

Государственное образовательное учреждение высшего
профессионального образования
«Ижевский государственный технический университет»

Кафедра «Промышленное и гражданское строительство»

Методические указания

к выполнению лабораторной работы №6
по строительной информатике
для студентов направления 270800
«Строительство»

Ижевск, 2012

УДК 624.014

Методические указания к выполнению лабораторной работы №6 по строительной информатике для студентов направления 270800 «Строительство»

Составитель: Чернов Г.М.

В методических указаниях описывается работа с базами данных на примере СУБД MS Access и на основании конкретного примера рассказывается о возможностях создания и ведения баз данных, излагаются эффективные приемы и методы их проектирования, даются пошаговые описания всех действий, что позволит освоить программу за короткое время.

Ижевский государственный технический университет, 2012

Введение

В этой работе вы узнаете, как проектировать законченные базы данных, поддерживать их, находить значимую информацию и создавать формы для быстрого и легкого ввода данных. Вы также освоите приемы и методы, применяемые для автоматизации широко распространенных задач.

В применении ЭВМ для решения задач информационного обслуживания можно выделить два периода.

Начальный период, когда решением задач обработки информации, организацией данных занимался небольшой круг людей — системные программисты. Этот период характерен тем, что создавались программные средства для решения конкретной задачи обработки данных. При этом для решения другой задачи в которой использовались эти же данные, нужно было создавать новые программы.

Период системного применения ЭВМ. Для решения на ЭВМ комплекса задач создаются программные средства, оперирующие одними и теми же данными, использующие единую информационную модель объекта. Причем эти средства таковы, что они не зависят от характера объекта, его модели, их можно применять для информационного обслуживания различных задач. Человечество пришло к организации информации в информационных системах.

Информационными системами (ИС) называют большие массивы данных вместе с программно-аппаратными средствами для их обработки. Различают следующие виды ИС: фактографические, документальные и экспертные системы.

Фактографическая ИС — это массив фактов — конкретных значений данных об объектах реального мира. Информация хранится в фактографической ИС в четко структурированном виде, поэтому она способна давать однозначные ответы на поставленные вопросы, например: «Кто является победителем чемпионата России по гимнастике в 1999 г.?», «Кому принадлежит автомобиль марки AUDI 80 с регистрационным номером RA899P18?», «Какой номер телефона в бухгалтерии ИжГТУ?» и т.д. Фактографические ИС используются буквально во всех сферах человеческой деятельности — в науке, материальном производстве, на транспорте, в медицине, государственной и общественной жизни, торговле, криминалистике, искусстве, спорте.

Документальные информационные системы обслуживают принципиально иной класс задач, которые не предполагают однозначного ответа на поставленный вопрос. Базу данных таких систем образует совокупность неструктурированных текстовых документов (статьи, книги, рефераты, тексты законов) и графических объектов, снабженная тем или иным формализованным аппаратом поиска. Цель системы, как правило, — выдать в ответ на запрос пользователя список документов или объектов, в какой-то мере удовлетворяющих сформулированным в запросе условиям. Например, выдать список всех статей, в которых встречается слово «Пушкин»

Экспертные системы (ЭС) — интеллектуальные системы, призванные играть роль «советчика», построенного на базе формализованного опыта и

знаний эксперта. Ядром ЭС являются базы знаний, в которых собраны знания экспертов (специалистов) в определенной области, на основе которых ЭС позволяет моделировать рассуждения специалистов из данной предметной области.

Указанная классификация и отнесение ИС к тому или иному типу устарело, так как современные фактографические системы часто работают с неструктурированными блоками информации (текстами, графикой, звуком, видео), снабженными структурированными описателями.

Базы данных

Основа информационной системы, объект ее обработки — база данных (БД). **База данных** — это совокупность сведений о конкретных объектах реального мира в какой-либо предметной области или разделе предметной области. Например, база данных по вузам (высшее образование), база данных по лекарственным препаратам (медицина), база данных по автомобилям (ГИБДД), база данных по стройматериалам (склад) и т.п. *Синоним* термина «база данных» — «банк данных».

