

**Кафедра автоматизированных систем обработки
информации и управления**

БАЗЫ ДАННЫХ И БАЗЫ ЗНАНИЙ

Учебно-методическое пособие

Сургут
Издательский центр СурГУ
2022

УДК 004.6(076.5)
ББК 32.973.26-018.2я73
Б179

Печатается по решению
редакционно-издательского совета СурГУ

Рецензенты:

канд. физ.-мат. наук, доцент кафедры информатики и
вычислительной техники СурГУ **С. А. Лысенкова**;
канд. техн. наук, доцент кафедры автоматизации и компьютерных
систем СурГУ **П. В. Гришмановский**

Б179

Базы данных и базы знаний : учеб.-метод. пособие / сост.: М. В.
Юрчишина, А. В. Гавриленко, А. В. Никифоров ; Сургут. гос. ун-т. –
Сургут : ИЦ СурГУ, 2022. – с.

В учебно-методическом пособии рассмотрены основные
разделы дисциплины «Базы данных и базы знаний» направления
бакалаврской подготовки УГН 01.00.00, 09.00.00. и даны
методические рекомендации для выполнения лабораторных работ.

УДК 004.6(076.5)
ББК 32.973.26-018.2я73

© Юрчишина М. В., Гавриленко А. В.,
Никифоров А. В., составление, 2022
© Сургутский государственный
университет, 2022

ОГЛАВЛЕНИЕ

Введение.....	5
Раздел 1. Базы данных.....	6
Тема 1.1 Банк данных.....	6
Тема 1.2 Концептуальная модель	7
Лабораторная работа № 1. Текстовое описание предметной области	
Лабораторная работа № 2. Построение ER-модели	12
Тема 1.3. Отношения и их нормализация.....	14
Лабораторная работа № 3. Понятие отношения.....	15
Лабораторная работа № 4 Нормализация отношения.....	18
Тема 1.4. Логическое моделирование.....	19
Лабораторная работа № 5 Логическая модель.	21
Тема 1.5. Физическая модель	22
Лабораторная работа № 6. Физическая модель	24
Тема 1.6. Иерархическая и сетевая модели представления данных	26
Лабораторная работа № 7. Модели представления данных ...	27
Лабораторная работа № 8. Создание таблиц базы данных в предложенной СУБД	29
Лабораторная работа № 9 Формы для отображения таблиц	
Лабораторная работа № 10. Маски ввода	33
Лабораторная работа № 11 Создание запросов в режиме мастера и конструктора	35
Лабораторная работа № 12 SQL (structured query language)	
Лабораторная работа № 13. Создание форм запросов.....	40
Лабораторная работа № 14. Отчеты	42
Лабораторная работа № 15 Элементы навигации, макросы и модули, разграничение доступа, сжатие БД	
Раздел 2. Технология управления Базой данных из среды программирования	
Тема 2.1 Объектно-реляционное отображение.....	47
Лабораторная работа № 16. Object-relational mapping	
Раздел 3. Базы знаний.....	66
Тема 3.1. Основные понятия	67
Лабораторная работа № 17. База знаний	68
Раздел 4. Курсовой проект.....	73
Задание на курсовой проект	73

Варианты предметных областей на курсовой проект	74
Требования к оформлению	75
Методические указания по выполнению курсового проекта	75

ВВЕДЕНИЕ

«Базы данных и базы знаний» – классическая базовая дисциплина для студентов, направление обучения которых связано с информационными технологиями. Практически любая информационная система основывается на некоторой базе данных или базе знаний. Формирование и правильная организация данного элемента являются очень важными и имеют множество особенностей, которые и рассмотрены в данной дисциплине. Основной целью дисциплины является изучение и освоение на практике теории баз данных и баз знаний методов, средств обследования предметной области, построение моделей баз данных с использованием типичных систем управления базами данных, в том числе отечественного производства и разработки собственных систем с использованием современных информационных технологий, формирование умения осваивать методики использования современных программных средств для управления базами данных в банках данных, формирование систематических знаний о современных моделях представления знаний и программных средствах управления ими.

Дисциплина «Базы данных и базы знаний» требует предварительного освоения первичных навыков работы с компьютером, умения осваивать новые программные продукты.

Во время изучения дисциплины важно конспектировать новую информацию для более качественного освоения. При выполнении лабораторных работ важным является не только непосредственное решение поставленных задач, но и оформление отчетов. Формирование отчетности систематизирует полученные навыки и формирует способность формализации.

Во избежание нарушения ст. 1274 ГК РФ об авторском праве и требований к оформлению печатных изданий следует оформить ссылки на источники заимствования, либо ввести в этот раздел уведомление следующего содержания:

Учебно-методическое пособие составлено с использованием материалов следующих изданий (перечислить основные) либо : изданий, указанных в списке литературы и открытых источников в интернет.

Раздел 1

БАЗЫ ДАННЫХ

Прежде всего необходимо вспомнить изученные ранее базовые понятия, наиболее часто используемые в данной дисциплине.

Данные – сведения, факты, показатели, выраженные в некоторой форме, пригодной для обработки, передачи и хранения с помощью ЭВМ.

Информация – это вся совокупность сведений об окружающем нас мире, о всевозможных протекающих в нем процессах, которые могут быть восприняты живыми организмами, электронными машинами и другими информационными системами.

Знание – результат процесса познания действительности, получивший подтверждение в практике; адекватное отражение объективной реальности в сознании человека (представления, понятия, суждения, теории).

Логика – это наука о законах и формах мышления, методах познания и условиях определения истинности знаний и суждений.

Автоматизированная информационная система (далее – АИС) представляет собой совокупность данных, экономико-математических методов и моделей, технических, программных средств и специалистов, предназначенную для обработки информации и принятия управленческих решений.

Тема 1.1

Банк данных

Ключевым понятием в теории баз данных является Банк данных. Банк данных (далее – БНД) – это система специальным образом организованных данных (баз данных), программных, технических, языковых, организационно-методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных. Из данного определения очевидно, что понятие «База данных» является элементом банка данных. Но формирование данного элемента требует особенного внимания.

База данных (далее – БД) представляет собой совокупность специальным образом организованных данных, хранимых в памяти

вычислительной системы и отображающих состояние объектов и их взаимосвязей в рассматриваемой предметной области.

Система управления базами данных (далее – СУБД) – это комплекс языковых и программных средств, предназначенный для создания, ведения и совместного использования БД многими пользователями.

Для эффективного функционирования БД компоненты его должны быть согласованы. При этом проектирование БД требует понимания особенностей работы с данными, поэтому необходимо изучить основные принципы и этапы построения БД.

Тема 1.2

Концептуальная модель

Основные понятия. Построение концептуальной или инфологической модели – это формализация описания предметной области. Концептуальное моделирование является важнейшим этапом проектирования БД. Именно этот этап фактически описывает формализацию реальной предметной области, т. е. переход от реальных объектов к их модельному представлению в БД. Именно здесь важно осознать цели создаваемой БД и выбрать те характерные особенности, которые будут существенными при работе проектируемой БД.

Существует множество подходов к построению концептуальной модели предметной области. Поскольку концептуальное моделирование является первичной формализацией, зачастую модель представления данных еще не выбрана. Важно построить универсальную модель, не основывающуюся ни на каких формах представления данных.

Перед построением концептуальной модели удобно выполнить описание предметной области в текстовом формате. Это не обязательный этап, однако, в случае, когда проектировщик БД не знаком с предметной областью, это значительно упростит процесс понимания всех процессов и явлений в ней.

Одним из наиболее удобных и часто используемых способов построения концептуальной модели является представление предметной области в виде диаграммы «сущность-связь», так называемой ER-модели. Для построения ER-модели можно

воспользоваться нотацией Чена. Она является достаточно простой для изучения, содержит небольшое количество элементов и правил, при этом позволяет наглядно отрисовать сущности и связи предметной области.

Выделяют три вида сущностей: стержневая, ассоциативная (ассоциация) и характеристическая (характеристика). Кроме этого, во множестве ассоциативных сущностей также определяют подмножество обозначений.

Стержневая (сильная) сущность – независимая от других сущность. Стержневая сущность не может быть ассоциацией, характеристикой или обозначением (см. ниже).

Ассоциативная сущность (или ассоциация) выражает собой связь «многие ко многим» между двумя сущностями. Является вполне самостоятельной сущностью. Например, между сущностями «мужчина» и «женщина» существует ассоциативная связь, выражаемая ассоциативной сущностью «брак».

Характеристическую сущность еще называют слабой сущностью. Она связана с более сильной сущностью связями «один ко многим» и «один к одному». Характеристическая сущность описывает или уточняет другую сущность. Она полностью зависит от нее и исчезает с исчезновением последней. Например, сущность «зарплата» является характеристикой конкретных работников предприятия и не может в таком контексте существовать самостоятельно – при удалении экземпляра сущности «работник» должны быть удалены и экземпляры сущности «зарплата», связанные с удаляемым работником.

Дополнительно можно выделить такой тип, как обозначение.

Обозначение – это такая сущность, с которой другие сущности связаны по принципу «многие к одному» или «один к одному». Обозначение, в отличие от характеристики, является самостоятельной сущностью. Например, сущность «факультет» обозначает принадлежность студента к данному подразделению института, но является вполне самостоятельной.

В нотации Чена для обозначения сущностей используется прямоугольник, внутри которого указывается имя сущности. Атрибуты сущности обозначаются овалами, внутри которых указано название атрибута. Графически принадлежность атрибута сущности обозначается соединительной линией. Название ключевого атрибута выделяется подчеркиванием или помечается специальным символом

перед первым символом. При построении модели с большим количеством сущностей и атрибутов допустимо не отображать сущности. Для обозначения связи используется ромб, внутри которого указано название связи. Ромб соединяется линиями с сущностями, участвующими в связи. На связывающих линиях можно указать тип связи. Обычно используются символы «1» и «М» или «1» и «∞», также встречаются обозначения в виде одинарного и двойного отрезка, перпендикулярного линии самой связи. Для указания ассоциативной сущности используют ромб, заключенный в прямоугольник или шестиугольник, для характеристической – трапецию, для обозначения – параллелепипед. Данный набор не является полным, но чаще всего его достаточно для построения ER-модели предметной области. Такая модель является универсальной, не привязана к модели представления данных. Однако наиболее распространенной моделью представления данных много лет являлась реляционная, для упрощения связи отображаются просто линией без ромба (для реляционной модели чаще всего название связи не имеет значения и может быть не определено), а ромб используется для ассоциативной сущности.

Поскольку нотация не является строгой, встречаются несколько вариаций обозначений. Поэтому принято на схеме изображать легенду.

Лабораторная работа № 1. Текстовое описание предметной области

Первичное знакомство с предметной областью удобно начинать с текстового описания, которое может быть выполнено экспертами, работающими в этой области. Такое описание будет наиболее точным.

Задание. В лабораторной работе необходимо в текстовом формате на естественном языке выполнить описание предложенной предметной области. Описание должно включать обязательные элементы:

1. Название предметной области.
2. Название и краткое пояснение всех элементов предметной области, имеющих отношение к проектируемой базе данных.
3. Описание всех характеристик выделенных сущностей.

4. Описание всех отношений между выделенными сущностями.

5. Описание всех процессов предметной области, требующих автоматизации в проектируемой БД.

Пример. БТИ (бюро технической инвентаризации) входит в число государственных институтов и осуществляет функции сбора и систематизации данных о:

- местонахождении объекта недвижимости;
- качестве;
- цене;
- техническом состоянии.

В базе организации собраны сведения обо всех домах, квартирах, земельных участках, находящихся в населенном пункте, а также об их изменениях на протяжении определенного промежутка времени.

В отличие от Росстата, который занимается регистрацией прав собственности на недвижимость, БТИ осуществляет только учетную деятельность и техническую инвентаризацию объектов.

Техническая инвентаризация объектов недвижимости – это проверка и определение на конкретную дату наличия, местоположения, назначения, фактического использования, состава и состояния объекта капитального строительства.

В основе работы всех сотрудников БТИ лежит полное сопровождение разнообразных сделок с недвижимостью. Для этого они используют нормативно-правовые акты и законы согласно инструкциям, оперируют огромным количеством правил стандартизации. Именно поэтому документы, выданные в БТИ, являются основными при регистрации объекта.

В БТИ обращаются по любым вопросам, связанным с недвижимым имуществом. Сотрудники бюро:

1. Осуществляют оценку недвижимости.
2. Оформляют документы, необходимые для заключения договоров купли-продажи.
3. Проводят инвентаризацию объектов.

Без БТИ нельзя ни заключить договор купли-продажи, ни провести инвентаризацию объекта. Даже сделать перепланировку собственной квартиры не получится без данной организации. Допустим, если вы решили сделать перепланировку в доме, вам

потребуется получить новый технический паспорт с чертежами жилья согласно проведенным работам. Оформлением такого паспорта занимаются в БТИ. Чтобы его получить, достаточно оформить заявку в бюро по месту своего проживания, заплатить пошлину, а через установленное время прийти за документом.

Технический паспорт – документ, отображающий результаты проведения технической инвентаризации, содержащих основные технические характеристики объекта недвижимости.

Помимо технического паспорта, в БТИ занимаются изготовлением различных видов документов. Наиболее актуальные из них перечислены в табл. 1.

Таблица 1

Перечень некоторых документов, которые выдает БТИ

Название документа	Описание
Позэтажный план	В данном документе содержится чертеж жилого помещения. Дополнением к нему выступает экспликация, содержащая подробные сведения о параметрах объекта недвижимого имущества, включая метраж жилплощади, как общий, так и жилой
Копия техпаспорта	Документ, выдаваемый БТИ для осуществления перепланировки. Включает: поэтажный план с дополнением в виде экспликации, а также адресный план
Выписка из техпаспорта на здания и строения (форма справки 1а)	Содержит сведения о метраже, этажности, дате постройки здания, о материалах, использованных для строительства стен и перегородок
Выписка из техпаспорта БТИ (форма справки 1б)	Выписка из техпаспорта БТИ (форма справки 1б) – документ содержит данные БТИ, а не на домовладение
Справка из БТИ о состоянии здания/помещения (форма 5)	Содержит данные об амортизации здания/помещения, о наличии инженерных коммуникаций и объектов инфраструктуры, о дате ввода в эксплуатацию, параметры размеров по этажам

Окончание табл. 1

Справка по форме № 3	Необходима для совершения действий с земельным участком
----------------------	---

Справка об инвентаризационной стоимости объекта недвижимости	Документ нужен для оформления сделок, вступления в наследство. Содержит: цену объекта с учетом его износа, показывает реальную цифру остаточной стоимости помещения
--	---

Благодаря учету недвижимого имущества органы власти в населенном пункте могут делать прогнозы относительно его дальнейшего развития, определять наиболее перспективные направления в строительной сфере или, напротив, принимать решения по поводу сноса и реконструкции отдельных объектов.

Лабораторная работа № 2. Построение ER-модели

В данной лабораторной работе необходимо изобразить ER-модель предложенной предметной области, с использованием нотации Чена.

