УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

УТВЕРЖДАЮ
Директор колледжа
_____ С.Н.Козлов
19.12.2025

СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ
ПО ИЗУЧЕНИЮ УЧЕБНОГО ПРЕДМЕТА,
ЗАДАНИЯ НА ДОМАШНЮЮ КОНТРОЛЬНУЮ РАБОТУ №1
ДЛЯ УЧАЩИХСЯ ЗАОЧНОЙ ФОРМЫ ПОЛУЧЕНИЯ ОБРАЗОВАНИЯ
ПО СПЕЦИАЛЬНОСТИ 5-04-0612-02
« РАЗРАБОТКА И СОПРОВОЖДЕНИЕ ПРОГРАММНОГО
ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ»

2025

Автор: Базанова А.Д., преподаватель учреждения образования «Могилевский государственный политехнический колледж»

Рецензент: Попкова Ю.А., преподаватель учреждения образования «Могилевский государственный политехнический колледж»

Разработано в соответствии с учебной программой по учебному предмету «Системы управления базами данных» профессионального компонента учебного плана учреждения образования по специальности 5-04-0612-02 «Разработка и сопровождение программного обеспечения информационных систем» для реализации образовательной программы среднего специального образования, обеспечивающей получение квалификации специалиста со средним специальным образованием, утвержденной директором колледжа, 29.08.2024

Обсуждено и одобрено
на заседании цикловой комиссии
специальностей в области программного
обеспечения информационных систем
Протокол № ____ от _____
Председатель цикловой комиссии
_____ Д.А.Денисовец

Пояснительная записка

Программа учебного предмета «Системы управления базами данных» предусматривает изучение основных понятий баз данных (БД), систем управления базами данных (СУБД), моделей данных, основных определений реляционной модели данных, средств манипулирования реляционными данными, основ структурированного языка запросов, основ проектирования и сопровождения реляционных баз данных, а также способов создания и ведения систем автоматизированной обработки информации на основе использования конкретных систем управления базами данных реляционного типа.

Изучение учебного предмета базируется на знаниях, умениях, и навыках, полученных при изучении следующих предметов: «Основы алгоритмизации и программирования», «Конструирование программ и языки программирования», «Технология разработки программного обеспечения».

Основной целью учебного предмета «Системы управления базами данных» является формирование профессиональной компетентности учащихся в вопросах создания и использования баз данных.

Основные задачи изучения учебного предмета:

- сформировать целостное представление о базах данных;
- сформировать умения создавать базы данных, проводить их нормализацию, создавать запросы к базе данных, создавать формы и отчеты в СУБД, разрабатывать SELECT-запросы, создавать и модифицировать индексы, триггеры и процедуры, образы;
- развивать познавательный интерес к базам данных и системам управления базами данных, интеллектуальные и творческие способности учащихся.

В результате изучения учебного предмета учащиеся должны знать на уровне представления:

- новые информационные технологии, применяемые в области СУБД;

знать на уровне понимания:

- этапы развития методов и средств обработки данных в информационных системах;
- современные СУБД реляционного типа;
- этапы проектирования информационных систем, основанных на реляционной модели данных;
- основные конструкции структурированного языка запросов;

− особенности проектирования и разработки документноориентированных БД;

− особенности создания многопользовательского приложения для работы с БД;

уметь:

− разрабатывать профессиональные проекты БД реляционного и документноориентированного типа;

− профессионально реализовывать разработанные проекты БД реляционного и документно-ориентированного типа с использованием современных СУБД;

− использовать основные конструкции структурированного языка запросов при реализации БД;

− разрабатывать многопользовательское приложение для работы с БД.

Общие методические рекомендации по выполнению домашней контрольной работы

Учащиеся-заочники выполняют две домашние контрольные работы. Основным методом изучения учебного предмета является самостоятельная работа, которая должна проводиться в последовательности, предусмотренной программой учебного предмета, и быть обязательно систематической.

Задания на домашнюю контрольную работу №1 разработаны в количестве 100 вариантов в соответствии с программой курса. Номера заданий выбираются в соответствии с двумя последними цифрами шифра учащегося, на пересечении соответствующей строки с соответствующим столбцом из таблицы 7.

Первым заданием является теоретический вопрос, на который нужно дать развернутый ответ, привести примеры. Объем – около двух-трех страниц.

Остальные задания – практические. В отчете подробно описать процесс создания базы данных (вставить код запроса, картинки выполнения запроса, картинки результата после выполнения запроса) и процесс выполнения всех практических заданий со вставкой изображений.

При оформлении домашней контрольной работы следует придерживаться следующих требований:

1 на проверку необходимо представить отчет по домашней контрольной работе на листах формата А4, а также в электронном виде;

2 в отчете следует применять шрифт Times New Roman (без переноса слов), размер шрифта 14 пт (для таблиц, допускается применять размер шрифта – 12 пт);

3 при расположении текста необходимо соблюдать межстрочные интервалы: - одинарный – интервал между заголовками раздела и подраздела, при оформлении текста; - двойной – для выделения рисунков, таблиц и формул (сверху и снизу);

4 на титульном листе указываются фамилия, имя, отчество учащегося, учебный предмет, номер работы, номер группы, шифр;

5 ответ следует начинать с номера и полного названия вопроса.

К домашней контрольной работе прикладывается диск с выполненными практическими заданиями.

Критерии оценки домашней контрольной работы

Домашняя контрольная работа, содержащая 75% положенного объема, оценивается «зачтено».

Домашняя контрольная работа будет не зачтена, если:

- выполнена не в соответствии с шифром учащегося;
- не раскрыто основное содержание теоретического вопроса (15%) и есть недочеты в практических заданиях (в сумме более 10%);
- не выполнена одна часть практического задания (10%) и есть недочеты в остальных заданиях (в сумме более 15%);
- есть существенные недочеты в нескольких заданиях (в сумме более 25%);
- отсутствует электронный вариант работы (диск с файлами).

Программа учебного предмета и методические рекомендации по его изучению

Введение

Цели и задачи учебного предмета, его связь с другими учебными предметами. Развитие методов и средств обработки данных в информационных системах (ИС). Понятие об информационных системах, банке данных и основных его компонентах. Новые информационные технологии, применяемые в области систем управления базами данных (СУБД)

Инструкция по охране труда для работающих на ПК

Литература: [4], с.20-24

Вопросы для самоконтроля

1 Дайте определение базы данных.
2 Дайте понятие информационная система.
3 Опишите основные этапы развития БД.
4 Опишите область применения БД.

Раздел 1 Основные концепции данных и реляционная модель данных

Тема 1.1 История развития представлений о базах данных

Ранние виды устройств внешней памяти. Функции и недостатки централизованных систем управления файлами.

Основные предпосылки возникновения новых подходов к управлению информацией.

Этапы развития методов и средств обработки данных в информационных системах.

Литература: [1], с.30-50; [4], с.36-72

Вопросы для самоконтроля

1 Раскройте суть первого этапа
2 Раскройте суть второго этапа
3 Раскройте суть третьего этапа
4 Раскройте суть четвертого этапа

Тема 1.2 Ранние подходы к организации систем управления базами данных. Типовая организация современной СУБД

Модели данных. Классификация моделей представления данных. Достоинства и недостатки иерархических и сетевых СУБД.

Новые информационные технологии, применяемые в области СУБД

Литература: [1], с.44-45; [4], с.36-72

Вопросы для самоконтроля

1 Дайте понятие модели данных.
2 Приведите классификацию моделей данных.
3 Опишите иерархическую модель данных.
4 Опишите сетевую модель данных.
5 Опишите реляционную модель данных.

Тема 1.3 Общие понятия реляционного подхода к организации БД

Современных СУБД реляционного типа. Основные компоненты реляционных БД. Определение таблицы, поля, записи, отношений. Первичный и внешний ключ. Виды связей между отношениями.

Литература: [4], с.90-103

Вопросы для самоконтроля

1 Раскройте принципы реляционного подхода к проектированию баз данных.
2 Приведите примеры реляционных СУБД.
3 Опишите назначение и область применения реляционных СУБД.
4 Дайте определение понятию домена, отношения, кортежа, первичного и возможного ключа отношения, мощности отношения и кардинального числа.
5 Опишите отличительные особенности и преимущества реляционного подхода.
6 Опишите основные правила построения реляционных баз данных.

Тема 1.4 Базисные средства манипулирования реляционными данными. Реляционная алгебра. Реляционное исчисление

Средства манипулирования данными. Основные операции реляционной алгебры: объединение, перечисление, разность, произведение, выбор, создание проекций, соединение, присвоение, деление. Описание реляционного исчисления

Литература: [1], с.212-250; [4], с.104-128

Вопросы для самоконтроля

1 Выскажите общее суждение о средствах манипулирования реляционными данными.

2 Охарактеризуйте операции реляционной алгебры.

3 Опишите особенности теоретико-множественных операций реляционной алгебры.

4 Опишите понятия кортежной переменной и правильно построенной формулы реляционного исчисления и охарактеризуйте их применение.

5 Опишите процедуру создания правильно построенной формулы.