Виды моделей данных. Ядром любой базы данных является модель данных, которая представляет собой структуру данных, соглашения о способах их представления и операций манипулирования ими. Модель данных представляет собой формализованное описание объектов предметной области и взаимосвязей между ними.

Различают три основных типа моделей данных: иерархическая, сетевая и реляционная. Иерархическая структура представляет собой совокупность элементов, в которой данные одного уровня подчинены данным другого уровня, а связи между элементами образуют древовидную структуру. В такой структуре исходные элементы порождают другие элементы, причем эти элементы в свою очередь порождают следующие элементы и т.д. Существенно то, что каждый порожденный элемент имеет только одного «родителя». Обратите внимание, что в иерархической структуре порождающим элементом может быть не объект сам по себе, а только *конкретный экземпляр* объекта.

Существуют и более сложные — *сетевые* — структуры, в которых каждый порожденный элемент может иметь более одного порождающего элемента. Сетевая модель данных отличается от иерархической тем, что каждый элемент сетевой структуры данных может быть связан с любым другим элементом. Примером сложной сетевой структуры может служить структура базы данных, содержащая сведения об учащих, занимающихся в различных кружках. При этом возможны занятия одного и того же ученика в разных кружках, а также посещение несколькими учениками занятий одного кружка. И сетевые и иерархические структуры можно свести к простым двумерным таблицам.

Реляционные базы данных

Наиболее удобным и для пользователя, и для компьютера является представление данных в виде двумерной таблицы — подавляющее большинство

современных информационных систем работает именно с такими таблицами. Базы данных, которые состоят из двумерных таблиц, называются *реляционными* (по-английски «relation» — отношение). Основная идея реляционного подхода состоит в том, чтобы представить произвольную структуру данных в виде простой двумерной таблицы.

Примером реализации реляционной модели данных может быть таблица с информацией об учащихся.

№ личного дела	Фамилия	Имя	Отчество	Дата рождения	Адрес	Класс
П-69	Петров	Иван	Васильевич	12.03.99	ул. Горького, 12-34	4А
С-97	Сидоров	Василий	Николаевич	03.12.98	ул. Зеленая, 34-123	4Б
Я-24	Яковлев	Иван	Семенович	15.01.99	пер. Садовый, 45-28	4В
И-35	Иванов	Павел	Николаевич	06.07.98	ул. Горького, 35-14	5А
Е-56	Епишев	Павел	Семенович	19.04.98	ул. Киевская, 78-92	5Б

Как видно из приведенного примера, реляционная таблица обладает следующими свойствами:

- каждая строка таблицы — один элемент данных (сведения об одном учащемся);
- все столбцы в таблице однородные, т.е. все элементы в столбце имеют одинаковый тип и длину (например, в столбце Имя отображаются имена учащихся символьного типа длиной не более 17 символов);
- каждый столбец имеет уникальное имя (например, в таблице нет двух столбцов **Имя**);
- одинаковые строки в таблице не допускаются (запись о каждом учащемся делается только один раз);
- порядок следования строк и столбцов в таблице может быть произвольным (запись об учащемся в таблицу делается при поступлении в школу, при этом порядок следования столбцов не имеет значения).

Структурные элементы реляционной базы данных. На примере реляционной таблицы рассмотрим основные структурные элементы базы данных.

1. В реляционных БД любые совокупности данных представляются в виде двумерных таблиц (отношений), подобных описанному выше списку учащихся. При этом каждая таблица состоит из фиксированного числа столбцов и некоторого (переменного) количества строк. Описание столбцов принято называть макетом таблицы (схемой отношения).

2. Каждый столбец таблицы представляет поле — элементарную единицу логической организации данных, которая соответствует неделимой единице информации — реквизиту объекта данных (например, фамилия учащегося, адрес).

Для описания поля используются характеристики:

- имя поля (например, № **личного дела**, **Фамилия**);
- тип поля (например, символьный, дата);
- дополнительные характеристики (длина поля, формат, точность).

Например, поле **Дата рождения** может иметь тип «дата» и длину 8 (6 цифр и 2 точки, разделяющих в записи даты день, месяц и год).