Задание:

1. Для построения ER-модели необходимо выделить сущности, оказывающие влияние на процессы предметной области, которые планируется автоматизировать.
2. Определить типы сущностей (стержневые, характеристики, ассоциации, обозначения).
3. Изобразить сущности в соответствии с нотацией.
4. Установить связи между сущностями, указав их тип (название не обязательно).
5. Изобразить параметры всех сущностей, оказывающие влияние на проектируемую БД.
6. Выделить ключевые параметры для всех сущностей.
7. Проверить визуально полученную модель, при необходимости исправить.

Пример:

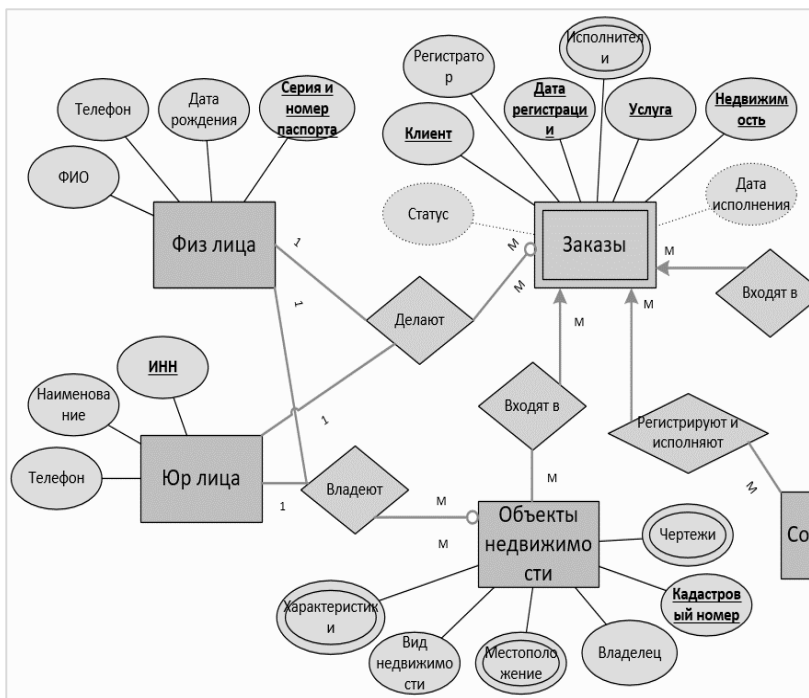


Рис. 1. Фрагмент ER-модели предметной области «БТИ»

Описание сущностей инфологической модели:

1. Физ. лица – содержит сведения о физических лицах.
2. Юр. лица – содержит сведения об юридических лицах.
3. Услуги – содержит информацию об услугах, которые предоставляет БТИ.
4. Объекты недвижимости – содержит информацию о недвижимостях клиентов.
5. Сотрудники – содержит информацию о сотрудниках БТИ.
6. Заказы – содержит информацию о заказах.

Диаграмма отражает сущности, их атрибуты, с пометкой ключевых, и связи между сущностями.

Тема 1.3

Отношения и их нормализация

Под отношением в теории реляционных БД понимается таблица с данными, состоящая из заголовка и тела. Заголовок представляет собой множество атрибутов (именованных вхождений типа данных в заголовок отношения), а тело – множество кортежей, соответствующих заголовку. При этом важным является уникальность каждого кортежа. Основные понятия представлены в табл. 2.

Таблица 2

Элементы реляционной модели

Элемент реляционной модели	Форма представления
Отношение	Таблица
Схема отношения	Строка заголовков столбцов таблицы (заголовок таблицы)
Кортеж	Строка таблицы
Атрибут	Заголовок столбца таблицы
Домен	Множество допустимых значений атрибута
Значение атрибута	Значение поля в записи
Первичный ключ	Один или несколько атрибутов, однозначно идентифицирующий каждый из его кортежей
Тип данных	Тип значений элементов таблицы
Кардинальность	Количество кортежей в отношении

Нормальная форма – требование, предъявляемое к структуре таблиц в теории реляционных баз данных, для устранения из базы данных избыточных функциональных зависимостей между атрибутами.

Нормализация позволяет исключить избыточность данных, что обеспечивает целостность данных.

Считается, что отношение находится в первой нормальной форме, если все его атрибуты являются простыми, все их значения атомарны.

Отношение находится во второй нормальной форме, если оно находится в первой нормальной форме и каждый не ключевой атрибут неприводимо зависит от первичного ключа. Неприводимость означает, что в составе потенциального ключа отсутствует меньшее подмножество атрибутов, от которого можно также вывести данную функциональную зависимость. Для приведения отношения ко второй нормальной форме обычно выбирают наборы атрибутов, зависящих от части первичного ключа, и выделяют их в отдельные отношения, исключая из первоначального.

Отношение находится в третьей нормальной форме, когда оно находится во второй нормальной форме и каждый неключевой атрибут нетранзитивно зависит от первичного ключа. Для приведения отношения к третьей нормальной форме выносят все неключевые поля, содержимое которых может относиться к нескольким записям таблицы в отдельные таблицы.

Лабораторная работа № 3. Понятие отношения

Составление отношений имеет большое значение в теории реляционных БД. В данной работе необходимо научиться составлять отношение, моделирующее проектируемую БД, научиться оперировать понятиями реляционных БД: отношение, атрибут, домен, кортеж, кардинальность и т. д.

Также данная работа является подготовительной для освоения понятия нормализации.

Задание:

1. Составить общее отношение, включающее в качестве столбцов все атрибуты всех выбранных сущностей.
2. Сформировать БД мощностью 15 или более. Определить кардинальность построенного отношения.
3. Указать все названия атрибутов.
4. Привести 3 примера кортежей.
5. Обозначить схему данных.
6. Описать домен любого атрибута.

Пример. Рассмотрим в качестве предметной области успеваемость студентов одного направления обучения по всем дисциплинам. Имеется несколько групп, с некоторым количеством студентов 10–30 человек. Каждый студент характеризуется ФИО, датой рождения и сдает по каждой дисциплине зачет или экзамен (форма контроля). Каждая дисциплина характеризуется количеством часов, количеством зачетных единиц, семестром (причем одна дисциплина может преподаваться в нескольких семестрах), преподавателем. Общая схема отношения (рис. 2) будет включать все эти характеристики.

Группа	Фамилия	Имя	Отчество	Дата рождения	Название дисциплины	
--------	---------	-----	----------	---------------	---------------------	--

Форма контроля	Количество часов	Количество ЗЕТ	Семестр	Преподаватель	Оценка
----------------	------------------	----------------	---------	---------------	--------

Рис. 2. Схема отношения

Фрагмент такой таблицы с данными представлен в табл. 3. Данный пример показывает, что составление подобных отношений вынужденно приводит к многократному дублированию данных во всех представленных атрибутах. Такой вариант хранения данных удобен тем, что одна таблица содержит все данные. Однако практический опыт показал, что многократное введение одних и тех же данных часто ведут к ошибкам в них. Кроме этого, нерационально используется память для хранения таких данных. Если такая таблица будет содержать значительное количество записей, то работа с ней будет сильно затруднена.

Таблица 3

Пример заполнения данными таблицы

Группа	Фамилия	Имя	Отчество	Дата рождения	Название дисциплины	Форма контроля	Количество часов	Количество ЗЕТ	Семестр	Преподаватель	Оценка
606-01	Иванов	Денис	Владимирович	01.12.2004	Мат.анализ	экзамен	180	5	1	Куликов А.О., Литвак В.	удовл.
606-01	Иванов	Денис	Владимирович	01.12.2004	Лин. ал.	экзамен	108	3	1	Семенова О.Ю.	удовл.
606-01	Иванов	Денис	Владимирович	01.12.2004	Алт. языки	зачет	108	3	1	Смирнов В.А.	зачтено
606-01	Федоров	Матвей	Андреевич	15.04.2004	Мат.анализ	экзамен	180	5	1	Куликов А.О., Литвак В.	хорошо
606-91	Кремень	Данила	Алексеевич	12.06.2003	Мат.анализ	экзамен	180	5	1	Куликов А.О., Литвак В.	хорошо
606-91	Кремень	Данила	Алексеевич	12.06.2003	Мат.анализ	экзамен	180	5	2	Куликов А.О., Литвак В.	отлично
606-91	Кремень	Данила	Алексеевич	12.06.2003	История	зачет	72	2	2	Туманова П.И.	зачтено
606-91	Урядова	Мария	Владимировна	24.08.2003	Основы прогр.	экзамен	108	3	2	Приторов П.А.	удовл.
606-91	Урядова	Мария	Владимировна	24.08.2003	Мат.анализ	экзамен	180	5	2	Куликов А.О., Литвак В.	хорошо
606-91	Урядова	Мария	Владимировна	24.08.2003	История	зачет	72	2	2	Туманова П.И.	зачтено
606-81	Пирогов	Александр	Юрьевич	28.02.2000	Сетевые техноло.	экзамен	108	3	6	Брагина Т.А.	отлично
606-81	Пирогов	Александр	Юрьевич	28.02.2000	Выч. мат.	экзамен	108	3	7	Комков П.И.	хорошо
606-81	Пирогов	Александр	Юрьевич	28.02.2000	Базы данных	экзамен	144	4	4	Викулин А.В.	удовл.
606-81	Любимая	Мария	Владимировна	08.04.2000	Сетевые техноло.	экзамен	108	3	6	Брагина Т.А.	хорошо
606-81	Любимая	Мария	Владимировна	08.04.2000	Выч. мат.	экзамен	108	3	7	Комков П.И.	отлично
606-81	Любимая	Мария	Владимировна	08.04.2000	Базы данных	экзамен	144	4	4	Викулин А.В.	отлично

Для решения проблем, возникающих при описанном построении таблицы, существует метод нормальных форм. В процессе нормализации отношение заменяется системой отношений и взаимосвязей между ними.

Лабораторная работа № 4. Нормализация отношения

В данной работе необходимо провести процедуру приведения отношения к 3-й нормальной форме.

Задание:

1. Выделить первичный ключ.
2. Привести 3 различных примера ключа.
3. Проверить выполнение условия нахождения отношения в 1-й нормальной форме. В случае невыполнения – преобразовать отношение так, чтобы оно выполнялось.
4. Проверить выполнение условия нахождения отношения во 2-й нормальной форме. В случае невыполнения – преобразовать отношение так, чтобы оно выполнялось.
5. Проверить выполнение условия нахождения отношения в 3-й нормальной форме. В случае невыполнения – преобразовать отношение так, чтобы оно выполнялось.
6. Проанализировать исходное отношение и полученное.

Пример. Первым этапом в процедуре нормализации достигается условие атомарности всех атрибутов. В рассмотренном примере поле «Преподаватель» для некоторых дисциплин содержит два элемента (например, один читает лекции, другой проводит практические занятия). В данном случае проблему можно решить простым удалением одной из фамилий, можно оставить только того, кто принимает экзамен. Возможны другие варианты решения проблемы перечислений (обычно это либо создание дополнительных полей, либо записей).

Когда все поля атомарны, отношение находится в первой нормальной форме.

Далее выделяется первичный ключ, находятся зависимости и последовательно выделяются новые отношения. До тех пор, пока не будет выполнено условие третьей нормальной формы.

В результате нормализации отношения, рассмотренного в лабораторной работе № 3 получена совокупность таблиц и связей.

Основное отношение «Ведомость» содержит ассоциативную информацию между студентом, дисциплиной и оценкой (табл. 4).

Таблица 4

Отношение «Ведомость» в 3-й нормальной форме

Идентификатор студента	Дисциплина	Семестр	Оценка
ИваДВ04	Мат. анализ	1	удовл.
ИваДВ04	Лин. ал.	1	удовл.
ИваДВ04	Алг. языки	1	зачтено
ФедМА04	Мат. анализ	1	хорошо
КреДА03	Мат. анализ	1	хорошо
КреДА03	Мат. анализ	2	отлично
КреДА03	История	2	зачтено
УряМВ03	Основы прогр.	2	удовл.
УряМВ03	Мат. анализ	2	хорошо
УряМВ03	История	2	зачтено
ПирАЮ00	Сетевые технологии	6	отлично
ПирАЮ00	Выч. мат.	7	хорошо
ПирАЮ00	Базы данных	4	удовл.
ЛюбМВ00	Сетевые технологии	6	хорошо
ЛюбМВ00	Выч. мат.	7	отлично
ЛюбМВ00	Базы данных	4	отлично

Кроме этого, получено несколько справочных таблиц о составе групп, студентах, дисциплинах.

Тема 1.4

Логическое моделирование

Логическое моделирование учитывает модель представления данных в проектируемой БД. В данной работе предполагается использование реляционной БД. Для некоторых нюансов необходимо учитывать особенности СУБД, поэтому сразу обозначим, что в качестве СУБД будет использована MS Access 2016/2019 или LibreOffice 7.1.5. База Данных, которые имеют лишь незначительные отличия в функциональных возможностях.

Логическая модель описывает состав БД, схему связей между всеми таблицами БД, описание атрибутов, ключей. Может содержать дополнительную информацию о типах данных и вариантах обеспечения целостности связей.

Построение логической модели является очень важным этапом проектирования БД. Именно на этом этапе фактически формируется вся будущая структура БД.

Наиболее важными свойствами, которые необходимо оценить при построении модели БД – это адекватность, полнота, адаптируемость и универсальность.

Под адекватностью понимается соответствие модели реальной предметной области. Для соблюдения адекватности важно не допустить искажений сведений предметной области. На практике часто бывает трудно оценить соответствие из-за представления данных в других формах. Но это свойство имеет ключевое значение и при его нарушении построение БД теряет смысл. Адекватность необходимо контролировать как на уровне структуры, так и при настройке способов ограничения целостности данных.

Полнота БД представляет собой возможность удовлетворения существующих и новых потребностей пользователей. Использование подхода «от предметной области» к проектированию баз данных и сама идеология банков данных как интегрированного взаимоувязанного хранилища данных способствуют обеспечению этого критерия. Хранение в БД детализированных показателей также повышает возможности удовлетворения разнообразных потребностей пользователей. Полнота является одним из проявлений адекватности.

Адаптируемость – приспособление к изменяющимся условиям. В теории БД ЭТО ПРАВИЛЬНО? широко используется понятие независимых программ от данных и данных от программ. Не менее, а даже более значимой, является проблема обеспечения независимости логической модели БД от изменений, происходящих

в предметной области. Обеспечение устойчивости модели БД к изменениям предметной области фактически снимает проблему независимости программ от данных, так как в этом случае структуры данных меняться не будут.

Также в [2] рассмотрены и другие важные свойства.

Лабораторная работа № 5. Логическая модель

В данной работе необходимо построить логическую модель БД. В качестве основы использовать нормализованное отношение из лабораторной работы № 4.

Задание:

1. Изобразить схематически все таблицы модели.
2. Указать ключи во всех таблицах.
3. Указать связи между таблицами.

