится в диалоговом окне *Create*, а второе – на вкладке *Table* окна конструктора таблицы. Имя, вводимое в диалоговом окне *Create* при создании таблицы, и будет именем файла, в котором таблица сохраняется на диске. При задании этого имени необходимо придерживаться ограничений, накладываемых операционной системой на количество символов в имени файла. Наименование таблицы может содержать буквы, цифры и знак подчёркивания. Создавая новую таблицу, необходимо помнить, что в базе данных не может быть двух таблиц с одинаковыми именами. Если в базе данных уже имеется таблица с таким именем, на экране появляется запрос, заменить ли существующую таблицу новой.

Второе имя таблицы является внутренним и хранится в базе данных. Внутреннее имя таблицы может содержать до 128 символов. Оно вводится в поле *Name* на вкладке *Table* в верхней части окна конструктора таблицы. Это имя будет отображаться в окне проекта, а также использоваться при создании форм, запросов и отчётов. Используя поле *Table Comment* вкладки *Table*, можно ввести описание таблицы. Для определения условия проверки правильности ввода информации на уровне записей, гарантирующих достоверность данных, вводимых в таблицу, используются поля области *Record validation*. Для проверки значений при добавлении, изменении и удалении записей таблицы предназначены поля области *Triggers: Insert trigger, Update trigger, Delete trigger*.

ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

Лабораторная работа выполняется в среде реляционной СУБД Visual FoxPro. Для выполнения лабораторной работы необходимо:

1) Запустить СУБД Visual FoxPro;
2) Повторить основной материал по технологии создания базы данных (см. список литературы, конспект лекций, справочную систему Visual FoxPro);
3) Изучить описание данной лабораторной работы, выполняя все указанные действия на компьютере в среде СУБД Visual FoxPro;
4) Проверить в системе наличие таблиц данных и других компонентов, которые будут включены в базу данных;
5) Используя командное окно и конструктор БД создать базу данных, включающую две таблицы и одно представление (одну таблицу добавить в БД, вторую таблицу создать с учетом дополнительных возможностей - задать триггеры, индексы и т.д.);
6) Сохранить БД и поработать с ней, выполняя просмотр, редактирование, модификацию структуры таблиц и т.д.;
7) Оформить отчет по работе.

Контрольные вопросы:

1) Дать понятие иерархической модели данных.
2) Перечислить достоинства и недостатки иерархической модели данных.
3) Дать понятие сетевой модели данных.

Лабораторная работа 2.

Тема 2. Функции СУБД

Общие понятия реляционного подхода к организации БД

Отчет — это гибкое и эффективное средство для организации данных при выводе на печать. С помощью отчета имеется возможность вывести необходимые сведения в том виде, в котором требуется. Отчет является объектом Visual FoxPro, предназначенным для представления данных, отбираемых и форматируемых в соответствии с заданными пользователем спецификациями. С помощью отчетов печатаются сводки продаж, накладные, различные ведомости, статистические отчеты, телефонные справочники и другие документы Отчеты можно создавать на основе таблиц и представлений, также в отчет можно
направить результат выполнения запроса SQL. Существуют следующие типы отчетов:

- табличный (по столбцам);
- отчет в свободной форме (например, письмо которое рассылаетсяразным адресатам, взятым из БД);
- стандартный отчет;
- отчет с упорядочением данных по возрастанию или убыванию;
- отчет с группировкой данных по категориям;
- многоколоночный отчет.

В качестве визуального инструментария при создании отчета используется *Конструктор отчетов* (Report Designer).

Для создания связи между отчетом и его исходными данными применяются следующие элементы управления:

- заголовок отчета;
- заголовок каждой страницы отчета;
- номер страницы отчета, используя системную переменную _PAGENO( );
- дату распечатки отчета, используя системную функцию DATE( );
- верхний и нижний колонтитулы столбца;
- значения полей, переменных, функций, элементов массива, вычисляемых полей;
- любой текст, рисунки, линии, прямоугольники.

Для создания отчета необходимо открыть окно конструктора отчетов, воспользовавшись системным меню Visual FoxPro. Для этого в меню *File* (Файл) выбирается команда *New* (Новый). В открывшемся диалоговом окне New необходимо установить опцию *Report* (Отчет) и нажать кнопку *New file*. После выполнения данных действий откроется *окно конструктора отчета* (Report Designer), которое

содержит панель инструментов *Report Controls* (Элементы управления отчета).

Панель инструментов *Report Controls* (Элементы управления отчета) используется для размещения следующих объектов:


-позволяет разместить в отчете любой текст;


-позволяет отображать в отчете данные из полей таблиц, вычисляемые

значения, значения переменных, элементов массива;

- позволяет представить в отчете горизонтальные/вертикальные линии;

-позволяет выделить информацию в отчете прямоугольной рамкой;

-позволяет выделить информацию в отчете рамкой с закругленными краями;

-позволяет вставить в отчет OLE объект или содержимое поля General.

Каждый *объект* помещается в Конструктор отчетов на ту или иную полосу путем выбора этого объекта на панели инструментов щелчком мыши. Щелкнув мышкой в окне

Конструктора отчетов и не отпуская ее, нужно провести по диагонали, чтобы определить месторасположение объекта. Объекты можно перетаскивать, изменять размеры, подписи. Когда объект находится на полосе *Report Designer*, двойным щелчком можно вызвать дополнительное окно, где можно задать свойства этого объекта.
Рабочая область Конструктора отчетов по умолчанию разделена на три полосы. При использовании в отчете группирования данных, итоговых данных и добавления в него титульной страницы появляются дополнительные полосы. Типы возможных полос:
− Title (Титул) – служит для размещения информации, появляющейся перед основным отчетом;
− Page Header (Верхний колонтитул) – служит для задания верхнего колонтитула;
− Group Header (Группа сверху) – служит для размещения информации, используемой при группировке;
− Detail (Детали) – служит для размещения полей из таблиц или результат вычисления над ними.
− Group Footer (Группа снизу) – служит для размещения итоговой информации по группе;
− Page Footer (Нижний колонтитул) – служит для указания нижнего колонтитула страницы;
− Summary (Итоги) - служит для размещения информации, появляющейся после основного отчета.
Для создания стандартного отчета (Рисунок 1) необходимо выполнить команду *Quick Report* из меню *Report*. В появившемся окне стандартного запроса файла нужно выбрать ту таблицу или представление, из которого будут взяты данные.

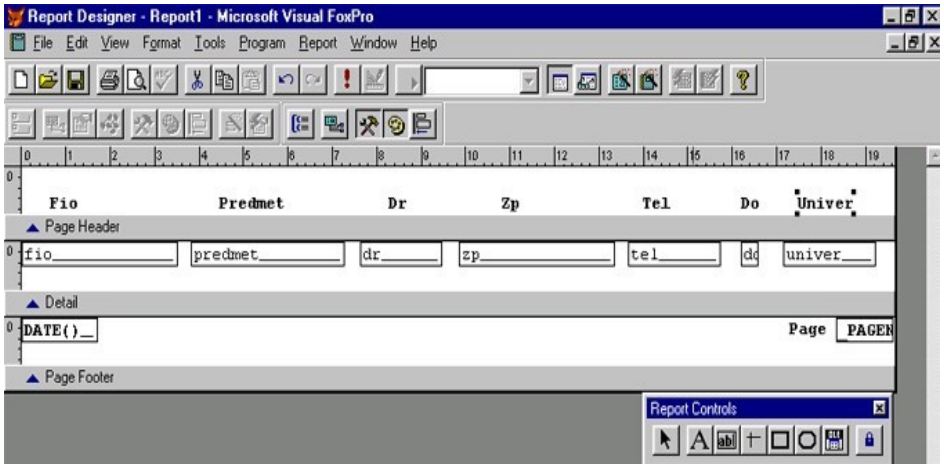


Рисунок 1 – Окно конструктора отчетов

Открывшееся окно *Quick Report* (Рисунок 2) позволяет выбирать расположение полей и опции:
- *Titles* - добавить заголовки полей;
- *Add alias* - добавить псевдонимы полей;
- *Add table to data environment* - добавить таблицу в среду окружения;

Field - выбрать нужные поля из таблицы(по умолчанию все).

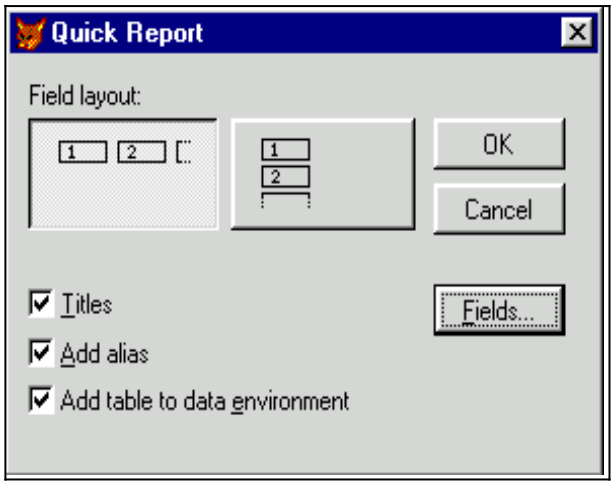


Рисунок 2 – Окно Quick Report (Стандартный отчет)

Quick Report содержит в полосе *Detail* выбранные поля из таблицы, в полосе *Page Header*-заголовки этих полей, а в полосе *Page Footer* - поле с функцией *DATE()* и поле с переменной *_PAGE( )* и словом *Page* (номер страницы). Этот отчет можно модифицировать. Для добавления в отчет суммарного итога необходимо использовать команду *Title/Summary* из меню *Report*. Опции окна *Title/Summary* (Рисунок 3) позволяют:

- *Title band* – печатать полосу заголовка;
- *New page* - печатать полосу заголовка на каждой странице;
- *Summary band* - печатать полосу суммарного итога;
- *New page* - печатать полосу суммарного итога на каждой странице.

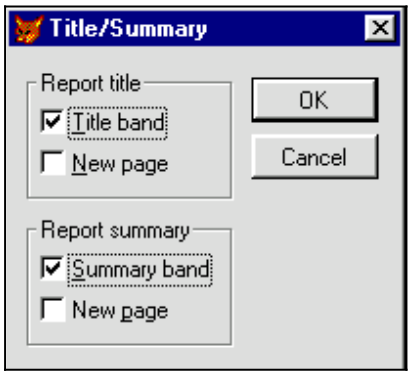
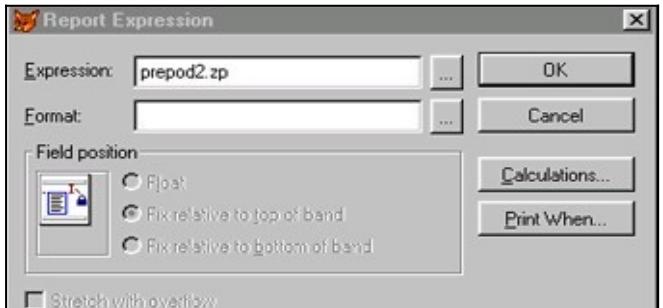


Рисунок 3 - Окно Title/Summary (Заголовок/Суммарный итог)

После этого в *окне Конструктора Отчетов* появляются две полосы:

- Title – здесь с помощью кнопки A пишется заголовок;
- Summary – - здесь с помощью кнопки abl выбирается то поле, по которому будет подводиться итог. В появившемся окне *Report Expression* (Рисунок 4) нужно выбрать необходимое поле суммирования и с помощью кнопки *Calculations…* , в



открывшемся окне *Calculate Field* (Рисунок 5) назначить необходимую стандартную функцию.

Рисунок 4- Окно Report Expression(Выражение отчета)

Окно Report Expression (Рисунок 4) позволяет:

− Expression (Выражение) – определить выражение, результат вычисления которого будет выводиться в данное поле;
− Format (Формат) - задать формат отображения данных в поле;
− Field position (Положение поля) – задать расположение поля в полосе.

Используя опции окна *Calculate Field* (Вычисляемое поле), открываемого при нажатии кнопки *Calculations* (Вычисления) (Рисунок 4) в отчет можно поместить статистические данные, размещенные в полях данных (Рисунок 5).

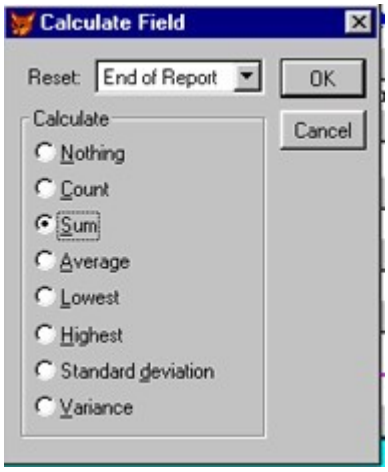


Рисунок 5 - Окно Calculate Field(Вычисляемое поле)

Для вывода упорядоченных данных можно воспользоваться командой *Data Grouping* в меню *Report* или в контекстно-зависимом меню, щелкнув правой кнопкой мыши на полосе отчета.

Чтобы выводить информацию в упорядоченном виде, нужно предварительно проиндексировать или отсортировать таблицу по соответствующему полю. Можно осуществить вложенную группировку до 128 уровней. В открывшемся окне *Data Grouping* (Рисунок 6) нужно выбрать параметры поля группировки.

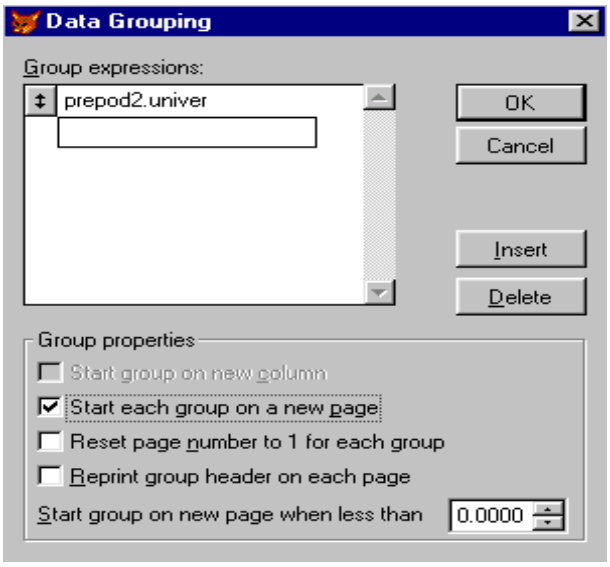


Рисунок 6 – Окно Data Grouping (Группировка данных)

Область *Group expressions* (Выражение группировки) *Data Grouping* позволяет задать выражение, по которому будут группироваться данные.
Область *Group properties* (Свойства группировки) окна *Data Grouping* позволяет задать параметры группировки:
- *Start each group on new page* – начинать каждую группу на новой странице;
- *Reset page number to 1 for each group* – сбросить номер страницы к 1 для каждой группы;
- *Reprint group header on each page* – печатать заголовок группы на каждой странице;
- *Start group on new page when less than* – печатать группу на новой странице, если под заголовком группы остается расстояние меньше указанного в данном поле.

Кнопки *Insert/Delete* – позволяют вставить/удалить поля группировки.
После задания всех параметров появляются две полосы: *Group Header* – здесь задается заголовок каждой группы и/или имя поля, по которому группируются данные *Group Footer* – здесь задается итог по каждой группе и/или имя поля, по которому группируются данные. После того как отчет сформирован можно отправить его на предварительный просмотр командой *Preview* меню *View* Для просмотра многостраничного отчета, установления масштаба, вывода на принтер, выхода в режим Конструктора служит панель предварительного просмотра (Рисунок 7). Также для печати отчета на принтере можно воспользоваться командой *Run Report* из меню *Report*.

Рисунок 7-Панель предварительного просмотра

ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ
Лабораторная работа выполняется в среде реляционной СУБД Visual FoxPro. Для выполнения лабораторной работы необходимо:
1) Запустить СУБД Visual FoxPro;
2) Повторить основной материал по технологии создания отчета (см. список литературы, конспект лекций, справочную систему Visual FoxPro);
3) Изучить описание данной лабораторной работы, выполняя все указанные действия на компьютере в среде СУБД Visual FoxPro;
4) Проверить в системе наличие таблиц данных, которые будут использованы при создании отчетов;
5) Создать отчет в свободной форме (в виде письма) При создании отчета использовать стандартные функции *xBase*. Например: alltrim(city)+' '+alltrim(address). Данные о получателе письма взять из таблицы, каждое письмо на новой странице;
6) Создать табличный отчет с заголовком и итогами, используя команду Title/Summary. В поле Summary разместить объекты TextBox, для вывода суммы значения числового поля и среднего значения числового поля, наименьшего и наибольшего общего значений полей таблицы, общего числа записей;
7) Создать отчет с вычисляемым полем, используя объект TextBox, для которого задается выражение, вычисляющее значение функции, например:
Price*Count или Price1- Price2 (Price, Price1, Price2, Count – названия полей таблицы);
8) Создать многоколоночный отчет с двумя колонками: создается образец отчета, затем выполняется команда *PageSetup* из меню *File*, в появившемся диалоговом окне *Page Setup* в области *Columns* (Колонки) определяются размеры колонок и их

количество *Number=2* ;

9) Оформить отчет по работе.

Контрольные вопросы:

4) Перечислить операции реляционной алгебры.
5) Пояснить принцип реляционного исчисления с переменными-кортежами.
6) Пояснить принцип реляционного исчисления с переменными на доменах.
7) Пояснить понятие функциональных зависимостей.
8) Пояснить правила вывода функциональных зависимостей.
9) Пояснить понятие избыточных функциональных зависимостей.