3. Каждая строка таблицы называется *записью*. Запись логически объединяет все поля, описывающие один объект данных, например, все поля в первой строке таблицы описывают данные об учащемся **Петрове Иване Васильевиче 12.03.99** рождения, проживающем по адресу **ул. Горького, 12—34**, обучающемся в **4А** классе, номер личного дела — **П-69**. Система нумерует записи по порядку: 1, 2, ... > *n*, где *n* — общее число записей (строк) в таблице на данный момент. В отличие от количества полей (столбцов) в таблице, количество записей в процессе эксплуатации БД может как угодно меняться (от нуля до миллионов). Количество полей, их имена и типы тоже можно изменить, но это уже особая операция, которая называется изменением макета таблицы.

4. В структуре записи файла указываются поля, значения которых являются *простым ключом*, которые идентифицируют экземпляр записи. Примером такого простого ключа в таблице **Учащиеся** является поле **№ личного дела**, значение которого однозначно определяет один объект таблицы — одного учащегося, так как в таблице нет двух учащихся с одинаковым номером личного дела.

5. Каждое поле может входить в несколько таблиц (например, поле **Фамилия** может входить в таблицу **Список занимающихся в театральном кружке**).

Что такое Microsoft Access?

MS Access — это система управления реляционными базами данных, предназначенная для работы на автономном ПК или в локальной вычислительной сети под управлением Microsoft Windows. Другими словами, MS Access — это набор инструментальных средств для создания и эксплуатации информационных систем. Средствами Access можно выполнить следующие операции:

1. Проектирование базовых объектов ИС — двумерных таблиц с разными типами данных, включая поля объектов OLE.

2. Установление связей между таблицами, с поддержкой целостности данных, каскадного обновления и удаления записей.

3. Ввод, хранение, просмотр, сортировку, модификацию и выборку данных из таблиц с использованием различных средств контроля информации, индексирования таблиц и аппарата логической алгебры (для фильтрации данных).

4. Создание, модификацию и использование производных объектов ИС (форм, запросов и отчетов), с помощью которых в свою очередь выполняются следующие операции:

- оптимизация пользовательского ввода и просмотра данных (формы);
- соединение данных из различных таблиц;
- проведение групповых операций (т.е. операций над группами за-

писей, объединенных каким-то признаком), с расчетами и формированием вычисляемых полей;

- отбор данных с применением аппарата логической алгебры (запросы).
- составление печатных отчетов по данным, которые содержатся в таблицах и запросах БД.

MS Access обладает исключительно мощными, удобными и гибкими средствами визуального проектирования объектов, и это дает возможность пользователю при минимуме предварительной подготовки довольно быстро создать полноценную ИС — на уровне таблиц, форм, запросов-выборок и отчетов.

Технология работы с MS Access

Вы можете запускать MS Access и завершать ее работу любым из стандартных способов, предусмотренных в среде Windows

Объектом обработки MS Access является файл базы данных, имеющий произвольное имя и расширение *.mdb. В этот файл входят основные объекты MS Access: таблицы, формы, запросы, отчеты, страницы, макросы и модули. После загрузки Access ожидает от вас одного из следующих распоряжений:

- создать новую базу данных, т.е. файл с расширением *.mdb;
- открыть существующую базу данных, т.е. файл с расширением *.mdb.

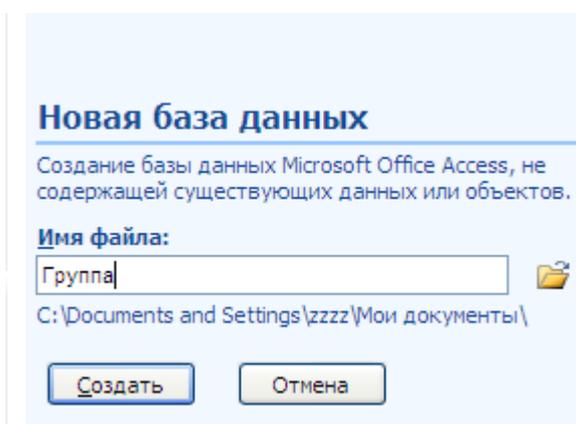
Создание БД. После выбора команды **Новая база данных...** на экране появляется стандартное диалоговое окно в котором следует открыть нужную папку и задать имя создаваемого файла базы данных. Например, как показано на рисунке, для нашей базы данных мы выбрали папку Мои документы, а имя файла — Группа.mdb.