Пример:

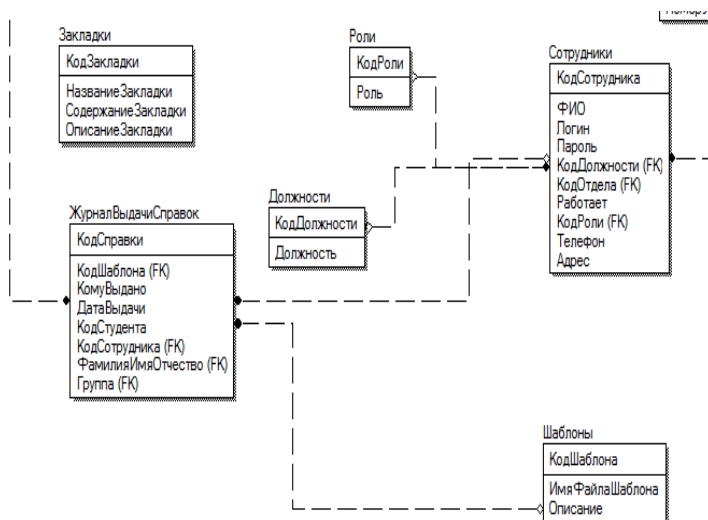


Рис. 3. Фрагмент логической модели БД

На рисунке 3 обозначены предполагаемые отношения будущей реляционной БД, отдельно выделен ключевой атрибут, далее

представлен список всех остальных атрибутов. Связи между сущностями изображены линиями между таблицами. Вид связи «один-ко-многим» изображается белой и черной точками соответственно.

Тема 1.5

Физическая модель

Финальным этапом проектирования БД можно считать построение физической модели. Ключевое значение при ее построении имеет выбранная СУБД. В основе физической модели, как правило, лежит логическая модель. До начала формирования физической модели необходимо изучить возможности и особенности выбранной СУБД. При наличии логической модели, переход от нее к физической производится простым установлением формальных соответствий между этой моделью и имеющимися в СУБД элементами для ее создания. На данном этапе в среде СУБД создаются описанные таблицы – схемы отношений, подбираются типы данных для каждого атрибута, проводится выбор ключей и индексов, создание масок ввода. После этого создаются связи между таблицами и устанавливаются механизмы обеспечения целостности связей.

После формирования физической модели в среде СУБД необходимо провести тестирование выделенных ранее качеств БД. Обычно требуется заполнить каждую таблицу некоторым количеством кортежей, произвести операции изменения, удаления в каждой таблице, проверяя при этом как работает обеспечение целостности связей. Кроме этого, обычно производятся всевозможные попытки введения некорректных данных, особенно в ключевые поля, поля с масками ввода и связанные поля.

Если все тестовые процедуры завершены успешно, можно считать физическое моделирование законченным.

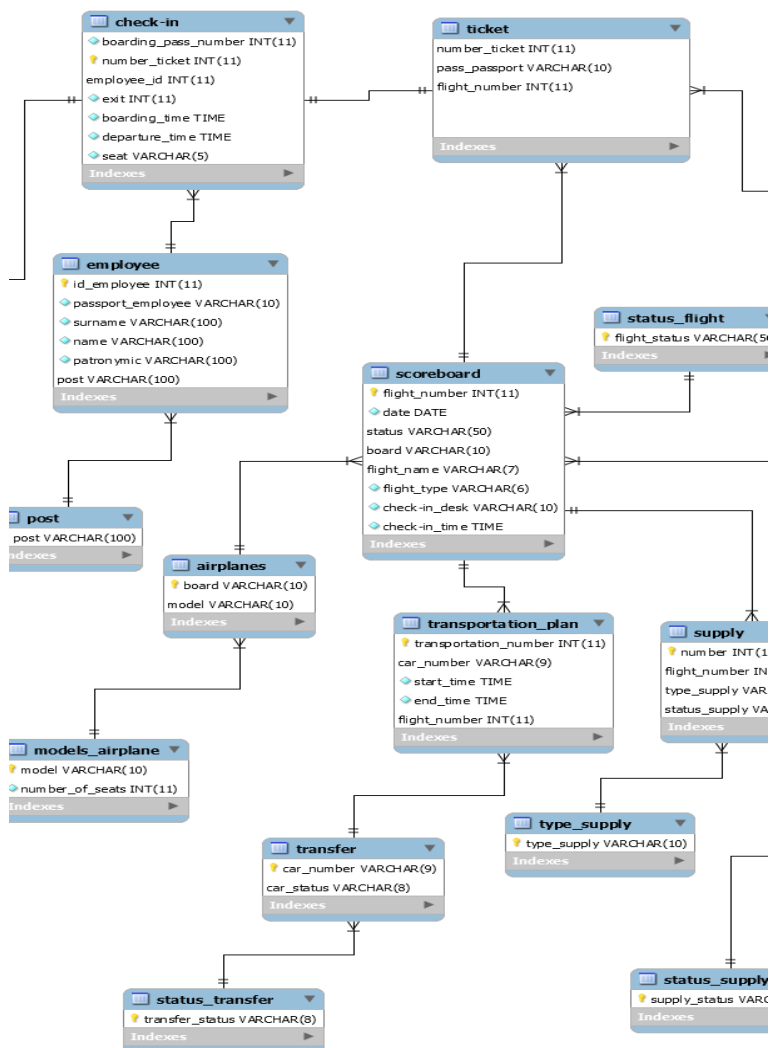


Рис. 4. Фрагмент физической модели БД

На рисунке 4 изображена часть таблиц структуры БД. Название таблицы отражает название сущности, далее перечислены названия атрибутов и указаны их типы данных. Связи отображены линиями,

соединяющими две таблицы. На представленном рисунке можно видеть свойства установленных связей.

Лабораторная работа № 6. Физическая модель

Основная задача проектирования – получить физическую модель БД в выбранной СУБД. После окончания концептуального и логического проектирования можно создавать структуру БД в среде разработки.

Задание:

1. Сформировать в СУБД все таблицы из построенной логической модели. Подобрать наиболее подходящие типы данных, обосновать выбор.
2. Установить ключи во всех таблицах.
3. Установить связи между таблицами.
4. Настроить типы связей и способы обеспечения целостности данных.
5. Отобразить схему данных.

Пример. Созданы таблицы: Заказы, Клиенты, Прейскурант, Сотрудники и состав заказа. Атрибуты, типы данных и ключи таблиц представлены на рис. 5 (а, б, в, г, д).



Заказы	
Имя поля	Тип данных
Код заказа	Счетчик
Клиент	Числовой
Дата приемки заказа	Дата/время
Дата выдачи заказа	Дата/время
Отметка о выдаче	Логический
Сотрудник	Числовой

Рис. 5. Описание таблицы

а) «Заказы»

Клиенты		
	Имя поля	Тип данных
🔑▶	Код клиента	Счетчик
	ФИО клиента	Текстовый
	Адрес	Текстовый
	Телефон	Текстовый

б) «Клиенты»

Прейскурант		
	Имя поля	Тип данных
🔑▶	Код услуги	Счетчик
	Наименование услуги	Текстовый
	Описание услуги	Текстовый
	Стоимость	Денежный

в) «Прейскурант»

Сотрудники		
	Имя поля	Тип данных
🔑▶	Код сотрудника	Счетчик
	ФИО сотрудника	Текстовый
	Адрес	Текстовый
	Телефон	Текстовый

г) «Сотрудники»

Состав заказа		
	Имя поля	Тип данных
🔑▶	Код заказа	Числовой
🔑▶	Код услуги	Числовой
	Количество	Числовой

д) «Состав заказа»

Сетевая модель расширяет иерархическую возможностью произвольных связей. Структура такой БД представляет собой произвольный граф. Такое расширение позволяет наиболее эффективно использовать ресурсы, но при этом сильно усложняет операции с данными.

Операции с данными в описанных структурах существенно отличаются от операций с реляционной БД. Механизмы поддержания целостности данных также существенно отличны от реляционных.

Лабораторная работа № 7. Модели представления данных

В данной работе необходимо освоить понятия иерархической или/и сетевой моделей.

Задание:

1. По выполненному описанию предметной области и ER-модели из предыдущих лабораторных выделить иерархическую (сетевую) структуру.

2. Построить логическую модель иерархической БД для выделенной структуры.

3. Описать корень, листья, связи.

Пример 1. В качестве предметной области рассматривается спортивный комплекс, готовящий спортсменов к соревнованиям. Сотрудниками являются тренеры. Каждый тренер может готовить несколько спортсменов к одному или нескольким соревнованиям.

В качестве корня можно выбрать сущность «тренер». Первая подчинительная связь может быть определена как «тренирует». Подчинительными элементами будут сущности «спортсмен». Следующая связь может быть рассмотрена «готовится к» и связывает сущность «спортсмен» с сущностью «соревнование». При таком построении получится иерархическая структура, представленная на рис. 7.

Листьями в данной структуре окажутся соревнования.

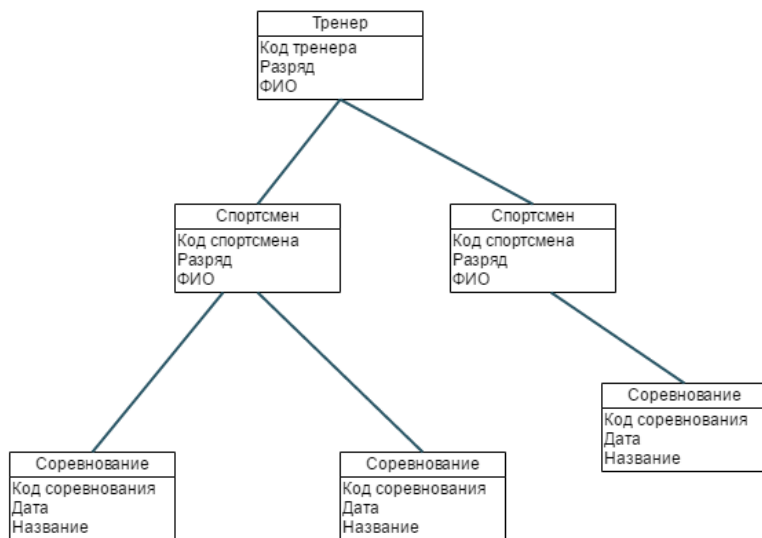


Рис. 7. Пример логической модели иерархической БД

Замечание. Учитывая, что одного и того же спортсмена тренировать могут два или более тренеров, а разные спортсмены могут готовиться к одному и тому же соревнованию, очевидно, такая структура будет содержать избыточные данные. Более экономичной, с точки зрения хранения данных, была бы сетевая модель.

Пример 2. Предметная область представляет собой детали приборов. Детали могут иметь различную форму и выполнены из различных материалов. На рисунке 8 представлена необходимая классификация деталей по форме. Но материал, из которого он изготовлен, может быть одинаков для различных деталей, поэтому классификация по материалу организована с помощью связей с каждой деталью, выполненной из него.



Рис. 8. Пример сетевой модели представления данных.

Сетевая структура чаще всего получается путем расширения иерархической. Из-за сложности в организации процедур обработки данных с произвольными связями, следует по возможности избегать множественных связей между элементами.

Лабораторная работа № 8. Создание таблиц базы данных в предложенной СУБД

До начала выполнения работы необходимо изучить возможности предложенной СУБД.

Задание:

1. По предложенному варианту создать таблицы БД. Подобрать наиболее подходящие типы данных, обосновать выбор.
2. Установить ключи во всех таблицах.
3. Установить связи между таблицами.
4. Настроить типы связей и способы обеспечения целостности данных.
5. Отобразить схему данных.
6. Заполнить все таблицы данными. Минимальное количество – 10 записей в каждой таблице.

7. Изучить возможности СУБД по управлению данными.

Пример. Для предметной области «Магазин настольных игр» выделены такие сущности: Клиент, Заказ, Поставщик, Товар, Свойства товара, Жанр, Серия. Сформированы соответствующие таблицы (рис. 9 а, б, в, г, д, е, ж).

Код клиента	Фамилия Клиента	Имя Клиента	Отчество Клиента	Дата Рождения	Серия паспорт	Номер Паспорта	Контактный номер телефона	Электронный ящик клиента
1	Сатоскич	Кирилл	Витальевич	06.11.2000	3232	756721	+7(922)913-58-75	pekeretyke-9429@mail.ru
3	Данилов	Науменко	Варвар	21.08.1999	2525	223313	+7(922)246-09-73	soddsquinte-2448@mail.ru
4	Чернов	Роман	Денисович	17.07.1999	1234	567343	+7(922)245-70-78	gattytnyle-9878@mail.ru
5	Соболев	Харитон	Здундарович	14.08.1999	3445	123333	+7(922)695-25-70	rffanahunru-9613@mail.ru
6	Морозов	Иван	Андреевич	25.07.1999	3354	668893	+7(922)930-71-54	axupellitt-1250@mail.ru
7	Рабов	Ефим	Викторович	17.01.2000	3456	43456	+7(922)003-97-65	xellehyqqoi-3781@mail.ru
8	Терещенко	Добрый	Викторович	14.02.2000	4344	232345	+7(922)148-22-73	rukaffaffeho-1300@mail.ru
9	Дименев	Стефан	Александрович	07.11.2000	3443	997863	+7(922)891-18-16	isobabime-1451@mail.ru
10	Полномарено	Прохор	Богданович	30.01.2001	6453	563789	+7(922)563-62-70	hesuluxe-7102@mail.ru

а) «Клиент»

Код поставщика	Название компании поставщика	Рабочий адрес	Контактный телефон компании	Сайт поставщика
1	Студия 101	г. Салехард, ул. Русановская Набережная, дом 75	+7(922)983-09-79	https://studio101.ru
2	Компания 2	г. Залесово, ул. Штурманская, дом 31	+7(922)930-84-95	сайт 2
3	Звезда	г. Москва м. Фовинизинская ул. Милашенкова, дом 22	+7(915)040-43-31	https://zvezda.org.ru
4	Smart	г.Москва м.Академическая ул. Дмитрия Ульянова, д. 16	+7 (700)778-46-40	https://hobbygames.ru
5	Games Workshop	г. Лабитанги, ул. Героев-Панфиловцев, дом 15	+7(922)451-49-42	www.games-workshop.com
6	Mosigra	г. Октябрьский, ул. Клязьминская, дом 1	+7(922)583-08-83	www.mosigra.ru

б) «Поставщик»

Свойства товара	Количество игроков	Время игры	ID жанра	ID Серия	Год Выпуска	Вес коробки	Размер коробки	ID Свойства товара
+	3	30	2	4	11.11.2010	2 кг	255x240x70 мм	1
+	8	60+	6	8	15.12.2007	Меньше килограмма	235x157x47 мм	2
+	4	120	4	2	17.06.1999	5 кг	296x296x67 мм	3
+	10	50	3	6	28.09.1994	5 кг	416x355x698mm	4
+	2	30	6	5	23.01.2003	3 кг	146x394x107	6
+	3	60+	2	7	05.12.2001	4 кг	705x612x898	8
+	4	180	3	5	01.10.2010	3 кг	750x831x100	9
+	6	120	6	7	08.04.2011	1 кг	499x835x390	10
+	3	100	5	2	25.11.2013	1 кг	215x420x565	11
+	5	10	5	5	24.07.2018	3 кг	850x215x445	13
+	1	120	6	8	06.02.2015	4 кг	340x780x400	14
+	0		0	0				0