Лабораторная работа 3
Тема 3. Подходы к организации баз данных
Базисные средства манипулирования реляционными данными

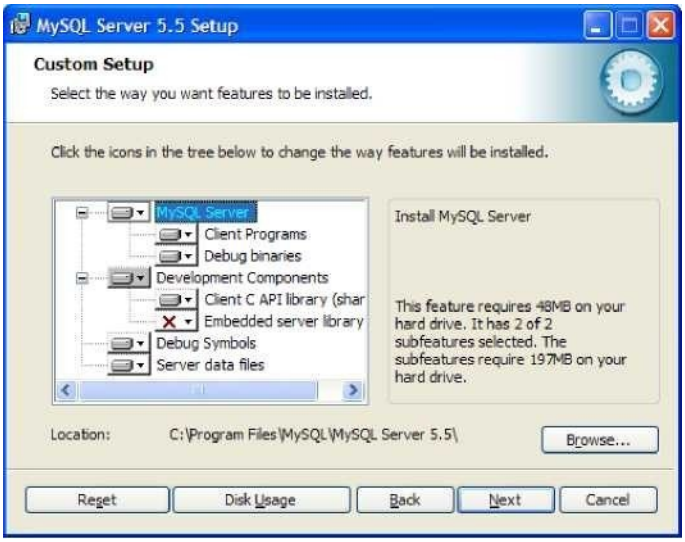
Установка и настройка MySQL Server

ii? MySQL Server 5.5 Setup f_~| П |Jx

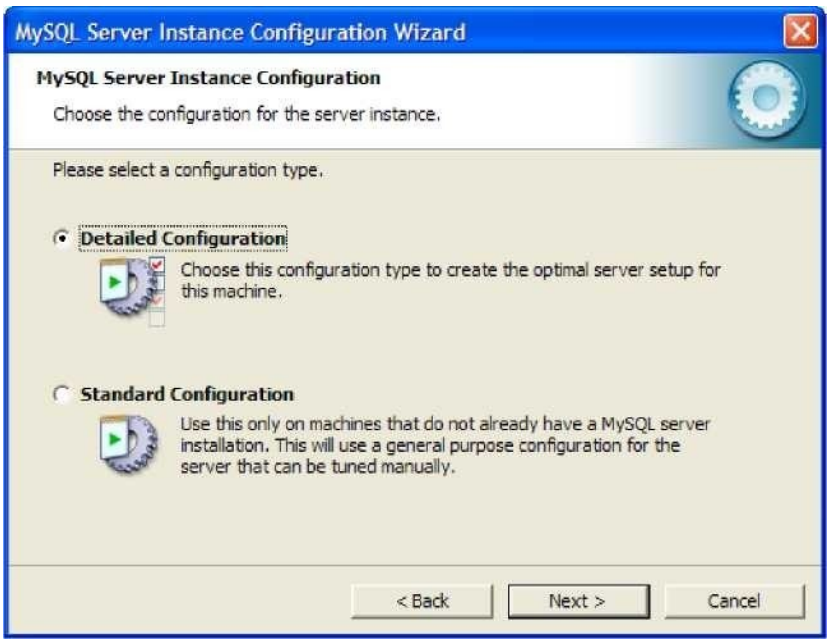
Для установки программы необходимо запустить инсталляционный файл программы.

Choose the setup type that best suits your needs

Нажмите в данном окне выборочную установку компонентов "Custom"



Здесь вы можете выбрать дополнительные компоненты и сменить установочную директорию программы.



Теперь приступим к настройке MySQL сервера.
Выбираем детализированную настройку - "Detailed Configuration".

MySQL Server Instance Configuration Wizard

Choose the configuration for the server instance.
:r type, This will influence Plecmemory, disk and CPU usage.
*(** I Developer Machine
1 d ^ˡˢˡˢ ª development machine, and many other applications will be I run on it. MySQL Server should only

use a minimal amount of memory.

C" Server Machine
d Several server applications will be running on this machine. Choose this option for web/application servers. MySQL will have medium memory usage,

C Dedicated MySQL Server Machine
d This machine is dedicated to run the MySQL Database Server. No other servers, such as a web or mail server, will be run. MySQL will utilize up to all available memory.

Отмечаем пункт "Developer Machine

Выбрав пункт "Multifunctional Database", вы сможете работать как с таблицами типа InnoDB (с возможностью использования транзакций), так и с высокоскоростной MyISAM (как правило для веб-разработок используется именно этот тип таблиц).
Далее необходимо осуществить выбор диска и директории для хранения таблиц типа InnoDB.

В данном диалоговом окне выбирается максимально возможное количество подключений к серверу MySQL.

При выборе "Decision Support (DSS)/OLAP", максимальное количество подключений будет ограничено двадцатью, чего более чем достаточно при установке сервера на домашнем компьютере и отсутствии большого количества одновременных подключений.

Отметив "Enable TCP/IP Networking" мы включаем поддержку TCP/IP соединений и

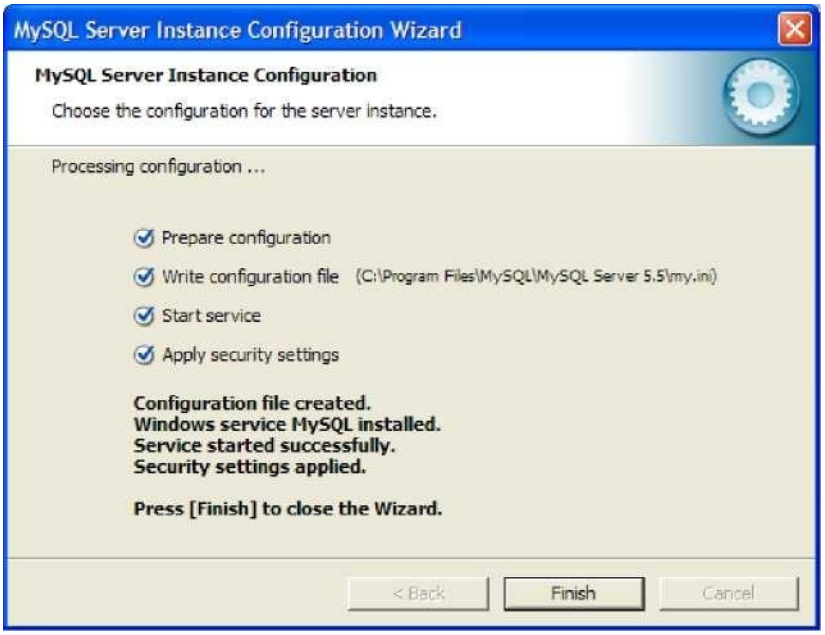
выбираем порт, через который они будут осуществляться. Стандартным для сервера MySQL является порт 3306. Отметив "Enable Strict Mode", мы задаем режим строгого соответствия стандарту SQL (данную опцию рекомендуется оставлять включенной).

Обратите внимание на выставление настроек данного окна. Отметив "Manual Selected Default Character Set / Collation" и выбрав из ниспадающего меню "cp1251" определяем,

что изначально для таблиц будет использоваться кодировка Cyrillic Windows (cp1251), что означает корректную работу с русским языком в данной кодировке.

Если отметить "Install As Windows Service", сервер будет запускаться в виде сервиса, что является рекомендуемым способом запуска. Ниже, в ниспадающем списке, задается имя сервиса. Далее, уберите галочку рядом с "Launch the MySQL Server automatically" - мы будем запускать сервер вручную. Также поставьте галочку рядом с "Include Bin Directory in Windows PATH" - это позволит установить видимость директории "bin", для командной строки.

Установите пароль пользователя "root". Не оставляйте поле пустым, это убережёт вас от возможных неприятностей в дальнейшем.



В данном окне обратите внимание на строку "Write configuration file", которая указывает на месторасположение конфигурационного файла MySQL - "my.ini", далее, его необходимо будет немного отредактировать.

Откройте для редактирования файл "my.ini".
В раздел [client], после строки: port=3306 добавьте строку определяющую каталог содержащий файлы описания кодировок:
character-sets-dir="C:/Program Files/MySQL/MySQL Server
5.5 /share/charsets"
В раздел [mysqld], после строки: port=3306 добавьте следующие две строки, первая из которых вам уже известна, вторая - устанавливает кодировку в которой данные передаются MySQL:
character-sets-dir="C:/Program Files/MySQL/MySQL Server 5.5/share/charsets" init-connect-'SET NAMES cp1251"
Далее, найдите строку: default-storage-engine-INNODB Замените изначально устанавливаемый тип таблиц на MYISAM: default-storage-engine- MYISAM.
Сохраните изменения и закройте файл "my.ini". Установка и настройка сервера MySQL - завершена.

Контрольные вопросы:

1) Пояснить, каким способом возможен запуск серверной части СУБД MySQL.
2) Дать определение привилегии и пояснить её предназначение.
3) Перечислить основные утилиты, входящие в состав СУБД, какие функции они выполняют.
4) Перечислить принципы использования кортежных переменных
5) Дать понятие минимального покрытия и декомпозиции отношений.
6) Перечислить принципы использования переменных в доменах

Задания творческого уровня:

7) Запустите сервер MySQL. Зарегистрируйте своего пользователя в консольномприложении, задайте ему права.
8) С помощью утилиты Mysqlshow выполните команду на просмотр структуры и состав таблиц базы Mysql.
9) С помощью утилиты Mysqldump получите полный дамп базы Mysql (данные и таблицы), а также отдельные дампы таблиц и данных.

Лабораторная работа 4
Тема 4. Общие понятия реляционного подхода к организации БД
Управление параллельностью работы транзакций

Содержание работы и методические указания к ее выполнению

1. Ознакомиться с возможностями работы клиентского приложения MySQL.
2. Изучить набор команд языка SQL, связанный с созданием базы данных, созданием, модификацией структуры таблиц и их удалением, вставкой, модификацией и удалением записей таблиц.

Функция	**Описание**
CREATE DATABASE DB_NAME	создание базы данных
USE DATABASE	выбор существующей базы данных
CLOSE DATABASE	закрытие файлов текущей базы данных
DROP DATABASE	удаление базы данных
CREATE TABLE	создание таблицы базы данных
ALTER TABLE	модификация структуры базы данных
DROP TABLE	удаление таблицы базы данных
INSERT	добавление одной или нескольких строк в таблицу
DELETE	удаление одной или нескольких строк из таблицы
UPDATE	модификация одной или нескольких строк таблицы
LOAD DATA INFILE	загрузка данных в таблицы из файла

3. Создать базу данных.

Создание базы данных в MySQL производится с помощью утилиты mysqladmin. Изначально существует только БД mysql для администратора и БД test, в которую может войти любой пользователь и которая по умолчанию пуста. Приведенный ниже пример иллюстрирует создание базы данных.

```
Mysql/bin>mysqladmin -u root -p create data_name
Enter password:******

Database "data_name" created.
```

Где data_name – имя создаваемой БД. Проверить, что БД создана можно ранее рассмотренной командой ***Show databases*** или утилитой **mysqlshow.**
По умолчанию, **root** имеет доступ ко всем базам данных и таблицам. Перейти в созданную базу данных можно, используя команду **mysql**. Use database

Mysql/bin>***mysql -u root -p data1***

Enter password:******

Или, находясь в другой базе данных, например в mysql ввести команду:

mysql>***use data1***

Создать базу данных можно непосредственно находясь в клиентском приложении MySQL, вводом команды:

CREATE DATABASE Base_name

Где **Base_name** имя создаваемой базы данных. В созданной базе можно создавать таблицы и вводить информацию. Указанные операции можно выполнить, используя специализированное программное обеспечение, например MySQL-Front, запуск которого осуществляется из меню ПУСК/ПРОГРАММЫ (см. рис 1, 2).

Необходимо указать:

- Имя;
- Хост;
- Пароль;
- Порт;
- Имя БД (при необходимости).

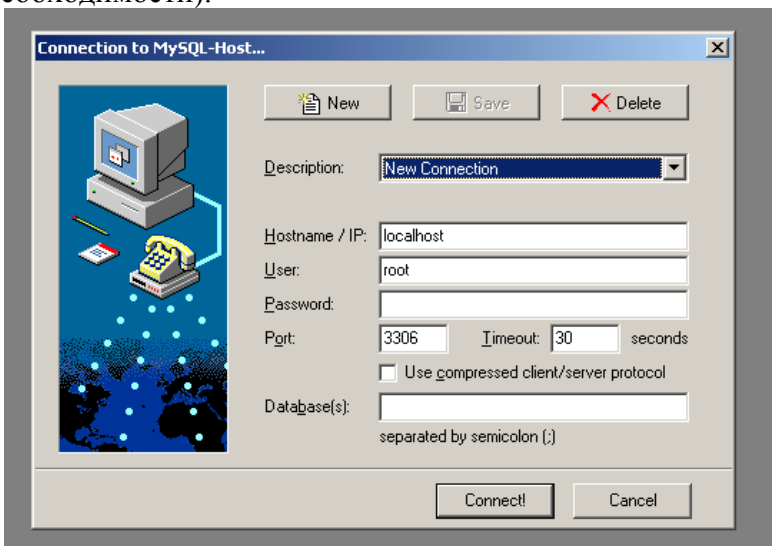
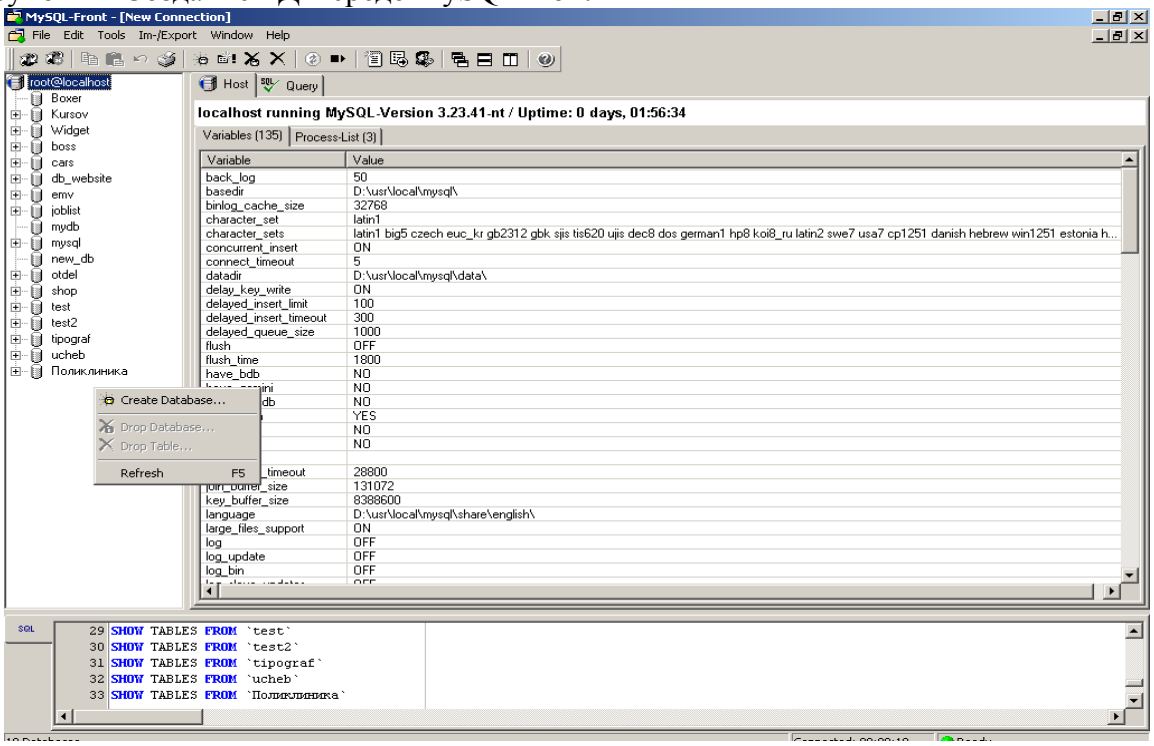


Рисунок 1 - Запуск MySQL-front

Рисунок 2 - Создание БД в среде MySQL-front



После задания активной БД можно с помощью средств, предоставляемых программой изменять структуру БД, вводить данные, задавать ключевые поля. Помимо этого, можно в специально отведенном окне напрямую вводить инструкции, используя синтаксис языка SQL, как показано на рисунке:

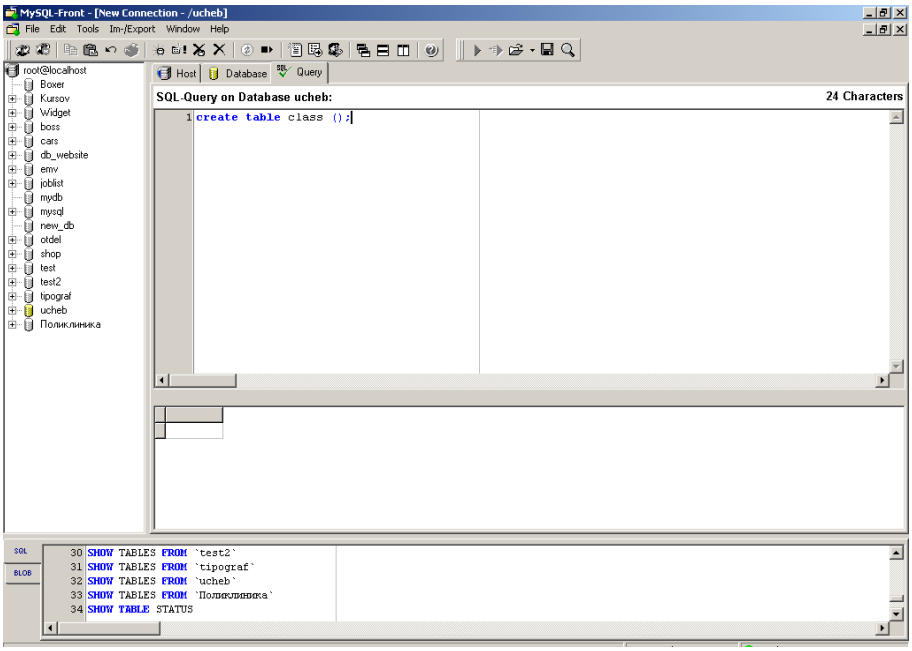


Рисунок 3 - Использование синтаксиса SQL

4. Средствами языка SQL необходимо создать четыре таблицы в базе данных, используя команду

CREATE TABLE, синтаксис которой приведен в приложении. Для таблицы J:

CREATE TABLE j (

Jnum varchar(6) NOT NULL default '', Jnam varchar(20) default NULL,

Ci varchar(20) default NULL, PRIMARY KEY (Jnum)

) TYPE=MyISAM;

Значками /* */ - выделяются комментарии в тексте запроса.