Создав файл, Access раскрывает пустое *окно базы данных*, и в этом окне можно будет проводить все операции — создавать и манипулировать объектами БД.

MS Access является *многооконным* приложением, однако в любой момент может быть открыта только одна база данных. Именно ее окно является *главным* окном документа в приложении Access, и его закрытие означает закрытие соответствующего файла *.mdb.

Окно базы данных порождает множество *дочерних* окон объектов (таблицы, запроса, формы и т.д.), и каждое такое окно может быть закрыто автономно — любым из стандартных способов Windows.

С окном любого объекта (дочерним окном) можно работать либо в *оперативном* режиме (например, вводить или просматривать данные в таблице),



либо в режиме конструктора (например, изменять макет таблицы).

Основные понятия MS Access. Объекты MS Access

База данных Access может иметь следующие объекты: таблицы, формы, запросы, отчеты. Кроме того, квалифицированные пользователи могут работать еще с двумя объектами: макросами и модулями. *Макрос* — это набор специальных макрокоманд (например, ОткрытьФорму, ПечататьОтчет и т.п.), а *модуль* — это программа, написанная на языке Access Basic или Visual Basic для приложений.

Таблица является базовым объектом MS Access. Все остальные объекты являются производными и создаются нами только на базе ранее подготовленных таблиц.

Форма не является самостоятельным объектом Access: она просто помогает вводить, просматривать и модифицировать информацию в таблице или запросе. Запросы и отчеты выполняют самостоятельные функции: выбирают, группируют, представляют, печатают информацию

Каждый объект MS Access имеет имя. В Microsoft Access действуют следующие ограничения на имена полей, элементов управления и объектов:

- имя должно содержать не более 64 символов;
- имя может включать любую комбинацию букв, цифр, пробелов и специальных символов за исключением точки (.), восклицательного знака (!), надстрочного символа (^) и квадратных скобок ([]);
- не должно начинаться с символа пробела;
- не должно включать управляющие символы (с кодами ASCII от 0 до 31);
- не должно включать прямые кавычки (") в именах таблиц, представлений и хранимых процедур в проекте Microsoft Access.

Хотя пробелы внутри имен полей, элементов управления и объектов являются допустимыми, при некоторых обстоятельствах они могут вызывать конфликты в программах Visual Basic.

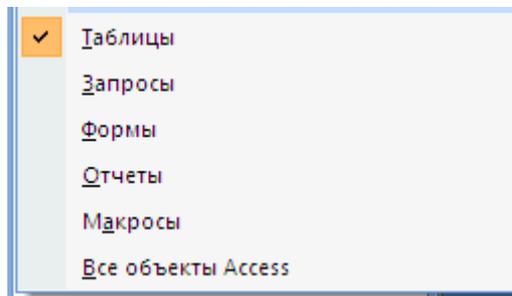
Определяя имя для поля, элемента управления или объекта, полезно проверить, не совпадает ли это имя с именем свойства или другого элемента, используемого Microsoft Access (для русских имен такая ситуация может возникнуть при совпадении с именем свойства или функции, определяемых пользователем).

С каждым объектом базы данных работа выполняется в отдельном окне, причем предусмотрено два режима работы:

1) *режим таблицы*, когда просматривается, изменяется или выбирается информация;

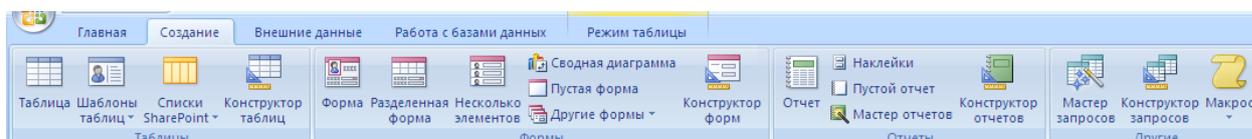
2) *режим конструктора*, когда создается или изменяется макет, структура объекта (например, структура таблицы).

Кроме этого, в файл базы данных входит еще один документ, имеющий собственное окно: *Схема данных*. В этом окне мы создаем, просматриваем,



изменяем и разрываем связи между таблицами. Эти связи помогают нам контролировать данные, создавать запросы и отчеты.