в) «Свойства товара»

Серия	СЕРИЯ ИГР	ID СЕРИИ ИГР
+	Триггерный пробуждение	1
+	Warhammer	2
+	Серпент	3
+	Magic the Gathering	4
+	Dungeons and Dragons	5
+	Pathfinder	6
+	Сантаус	7
+	Мавелон	8
+		0

Товар	Наименование	Цена	Количество на складе	ID Поставщика	Свойство ID
+	1. Мавелон	29	1	1	1
+	2. Magic Game Night	2 189,53 р.	2	2	2
+	3. Warhammer: The Lionade	1 976,49 р.	4	3	10
+	4. Glorionatent, Мрамная Гавань	7 248,89 р.	8	3	4
+	6. Warhammer Fantasy: Treator Company	2 402,34 р.	3	4	14
+	7. Турнир Крестоносцев	738,34 р.	22	6	10
+	8. Крестовые Святыни	282,98 р.	17	2	3
+	9. Мавелон 3. Клонирование особей	300,06 р.	5	1	9
+	10. Мавелон 3. Клонирование особей	573,22 р.	6	6	8
+	11. Партнерство	1 898,73 р.	14	4	6
+	14. Дригеловый Ручей	2 487,02 р.	2	4	11

г) «Серия»

д) «Товар»

Жанр	Жанр	ID жанра
+	Кооперативная игра	1
+	Коллекционная карточная игра	2
+	Семейные настольные игры	3
+	Варгеймы	4
+	Настольная ролевая игра	5
+	Кооперативные	6

Заказы	ID Клиента	ID Товара	Дата заказа	Статус заказа
+	1	1	21.11.2019	Выполнено
+	7	3	11.09.2019	Выполнено
+	9	4	09.10.2019	Заказ отменен
+	5	6	10.12.2019	Заказ формируется
+	8	8	16.10.2019	Выполнено
+	2	9	12.11.2019	Выполнено
+	10	9	18.12.2019	Ожидание оплаты

е) «Жанр»

ж) «Заказ»

Рис. 9. Таблицы, заполненные данными

На рисунке 10 продемонстрированы связи между таблицами и отражены ключевые поля в таблицах.

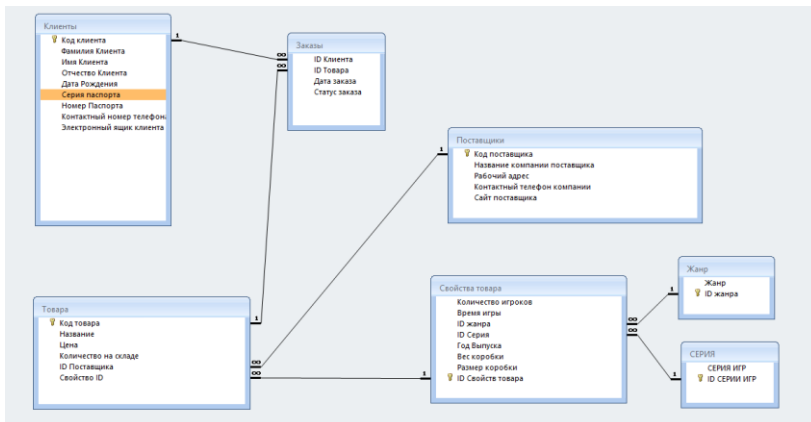


Рис. 10. Схема данных

Для обеспечения целостности данных установлены каскадное обновление и удаление данных.

Лабораторная работа № 9. Формы для отображения таблиц

Для удобства пользователя в современных СУБД информация по запросам не предоставляется в виде таблиц, в которых организовано хранение, а создаются специальные формы для отображения необходимой информации в удобном виде.

Для создания формы в режиме конструктора необходимо изучить инструментарий СУБД.

Форма содержит три области: заголовочную, область примечаний и область с отображением данных. Для представления данных имеется набор элементов: поле, кнопка, переключатель, надпись и т. д. Каждый из объектов имеет свои свойства для определения местоположения, размеров, режима работы и т. д. Для тестирования работы созданной формы имеется возможность переключения режима конструктора и просмотра формы. Кроме прочих элементов на форме может располагаться подчиненная форма.

Задание:

1. Создать формы для отображения данных каждой таблицы. Реализовать создание различными способами: мастер форм, конструктор форм.

2. Создать главную кнопочную форму, настроить кнопки перехода между формами.

Пример. Для отображения данных, требуемых пользователю, создаются формы, на которых данные располагаются в удобном для восприятия виде, сопровождаются полными, информативными заголовками, панелью навигации по записям, кнопками управления, дополнительными элементами (это могут быть рисунки, ссылки, кнопки печати и т. д.) На рисунке 11 представлен пример простой формы, созданной с помощью мастера форм и отражающей данные одной из таблиц БД.

Код сотрудника	ФИО сотрудника	Адрес	Телефон
1	Лазарев Степан Александрович	ул. Гурского 6-7	8(965)405-64-65
2	Карпов Егор Михайлович	ул. Сухаревская 1-7	8(946)546-54-65
3	Иванов Алексей Алексеевич	ул. Тимошенко 5-5	8(984)065-46-54
*	(№)		

Рис. 11. Пример формы для отображения данных из таблицы

Для навигации по элементам БД, для информирования пользователя об имеющихся элементах БД создаются кнопочные формы (меню), которые не содержат данные, на них расположены кнопки и закладки, которые осуществляют либо переход на новые кнопочные формы (рис. 12), либо на формы с данными, запросами, либо другие дополнительные элементы.



Рис. 12. Пример главной кнопочной формы (меню)

Кнопочных форм можно создать любое количество. С помощью них формируется меню для каждой группы пользователей, формируются различные группы элементов по типу выполняемых функций и т. п.

Лабораторная работа № 10. Маски ввода

Для удобства ввода данных и дополнительного контроля корректности данных можно задать формат ввода данных в некоторых полях. Для организации формата используются маски ввода данных. Наиболее распространенными полями для использования масок являются номера телефонов, номер, серия паспорта, всевозможные идентификаторы и т. п., т. е. для тех данных, которые содержат установленное количество символов, установленные разделители, ограничение на ввод некоторых символов и т. п.

Таблица 4

Основные символы, позволяющие сформировать маску

Символ маски	Описание действий пользователя
0	Необходимо ввести любую цифру

9	Можно ввести любую цифру
#	Можно ввести пробел, знак «+» или «-» (по умолчанию – пробел)
L	Необходимо ввести букву
?	Можно ввести букву
A	Необходимо ввести букву или цифру
a	Можно ввести букву или цифру
&	Необходимо ввести любой знак или пробел
C	Можно ввести любой знак или пробел
.,:;- /	Разделители
>	Все последующие знаки будут переведены в верхний регистр
<	Все последующие знаки будут переведены в нижний регистр
!	Маска ввода заполняется слева направо, а не справа налево
\	Маска ввода заполняется слева направо, а не справа налево
“”	Знаки, заключенные в двойные кавычки, отображаются без изменений

Задание:

1. Изучить правила создания масок ввода в СУБД.
2. Выбрать 2 поля, содержащих данные фиксированного формата.
3. Создать маски ввода для выбранных полей.
4. Протестировать ввод данных по созданным маскам.

Пример 1.Стандартные номера, которые имеют установленное количество символов, общую группу символов, можно вводить по маске:

IDFN 0-#####-0

Здесь первые четыре символа установлены, далее один символ в начале и один в конце должны быть цифрами, между ними присутствует группа цифро-буквенных символов, разделители установлены дефисами.

Пример 2. Часто предприятия устанавливает некоторый формат своих идентификаторов. Для удобства введения таких значений можно сформировать маску.

>LL00000-0000

Данная маска подразумевает обязательный ввод 2 букв и 9 цифр с разделителем в установленном месте в верхнем регистре.

Лабораторная работа № 11. Создание запросов в режиме мастера и конструктора

Основным инструментом работы с БД для пользователя являются запросы, которые позволяют выполнять основные манипуляции с данными.

Для создания запроса в режиме мастера необходимо выполнить следующие действия:

1. При открытой БД перейти в меню «Создать».
2. Нажать кнопку «Мастер запросов».
3. Выбрать тип запроса: простой, перекрестный, повторяющиеся значения в поле или записи без подчинения.
4. Выбрать источники данных (таблицы, запросы, поля).
5. Ввести название для запроса.
6. При необходимости макет можно изменить в ручном режиме.
7. Нажать «Готово».

Для создания запроса в режиме конструктора необходимо выполнить действия:

1. При открытой БД перейти в меню «Создать».
2. Нажать кнопку «Конструктор запросов».
3. Выбрать необходимые поля из таблиц или запросов.
4. На панели инструментов выбрать тип запроса. Установить настройки.
5. В окне конструктора построить запрос: установить настройки полей, сортировки, условия.
6. Проанализировать содержимое в различных режимах отображения запроса.
7. Проверить выполнение запроса.

8. Сохранить запрос, присвоив ему подходящее имя.

Задание:

1. Создать запрос на добавление записи в таблицу.
2. Создать перекрестный запрос.
3. Создать параметрический запрос.
4. Сформулировать 3 наиболее востребованных запроса на выборку данных к созданной БД и создать их в СУБД.
5. Создать запрос на обновление.
6. Создать запрос на удаление.

Пример:

Перекрестный запрос. Имеется таблица, содержащая сведения о студентах учебного заведения. Создается перекрестный запрос, который выберет из базы данных всех студентов, которые начали обучение в указанном году. Для каждого студента будут выведены ФИО, группа, в которой студент обучается, его пол.

Выбираются поля для заголовков строк – «ФИО», «группа» и «пол» (рис. 13).

Создание перекрестного запроса

Выберите поля, значения которых будут использованы в качестве заголовков строк.

Допускается выбор не более трех полей.

Выберите поля по порядку сортировки данных. Например, можно сначала выполнить сортировку значений по странам, а затем по городам.

Доступные поля:

- КодСтудента
- УсловияОбучения
- Статус
- ГодПоступления
- НомерЗачетнойКнижки
- ФормыОбучения
- НомерДокумента
- КемВыдан
- ДатаВыдачи
- ДатаРождения
- ГодРождения

Выбранные поля:

- ФамилияИмяОтчество
- Группа
- Пол

Образец:

ФамилияИмя	Группа	Пол	КодСтудент	КодСтудент	КодСтудент
ФамилияИмяОт	Группа1	Пол1	Min(КодСтудента)		
ФамилияИмяОт	Группа2	Пол2			
ФамилияИмяОт	Группа3	Пол3			
ФамилияИмяОт	Группа4	Пол4			

Рис. 13. Выбор строк

Для столбцов выбрано поле «Годрождения», как показано на рис. 14.

Создание перекрестного запроса

Выберите поле, значения которого нужно использовать в качестве заголовков столбцов.

Например, чтобы использовать имя каждого сотрудника в качестве заголовка столбца, выберите поле ИмяСотрудника.

КодСтудента
УсловияОбучения
Статус
Годпоступления
номерзачеткички
ФормыОбучения
НомерДокумента
Кемвыдан
ДатаВыдачи
Датарождения
Годрождения
Датазачисления

Образец:

ФамилияИмя	Группа	Пол	Годрождения	Годрождения	Годрождения
ФамилияИмяОт	Группа1	Пол1	Min(КодСтудента)		
ФамилияИмяОт	Группа2	Пол2			
ФамилияИмяОт	Группа3	Пол3			
ФамилияИмяОт	Группа4	Пол4			

Отмена < Назад Далее > Готово

Рис. 14. Выбор столбца

В ячейках на пересечении указываются средние значения поля «Годпоступления» (рис. 15).

Создание перекрестного запроса

Какие вычисления нужно выполнить для каждой ячейки на пересечении строк и столбцов?

Например, можно вычислить сумму заказов для каждого сотрудника (столбец) по странам и регионам (строка).

Вычислить итоговое значение для каждой строки?

☒ Да.

Поля:
КодСтудента
УсловияОбучения
Статус
Годпоступления
номерзачеткички
ФормыОбучения
НомерДокумента
Кемвыдан
ДатаВыдачи
Датарождения
Датазачисления

Функции:
Min
StDev
Var
Максимум
Первый
Последний
Среднее
Сумма
Число

Образец:

ФамилияИмя	Группа	Пол	Годрождения	Годрождения	Годрождения
ФамилияИмяОт	Группа1	Пол1	Среднее(Годпоступления)		
ФамилияИмяОт	Группа2	Пол2			
ФамилияИмяОт	Группа3	Пол3			
ФамилияИмяОт	Группа4	Пол4			

Рис. 15. Значения в ячейках

Результаты выполнения данного запроса представлены на рис.

16.

Алиев Вели Видди Оглы	402-72	Муж						
Алиев Вусал Абульфат Оглы	609-71	Муж						
Алиев Камил Исмет Оглы	403-71м	Муж						
Алиев Ровшан Намаз оглы	604-82м	Муж						
Алиев Фаик Хикмет оглы	102-73	Муж						
Алиева Асмар Муталлим Кызы	203-71	Жен						
Алиева Виктория Раджевна	502-81	Жен						
Алиева Гунель Муталлим Кызы	208-61	Жен	2016					2016
Алиева Диана Вагифовна	607-61	Жен	2016				2016	
Алиева Зейнаб Равиевна	501-82	Жен						
Алиева Манижа Джамшедовна	208-82	Жен						
Алиева Нозанин Рустамовна	501-63	Жен	2016					2016
Алиева Сабина Темразовна	501-54	Жен	2015				2015	
Алиева Фаина Джаферовна	501-34	Жен	2013		2013			
Аликанов Фарходжон Махаматович	501-75	Муж						
Аликова Анастасия Ивановна	404-81	Жен						
Алимарданова Илаха Элдар кызы	209-81м	Жен						
Алимова Александрина Леонидовна	211-71	Жен						
Алимова Регина Рафидиновна	501-52	Жен	2015					2015
Алимуратов Тельман Темиршахович	501-65	Муж	2014			2014		

Рис. 16. Фрагмент результатов выполнения запроса

Параметрический запрос. Такой запрос подразумевает ввод выражения в условие отбора в процессе выполнения запроса в диалоге с пользователем, не переходя в режим конструктора.

Сначала создается необходимый запрос. Далее его необходимо открыть в режиме конструктора.