6 Опишите процедуру использования кванторов общности и существования.

Тема 1.5 Общая методология проектирования БД. Проектирование ER-модели

Этапы проектирования информационных систем, основанных на реляционной модели данных.

Общая методология проектирования реляционных БД. Логическое и физическое проектирование ER-модель реляционной БД.

Литература: [1], с.278-346; [4], с.226-255

Вопросы для самоконтроля

1 Перечислите и кратко охарактеризуйте основные фазы (этапы) проектирования информационной системы, основанной на реляционной БД.

2 В чем заключается основная цель концептуального (логического) проектирования БД? Каков его главный результат?

3 Дайте определение ER-модели. Что такое сущность, атрибут, связь?

4 Какие типы связей (мощности) между сущностями вы знаете?

5 Что такое первичный ключ сущности? Какова его роль в ER-диаграмме и в последующей реляционной модели?

Тема 1.6 Проектирование реляционных БД с использованием нормализации

Процесс нормализация. Нормальные формы.

Процесс приведения реляционной БД к третьей нормальной форме

Литература: [1], с.278-346; [4], с.226-255

Вопросы для самоконтроля

1 Какова главная цель процесса нормализации базы данных?

2 Какие основные проблемы (аномалии) возникают в ненормализованных или плохо спроектированных таблицах?

3 Дайте определение функциональной зависимости между атрибутами.

4 Что такое полная функциональная зависимость и чем она отличается от частичной?

5 Что такое транзитивная зависимость?

Раздел 2 Внутренняя организация реляционной СУБД

Тема 2.1 Структуры внешней памяти. Методы организации индексов

Разновидности объектов во внешней памяти. Хранение отношений. Индексы. Журнальная информация. Служебная информация

Литература: [1], с.334-378; [4], с.128-129

Вопросы для самоконтроля

1 Охарактеризуйте структуры внешней памяти.

2 Опишите способы хранения данных во внешней памяти.

3 Опишите назначение индексов.

4 Охарактеризуйте методы организации индексов.

Тема 2.2 Понятие транзакции. Управление транзакциями. Сериализация транзакций. Методы сериализации транзакций

Транзакция и целостность БД. Изолированность пользователей. Сериализация транзакций. Синхронизационные захваты. Метод временных меток

Литература: [1], с.543-546; [4], с.557-583

Вопросы для самоконтроля

1 Раскройте понятие целостности данных.

2 Опишите процесс реализации целостности сущностей и ссылок.

3 Раскройте понятие транзакции.

4 Опишите процесс сериализации транзакций и построения сериального плана выполнения транзакций.

5 Раскрывает понятие блокировки.

6 Опишите процесс разрешения конфликтов транзакций при доступе к общим ресурсам.

Тема 2.3 Журнализация изменений БД

Журнализация и буферизация. Индивидуальный откат транзакции. Восстановление после мягкого сбоя. Физическая согласованность БД.

Восстановление после жесткого сбоя

Литература: [1], с.546-563; [4], с.563-564, 203

Вопросы для самоконтроля

1 Раскройте понятие журнализации изменений и журнала базы данных.

2 Дайте общую характеристику мягких, жестких и программных сбоев.

3 Опишите логическую и физическую структура журнала базы данных.

4 Опишите процедуры восстановления базы данных после мягкого, жесткого и программного сбоя.

Раздел 3 Работа с реляционными СУБД начального уровня

Тема 3.1 Особенности архитектуры клиент-сервер. Трехзвенная архитектура

Клиент и сервер локальных сетей. Системная архитектура клиент-сервер. Серверы БД и область их применения

Литература: [1], с.80-546; [4], с.270-272

Вопросы для самоконтроля

1 Что представляет из себя архитекрута клиент-сервер?

2 Перечислите основные преимущества архитектуры клиент-сервер

3 Дайте определение понятию CASE-средство

4 Перечислите компоненты клиент-серверной информационной системы

Тема 3.2 Функции и основные возможности языка SQL

Основные конструкции структурированного языка запросов (SQL)

Возможности SQL. Отличие SQL от процедурных языков программирования.

Составные части SQL. Типы данных SQL

Литература: [1], с.80-546; [4], с.270-272

Тема 3.3 Описание данных на основе SQL. Типы данных. Таблицы: создание, изменение, удаление

Организация запросов на языке SQL. Синтаксис команд CREATE DATABASE, USE. Типы данных. Синтаксис команд CREATE TABLE, ALTER TABLE, DROP TABLE.

Атрибуты на уровне поля, атрибуты на уровне таблицы.

Литература: [1], с.80-120; [4], с.272-291

Тема 3.4 Манипулирование данными

Синтаксис команд манипулирования данными (INSERT, DELETE, UPDATE)

Литература: [1], с.543-546; [4], с.304-312

Вопросы для самоконтроля

1 Охарактеризуйте и опишите синтаксис команды вставки данных в таблицу INSERT.

2 Охарактеризуйте и опишите синтаксис команды изменения данных в таблице UPDATE.

3 Охарактеризуйте и опишите синтаксис команды удаления данных из таблицы DELETE.

Тема 3.5 Администрирование БД. Выборка данных с использованием предложения SELECT

Простейшие SELECT-запросы. Операторы IN, BETWEEN, LIKE, WHERE, GROUP BY, HAVING.

Команды копирования и восстановления БД.

Литература: [1], с.80-120; [4], с.272-291

Вопросы для самоконтроля

1 Опишите область применения оператора выборки данных SELECT, его основные разделы: FROM, WHERE, GROUP BY, HAVING, ORDER BY.

2 Перечислите способы изменения и добавления записей в таблицу.

3 Опишите процесс объединения таблиц по заданному условию.

Тема 3.6 Оконные функции

Понятие оконных функций, партиций. Отличие оконных функций от агрегирующих функций.

Виды оконных функций: агрегирующие, ранжирующие, функции смещения. Способы их применения

Литература: [1], с.80-120; [4], с.272-291

Вопросы для самоконтроля

1 Дайте определение оконной функции в SQL.

2 Что делает ключевое слово OVER()?

3 Чем принципиально отличается работа оконной функции от работы агрегирующей функции с GROUP BY?

4 Как меняется количество строк в результирующем наборе в каждом случае?

5 Без каких секций (PARTITION BY, ORDER BY) может существовать окно? Приведите пример простейшей оконной функции.

Тема 3.7 Вложенные подзапросы. Формирование связанных подзапросов. Использование операторов EXISTS и FORALL

Вложенные подзапросы. Формирование связанных подзапросов. Использование операторов EXISTS и FORALL

Литература: [1], с.108-160; [4], с.292-227

Вопросы для самоконтроля

1 Опишите способы выборки данных из нескольких таблиц и представлений средствами SQL.

2 Опишите процедуры использования визуальных средств СУБД для создания вложенных подзапросов.

3 Опишите структуру многотабличного запроса.

4 Опишите процесс выборки данных из нескольких таблиц и представлений средствами SQL.

5 Классифицируйте визуальные средства создания вложенных подзапросов и их основные возможности.

Тема 3.8 Использование оператора UNION для соединения запросов. Объединение таблиц с использованием оператора JOIN

Оператор объединения UNION. Внешнее объединение.

Соединение таблиц с использованием оператора JOIN. Операции объединения таблиц посредством ссылочной целостности.

Литература: [1], с. 120-124

Вопросы для самоконтроля:

1 Объясните принцип работы оператора Union

2 Объясните принцип работы оператора Inner Join

3 Объясните принцип работы оператора Outer Join

Тема 3.9 Индексы: назначение, создание и удаление

Синтаксис команд назначения, создания, обновления и удаления индексов.

Литература: [4], с.52-56

Вопросы для самоконтроля:

1 Что такое индекс и для чего он необходим?
2 Приведите пример создания индекса
3 Приведите пример уникального индекса

Тема 3.10 Представления: создание, редактирование, удаление

Синтаксис команд создания, обновления и удаления представлений
Литература: [1], с. 139-140

Вопросы для самоконтроля

1 Дайте определение понятию представления и объясните его назначение
2 Приведите синтаксис создания представления
3 Приведите пример представления

Тема 3.11 Встроенные функции SQL. Оператор EXPLAIN

Встроенные функции: работа со строками и числами, математические, преобразование данных, работа с датой и временем. Работа оператора EXPLAIN.

Литература: [1], с. 792-794

Вопросы для самоконтроля

1 Приведите примеры встроенных функций для работы со строками
2 Приведите примеры встроенных функций для работы с числами
3 Приведите примеры встроенных функций для работы с датой и временем
4 Объясните назначение оператора EXPLAIN

Тема 3.12 Триггеры. Хранимые процедуры и пользовательские функции

Триггеры и их назначение. Синтаксис команд создания, изменения и удаления триггеров.

Хранимые процедуры и пользовательские функции, их назначение. Синтаксис создания, удаления и модификации хранимых процедур и пользовательских функций.