В окне базы данных имеется лента **Создать** с вкладками: **Таблица, Форма, Отчет, Другие (Запрос, Макрос / Модуль)**. Если выбрана какая-либо вкладка, то в ее окне отображается список существующих объектов это-



го типа данной БД.

Например, если выбрать вкладку **Таблица**, то в окне отображается список таблиц открытой базы данных. Чтобы открыть таблицу, надо *выделить* ее имя в этом списке и нажать кнопку **Открыть**. Чтобы включить в БД *новую* таблицу, надо нажать выбрать команду **Создать** ⇒ **Таблица**. Чтобы исправить *макет* существующей таблицы, надо выделить ее имя в списке и выбрать команду **Конструктор**.

Такие же операции выполняются со всеми другими объектами базы данных Access.

Если выбрать команду **Работа с базами данных** ⇒ **Схема данных**, на экране появится окно документа **Схема данных**.

Типы данных, которые могут иметь поля в Microsoft Access, приведены в таблице.

Типы данных

Тип данных	Использование	Размер
Текстовый	Текст или комбинация текста и чисел, например, адреса, а также числа, не требующие вычислений, например, номера телефонов, инвентарные номера или почтовые индексы	До 255 символов
Числовой	Числовые данные, используемые для математических вычислений, за исключением финансовых расчетов (для них следует использовать тип «Денежный»). Для более точного определения типа числа используйте свойство Размер поля	1, 2, 4 или 8 байт. 16 байт только для кодов репликации (GUID)
Поле МЕМО	Длинный текст или числа, например примечания или описания	До 64 000 символов
Дата/время	Даты и время	8 байт
Денежный	Значения валют. Денежный тип используется для предотвращения округлений во время вычислений. Предполагает до 15 символов в целой части числа и 4 — в дробной	8 байт

Счетчик	Автоматическая вставка последовательных (увеличивающихся на 1) или случайных чисел при добавлении записи. Этот тип поля удобно применять для первичного ключа таблицы. В качестве значений таких полей Access автоматически выбирает целые порядковые номера (1,2,...). В дальнейшем номер, присвоенный записи при ее создании, не изменяется (независимо от удаления, вставки новых записей и т.п.)	4 байта. 16 байт только для кодов репликации (GUID).
Логический	Поля, содержащие только одно из двух возможных значений, таких как «Да/Нет», «Истина/ Ложь», «Вкл/Выкл»	1 бит
Поле объекта OLE	Объекты (например, документы Microsoft Word, электронные таблицы Microsoft Excel, рисунки, звуки и другие двоичные данные), созданные в других программах, использующих протокол OLE. Объекты могут быть связанными или внедренными в таблицу Microsoft Access. Для отображения объекта OLE в форме или отчете необходимо использовать присоединенную рамку объекта	До 1 гигабайта (ограничено объемом диска)
Гиперссылка	Поле, в котором хранятся <i>гиперссылки</i> , имеющие вид пути или URL-адреса	До 64 000 символов
Мастер подстановок	Создает поле, позволяющее выбрать значение из другой таблицы или из списка значений, используя поле со списком. При выборе данного параметра в списке типов данных запускается мастер для автоматического определения этого поля	Тот же размер, который имеет первичный ключ, являющийся также и полем подстановок; обычно — 4 байта

Примечание. Поля типов «Числовой», «Дата/время», «Денежный» и «Логический» имеют predetermined форматы вывода данных. Формат вывода можно выбрать в ячейке свойства **Формат** поля **Format**. Можно также создать собственные форматы вывода для всех типов данных, кроме объектов OLE.

Технология создания базы данных в Access

Любой объект базы данных можно создать либо вручную с помощью Конструктора, либо с помощью Мастера. При создании базы данных, как правило, выполняется следующая последовательность шагов:

1. Сначала мы должны выполнить проектирование базы данных, которое заключается в следующем:

- определение цели создания базы данных (назначение базы данных, как она будет использоваться, и какие сведения она должна содержать);
- определение таблиц, которые должна содержать база данных (определить, какие сведения будут храниться в таблицах);
- определение полей в таблице;
- определение ключевых полей;

- определение связей между таблицами.

