

*Приложение 2.32.1 к ОПОП по специальности
09.02.07 Информационные системы и
программирование*

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет им. Г.И. Носова»

Многопрофильный колледж

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ
ДЛЯ ЛАБОРАТОРНЫХ И ПРАКТИЧЕСКИХ ЗАНЯТИЙ
УЧЕБНОЙ ДИСЦИПЛИНЫ
ОП.08**

«Основы проектирования баз данных»

**для обучающихся специальности
09.02.13 Интеграция решений с применением технологий искусственного интеллекта**

Квалификация: Специалист по работе с искусственным интеллектом

Магнитогорск, 2025

ОДОБРЕНО

Предметно-цикловой комиссией

«Информатики и вычислительной техники» Председатель

Т.Б. Ремез

Протокол № 5 от 22.01.2025

Методической комиссией МпК

Протокол №4 от №3 от «19» февраля
2025

Разработчик:

преподаватель ФГБОУ ВО «МГТУ им. Г.И. Носова» Многопрофильный колледж

Т.Б. Осолодкова

Методические указания по выполнению практических и лабораторных работ разработаны на основе рабочей программы учебной дисциплины «Основы проектирования баз данных».

Содержание практических и лабораторных работ ориентировано на подготовку обучающихся к освоению профессиональных модулей программы подготовки специалистов среднего звена по специальности 09.02.07 Информационные системы и программирование и овладению профессиональными компетенциями.

СОДЕРЖАНИЕ

1 ВВЕДЕНИЕ	4
2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ	6
Практическая работа №1	6
Практическое занятие №2	8
Практическая работа №3.	11
Практическая работа №4	13
Практическая работа №5	15
Практическая работа №6	20
Практическая работа №7	22
Практическая работа №8	23
Практическая работа №9	25
Практическая работа №10	26
Практическая работа №11	28
Практическая работа №12	30

1 ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки обучающихся составляют практические и лабораторные занятия.

Состав и содержание практических и лабораторных занятий направлены на реализацию Федерального государственного образовательного стандарта среднего профессионального образования.

Ведущей дидактической целью практических занятий является формирование учебных практических умений (использовать современные case-средства для проектирования баз данных; создавать объекты баз данных в современных СУБД; применять стандартные методы для защиты объектов базы данных), необходимых в последующей учебной деятельности по профессиональному модулю ПМ.11 Разработка, администрирование и защита баз данных.

Ведущей дидактической целью лабораторных занятий является экспериментальное подтверждение и проверка существенных теоретических положений (законов, зависимостей).

В соответствии с рабочей программой учебной дисциплины «Основы проектирования баз данных» предусмотрено проведение практических и лабораторных занятий. В рамках практического занятия обучающиеся могут выполнять одну или несколько практических работ.

В результате их выполнения, обучающийся должен:

уметь:

- проектировать реляционную базу данных
- использовать язык запросов для программного извлечения сведений из баз данных,
- анализировать задачу и/или проблему и выделять её составные части;
- определять этапы решения задачи;
- выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы
- применять средства информационных технологий для решения профессиональных задач;
- использовать современное программное обеспечение;
- понимать тексты на базовые профессиональные темы;
- читать, понимать и находить необходимые технические данные и инструкции в руководствах в любом доступном формате.

Содержание практических и лабораторных занятий ориентировано на подготовку обучающихся к освоению профессионального модуля программы подготовки специалистов среднего звена по специальности и овладению **профессиональными компетенциями:**

ПК 11.1 Осуществлять сбор, обработку и анализ информации для проектирования баз данных.

ПК11.2 Проектировать базу данных на основе анализа предметной области.

ПК11.3 Разрабатывать объекты базы данных в соответствии с результатами анализа предметной области.

ПК 11.4 Реализовывать базу данных в конкретной системе управления базами данных.

ПК 11.5 Администрировать базы данных.

ПК11.6 Защищать информацию в базе данных с использованием технологии защиты информации.

А также формированию **общих компетенций:**

ОК1 Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.

ОК 2 Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности.

ОК 4 Эффективно взаимодействовать и работать в коллективе и команде.

ОК 5 Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста.

ОК 7 Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата, принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях.

ОК 9 Пользоваться профессиональной документацией на государственном и иностранном языках.

Выполнение обучающихся практических и лабораторных работ по учебной дисциплине «Основы проектирования баз данных» направлено на:

- *обобщение, систематизацию, углубление, закрепление, развитие и детализацию полученных теоретических знаний по конкретным темам учебной дисциплины;*

- *формирование умений применять полученные знания на практике, реализацию единства интеллектуальной и практической деятельности;*

- *выработку при решении поставленных задач профессионально значимых качеств, таких как самостоятельность, ответственность, точность, творческая инициатива.*

Практические и лабораторные занятия проводятся в рамках соответствующей темы, после освоения дидактических единиц, которые обеспечивают наличие знаний, необходимых для ее выполнения.

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Тема 1.1 Введение в базы данных

Практическая работа №1

Анализ структуры базы данных на примере реальной системы.

Цель: получение практических навыков проектирования базы данных

Выполнив работу, Вы будете: уметь:

- У1. Проектировать реляционную базу данных
- У01.2 анализировать задачу и/или проблему и выделять её составные части;
- У01.3 определять этапы решения задачи;
- У 01.4 Выявлять и эффективно искать информацию, необходимую для решения задачи

и/или проблемы

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение.

Задание:

В соответствии со своим вариантом проанализируйте предметную область решаемой задачи и разработайте логическую структуру соответствующей базы данных.

Порядок выполнения работы:

1. Выполнить анализ предметной области. Выполнить анализ данных.
2. Определить набор атрибутов для данной предметной области. 4. Определить набор таблиц.
3. При проектировании таблиц рекомендуется руководствоваться следующими основными принципами:
 - каждая таблица должна содержать данные только на одну тему;
 - данные не должны дублироваться.
4. Создать словарь имен.
5. Определить состав и типы полей.
6. Создать связи между таблицами.

Ход работы.

Краткие теоретические сведения:

Проектирование базы данных заключается в ее многоступенчатом описании с различной степенью детализации и формализации, в ходе которого производится уточнение и оптимизация структуры базы данных. Проектирование начинается с описания предметной области и задач информационной системы, идет к более абстрактному уровню логического описания данных и далее – к схеме физической (внутренней) модели базы данных. Трем основным уровням моделирования системы – **концептуальному, логическому и физическому** соответствуют три последовательных этапа детализации описания объектов базы данных и их взаимосвязей.

На **концептуальном уровне** проектирования производится смысловое описание информации предметной области, определяются ее границы, производится абстрагирование от несущественных деталей. В результате определяются моделируемые объекты, и их свойства, и связи. Выполняется структуризация знаний о предметной области, стандартизируется терминология. Затем строится концептуальная модель, описываемая на естественном языке. Для описания свойств и связей объектов применяют различные диаграммы.

На следующем шаге принимается решение о том, в какой конкретно СУБД будет реализована база данных. **Выбор СУБД** является сложной задачей и должен основываться на потребностях с точки зрения информационной системы и пользователей. Определяющими здесь являются вид программного продукта и категория пользователей (или профессиональные программисты, или конечные пользователи, или то и другое).

Другими показателями, влияющими на выбор СУБД, являются:

- Удобство и простота использования;

- Качество средств разработки, защиты и контроля базы данных;
- Уровень коммуникационных средств (в случае применения ее в сетях);
- Фирма-разработчик;
- Стоимость.

Каждая конкретная СУБД работает с определенной моделью данных. Под моделью данных понимается способ их взаимосвязи: в виде иерархического дерева, сложной сетевой структуры или связанных таблиц. В настоящее время большинство СУБД использует табличную модель данных, называемую *реляционной*.

На **логическом уровне** производится отображение данных концептуальной модели в логическую модель в рамках той структуры данных, которая поддерживается выбранной СУБД. Логическая модель не зависит от конкретной СУБД и может быть реализована на любой СУБД реляционного типа.

На **физическом уровне** производится выбор рациональной структуры хранения данных и методов доступа к ним, которые обеспечивает выбранная СУБД. На этом уровне решаются вопросы эффективного выполнения запросов к БД, для чего строятся дополнительные структуры, например, индексы. В физической модели содержится информация обо всех объектах базы данных (таблицах, индексах, процедурах и др.) и используемых типах данных. Физическая модель *зависит* от конкретной СУБД. Одной и той же логической модели может соответствовать несколько разных физических моделей. Физическое проектирование является начальным этапом реализации базы данных.