Из таблицы со сведениями о студентах выбираются столбцы с ФИО, полом, группой и датой рождения, эти данные необходимы для тех студентов, которые в указанном году заканчивают обучение. В качестве параметра следует указать год окончания. В режиме конструктора добавляется поле с годом окончания, указывается для него необходимость ввода условия, не ставится отметка выводимости его на экран (в этом нет необходимости, поскольку значения все будут одинаковыми и равными введенному значению). Остальные выбранные поля выводятся (рис. 17).

Поле:	ФамилияИмяОтч	Пол	Группа	Датараждения	Годокончания
Имя таблицы:	Студенты	Студенты	Студенты	Студенты	Студенты
Сортировка:	по возрастанию				
Вывод на экран:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Условие отбора:					
или:					[Введите год оконча

Рис. 17. Конструктор параметрического запроса, «Годокончания» – поле, для выбора по вводимому параметру

После нажатия выполнения запроса пользователю будет представлено окно для ввода значения параметра (рис. 18).

Введите значение параметра ? X

Введите год окончания:

2021

ОК Отмена

Рис. 18. Окно ввода параметра

Результат выполнения данного запроса в табличном виде представлен на рис. 19.

ФамилияИмяОтчество ▾	Пол ▾	Группа ▾	Датарожде ▾
Абазов Альберт Радикович	Муж	305-71	17.02.2014
Аббасова Конул Бахлул Кызы	Жен	208-72	24.01.2014
Абдрахимова Карина Салаватовна	Жен	501-64	09.11.2011
Абдулвалиев Владимир Витальевич	Муж	603-72	26.08.2013
Абдусеменова Альфия Медагаевна	Жен	405-71	11.04.2013
Абидинова Аминат Абдулмеджидовна	Жен	502-52	28.11.2013
Абишева Наталия Сергеевна	Жен	608-71	21.05.2014
Абубакирова Каролина Данияровна	Жен	305-71	22.11.2013
Агадуллина Альбина Ильдусовна	Жен	203-51	10.08.2011
Анков Иван Евгеньевич	Муж	102-71	21.01.2013
Ахмедова Айшан Мобил Кызы	Жен	102-71	22.04.2013
Волошин Дмитрий Александрович	Муж	101-71м	27.01.2016
Гнедаш Елена Дмитриевна	Жен	102-71	30.05.2013
Дубинина Екатерина Вячеславовна	Жен	102-71м	24.08.2015

Рис. 19. Результаты выполнения параметрического запроса – необходимые сведения о студентах, заканчивающих обучение в 2021 году

Таким же образом можно установить несколько параметров, вводимых пользователем.

Лабораторная работа № 12. SQL (structured query language)

В данной работе необходимо освоить язык SQL. Для этого формируется набор запросов.

Задание:

1. С помощью языка SQL написать запросы к БД, используя операторы и ключевые слова:

- a. SELECT
- b. DISTINCT
- c. ORDER BY
- d. DESC
- e. JOIN
- f. WHERE
- g. LIKE
- h. GROUP BY
- i. HAVING
- j. INSERT INTO
- k. DELETE
- l. UPDATE

2. Требования:

- a. Минимальное количество различных созданных запросов – 5.
- b. Работоспособность запросов.
- c. Корректность выполнения запросов.
- d. Нетривиальные результаты работы запросов.

Пример. Запрос, использующий SELECT, DISTINCT, ORDER BY, WHERE:

```
SELECT DISTINCT Студенты.[ФамилияИмяОтчество],  
Студенты.[Пол], Студенты.[Группа], Студенты.[Датарождения]  
FROM Студенты  
WHERE (((Студенты.Годокончания)=[Введите год  
окончания:])))  
ORDER BY Студенты.[ФамилияИмяОтчество] DESC;
```

Один запрос может включать любое количество операторов и ключевых слов. Одни и те же операторы и ключевые слова можно использовать повторно.

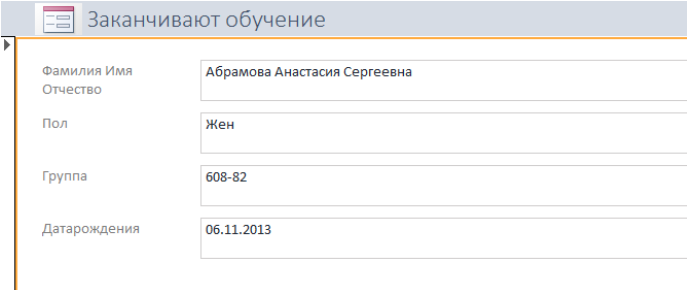
Лабораторная работа № 13. Создание форм запросов

Запросы – это инструмент для взаимодействия пользователя с БД, поэтому результаты выполнения запросов также, как и сами первичные таблицы, представляются с помощью форм, на которых

понятным и удобным способом отображаются не только данные, но и дополнительная необходимая информация.

Задание. Создать формы для каждого имеющегося запроса. Использовать мастер и конструктор форм. Обязательно сопровождение данных поясняющими надписями, рисунками, ссылками и т. д.

Пример. Форма, созданная с помощью мастера форм (автоформа), отражает все поля запроса (рис. 20), выводимые на экран, в качестве заголовков полей используются заголовки полей из схемы отношения. Поскольку они иногда носят сокращенный, иностранный или условный вид, рекомендуется скорректировать их, открыв форму в режиме конструктора (рис. 21).



The screenshot shows an Access autoform titled "Заканчивают обучение". The form contains the following fields and values:

Field Label	Value
Фамилия Имя Отчество	Абрамова Анастасия Сергеевна
Пол	Жен
Группа	608-82
Датарождения	06.11.2013

Рис. 20. Пример автоформы для запроса



The screenshot shows an Access autoform titled "В выбранном году заканчивают обучение следующие студенты". The form contains the following fields and values:

Фамилия Имя Отчество студента			
Аскерова Юлия Ильхамовна			
Пол	Группа	Дата рождения	
Жен	101-81	06.06.2014	

Рис. 21. Пример формы, созданной в режиме конструктора

Формы позволяют представить необходимую информацию именно в таком виде, как будет удобно пользователю. Имеется огромное количество инструментов настройки, что позволяет добиться оптимального варианта формы.

Лабораторная работа № 14. Отчеты

Отчеты служат для предоставления информации в удобной для печати и изучения форме. Чаще всего это инструмент для представления агрегированной информации.

Задание:

1. Создать отчет для представления суммарной информации по данным, например, за некоторый период. Оформить заголовочную и резолюционную области.

2. Создать отчет, отражающий некоторые средние показатели по данным. Отчет должен содержать данные из нескольких таблиц. Оформить заголовочную и резолюционную области.

Пример:

Прейскурант			
 Прейскурант		24 октября 2018 г. 11:38:43	
Код услуги	Наименование фотоуслуги	Описание фотоуслуги	Стоимость
1	Фото на документы	Комплект фотографий выбранного формата	155,00р.
2	Дополнительный комплект	Комплект фотографий любого формата	80,00р.
3	Запись на флэш-носитель	Или отправка на e-mail клиента	50,00р.
4	Замена одежды	Форма, пиджак	90,00р.
5	Фотопечать 10-15	Глянцевая и матовая бумага	15,00р.
6	Фотопечать 13-18	Глянцевая и матовая бумага	18,00р.
7	Фотопечать 15-20	Глянцевая и матовая бумага	22,00р.
9	Копирование	Размерность до А3	5,00р.
10	Сканирование	Размерность до А3	10,00р.
11	Оцифровка	Негативы и видеокассеты	200,00р.
23	Цветная печать	Размерность до А3	50,00р.

Страница 1 из 1

Рис. 22. Пример оформления отчета

Такой отчет (рис. 22) является демонстрацией предоставляемых услуг – преysкурантом фотоателье.

Лабораторная работа № 15. Элементы навигации, макросы и модули, разграничение доступа, сжатие БД

Для удобства работы с БД пользователю предоставляется набор элементов управления базой. Для переключения между элементами БД создаются кнопки перехода, создаются кнопки печати формы и т. п.

Поскольку различные данные могут быть доступны или не доступны различным пользователям, доступны в разном режиме, необходимо настроить разграничение доступа.

Сжатие БД производится с целью сокращения используемого дискового пространства.

Задание:

1. Создать элементы навигации, управления БД.
2. Использовать 3–5 различных макросов.
3. Создать модуль VBA.
4. Создать 3–4 пользователя с различными правами доступа.
5. Выполнить сжатие и восстановление БД, оценить размер.
6. Настроить запуск главной кнопочной формы при запуске.

Примеры. На каждой форме кроме данных имеются элементы управления. Организовать выполнение функций с кнопки можно с помощью макросов.

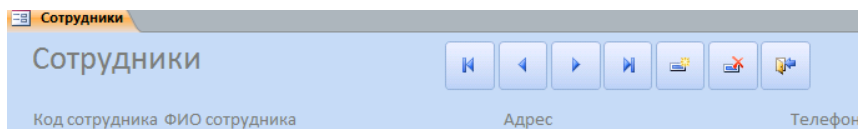


Рис. 21. Пример панели кнопок на форме, для управления данными

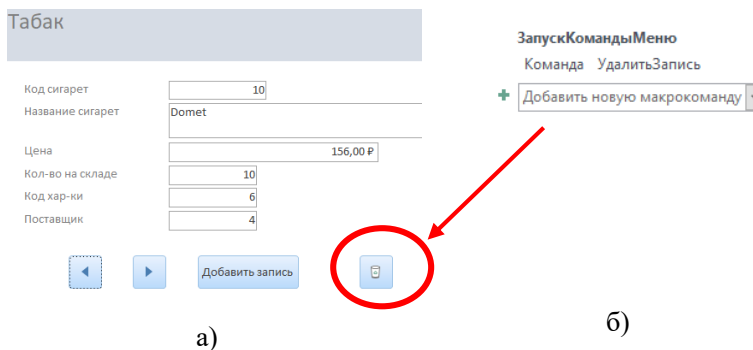


Рис. 22. Пример использования макроса: а) – кнопка удаления на форме; б) – макрос

Очень часто пользователи БД очень многочисленны и используют данные для различных операций. Если при этом пользователям доступны не все данные, а только часть, удобно использовать группы пользователей и разграничение доступа к данным. При этом разработанное меню для различных групп также должно включать различные наборы кнопок, выполняющих различные функции. Некоторые пункты меню могут для них быть одинаковыми. Возникает необходимость создания нескольких наборов форм-меню, предоставляющих возможности пользователям каждой группы (рис. 23).

Если возможностей СУБД оказывается не достаточно для разработки какой-то функции, имеется среда разработки, использующая встроенные язык программирования (VBA – visual basic for application), с помощью которой можно реализовать функции в виде модулей (рис. 24).



Рис. 23. Пример разграничения доступа по пользователям

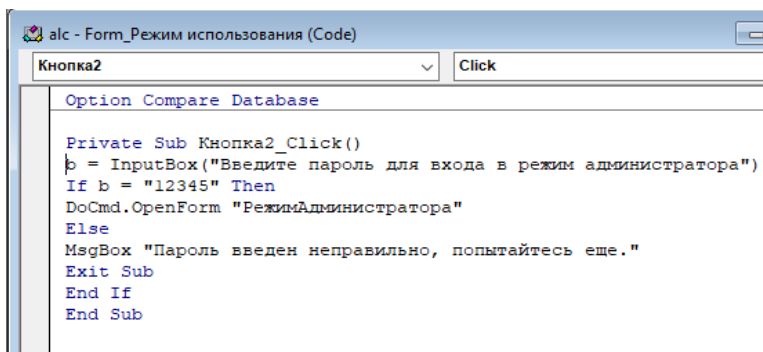
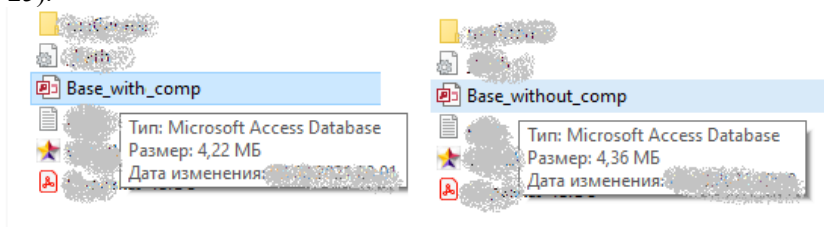


Рис. 24. Пример модуля

Представленный модуль позволяет обработать вводимый пароль от одной из групп пользователей – администраторов. Осуществляет сверку значения пароля с сохраненным.

Хранение БД на носителях информации ставит вопрос о занимаемом объеме памяти. При увеличении количества записей, он может существенно расти со временем. Можно сократить объем памяти, необходимый для ее расположения с помощью сжатия БД (рис. 25).



а) БД со сжатием

б) БД без сжатия

Рис. 25. Пример сжатия БД

На рисунке 25 представлены размеры БД со сжатием и без сжатия. Разница в размере файла относительно небольшая, что вызвано маленькой мощностью сжимаемой БД. С ростом количества записей результаты сжатия становятся более существенными.

Раздел 2

ТЕХНОЛОГИЯ УПРАВЛЕНИЯ БАЗОЙ ДАННЫХ ИЗ СРЕДЫ ПРОГРАММИРОВАНИЯ

Тема 2.1. Объектно-реляционное отображение

ORM (англ. Object-Relational Mapping, рус. объектно-реляционное отображение) – это технология, которая создает связь между базой данных и ее объектным представлением, выраженным при помощи концепции объектно-ориентированного программирования.

Достоинства технологии:

1. Использование провайдеров для работы с СУБД.
2. Возможность не использовать язык запросов SQL для работы с СУБД.
3. Автоматическое преобразование типов СУБД в типы объектно-ориентированного языка программирования и наоборот.

Недостатки технологии:

1. Обязательное знание концепции объектно-ориентированного программирования.
2. Высокая сложность проектирования объектного представления базы данных.

Технология ORM имеет программные реализации на C++, C#, Java, Python и других языках, реализующих концепцию объектно-ориентированного программирования.

Лабораторная работа № 16. Object-relational mapping

Первичное знакомство с технологией ORM удобно начинать с платформы .NET 5, ORM-инструмента EF Core (Entity Framework Core) и языка программирования C# [6].

Задание. Реализовать взаимодействие базы данных и консольного приложения средствами инструмента EF Core и языка программирования C#. Выполнение задания состоит из следующих этапов:

1. Создать на основе логической модели базу данных в СУБД SQLite.

2. Создать проект консольного приложения на языке программирования C# в среде программирования Visual Studio 2019.

3. Установить пакеты Microsoft.EntityFrameworkCore и Microsoft.EntityFrameworkCore.Sqlite для работы с EF Core, используя пакетный менеджер «Nuget», встроенный в Visual Studio 2019.

4. Создать на основе логической модели и языка программирования C# объектное представление базы данных, именуемое далее «Контекстом данных».

5. Реализовать запросы на добавление, редактирование, удаление, поиск и чтение из базы данных в консольном приложении, используя контекст данных.