При создании таблиц выполнить такую реализацию, чтобы она отражала структуру таблиц, указанную ниже (таблице S, P, J, SPJ) и должны быть наложены следующие ограничения:

- поля номер_поставщика, номер_детали, номер_изделия во всех таблицах имеет символьный тип и длину 6 (**varchar**(6));
- поля рейтинг, вес и количество имеют целочисленный тип (**integer)**;
- поля фамилия, город (поставщика, детали или изделия), название (детали или изделия) имеют символьный тип и длину 20 (**varchar**(20));
- ни для одного поля не предусматривается использование индексов;

-для всех полей допускаются значения NULL и значения-дубликаты, кроме полей первичного и внешнего ключей.

После создания пустых таблиц их необходимо наполнить данными. Вводить данные в нее можно несколькими способами:

а)Вручную, используя команду **insert into**;

Пример ввода данных вручную (команда INSERT):

mysql>***insert into J (Jnum, Jnam, Ci)values ('J1','Жесткий диск','Париж')***;

или

mysql>***insert into J values ('J1','Жесткий диск','Париж');***

//т.е в случае если вы вставляете данные во все поля таблицы то их перечислять не обязательно.

Таким образом SQL инструкция имеет следующий вид

INSERT INTO table_name (id, name) VALUES ('id_value', 'name_value');

Записать и выполнить совокупность запросов для занесения нижеприведенных данных в созданные таблицы

insert into имя_таблицы [(поле [,поле]...)] values (константа [,константа]...)

б)Загрузить данные из текстового файла, что является более предпочтительным, особенно если нужно ввести несколько тысяч записей.

Синтаксис команды LOAD DATA INFILE.

DATA [LOW_PRIORITY] [LOCAL] INFILE 'file_name.txt' [REPLACE | IGNORE]
INTO TABLE tbl_name [FIELDS [TERMINATED BY 't']

[OPTIONALLY] ENCLOSED BY ''] [ESCAPED BY '']]

[LINES TERMINATED BY 'n']
[IGNORE number LINES] [(col_name,...)]

Пример:

LOAD DATA LOCAL INFILE '/MyDocs/categories.txt' REPLACE INTO TABLE category FIELDS TERMINATED BY ';' OPTIONALLY ENCLOSED BY '\"' LINES TERMINATED BY '\n'

В данном случае файл ***categories.txt*** находится на машине под управлением MS Windows, в каталоге ***C:\MyDocs***.

Обратите внимание на UNIX стиль написания пути. Слово
REPLACE
В SQL запросе означает, что необходимо замещать записи с совпадающими значениями ключей.

INTO TABLE[1]
указывает имя таблицы, куда будут импортированы данные.
FIELDS TERMINATED BY ';' [2]
указывает разделители полей, порядок полей должен быть таким же, как и в таблице назначения,

OPTIONALLY ENCLOSED BY '\"'
указывает, что поля VARCHAR взяты в двойные кавычки, и LINES TERMINATED BY '\r' [3]

в) Использовать утилиту mysqlimport также для загрузки данных из текстового файла. Эти и другие операции можно выполнить также и в программе MySQL-Front.

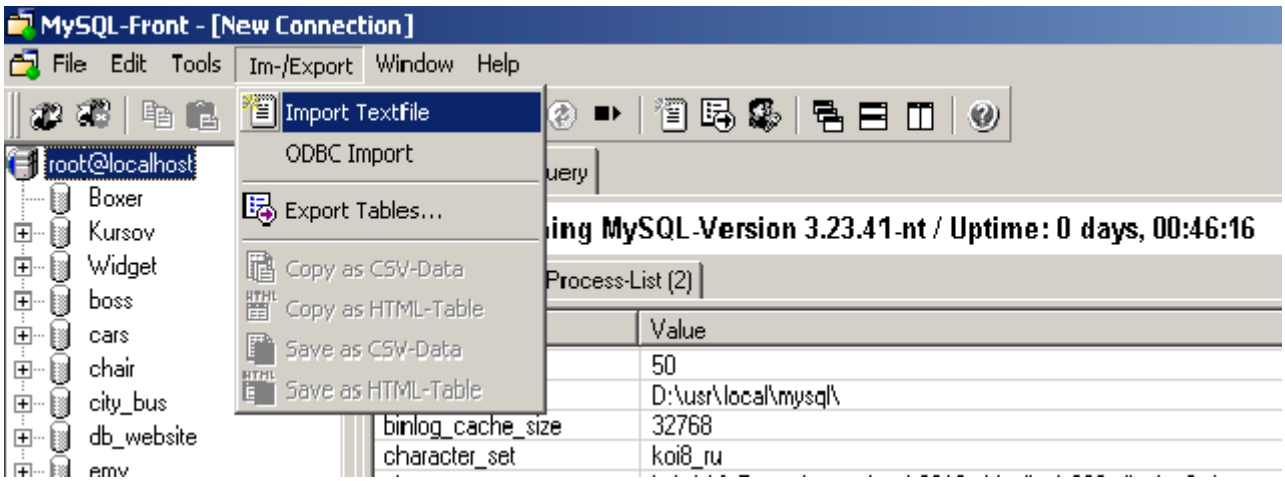


Рисунок 4 - Использование программы MySQL-front для заполнения таблиц данными из файла

Таблица поставщиков (S)

Номер поставщика	**Фамилия**	**Рейтинг**	**Город**
S1	Смит	20	Лондон

S2	Джонс	10	Париж
S3	Блейк	30	Париж
S4	Кларк	20	Лондон
S5	Адамс	30	Афины

Таблица деталей (P)

Номер детали	**Название**	**Цвет**	**Вес**	**Город**
P1	Гайка	Красный	12	Лондон
P2	Болт	Зеленый	17	Париж
P3	Винт	Голубой	17	Рим
P4	Винт	Красный	14	Лондон
P5	Кулачок	Голубой	12	Париж
P6	Блюм	Красный	19	Лондон

Таблица изделий (J)

Номер изделия	**Название**	**Город**
J1	Жесткий диск	Париж
J2	Перфоратор	Рим
J3	Считыватель	Афины
J4	Принтер	Афины
J5	Флоппи-диск	Лондон
J6	Терминал	Осло
J7	Лента	Лондон

Таблица поставок (SPJ)

Номер поставщика	**Номер детали**	**Номер изделия**	**Количество**
S1	P1	J1	200
S1	P1	J4	700
S2	P3	J1	400
S2	P3	J2	200
S2	P3	J3	200

S2	P3	J4	500
S2	P3	J5	600
S2	P3	J6	400
S2	P3	J7	800
S2	P5	J2	100
S3	P3	J1	200
S3	P4	J2	500
S4	P6	J3	300
S4	P6	J7	300
S5	P2	J2	200
S5	P2	J4	100
S5	P5	J5	500
S5	P5	J7	100
S5	P6	J2	200
S5	P1	J4	100
S5	P3	J4	200
S5	P4	J4	800
S5	P5	J4	400
S5	P6	J4	500

Убедиться в успешности выполненных действий. При необходимости исправить ошибки. Для ускорения процесса ввода данных рекомендуется воспользоваться командой LOAD DATA (синтаксис см. в приложении), предварительно скопировав содержимое перечисленных таблиц сначала в Excel, а оттуда в текстовые файлы. Такой порядок необходим, для того, чтобы текстовый файл был с табуляцией.

5. Выполнить модификацию структуры таблицы SPJ, добавив в SPJ поле с датой поставки. Убедиться в успешности выполненных действий. При необходимости исправить ошибки (команда Alter table).
6. Уничтожить созданные таблицы, предварительно сохранив инструкции для восстановления структуры БД и информационного наполнения, используя средства работы СУБД[4]. Убедиться в успешности выполненных действий.
7. Выполнить необходимые действия, написав и выполнив соответствующие запросы для модификации таблиц, чтобы структура соответствовала концептуальной модели учебной базы данных (рисунок 5). Убедиться в успешности выполненных действий. При необходимости исправить ошибки.

Проверить результат заполнения таблиц, написав и выполнив простейший запрос:

select * from имя_таблицы

При наличии ошибок выполнить корректировку, исправив либо удалив ошибочные строки таблиц

Контрольные вопросы

1. В каких режимах возможно создание базы данных?
2. Какие типы данных допустимы при создании таблицы?
3. Как выполнить создание таблицы средствами СУБД?
4. Как выполнить создание таблицы средствами языка SQL?
5. Как разделяются операторы SQL в случае нескольких операторов в запросе?
6. Каким образом выполнить простейшие операции вставки строк данных в таблицу средствами SQL?
7. Каким образом выполнить простейшие операции модификации строк таблицы средствами SQL?
8. Каким образом выполнить просмотр таблицы?
9. Как получить информацию о структуре таблицы в рамках СУБД MySQL?

Лабораторная работа 5
Тема 5. Базисные средства манипулирования реляционными данными
Методы сериализации транзакций

Предоставление доступа к базам данных

СУБД MySQL использует специальную базу данных для предоставления прав доступа к своим базам данных. Эти права могут базироваться на именах серверов и/или пользователей и предоставляться для одной или нескольких баз данных

Пользовательские учетные записи могут быть снабжены паролями. При обращении к базе данных, пароль шифруется. Поэтому он не может быть перехвачен и использован посторонним (это мнение автора СУБД...).

СУБД MySQL имеет три таблицы, а именно: База данных: mysql Таблица: db

База данных: mysql Таблица: db					
Поле	**Тип**	**Null**	**Ключ**	**Умолчание**	**Extra**
Хост	char(60)		PRI		
Db	char(32)		PRI		
Пользователь	char(16)		PRI		
Select_priv	char(1)			N	
Insert_priv	char(1)			N	
Update_priv	char(1)			N	
Delete_priv	char(1)			N	
Create_priv	char(1)			N	
Drop_priv	char(1)			N	

База данных: mysql Таблица: host

Поле	**Тип**	**Null**	**Ключ**	**Умолчание**	**Extra**
Хост	char(60)		PRI		
Db	char(32)		PRI		
Select_priv	char(1)			N	
Insert_priv	char(1)			N	
Update_priv	char(1)			N	
Delete_priv	char(1)			N	
Create_priv	char(1)			N	
Drop_priv	char(1)			N	

База данных: mysql Таблица: user

База данных: mysql Таблица: user

Поле	Тип	Null	Key	Умолчание	Extra
Хост	char(60)		PRI		
Пользователь	char(16)		PRI		
Пароль	char(8)				
Select_priv	char(1)			N	
Insert_priv	char(1)			N	
Update_priv	char(1)			N	
Delete_priv	char(1)			N	
Create_priv	char(1)			N	
Drop_priv	char(1)			N	
Reload_priv	char(1)			N	
Shutdown_priv	char(1)			N	
Process_priv	char(1)			N	
File_priv	char(1)			N	

Текущей базой данных называется база данных, открытая с помощью операторов use Database или с помощью утилиты mysqladmin. Любая другая база данных называется внешней. Для ссылки на таблицу во внешней базе данных необходимо указать имя этой базы данных как часть имени таблицы, например, salesdb:contracts, где salesdb - имя внешней базы данных, contracts - имя таблицы. К имени базы данных можно добавить имя сервера, т.е. сетевой машины, где запущен еще один сервер баз данных mysql, и таким образом в случае распределенной базы данных обращение к таблице contracts базы данных salesdb, размещенной на сервере central, будет выглядеть следующим образом: salesdb@central:contracts.

В программе MYSQL-FRONT также существует механизм, обеспечивающий наделение пользователей определенными правами (см. Рисунок 6).

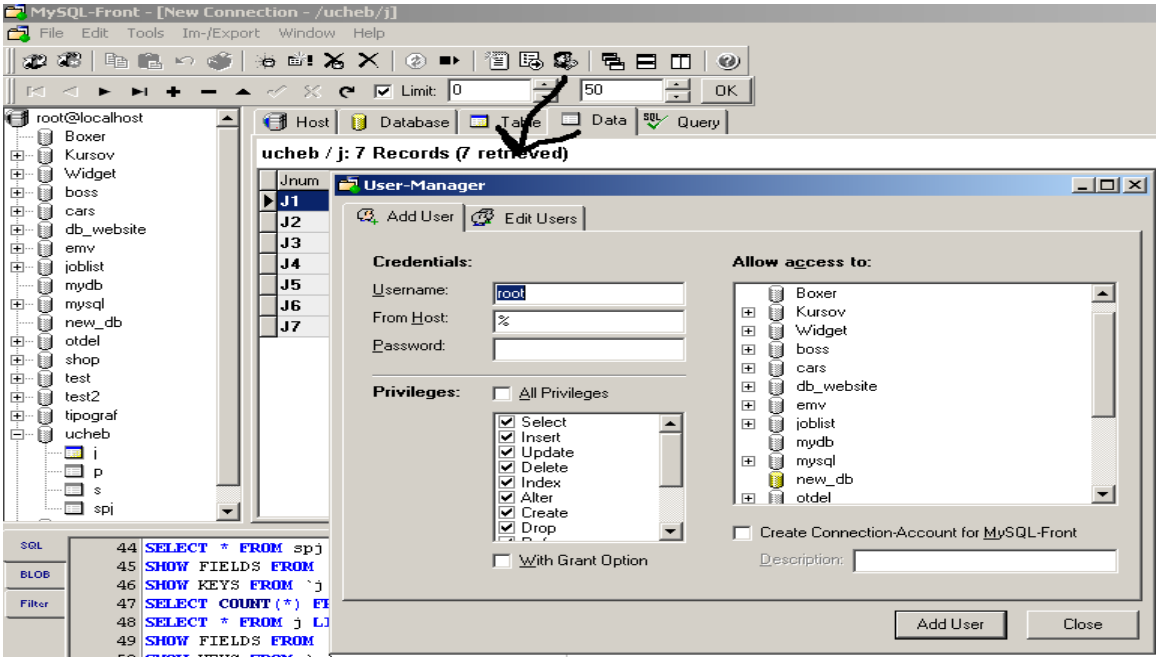


Рисунок 6 - Редактирование прав пользователя

При наличии сетевого соединения с сервером выполните приведенную последовательность выполнения лабораторной работы, при отсутствии соединения создать еще 1го пользователя в вашей БД, наделив его привилегиями лишь для просмотра таблиц, в этом случае все приведенные операции осуществлять от имени созданного пользователя.

Задание (общее):

1. Убедиться, что в таблице поставщиков S имеются строки с Вашими фамилиями (задание выполнялось в третьей лабораторной работе).
2. Откорректировать экранную форму, созданную в третьей лабораторной работе для работы с таблицей поставок SPJ, обеспечив возможность ввода и модификации данных. Занести произвольным образом несколько строк (5-10 строк) о поставках, связанных с Вашими фамилиями.
3. Выполнить два запроса к базе данных, согласно номера Вашего варианта. При выполнении запроса данные должны выбираться из таблиц Вашей базы данных.
4. Повторить задание п.3 с той разницей, что сведения о номенклатуре деталей и изделий (P и J) должна браться из собственной базы данных, а сведения о поставщиках и поставках (S и SPJ)

должны браться из базы данных соседней бригады. Предварительно необходимо узнать имя этой базы данных. Убедитесь в невозможности выполнения задания.

5. Обеспечьте, чтобы владелец внешней используемой Вами базы данных предоставил Вам полномочия на просмотр используемых Вами таблиц в его базе данных, дав соответственно ему такие же полномочия для выполнения аналогичных действий.
6. Повторите задание п.4. Сравните результаты с результатами, полученными в п.3.
7. Сделайте попытку изменить информацию о поставщиках-владельцах базы данных (город, рейтинг и т.д.) в таблице S внешней базы данных. Убедитесь в невозможности выполнения задания.
8. Обеспечьте, чтобы владелец внешней используемой Вами базы данных предоставил Вам полномочия на модификацию данных из используемых Вами таблиц в

его базе данных, дав соответственно ему такие же полномочия для выполнения аналогичных действий.

9. Повторите задание п.7. Проверьте успешность выполнения действий.

10. Дождавшись, когда владелец внешней базы данных закончит выполнение п.9, сделайте попытку удалить из таблицы S используемой Вами внешней базы данных поставщиков с именами, принадлежащими владельцам базы данных, и связанные с ними поставки из таблицы SPJ. Убедитесь в невозможности выполнения задания.