Варианты заданий:

1. *БД «Зарплата»*. Таблица «Должности»: код должности; должность. Таблица «Тарифы»: код должности; разряд; ставка, р./ч. Таблица «Работники»: отдел; код должности; разряд; табельный номер; Ф.И.О. Таблица «Табель»: год; месяц; табельный номер; количество отработанных часов; дата начисления зарплаты
2. *БД «Оптовая база»*. Таблица «Товары»: код товара; товар; единица измерения; цена. Таблица «Склад»: дата поступления, код товара; количество. Таблица «Заявки»: номер заявки; организация; код товара; требуемое количество товара. Таблица «Отпуск товаров»: номер заявки; код товара; дата отпуска товара; количество отпущенного товара
3. *БД «Библиотека»*. Таблица «Книги»: жанр; шифр книги; автор; название; год издания; количество экземпляров. Таблица «Читатели»: номер читательского билета; Ф.И.О.; адрес. Таблица «Выдачи»: дата выдачи; номер читательского билета; шифр книги; количество экземпляров; срок возврата; фактическая дата возврата
4. *БД «ГИБДД»*. Таблица «Автомобили»: модель автомобиля; номер двигателя; номер кузова; серия и номер технического паспорта; государственный номер автомобиля. Таблица «Владельцы»: государственный номер автомобиля; Ф.И.О.; адрес, серия и номер водительского удостоверения. Таблица «Виды нарушений»: код нарушения; вид нарушения. Таблица «Нарушители»: дата нарушения; код нарушения; государственный номер автомобиля; размер штрафа, р.
5. *БД «Соревнования»*. Таблица «Команда»: спортивная организация; шифр команды; название команды. Таблица «Участники»: шифр команды; номер участника; Ф.И.О. Таблица «Старт»: стартовый номер; номер участника; время старта; отметка о невыходе на старт. Таблица «Финиш»: порядковый номер финиширования; стартовый номер; время финиша; отметка о сходе с дистанции
6. *БД «Перевозки»*. Таблица «Транспорт»: модель автомобиля; государственный номер автомобиля; удельный расход топлива, л/100 км. Таблица «Заявки»: номер заявки;
7. дата; пункт отправления; пункт назначения; груз; единица измерения; количество груза. Таблица «Доставка»: номер заявки; государственный номер автомобиля; дата отправления; дата возвращения; пройденное расстояние; расход топлива
8. *БД «Сессия»*. Таблица «Кафедры и дисциплины»: номер кафедры; кафедра; шифр дисциплины; наименование дисциплины; вид аттестации (зачет или экзамен). Таблица

«Преподаватели»: номер кафедры; табельный номер преподавателя; Ф.И.О. Таблица «Студенты»: шифр студента; Ф.И.О. Таблица «Оценки»: дата; шифр дисциплины; табельный номер преподавателя; шифр студента; оценка

9. БД «Телефонная компания». Таблица «Тарифы»: код тарифа; вид тарифа; цена, р./мин. Таблица «Виды льгот»: код льготы; вид льготы; размер, %. Таблица «Абоненты»: лицевой счет; телефон; Ф.И.О.; адрес; код льготы. Таблица «Платежи»: лицевой счет; код тарифа; дата оплаты; сумма платежа; дата отключения за неуплату

Форма представления результата: Оформленная схема базы данных в тетради. **Критерии оценки:**

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания. Оценка «неудовлетворительно» ставится, если задание не выполнено.

Практическое занятие №2.

Построение ER-диаграммы для простой предметной области.

Цель: получение практических навыков по нормализации базы данных

Выполнив работу, Вы будете: уметь:

У1. Проектировать реляционную базу данных

Уо 04.01 организовывать работу коллектива и команды,

Уо 04.03 эффективно работать в команде,

Уо 09.01 - понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать тексты на базовые профессиональные темы,

Уо 09.02 - участвовать в диалогах на знакомые общие и профессиональные темы.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение.

Задание:

В соответствии со своим вариантом задания, выданным на предыдущем занятии привести базу данных к 3НФ. Реализовать схему базы данных в среде MySQL Workbench.

Порядок выполнения работы:

Используя метод нормальных форм спроектировать базу данных. Процесс проектирования должен быть представлен в форме отчета, показан процесс формирования таблиц под выделенные сущности и их последовательная нормализация методом нормальных форм.

Ход работы.

Краткие теоретические сведения:

Нормализация баз данных

Нормальные формы – это рекомендации по проектированию баз данных. Рекомендуется нормализовать базу данных в некоторой степени потому, что этот процесс имеет ряд существенных преимуществ с точки зрения эффективности и удобства обращения с базой данных.

В нормализованной структуре базы данных можно производить сложные выборки данных относительно простыми SQL-запросами.

Целостность данных. Нормализованная база данных позволяет надежно хранить данные.

Нормализация предотвращает появление избыточности хранимых данных. Данные всегда

хранятся только в одном месте, что делает легким процесс вставки, обновления и удаления данных. Есть исключение из этого правила. Ключи, сами по себе, хранятся в нескольких местах потому, что они копируются как внешние ключи в другие таблицы.

Масштабируемость – это возможность системы справляться с будущим ростом. Для базы данных это значит, что она должна быть способна работать быстро, когда число пользователей и объемы данных возрастают. Масштабируемость – это очень важная характеристика любой модели базы данных и для РСУБД.

Нормализация баз данных заключается в приведении структуры хранения данных к нормальным формам (NF). Всего таких форм существует 8, но часто достаточным является соблюдение первых трех. Рассмотрим их более подробно на примере учебной базы данных. Примеры будут строиться по принципу «что было бы, если было иначе, чем сейчас».

Первая нормальная форма

Основным правилом первой формы является необходимость неделимости значения в каждом поле (столбце) строки – **атомарность** значений.

и

Сотрудники_Линии		
Имя столбца	Сжатый тип	Допускает значения NULL
Сотрудник	int	Нет
Линия	int	Нет
Описание	varchar(255)	Да
Приоритет	int	Нет

Рассмотрим таблицы сотрудников телефонных линий.

Чтобы избавиться от связывающей таблицы «Сотрудники_Линии», мы могли бы записать идентификаторы сотрудников для каждой линии в виде перечня в дополнительном столбце:

	id	Описание	Приоритет	Сотрудники
1	1	Поддержка держателей пластиковых карт	2	20, 21, 8
2	2	Поддержка потребительского и наличного кредитова...	2	16, 5, 22, 15
3	3	Поддержка автокредитования	2	21, 10
4	4	Поддержка ипотечного кредитования	2	16, 14, 15, 4, 8

Но подобная структура не является надежной. Представьте, что Вам необходимо поменять некоторым сотрудникам подключенные линии. Потребуется осуществить разбор составного поля, чтобы определить наличие id сотрудника в каждой записи линий, затем скорректировать перечень. Получается слишком сложный и долгий процесс для такой простой операции.

Организации структуры таблиц с применением дополнительной связывающей избавляет от подобных проблем.

Помимо атомарности к первой нормальной форме относятся следующие правила:

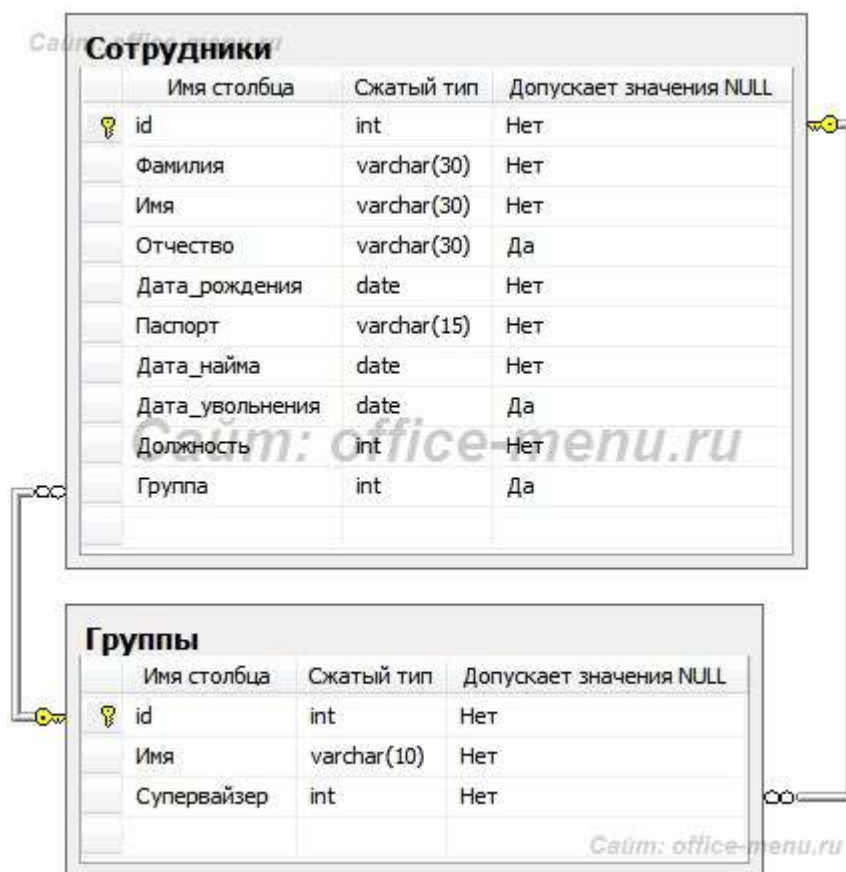
Строки таблиц не должны зависеть друг от друга, т.е. первая запись не должна влиять на вторую и наоборот, вторая на третью и т.д. Размещение записей в таблице не имеет никакого значения.

Аналогичная ситуация со столбцами записей. Их порядок не должен влиять на понимание информации.

Каждая строка должна быть уникальна, поэтому для нее определяется **первичный ключ**, состоящий из одного либо нескольких полей (**составной ключ**). Первичный ключ не может повторяться в пределах таблицы и служит идентификатором записи.

Вторая нормальная форма

Условием этой формы является отсутствие зависимости неключевых полей от части



составного ключа.

Так как составной ключ в учебной базе наблюдается только в таблице «Сотрудники_Линии», то рассмотрим пример на ней.

На представленной диаграмме столбцы описания и приоритета зависят от столбца «Линия», входящего в составной ключ. Это значит, что для каждой линии, подключенной разным сотрудникам, потребуется повторно указывать описание и приоритетность. Подобная структура приводит к **избыточности данных**.

Также велика вероятность возникновения противоречивой информации. Изменяя приоритет или описание для линии, можно по ошибке оставить некоторые строки не обработанными. В таком

случае, для одного и того же идентификатора линии значения зависимых полей будут различными.

Если соблюдены правила первой нормальной формы, то создание таблицы «Линии» и перенос в нее зависимых столбцов удовлетворяет второй нормальной форме.

Третья нормальная форма

3NF схожа по логике с 2NF, но с некоторым отличием. Если 2 форма ликвидирует зависимости неключевых полей от части ключа, то третья нормальная форма исключает зависимость неключевых полей от других неключевых полей.