Литература: [4], с. 274-277

Вопросы для самоконтроля

1 Дайте определение понятию триггера и объясните его назначение

2 Приведите синтаксис создания триггера

4 Приведите пример создания триггера

Тема 3.13 Транзакции. Механизм транзакций

Режимы работы транзакций. Синтаксис установки параметров транзакции.

Литература: [4], с. 274-277

Вопросы для самоконтроля

1 Дайте определение транзакции в контексте СУБД.

2 Какая основная задача решается с помощью механизма транзакций?

3 Раскройте смысл акронима ACID. Кратко охарактеризуйте каждое из четырёх свойств транзакции

Тема 3.14 Изучение сред для работы с СУБД SQL с использованием графического интерфейса

Интерфейс сред MySQL Workbench.

Интерфейс среды SQLite Studio.

Литература: [4], с. 274-277

Вопросы для самоконтроля

1 Каковы основные преимущества использования графических сред (GUI) для работы с СУБД по сравнению с работой только через командную строку (CLI)?

2 Перечислите ключевые функциональные возможности, которые должны быть у современной графической среды для работы с SQL-базами данных (как минимум 4-5). Чем эти возможности помогают разработчику или администратору?

3 Что такое визуальное проектирование схемы БД (EER Diagram) и в каких инструментах (из изученных) эта возможность присутствует? Какую пользу приносит эта функция на этапах проектирования и сопровождения базы данных?

Список использованных источников

1 Бен-Ган, И. Microsoft SQL Server 2012. Основы T-SQL / И. Бен-Ган. М. : Эксмо, 2015. 400 с.

2 Дейт, К.Дж. Введение в системы баз данных / К.Дж. Дейт. М. : Вильямс, 2018. 1382 с.

3 Лазицкас, Е.А. Базы данных и системы управления базами данных / Е.А. Лазицкас, И.Н. Загуменникова, П.Г. Гилевский. Минск : РИПО, 2016. 268 с.

4 Федорова, Г. Разработка и администрирование баз данных / Г. Федорова. М. : Академия, 2015. 313 с.

5 SQL справочник / К. Кляйн [и др.]. СПб. : Символ-плюс, 2016. 56 с.

Задания на домашнюю контрольную работу по учебному предмету «Базы данных и системы управления базами данных» №1

1 Классифицируйте базы данных.

2 Назовите применение баз данных и систем управления базами данных в современных информационных системах.

3 Объясните понятие модели БД. Что называют иерархической и сетевой моделью? Охарактеризуйте реляционную модель данных.

4 Назовите основные структурные компоненты СУБД и их назначение.

5 Назовите назначение и область применения реляционных СУБД.

6 Дайте определение понятий домена, отношения, кортежа, первичного и возможного ключа отношения, мощности отношения и кардинального числа.

7 Что называют реляционной алгеброй? Укажите общую интерпретацию реляционных операций.

8 Назовите теоретико-множественные и специальные реляционные операции. Укажите особенности теоретико-множественных операций реляционной алгебры.

9 Что называют реляционным исчислением? Охарактеризуйте кортежные переменные и правильно построенные формулы.

10 Что называют реляционным исчислением кортежей? Что называют реляционным исчислением доменов?

11 Дайте определение понятию целостности данных. Назовите способы реализации ссылочной целостности. Охарактеризуйте целостность сущностей и ссылок.

12 Дайте определение понятию нормализации базы данных. Охарактеризуйте первую, вторую и третью нормальную форму.

13 Дайте определение понятию нормализации базы данных. Охарактеризуйте нормальную форму Бойса-Кодда.

14 Дайте определение понятию нормализации базы данных. Охарактеризуйте четвертую и пятую нормальную форму.

15 Охарактеризуйте хранение данных во внешней памяти как функцию СУБД.

16 Охарактеризуйте организацию структур внешней памяти для хранения данных в базе данных.

17 Охарактеризуйте индексы. Назовите назначение индексов. Перечислите методы организации индексов в современных СУБД. Охарактеризуйте индексирование отношений.

18 Охарактеризуйте буферизацию данных в оперативной памяти СУБД. Опишите управление буферами оперативной памяти.

19 Что называют транзакциями? Что называют сериализацией транзакций? Опишите сериальный план выполнения транзакций.

20 Что называют блокировкой? Опишите разрешение конфликтов транзакций при доступе к общим ресурсам.

21 Опишите журнал базы данных. Опишите логическую и физическую структуру журнала базы данных.

22 Охарактеризуйте мягкие, жесткие и программные сбои. Назовите процедуры восстановления базы данных после мягкого, жесткого и программного сбоя.

23 Что называют математическими и статистическими функциями?

24 Охарактеризуйте строковые функции, функции работы с датами, логические функции.

25 Назовите функции и команды управления базой данных.

26 Назовите функции управления средой, ввода-вывода и общего назначения.

27 Назовите команды языка SQL для выборки и изменения записей таблиц, добавления записей в таблицу, объединения таблиц.

28 Что называют многотабличным запросом?

29 Охарактеризуйте синтаксис команд вставки, изменения и удаления записей из таблиц базы данных.

30 Опишите выборку данных из нескольких таблиц и представлений средствами SQL.

Практическое задание

Создайте базу данных на язуке MySQL, выполнение всех практических заданий подробно описать в текстовом редакторе со вставкой изображений созданных объектов. Выполненное практическое задание представить в печатном виде и на диске.

База данных «Колледж» включает следующие таблицы. «Учащиеся» - содержит сведения об учащихся. «Преподаватели» - содержит сведения о преподавателях. «Сессия» - содержит данные об итогах сессии. «Общежитие» - содержит информацию об адресах учащихся. «Экзамены» - содержит коды и названия экзаменов. «Стипендия» - содержит информацию о стипендии учащихся.

Структура таблиц БД представлена ниже в таблицах 1-6.

Таблица 1 - Учащиеся

Имя поля	Тип	Длина	Индекс
Код учащегося	Числовой	6	Да
Группа	Текстовый	4	
Фамилия Имя	Текстовый Текстовый	20 20	
Отчество Курс	Текстовый Числовой	20	
№ зачкн	Числовой		
Пол	Текстовый	1	
Дата рождения	Дата/время		

Таблица 2 – Сессия

Имя поля	Тип	Длина	Индекс
Код учащегося	Числовой	6	Да
Код преподавателя	Числовой		Да
Код экзамена	Числовой	20	
Оценка	Числовой		
Дата экзамена	Дата/время		

Таблица 3 - Преподаватели

Имя поля	Тип	Длина	Индекс
Код преподавателя	Числовой		Да
Фамилия	Текстовый	20	
Имя	Текстовый	20	
Отчество	Текстовый	20	
Дисциплина	Текстовый	20	
Адрес	Текстовый	40	
Рабочий телефон	Текстовый	8	
Домашний телефон	Текстовый	8	
Дата рождения	Дата/время		

Таблица 4 - Экзамены

Имя поля	Тип	Длина	Индекс
Код экзамена	Числовой		Да
Название	Текстовый	20	

Таблица 5 - Общежитие

Имя поля	Тип	Длина	Индекс
Код учащегося	Числовой		Да
Проживание в общежитии	Текстовый	3	
Общежитие	Числовой		
Комната	Числовой		

Таблица 6 - Стипендия

Имя поля	Тип	Длина	Индекс
Код учащегося	Числовой		Да
Сентябрь	Денежный		
Октябрь	Денежный		
Ноябрь	Денежный		
Декабрь	Денежный		
Январь	Денежный		
Февраль	Денежный		
Март	Денежный		
Апрель	Денежный		
Май	Денежный		
Июнь	Денежный		
Июль	Денежный		
Август	Денежный		

Связать таблицы, заполнить данными все таблицы (не менее 10 записей). Создать запросы, форму, отчет. В отчете по выполнению практического задания привести код запроса, результат его выполнения.

Запросы:

31 Создайте запрос, выбирающий из таблицы «Учащиеся» записи с информацией об учащихся определенной группы. Проведите сортировку отобранных записей по фамилиям учащихся, а затем по именам.

32 Извлеките из таблицы «Учащиеся» записи, содержащие сведения о девушках из трех определенных групп. Произведите сортировку отобранных записей по номеру группы, а затем по фамилиям учащихся.

33 Найдите учащихся, фамилии которых состоят из 8 букв и заканчиваются буквой "й".

34 Найдите учащихся, фамилии которых начинаются с буквы "С" и заканчиваются буквой "в".

35 Найдите учащихся, фамилии которых начинаются с букв «А-К» и состоят из 7 букв.

36 Найдите учащихся, фамилии которых не начинаются с букв «М» и «Д».

37 Найдите девушек, родившихся в 2000 году.

38 Найдите юношей, родившихся зимой 1999 года.

39 Найдите учащихся, родившихся летом и осенью 1999 года.

40 Найдите учащихся, родившихся до I сентября 1999 года и проживающих в общежитии.

41 Найдите преподавателей, которые родились в 70-е годы и имеют домашний телефон.

42 Найдите преподавателей, которые старше 40 лет, фамилии которых начинаются с буквы "П".

43 Найдите девушек, которые имеют не более одной 6.

44 Найдите, юношей, которые имеют хотя бы одну 9.