11. Обеспечьте, чтобы владелец используемой Вами внешней базы данных предоставил Вам полномочия на удаление из используемых Вами таблиц в его базе данных, дав соответственно ему такие же полномочия для выполнения аналогичных действий.

12. Повторите задание п.10. Проверьте успешность выполнения действий.

13. Отнимите предоставленные Вами права на пользование Вашей базой данных.

Контрольные вопросы

1. Кто является владельцем базы данных?
2. Какими правами обладают другие пользователи по отношению к Вашей базе данных?
3. Какими правами обладает администратор базы данных по отношению к Вашей базе данных?
4. Каким образом предоставляются права на пользование базой данных и отдельными ее таблицами?
5. Каким образом изымаются права на пользование базой данных и отдельными ее таблицами?
6. Что такое внешняя база данных?
7. Как идентифицируется таблица внешней базы данных?
8. Как идентифицируется таблица внешней распределенной базы данных?

Лабораторная работа 6
Тема 6. Проектирование реляционных БД
Язык SQL. Функции и основные возможности

Содержание работы и методические указания к ее выполнению

С помощью динамического SQL программа-клиент выполняет программное формирование оператора SQL для его последующего исполнения, делая это в три этапа:

− программа собирает текст оператора SQL в виде символьной строки, хранящейся в программной переменной; в общем случае это может быть не один, а несколько операторов SQL, разделенных точкой с запятой;

− программа выполняет оператор *Prepare*, который обращается к серверу баз данных для анализа текста оператора и подготовки его к выполнению;

− программа использует оператор Execute для выполнения подготовленного оператора.

Динамический оператор SQL по форме напоминает любой другой оператор SQL, записанный в программе, с тем ограничением, что он не может содержать имена главных переменных. Поэтому

− в динамический оператор *Select* нельзя включать спецификатор *Into*;

− в любом месте, где главная переменная могла бы появиться в выражении, ей соответствует знак вопроса.

Оператор *Prepare* создает структуру данных, имеющую имя и отображающую строку символов с текстом оператора SQL.

Подготовленный по оператору *Prepare* динамический оператор (группу операторов) можно многократно выполнять. С помощью оператора *Execute* выполняются операторы, отличные от операторов Select, а также операторы Select, которые возвращают в качестве результата одну строку. Если оператор *Select* возвращает более одной строки, динамический оператор *Select* выполняется не с помощью оператора *Execute*, а подключается к курсору и в дальнейшем используется обычным образом с помощью курсорных средств. В обоих случаях при выполнении динамического оператора с помощью спецификатора *Using* ему передаются главные переменные, участвующие в выражениях и принимающие возвращаемые значения и,

по сути, играющие роль фактических параметров. Как ограничение, следует отметить, что знак вопроса нельзя использовать вместо идентификаторов SQL, таких как имя базы данных, таблицы или столбца: эти идентификаторы должны указываться в тексте оператора при его подготовке, возможно, как полученные из пользовательского ввода.

Последовательность выполнения лабораторной работы:

1. Изучить синтаксис и правила использования операторов *Prepare, Execute* (см. Приложение 2), а также особенности работы с курсором при выполнении динамического оператора SQL.
2. Разработать и отладить набор ESQL/C-программ, решающих задачи из соответствующего варианта заданий. Результатом работы программ является одна или несколько строк, которые подлежат выводу на экран с соответствующими пояснительными заголовками.

Требования к разрабатываемой программе

Разрабатываемые ESQL/C-программы должна удовлетворять следующим требованиям:

− обеспечивать необходимую обработку ошибок;

− использовать аппарат транзакций;

− все используемые в программах операторы SQL, включая операторы, реализующие аппарат транзакций, должны быть динамически подготовлены;
− необходимые параметры, определяющие условия задачи, вводятся с клавиатуры и передаются в строку с текстом динамического оператора SQL;
− все параметры, специфицирующие выполняемые действия, должны передаваться через главные переменные; такими параметрами в условиях задач являются название города, название детали, номер поставщика и т.д.;
− должен быть предусмотрен вывод сообщений обо всех шагах выполнения программы, в том числе и о возможных ошибках;
− программа должна быть достаточно документирована.

Варианты заданий

Вариант 1.

1. Выдать полную информацию о поставщике, имеющим максимальный рейтинг (с использованием оператора *Execute*).

2. Получить номера изделий, для которых детали полностью поставляет поставщик с указанным номером (параметр - номер поставщика (S1)).
3. Выдать номера и фамилии поставщиков, поставляющих детали для какого-либо изделия с деталью, номер которой указывается, в количестве, большем, чем средний объем поставок данной детали для этого изделия (параметр - номер детали (P1)).

Вариант 2.

1. Выдать полную информацию об изделии, изготавливаемом в городе, в котором проживает поставщик с максимальным рейтингом (с использованием оператора *Execute*).
2. Получить общее количество деталей с указанным номером, поставляемых некоторым поставщиком (параметры - номер детали (P1), номер поставщика (S1)).
3. Выдать номера изделий, использующих только детали, поставляемые некоторым поставщиком (параметр - номер поставщика (S1)).

Вариант 3.

1. Выдать полную информацию о детали, имеющей максимальный вес (с использованием оператора *Execute*).
2. Получить общее число изделий, для которых поставляет детали поставщик с указанным номером (параметр - номер поставщика (S1)).
3. Выдать номера изделий, детали для которых поставляет каждый поставщик, поставляющий какую-либо деталь указанного цвета (параметр - цвет детали (красный)).

Вариант 4.

1. Выдать общий объем поставок деталей красного цвета (с использованием оператора *Execute*).
2. Получить полный список деталей для всех изделий, изготавливаемых в некотором городе (параметр - название города (Лондон)).
3. Выдать номера деталей, поставляемых каким-либо поставщиком из указанного города (параметр - название города (Лондон)).

Вариант 5.

1. Выдать полную информацию об изделии, имеющем максимальный объем поставок деталей (с использованием оператора *Execute*).
2. Получить список всех поставок, в которых количество деталей находится в некотором диапазоне (параметры - границы диапазона (от 300 до 750)) .
3. Выдать номера и названия деталей, поставляемых для какого-либо изделия из указанного города (параметр - название города (Лондон)).

Вариант 6.

1. Выдать общий объем поставок деталей для изделия J2 (с использованием оператора *Execute*).

2. Получить цвета деталей, поставляемых некоторым поставщиком (параметр - номер поставщика (S1)).
3. Выдать номера и фамилии поставщиков, поставляющих некоторую деталь для какого-либо изделия в количестве, большем среднего объема поставок данной детали для этого изделия (параметр - номер детали (P1)).

Вариант 7.

1. Выдать общий объем поставок деталей для изделия с максимальным объемом поставок (с использованием оператора *Execute*).
2. Получить названия изделий, для которых поставляются детали некоторым поставщиком (параметр - номер поставщика (S1)).
3. Выдать номера изделий, для которых средний объем поставки некоторой детали больше максимального объема поставки любой детали для указанного изделия (параметры - номер детали (P1), номер изделия (J1)).

Контрольные вопросы

1. Каково назначение и синтаксис оператора *Prepare*?

2. Каково назначение и синтаксис оператора *Execute?*

3. Каковы особенности использования динамических операторов SQL?

4. Что такое динамические главные переменные? Для чего они используются?

5. С какими операторами связано использование динамических главных переменных?

6. Каково назначение оператора *Execute Immediate*?

Лабораторная работа 7
Структуры внешней памяти в реляционных СУБД
Язык SQL. Функции и основные возможности

Содержание работы и методические указания к ее выполнению
Для выполнения работы необходимо
– изучить базовые функции выборки данных **SQLBindCol, SQLFetch(), SQLGetData()**;

– ознакомиться с алгоритмами извлечения данных из результирующего множества с использованием средств ODBC;

– настроить среду выполнения, разработать и отладить ODBC-программу выборки данных.

В ODBC существует две функции базового уровня для выборки результатов
- **SQLBindCol()** и **SQLFetch()**. Функция **SQLBindCol()** определяет область хранения данных результирующего множества, функция **SQLFetch()** осуществляет выборку данных в области хранения.

Каждый столбец, который требуется выбрать, связывается с помощью отдельного вызова функции **SQLBindCol()**. Функция **SQLBindCol()** назначает область хранения в памяти и тип данных для столбца результирующего множества. Она определяет:

− буфер хранения для получения содержимого столбца данных в результирующем множестве.

− длину указанного буфера хранения.

− область памяти для хранения длины столбца выборки.

− преобразование типа данных.

Алгоритм программы, использующей **SQLFetch()** и **SQLBindCol()** для возвращения данных из результирующего множества предполагает выполнения следующих шагов:

1. Вызывается функция **SQLBindCol()** один раз для каждого столбца, который должен быть возвращен из результирующего множества.
2. Вызывается функция **SQLFetch()** для перемещения курсора на следующую строку и возврата данных из связанных столбцов.
3. Повторяется шаг 2 до тех пор, пока функция **SQLFetch()**не возвратит SQL_NO_DATA_FOUND. Это указывает на то, что был достигнут конец результирующего множества. Если результирующее множество является пустым, то SQL_NO_DATA_FOUND будет возвращен при первом вызове **SQLFetch()**.

RETCODE **SQLBindCol** (*hstmt, icol, fcType, rgbValue, cbValueMax, pcbValue*) HSTMT *stmt* - идентификатор оператора;

UWORD *icol* - идентификатор окружения;
SWORD *fCtype* - С-тип данных результирующего множества;
PTR *rgbValue* - указатель области хранения данных. Если rgbValue является нулевым указателем, то драйвер "отвязывает" столбец; для отвязывания всех столбцов прикладная программа вызывает **SQLFreeStmt()** с опцией SQL_UNBIND;
SDWORD *cbValueMax* - максимальная длина буфера rgbValue; для символьных данных, rgbValue должен также включать в себя место для байта нулевого окончания; SDWORD *pcbValue* - число байт, которое может возвращаться в rgbValue при вызове **SQLFetch()**.

Как только все требуемые столбцы были связаны, вызывается **SQLFetch** для выборки данных в определенной области хранения.

RETCODE **SQLFetch** (*hstmt*)

HSTMT *stmt* - идентификатор оператора.

Функция **SQLGetData()** позволяет выполнить выборку данных из столбцов, для которых область хранения не была заранее подготовлена с помощью функции **SQLBindCol()**. Функция **SQLGetData()** вызывается после вызова функции**SQLFetch()** для выборки данных из текущей строки.

Алгоритм программы, использующей **SQLFetch()** и **SQLGetData()** для извлечения данных из каждой строки результирующего множества предполагает выполнения следующих шагов:

1. Вызывается функция **SQLFetch()** для продвижения на следующую строку.

2. Вызывается функция **SQLGetData()** для каждого столбца, который должен быть возвращен из результирующего множества.

3. Повторяется шаги 1 и 2 до тех пор, пока функция **SQLFetch()** не возвратит SQL_NO_DATA_FOUND. Это указывает на то, что был достигнут конец результирующего множества. Если результирующее множество является пустым, то SQL_NO_DATA_FOUND будет возвращен при первом вызове **SQLFetch()**.

RETCODE **SQLGetData** (*hstmt, icol, fcType, rgbValue, cbValueMax, pcbValue*);

HSTMT *stmt;* - идентификатор оператора;

UWORD *icol;* - номер столбца результирующего множества, упорядоченный слева направо, начиная с 1;

SWORD *fCtype;* - С-тип данных результирующего множества;

PTR *rgbValue;* - указатель области хранения данных. Если rgbValue является нулевым указателем, то драйвер "отвязывает" столбец. Для отвязывания всех столбцов прикладная программа вызывает функцию **SQLFreeStmt()** с опцией SQL_UNBIND.

SDWORD *cbValueMax;* - максимальная длина буфера rgbValue; для символьных данных, rgbValue должен также включать в себя место для байта нулевого окончания.

SDWORD FAR* *pcbValue;* - число байт, которое может возвращаться в rgbValue при вызове **SQLGetData()**.

Замечание. Прикладная программа может использовать **SQLBindCol()** для некоторых столбцов, а **SQLGetData()** - для других столбцов в пределах той же самой строки.

Последовательность выполнения лабораторной работы

1. Убедиться в наличии и заполненности базы данных поставщиков, деталей, изделий, поставок.

2. Разработать ODBC-программу для решения задачи из соответствующего варианта с помощью функций **SQLBindCol(), SQLFetch()**.

3. Разработать ODBC-программу для решения задачи из соответствующего варианта с помощью функций **SQLFetch(), SQLGetData()**.

4. После выполнения лабораторной работы привести базу данных в исходное состояние.

Требования к разрабатываемой программе

Разрабатываемая программа должна удовлетворять следующим требованиям:

− все используемые функции ODBC должны анализироваться на корректность кода возврата;

− в программе должен быть предусмотрен вывод сообщений обо всех шагах ее выполнения, в том числе и о возможных ошибках;

− при выполнении запросов должно быть предусмотрено использование параметров; параметры варианта задания должны быть введены в ходе выполнения программы и переданы в SQL-запрос;

− при выполнении программы должна контролироваться целостность базы данных;

− программа должна быть достаточно документирована.

Варианты заданий

1. Вывести информацию о поставщиках, поставивших детали для изделий из указанного города.

2. Вывести информацию о деталях, поставки которых были осуществлены для указанного изделия.

3. Вывести информацию о поставщиках, которые осуществляли поставки деталей из заданного города в указанный период.

4. Вывести информацию о поставщиках, поставивших указанную деталь в заданный период.

5. Вывести информацию обо всех деталях, поставляемых для указанного изделия более чем одним поставщиком.

6. Вывести информацию о деталях, поставки которых были осуществлены для указанного изделия всеми поставщиками.

7. Вывести информацию об изделиях, для которых была поставлена указанная деталь.

8. Вывести информацию о поставщиках, которые осуществляли поставки деталей для указанного изделия.

Контрольные вопросы

1. Какие существуют базовые функции выборки данных? Охарактеризуйте каждую из них.

2. Как "отвязать" отдельный столбец результирующего множества?

3. Каким образом можно "отвязать" все столбцы результирующего множества?

4. Что возвращает функция **SQLFetch,** если в результирующем множестве больше не осталось данных?

5. Как определить область хранения данных?

6. Каков алгоритм выборки данных с помощью курсора?

7. Как работает функция **SQLFetch**?

Можно ли использовать **SQLFetch** для продвижения курсора в обратном направлении?

Лабораторная работа 8
Управление параллельностью работы транзакций
Безопасность SQL при прикладном программировании

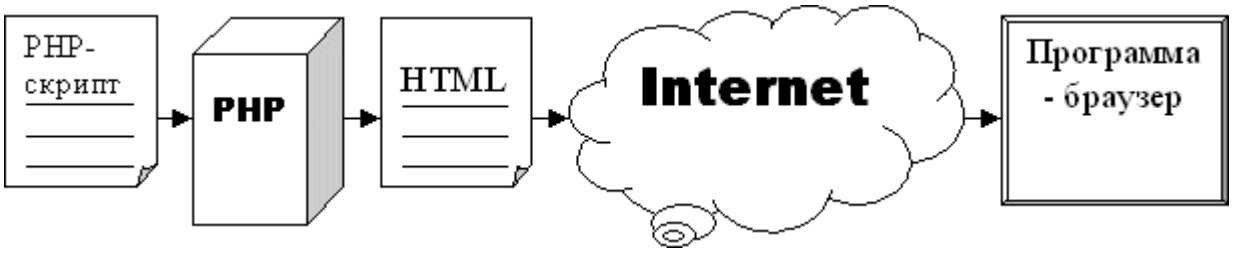
Содержание работы и методические указания к ее выполнению
Для выполнения работы необходимо
- ознакомиться с синтаксисом языка PHP;

- изучить особенности передачи значений переменных HTML-формы в переменные PHP;

- ознакомиться с набором функций для общения с СУБД Informix;

- с использованием средств языка PHP разработать и отладить программу доступа к базе данных.

Язык PHP - это действующий на стороне сервера встраиваемый в HTML язык, имеющий синтаксис, близкий к языку Си. Язык PHP дает возможность вставлять в файлы HTML инструкции языка PHP для создания динамического содержания. Эти инструкции обрабатывает препроцессор-интерпретатор PHP и заменяет их тем содержимым, которое производит этот код. PHP-программа может целиком состоять из конструкций языка PHP, а может быть смесью конструкций языков PHP и HTML. Стандартное расширение файла с



PHP-программой - *php.*

Одним из распространенных применений PHP является работа с базами данных. Для целого ряда баз данных PHP имеет собственную поддержку, а другие доступны через ODBC-функции PHP. При вызове PHP-программы URL-адрес должен содержать номер порта, через который работает PHP:

html://fpm.ami.nstu.ru:81/~pmxxyy/t1.php

К особенностям языка PHP относятся:

- возможность встраивать конструкции языка PHP в HTML-документ;
- возможность включать в PHP-программу файлы;
- наличие достаточного набора встроенных функций;
- возможность определять собственные переменные, строки, массивы, объекты;
- наличие необходимого набора управляющих структур;
- возможность вводить собственные функции.

Одна из наиболее удобных особенностей PHP - это способность автоматически передавать значения переменных из HTML-форм в переменные PHP. PHP автоматически генерирует набор переменных, имена которых совпадают с именами объектов в HTML-форме и содержащих значения данных объектов. В результате отпадает необходимость в выполнении рутинного преобразования, связанного с разбором последовательности *имя=значение&имя1=значение1&...&имяN=значениеN*

Для связи с любой из СУБД PHP в своем наборе имеет ряд функций, которые очень похожи между собой и имеют одинаковую логику работы и аналогичные параметры.