На приведенном примере таблицы сотрудников видно, что столбец «Супервайзер» имеет зависимость от столбца «Группа», а это значит, что при изменении значения поля группы, потребуется изменить значение поля супервайзера.

Все риски, которые были рассмотрены для 2NF, так же относятся к 3NF и устраняются переносом зависимых полей в отдельную таблицу.

Денормализация базы данных

Теория нормальных форм не всегда применима на практике. Например, неатомарные значения не всегда являются «злом», а иногда наоборот. Связано это с необходимостью дополнительного объединения (следовательно, затрат производительности системы) при выполнении запросов, особенно когда производится обработка большого массива информации.

Форма представления результата:

представление отчета на образовательном портале (в соответствующем курсе). **Критерии оценки:**

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 1.2. Концептуальное проектирование баз данных

Практическая работа №3.

Определение сущностей и атрибутов для заданной предметной области.

Цель: получение практических навыков по освоению операций настройки схемы базы данных.

Выполнив работу, Вы будете: уметь:

У1. Проектировать реляционную базу данных

Уо 04.01 организовывать работу коллектива и команды,

Уо 04.03 эффективно работать в команде,

Уо 09.01 - понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать тексты на базовые профессиональные темы,

Уо 09.02 - участвовать в диалогах на знакомые общие и профессиональные темы.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MySQL Workbench.

Задание:

Создать схему базы данных по заданному варианту.

Порядок выполнения работы:

1. Разработать логическую модель схемы по своему варианту. 2. Посмотреть и проанализировать физическую модель.
2. Сгенерировать скрипт создания таблиц.
3. Используя сгенерированный скрипт создать базу данных в выбранной СУБД.
4. Настроить схему базы данных.

Краткие теоретические сведения:

Этапы проектирования базы данных:

1. инфологическое проектирование;
2. определение требований к операционной обстановке, в которой будет функционировать информационная система;
3. выбор системы управления базой данных (СУБД) и других инструментальных программных средств;
4. логическое проектирование БД;
5. физическое проектирование БД.

Инфологическое проектирование.

Инфологическое проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создается без ориентации на какую-либо конкретную СУБД и модель данных. Конкретный вид и содержание концептуальной модели базы данных определяется выбранными для этого формальным аппаратом. Обычно используют графические нотации, подобные ER-диаграммам. Чаще всего инфологическая (концептуальная) модель базы данных включает в себя: □ описание информационных объектов или понятий предметной области и связей между ними; □ описание ограничений целостности, то есть требований к допустимым значениям данных и к связям между ними. Пример инфологического проектирования показан на рисунке 1.

Entity-Relationship Diagrams

Имеется целый ряд методик создания информационно-логических моделей. Одна из наиболее популярных в настоящее время методик при разработке моделей использует ERD (Entity-Relationship Diagrams). В русскоязычной литературе эти диаграммы называют «объект – отношение» либо «сущность – связь». Модель ERD была предложена Питером Пин Шен Ченом в 1976 г. К настоящему времени разработано несколько ее разновидностей, но все они базируются на графических диаграммах, предложенных Ченом. Диаграммы конструируются из небольшого числа компонентов. Благодаря наглядности представления они широко используются в CASE-средствах (Computer Aided Software Engineering). Рассмотрим используемую терминологию и обозначения.

Сущность (Entity) – реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области, информация о котором подлежит хранению.

Каждая сущность должна обладать уникальным идентификатором. Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа (сущности).

Каждая сущность должна обладать некоторыми свойствами:

- иметь уникальное имя; причем к этому имени должна всегда применяться одна и та же интерпретация (определение сущности). И наоборот: одна и та же интерпретация не может применяться к различным именам, если только они не являются псевдонимами;
- обладать одним или несколькими атрибутами, которые либо принадлежат сущности, либо наследуются ею через связь;
- обладать одним или несколькими атрибутами, которые однозначно идентифицируют каждый экземпляр сущности.

Связь (Relationship) – поименованная ассоциация между двумя сущностями, значимая для рассматриваемой предметной области. Одна из участвующих в связи сущностей – независимая, называется родительской сущностью, другая – зависимая, называется дочерней или сущностью-потомком. Как правило, каждый экземпляр родительской сущности ассоциирован с произвольным (в том числе нулевым) количеством экземпляров дочерней сущности. Каждый экземпляр сущности-потомка ассоциирован в точности с одним экземпляром сущности-родителя. Таким образом, экземпляр сущности-потомка может существовать только при существовании сущности-родителя. Связи дается имя, выражаемое грамматическим оборотом глагола и помещаемое возле линии связи.

При построении ER-модели используются следующие принципы:

- сущности на диаграмме представляются прямоугольниками;
- каждый прямоугольник может иметь различные визуальные атрибуты; □ каждой



- сущности должно быть присвоено уникальное имя;
- имена сущностей необходимо задавать в единственном числе;
- связи на диаграмме представляются линиями, идущими от одной сущности (таблицы) к другой;
- каждой связи присваивается уникальное имя;
- связанные таблицы разделяют на родительские и дочерние;
- родительские таблицы отображаются прямоугольниками с прямыми углами, дочерние – со скругленными.

Логическое проектирование

Логическое проектирование – создание схемы базы данных на основе конкретной модели базы данных, например, реляционной модели данных. Для реляционной модели данных даталогическая модель – набор схем отношений, обычно с указанием первичных ключей, а также

«связей» между отношениями, представляющих собой внешние ключи.

Преобразование концептуальной модели в логическую модель, как правило, осуществляется по формальным правилам. Этот этап может быть в значительной степени автоматизирован. На этапе логического проектирования учитывается специфика конкретной модели данных, но может не учитываться специфика конкретной СУБД.

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания. Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 1.2 Концептуальное проектирование баз данных

Практическая работа №4

Построение сложной ER-диаграммы с учётом нормализации.

Цели:

- получение практических навыков по освоению операций создания таблиц;
- получение практических навыков по освоению операций подстановок.

Выполнив работу, Вы будете: *уметь*:

У2 Использовать язык запросов для программного извлечения сведений из баз данных,

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access, MySQL Workbench.

Задание 1:

Создать базу данных Туризм и таблицы базы данных в СУБД MS Access.

Порядок выполнения работы:

1. Создайте в своей рабочей папке папку с именем *База данных*.
2. Запустите **MS Access**. Создайте в созданной папке новую базу данных с именем *Туризм*.

Ход работы.

Краткие теоретические сведения:

Связи *автоматические* устанавливаются **Мастером подстановок**. Посмотреть, установить, отредактировать связи можно командой **Работа с базой данных – Схема данных**.

Если связи устанавливаются первично, то откроется окно **Таблицы**, а если повторно, то окно **Связи**. Двойной щелчок на нужной таблице позволит перенести их в окно **Связи**.

Для установки связи между таблицами *вручную* нужно перетянуть связываемое поле из

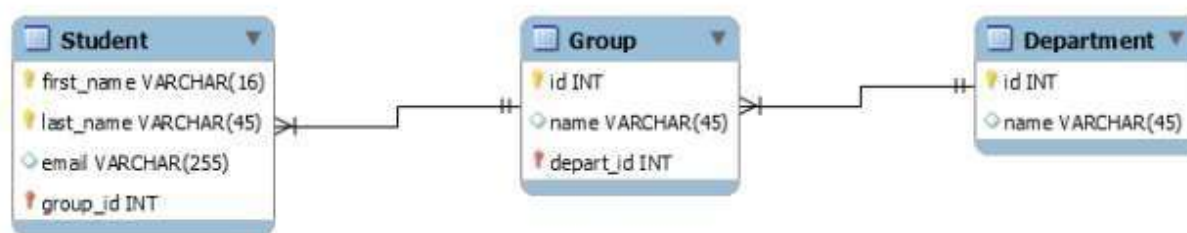


Рисунок 2 – Логическая модель

главной таблицы и наложить его на соответствующее поле подчиненной таблицы.

Удаление и изменение связей производится с помощью контекстного меню на линии связи, а также клавишей DEL.

В окне **Схема данных** двойной щелчок по линии связи позволит открыть окно **Связи**. В нем можно установить флажок у опции **Целостность данных**, линия связи станет гораздо темнее и появятся значки «1» и «∞», означающие отношение «один» или «многие».

Если система определила тип связи (в нижней части диалогового окна) «один-к-одному» или «один-ко-многим», то можно поставить флажок «Поддерживать целостность данных».

Целостность данных – это набор правил, защищающих данные от случайных изменений или удалений с помощью механизма поддержки корректности связей между связанными таблицами.

Если связь определена и система взяла на себя поддержку целостности данных, то при просмотре главной таблицы (отношение «один») слева, рядом с полосой выделения появится колонка со знаками «+». Щелчок на «+» позволит открыть подчиненную таблицу (отношение «много» или «один»).

Создание таблиц с помощью Конструктора

1. Создайте таблицу **Сотрудники** в режиме Конструктора. Наименования и типы полей представлены в приведенной таблице.

Название поля	Тип данных
Код клиента	Числовой
Название клиента	Текст
Контактное лицо	Текст
Признак группы	Логический
Телефон	Текст
Адрес	Текст

Название поля	Тип данных
Код сотрудника	Числовой
ФИО	Текст
Должность	Текст
Дата найма	Дата/Время
Дата рождения	Дата/Время
Домашний телефон	Текст
Адрес	Текст

Размер оклада Числовой

2. Для поля **Домашний телефон** задайте маску, набрав, например, следующий шаблон (999) 999-99-99.

3. Для поля **Оклад** задайте условие, что он больше 5000 р., но не больше 10000. Для этого в свойстве «Условие на значение» установите (>5000) AND (<10000). Предусмотрите выдачу сообщения при ошибке ввода данных.