Пример. Создадим базу данных на основе логической модели в СУБД SQLite. Предметной областью, рассматриваемой в примере, будет интернет-магазин. На рисунке 26 представлена логическая модель базы данных интернет-магазина. Важно помнить, что имена атрибутов и таблиц в базе данных должны полностью совпадать с именами свойств в классах и контексте данных, включая регистр и не алфавитные знаки.

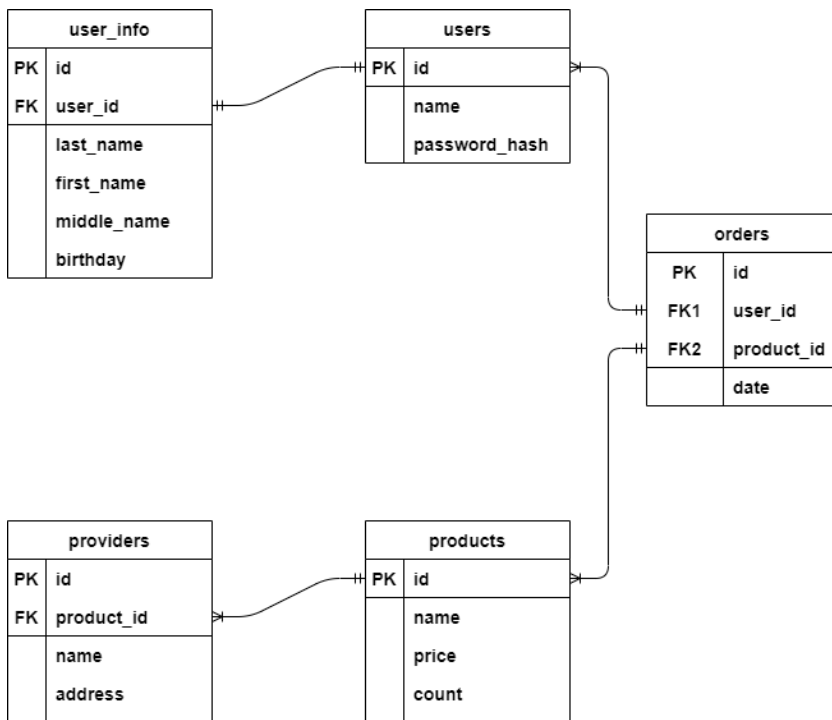


Рис. 26. Логическая модель БД интернет-магазина

После работы с СУБД SQLite создадим проект консольного приложения в Visual Studio 2019 и установим в него требуемые для работы с базами данных пакеты. Существуют два способа создания проекта консольного приложения:

1. Нажать кнопку «Создание проекта» в стартовом меню → Выбрать тип проекта «Консольное приложение (.NET Core)» → Задать имя проекта и выбрать каталог для сохранения проекта → Выбрать в качестве текущей платформы .NET 5 → Нажать кнопку «Создать».

2. Выбрать пункт «Файл» в главном меню → Выбрать пункт «Создать» → Выбрать пункт «Проект» → Выбрать тип проекта «Консольное приложение (.NET Core)» → Задать имя проекта и выбрать каталог для сохранения проекта → Выбрать в качестве текущей платформы .NET 5 → Нажать кнопку «Создать».

В качестве имени проекта напомним «InternetShop» и сохраним его на рабочий стол. После создания проекта консольного приложения приступим к созданию контекста данных, поэтому в первую очередь установим через пакетный менеджер «Nuget» необходимые пакеты. Чтобы перейти в пакетный менеджер и выполнить установку требуется:

3. Выбрать пункт «Проект» → Выбрать пункт «Управление пакетами Nuget» → В открывшемся окне выбрать вкладку «Обзор» → В поисковой строке набрать название пакета → Выбрать нужный пакет из списка ниже и нажать установить.

Прделаем данную операцию пакетов Microsoft.EntityFrameworkCore и Microsoft.EntityFrameworkCore.Sqlite. После чего приступим к созданию контекста данных. Ниже в листингах 1–7 представлен исходный код объектного представления базы данных и код главной функции консольного приложения, в котором реализуются операции добавления, редактирования, удаления, поиска и чтения данных из базы.

```
using System.Collections.Generic;
namespace InternetShop
{
    #region Тип описывающий кортежи и атрибуты таблицы в
БД

    public class User
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string PasswordHash { get; set; }
        public UserProfile UserProfile { get; set; }
        public ICollection<Order> Orders { get; set; } = new
List<Order>();
        public ICollection<Product> Products { get; set; } = new
List<Product>();
    }
    #endregion
}
```

Листинг 1. Структура класса User

```

using System;
namespace InternetShop
{
    #region Тип описывающий кортежи и атрибуты таблицы в
    БД
    public class UserProfile
    {
        public int Id { get; set; }
        public string LastName { get; set; }
        public string FirstName { get; set; }
        public string MiddleName { get; set; }
        public DateTime Birthday { get; set; }
        public int UserId { get; set; }
        public User User { get; set; }
    }
    #endregion
}

```

Листинг 2. Структура класса UserProfile

```

using System.Collections.Generic;
namespace InternetShop
{
    #region Тип описывающий кортежи и атрибуты таблицы
    в БД
    public class Product
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public int Price { get; set; }
        public int Count { get; set; }
        public int ProviderId { get; set; }
        public Provider Provider { get; set; }
        public ICollection<Order> Orders { get; set; } = new
        List<Order>();
        public ICollection<User> Users { get; set; } = new
        List<User>();}
    #endregion
}

```

Листинг 3. Структура класса Product

```

using System.Collections.Generic;
namespace InternetShop
{
    #region Тип описывающий кортежи и атрибуты таблицы в
БД
    public class Provider
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Address { get; set; }

        public ICollection<Product> Products { get; set; } = new
List<Product>();
    }
    #endregion
}

```

Листинг 4. Структура класса Provider

```

using System;
namespace InternetShop
{
    #region Тип описывающий кортежи и атрибуты таблицы в
БД
    public class Order
    {
        public int Id { get; set; }

        public int UserId { get; set; }
        public User User { get; set; }

        public int ProductId { get; set; }
        public Product Product { get; set; }
        public DateTime Date { get; set; }
    }
    #endregion
}

```

Листинг 5. Структура класса Order

Листинги 1–5 описывают кортежи и атрибуты таблиц БД, представленных на рис. 26 логической модели. Описание кортежей и атрибутов производится с помощью создания классов (кортежи), содержащих свойства (атрибуты), которые соответствуют именам и типам атрибутов из базы данных. Однако типы атрибутов базы данных выражаются при помощи стандартных типов языка C# [6].

Особого внимания на фоне других заслуживают следующие свойства (атрибуты):

1. С типом другого класса (кортежем), например, User в UserProfile.

2. С универсальным типом ICollection<T>, где T – любой класс (кортеж), например, Orders в User.

Такие свойства называются навигационными и служат для создания связей (один к одному, один ко многим, многие ко многим) между таблицами в контексте данных. Без навигационных свойств любая работа со связными таблицами невозможна. Подробнее об их использовании будет рассказано далее.

```
using Microsoft.EntityFrameworkCore;
namespace InternetShop
{
    #region Контекст данных
    public sealed class InternetShopContext : DbContext
    {
        #region Таблицы в БД
        public DbSet<User> Users { get; set; }
        public DbSet<UserProfile> UserProfiles { get; set; }
        public DbSet<Product> Products { get; set; }
        public DbSet<Provider> Providers { get; set; }
        #endregion
        protected override void
        OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            #region Подключение провайдера базы данных по
            строке подключения
            optionsBuilder.UseSqlite("Data
            Source=InternetShop.db");
            #endregion
        }
    }
}
```

```

        protected override void OnModelCreating(ModelBuilder
modelBuilder)
        {
            #region Задание первичного ключа

            modelBuilder.Entity<User>()
                .HasKey(user => user.Id);

            modelBuilder.Entity<UserProfile>()
                .HasKey(profile => profile.Id);

            modelBuilder.Entity<Product>()
                .HasKey(product => product.Id);

            modelBuilder.Entity<Provider>()
                .HasKey(provider => provider.Id);

            #endregion

            #region Настройка связей между таблицами

            modelBuilder.Entity<User>()
                .HasOne(user => user.UserProfile)
                .WithOne(profile => profile.User)
                .HasForeignKey<UserProfile>(profile =>
profile.UserId);

            modelBuilder.Entity<Product>()
                .HasOne(product => product.Provider)
                .WithMany(provider => provider.Products)
                .HasForeignKey(product => product.ProviderId);

            modelBuilder.Entity<Product>()
                .HasMany(product => product.Users)
                .WithMany(user => user.Products)
                .UsingEntity<Order>(right =>
                {
                    return right.HasOne(order => order.User)

```

```

        .WithMany(user => user.Orders)
        .HasForeignKey(order => order.UserId);
    }, left =>
    {
        return left.HasOne(order => order.Product)
            .WithMany(product => product.Orders)
            .HasForeignKey(order => order.ProductId);
    }, join =>
    {
        join.HasKey(order => order.Id);
    });

```

#endregion

#region Настройка ограничений целостности для
атрибутов

```

modelBuilder.Entity<User>()
    .Property(user => user.Name)
    .IsRequired();

```

```

modelBuilder.Entity<User>()
    .Property(user => user.PasswordHash)
    .IsRequired();

```

```

modelBuilder.Entity<UserProfile>()
    .Property(profile => profile.FirstName)
    .IsRequired();

```

```

modelBuilder.Entity<UserProfile>()
    .Property(profile => profile.LastName)
    .IsRequired();

```

```

modelBuilder.Entity<UserProfile>()
    .Property(profile => profile.MiddleName)
    .IsRequired();

```

```

modelBuilder.Entity<Product>()
    .Property(product => product.Name)

```

```

        .IsRequired();

        modelBuilder.Entity<Provider>()
            .Property(provider => provider.Name)
            .IsRequired();

        modelBuilder.Entity<Provider>()
            .Property(provider => provider.Address)
            .IsRequired();

        #endregion
    }
}
#endregion
}

```

Листинг 6. Структура класса InternetShopContext

В листинге 6 приведен исходный код контекста данных. Для создания контекста данных также применяется класс, но в отличие от кортежей он содержит не только свойства, но методы конфигурации. Начнем разбор класса, реализующего контекст данных согласно выделенным блокам в исходном коде:

1. Блок «Таблицы БД» предназначен для описания таблиц, находящихся в базе данных при помощи свойств типа `DbSet<T>`, где `T` – любой класс (кортеж). Имя свойства обязательно должно соответствовать имени таблицы в базе данных. Однако стоит также запомнить, что промежуточные таблицы, применяемые для разрешения связей многие ко многим, не добавляются в данный блок. Например, таблица `Orders` с типом кортежа `Order`, который представлен в листинге 5 [6].

2. Блок «Подключение провайдера базы данных по строке подключения» предназначен для подключения провайдера и задания строки подключения к базе данных. Под провайдером базы данных понимается конкретная СУБД, которая используется для работы с базой данных. В примере таковой является `SQLite`. Строкой подключения к базе данных является строка определенного формата, содержащая информацию о способе подключения к базе данных и настройках данного подключения. Строка подключения имеет уникальный формат для всех существующих СУБД. Чтобы контекст

данных мог подключиться к базе данных, требуется переопределить метод «OnConfiguring» и вызвать функцию вида «UseИмяПровайдера», в которую будет передана строка подключения. В примерах вызывается функция «UseSqlite», так как в проекте используется СУБД SQLite [6].

3. Блок «Задание первичного ключа» предназначен для задания первичного ключа для всех таблиц, имеющихся в БД. Данные действия необходимы для того, чтобы контекст данных был способен осуществлять операции поиска, добавления, обновления и удаления данных из таблиц базы. Чтобы настроить первичный ключ для таблицы, требуется сначала вызвать метод «Entity<T>», где Т – любой класс (кортеж). Далее требуется вызвать метод «HasKey», в нем выбрать свойство или свойства (атрибут или атрибуты), которое является первичным ключом в данной таблице. Такую операцию нужно проделать для всех таблиц, находящихся в базе данных, иначе контекст не сможет осуществлять операции по работе с БД [6].

4. Блок «Настройка связей между таблицами» предназначен для создания связей (один к одному, один ко многим, многие ко многим) между таблицами в контексте данных. Чтобы настроить связи между таблицами, сначала требуется вызвать метод «Entity<T>», где Т – любой класс (кортеж). Далее, в зависимости от типа связи, вызывается один из методов HasOne или HasMany, в котором требуется выбрать навигационное свойство, указывающее на зависимую таблицу. Следующим этапом является вызов методов WithOne или WithMany, где требуется выбрать навигационное свойство, указывающее на главную таблицу. В конце вызывается метод «HasForeignKey», в котором выбирается свойство, выступающее в качестве внешнего ключа при связывании таблиц. Отдельно стоит выделить тип связи «многие ко многим», так как чтобы создать данную связь требуется задать промежуточную связывающую таблицу. Для этих целей требуется использовать метод «UsingEntity», где Т – любой класс (кортеж), представляющий промежуточную таблицу [6].

5. Блок «Настройка ограничений целостности для атрибутов» предназначен для задания атрибутам таблицы ограничений целостности. Чтобы задать ограничение целостности, для атрибута требуется сначала вызвать метод «Entity<T>», где Т – любой класс (кортеж). Далее необходимо вызвать метод «Property» и в нем выбрать свойство (атрибут), которое присутствует в данной таблице.

После чего следует вызвать метод, который соответствует тому или иному типу ограничения целостности. Например, метод «IsRequired» используется, чтобы атрибут являлся обязательным к заполнению. Полный перечень доступных методов зависит от типа атрибута в базе данных [6].