45 Найдите учащихся, которые получили пятерку по информатике и английскому языку.

46 Найдите учащихся, которые сдали сессию на все 9.

47 Найдите самого молодого учащегося в определенной группе.

48 Найдите самую молодую девушку на 2 курсе.

49 Найдите самого старшего учащегося на каждом курсе.

50 Найдите учащихся, которые учатся в одной группе с Петровым.

51 Найдите учащихся, которые получили ту же оценку по информатике, что и Петров.

52 Найдите учащихся, которые имеют средний балл выше, чем Петров.

53 Найдите учащихся, которые сдали сессию так же, как и Петров.

54 Найдите фамилии и оценки жильцов определенной комнаты общежития 1.

55 Найдите фамилии и адреса девушек определенной группы, родившихся зимой 1999 года.

56 Найдите фамилии и оценки юношей определенной группы, родившихся осенью 2000 года.

57 Найдите фамилии учащихся 2 курса, которые имеют повышенную стипендию.

58 Найдите фамилии учащихся 3 курса, которые не имеют стипендии.

59 Найдите фамилии преподавателей, принимающих экзамены на 1 курсе.

60 Найдите фамилии преподавателей, принимающих экзамены по информатике.

61 Создайте запрос «Группы N курса», содержащий список учебных групп. Номер курса - параметр запроса.

62 Создайте запрос «Список N группы», содержащий отсортированные фамилии и имена учащихся группы. Номер группы берется из поля «Группа» загруженной формы «Список учащихся по группам».

63 Найдите всех девушек данной группы. Номер группы - параметр запроса.

64 Найдите учащихся, фамилии которых начинаются с данной буквы. Буква - параметр запроса.

65 Найдите фамилии учащихся определенной группы, родившихся в данном месяце. Номер месяца - параметр запроса.

66 Найдите фамилии юношей данной группы, родившихся в данном году. Номер группы и год – параметры запроса.

67 Найдите фамилии учащихся, имеющих средний балл больше заданного числа X. Число X - параметр запроса.

68 Найдите оценки данного учащегося. Фамилия учащегося - параметр запроса.

69 Найдите учащихся, у которых принимал экзамен данный преподаватель. Вывести фамилии, номера их групп и оценки. Код преподавателя - параметр запроса.

70 Создайте запрос «Оценки», подсчитывающий количество различных оценок по каждому экзамену.

71 Определите количество учащихся в каждой группе.

72 Определите количество девушек в каждой группе.

73 Определите количество юношей и девушек в группах N1- N 3 и N 5.

74 Определите количество учащихся на курсе. Номер курса - параметр запроса.

75 Найдите средний балл по дисциплине. Наименование дисциплины - параметр запроса.

76 Найдите средний балл по информатике в N группе. Номер группы - параметр запроса.

77 Найдите средний балл по дисциплине на всем курсе. Наименование дисциплины - параметр запроса.

78 Найдите номера комнат, в которых все жильцы сдали дисциплину на 8 или 9. Наименование дисциплины - параметр запроса.

79 Найдите номера комнат, в которых живет три учащихся.

80 Найти номера комнат, в которых все жильцы родились в одном году.

81 Найдите номера комнат, в которых все жильцы учатся в одной группе.

82 Найдите число комнат, в которых средний балл жильцов по информатике больше 4.

83 Найдите номера комнат, в которых живет четыре девушки.

84 Найдите учащихся, сдавших хотя бы один экзамен позднее определенной даты. Вывести их фамилии, а также названия и дату сдачи таких экзаменов. Дата- параметр запроса.

85 Найдите суммарную стипендию в каждой группе за сентябрь-декабрь.

86 Найдите число учащихся в каждой группе, получивших в декабре стипендию.

87 Найдите номера групп, в которых учится более 7 девушек.

88 Найдите фамилии учащихся, имеющих сумму баллов больше 20.

89 Найдите группы, в которых средний балл по информатике больше 6.

90 Найдите в каждой группе число учащихся, у которых принял экзамен тот или иной преподаватель. Код экзамена - параметр запроса.

91 Найдите учащихся, получивших по данному экзамену оценку; выше средней в группе. Код экзамена и номер группы - параметры запроса.

92 Найдите десять лучших (имеющих максимальную сумму баллов) учащихся курса. Номер курса - параметр запроса.

93 Создайте запрос «Лучшие в группах» со списком учащихся, каждый из которых является лучшим в своей группе.

94 Определите число учащихся в каждой группе, живущих в общежитии.

95 Определите число учащихся по курсам, не живущих в общежитии.

96 Определите фамилии учащихся, не живущих в общежитии.

97 Найдите всех учащихся, имеющих однофамильцев.

98 Создайте список учащихся -однофамильцев, содержащий их фамилии, имена и номера групп.

99 Найдите число учащихся, сдававших данный экзамен. Код экзамена - параметр запроса.

100 Определите сколько учащихся в каждой группе не явилось хотя бы на один экзамен.

101 Найдите фамилии учащихся, не явившихся хотя бы на один экзамен.

102 Создайте список учащихся, не получавших в январе стипендию. Отсортировать его по номерам групп, а затем по фамилиям.

103 Найдите фамилии учащихся, не получавших стипендию в течение 1 семестра.

104 Найдите фамилии учащихся, получивших по данному экзамену «8». Код экзамена - параметр запроса.

105 Найдите фамилии учащихся, получивших у данного преподавателя «9». Код преподавателя – параметр запроса.

106 Найдите группы, имеющие средний балл по информатике выше, чем средний балл по информатике на курсе.

107 Найдите группу, имеющую наибольший средний балл на курсе. Номер курса - параметр запроса.

108 Найдите группу, имеющую наибольший средний балл по дисциплине. Наименование дисциплины - параметр запроса.

109 Найдите группы, сдавшие английский язык лучше (по среднему баллу), чем N1 группа.

110 Найдите группу с наименьшим числом двоек по данному экзамену. Код экзамена - параметр запроса.

111 Найдите распределение по группам юношей, родившихся в данном году. Год - параметр запроса.

112 Найдите в данной группе распределение оценок по информатике. Номер группы - параметр запроса.

113 Найдите в данной группе распределение учащихся по месяцу рождения. Номер группы - параметр запроса.

114 Найдите в определенной группе учащихся, которые будут получать повышенную стипендию (сдали сессию на 9 и 10 и имеют средний балл больше 9,1).

115 Найдите самого «доброго» преподавателя (имеющего максимальное среднее из поставленных оценок) по данной дисциплине. Наименование дисциплины - параметр запроса.

116 Найдите самого «злого» преподавателя (поставившего наибольшее число двоек) по данной дисциплине. Наименование дисциплины - параметр запроса.

117 Постройте запрос «Стипендиальная ведомость групп курса» с полями «№ зач.кн», «ФИО» и «Стипендия» по итогам сессии. Значение поля «Стипендия» равно 500000, если учащийся имеет средний балл больше 6; в противном случае поле содержит текст «нет стипендии». Номер курса - параметр запроса.

118 Постройте запрос «Стипендиальная ведомость группы» с полями «№ зачкн», «ФИО» и «Стипендия» по итогам сессии. Значение поля «Стипендия» равно 800000, если учащийся сдал сессию без 4, и ничего не содержит в противном случае. Номер группы - параметр запроса.

119 Постройте запрос «Информация о стипендии в группе» с полями «№ зач.кн», «Учащийся» и «Стипендия» по итогам сессии. Поле «Учащийся» содержит фамилию и имя учащегося. Поле «Стипендия» ничего не содержит, если учащийся получил хотя бы одну 4; содержит текст «повышенная», если учащийся сдал сессию на 9 и 10 и имеет средний балл больше 9,1; и текст «обычная» - в остальных случаях. Номер группы – параметр запроса.

120 Постройте запрос «Стипендия на курсе в феврале» с полями «Группа», «№ зачкн», «Фамилия» и «Стипендия» по итогам сессии. Поле «Стипендия» ничего не содержит, если учащийся получил хотя бы одну 4; содержит число 800000, если учащийся сдал сессию на 9 и 10, содержит число 660000 - в остальных случаях.

121 Создайте перекрестный запрос, дающий распределение учащихся в группах по году рождения.

122 Создайте перекрестный запрос «Распределение оценок в группе», подсчитывающий количество различных оценок в данной группе по каждому экзамену. Названия строк - экзамены. Названия столбцов - оценки. Номер группы - параметр запроса.

123 Создайте перекрестный запрос «Распределение оценок по информатике», подсчитывающий количество различных оценок в

группах по информатике. Названия строк - номера групп. Названия столбцов - оценки.

124 Создайте перекрестный запрос «Распределение оценок по экзамену», подсчитывающий количество различных оценок в группах по данному экзамену. Названия строк - номера групп. Названия столбцов -оценки. Код экзамена - параметр запроса.

125 Создайте перекрестный запрос, подсчитывающий для данного экзамена среднюю оценку по группам для каждого преподавателя. Названия строк - фамилии преподавателей. Названия столбцов - номера групп. Формат вывода среднего балла - два десятичных знака после запятой. Код экзамена - параметр запроса.