4. Установите для **Даты рождения** и **Даты найма** маску ввода. Используйте краткий формат даты.

5. Создайте первичный ключ.

6. Перейдите в режим просмотра таблицы.

7. Спрячьте некоторые столбцы. Сделайте их опять видимыми командами **Контекстное меню — Скрыть/Показать столбцы**.

8. Зафиксируйте столбцы, содержащие фамилию и имя. Освободите столбцы. 11. Поменяйте тип шрифта и его начертание (**Формат — Шрифт**).

9. Закройте окно таблицы **Сотрудники**, сохранив изменения.

10. Создайте новые таблицы **Клиенты** и **Страны**. В качестве первичного ключа задайте **Код Клиента** и **КодТура**.

Использование Мастера подстановок

Создайте в режиме **Конструктора** таблицу **Договоры**, которая должна иметь следующие поля:

Поля **Код сотрудника**, **Код клиента**, **Код тура** являются полями подстановки. Для их задания используется Мастер подстановок:

- в Типе данных поля **Код сотрудника** раскрыть список типов и выбрать **Мастер подстановок**;

- указать, что столбец подстановки получает свои значения из таблицы **Сотрудники**; выбрать поля *Код сотрудника* и *Фамилия*;
- установить мышью подходящую ширину столбца;
- согласиться с предлагаемой подписью столбца подстановок *Фамилия*;
- сохранить таблицу с именем **Договоры**.

Аналогично для подстановки *Кода клиента* и *Кода тура* вызывается **Мастер подстановок**. При этом для *Кода клиента* выбираем поля *Код клиента* и *Название клиента* из таблицы **Клиенты**, а для *Кода тура* — поля *Код тура* и *Страна* из таблицы **Страны**.

Просмотрите и проанализируйте уже установленные при работе с **Мастером подстановки** связи в окне **Схема данных**.

В окне **Схема данных** двойным щелчком по линии связи откройте окно Связи и установите *Целостность данных*, *Каскадное обновление данных*, *Каскадное удаление данных*.

Введите в таблицы 10 разнообразных записей. В таблице **Сотрудники** *осуществите* ввод заведомо некорректных данных для проверки работоспособности условия на значение.

Просмотрите главную таблицу каждой связи (с помощью «+») и вызовите *подчиненную* таблицу для каждой записи

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.1 Основные понятия языка SQL.

Практическая работа №5

Преобразование ER-диаграммы в таблицы реляционной базы данных.

Цель: получение практических навыков по созданию ключевых полей и индексов, установлению связей между таблицами

Выполнив работу, Вы будете: уметь:

У2 Использовать язык запросов для программного извлечения сведений из баз данных,

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access.

Задание 1:

По своему варианту базы данных в СУБД Access для каждой таблицы создайте первичные и внешние ключи, установите связи и укажите индексируемые поля.

Порядок выполнения работы:

1. Используя рассмотренный ниже пример, установить для своей базы данных первичные ключи.

2. Установить необходимы индексируемые поля.

3. Установить связи между таблицами на схеме данных.

Краткие теоретические сведения:

Индексы обеспечивают поиск и сортировку записей в Access. В индексе хранится местоположение записей на основе одного или нескольких полей, которые были выбраны для индексирования. После того как Access получает сведения о позиции в индексе, он может извлечь

данные путем перемещения непосредственно к нужной позиции. Благодаря этому использование индекса гораздо эффективнее просмотра всех записей для поиска необходимых данных.

Выбор полей для индексирования

Вы можете создавать индексы, основанные на одном или нескольких полях. В основном требуется индексировать поля, в которых часто осуществляется поиск, сортируемые поля и поля, объединенные с полями в других таблицах, что часто используется в запросах по нескольким таблицам. Индексы ускоряют поиск и выполнение запросов, однако они могут привести к снижению производительности при добавлении или обновлении данных. Каждый раз, когда вы добавляете или изменяете запись в таблице, содержащей один или несколько индексов, Access приходится обновлять индексы. Добавление записей с помощью запроса на добавление или с помощью импортирования записей также, скорее всего, будет происходить медленнее, если таблица-получатель содержит индексы.

Примечание: Первичный ключ таблицы индексируется автоматически.

Индексировать поля с типом данных "Объект OLE", "Вычисляемый" или "Вложение" невозможно. Индексировать другие поля следует в тех случаях, когда выполняются все указанные ниже условия.

Тип данных поля: короткий текст (текст в Access 2010), длинный текст (МЕМО в Access 2010), число, Дата и время, счетчик, денежная единица, да/нет или гиперссылка.

Предполагается поиск значений в поле.

Предполагается сортировка значений в поле.

Предполагается сохранение большого числа различных значений в поле. Если поле содержит много одинаковых значений, то применение индекса может не дать значительного ускорения выполнения запросов.

Составные индексы

Если предполагается, что необходимо будет часто выполнять поиск или сортировку по нескольким полям, вы можете создать индекс для этого сочетания полей. Например, если в одном запросе часто задаются условия для полей "Поставщик" и "Наименование_продукта", имеет смысл создать для этих полей составной индекс.

При сортировке таблицы по составному индексу Access сначала выполняет сортировку по первому полю, заданному для индекса. Последовательность полей определяется при создании составного индекса. Если в первом поле содержатся записи с повторяющимися значениями, затем выполняется сортировка по второму полю, заданному для индекса, и т. д.

В составной индекс можно включить до 10 полей. Создание индекса

Перед созданием индекса необходимо решить, следует ли создать индекс для одного поля или составной индекс. Индекс для одного поля создается с помощью установки свойства **Индексированное поле**.

При создании уникального индекса невозможно ввести новое значение в определенном поле, если такое значение уже существует в том же поле другой записи. Access автоматически создает уникальный индекс для первичных ключей, однако может потребоваться запретить создание повторяющихся значений и в других полях. Например, вы можете создать уникальный индекс для поля, в котором содержатся серийные номера, чтобы двум продуктам нельзя было присвоить один и тот же серийный номер.

Создание индекса для одного поля

1. В области навигации щелкните правой кнопкой мыши имя таблицы, в которой необходимо создать индекс, и выберите в контекстном меню пункт **Конструктор**.

2. Щелкните пункт **Имя поля** для поля, которое следует индексировать.

3. В разделе **Свойства поля** откройте вкладку **Общие**.

4. В свойстве **Индексированное** выберите значение **Да (допускаются совпадения)**, если следует разрешить повторяющиеся значения, или значение **Да (совпадения не допускаются)**, чтобы создать уникальный индекс.

5. Чтобы сохранить изменения, щелкните элемент **Сохранить** на панели быстрого доступа или нажмите клавиши CTRL+S.

Создание составного индекса

Чтобы создать составной индекс для таблицы, добавьте строку для каждого поля в индексе и укажите имя индекса только в первой строке. Все строки будут обрабатываться как часть одного индекса, пока не будет обнаружена строка с другим именем индекса. Чтобы вставить строку, щелкните правой кнопкой мыши место, куда вы хотите ее вставить, и выберите в контекстном меню команду **Вставить строки**.

1. В области навигации щелкните правой кнопкой мыши имя таблицы, в которой необходимо создать индекс, и выберите в контекстном меню пункт **Конструктор**.
2. На вкладке **Конструктор** в группе **Показать или скрыть** щелкните пункт **Индексы**.

Появится окно "Индексы". Измените размеры этого окна, чтобы отображались пустые строки и свойства индекса.

3. В первой пустой строке столбца **Индекс** введите имя индекса. Для индекса можно использовать либо имя одного из индексируемых полей, либо другое подходящее имя.

4. В столбце **Имя поля** щелкните стрелку, затем щелкните первое поле, которое следует использовать в индексе.

5. Следующую строку столбца **Индекс** оставьте пустой, затем в столбце **Имя поля** укажите второе индексируемое поле. Повторите этот шаг для всех полей, которые необходимо включить в индекс.

6. Чтобы изменить порядок сортировки значений полей, в столбце **Порядок сортировки** окна "Индексы" щелкните пункт **По возрастанию** или **По убыванию**. По умолчанию выполняется сортировка по возрастанию.

7. В разделе **Свойства индекса** окна **Индексы** укажите свойства индекса для строки в столбце **Имя индекса**, содержащем имя индекса.

Чтобы сохранить изменения, нажмите кнопку **Сохранить** на панели быстрого доступа или нажмите клавиши CTRL+S.

9. Закройте окно "Индексы".

Удаление индекса

Если индекс становится ненужным или приводит к значительному снижению производительности, его можно удалить. При этом удаляется только сам индекс, а не поля, на которых он основан.

1. В области навигации щелкните правой кнопкой мыши имя таблицы, для которой необходимо удалить индекс, и выберите в контекстном меню пункт **Конструктор**.

2. На вкладке **Конструктор** в группе **Показать или скрыть** щелкните пункт **Индексы**.

Появится окно "Индексы". Измените размеры этого окна, чтобы отображались пустые строки и свойства индекса.

3. В окне "Индексы" выделите строки, содержащие индекс, который следует удалить, и нажмите клавишу DELETE.

4. Чтобы сохранить изменения, нажмите кнопку **Сохранить** на панели быстрого доступа или нажмите клавиши CTRL+S.

5. Закройте окно **Индексы**.

Просмотр или редактирование индексов

Чтобы оценить влияние индексов на производительность или убедиться, что необходимые

поля проиндексированы, просмотрите индексы в таблице.

1. В области навигации щелкните правой кнопкой мыши имя таблицы, индекс которой вы хотите изменить, и выберите в контекстном меню пункт **Конструктор**.

2. На вкладке **Конструктор** в группе **Показать или скрыть** щелкните пункт **Индексы**.