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;
namespace InternetShop
{
    public static class Program
    {
        public static void Main()
        {
            #region Создание контекста данных
            using InternetShopContext context = new
InternetShopContext();

            #endregion

            #region Добавление поставщика
            Provider provider1 = new Provider
            {
                Name = "Apple",
                Address = "Cupertino, California, USA"
            };

            Provider provider2 = new Provider
            {
                Name = "Google",
                Address = "Mountain View, California, USA"
            };

            context.Providers.AddRange(provider1, provider2);
            context.SaveChanges();
            #endregion

            #region Добавление товаров (Связь один ко многим)
```

```

Product product1 = new Product
{
    Name = "iPhone 1800",
    Count = 1000,
    Price = 1000000,
    ProviderId = provider1.Id
};

Product product2 = new Product
{
    Name = "iPhone 1800 Pro",
    Count = 1000,
    Price = 1200000,
    ProviderId = provider1.Id
};

Product product3 = new Product
{
    Name = "Pixel 600",
    Count = 1000,
    Price = 200000,
    ProviderId = provider2.Id
};

Product product4 = new Product
{
    Name = "Pixel 600 Pro",
    Count = 100,
    Price = 400000,
    ProviderId = provider2.Id
};
context.Products.AddRange(product1, product2,
product3, product4);
context.SaveChanges();
#endregion

#region Добавление пользователя
User user1 = new User

```

```

{
    Name = "Jim",
    PasswordHash = "fhty6er54"
};

User user2 = new User
{
    Name = "Alice",
    PasswordHash = "fhty6er54"
};

context.Users.AddRange(user1, user2);
context.SaveChanges();
#endregion

```

#region Добавление информации о пользователе
(Связь один к одному)

```

UserProfile profile1 = new UserProfile
{
    LastName = "Smith",
    FirstName = "Alice",
    MiddleName = "-",
    Birthday = new DateTime(1900, 10, 12),
    UserId = user1.Id
};

UserProfile profile2 = new UserProfile
{
    LastName = "Smith",
    FirstName = "Jim",
    MiddleName = "-",
    Birthday = new DateTime(1900, 7, 10),
    UserId = user2.Id
};

context.UserProfiles.AddRange(profile1, profile2);
context.SaveChanges();
#endregion

```

```
#region Добавление заказов (Многие ко многим)
```

```
user1.Products.Add(product1);  
user1.Products.Add(product3);  
user2.Products.Add(product2);  
user2.Products.Add(product4);  
context.SaveChanges();  
#endregion
```

```
#region Вывод записей из БД в консоль
```

```
foreach (User user in context.Users)  
{  
    Console.WriteLine($"Id: {user.Id}. Name:  
{user.Name}. Password hash: {user.PasswordHash}.");  
}  
Console.WriteLine();  
foreach (UserProfile profile in context.UserProfiles)  
{  
    Console.WriteLine($"Id: {profile.Id}. User id:  
{profile.UserId}. Last name: {profile.LastName}. First name:  
{profile.FirstName}. Middle name: {profile.MiddleName}. Birthday:  
{profile.Birthday:d}.");  
}  
Console.WriteLine();  
  
foreach (Provider provider in context.Providers)  
{  
    Console.WriteLine($"Id: {provider.Id}. Name:  
{provider.Name}. Address: {provider.Address}.");  
}  
Console.WriteLine();  
  
foreach (Product product in context.Products)  
{  
    Console.WriteLine($"Id: {product.Id}. Name:  
{product.Name}. Price: {product.Price}. Count: {product.Count}.");  
}  
#endregion
```

```
#region Обновление записи в БД
```

```
UserProfile findUserProfile =  
context.UserProfiles.Find(1);  
  
findUserProfile.Birthday = new DateTime(1910, 11, 11);  
  
context.UserProfiles.Update(findUserProfile);  
context.SaveChanges();  
  
Console.WriteLine();  
  
foreach (UserProfile profile in context.UserProfiles)  
{  
    Console.WriteLine($"Id: {profile.Id}. User id:  
{profile.UserId}. Last name: {profile.LastName}. First name:  
{profile.FirstName}. Middle name: {profile.MiddleName}. Birthday:  
{profile.Birthday:d}.");  
}  
  
#endregion
```

```
#region Удаление записи в БД
```

```
User findUser = context.Users.Find(1);  
  
context.Users.Remove(findUser);  
context.SaveChanges();  
  
Console.WriteLine();  
  
foreach (User user in context.Users)  
{  
    Console.WriteLine($"Id: {user.Id}. Name:  
{user.Name}. Password hash: {user.PasswordHash}.");  
}  
  
#endregion
```

```

#region Запросы к БД

Console.WriteLine();

string[] names1 = context.Users.Where(user => user.Id %
2 == 0).Select(user => user.Name).ToArray();

string[] names2 = context.Users.FromSqlRaw("SELECT
* FROM Users WHERE Id % 2 = 0").Select(user =>
user.Name).ToArray();

foreach (string name in names1)
{
    Console.WriteLine($"User name: {name}");
}

foreach (string name in names2)
{
    Console.WriteLine($"User name: {name}");
}

#endregion

Console.ReadKey();
}
}
}

```

Листинг 7. Структура класса Program

В листинге 7 приведен исходный код главной функции консольного приложения, в которой осуществляется работа с базой данных. Разберем устройство главной функции программы и принципы работы с базой данных согласно выделенным в исходном коде блокам:

1. Блок «Создание контекста данных» показывает процесс создания контекста данных для работы с базой данных из консольного приложения [6].

2. Блоки «Добавление поставщика», «Добавление товаров», «Добавление пользователя» и «Добавление информации о

пользователе» показывают процесс добавления записей в базу данных и создание связи «один к одному» и «один ко многим». Чтобы добавить запись в базу данных сначала требуется создать объект или объекты класса (кортеж или кортежи) и проинициализировать у них все свойства (атрибуты) нужными значениями. Далее вызвать свойство у контекста данных, которое соответствует требуемой таблице из базы данных. После чего вызвать метод «Add» или «AddRange», чтобы добавить созданный ранее объект, который требуется добавить в базу данных. Затем требуется вызвать метод «SaveChanges» для сохранения добавленных объектов в базу данных. Особое внимание стоит обратить на создание связей «один к одному» и «один ко многим». Чтобы создать связь между таблицами, требуется установить у объекта, который реализует зависимую таблицу, значение для внешнего ключа. Это можно сделать, взяв значение первичного ключа у объекта, который реализует основную таблицу. Например, взять свойство «Id» у объекта класса «User» и присвоить его значение свойству «UserId» объекта класса «UserProfile». При этом стоит обратить внимание на то, что, если значение первичного ключа является счетчиком, то значение ключа будет присвоено свойству только после сохранения записи в базе данных. Поэтому, если значение свойства «UserId» будет присвоено до сохранения нового пользователя в базу данных, то возникнет ошибка при создании связи между таблицами и сохранение записи в связанной таблице не произойдет [6].

3. Блок «Добавление заказов» показывает процесс добавления записей в БД и создание связи «многие ко многим». Чтобы создать связь многие ко многим требуется выбрать свойство, которое имеет тип `ICollection<T>` у объекта класса главной таблицы. Далее вызвать метод «Add», чтобы добавить объект класса связанной таблицы к связи. После чего требуется вызвать метод «SaveChanges» [6].

4. Блок «Вывод записей из БД в консоль» показывает процесс чтения данных из таблицы. Чтобы осуществить вывод всех данных, находящихся в таблице, требуется вызвать свойство у контекста данных, которое соответствует требуемой таблице. Далее вызвать любой из перечисленных методов: «ToArray», «ToList», «ToDictionary» или «ToHashSet». После чего контекст выполнит запрос вида «SELECT * FROM ИмяТаблицы» к базе данных и вернет

множество записей. Элементы данного множества будут иметь тип класса, который применялся при создании данной таблицы [6].

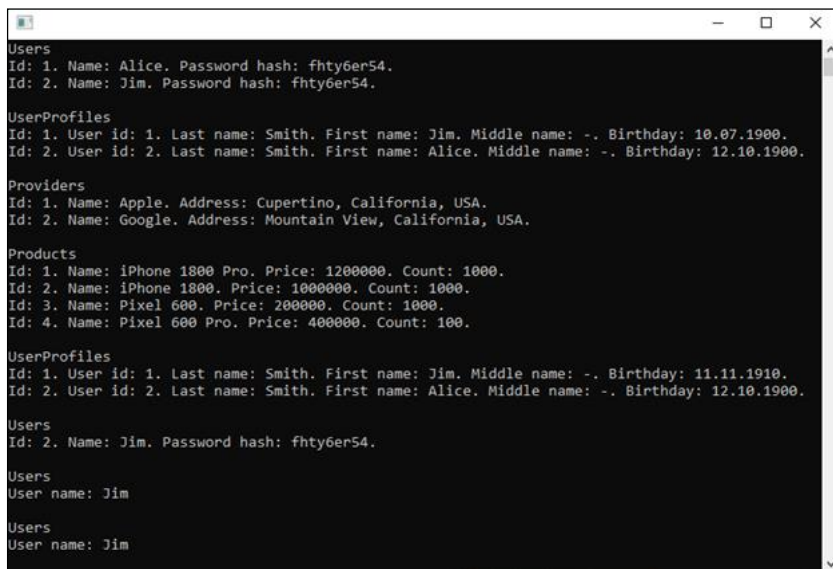
5. Блок «Обновление записи в БД» показывает процесс поиска записи по первичному ключу и обновлению значений атрибутов для нее в базе данных. Чтобы найти запись по первичному ключу, требуется вызвать свойство у контекста данных, которое соответствует требуемой таблице из базы данных, а затем вызвать метод «Find». Результатом работы функции будет объект класса (кортеж). Далее можно изменить значения любых свойств (атрибутов) данного объекта и внесенные изменения сохранить в базе данных. Для сохранения изменений требуется сначала вызвать метод «Update» и передать в него объект, а затем вызвать метод «SaveChanges» по аналогии с добавлением записи [6].

6. Блок «Удаление записи в БД» показывает процесс поиска записи по первичному ключу и удаления ее из базы данных. Процесс поиска полностью аналогичен тому, что был описан выше для блока «Обновление записи в БД». Однако процесс удаления незначительно отличается от процесса обновления. Чтобы удалить запись из базы данных, требуется получить объект, как и для процесса обновления, но требуется вызвать метод «Remove» и передать в него объект соответствующей найденной записи. После чего также требуется вызвать метод «SaveChanges». В результате описанных действий запись будет удалена из базы данных [6].

7. Блок «Запросы к БД» показывает процесс выполнения запроса к базе данных при помощи языка SQL и технологии LINQ to SQL. Чтобы выполнить запрос на языке SQL, требуется вызвать метод «FromSqlRaw», «ExecuteSqlRaw», «FromSqlInterpolated» или «ExecuteSqlInterpolated» и передать в него строку, содержащую запрос на языке SQL. Результат работы данных методов аналогичен результатам методов, приведенных в пункте 4, используемых для чтения всех записей из таблицы базы данных. Основным отличием является возможность дополнительной фильтрации, сортировки и т. д. при помощи языка SQL. Аналогичным образом работает технология LINQ to SQL, но в отличие от языка SQL она реализована в виде методов для языка программирования C#, которые конвертируются в запросы на языке SQL. Имена методов, которые можно использовать в запросах к базе данных, эквивалентны методам, доступным в языке SQL. Преимуществом LINQ to SQL в сравнении с SQL является абстрагирование от СУБД, в которой

размещена база данных. Абстрагирование от СУБД позволяет не зависеть от специфичных команд, расширяющих язык SQL, которые доступны в некоторых СУБД, помимо стандартных [6].

На рис. 27 показан результат работы описанных выше блоков консольного приложения.



```
Users
Id: 1. Name: Alice. Password hash: fhty6er54.
Id: 2. Name: Jim. Password hash: fhty6er54.

UserProfiles
Id: 1. User id: 1. Last name: Smith. First name: Jim. Middle name: -. Birthday: 10.07.1900.
Id: 2. User id: 2. Last name: Smith. First name: Alice. Middle name: -. Birthday: 12.10.1900.

Providers
Id: 1. Name: Apple. Address: Cupertino, California, USA.
Id: 2. Name: Google. Address: Mountain View, California, USA.

Products
Id: 1. Name: iPhone 1800 Pro. Price: 1200000. Count: 1000.
Id: 2. Name: iPhone 1800. Price: 1000000. Count: 1000.
Id: 3. Name: Pixel 600. Price: 200000. Count: 1000.
Id: 4. Name: Pixel 600 Pro. Price: 400000. Count: 100.

UserProfiles
Id: 1. User id: 1. Last name: Smith. First name: Jim. Middle name: -. Birthday: 11.11.1910.
Id: 2. User id: 2. Last name: Smith. First name: Alice. Middle name: -. Birthday: 12.10.1900.

Users
Id: 2. Name: Jim. Password hash: fhty6er54.