126 Создайте перекрестный запрос «Средние баллы в группах», дающий значения среднего балла в группах по каждому экзамену. Названия строк - экзамены. Названия столбцов - номера групп. Формат вывода среднего балла - два десятичных знака после запятой.

127 Создайте перекрестный запрос «Итоги сессии на курсе», дающий значения среднего балла в группах. Названия строк - номера групп. Названия столбцов - «Средний балл» и названия экзаменов. Формат вывода среднего балла - два десятичных знака после запятой.

128 Создайте перекрестный запрос, с информацией об оценках учащихся, живущих в одной комнате. Названия столбцов - экзамены. Названия строк - фамилии жильцов. Номер комнаты - параметр запроса.

129 Создайте перекрестный запрос «Итоги сессии в группе», содержащий поля «ФИО», «№ зачкн» и названия экзаменов. Номер группы - параметр запроса.

130 Создайте перекрестный запрос, содержащий поля «ФИО», «Средний балл» и названия экзаменов. Номер группы - параметр запроса.

131 Создайте запрос, добавляющий в таблицу «Учащиеся» новое поле «Стипендия». Значение этого поля равно 800000, если учащийся сдал сессию на 9 10, 660000, если учащийся сдал сессию без оценок 0-3 и ничего не содержит в противном случае.

132 Создайте запрос, добавляющий информацию из таблицы «Новые учащиеся» в таблицу «Учащиеся». Таблица «Новые учащиеся» содержит сведения о новых учащихся и имеет такую же структуру, что и таблица «Учащиеся».

133 Создайте запрос, добавляющий информацию из таблицы «Адреса новых учащихся» в таблицу «Общежитие». Таблица «Адреса новых учащихся» имеет такую же структуру, что и таблица «Общежитие» и содержит сведения о размещении новых учащихся.

134 Создайте запрос, добавляющий информацию из таблицы «Новые преподаватели» в таблицу «Преподаватели». Таблица «Новые преподаватели» содержит сведения о новых учащихся и имеет такую же структуру, что и таблица «Преподаватели».

135 Постройте запрос на создание таблицы «Список N1 группы», содержащей список учащихся N1 группы.

136 Постройте запрос на создание таблицы «Список групп», содержащей номера учебных групп.

137 Постройте запрос на создание таблицы «Список экзаменов», содержащей названия экзаменов в сессии.

138 Постройте запрос на создание таблицы «Итоги сессии», содержащей средние баллы в группах по каждому экзамену.

139 Постройте запрос на создание таблицы «Список отличников», содержащей фамилии и номера групп учащихся, сдавших сессию на 9 и 10.

140 Постройте запрос на создание таблицы «Список двоечников», содержащей коды, фамилии и номера групп учащихся, получивших в сессию хотя бы одну двойку или тройку.

141 Постройте запрос на создание таблицы «Подлежат отчислению», содержащей коды, фамилии и номера групп учащихся, получивших в сессию более одной неудовлетворительной оценки (ниже 4).

142 Постройте запрос на создание таблицы «Итоги сдачи информатики в группе», содержащей поля «№ зачкн», «ФИО учащийся», «Оценка» и «Экзаменатор». Записи таблицы должны быть отсортированы по полю «ФИО учащийся» Номер группы -параметр запроса.

143 Постройте запрос на создание таблицы «Бланк экзаменационной ведомости группы», содержащей поля «№ зачкн». «ФИО» с информацией из таблицы «Учащиеся», а также пустые поля «Оценка» и «Подпись». Номер группы параметр запроса. Записи таблицы должны быть отсортированы по полю «ФИО».

144 Постройте запрос на удаление из таблицы «Учащиеся» записи об отчисленном учащемся (фамилия учащегося - параметр запроса).

145 Постройте запрос на удаление из таблицы «Учащиеся» записей об учащихся, ФИО которых содержатся в поле «ФИО» таблицы «Подлежат отчислению».

146 Постройте запрос на удаление из таблицы «Учащиеся» записей об учащихся, коды которых содержатся в поле «Код учащегося» таблицы «Подлежат отчислению».

147 Постройте запрос на удаление из таблицы «Учащиеся» записей об учащихся, получивших более одной двойки в сессию.

148 Постройте запрос на удаление из таблицы «Стипендия» записей об учащихся, не получавших стипендию в течение всего года.

149 Постройте запрос на обновление таблицы «Стипендия», увеличивающий январскую стипендию учащихся на 10%.

150 Жильцы 22 комнаты общежития 3 переселились в 46 комнату общежития 2. Используя запрос на обновление, внесите соответствующие изменения в таблицу «Общежитие».

Методические рекомендации по выполнению домашней контрольной работы №1

Методические рекомендации по выполнению заданий 31-150

MySQL представляет систему управления реляционными базами данных (СУБД). На сегодняшний день это одна из самых популярных систем управления базами данных.

Изначальным разработчиком данной СУБД была шведская компания MySQL AB. В 1995 году она выпустила первый релиз MySQL. В 2008 году компания MySQL AB была куплена компание Sun Microsystems, а в 2010 году уже компания Oracle поглотила Sun и тем самым приобрела права на торговую марку MySQL. Поэтому MySQL на сеголняшней день развивается под эгидой Oracle.

MySQL обладает кроссплатформенностью, имеются дистрибутивы под самые различные ОС, в том числе наиболее популярные версии Linux, Windows, MacOS.

При установке сервера MySQL устанавливается консольный клиент для работы с базами данных. Например, в Windows в меню Пуск можно найти программу MySQL Command Line Client, как показано на рисунке 1. Это и есть собственно консольный клиент:

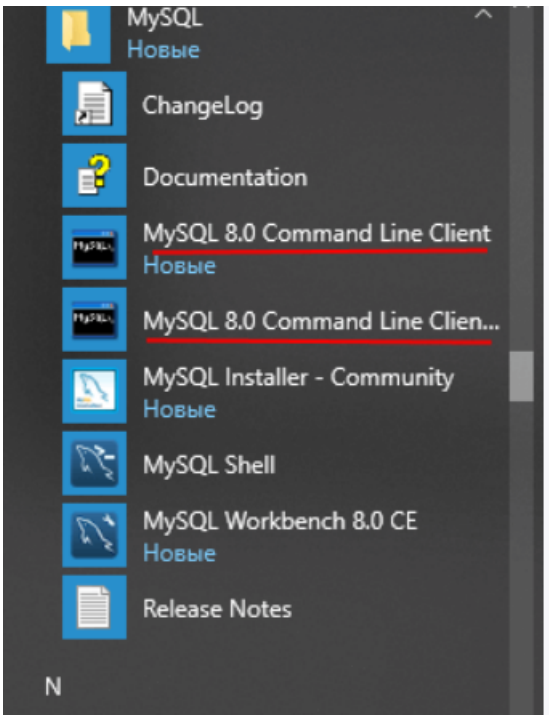


Рисунок 1 – Консольный клиент MySQL

При запуске MySQL Command Line Client вначале отобразится предложение ввести пароль. Здесь необходимо ввести пароль, который был установлен при установке MySQL для пользователя root. И после удачного подключения можно будет отправлять серверу команды через данный консольный клиент.

Для начала посмотрим, какие базы данные есть на сервере по умолчанию. Для этого введем команду show databases, как показано на рисунке 2.

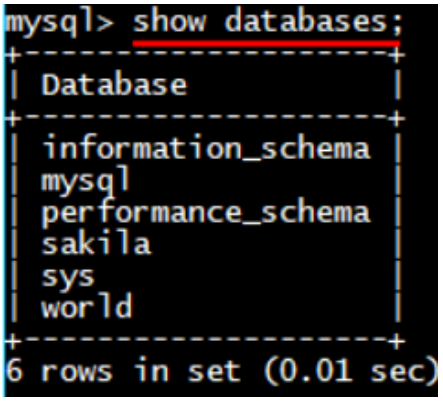


Рисунок 2 – Команда show databases;

После выполнения этой команды мы увидим, что на сервере по умолчанию уже есть ряд баз данных, которые выполняют административные функции.

Для создания базы данных используется команда CREATE DATABASE. Она имеет следующий синтаксис:

CREATE DATABASE [IF NOT EXISTS] имя_базы_даных;

В конце команды указывается имя базы данных.

Первая форма CREATE DATABASE имя_базы_даных пытается создать базу данных, но если такая база данных уже существует, то операция возвратит ошибку.

Вторая форма CREATE DATABASE IF NOT EXISTS имя_базы_даных пытается создать базу данных, если на сервере отсутствует бд с таким именем.

Например, в MySQL Command Line Client (или в MySQL Workbench CE) выполним следующую команду:

CREATE DATABASE productsdb;

Она создаст на сервере БД productsdb.