Появится окно "Индексы". Измените размеры этого окна, чтобы отображались пустые строки и свойства индекса.

3. Просмотрите или измените индексы и свойства индексов в соответствии со своими задачами.

4. Чтобы сохранить изменения, нажмите кнопку **Сохранить** на панели быстрого доступа или нажмите клавиши CTRL+S.

5. Закройте окно **Индексы** Автоматическое создание индексов

В некоторых случаях индексы создаются автоматически. Например, индексы создаются для любых полей, которые определяются пользователем в качестве первичного ключа таблицы.

Для автоматического создания индекса также можно использовать параметр **Автоиндекс при импорте и создании** в диалоговом окне **Параметры Access**. Access автоматически проиндексирует все поля, имена которых начинаются с указанных в поле **Автоиндекс при импорте и создании** знаков или заканчиваются ими, например **ID**, **ключ**, **код** или **число**. Чтобы просмотреть или изменить текущие параметры, сделайте следующее:

1. Выберите **Файл > Параметры**.

2. Щелкните **Конструкторы объектов**, а затем в разделе **Конструктор таблиц** добавьте, измените или удалите значения в поле **Автоиндекс при импорте и создании**. Для разделения значений используйте точку с запятой (;).

Примечание: Если имя поля начинается со значения, указанного в списке, или заканчивается им, поле будет автоматически проиндексировано.

3. Нажмите кнопку **ОК**.

Так как каждый индекс требует дополнительной обработки, производительность при добавлении или обновлении данных снижается. Поэтому рекомендуется изменить значения, указанные в поле **Автоиндекс при импорте и создании** или уменьшить их число, чтобы сократить количество создаваемых индексов.

Пример.

Установка ключевых полей.

Отдельные таблицы, содержащие информацию по определенной теме, необходимо связать в единую структуру базы данных. Для связывания таблиц следует задать **ключевые поля**.

Ключ состоит из одного или нескольких полей, значения которых однозначно определяют каждую запись в таблице. Наиболее подходящим в качестве ключевого поля является *Счетчик*, так как значения в данном поле являются уникальными (т. е. исключают повторы).

При создании таблиц в режиме конструктора ключевое поле устанавливается автоматически.

Откройте созданные Вами таблицы в режиме **Конструктор** и проверьте установленные ключевые поля:

1) в таблице *Сотрудники* ключевое поле *Код сотрудника*



Имя поля	Тип данных
Код сотрудника	Счетчик
Фамилия	Текстовый
Имя	Текстовый
Отчество	Текстовый
Должность	Текстовый
Телефон	Текстовый
Адрес	Текстовый
Дата рождения	Дата/время
Зароботная плата	Денежный
Фото	Поле объекта OLE
Электронная почта	Гиперссылка

2) в таблице *Клиенты* ключевое поле *Код клиента*

3) в таблице *Заказы* ключевое поле *Код заказа*

Если значение *Ключевых полей* не задалось автоматически, то задайте их вручную. Для этого откройте таблицу **Сотрудники** в режиме **Конструктора**.

Нажмите правой кнопкой мыши на поле **Код сотрудника** и в появившемся контекстном меню выберите команду **Ключевое поле**. Если в таблице необходимо установить несколько ключевых полей, то выделить их можно, удерживая клавишу Ctrl. Для **таблицы Клиенты** установите ключевое поле **Код клиента**, а для **таблицы Заказы** - **Код заказа**.

Создание связей между таблицами.

Существует несколько типов отношений между таблицами:

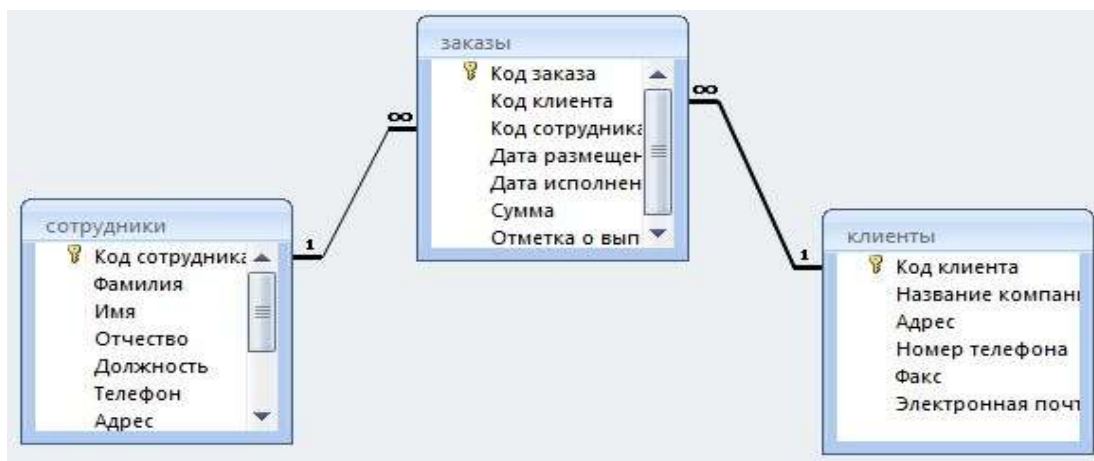
- при отношении «**один-к-одному**» каждой записи ключевого поля в первой таблице соответствует только одна запись в связанном поле другой таблицы, и наоборот. Отношения такого типа используются не очень часто. Иногда их можно использовать для разделения таблиц, содержащих много полей, для отделения части таблицы по соображениям безопасности;

заказы	
Имя поля	Тип данных
Код заказа	Счетчик
Код клиента	Числовой
Код сотрудника	Числовой
Дата размещения	Дата/время
Дата исполнения	Дата/время
Сумма	Денежный
Отметка о выполнении	Логический

- при отношении «**один-к-многим**» каждой записи в первой таблице соответствует несколько записей во второй, но запись во второй таблице не может иметь более одной связанной записи в первой таблице;

- при отношении «**многие-к-многим**» одной записи в первой таблице могут соответствовать

несколько записей во второй таблице, а одной записи во второй таблице могут соответствовать несколько записей в первой.



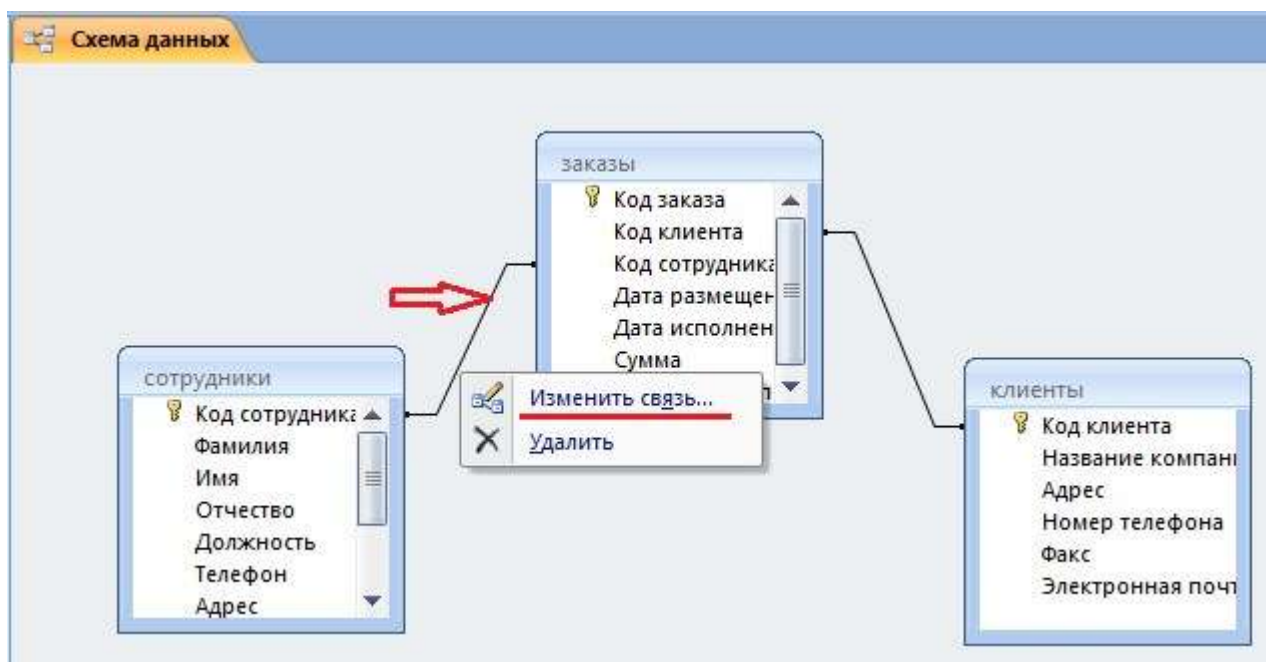
Закройте все открытые таблицы, так как создавать или изменять связи между открытыми таблицами нельзя.

Выполните команду вкладки Лента **Работа с базами данных** кнопка **Схема данных**

Если ранее никаких связей между таблицами базы не было, то при открытии окна **Схема данных** одновременно открывается окно **Добавление таблицы**, в котором выбираются нужные таблицы. Для добавления в схему данных новой таблицы необходимо щелкнуть правой кнопкой мыши на схеме данных и в контекстном меню выбрать пункт **Добавить таблицу**.

Если связи между таблицами уже были заданы, то откроется окно **Схема данных**, на котором будут отображены таблицы и связи между ними.

Отредактируйте связь между таблицами Сотрудники и Заказы, для этого щелкните правой



кнопкой мыши (ПКМ) на линию связи и в открывшемся контекстном меню выберите команду **Изменить связь**.