В приведенной ниже таблице представлен минимальный набор функций, необходимых для написания PHP-программ, общающихся с СУБД Informix.

1. **int ifx_connect (string database, string user, string password)** - создать соединение с сервером Informix

. Входные параметры: *Database* - имя базы данных; *user* - имя пользователя *password* - пароль.

. Возвращаемое значение: идентификатор соединения, если соединение прошло успешно, и равен 0 в противном случае.

2. **int ifx_query (string query, int link_id, int cursor_type)** - выполнить запрос к базе

. Входные параметры: *query* - строка SQL-запроса; *link_id* - идентификатор соединения; *cursor type* - тип курсора

. Возвращаемое значение: значение 1 или 0 в зависимости от успеха выполнения операции.

3. **array ifx_fetch_row (int result_id, mixed [position])** - получить строку запроса как массив

. Входные параметры: *result_id* - идентификатор результата, возвращенный функцией ifx_query() (только для запросов типа select); *[position]* - параметр из списка: "NEXT", "PREVIOUS", "CURRENT", "FIRST", "LAST" или номер.

. Возвращаемое значение: строка таблицы базы данных, возвращаемая как массив.

4. **string current (array row)** - получить очередное поле из строки таблицы базы данных.

. Входные параметры: *array row*- строка таблицы базы данных, возвращенная функцией **ifx_fetch_row().**

. Возвращаемое значение: очередное поле строки таблицы.

5. **string next (array row)** - получить следующее поле из строки таблицы базы данных.

. Входные параметры: *array row* - строка таблицы базы данных, возвращенная функцией **ifx_fetch_row().**

. Возвращаемое значение: следующее поле строки таблицы.

6. **int reset(array$row)** - перейти в начало строки.

array *row* - строка таблицы базы данных, возвращенная

. Входные функцией **ifx_fetch_row().**

параметры:

. Возвращаемое значение: нулевая позиция строки результата.

7. **string key(array$row)** - перейти в начало строки.

array *row* - строка таблицы базы данных, возвращенная

. Входные функцией **ifx_fetch_row().**

параметры:

. Возвращаемое значение: имя очередного поля строки результата.

int ifx_affected_rows (int

8. - получить число столбцов, обработанных запросом

result_i

d)

. Входные параметры: *result_id* - результат, возвращенный функцией

ifx_query().

Возвращается число столбцов, обработанных запросом, ассоциированных с *result_id.* Для вставок, обновлений и удалений

. - это реальное количество обработанных столбцов. Для выборок - ожидаемое количество.

Возвращаемое значение:

- освободить ресурсы запроса

int

ifx_free_result (int

9.

result_id)

. Входные параметры: *result_id* - результат, возвращенный функцией

ifx_query().

Освобождает ресурсы, занятые запросом с идентификатором

. результата *result_id.* Возвращает 0 в случае ошибки.

Возвращаемое значение:

- закрыть соединение с Informix

int

ifx_close (int

10.

[link_identifier])

. Входные параметры: *link_id* - идентификатор соединения;

закрывает соединение с Informix. Если идентификатор ссылки не

. указан, предполагается последнее установленное соединение.

Возвращаемое значение:

Общая схема написания PHP-программы, выполняющей взаимодействие с базой данных, мало отличается от структуры CGI-скрипта, написанного любыми другими средствами,

разница состоит лишь в используемых средствах:

- подключиться к серверу баз данных и зарегистрироваться;
- выбрать базу данных, которая будет использоваться;
- отправить запрос SQL на сервер и получить данные;

− отключиться от сервера баз данных.

При этом остаются актуальными все замечания, сделанные в предыдущей лабораторной работе относительно установки переменных окружения и обеспечения мер безопасности при работе с базой данных.

Последовательность выполнения лабораторной работы

1. Убедиться в наличии и заполненности базы данных поставщиков, деталей, изделий, поставок.
2. Разработать и отладить HTML-формы для ввода данных пользователя согласно варианту задания (можно модифицировать HTML-формы из предыдущей лабораторной работы).
3. Разработать и отладить PHP-программы для обработки данных HTML-форм и доступа к базе данных.
4. После выполнения лабораторной работы привести базу данных в исходное состояние.

Требования к разрабатываемой программе

Разрабатываемые программы должна удовлетворять следующим требованиям:

− разрабатываемое программное приложение должно содержать HTML-документ с формой для ввода данных и PHP-программу, вызываемую по окончании работы с HTML-формой;

− ввод параметров задания в HTML-форме может быть осуществлен либо путем ввода значений в текстовом виде, либо посредством выбора значений из предлагаемого списка;

− программа должна быть написана в предположении, что любой пользователь без ограничений может иметь доступ к данным;

− в программе должен быть предусмотрен вывод сообщений обо всех шагах ее выполнения, в том числе и о возможных ошибках;

− программа должна быть достаточно документирована.

Варианты заданий

Вариант 1

1. Вывести информацию о поставщиках, поставивших детали для изделий из указанного города.
2. Увеличить рейтинг поставщика, выполнившего наибольшую поставку некоторой детали, на указанную величину.

Вариант 2

1. Вывести информацию о деталях, поставки которых были осуществлены для указанного изделия.
2. Изменить цвет самой тяжелой детали на указанный.

Вариант 3

1. Вывести информацию о поставщиках, которые осуществляли поставки деталей из заданного города в указанный период.
2. Вставить поставщика с заданными параметрами.

Вариант 4

1. Вывести информацию о поставщиках, поставивших указанную деталь в заданный период.
2. Удалить поставщика, выполнившего меньше всего поставок.

Вариант 5

1. Вывести информацию обо всех деталях, поставляемых для указанного изделия более чем одним поставщиком.
2. В таблице поставок изменить номер поставщика при заданном номере детали и изделия.

Вариант 6

1. Вывести информацию о деталях, поставки которых были осуществлены для указанного изделия всеми поставщиками.
2. Увеличить рейтинг поставщика, выполнившего больший суммарный объем поставок, на указанную величину.

Вариант 7

1. Вывести информацию об изделиях, для которых была поставлена указанная деталь.
2. Изменить название и город детали с максимальным весом на указанные значения.

Вариант 8

1. Вывести информацию о поставщиках, которые осуществляли поставки деталей для указанного изделия.
2. Увеличить рейтинг поставщика, выполнившего большее число поставок, на указанную величину.

Контрольные вопросы

1. Каким образом вставить конструкции PHP в HTML-документ?
2. Каким образом обеспечить, чтобы встречающиеся в строке переменные были заменены их значениями?
3. Каковы правила определения функций в языке PHP?
4. Каковы особенности передачи значений переменных из HTML-формы в переменные PHP?
5. Каким образом осуществляется взаимодействие с базами данных в языке PHP?
6. В чем заключается технология "cookies"?

Лабораторная работа 9
Тема 9. Методы сериализации транзакции
Безопасность архитектуры «клиент-сервер»

Работа с phpmyadmin

1. Установите веб-сервер Denwer.
2. Запуск phpmyadmin.

Откройте браузер и введите адрес http://localhost/phpmyadmin/, после чего в браузер загрузится следующая страница (Рисунок 1):

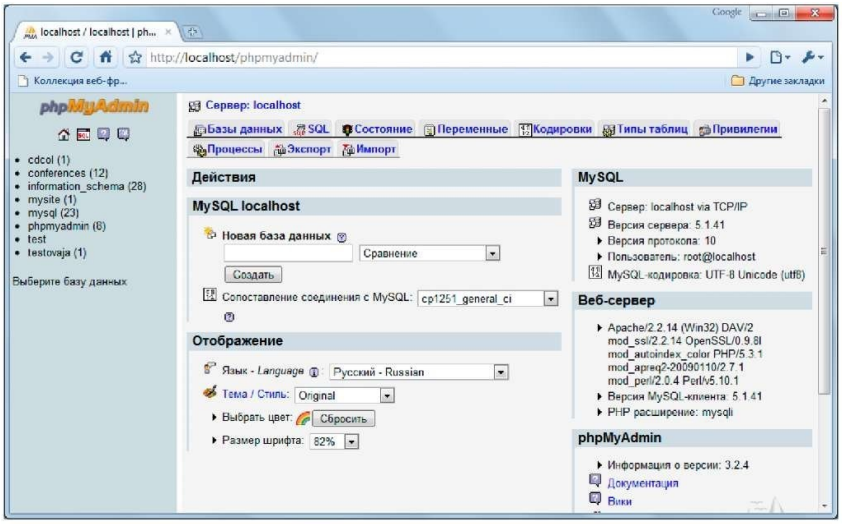


Рисунок 1. Интерфейс главной страницы phpmyadmin

3. Если используете систему в первый раз, и не изменяли данные пользователя root, то будет выведено соответствующее предупреждение.
4. Изменение пароля для root

Чтобы добавить нового пользователя для СУБД MySQL или изменить данные существующих пользователей, необходимо активировать вкладку **«Привилегии»** (Рисунок 2):

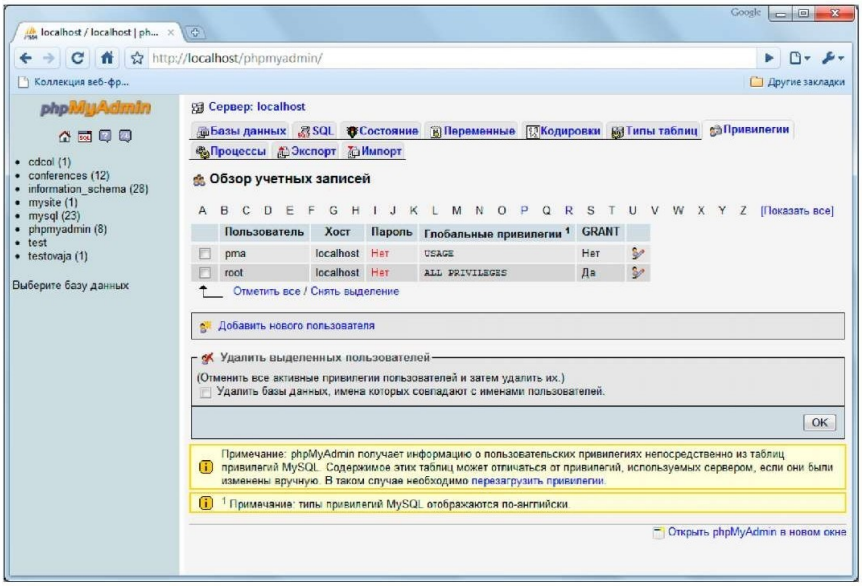


Рисунок 2. Интерфейс вкладки «Привилегии» системы phpmyadmin

5. На данной странице отображается список пользователей MySQL, в том числе и главной пользователь (суперпользователь) - root (по аналогии с операционными системами

семейства UNIX). По умолчанию, данный пользователь не имеет пароля после установки системы. Вам необходимо задать для него пароль.

6. Изменение пароля для пользователя root.
7. В списке учетных записей СУБД MySQL отметьте необходимого пользователя (в данном случае root).

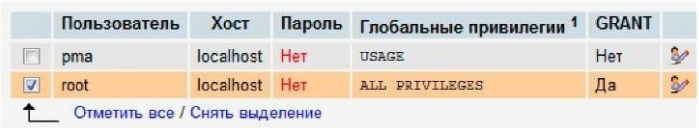


Рисунок 3. Выбор пользователя для редактирования

Дальше нажмите в последней колонке таблицы с учетными записями пользователей на пиктограмму, которая вызывает интерфейс для редактирования данных выбранного пользователя (Рисунок 4).

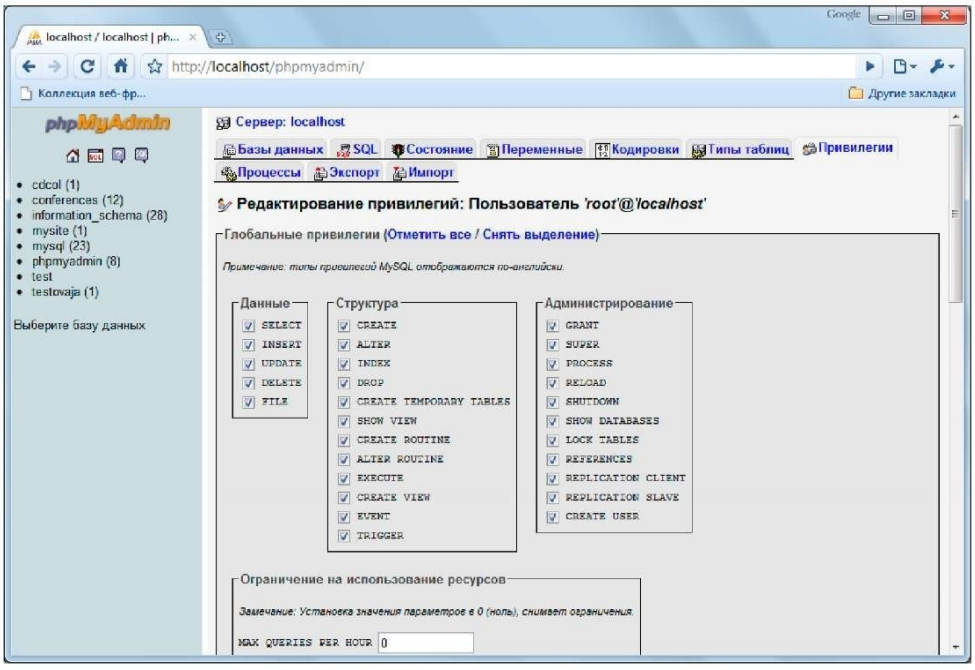


Рисунок 4. Интерфейс страницы для редактирования данных учетной записи

В блоке **«Изменить пароль»** введите новый пароль и подтвердите его, введя в поле «Подтверждение» тот же самый текст. Если вы не хотите самостоятельно определять пароль для пользователей, то можете воспользоваться пунктом «Создать пароль» и нажать кнопку

«Генерировать», после чего система предложит свой вариант пароля, сгенерированный случайным образом.

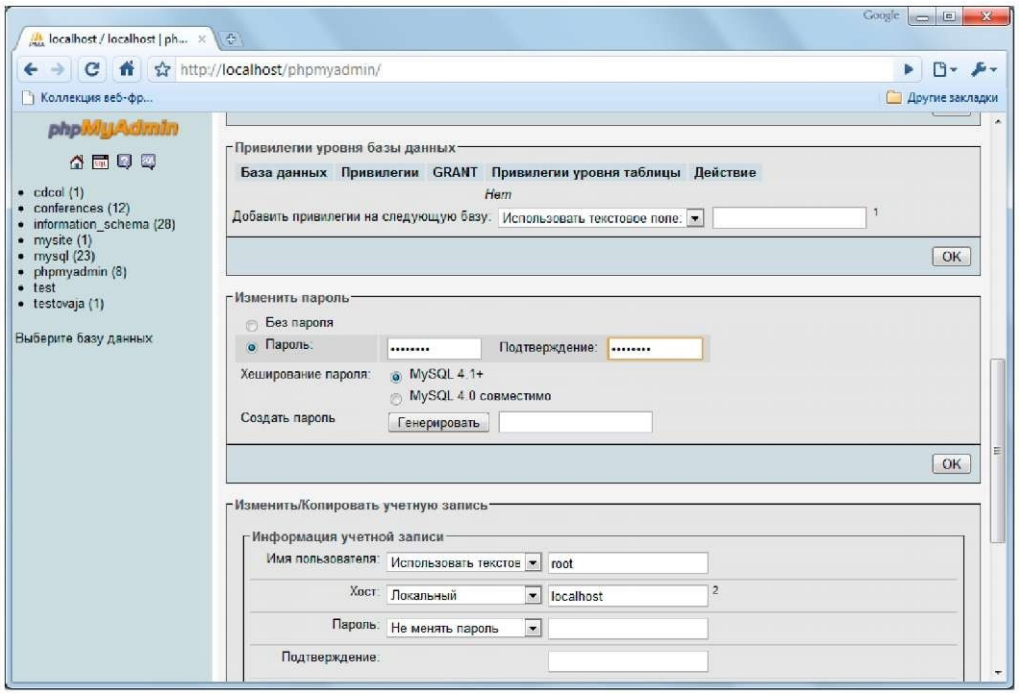


Рисунок 5. Изменение пароля пользователя

После ввода необходимых данных нажмите на кнопку **«ОК».** На странице появится надпись «Пароль для 'root'@'localhost' был успешно изменен».

8. Вернитесь в обзор учетных записей пользователей СУБД MySQL.

Обновите страницу. Так как вы изменили пароль пользователя root, а именно от него Вы работаете в данный момент в системе phpmyadmin, то будет выведено сообщение следующего вида и содержания:

phpMyAdmin не смог установить соединение с сервером MySQL. Проверьте хост, имя пользователя и пароль установленные в конфигурационном файле config.inc.php и удостоверьтесь, что они соответствуют данным полученным от администратора сервера MySQL.

Это означает, что доступ для пользователя с именем root и с пустым паролем запрещен. Для устранения данной проблемы необходимо в конфигурационном файле phpmyadmin также изменить пароль для root, так как по умолчанию там стоит пустая строка.