Users
User name: Jim

Users
User name: Jim
```

Рис. 27. Результат работы консольного приложения

Результатом проделанной работы является консольное приложение, способное взаимодействовать с базой данных, размещенной в СУБД SQLite. Стоит отметить, что созданное консольное приложение может работать не только в СУБД SQLite, но и в других СУБД. Для этого потребуется заменить провайдера для работы с СУБД в методе конфигурации подключения в классе, описывающем контекст данных.

Раздел 3

БАЗЫ ЗНАНИЙ

Тема 3.1

Основные понятия

Данные и знания (рис. 28). Данные – это отдельные факты, характеризующие объекты, процессы и явления предметной области, а также их свойства.

Знания – это закономерности предметной области (принципы, связи, законы), полученные в результате практической деятельности и профессионального опыта, позволяющие специалистам ставить и решать задачи в этой области.



Рис. 28. НАЗВАТЬ РИСУНОК

Модели представления знаний (рис. 29). Как и для представления данных, знания можно представлять в различной форме. Основные виды моделей представления знаний:

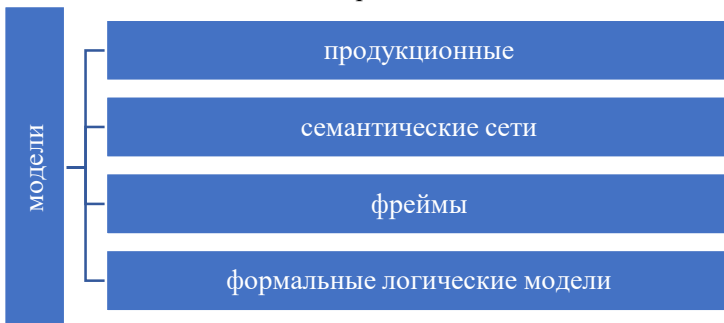


Рис. 29. НАЗВАТЬ РИСУНОК

Продукционная модель. Основана на правилах. Позволяет представить знания в виде предложений типа (рис. 30):



Рис. 30. НАЗВАТЬ РИСУНОК

Семантическая сеть. Семантическая сеть – это ориентированный граф, вершины которого – понятия, а дуги – отношения между ними.

Фреймы. Фрейм – это абстрактный образ для представления некоего стереотипа восприятия (рис. 31).

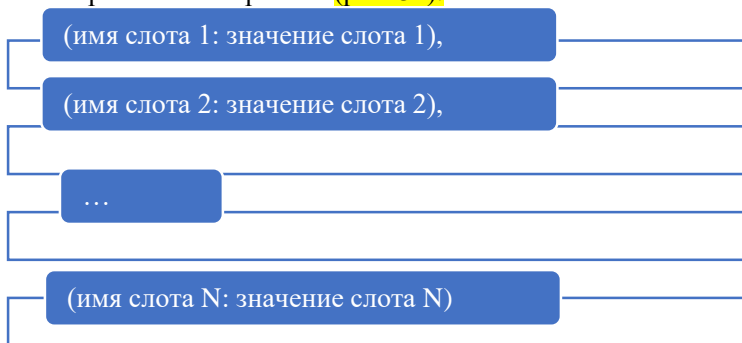


Рис. 31 НАЗВАТЬ РИСУНОК

Формальные логические модели. Основаны на предикатах 1 порядка. Предметная область или задача описывается в виде аксиом.

Лабораторная работа № 17. База знаний

Основной целью данной лабораторной работы является получение навыка формализации знаний, представления знаний в виде правил для продукционной базы знаний.

Задание. Спроектировать и создать базу знаний по указанной предметной области. Варианты предметных областей:

1. Заболевания сердца.
2. Неисправности ЭВМ.
3. ЛОР заболевания.
4. Компьютерные игры.
5. Инфекционные заболевания.
6. Животные.

7. Заболевания эндокринной системы.
8. Мобильные устройства.
9. Заболевания пищеварительной системы.
10. Операционные системы.
11. Заболевания системы кровообращения.
12. Растения.
13. Заболевания органов дыхания.
14. Блюда.
15. Заболевания зубов.
16. Компьютерные комплектующие.
17. Болезни кожи.
18. Математические термины.
19. Врожденные заболевания.
20. Физические понятия.
21. Травмы.
22. Информатика.
23. Ожоги.
24. Мебель.
25. Отравления.

Для выбранной предметной области необходимо сформировать правила, их количество должно составлять не менее 20, причем минимум 10 из них должны быть сложными. При построении правил необходимо ввести переменные, их количество должно составлять не менее 10.

При использовании прямого вывода необходимо построить правила так, чтобы количество циклов для получения ответа составляло не менее 3, а при обратном – необходимо использовать минимум 5 переменных прежде, чем будет достигнута цель.

При описании переменных необходимо их описать и указать какие значения они могут принимать.

В качестве построенной базы предоставить описание переменных и набор правил. После чего описать несколько примеров работы машины вывода (прямого и обратного).

Пример. Описание переменных:

- 1) СГ – состояние горла;
- 2) Н – насморк;
- 3) К – кашель;
- 4) СМ – состояние миндалин;
- 5) Г – голос;

- 6) Т – температура;
- 7) ГБ – головная боль;
- 8) НС – наличие сыпи;
- 9) ОИ – общая интоксикация;
- 10) ДЛ – давление;
- 11) Д – диагноз.

Правила для диагностирования заболевания:

Правила:

- 1) ЕСЛИ СГ = красное, ТО К = сухой;
- 2) ЕСЛИ К = сухой И Т = невысокая И Н = да, ТО Д = ОРВ;
- 3) ЕСЛИ ОИ = тяжелая И СМ = воспаленные И ГБ = да, ТО Д = ангина;
- 4) ЕСЛИ ДЛ = высокое И СГ = красное И СМ = фиброзная пленка И ГБ = да, ТО Д = дифтерия;
- 5) ЕСЛИ Т = высокая И Н = да И К = сухой И НС = да, ТО Д = краснуха;
- 6) ЕСЛИ ОИ = тяжелая И Г = охрипший И НС = да, ТО Д = корь;
- 7) ЕСЛИ Т = высокая, ТО ОИ = тяжелая;
- 8) ЕСЛИ ОИ = тяжелая И НС = да, ТО Д = оспа;
- 9) ЕСЛИ К = влажный И ОИ = тяжелая, ТО Д = бронхит;
- 10) ЕСЛИ К = сухой, ТО Г = охрипший;
- 11) ЕСЛИ ОИ = тяжелая, ТО ДЛ = высокое.

Прямой вывод. Пусть пациент приходит с жалобами на сухой кашель и высокую температуру. То есть запрос будет содержать факты:

К = сухой и Т = высокая.

В первом цикле при просмотре правил машина вывода в правиле 5 встретит комбинацию представленных фактов. Для срабатывания правила в этом правиле присутствуют еще 2 факта: наличие насморка и наличие сыпи. Для срабатывания правила следует спросить пользователя о насморке. Пусть пациент ответил отрицательно (Н = нет). Тогда условие 5 правила срабатывает как ложное. В правиле 7 срабатывает условие и результатом будет определение переменной ОИ = тяжелая. При этом правило исключается из просмотра. Для правила 10 срабатывает условие и в результате добавляется пятый факт Г = охрипший. И правило 10 исключается из дальнейшего просмотра. На это первый цикл окончен.

Второй цикл просмотра правил выполняется с учетом фактов, полученных в первом:

К = сухой, Т = высокая, Н = нет, ОИ = тяжелая, Г = охрипший.

Правило 3 в условной части содержит факт 4 (ОИ = тяжелая), но для срабатывания не хватает значения переменной СМ, поэтому опрашивается пациент. Пусть пациент о состоянии миндалин отвечает, что имеется фиброзная пленка (СМ = фиброзная пленка). Тогда правило 3 признается ложным и продолжается просмотр правил.

Правило 6 содержит в условной части 2 установленных факта, для срабатывания требуется установить значение переменной НС – опрашиваем пациента. Пусть пациент отрицает наличие сыпи, т. е. НС = нет. Правило 6 признается ложным.

Следующее правило, для которого в условной части имеется установленный факт – 11. Правило срабатывает и определяет переменную ДЛ = высокое, правило исключается из дальнейшего просмотра.

По итогам двух циклов получены факты:

К = сухой, Т = высокая, Н = нет, ОИ = тяжелая, Г = охрипший, СМ = фиброзная пленка, НС = нет, ДЛ = высокое.

Третий цикл просмотра правил выполняется с учетом этих фактов.

Правило 4 имеет в условной части два истинных факта, но для срабатывания требуется определить значение переменных СГ и ГБ. Опрашиваем пациента о состоянии горла и головной боли. Пусть пациент отвечает, что горло красное (СГ = красное) и он испытывает головную боль (ГБ = да). Таким образом, правило 4 срабатывает и в результате присваивается значение переменной вывода Д = дифтерия. Диагноз поставлен. Все переменные получили значения и прямой вывод считается окончанным.

Обратный вывод. Начало работы при обратном выводе заключается в выборе переменной логического вывода – Д (диагноз). То есть от системы требуется установить ее значение.

Производится поиск всех правил, в заключении которых присваивается значение этой переменной и проверка фактов их условных частей.

Правило 2 в выводе дает Д = ОРВ. Для определения переменных из условной части этого правила просматриваются правила, в заключении которых они присутствуют. Первая переменная – К –

присутствует в правиле 1, но в условной части требуется значение переменной СГ, которая не определена и не встречается в заключениях правил. Опрашиваем пациента о состоянии горла. Пусть пациент отвечает, что горло красное. Тогда срабатывает правило 1 и определяет переменную $K = \text{сухой}$. Правило 1 исключается из просмотра.

Правило 2 (которое система проверяет) далее требует определения переменной Т. Значение этой переменной не определено и не встречается в заключительных частях правил, поэтому опрашиваем пациента о температуре. Пусть пациент отвечает, что температура высокая ($T = \text{высокая}$).

Далее правило 2 требует знания переменной Н. Она не определена и не встречается в заключительных частях правил, поэтому опрашиваем пациента о наличии насморка. Пусть пациент отрицает наличие насморка ($H = \text{нет}$). При этом правило 2 срабатывает как ложное и удаляется, а из базы знаний выбирается следующее правило с определением переменной Д.

Рассматривается правило 3, которое определяет переменную Д = ангина. Условная часть требует определения переменной ОИ, которая не определена.

Просматриваем правила, которые в заключительной части содержат ОИ. Правило 7 требует определение значения переменной Т, которая известна – $T = \text{высокая}$. Правило 7 срабатывает как истинное и присваивается значение переменной ОИ = тяжелая. Правило 7 исключается из просмотра.

Для правила 3 необходимо определение переменной СМ, которая не определена и не встречается в заключительных частях правил, поэтому опрашиваем пациента о состоянии миндалин. Пусть пациент отвечает, что $CM = \text{фиброзная пленка}$.

Третья переменная в условии правила 3 – ГБ не определена и не встречается в заключительных частях правил базы, опрашиваем пациента. Пусть пациент подтверждает наличие головной боли ($GB = \text{да}$).

Условная часть правила 3 признается ложной и удаляется. Следующее правило, определяющее значение переменной Д – правило 4 – Д = дифтерия.

Условная часть правила 4 требует знания переменной ДЛ, которая не определена, но встречается в заключительной части правила 11.

Правило 11 требует знания переменной ОИ, которая уже определена: ОИ = тяжелая. То есть по правилу 11 получаем переменную ДЛ = высокое. Правило 11 удаляем из списка.

Следующая переменная для правила 4 (которое система проверяет) – СГ. Она известна: СГ = красное.

Третья переменная правила 4 требует значение переменной СМ. Данная переменная определена СМ = фиброзная пленка.

Все условия правила 4 выполнены, правило срабатывает и присваивает значение переменной Д = дифтерия. На этом обратная цепочка рассуждений окончена.

Раздел 4

КУРСОВОЙ ПРОЕКТ

Задание на курсовой проект. Целью выполнения курсового проекта является создание базы данных в среде конкретной СУБД.

Для выполнения проекта необходимо решить следующие задачи:

- описать предметную область;
- сформулировать цель проектирования базы данных;
- описать возможного пользователя базы данных;
- определить круг запросов и задач, которые предполагается решать с использованием созданной базы данных;
- построить концептуальную модель;
- сформулировать требования к базе данных;
- построить реляционную модель, проверить соответствие всех отношений условиям третьей нормальной формы, при несоблюдении — выполнить ее нормализацию или обосновать ее невыполнение;
- создать спроектированную базу данных в среде MS Access;
- разработать приложение для реализации запросов и решения задач:

- 1) количество отношений – не менее 5;
- 2) формы для отображения всех таблиц;
- 3) количество запросов – не менее 4, разного типа (в т. ч. минимум один на SQL);

- 4) формы для отображения результатов всех запросов;
 - 5) количество отчетов – не менее 2 (разного типа);
 - 6) использование масок ввода;
 - 7) наличие главной кнопочной формы;
 - 8) использование разграничения доступа;
 - 9) использование макросов;
 - 10) разработка модуля;
- оценить базу данных с точки зрения возможностей ее дальнейшего развития.

Варианты предметных областей на курсовой проект:

1. Интернет-магазин.
2. Салон автопроката.
3. Магазин розничной торговли.
4. Автосалон.
5. Библиотека.
6. Склад.
7. Сервисный центр.
8. Салон связи.
9. Пункт выдачи товаров.
10. Бухгалтерия.
11. Учет персонала.
12. Учебная часть.
13. Регистратура.
14. Кинопрокат.
15. Прокат товаров для детей.
16. Отдел кадров.
17. Редакция журнала.
18. Ресторан.
19. Доставка готовой еды.
20. Изготовление мебели под заказ.
21. Подбор автозапчастей.
22. Ветеринарная клиника.
23. ЗАГС.
24. Учет исковых заявлений.
25. Типография.

Требования к оформлению. Курсовой проект оформляется с заполнением одной стороны листа формата А4, листы должны быть пронумерованы. Объем работы – не менее 15 страниц (без списка использованной литературы и приложений), но при этом не должен превышать 40 страниц печатного текста.

Текст должен быть отредактирован и напечатан с соблюдением правил оформления научных работ, предусмотренных ГОСТом.

Поля: слева – 30 мм, справа – 10 мм; сверху, снизу – 20 мм.

Шрифт – Times New Roman, размер – 14, интервал – 1,5.

В курсовом проекте обязательно должны присутствовать разделы: ТИТУЛЬНЫЙ ЛИСТ, ОГЛАВЛЕНИЕ, ВВЕДЕНИЕ, ЦЕЛИ И ЗАДАЧИ, «основная часть», ЗАКЛЮЧЕНИЕ, СПИСОК ИСПОЛЬЗОВАННЫХ ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ.

Методические рекомендации по выполнению курсового проекта

Целью выполнения курсового проекта является закрепление теоретических знаний и приобретение навыка создания БД.

При выполнении курсового проекта рекомендуется изучить теоретические основы БД и выполнить все лабораторные работы, описанные в разделах 1–2. Перед началом проектирования дополнительно рекомендуется построить календарный план выполнения проекта для самоконтроля сроков выполнения. Выполнение курсового проекта рассчитано на 16 учебных недель. Важно помнить о том, что необходимо не просто создать БД, но и оформить отчетный документ.

Рекомендуемый порядок выполнения курсового проекта:

1. Создание отчетного документа. Оформление титульного листа, автореферата, оглавления, разделов и пунктов.
2. Подбор литературных источников, оформление библиографических ссылок.
3. Оформление введения (включает вводную часть, постановку цели, задач).
4. Изучение и описание предметной области.
 - Общее описание.
 - а. Выделение сущностей.
 - б. Описание характеристик выделенных сущностей.
 - с. Описание автоматизируемых процессов.

- d. Описание участников автоматизируемых процессов.
 5. Построение ER-модели в нотации Чена.
 6. Построение логической модели.
 7. Построение физической модели.
 8. Заполнение БД.
 9. Создание форм для таблиц. Описание в отчете.
 10. Создание запросов и отчетов. Описание в отчете. (Для запросов и отчетов рекомендовано сделать текстовое описание, представить скрин конструктора и скрин результатов выполнения).
 11. Создание форм для запросов. (Обязательно наличие скринов. Для однотипных форм – представить один и текстом описать, что остальные выполнены аналогично.)
 12. Создание интерфейса (главная кнопочная форма – меню, система навигации, использование макросов, модуля). Описание всех создаваемых элементов.
 13. Написание заключения.
 14. Окончательное оформление отчета.
 15. Создание презентации и подготовка текста доклада.
- Продолжительность доклада 5–7 минут, рекомендуемое количество слайдов – 10.

Окончательный вариант системы и отчет должен удовлетворять следующим требованиям:

1. Мощность БД не менее 50 записей.
2. Наличие и проверка ограничений целостности данных.
3. Наличие автоматизации ввода через справочник-списки.
4. Соответствие функционала предметной области.
5. Наличие кнопок навигации, кнопок выхода, печати.
6. Рациональное использование области окна-экрана.
7. Эргономика интерфейса.
8. Наличие своего текста (не менее 50 % от общего объема).
9. Грамотность.
10. Использование моделей-схем.

Для сдачи курсового проекта необходимо правильно оформить отчет, подготовить презентацию и выступить с устным докладом.

СПИСОК ЛИТЕРАТУРЫ

1. Советов Б. Я., Цехановский В. В., Чертовской В. Д. Базы данных: теория и практика. – Москва : Юрайт, 2012. – 463 с.
2. Дигио С.М. Базы данных. Проектирование и создание. – Москва : ЕАОИ. 2008. – 171 с.
3. Бекаревич Ю. Б. , Пушкина Н. В. Самоучитель Microsoft Access 2000. – Санкт-Петербург : BHV – Санкт-Петербург, 2001.
4. Дженнингс Р. Использование Microsoft Office Access 2003. – М. : Вильямс, 2000.
5. Дубнов П. Ю. Access 2002. – Москва : ДМК Пресс, 2004.
6. Документация по Entity Framework Core. URL: <https://docs.microsoft.com/ru-ru/ef/core/>
7. Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем : учеб. – Санкт-Петербург : Питер. – 2000. – 382 с.

Учебное издание

БАЗЫ ДАННЫХ И БАЗЫ ЗНАНИЙ

Учебно-методическое пособие

Составители: *расшифровка ПО*

Юрчишина М. В.,

Гавриленко А. В.,

Никифоров А. В.