Откроется диалоговое окно **Изменение связей**, в котором включите флажок Обеспечение целостности данных. Это позволит предотвратить случаи удаления записей из одной таблицы, при которых связанные с ними данные других таблиц останутся без связи. Обратите внимание на **тип отношений: один-ко-многим**.

Флажки **Каскадное обновление связанных полей** и **Каскадное удаление связанных записей** обеспечивают одновременное обновление или удаление данных во всех подчиненных таблицах при их изменении в главной таблице. Параметры связи можно изменить, нажав на кнопку **Объединение**. После установления всех необходимых параметров нажмите **кнопку ОК**

Аналогично измените связь между таблицами Клиенты и Заказы.

В результате должна получиться схема данных, представленная на рисунке

На схеме данных связи отображаются в виде соединительных линий со специальными значками около таблиц. Связь «один-к-многим» помечается «1» вблизи главной таблицы (имеющей первичный ключ) и «∞» вблизи подчиненной таблицы (имеющей внешний ключ). Связь «один-к-одному» помечается двумя «1» (оба поля таблиц имеют первичные ключи). Неопределенная связь не имеет никаких знаков. Если установлено объединение, то его направление отмечается стрелкой на конце соединительной линии (ни одно из объединенных полей не является ключевым и не имеет уникального индекса).

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.1 Логическое проектирование баз данных

Практическая работа №6

Определение первичных и внешних ключей в таблицах.

Цель: получение практических навыков по освоению операций создания резервной копии базы данных и восстановления базы данных из резервной копии на другом ПК.

Выполнив работу, Вы будете: уметь:

У2 Использовать язык запросов для программного извлечения сведений из баз данных,



от пользователя с полными правами (Рисунок 24).

Материальное обеспечение:

Методические указания для выполнения практических работ, компьютер, программное обеспечение: СУБД MySQL

Порядок выполнения работы:

Для создания резервных копий базы данных в СУБД MySQL как одним из вариантов можно воспользоваться программой MySQL Administrator из пакета программ для работы с СУБД MySQL .

Чтобы создать резервную копию текущего состояния базы данных необходимо: Запустить MySQL Administrator

Выполнить подключение к СУБД MySQL

Рисунок 1 - Подключение к СУБД MySQL

После того как подключение установлено появиться основной интерфейс программы MySQL Administrator, в котором необходимо выбрать раздел **Backup**

Далее следует нажать **New Project**

Затем необходимо выбрать базу данных, резервную копию которой необходимо сделать и нажать на кнопку со значком стрелки, направленной вправо

Для начала процесса резервного копирования выбранной базы данных в файл необходимо нажать кнопку **Execute Backup Now**.

Восстановление из резервной копии базы данных также происходит через приложение **MySQL Administrator**. Для восстановления необходимо выполнить следующие шаги:

1. Запустить MySQL Administrator
2. Выполнить подключение к СУБД MySQL от пользователя с полными правами. Выполнение подключения демонстрируются на рисунке 1.
3. После того как подключение установлено появиться основной интерфейс программы MySQL Administrator, в котором необходимо выбрать раздел Restore
4. Выполнить нажатие на кнопку **Open Backup File**
5. Выбрать файл резервной копии с расширением **.sql**
6. Нажать на кнопку **Start Restore**

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2 Физическое проектирование баз данных

Практическая работа №7

Создание базы данных в СУБД на основе логической модели.

Цель: получение практических навыков модификации данных в базе.

Выполнив работу, Вы будете: уметь:

У2 Использовать язык запросов для программного извлечения сведений из баз данных.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MySQL, MS Office

Задание:

Разработать модифицирующие запросы к базе данных Туризм.

Порядок выполнения работы:

1. Создайте обобщенную таблицу **Договоры по странам**
2. Создайте запросы по заданию.

Ход работы.

Краткие теоретические сведения:

Модификация базы данных с помощью запросов на изменение **Запрос на обновление**

Запрос этого типа используется при необходимости внесения изменений во множество записей базы данных, поэтому целесообразно сделать резервную копию таблицы.

Выполняется этот вид запроса в два этапа: сначала проверяется правильность отбора обновляемых записей с помощью запроса на выборку, затем он преобразуется в запрос на обновление и выполняется повторно.

При обновлении полей следует иметь в виду, что если при проектировании таблицы в свойствах поля было указано «условие на значение», то при обновлении этого поля условие может быть нарушено, чего не допустит MS access. Поэтому нужно: или изменить условие на значение, или удалить это условие в Конструкторе. **Запрос на добавление**

Периодически убирая в архивные таблицы «старые» записи, можно увеличить быстродействие основных частей и улучшить обзорность базы данных.

Кроме того, при необходимости добавить данные в таблицу базы данных из другой базы можно также использовать запросы на добавление.

Запрос на удаление

«Старые» или неиспользуемые записи таблиц можно удалить, но обязательно сначала произвести выборку и проверить ее. Целесообразно сделать копию.

Запрос на создание

1. Создайте обобщенную таблицу **Договоры по странам**, включив в нее следующие поля:

Из таблицы **Договоры: Номер договора; Название клиента** Из таблицы **Страны: Название страны; Регион.**

Для этого:

- Создайте запрос на выборку этих данных, выполните его и проверьте результаты; ☐
Если результаты корректны, то поменяйте статус у запроса: **Запрос – Создание**

- **таблицы** – укажите новое имя таблицы **Договоры по странам**; ☐ Выполните запрос с новым статусом еще раз;
- Перейдите на вкладку **Таблицы** и убедитесь, что появилась новая таблица. Просмотрите ее.

Запрос на обновление

2. Увеличьте **Размер оклада** у менеджеров по продажам на 15%. Для этого:

- Составьте новый запрос на выборку, включив в него поля **Фамилия**, **Должность** и **Размер оклада**;
- Проверьте составленный запрос;
- Видоизмените запрос, установив ему статус «**Обновление**» (**Запрос – Обновление**). В появившейся в бланке запроса строке «**Обновление**» для поля **Размер оклада** внесите с помощью **Построить** выражение $[\text{Размер оклада}] * 1,15$;
- Выполните запрос, подтвердите обновление; сохраните запрос, дав ему имя и обратив внимание на появившийся значок у его имени; просмотрите результаты.

Запрос на добавление

3. Создайте путем копирования дубликат таблицы **Договоры** без данных, назвав ее **Договоры 2008 года**. Для этого в контекстном меню для таблицы **Договоры** выберите **Копировать**, затем выполните команду **Вставить**, в параметрах вставки укажите «**Только структуру**». Просмотрите таблицу **Договоры 2008 года** – она должна быть пустой и иметь такую же структуру, как и таблица **Договоры**.

4. Отберите в таблицу **Договоры 2008 года** записи обо всех договорах этого года. Для этого:

- Создайте запрос на выборку, включив в него все поля таблицы **Договоры** в любой последовательности, и критерий по дате, выполните его для проверки правильности;
- Измените статус запроса на «**Добавление**», в появившемся окне задайте имя таблицы для добавления **Договоры 2008 года**, обратите внимание на появление строки «**Добавление**» в бланке запроса;
- Выполните запрос и подтвердите добавление; просмотрите результаты архивации и сохраните запрос, обратив внимание на значок у его имени.

Форма представления результата:

Отчет по выполненной лабораторной работе

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2 Физическое проектирование баз данных

Практическая работа №8

Настройка индексов для ускорения запросов.

Цель: получение практических навыков по освоению операций создания запросов с помощью языка SQL

Выполнив работу, Вы будете: уметь:

У2. Использовать язык запросов для программного извлечения сведений из баз данных.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access, MySQL.

Задание :

Разработать запросы для базы данных Туризм с помощью конструктора в СУБД MS Access.

Порядок выполнения работы:

1. Выполнить задания в MySQL.
2. Вставьте скриншоты и описание в текстовый процессор MS Word. Представьте выполненную работу в виде файла с названием Практика1_Фамилия, с расширением doc, сохраненного в своей папке.

Краткие теоретические сведения:

Запрос является объектом базы данных. Он представляет собой сформулированную информационную потребность.

При работе с запросом можно выделить два этапа: формирование (проектирование) и выполнение. При выполнении запроса выбирается информация из всех таблиц базы данных в соответствии с критерием запроса.

В верхней части окна **Конструктора** размещаются нужные таблицы посредством команды **Запрос – Добавить таблицу** или та же команда в контекстном меню. В нижней части окна расположен бланк запроса, информация в него заносится путем перетаскивания нужных полей из таблиц в верхней части окна в строку «поле» или двойным щелчком мыши. При этом имя таблицы в бланке подставляется автоматически.

Наличие «галочки» в строке «Вывод на экран» означает присутствие данного поля в таблице результатов поиска. Критерии запроса устанавливаются в строке «Условия отбора» и последующих строках, связанных логическим оператором OR. Все критерии отбора, указанные в одной строке, объединяются оператором AND.

В качестве «Условия отбора» могут быть выражения (вычисляемое поле) даты, текст, которые либо вносятся вручную, либо инструментом, либо с помощью команды контекстного меню **Построить**. Константы типа Дата/Время заключаются в #.

Запросы бывают разных типов: *на выборку, на создание, на обновление, на добавление, на удаление, перекрестный, итоговый, параметрический* и др. По умолчанию формируется запрос на выборку. Тип запроса может быть преобразован в любой другой командой **Запрос**.

При создании критерия можно использовать инструмент **Построить** или такую же команду контекстного меню для категории «условие отбора».

Ход работы.