Откройте файл config.inc.php из директории с установленным phpmyadmin. Найдите строки

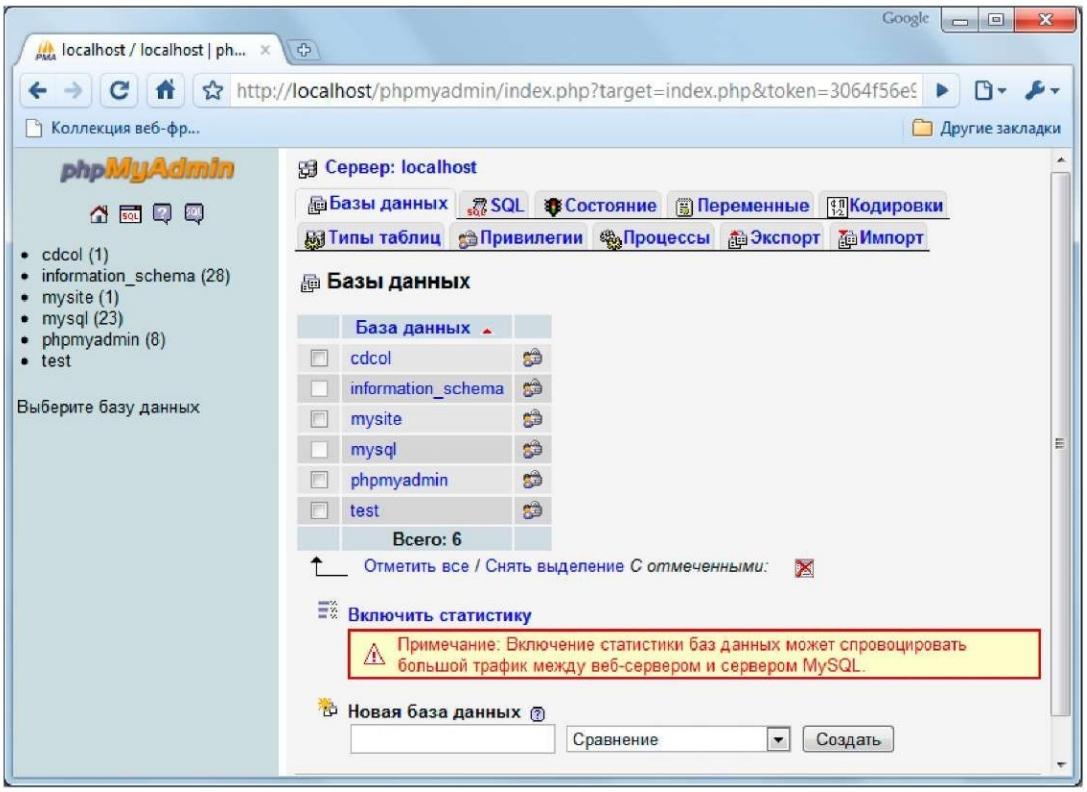
```
$cfg['Servers'][$i]['user']= 'root';
$cfg['Servers'][$i]['password'] = '';
```

Измените значение переменной $cfg['Servers'][$i]['password'] на то значение, которое вы указали как новый пароль для пользователя root: $cfg['Servers'][$i]['password'] = 'password';

9. Обновите страницу Phpmyadmin **(F5).**

10. Активируйте вкладку **«База данных».** На данной странице будет отображен список созданных баз данных в СУБД MySQL.

Рисунок 7. Интерфейс страницы «Базы данных»



С помощью данной страницы вы можете редактировать параметры существующих БД, удалять БД и создавать новые.

11. В блоке «Новая база данных» на текущей странице введите в текстовое поле «webProject», а в выпадающем меню с доступными кодировками для БД выберите CP1251-bin (аналогично Windows-1251), нажмите на кнопку «Создать» (Рисунок 8).

Рисунок 8. Ввод параметров для новой БД

12. Страница обновится, и появится сообщение следующего вида (Рисунок 9):

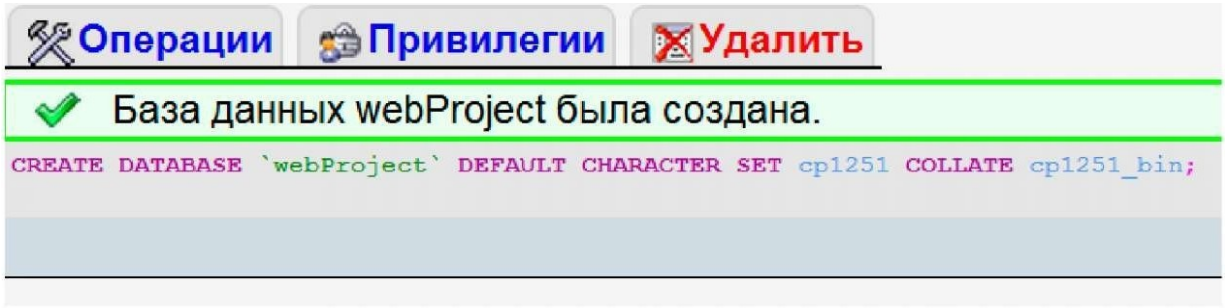


Рисунок 9. Сообщение об успешном создании БД

13. Обратите внимание на то, что практически при любых манипуляциях с БД и пользователями БД система phpmyadmin выводит на страницу соответствующий SQL-запрос. В данном случае это SQL- запрос для создания новой БД.

Для удобства разработчика система предлагает также генерацию PHP-кода для соответствующего запроса. Т.е. при разработке конкретно системы, которая обрабатывает данные из БД, вы можете сначала протестировать необходимые запросы в phpmyadmin, а затем скопировать PHP-код с SQL-запросом в свою систему.

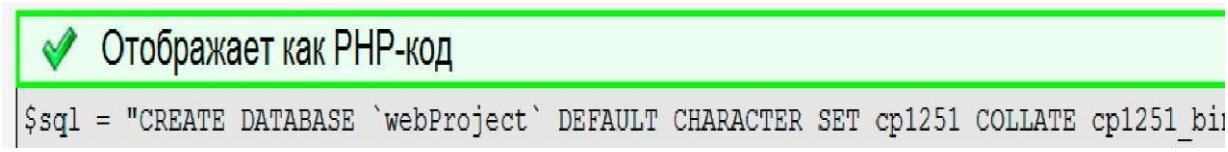


Рисунок 10. PHP-код с SQL-запросом для создания БД

14. После создания новой БД, система автоматически активирует работу с ней. Перейдите во вкладку **«Структура».** Здесь находится список таблиц выбранной БД (вновь созданная БД является пустой).

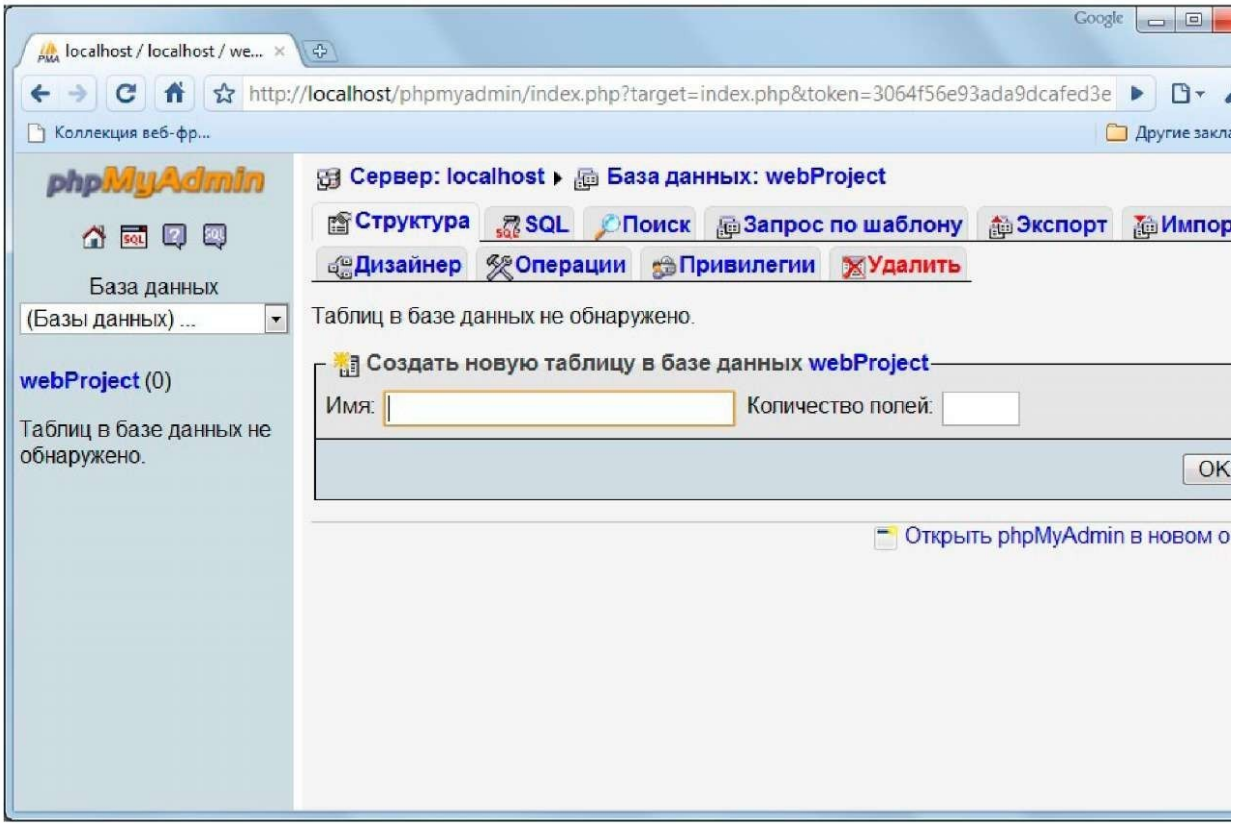


Рисунок 13. Ввод параметров для новой таблицы БД 17. Заполняем форму (водите только те параметры, которые указаны далее):

Имя поля - id (данное поле будет идентификатором записи и должно содержать уникальные значения);

тип - INT;

A_I (Auto increment) - ON (что означает, что при добавлении новой записи в таблицу данное поле будет автоматически увеличено на единицу);

Индекс - PRIMARY (что значит, что это поле - первичный ключ)

Рисунок 11. Интерфейс страницы, содержащей структуру БД (список таблиц БД)

15. Создайте новую таблицу, введя строку «sections» в текстовое поле «Имя» и «5» в поле «Количество полей». Нажмите «ОК».

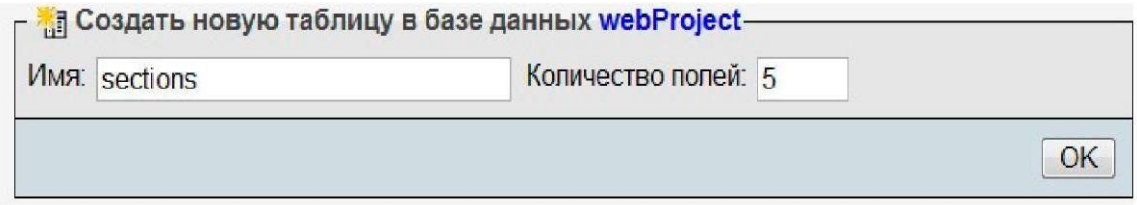
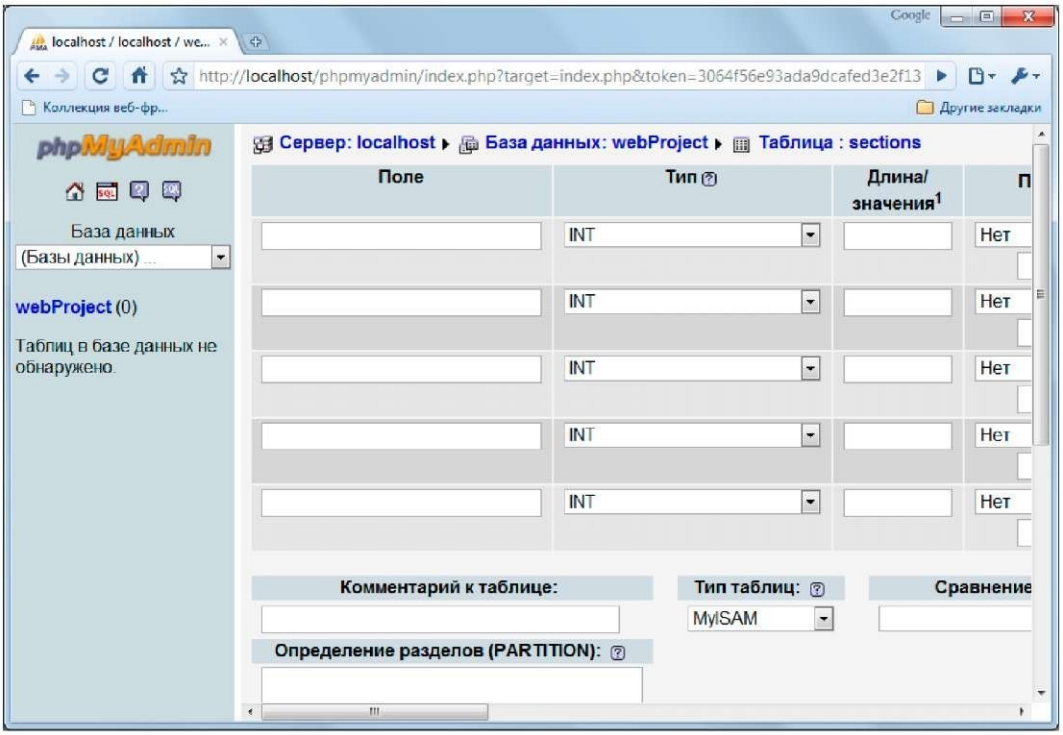


Рисунок 12. Ввод параметров для новой таблицы БД 16. Далее страница обновится и появится форма для добавления параметров каждого из пяти полей нашей таблицы.



Имя поля: caption; **тип** - VARCHAR, **длина** - 50;
Имя поля: text; **тип** - TEXT;
Имя поля: module; **тип** - VARCHAR, **длина** - 30; Allow **NULL** - ON (что означает, что мы разрешаем пустое значение для данного поля)

Имя поля: position; **тип** - INT; **По умолчанию** - Как определено **(Ю).**

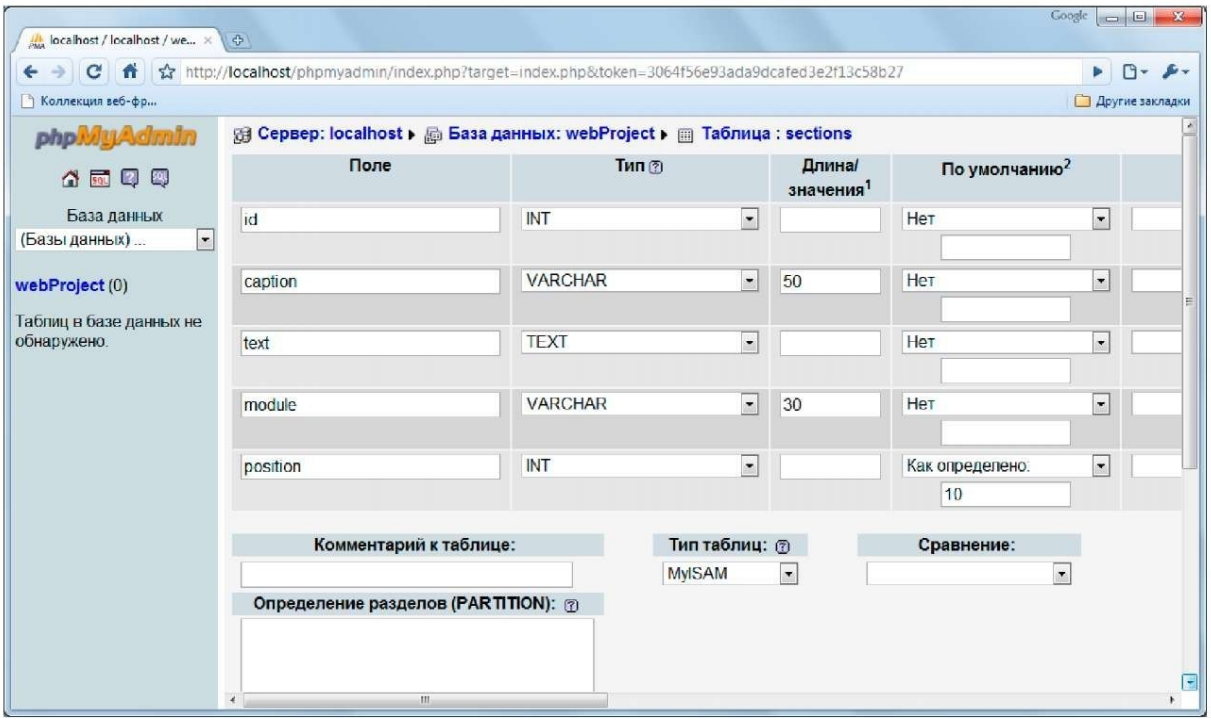


Рисунок 14. Добавление полей в БД 18. Обратите внимание на то, что в любой момент вы можете добавить новое поле, указав, какое количество полей вы хотите добавить и нажав на кнопку ОК:

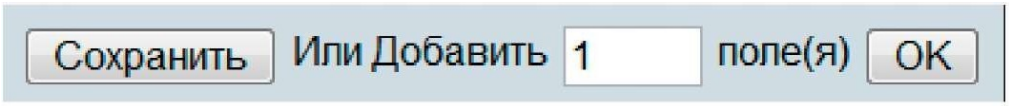


Рисунок 15. Добавление полей в БД 19. Нажмите на кнопку «Сохранить». Страница после обновления примет следующий вид:

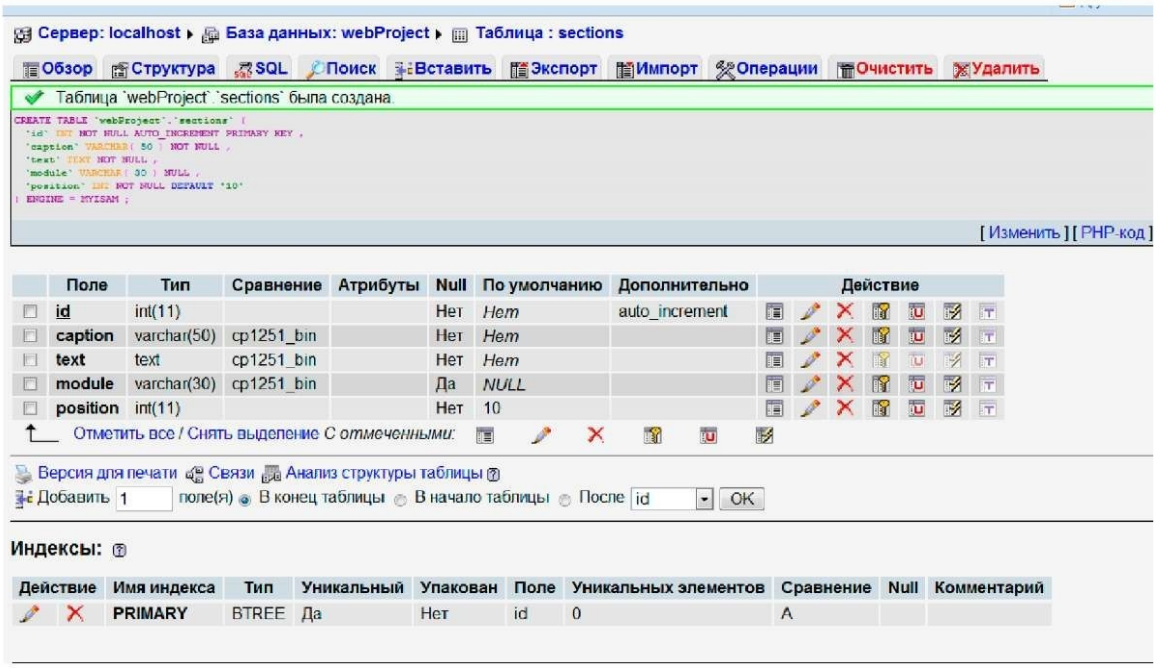


Рисунок 16. Интерфейс страницы «Структура» после сохранения таблицы

```
CREATE TABLE `webProject`.`sections` (
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `caption` VARCHAR( 50 ) NOT NULL ,
  `text` TEXT NOT NULL ,
  `module` VARCHAR( 30 ) NULL ,
  `position` INT NOT NULL DEFAULT '10'
) ENGINE = MYISAM ;
```

Рисунок 17. SQL-запрос, выполненный для создания таблицы

20. После создания таблицы вы можете в любой момент изменить ее структуру (добавлять, редактировать или удалять поля и т.д.)

21. Добавление нового поля в таблицу осуществляется с использованием следующей формы (Рисунок 18):

Рисунок 18. Вид формы для добавления нового поля в существующую таблицу

22. Чтобы добавить запись, перейдите во вкладку «Вставить» и заполните форму согласно следующему изображению:

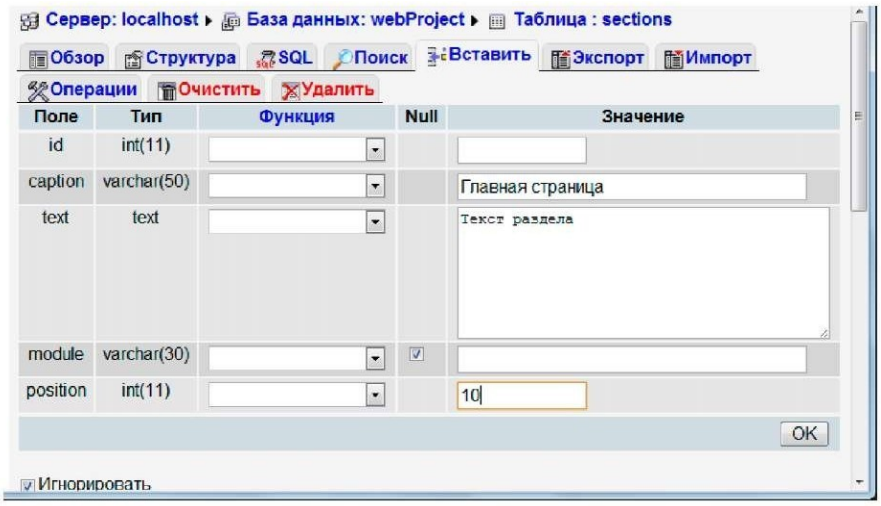


Рисунок 19. Добавление новых записей Нажмите ОК. После обновления страницы появится сообщение (Рисунок 20):

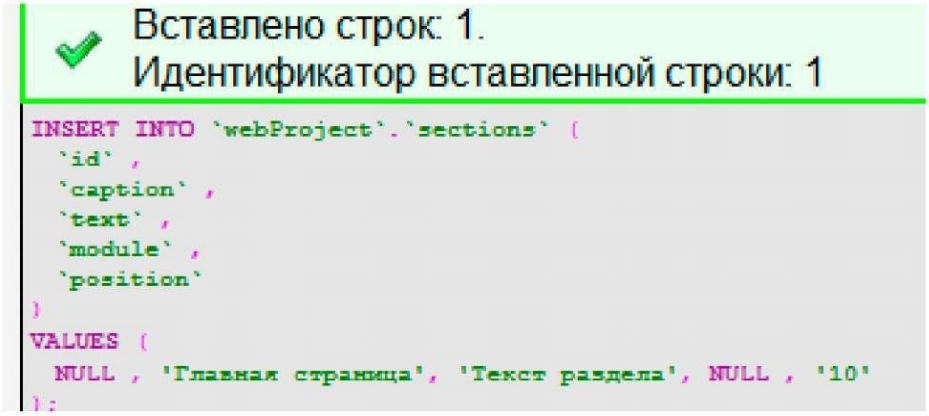


Рисунок 20. SQL-запрос, выполненный для вставки новой записи в таблицу

23. Самостоятельно

24. Добавьте новое поле «idParent» в таблицу (целый тип). Данное поле будет указывать на «родителя» данной записи и позволит организовать древовидную структуру на сайте.

25. Спроектируйте и создайте таблицу в этой же БД для хранения новостей и организации новостной ленты на сайте.

26. Добавьте нового пользователя для созданной БД и установите ему привилегии, позволяющие работать только с данной БД (webProject).

5. Учебно-методическое и информационное обеспечение дисциплины

5.1. Перечень основной и дополнительной литературы, необходимой для освоения дисциплины

5.1.1. Перечень основной литературы:

1. Гайдамакин Н.А. Автоматизированные информационные системы, базы и банки данных вводный курс: учеб.пособие для вузов / Н. А. Гайдамакин. - М.: Гелиос АРВ, 2013. - 3 с. ил. - Библиогр.: с. 354-355. - Алф.-предм. указ.: с. 356-364. - ISBN 5-85438-035-8
2. Основы проектирования и разработки реляционных баз данных: (Спец. 075200 Компьютерная безопасность): учеб. пособие / авт.-сост.: О. М. Лепешкин, Д. Л. Осипов Федеральное агентство по образованию, Ставроп. гос. ун-т. - Ставрополь : Изд-во СГ 2007. - 203 с. : прил. - Библиогр.: с. 199-200

5.1.2. Перечень дополнительной литературы:

1 Гущин, А.Н. Базы данных / А.Н. Гущин. – Москва :Директ-Медиа, 2014. – 266 с.: ил., таб. схем. – Режим доступа: по подписке. – URL: http://biblioclub.ru/index.php page=book&id=222149

2 Карпова, Т.С. Базы данных: модели, разработка, реализация / Т.С. Карпова. – 2-е из исправ. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 241 с.: – Режим доступа: по подписке. – URL: http://biblioclub.ru/index.php?page=book&id=429003

5.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине (модулю)

1. Методические рекомендации по выполнению лабораторных работ по дисциплине «Безопасность баз данных».
2. Методические рекомендации по организации самостоятельной работы студентов по дисциплине «Безопасность баз данных».

5.3. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины (модуля)

1. http://el.ncfu.ru/ – система управления обучением ФГАОУ ВО СКФУ. Дистанционная поддержка дисциплины «Цифровая грамотность и обработка данных»
2. http://www.un.org - Сайт ООН Информационно-коммуникационные технологии
3. http://www.intuit.ru – Интернет-Университет Компьютерных технологий.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Пятигорский институт (филиал) СКФУ

Методические указания

для обучающихся по организации и проведению самостоятельной работы
по дисциплине «**БЕЗОПАСНОСТЬ БАЗ ДАННЫХ**»
для студентов направления подготовки **10.03.01 Информационная безопасность**
направленность (профиль) **Безопасность компьютерных систем**

Пятигорск, 2024

СОДЕРЖАНИЕ

1. Общие положения

Самостоятельная работа - планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).
Самостоятельная работа студентов (СРС) в ВУЗе является важным видом учебной и научной деятельности студента. Самостоятельная работа студентов играет значительную роль в рейтинговой технологии обучения.
К основным видам самостоятельной работы студентов относятся:
− формирование и усвоение содержания конспекта лекций на базе рекомендованной лектором учебной литературы, включая информационные образовательные ресурсы (электронные учебники, электронные библиотеки и др.);
− написание докладов;
− подготовка к семинарам, практическим и лабораторным работам, их оформление; − составление аннотированного списка статей из соответствующих журналов по отраслям знаний (педагогических, психологических, методических и др.);
− выполнение учебно-исследовательских работ, проектная деятельность;
− подготовка практических разработок и рекомендаций по решению проблемной ситуации;
− выполнение домашних заданий в виде решения отдельных задач, проведения типовых расчетов, расчетно-компьютерных и индивидуальных работ по отдельным разделам содержания дисциплин и т.д.;
− компьютерный текущий самоконтроль и контроль успеваемости на базе электронных обучающих и аттестующих тестов;
− выполнение курсовых работ (проектов) в рамках дисциплин;
− выполнение выпускной квалификационной работы и др.
Методика организации самостоятельной работы студентов зависит от структуры, характера и особенностей изучаемой дисциплины, объема часов на ее изучение, вида заданий для самостоятельной работы студентов, индивидуальных качеств студентов и условий учебной деятельности.
Процесс организации самостоятельной работы студентов включает в себя следующие этапы:

- подготовительный (определение целей, составление программы, подготовка методического обеспечения, подготовка оборудования);
- основной (реализация программы, использование приемов поиска информации, усвоения, переработки, применения, передачи знаний, фиксирование результатов, самоорганизация процесса работы);
- заключительный (оценка значимости и анализ результатов, их систематизация, оценка эффективности программы и приемов работы, выводы о направлениях оптимизации труда).

Самостоятельная работа по дисциплине Безопасность баз данных направлена на формирование следующих **компетенций**:

Код	Формулировка:
ПК-4	Способность участвовать в работах по реализации политики информационной безопасности, применять комплексный подход к обеспечению информационной безопасности объекта защиты

2. Цель и задачи самостоятельной работы

Ведущая цель организации и осуществления СРС совпадает с целью обучения студента – формирование набора общенаучных, профессиональных и специальных компетенций будущего бакалавра по соответствующему направлению подготовки

При организации СРС важным и необходимым условием становятся формирование умения самостоятельной работы для приобретения знаний, навыков и возможности организации учебной и научной деятельности. Целью самостоятельной работы студентов является овладение фундаментальными знаниями, профессиональными умениями и навыками деятельности по профилю, опытом творческой, исследовательской деятельности. Самостоятельная работа студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня.

Задачами СРС являются:

- систематизация и закрепление полученных теоретических знаний и практических умений студентов;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;
- развитие исследовательских умений;
- использование материала, собранного и полученного в ходе самостоятельных занятий на семинарах, на практических и лабораторных занятиях, при написании курсовых и выпускной квалификационной работ, для эффективной подготовки к итоговым зачетам и экзаменам.

3. Технологическая карта самостоятельной работы студента

Коды реализуемых копетенций	Вид деятельности студентов	Средства и технологии оценки	Объем часов, в том числе		
			СРС	Контактная работа с преподавателем	Всего
5 семестр					
ПК-4 (ИД-1ИД-2ИД-3)	Самостоятельное изучение	Собеседование	4,86	0,54	5,4
ПК-4 (ИД-1ИД-2ИД-3)	Подготовка к лекции	Собеседование	1,62	0,18	1,8
ПК-4 (ИД-1ИД-2ИД-3)	Подготовка к лабораторной работе	Отчет ЛР	9,72	1,08	10,8
Итого за 5 семестр			16,2	1,8	18
Итого			16,2	1,8	18

4. Порядок выполнения самостоятельной работы студентом

4.1. Методические рекомендации по работе с учебной литературой

При работе с книгой необходимо подобрать литературу, научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги.

Важно помнить, что рациональные навыки работы с книгой - это всегда большая экономия времени и сил.

Правильный подбор учебников рекомендуется преподавателем, читающим лекционный курс. Необходимая литература может быть также указана в методических разработках по данному курсу.

Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и вычисления (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода).

При изучении любой дисциплины большую и важную роль играет самостоятельная индивидуальная работа.

Особое внимание следует обратить на определение основных понятий курса. Студент должен подробно разбирать примеры, которые поясняют такие определения, и уметь строить аналогичные примеры самостоятельно. Нужно добиваться точного представления о том, что изучаешь. Полезно составлять опорные конспекты. При изучении материала по учебнику полезно в тетради (на специально отведенных полях) дополнять конспект лекций. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем.

Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при перечитывании записей лучше запоминались.

Опыт показывает, что многим студентам помогает составление листа опорных сигналов, содержащего важнейшие и наиболее часто употребляемые формулы и понятия. Такой лист помогает запомнить формулы, основные положения лекции, а также может служить постоянным справочником для студента.

Чтение научного текста является частью познавательной деятельности. Ее цель – извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия.

Выделяют ***четыре основные установки в чтении научного текста***:

информационно-поисковый (задача – найти, выделить искомую информацию)

усваивающая (усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить, как сами сведения, излагаемые автором, так и всю логику его рассуждений)

аналитико-критическая (читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему)

творческая (создает у читателя готовность в том или ином виде – как отправной пункт для своих рассуждений, как образ для действия по аналогии и т.п. – использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке).

Основные виды систематизированной записи прочитанного:

Аннотирование – предельно краткое связное описание просмотренной или прочитанной книги (статьи), ее содержания, источников, характера и назначения;

Планирование – краткая логическая организация текста, раскрывающая содержание и структуру изучаемого материала;

Тезирование – лаконичное воспроизведение основных утверждений автора без привлечения фактического материала;

Цитирование – дословное выписывание из текста выдержек, извлечений, наиболее существенно отражающих ту или иную мысль автора;

Конспектирование – краткое и последовательное изложение содержания прочитанного.

Конспект – сложный способ изложения содержания книги или статьи в логической последовательности. Конспект аккумулирует в себе предыдущие виды записи, позволяет всесторонне охватить содержание книги, статьи. Поэтому умение составлять план, тезисы, делать выписки и другие записи определяет и технологию составления конспекта.

Методические рекомендации по составлению конспекта:

1. Внимательно прочитайте текст. Уточните в справочной литературе непонятные слова. При записи не забудьте вынести справочные данные на поля конспекта;
2. Выделите главное, составьте план;
3. Кратко сформулируйте основные положения текста, отметьте аргументацию автора;
4. Законспектируйте материал, четко следуя пунктам плана. При конспектировании старайтесь выразить мысль своими словами. Записи следует вести четко, ясно.
5. Грамотно записывайте цитаты. Цитируя, учитывайте лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной

последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля.
Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

4.2. Методические рекомендации по подготовке к практическим и лабораторным занятиям

Для того чтобы практические и лабораторные занятия приносили максимальную пользу, необходимо помнить, что упражнение и решение задач проводятся по вычитанному на лекциях материалу и связаны, как правило, с детальным разбором отдельных вопросов лекционного курса. Следует подчеркнуть, что только после усвоения лекционного материала с определенной точки зрения (а именно с той, с которой он излагается на лекциях) он будет закрепляться на практических занятиях как в результате обсуждения и анализа лекционного материала, так и с помощью решения проблемных ситуаций, задач. При этих условиях студент не только хорошо усвоит материал, но и научится применять его на практике, а также получит дополнительный стимул (и это очень важно) для активной проработки лекции.
При самостоятельном решении задач нужно обосновывать каждый этап решения, исходя из теоретических положений курса. Если студент видит несколько путей решения проблемы (задачи), то нужно сравнить их и выбрать самый рациональный. Полезно до начала вычислений составить краткий план решения проблемы (задачи). Решение проблемных задач или примеров следует излагать подробно, вычисления располагать в строгом порядке, отделяя вспомогательные вычисления от основных. Решения при необходимости нужно сопровождать комментариями, схемами, чертежами и рисунками.
Следует помнить, что решение каждой учебной задачи должно доводиться до окончательного логического ответа, которого требует условие, и по возможности с выводом. Полученный ответ следует проверить способами, вытекающими из существа данной задачи. Полезно также (если возможно) решать несколькими способами и сравнить полученные результаты. Решение задач данного типа нужно продолжать до приобретения твердых навыков в их решении.