После создания БД с ней производятся различные операции: создание таблиц, добавление и получение данных и т.д. Но, чтобы производить эти операции, надо установить определенную базу данных в качестве используемой. Для этого применяется оператор USE:

USE productsdb;

4.2 Создание и удаление таблиц

Для создания таблиц используется команда CREATE TABLE. Эта команда применяет ряд операторов, которые определяют столбцы таблицы и их атрибуты. Общий формальный синтаксис команды CREATE TABLE:

CREATE TABLE название_таблицы
(название_столбца1 тип_данных атрибуты_столбца1,
название_столбца2 тип_данных атрибуты_столбца2,
название_столбцаN тип_данных атрибуты_столбцаN,
атрибуты_уровня_таблицы)

После команды CREATE TABLE идет название таблицы. Имя таблицы выполняет роль ее идентификатора в базе данных, поэтому оно должно быть уникальным. Затем в скобках перечисляются названия столбцов, их типы данных и атрибуты. В самом конце можно определить атрибуты для всей таблицы. Атрибуты столбцов, а также атрибуты таблицы указывать необязательно.

Создадим простейшую таблицу. Для этого выполним следующий скрипт:

```
CREATE DATABASE productsdb;
 USE productsdb;
 CREATE TABLE Customers
(
   Id INT,
   Age INT,
   FirstName VARCHAR(20),
   LastName VARCHAR(20));
```

Таблица не может создаваться сама по себе. Она всегда создается в определенной базе данных. Вначале здесь создается база данных productsdb. И затем, чтобы указать, что все дальнейшие операции, в том числе создание таблицы, будут производиться с этой базой данных, применяется команда USE.

Далее собственно идет создание таблицы, которая называется Customers. Она определяет четыре столбца: Id, Age, FirstName, LastName. Первые два столбца представляют идентификатор клиента и его возраст и имеют тип INT, то есть будут хранить числовые значения. Следующие столбцы представляют имя и фамилию клиента и имеют тип VARCHAR(20), то есть представляют строку длиной не более 20 символов. В данном случае для каждого столбца определены имя и тип данных, при этом атрибуты столбцов и таблицы в целом отсутствуют.

Если после создания таблицы мы захотим ее переименовать, то для этого нужно использовать команду RENAME TABLE, которая имеет следующий синтаксис:

```
RENAME TABLE старое_название TO новое_название;
```

Например, переименуем таблицу Customers в Clients:

```
RENAME TABLE Customers TO Clients;
```

Для полного удаления данных, очистки таблицы применяется команда TRUNCATE TABLE. Например, очистим таблицу Clients:

```
TRUNCATE TABLE Clients;
```

Для удаления таблицы из БД применяется команда DROP TABLE, после которой указывается название удаляемой таблицы. Например, удалим таблицу Clients:

```
DROP TABLE Clients;
```

Для предоставления информации о столбцах таблицы используется команда DESCRIBE. Формальный синтаксис команды:

```
{DESCRIBE | DESC} tbl_name {col_name | wild}
```

Команда DESCRIBE представляет собой сокращенный вариант команды SHOW COLUMNS FROM. Параметр col_name может содержать имя столбца или строки, включающей такие групповые символы SQL, как `%' и `_' (шаблонные символы, позволяющие получить информацию о всех подходящих столбцах). Следует отметить, что типы столбцов в полученном описании могут отличаться от ожидаемых, первоначально заданных командой CREATE TABLE при создании таблицы, поскольку MySQL иногда изменяет типы столбцов.

Если таблица уже была ранее создана, и ее необходимо изменить, то для этого применяется команда ALTER TABLE. Ее сокращенный формальный синтаксис:

```
ALTER TABLE название_таблицы
{ ADD название_столбца тип_данных_столбца [атрибуты_столбца] |
  DROP COLUMN название_столбца |
        MODIFY COLUMN название_столбца тип_данных_столбца [атрибуты_столбца] |
          ALTER COLUMN название_столбца SET DEFAULT значение_по_умолчанию |
  ADD [CONSTRAINT] определение_ограничения |
  DROP [CONSTRAINT] имя_ограничения}
```

Например, существует таблица Customers, содержащая столбцы Id, Age, FirstName, LastName. Для вывода информации о таблице можно воспользоваться командой DESCRIBE, как показано на рисунке 3.

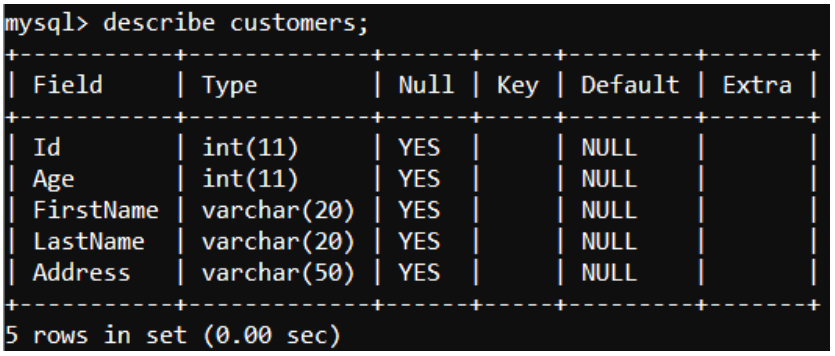
```
mysql> describe customers;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Id        | int(11)     | YES  |     | NULL    |       |
| Age       | int(11)     | YES  |     | NULL    |       |
| FirstName | varchar(20) | YES  |     | NULL    |       |
| LastName  | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

Рисунок 3 – Результат выполнения команды DESCRIBE

Добавим в таблицу Customers новый столбец Address:

```
ALTER TABLE Customers
ADD Address VARCHAR(50) NULL;
```

Результат выполнения команды DESCRIBE после добавления столбца представлен на рисунке 4.



```
mysql> describe customers;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Id        | int(11)     | YES  |     | NULL    |       |
| Age       | int(11)     | YES  |     | NULL    |       |
| FirstName | varchar(20) | YES  |     | NULL    |       |
| LastName  | varchar(20) | YES  |     | NULL    |       |
| Address   | varchar(50) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Рисунок 4 – Результат добавления столбца в таблицу

В данном случае столбец Address имеет тип VARCHAR и для него определен атрибут NULL.

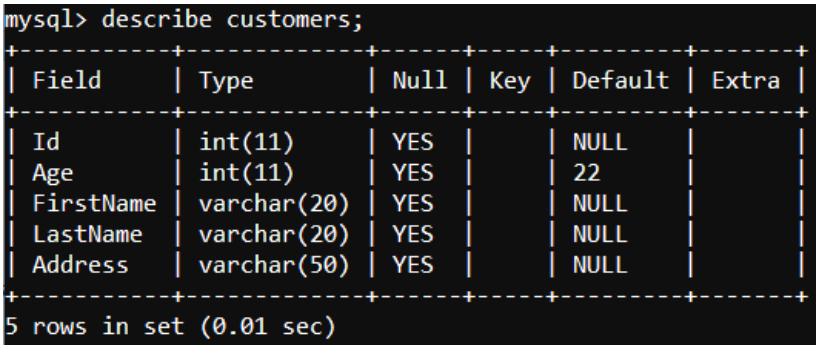
Удалим столбец Address из таблицы Customers:

```
ALTER TABLE Customers
DROP COLUMN Address;
```

Установим в таблице Customers для столбца Age значение по умолчанию 22:

```
ALTER TABLE Customers
ALTER COLUMN Age SET DEFAULT 22;
```

Результат выполнения команды DESCRIBE после изменения столбца Age представлен на рисунке 5.



```
mysql> describe customers;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Id        | int(11)     | YES  |     | NULL    |       |
| Age       | int(11)     | YES  |     | 22      |       |
| FirstName | varchar(20) | YES  |     | NULL    |       |
| LastName  | varchar(20) | YES  |     | NULL    |       |
| Address   | varchar(50) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

Рисунок 5 – Результат изменения столбца Age

Изменим в таблице Customers тип данных у столбца FirstName на CHAR(100) и установим для него атрибут NULL:

```
ALTER TABLE Customers
MODIFY COLUMN FirstName CHAR(100) NULL;
```

Результат выполнения команды DESCRIBE после изменения столбца FirstName представлен на рисунке 6.

```
mysql> alter table Customers modify column FirstName char(100) null;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe customers;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Id        | int(11)     | YES  |     | NULL    |       |
| Age       | int(11)     | YES  |     | 22      |       |
| FirstName | char(100)   | YES  |     | NULL    |       |
| LastName  | varchar(20) | YES  |     | NULL    |       |
| Address   | varchar(50) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

Рисунок 6 – Результат изменения столбца FirstName

Внешние ключи позволяют установить связи между таблицами. Внешний ключ устанавливается для столбцов из зависимой, подчиненной таблицы, и указывает на один из столбцов из главной таблицы. Как правило, внешний ключ указывает на первичный ключ из связанной главной таблицы.

Общий синтаксис установки внешнего ключа на уровне таблицы:

```
 [CONSTRAINT имя_ограничения]
FOREIGN KEY (столбец1, столбец2, ... столбецN)
REFERENCES главная_таблица (столбец_главной_таблицы1, столбец_главной_таблицы2, ... столбец_главной_таблицыN)
[ON DELETE действие]
[ON UPDATE действие]
```

Для создания ограничения внешнего ключа после FOREIGN KEY указывается столбец таблицы, который будет представляет внешний ключ. А после ключевого слова REFERENCES указывается имя связанной таблицы, а затем в скобках имя связанного столбца, на который будет указывать внешний ключ. После выражения REFERENCES идут выражения ON DELETE и ON UPDATE, которые задают действие при удалении и обновлении строки из главной таблицы соответственно.