Запрос на выборку

1. Откройте базу данных **Туризм** и перейдите на вкладку **Создать - Запрос**.
2. В режиме **Конструктора** создайте и сохраните следующие запросы на выборку, определив нужные таблицы:

- список всех путешествий в определенную страну (например, Испанию);
- список всех регионов в конкретной стране (например, Англии). Сохраните запрос под именем «*Страна-Регион*»;
- все туры, проданные в 2008 году. Сохраните запрос с именем «*Туры 2008*»;
- список сотрудников, работающих с 1999 года и раньше. Сохраните запрос с именем «Ветераны». Добавьте в запрос строку «Сортировка» и установите сортировку по фамилиям.
- сотрудникам, которые родились в 1973 г., используя в качестве критерия выражение: **Between... and** (**Построить — Операторы — Сравнения — Between**), а затем повторите запрос, построив выражение с помощью знаков «<» и «>»;
- сотрудникам, фамилии которых с «Г» по «Я»;
- сотрудникам, фамилии которых начинаются с «Н» по «Я» и с «А» по «В»; ☐ индивидуальным клиентам, фамилии которых имеют вторую букву «о»;
- пяти фамилиям сотрудников, которые начинаются с букв «А» или «В». ☐ постоянным клиентам, количество договоров с которыми больше 3.

Форма представления результата:

Отчет по выполненной лабораторной работе **Критерии оценки:**

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 3.1 Основы SQL и работа с запросами

Практическая работа №9

Написание SQL-запросов для выборки данных из базы.

Цель: получение практических навыков по освоению операции конкатенации и строковых функций в запросах на языке SQL

Выполнив работу, Вы будете: уметь:

У2. Использовать язык запросов для программного извлечения сведений из баз данных.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access, MySQL.

Задание: Составить запросы для Учебной базы данных

Порядок выполнения работы:

1. Выполнить задания в MySQL.
2. Вставьте скриншоты и описание в текстовый процессор MS Word. Представьте выполненную работу в виде файла с названием Практика1_Фамилия, с расширением doc, сохраненного в своей папке.

Ход работы.

Краткие теоретические сведения:

Операция конкатенации «&» в СУБД Access позволяет соединять («склеивать») значения двух или более столбцов символьного типа или символьных констант в одну строку.

Пример:

```
SELECT SURNAME&'_'&NAME, STIPEND FROM STUDENT  
WHERE KURS = 4 AND STIPEND> 0;
```

Функции преобразования символов в строке

LCASE(строка)-перевод в строчные символы **UCASE(строка)**-перевод в прописные символы **Пример:**

```
SELECT LCASE (SURNAME), UCASE (NAME) FROM STUDENT  
WHERE KURS = 4 AND STIPEND> 0;
```

Строковые функции

LEN (строка) –определение длины строки □тип возвращаемого значения INT

□функция возвращает NULL, если строка имеет NULL значение **LEFT (строка, длина)** – выделение левой подстроки

RIGHT (строка, длина) – выделение правой подстроки **MID (строка, начало, количество)** – выделение подстроки **LTRIM (строка)** – удаляет пробелы в начале строки **RTRIM (строка)** – удаляет пробелы в конце строки

STR (число) – выполняет конвертирование числового значения в символьный формат **INSTR (строка, подстрока [, <начало поиска> [, <номер вхождения>]])** -

возвращает позицию найденной подстроки (тип возвращаемого значения- INT), где

<начало поиска> задает начальную позицию в строке для поиска<подстроки>. Если не задано, то по умолчанию применяется значение 1;

<номер вхождения> задает порядковый номер искомой подстроки. Если не задан, то по умолчанию применяется значение 1.

Значение выражений в <начале поиска> или <номера вхождения> должны иметь беззнаковый целый тип или приводиться к этому типу

Примеры:

```
SELECT SURNAME,LEN (SURNAME),KURS FROM STUDENT  
WHERE KURS = 3;
```

```
SELECT SURNAME, Right(SURNAME,2) FROM STUDENT  
WHERE KURS IN (2, 3, 4);
```

Задания:

1. Составьте запрос для таблицы STUDENT «Учебной базы данных» таким образом, чтобы выходная таблица была представлена одним столбцом, содержащим последовательность, разделенных символом «;» значений всех столбцов этой таблицы; результат необходимо представить прописными символами (например, **ИВАН;ИВАНОВ;150;1;ОПЁЛ;3/12/82;10**).

2. Составьте запрос для таблицы STUDENT «Учебной базы данных» таким образом, чтобы выходная таблица содержала один столбец в следующем виде: **Б.КУЗНЕЦОВ; место жительства – БРЯНСК; родился – 8.12.81**.

3. Составьте запрос для таблицы STUDENT «Учебной базы данных» таким образом, чтобы столбец выходной таблицы был представлен в следующем виде: **Б;Кузнецов;место жительства-Брянск;родился-08.12.1981**

4. Составьте запрос для таблицы STUDENT «Учебной базы данных» таким образом, чтобы выходная таблица содержала один столбец в следующем виде: **Борис Кузнецов родился в 1981 году**.

5. Выведите имена и фамилии студентов и объем получаемой ими стипендии, увеличенной в 10 раз.

6. Составьте запрос для таблицы STUDENT «Учебной базы данных» следующим образом: **БОРИС КУЗНЕЦОВ родился в 1981 году**, только для студентов 1, 2 и 4-го курсов.

7. Составьте запрос для таблицы UNIVERSITY «Учебной базы данных» таким образом, чтобы выходная таблица содержала один столбец в следующем виде: **Код – 10;ВГУ – г.ВОРОНЕЖ;Рейтинг=296**.

Форма представления результата:

Отчет по выполненной лабораторной работе **Критерии оценки:**

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 3.1 Основы SQL и работа с запросами

Практическая работа №10

Создание и модификация данных в таблицах.

Цель: получение практических навыков по освоению операций создания запросов с вычисляемыми полями с помощью языка SQL

Выполнив работу, Вы будете: уметь:

У2. Использовать язык запросов для программного извлечения сведений из баз данных.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access, MySQL.

Задание 1:

Разработать запросы для базы данных Туризм с помощью конструктора в СУБД MS Access.

Порядок выполнения работы:

1. Выполнить задания в MySQL.
2. Вставьте скриншоты и описание в текстовый процессор MS Word. Представьте выполненную работу в виде файла с названием Практика1_Фамилия, с расширением doc, сохраненного в своей папке.

Краткие теоретические сведения:

3. Запрос является объектом базы данных. Он представляет собой сформулированную информационную потребность.

4. При работе с запросом можно выделить два этапа: формирование (проектирование) и выполнение. При выполнении запроса выбирается информация из всех таблиц базы данных в соответствии с критерием запроса.

5. В верхней части окна **Конструктора** размещаются нужные таблицы посредством команды **Запрос – Добавить таблицу** или та же команда в контекстном меню. В нижней части окна расположен бланк запроса, информация в него заносится путем перетаскивания нужных полей из таблиц в верхней части окна в строку «поле» или двойным щелчком мыши. При этом имя таблицы в бланке подставляется автоматически.

6. Наличие «галочки» в строке «Вывод на экран» означает присутствие данного поля в таблице результатов поиска. Критерии запроса устанавливаются в строке «Условия отбора» и последующих строках, связанных логическим оператором OR. Все критерии отбора, указанные в одной строке, объединяются оператором AND.

7. В качестве «Условия отбора» могут быть выражения (вычисляемое поле) даты, текст, которые либо вносятся вручную, либо инструментом, либо с помощью команды контекстного меню **Построить**. Константы типа Дата/Время заключаются в #.

8. Запросы бывают разных типов: *на выборку, на создание, на обновление, на добавление, на удаление, перекрестный, итоговый, параметрический* и др. По умолчанию формируется запрос на выборку. Тип запроса может быть преобразован в любой другой командой **Запрос**.

9. При создании критерия можно использовать инструмент **Построить** или такую же команду контекстного меню для категории «условие отбора».

10. 1. Вычисляемые поля в запросах

11. С помощью запросов можно задать вычисления над данными и сделать вычисляемое поле новым полем в наборе данных. Для создания нового поля в пустой ячейке строки **Поле** в бланке запроса вводится формула:

12. Имя поля:	выражение
----------------------	------------------

13. Для построения выражений имеется специальное средство – **Построитель** выражений, вызываемый правой кнопкой мыши на поле или кнопкой **Построить**.

14. В верхней части размещается область ввода. Нижняя содержит три списка для выбора имен полей и функций. В папке **Функции** размещаются встроенные функции, сгруппированные по категориям.

15. 2. Параметрические запросы

16. Условия запроса могут быть включены непосредственно в бланк запроса, но для того чтобы сделать его более универсальным, можно вместо конкретного значения отбора включить в запрос параметр, т.е. создать параметрический запрос. Для этого в строку «условия отбора» вводится фраза в квадратных скобках, которая будет выводиться в качестве «подсказки» в процессе диалога, например, [введите фамилию]. Таких параметров может быть несколько,

каждый для своего поля.

Задания:

Запросы с вычисляемыми полями

1. Создайте запрос для расчета ведомости заработной платы для сотрудников агентства, включив в нее следующие поля: **Фамилия сотрудника, Размер оклада, Стаж, Надбавка, Налог, На руки.**

Для поля **Стаж** нужно использовать формулу, построенную с помощью кнопки **Построить**, в которой учитывается сегодняшняя дата и **Дата найма** на работу:

Стаж = (Date()-Сотрудники!ДатаНайма)\365

Для поля **Надбавка** нужно исходить из того, что она составляет 20% от **Размера оклада**, если **Стаж** меньше 5 лет, и 30% — если стаж больше 5 лет:

If ([стаж]<5;0,2*[Сотрудники!Размер оклада]; 0,3* [Сотрудники!Размер оклада])

Поле **Налог** рассчитывается как 13% от **Размера оклада**: [Сотрудники!Размер оклада] *0,13

Поле **На руки** рассчитывается как: [Размер оклада]+[надбавка]—[налог].

В результате выполнения запроса будет получена новая ведомость.