4.3. Методические рекомендации по самопроверке знаний

После изучения определенной темы по записям в конспекте и учебнику, а также решения достаточного количества соответствующих задач на практических занятиях и самостоятельно студенту рекомендуется, провести самопроверку усвоенных знаний, ответив на контрольные вопросы по изученной теме.
В случае необходимости нужно еще раз внимательно разобраться в материале.
Иногда недостаточность усвоения того или иного вопроса выясняется только при изучении дальнейшего материала. В этом случае надо вернуться назад и повторить плохо усвоенный материал. Важный критерий усвоения теоретического материала - умение решать задачи или пройти тестирование по пройденному материалу. Однако следует помнить, что правильное решение задачи может получиться в результате применения механически заученных формул без понимания сущности теоретических положений.

4.4. Методические рекомендации по написанию научных текстов (докладов, эссе, научных статей и т.д.)

Перед тем, как приступить к написанию научного текста, важно разобраться, какова истинная цель вашего научного текста - это поможет вам разумно распределить свои силы и время.
Во-первых, сначала нужно определиться с идеей научного текста, а для этого необходимо научиться либо относиться к разным явлениям и фактам несколько критически (своя идея – как иная точка зрения), либо научиться увлекаться какими-то известными идеями,

которые нуждаются в доработке (идея – как оптимистическая позиция и направленность на дальнейшее совершенствование уже известного). Во-вторых, научиться организовывать свое время, ведь, как известно, свободное (от всяких глупостей) время – важнейшее условие настоящего творчества, для него наконец-то появляется время. Иногда именно на организацию такого времени уходит немалая часть сил и талантов.

Писать следует ясно и понятно, стараясь основные положения формулировать четко и недвусмысленно (чтобы и самому понятно было), а также стремясь структурировать свой текст. Каждый раз надо представлять, что ваш текст будет кто-то читать и ему захочется сориентироваться в нем, быстро находить ответы на интересующие вопросы (заодно представьте себя на месте такого человека). Понятно, что работа, написанная «сплошным текстом» (без заголовков, без выделения крупным шрифтом наиболее важным мест и т, п.), у культурного читателя должна вызывать брезгливость и даже жалость к автору (исключения составляют некоторые древние тексты, когда и жанр был иной и к текстам относились иначе, да и самих текстов было гораздо меньше – не то, что в эпоху «информационного взрыва» и соответствующего «информационного мусора»).

Объем текста и различные оформительские требования во многом зависят от принятых в конкретном учебном заведении порядков.

Доклад - это самостоятельное исследование студентом определенной проблемы, комплекса взаимосвязанных вопросов.

Доклад не должна составляться из фрагментов статей, монографий, пособий. Кроме простого изложения фактов и цитат, в доклад е должно проявляться авторское видение проблемы и ее решения.

Рассмотрим основные этапы подготовки

а студентом.

Выполнение доклада начинается с выбора темы.

Затем студент приходит на первую консультацию к руководителю, которая предусматривает:

- обсуждение цели и задач работы, основных моментов избранной темы;
- консультирование по вопросам подбора литературы;
- составление предварительного плана.

Следующим этапом является работа с литературой. Необходимая литература подбирается студентом самостоятельно.

После подбора литературы целесообразно сделать рабочий вариант плана работы. В нем нужно выделить основные вопросы темы и параграфы, раскрывающие их содержание.

Составленный список литературы и предварительный вариант плана уточняются, согласуются на очередной консультации с руководителем.

Затем начинается следующий этап работы - изучение литературы. Только внимательно читая и конспектируя литературу, можно разобраться в основных вопросах темы и подготовиться к самостоятельному (авторскому) изложению содержания доклада. Конспектируя первоисточники, необходимо отразить основную идею автора и его позицию по исследуемому вопросу, выявить проблемы и наметить задачи для дальнейшего изучения данных проблем.

Систематизация и анализ изученной литературы по проблеме исследования позволяют студенту написать работу.

Рабочий вариант текста доклада предоставляется руководителю на проверку. На основе рабочего варианта текста руководитель вместе со студентом обсуждает возможности доработки текста, его оформление. После доработки доклад сдается на кафедру для его оценивания руководителем.

Требования к написанию доклада

Написание 1 доклада является обязательным условием выполнения плана СРС по любой дисциплине профессионального цикла.
Тема доклада может быть выбрана студентом из предложенных в рабочей программе или фонде оценочных средств дисциплины, либо определена самостоятельно, исходя из интересов студента (в рамках изучаемой дисциплины). Выбранную тему необходимо согласоваться с преподавателем.
Доклад должен быть написан научным языком.
Объем доклада должен составлять 20-25 стр.
Структура доклада:

- Введение (не более 3-4 страниц). Во введении необходимо обосновать выбор темы, ее актуальность, очертить область исследования, объект исследования, основные цели и задачи исследования.
- Основная часть состоит из 2-3 разделов. В них раскрывается суть исследуемой проблемы, проводится обзор мировой литературы и источников Интернет по предмету исследования, в котором дается характеристика степени разработанности проблемы и авторская аналитическая оценка основных теоретических подходов к ее решению. Изложение материала не должно ограничиваться лишь описательным подходом к раскрытию выбранной темы. Оно также должно содержать собственное видение рассматриваемой проблемы и изложение собственной точки зрения на возможные пути ее решения.
- Заключение (1-2 страницы). В заключении кратко излагаются достигнутые при изучении проблемы цели, перспективы развития исследуемого вопроса
- Список использованной литературы (не меньше 10 источников), в алфавитном порядке, оформленный в соответствии с принятыми правилами. В список использованной литературы рекомендуется включать работы отечественных и зарубежных авторов, в том числе статьи, опубликованные в научных журналах в течение последних 3-х лет и ссылки на ресурсы сети Интернет.
- Приложение (при необходимости).

Требования к оформлению:

- текст с одной стороны листа;
- шрифт Times New Roman;
- кегль шрифта 14;
- межстрочное расстояние 1,5;
- поля: сверху 2,5 см, снизу – 2,5 см, слева - 3 см, справа 1,5 см;
- доклад должен быть представлен в сброшюрованном виде.

Порядок защиты доклада:
Защита доклада проводится на практических занятиях, после окончания работы студента над ним и исправления всех недочетов, выявленных преподавателем в ходе консультаций. На защиту доклада отводится 5-7 минут времени, в ходе которого студент должен показать свободное владение материалом по заявленной теме. При защите доклада приветствуется использование мультимедиа-презентации.
Оценка доклада
Доклад оценивается по следующим критериям:
• соблюдение требований к его оформлению;
• необходимость и достаточность для раскрытия темы приведенной в тексте доклада информации;
• умение студента свободно излагать основные идеи, отраженные в докладе;
• способность студента понять суть задаваемых преподавателем и сокурсниками вопросов и сформулировать точные ответы на них.
Критерии оценки:

Оценка «отлично» выставляется студенту, если в докладе студент исчерпывающе, последовательно, четко и логически стройно излагает материал; свободно справляется с задачами, вопросами и другими видами применения знаний; использует для написания доклада современные научные материалы; анализирует полученную информацию; проявляет самостоятельность при написании доклада.
Оценка «хорошо» выставляется студенту, если качество выполнения доклада достаточно высокое. Студент твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопросы по теме доклада.
Оценка «удовлетворительно» выставляется студенту, если материал доклада излагается частично, но пробелы не носят существенного характера, студент допускает неточности и ошибки при защите доклада, дает недостаточно правильные формулировки, наблюдаются нарушения логической последовательности в изложении материала.
Оценка «неудовлетворительно» выставляется студенту, если он не подготовил доклад или допустил существенные ошибки. Студент неуверенно излагает материал доклада, не отвечает на вопросы преподавателя.
Описание шкалы оценивания
Максимально возможный балл за весь текущий контроль устанавливается равным 55. Текущее контрольное мероприятие считается сданным, если студент получил за него не менее 60% от установленного для этого контроля максимального балла. Рейтинговый балл, выставляемый студенту за текущее контрольное мероприятие, сданное студентом в установленные графиком контрольных мероприятий сроки, определяется следующим образом:

Уровень выполнения контрольного задания	Рейтинговый балл (в % от максимального балла за контрольное задание)
Отличный	100
Хороший	80
Удовлетворительный	60
Неудовлетворительный	0

4.5. Методические рекомендации по выполнению исследовательских проектов
Исследовательская проектная работа – это групповая работа, для выполнения которой необходим выбор и приложение научной методики к поставленной задаче, получение собственного теоретического или экспериментального материала, на основании которого необходимо провести анализ и сделать выводы об исследуемом явлении. Выполнение проекта – это всегда коллективная, творческая практическая работа, предназначенная для получения определенного продукта или научно-технического результата. Такая работа подразумевает четкое, однозначное формирование поставленной задачи, определение сроков выполнения намеченного, определение требований к разрабатываемому объекту.
Выполнение 1 группового проекта является обязательным условием выполнения самостоятельной работы по любой дисциплине профессионального цикла. Тема проектного задания может быть выбрана студентом из предложенных в рабочей программе или фонде оценочных средств дисциплины, либо определена самостоятельно, исходя из интересов студента (в рамках изучаемой дисциплины). Выбранную тему необходимо согласоваться с преподавателем.
Требования по выполнению и оформлению проекта
При выполнении проекта приветствуется работа в группе (2-3 человека). Проект – это исследовательская работа, в ходе которой студенты должны продемонстрировать владение навыками научного исследования, умения проводить анализ, обобщать

информацию, делать выводы, предлагать свои решения проблемы, рассматриваемой в проекте.

При подготовке материалов проекта студенты должны продемонстрировать владение современными методами компьютерной обработки данных.

Критерии оценки работы участника проекта.

Для каждого из участников проекта оцениваются:

• профессиональные теоретические знания в соответствующей области;

• умение работать со справочной и научной литературой, осуществлять поиск необходимой информации в Интернет;

• умение работать с техническими средствами;

• умение пользоваться соответствующими выполняемому проекту информационными технологиями;

• умение готовить материалы проекта для презентации: составлять и редактировать тексты, формировать презентацию проекта;

• умение работать в команде;

• умение публично представлять результаты собственной деятельности;

• коммуникабельность, инициативность, творческие способности.

Критерии выставления оценки участникам проекта

Оценка	Профессиональные компетенции	Компетенции, связанные с использованием соответствующих выполняемому проекту технических средств и информационных технологий	Иные универсальные компетенции (коммуникабельность, инициативность, умение работать в «команде», управленческие навыки и т.д.)	Отчетность
«Отлично»	Работа выполнена на высоком профессиональном уровне. Представленный материал в основном фактически верен, допускаются негрубые фактические неточности. Студент свободно отвечает на вопросы, связанные с проектом.	Технические средства и информационные технологии освоены и использованы для реализации проекта полностью	Студент проявил инициативу, творческий подход, способность к выполнению сложных заданий, навыки работы в коллективе, организационные способности.	Проект представлен полностью и в срок.
«Хорошо»	Работа выполнена на достаточно высоком профессиональном уровне. Допущено до 4–5 фактических ошибок. Студент отвечает на вопросы,	Обнаруживаются некоторые ошибки в использовании соответствующих технических средств и	Студент достаточно полно, но без инициативы и творческих находок выполнил	Проект представлен достаточно полно и в срок, но с некоторыми недоработка

Оценка	Профессиональные компетенции	Компетенции, связанные с использованием соответствующих выполняемому проекту технических средств и информационных технологий	Иные универсальные компетенции (коммуникабельность, инициативность, умение работать в «команде», управленческие навыки и т.д.)	Отчетность
	связанные с проектом, но недостаточно полно.	информационных технологий	возложенные на него задачи.	ми.
«Удовлетворительно»	Уровень недостаточно высок. Допущено до 8 фактических ошибок. Студент может ответить лишь на некоторые из заданных вопросов, связанных с проектом.	Обнаруживает недостаточное владение навыками работы с техническими средствами и соответствующими информационными технологиями	Студент выполнил большую часть возложенной на него работы.	Проект сдан со значительным опозданием (более недели) и не полностью
«Неудовлетворительно»	Работа не выполнена или выполнена на низком уровне. Допущено более 8 фактических ошибок. Ответы на связанные с проектом вопросы обнаруживают непонимание предмета и отсутствие ориентации в материале проекта.	Навыков работы с техническими средствами нет, информационные технологии не освоены	Студент практически не работал, не выполнил свои задачи или выполнил лишь отдельные не существенные поручения в групповом проекте.	Проект не сдан.

Студенты должны: защитить проект в режиме презентации, предъявить файлы выполненного проекта, уметь рассказать о технологиях, использованных ими при выполнении проекта, дать оценку работы каждого члена группы (*если проект групповой*).
Максимально возможный балл за весь текущий контроль устанавливается равным **55.** Текущее контрольное мероприятие считается сданным, если студент получил за него не менее 60% от установленного для этого контроля максимального балла. Рейтинговый балл, выставляемый студенту за текущее контрольное мероприятие, сданное студентом в установленные графиком контрольных мероприятий сроки, определяется следующим образом:

Уровень выполнения контрольного	Рейтинговый балл (в % от максимального

задания	балла за контрольное задание)
Отличный	100
Хороший	80
Удовлетворительный	60
Неудовлетворительный	0

4.6. Методические рекомендации по подготовке к экзаменам и зачетам

Изучение многих общепрофессиональных и специальных дисциплин завершается экзаменом. Подготовка к экзамену способствует закреплению, углублению и обобщению знаний, получаемых, в процессе обучения, а также применению их к решению практических задач. Готовясь к экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На экзамене студент демонстрирует то, что он приобрел в процессе обучения по конкретной учебной дисциплине.

Экзаменационная сессия - это серия экзаменов, установленных учебным планом. Между экзаменами интервал 3-4 дня. Не следует думать, что 3-4 дня достаточно для успешной подготовки к экзаменам.

В эти 3-4 дня нужно систематизировать уже имеющиеся знания. На консультации перед экзаменом студентов познакомят с основными требованиями, ответят на возникшие у них вопросы. Поэтому посещение консультаций обязательно.

Требования к организации подготовки к экзаменам те же, что и при занятиях в течение семестра, но соблюдаться они должны более строго. Во-первых, очень важно соблюдение режима дня; сон не менее 8 часов в сутки, занятия заканчиваются не позднее, чем за 2-3 часа до сна. Оптимальное время занятий - утренние и дневные часы. В перерывах между занятиями рекомендуются прогулки на свежем воздухе, неутомительные занятия спортом. Во-вторых, наличие хороших собственных конспектов лекций. Даже в том случае, если была пропущена какая-либо лекция, необходимо вовремя ее восстановить (переписать ее на кафедре), обдумать, снять возникшие вопросы для того, чтобы запоминание материала было осознанным. В-третьих, при подготовке к экзаменам у студента должен быть хороший учебник или конспект литературы, прочитанной по указанию преподавателя в течение семестра. Здесь можно эффективно использовать листы опорных сигналов.

Вначале следует просмотреть весь материал по сдаваемой дисциплине, отметить для себя трудные вопросы. Обязательно в них разобраться. В заключение еще раз целесообразно повторить основные положения, используя при этом листы опорных сигналов.

Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Контроль самостоятельной работы студентов

Контроль самостоятельной работы проводится преподавателем в аудитории.

Предусмотрены следующие виды контроля: собеседование, оценка доклада, оценка презентации, оценка участия в круглом столе, оценка выполнения проекта.

Подробные критерии оценивания компетенций приведены в Фонде оценочных средств для проведения текущей и промежуточной аттестации.

Список литературы для выполнения СРС

Основная литература:

1. Гайдамакин Н.А. Автоматизированные информационные системы, базы и банки данных вводный курс: учеб.пособие для вузов / Н. А. Гайдамакин. - М.: Гелиос АРВ, 2013. - 3 с. ил. - Библиогр.: с. 354-355. - Алф.-предм. указ.: с. 356-364. - ISBN 5-85438-035-8
2. Основы проектирования и разработки реляционных баз данных: (Спец. 075200 Компьютерная безопасность): учеб. пособие / авт.-сост.: О. М. Лепешкин, Д. Л. Осипов Федеральное агентство по образованию, Ставроп. гос. ун-т. - Ставрополь : Изд-во СГ 2007. - 203 с. : прил. - Библиогр.: с. 199-200

Дополнительная литература:

3 Гущин, А.Н. Базы данных / А.Н. Гущин. – Москва :Директ-Медиа, 2014. – 266 с.: ил., таб. схем. – Режим доступа: по подписке. – URL: http://biblioclub.ru/index.php page=book&id=222149

4 Карпова, Т.С. Базы данных: модели, разработка, реализация / Т.С. Карпова. – 2-е из исправ. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 241 с.: – Режим доступа: по подписке. – URL: http://biblioclub.ru/index.php?page=book&id=429003

Методические рекомендации для самостоятельной работы студентов по дисциплине

1. Методические рекомендации по выполнению лабораторных работ по дисциплине «Безопасность баз данных».
2. Методические рекомендации по организации самостоятельной работы студентов по дисциплине «Безопасность баз данных».

Интернет-ресурсы:

1. http://el.ncfu.ru/ – система управления обучением ФГАОУ ВО СКФУ. Дистанционная поддержка дисциплины «Цифровая грамотность и обработка больших данных»
2. http://www.un.org - Сайт ООН Информационно-коммуникационные технологии
3. http://www.intuit.ru – Интернет-Университет Компьютерных технологий.