Например, определим две таблицы и свяжем их посредством внешнего

ключа:

```
CREATE TABLE Customers
(
   Id INT PRIMARY KEY AUTO_INCREMENT,
   Age INT,
   FirstName VARCHAR(20) NOT NULL,
   LastName VARCHAR(20) NOT NULL,
   Phone VARCHAR(20) NOT NULL UNIQUE
);
CREATE TABLE Orders
(
   Id INT PRIMARY KEY AUTO_INCREMENT,
   CustomerId INT,
   CreatedAt Date,
   FOREIGN KEY (CustomerId) REFERENCES Customers (Id)
);
```

Для добавления новых записей в таблицу мы используем SQL выражение INSERT INTO. Общий вид выражения имеет следующий вид:

```
INSERT INTO имя_таблицы (колонка1, колонка2 ...)
VALUES (значение1, значение2 ...);
```

Пример:

Предположим, что есть пустая таблица developers, представленная на рисунке 7

```
mysql> desc developers;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| ID         | int(11)     | NO   | PRI | NULL    |       |
| NAME       | varchar(100)| NO   |     | NULL    |       |
| SPECIALTY  | varchar(100)| YES  |     | NULL    |       |
| EXPERIENCE | int(11)     | NO   |     | NULL    |       |
| SALARY     | int(11)     | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Рисунок 7 – Таблица developers

Теперь попробуем добавить в нашу таблицу пять записей. Для этого мы выполним следующие команды:

```
INSERT INTO developers (ID, NAME, SPECIALTY, EXPERIENCE, SALARY)
 VALUES (1, 'Eugene Suleimanov', 'Java', 2, 2000);

INSERT INTO developers (ID, NAME, SPECIALTY, EXPERIENCE, SALARY)
VALUES (2, 'Peter Romanenko', 'C++', 3, 3500);

INSERT INTO developers (ID, NAME, SPECIALTY, EXPERIENCE, SALARY)
VALUES (3, 'Andrei Komarov', 'JavaScript', 2, 2100);

INSERT INTO developers (ID, NAME, SPECIALTY, EXPERIENCE, SALARY)
VALUES (4, 'Konstantin Geiko', 'C#', 2, 2000);

INSERT INTO developers (ID, NAME, SPECIALTY, EXPERIENCE, SALARY)
VALUES (5, 'Asya Suleimanova', 'UI/UX', 2, 1800);
```

Мы также можем добавить шестую запись, используя синтаксис, приведённый ниже:

```
INSERT INTO developers
VALUES (6, 'Ludmila Geiko', 'UI/UX', 2, 1800);
```

После поочерёдного выполнения данного запроса таблица будет выглядеть как показано на рисунке 8.

```
mysql> SELECT * FROM developers;
+----+------------------+------------+------------+--------+
| ID | NAME             | SPECIALTY  | EXPERIENCE | SALARY |
+----+------------------+------------+------------+--------+
|  1 | Eugene Suleimanov | Java       |          2 |   2000 |
|  2 | Peter Romanenko   | C++        |          3 |   3500 |
|  3 | Andrei Komarov    | JavaScript |          2 |   2100 |
|  4 | Konstantin Geiko  | C#         |          2 |   2000 |
|  5 | Asya Suleimanova  | UI/UX      |          2 |   1800 |
|  6 | Ludmila Geiko     | UI/UX      |          2 |   1800 |
+----+------------------+------------+------------+--------+
6 rows in set (0.00 sec)
```

Рисунок 8 – Таблица после заполнения данными

UPDATE — это команда, которая обновляет данные в таблице. Ее общий синтаксис такой:

```
UPDATE [table] table_name
SET column1 = value1, column2 = value2, ...
[WHERE condition]
[ORDER BY expression [ ASC | DESC ]]
[LIMIT number_rows];
```

Сначала мы указываем обязательные параметры: название таблицы, названия колонок и нужные значения для обновления. Обратите внимание, что в MySQL можно использовать ключевое слово table (update table), а можно его опустить и сразу указать название таблицы.

Затем идут необязательные блоки WHERE (условие обновления), ORDER BY (сортировка) и LIMIT (ограничение количества обновляемых записей).

Мы будем рассматривать работу с командой UPDATE на примере простой схемы БД. Представим, что есть сеть магазинов бытовой техники. База данных состоит из 3 таблиц:

– categories: таблица категорий товаров. В ней хранятся только идентификаторы и названия категорий (рисунок 9).

– stores: таблица магазинов. В ней хранятся идентификаторы, названия магазинов, город и адрес(рисунок 10).

– products: таблица товаров. В ней хранятся идентификаторы, названия товаров, ссылки на категорию товара и магазин, цена товара и его количество(рисунок 11).

id	name
1	Ноутбуки
2	Планшеты
3	Телефоны

Рисунок 9 – Таблица «categories»

id	name	city	address
1	Магазин 1	Санкт-Петербург	Цветочная 63
2	Магазин 2	Санкт-Петербург	Советская 20
3	Магазин 3	Москва	Берзарина 42
4	Магазин 4	Москва	Авиамоторная 96

Рисунок 10 – Таблица «stores»

id	name	category_id	store_id	price	quantity
1	Ноутбук 1	1	1	50000	1
2	Ноутбук 2	1	3	79000	2
3	Планшет 1	2	2	8000	5
4	Планшет 2	2	4	12000	3
5	Телефон 1	3	1	18000	5
6	Телефон 2	3	2	25000	1
7	Телефон 3	3	3	78000	1

Рисунок 11 – Таблица «products»

Допустим, необходимо изменить стоимость одного конкретного товара. Для этого указываем имя MySQL таблицы (update products), название обновляемого поля и значение (set price = 50500), а также условие, какую именно стоку нужно обновить (where id = 1):

```
UPDATE products
SET price = 50500
WHERE id = 1;
```

Также есть возможность обновить несколько полей в одной строке. Например, у товара одновременно изменилась стоимость и количество. Вместо того, чтобы писать два отдельных запроса, мы укажем сразу оба изменяемых поля и их значения:

```
UPDATE products
SET price = 78500, quantity = 3
WHERE id = 2;
```

Также можно обновить сразу несколько строк в одной таблице, или даже все. Например, мы хотим выровнять количество всех товаров во всех магазинах, и установить его равным трем. Для этого нам нужно просто опустить блок WHERE, и тогда оператор UPDATE применит все, что указано в блоке SET сразу ко всем строкам в таблице:

```
UPDATE products
SET quantity = 3;
```

Вместо конкретного значения можно использовать выражение, на основании которого будет вычисляться значение. Например, в магазинах проходит

акция, и нужно снизить цены всех ноутбуков на 10%. Рассчитывать значение вручную для каждого товара неудобно, поэтому мы используем выражение. Укажем, что цену нужно умножить на 0.9, то есть сделать ее равной 90% от начальной:

```
UPDATE products
SET price = (price * 0.9);
```

В условии WHERE можно использовать данные, полученные из других таблиц. Расширим предыдущий пример с выражением. В этот раз акция проходит только в магазине с названием «Магазин 2». Для этого нужно в условии WHERE указать идентификатор магазина, для которого нужно обновить цены. Но вместо того, чтобы искать в таблице идентификатор и потом подставлять его в SQL-запрос, мы можем указать название, а идентификатор подставится сам.

Используем вложенный оператор SELECT, который сначала вернет идентификатор нужного нам магазина, а затем подставит его в блок WHERE:

```
UPDATE products
SET price = (price * 0.9)
WHERE store_id = (
  SELECT id
  FROM stores
  WHERE name = 'Магазин 2'
);
```

Команда UPDATE может обновить значения сразу в нескольких таблицах за один раз. Допустим, мы хотим обновить адрес одного из магазинов, и тут же обновить количество товара в нем:

```
UPDATE stores, products
SET stores.address = 'Пятницкая 23', products.quantity = 3
WHERE stores.id = 4
and products.store_id = 4;
```

Иногда значение в блоке SET может быть не явным, а зависеть от какого-либо условия. Например, мы хотим уменьшить цены на все ноутбуки на 100р. Мы уже знаем, как это можно сделать с помощью условия WHERE. А теперь покажем, как то же самое можно сделать с помощью условного оператора IF:

```
UPDATE products
SET price = IF(category_id=1, price-100, price);
```

Усложним пример. Теперь нам нужно уменьшить цены на ноутбуки на 100р, на планшеты поднять на 100р, а телефоны — уменьшить на 5%. Для этого лучше подойдет другой условный оператор — CASE. В нем мы можем перечислить сразу несколько условий:

```
UPDATE products SET price = CASE
  WHEN category_id = 1 THEN price-100
  WHEN category_id = 2 THEN price+100
  WHEN category_id = 3 THEN price*0.95
  END;
```

Простой синтаксис оператора **DELETE** в MySQL:

```
DELETE FROM table
[WHERE conditions];
```

Полный синтаксис оператора DELETE в MySQL:

```
DELETE [ LOW_PRIORITY ] [ QUICK ] [ IGNORE ] FROM table
[WHERE conditions]
[ORDER BY expression [ ASC | DESC ]]
[LIMIT number_rows];
```

Рассмотрим пример запроса MySQL DELETE, в котором у нас есть только одно условие в операторе DELETE

```
DELETE FROM products
WHERE name = 'Телефон 1';
```

Рассмотрим MySQL пример DELETE, где у нас есть только два условия в инструкции DELETE:

```
DELETE FROM products
WHERE quantity = 3
AND price < 20000;
```

Также можно выполнять более сложные удаления. Вы можете удалить записи в одной таблице на основе значений в другой таблице. Поскольку вы не можете перечислить более чем одну таблицу в MySQL операторе FROM при выполнении удаления, вы можете использовать MySQL оператор EXISTS.