2. Создайте запрос для определения стоимости путевок корпоративных клиентов, включив в него поля **Клиент, Стоимость путевки**

Стоимость путевки = Sum(договоры![Цена тура] *договоры![Число туристов])

Параметрические запросы

3. Сформируйте запрос для выборки всех туров по названию страны.

4. Создайте запрос для получения данных на сотрудников, работающих по турам в конкретную страну.

5. Создайте запрос по всем клиентам, оформившим договоры в определенную страну и регион.

Форма представления результата:

Отчет по выполненной лабораторной работе

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату. Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 3.2. Оптимизация запросов и работы баз данных

Практическая работа №11

Оптимизация сложных SQL-запросов.

Цель: получение практических навыков по освоению вычислений в запросах и применение упорядочения выборки.

Выполнив работу, Вы будете: уметь:

У2. Использовать язык запросов для программного извлечения сведений из баз данных.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access, MySQL.

Задание 1:

Разработать запросы для Учебной базы данных

Порядок выполнения работы:

1. Выполнить задания в MySQL.

2. Вставьте скриншоты и описание в текстовый процессор MS Word. Представьте выполненную работу в виде файла с названием Практика1_Фамилия, с расширением doc, сохраненного в своей папке.

Ход работы.

Краткие теоретические сведения:

Оператор SELECT (выбрать) языка SQL является самым важным и самым часто используемым оператором. Он предназначен для выборки информации из таблиц базы данных. Упрощенный синтаксис оператора SELECT выглядит следующим образом: SELECT [DISTINCT] <список атрибутов>

FROM <список таблиц> [WHERE <условие выборки>]

[ORDER BY <список атрибутов>] [GROUP BY <список атрибутов>] [HAVING <условие>]

[UNION <выражение с оператором SELECT>];

В квадратных скобках указаны элементы, которые могут отсутствовать в запросе. Ключевое слово SELECT сообщает базе данных, что данное предложение является

запросом на извлечение информации. После слова SELECT через запятую перечисляются наименования полей, содержимое которых запрашивается.

Обязательным ключевым словом в предложении-запросе SELECT является слово FROM (из). За ключевым словом FROM указывается список разделенных запятыми имен таблиц, из которых извлекается информация.

Например,

```
SELECT NAME, SURNAME FROM STUDENT;
```

Любой SQL-запрос должен заканчиваться символом «;».

Если необходимо вывести значения всех столбцов таблицы, то можно вместо перечисления их имен использовать символ «*/»

```
SELECT *
```

```
FROM STUDENT;
```

Для исключения из результата SELECT-запроса повторяющихся записей используется ключевое слово DISTINCT (отличный). Если запрос SELECT извлекает множество полей, то DISTINCT исключает дубликаты строк, в которых значения всех выбранных полей идентичны.

Использование в операторе SELECT предложения, определяемого ключевым словом WHERE (где), позволяет задавать выражение условия, принимающее значение истина или ложь для значений полей строк таблиц, к которым обращается оператор SELECT. Предложение WHERE определяет, какие строки указанных таблиц должны быть выбраны.

3.

Упорядочение выходных полей (ORDER BY) .

ORDER BY позволяет упорядочивать записи в соответствии со значениями одного или нескольких выбранных полей. При этом можно задать возрастающую (ASC) и убывающую (DESC) последовательность сортировки для каждого из столбцов.

По умолчанию принята возрастающая последовательность сортировки.

Задания:

1. Выбрать все данные из таблицы предметов обучения SUBJECT с упорядочением по наименованиям предметов.

```
SELECT * FROM SUBJECT
```

```
ORDER BY SUBJ_NAME;
```

2. Тот же список, но упорядоченный в обратном порядке.

```
SELECT * FROM SUBJECT
```

```
ORDER BY SUBJ_NAME DESC;
```

3. Упорядочить выводимый список предметов по семестрам, а внутри семестров – по наименованиям предметов.

```
SELECT * FROM SUBJECT  
ORDER BY SEMESTER, SUBJ_NAME;
```

4. При упорядочении вместо наименований столбцов можно указывать их номера, имея, однако, в виду, что данном случае это номера столбцов указанные при определении выходных данных в запросе, а не номера столбцов в таблице. Полем с номером 1 является первое поле, указанное в предложении ORDER BY- независимо от его расположения в таблице.

```
SELECT SUBJ_ID, SEMESTER FROM SUBJECT  
ORDER BY 2 DESC;
```

Если в поле, которое используется для упорядочения, существуют NULL- значения, то все они размещаются в конце или предшествуют всем остальным значениям этого поля.

Форма представления результата:

Отчет по выполненной лабораторной работе

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено

Тема 3.2. Оптимизация запросов и работы баз данных

Практическая работа №12

Настройка параметров производительности базы данных.

Цель: получение практических навыков по формированию запросов с составными условиями.

Выполнив работу, Вы будете: уметь:

У2. Использовать язык запросов для программного извлечения сведений из баз данных.

Материальное обеспечение:

Методические указания для выполнения практических работ, вариант задания, компьютер, программное обеспечение: MS Access, MySQL.

Задание 1:

Разработать запросы для Учебной базы данных

Порядок выполнения работы:

1. Выполнить задания в MySQL.
2. Вставьте скриншоты и описание в текстовый процессор MS Word. Представьте выполненную работу в виде файла с названием Практика1_Фамилия, с расширением doc, сохраненного в своей папке.

Краткие теоретические сведения:

Использование в операторе SELECT предложения, определяемого ключевым словом WHERE(где), позволяет задавать выражения условия (предикат), принимающее значение *истина* или *ложь* для значений полей строк таблиц, к которому обращается оператор SELECT. Предложение WHERE определяет, какие строки *указанных таблиц должны быть выбраны*. В таблицу, являющуюся результатом запроса, включается только те строки, для которых условие(предикат), в предложении WHERE, принимает значение *истина*.

Операторы **IN** (равен любому из списка) **NOT IN** (не равен ни одному из списка) используется для сравнения проверяемого значения поля с заданным списком. Этот список значений указывается в скобках справа от оператора IN. Построенный с использованием IN предикат (условие) считается истинным, если значения поля, имя которого указано слева от IN,

совпадает (подразумевается точное совпадение) с одним из значений, перечисленных в списке, указанном в скобках справа от IN.

Предикат, построенный с использованием NOT IN, считается истинным, если значение поля, имя которого указано слева от NOT IN, не совпадает ни с одним из значений, перечисленных в списке, указанном в скобках от NOT IN.

Оператор **BETWEEN** используется для проверки условия вхождения значения поля в заданный интервал, то есть вместо списка значений атрибута этот оператор задает границы его изменения.

Оператор **LIKE** применим только к символьным полям типа CHAR или VARCHAR. Этот оператор просматривает строковые значения полей с целью определения, входит ли заданная в операцию LIKE подстрока (образец поиска) в символьную строку-значение проверяемого поля. Можно применять шаблон:

Символ подчеркивания «?» указанный в шаблоне, определяет возможность наличия в указанном месте одного любого символа;

Символ «*» допускает присутствие в указанном месте проверяемой строки последовательности любых символов произвольной длины.

Like [A-K]* – значения атрибутов символьного типа, начинающиеся с A по K

Like ![A-K]* – значения атрибутов символьного типа, начинающиеся на любые символы кроме символов с A по K (с Л по Я)

Like [A,K]* – значения атрибутов символьного типа, начинающиеся с A или K Оператор NULL используется для проверки содержимого поля на наличие в нем пустого значения. **IS NULL** (является пустым), **IS NOT NULL** (является не пустым).

Задания:

1. Написать запрос, выполняющий выборку имен всех студентов с фамилией Петров, сведения о которых находятся в таблице STUDENT

```
SELECT NAME, SURNAME FROM STUDENT  
WHERE SURNAME = 'Петров';
```

2. В задаваемых предложениях WHERE условиях могут использоваться операции сравнения, определяемые операторами =(равно), >(больше), <(меньше), >=(больше или равно), <=(меньше или равно), <>(не равно), а также логические операторы AND, OR и NOT. Запрос для получения имен и фамилий студентов, обучающихся на третьем курсе и получающих стипендию (размер стипендии больше нуля), будет выглядеть таким образом:

```
SELECT NAME, SURNAME FROM STUDENT  
WHERE KURS = 3 AND STIPEND > 0;
```

3. . Получить из таблицы EXAM_MARKS сведения о студентах, имеющих экзаменационные оценки только 4 и 5.

```
SELECT *  
FROM EXAM_MARKS WHERE MARK IN (4, 5);
```

4. Получить из таблицы EXAM_MARKS сведения о студентах, не имеющих ни одной экзаменационной оценки только, 4 и 5.

```
SELECT *  
FROM EXAM_MARKS WHERE MARK NOT IN (4, 5);
```

5. Запрос на вывод записей о предметах, на изучение которых отводится количество часов, находящееся в пределах между 30 и 40, имеет вид:

```
SELECT * FROM SUBJECT  
WHERE HOUR BETWEEN 30 AND 40;
```

6. Написать запрос, выбирающий из таблицы STUDENT сведения о студентах, фамилии которых начинаются на букву «К». SELECT *
FROM STUDENT
WHERE SURNAME LIKE —K*!;
7. Вывести все данные студента с номером 265;
8. Вывести информацию о студентах с именем Вадим;
9. Выбрать фамилии студентов со 2-го и 3-го курсов получающих стипендию; 10. Вывести имена, фамилии и даты рождения студентов 5 курса;
10. Вывести информацию о студентах из Воронежа;
11. Выбрать фамилию, имя, курс, город студентов получающих стипендию от 100 до 200 рублей;
12. Вывести список идентификаторов университетов, исключая повторения.

Форма представления результата:

Отчет по выполненной лабораторной работе

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.