```
DELETE FROM products
WHERE EXISTS
 ( SELECT *
   FROM stores
   WHERE stores.id = products.id_store
   AND stores.city = ‘Москва’ );
```

Рассмотрим пример **MySQL условия IN**, используя символьные значения.

Ниже приведен пример MySQL оператора SELECT, который использует условие IN для сравнения значений символов:

```
SELECT *
  FROM contacts
 WHERE last_name IN ('Bernard', 'Boy', 'Tomas');
```

Этот пример MySQL условия IN возвращает все строки из таблицы contacts, где last_name - это Bernard, Boy или Tomas. Поскольку в SELECT используется *, то все поля из таблицы contacts будут отображаться в результирующем наборе.

Вышеприведенный пример IN эквивалентен следующему оператору SELECT:

```
SELECT *
  FROM contacts
 WHERE last_name = 'Bernard'
    OR last_name = 'Boy'
    OR last_name = 'Tomas';
```

Как вы можете видеть, использование MySQL условия IN облегчает чтение и повышает эффективности.

Рассмотрим пример MySQL условия IN, используя числовые значения.

```
SELECT *
  FROM suppliers
 WHERE supplier_id IN (20, 23, 31, 40);
```

Этот пример использования MySQL условия IN будет возвращать всех suppliers, где supplier_id равен 20, 23, 31 или 40.

Рассмотрим пример условия IN, используя оператор NOT.

```
SELECT *
  FROM contacts
 WHERE last_name NOT IN ('Bernard', 'Boy', 'Tomas');
```

Этот пример MySQL условия IN возвращает все строки из таблицы contacts, где last_name не является Bernard, Boy или Tomas. Иногда более эффективно перечислять значения, которые вы не хотите видеть в результате, в отличие от желаемых значений.

Рассмотрим некоторые примеры **MySQL условий BETWEEN**, используя числовые значения. Следующий пример использует условие BETWEEN для извлечения значений в числовом диапазоне.

```
SELECT *
  FROM contacts
 WHERE contact_id BETWEEN 50 AND 100;
```

Этот MySQL пример BETWEEN возвращает все строки из таблицы contacts, где contact_id находится между 50 и 100 (включительно). Это эквивалентно следующему оператору SELECT:

```
SELECT *
  FROM contacts
 WHERE contact_id >= 50
   AND contact_id <= 100;
```

Рассмотрим, как вы будете использовать MySQL условие BETWEEN с датами. При использовании условия BETWEEN в MySQL с датами обязательно используйте функцию CAST для явного преобразования значений в даты.

В следующем примере используется условие BETWEEN для извлечения значений в диапазоне дат.

```
SELECT *
  FROM order_details
   WHERE order_date BETWEEN CAST('2017-12-01' AS DATE) AND
CAST('2017-12-31' AS DATE);
```

Этот пример MySQL условия BETWEEN возвращает все записи из таблицы order_details, где order_date находится между 1 декабря 2017 года и 31 декабря 2017 года (включительно). Это будет эквивалентно следующему оператору SELECT:

```
SELECT *
  FROM order_details
 WHERE order_date >= CAST('2017-12-01' AS DATE)
   AND order_date <= CAST('2017-12-31' AS DATE);
```

Рассмотрим как % работает в **MySQL условии LIKE**. Мы хотим найти всех customers, last_name которых начинается с «Ber».

```
SELECT customer_name
  FROM customers
 WHERE last_name LIKE 'Ber%';
```

Вы также можете использовать % несколько раз в одной строке. Например:

```
SELECT customer_name
  FROM customers
```

```
WHERE last_name LIKE '%ns%';
```

В этом примере MySQL условия LIKE мы ищем всех customers, у которых last_name содержит символы 'ns'.

Рассмотрим, как знак _ (подстановочный символ подчеркивания) работает в MySQL условии LIKE. Помните, что подстановочный символ _ означает только один символ. Например:

```
SELECT supplier_name
  FROM suppliers
 WHERE supplier_name LIKE 'Ber_ard';
```

Этот пример MySQL условия LIKE возвращает всех suppliers, supplier_name которых составляет 7 символов, причем первые три символа - «Ber», а последние три символа - «ard». Например, он может вернуть всех, supplier_name которых - 'Bernard', 'Berzard', 'Bermard', 'Bersard' и т.д.

И еще один пример:

```
SELECT *
  FROM suppliers
 WHERE account_number LIKE '12345_';
```

Таблица 7 – Варианты заданий на домашнюю контрольную работу №1 по учебной дисциплине «Базы данных и системы управления базами данных»

Предпоследняя цифра шифра	Последняя цифра шифра									
	0	1	2	3	4	5	6	7	8	9
0	26, 59, 90, 106,142	1, 47, 69, 100,139	30, 53, 80, 102,143	23, 40, 84, 94,131	10, 60, 75, 99,127	20, 44, 72, 95,135	1, 45, 71, 107148	26, 48, 76, 110141	12, 34, 86, 111,	2, 38, 81, 112,149
1	24, 49, 90, 94,141	18, 44, 72, 96,140	17, 54, 63, 120,148	10, 59, 87, 102,132	25, 56, 62, 115,124	19, 36, 75, 114,129	9, 41, 83, 92,139	3, 57, 70, 99,128	4, 33, 88,110, 147	15, 60, 61, 116,142
2	27, 46, 81, 106,122	13, 55, 67, 98,126	22, 38, 86, 119,135	2, 47, 85, 95,121	11, 31, 69, 100,125	16, 52, 82, 113,130	28, 40, 68, 97,149	12, 45, 79, 108,146	6, 53, 74, 112,143	29, 51, 76, 103,136
3	24, 32, 74, 106,125	21, 56, 81, 115,137	25, 52, 62, 117,124	15, 37, 76, 119,139	19 36, 69, 113135	26, 45, 72, 120,121	9, 44, 63, 98,132	14, 49, 80, 101,149	6, 33, 77, 91,142	12 ,51, 82, 95,143
4	3, 54, 89, 103,133	16, 31, 68, 105,122	5, 34, 65, 118,123	4, 58, 86, 100,134	22, 38, 67, 93,146	28, 42, 79, 114,145	11, 35, 84, 94,148 ,	23, 41, 73, 108,141 ,	27, 59, 90, 99, 150	1, 47, 87, 107,126
5	10, 45, 74, 99,140,	14, 47, 84, 119,147	20, 57, 87, 113,122	17, 32, 88, 97,149	11, 38, 79, 109,121	24, 52 86,100,126	1, 48, 89, 93,124	7, 55, 70, 98,129	9, 34, 73, 116,132	27, 44, 77, 107,131
6	18, 37, 66, 101,135	29, 54, 82, 110,148	25, 56, 69, 117,139	19, 43, 80 115,150	21, 40, 78, 92,134	30, 49, 75, 94,127	6, 46, 81, 120,128	3, 39, 83, 103,125	2, 51, 68, 96,133,	15, 59, 71, 112,136

Продолжение таблицы 7

Предпоследняя цифра шифра	Последняя цифра шифра									
	0	1	2	3	4	5	6	7	8	9
7	26, 60, 63, 108,146	22, 42, 61, 104,145	5, 41, 90, 118,138	23, 53, 65, 106,123	4,40, 74, 116,131	28, 33, 76, 95,130	4, 50, 72, 102,137	16, 35, 67, 105,144	8, 58, 64, 114,141	12, 31, 85, 111,143
8	24, 32, 74, 106,125	21, 56, 81, 115,137	25, 52, 62, 117,124	15, 37, 76, 119,139	19 36, 69, 113,135	26, 45, 72, 120,121	9, 44, 63, 98,132	14, 49, 80, 101,149	6, 33, 77, 91,142	12 ,51, 82, 95,143
9	7, 57, 83, 112,128	2, 43, 61, 97,131	29, 39, 78, 104,136	10, 40, 66, 92,147	13, 50, 88, 110,129	20, 55, 71, 102,127	18, 48, 75, 96,144	8, 53, 85, 116,138	30, 60, 64, 109,140	17, 46, 70, 111,130