2. После создания нужных таблиц, полей и связей необходимо еще раз просмотреть структуру базы данных и выявить возможные недочеты. Желательно это сделать, пока таблицы не заполнены данными.

3. Затем следует ввести в таблицы достаточный объем данных для проверки структуры. После этого создаются черновые формы, отчеты и выполняется анализ, отображаются ли в них те данные, что ожидалось. Чтобы проверить связи в базе данных, проверяется, удастся ли создать запросы для получения нужных сведений. При обнаружении проблем следует выполнить доработку структуры базы данных.

4. Если структуры таблиц отвечают поставленным требованиям, то можно ввести все данные. Затем можно создать все необходимые объекты базы данных (формы, отчеты, запросы).

5. В заключение можно выполнить разработку макросов и программ на языке Visual Basic для приложений, с помощью которых объекты базы данных объединяются в единое приложение. В приложении связанные между собой задачи организуются таким образом, чтобы пользователи могли сконцентрироваться на конкретной работе, а не на изучении механизма работы приложения или программ, обеспечивающих его работу.

Рассмотрим технологию разработки базы данных на примере БД **Группа**.

Определим цель создания данной базы — хранение сведений об учащихся. В качестве базового объекта базы данных определим таблицу, в которой будут храниться следующие данные об учащихся: **№ личного дела, фамилия, имя, отчество, дата рождения, домашний адрес, класс**. Для их размещения определим одноименные поля таблицы. В качестве ключа таблицы зададим поле **Код**.

Создание таблицы. Выберем значок **Таблицы** в ленте **Создать**. Из предложенных способов создания выберем вариант **Конструктор**, для чего в ленте **Главная** выберем **Режим Конструктора**.

После этого в окне Access раскроется окно таблицы в режиме конструктора, как показано на рисунке. В верхней части окна находится создаваемый или модифицируемый **макет** таблицы, который представляет собой просто список полей с указанием имени поля, типа данных и описания.

В столбце **Поле** вы набираете имя поля, а в следующем столбце указываете тип данных для этого поля. Тип данных нужно выбрать из раскрывающегося списка. Как только курсор оказывается в столбце **Тип данных**, в нижней части окна возникает **бланк свойств** (характеристик) данного поля. Он представляет собой перечень свойств (слева **название** свойства, справа — **значение** этого свойства) с окном подсказки по каждому свойству. Перечень свойств меняется, в зависимости от типа данных, который в текущий момент отображается в столбце **Тип данных**. Щелкнув мышью на поле **значения** в бланке свойств, вы можете изменить это значение (в рамках допустимого для этого типа данных). Большинство значений принимается системой по умолчанию, многие свойства можно изучить самостоятельно. Некоторые значения

можно выбрать из раскрывающегося списка.

Имя поля	Тип данных
Код	Счетчик
Фамилия	Текстовый
Имя	Текстовый
Отчество	Текстовый
Дата рождения	Дата/время
Домашний адрес	Текстовый
Класс	Текстовый

Свойства поля	
Общие	Подстановка
Размер поля	3
Формат поля	
Маска ввода	
Подпись	
Значение по умолчанию	
Условие на значение	
Сообщение об ошибке	
Обязательное поле	Нет
Пустые строки	Да
Индексированное поле	Нет
Сжатие Юникод	Да
Режим IME	Нет контроля
Режим предложений IME	Нет
Смарт-теги	

При выборе значения свойства принципиально важно следовать следующим рекомендациям:

- Для текстового и числового поля надо указать **размер** поля, причем для текста — это допустимая длина значения (например, 20 или 40 символов), а для числа — формат представления в машине (байт, целое (два байта), длинное целое и т.д.).
- Для поля **Дата/время** обязательно надо указать формат, чтобы система знала, как обрабатывать вводимые данные. Например, если выбрать **Краткий формат даты**, система будет ожидать от вас ввода именно *даты* (в русской версии — ДД.ММ.ГГГГ), а если выбрать **Краткий формат времени**, в этом поле придется набирать ЧЧ:ММ (часы и минуты).
- В качестве значения свойства **Условие на значение** вы можете указать логическое выражение, которое должно принимать значение **True** («Истина») при вводе данных в это поле. В следующем свойстве можно записать произвольное сообщение об ошибке, которое будет выдано системой, например: «Это значение поля недопустимо». В свойстве **Обязательное поле** можно указать «Да» (пустые значения не допускаются) или «Нет» (пустые значения допускаются).
- Если в **первичный** ключ вашей таблицы входит одно поле, в свойстве **Индексированное поле** для него выберите: «Да, совпадения не допускаются», а затем щелкните на кнопке **Определить ключ** (с изображением ключа). Тем самым вы определите первичный ключ своей таблицы (и запретите ввод записей с повторяющимся значением первичного ключа).

Итак, следуя вышеприведенным рекомендациям, определим поля таблицы. В графе **Имя поля** зададим имя «№ личного дела». Для определения типа данных этого поля, щелкнув стрелку в графе **Тип данных**, раскроем список возможных типов данных и выберем вариант **Текстовый**. В области окна конструктора **Свойства поля** выберем вкладку **Размер поля** и определим максимальное количество знаков для ввода в этом поле — 10 символов.

Обратите внимание, что при выборе различных параметров свойства поля в правой части выводится подсказка о назначении параметра.

Действуя аналогично, введем следующие данные о других полях таблицы.

Имя поля	Тип данных	Свойства (формат поля)
Фамилия	Текстовый	20 символов
Имя	Текстовый	20 символов
Отчество	Текстовый	20 символов
Дата рождения	Дата/время	Длинный формат даты
Домашний адрес	Текстовый	50 символов
Класс	Текстовый	3 символа

До того, как сохранить таблицу, определим первичный ключ. Это будет поле Код с типом данных Счетчик.

Переключим отображение созданной таблицы в **Режим таблицы**. При этом обязательно сохраним таблицу под именем **Учащиеся**. Для переключения отображения таблицы выберем команду **Режим таблицы** в ленте **Главная**.

Примечание. Поле первичного ключа определять не обязательно, но желательно. Если первичный ключ не был определен, Microsoft Access при сохранении таблицы спросит, нужно ли создать ключевое поле.

Операции с данными в таблице

Ввод данных. Выбрав в окне таблицу **Учащиеся**, щелкнем кнопку **Открыть**. Установим курсор в поле Фамилия и введем значение **Ушаков**. По окончании ввода значения поля нажмем клавишу **Tab** для перехода к следующему полю. В остальные поля этой записи введем данные: **Дмитрий; Александрович; 8.11.98; ул. Зеленая, 33-66; 5в; У-55** После окончания ввода значений всех полей записи нажмем клавишу **Tab** для перехода к следующей записи. Введем еще несколько записей. Заполненная таблица будет выглядеть, как показано на рисунке.

Код	Фамилия	Имя	Отчество	Дата рождения	Домашний адрес	Класс	№ личного	Добавить поле
1	Ушаков	Дмитрий	Александрович	8 ноября 1998 г.	ул. Зеленая, 33-66	5в	У-55	
2	Петров	Иван	Васильевич	12 марта 1999 г.	Ул. Красная, 22-11	4а	П-69	
3	Лыкова	Екатерина	Михайловна	3 августа 1998 г.	Ул. Синяя, 44-88	5а	Л-32	
4	Епишев	Павел	Николаевич	3 декабря 1998 г.	Ул. Серая, 35-55	5б	Е-54	
5	Сидоров	Василий	Сидорович	3 декабря 1999 г.	Ул. Зеленая, 35-64	4б	С-97	
6	Окунев	Олег	Карпович	6 июля 1998 г.	Ул. Малиновая, 34-12	5а	О-67	
7	Карпов	Семен	Семенович	9 февраля 1998 г.	Ул. Красная, 23-11	5а	К-35	
8	Селезнев	Федор	Максимович	15 января 1999 г.	Ул. Синяя, 15-5	4в	С-24	
*	(№)							

Если вам не нравится ширина столбца таблицы (например, она слишком

велика или наоборот мала и скрывает часть данных), ее можно уменьшить или увеличить — точно так же, как вы изменяли ширину столбца в Excel.

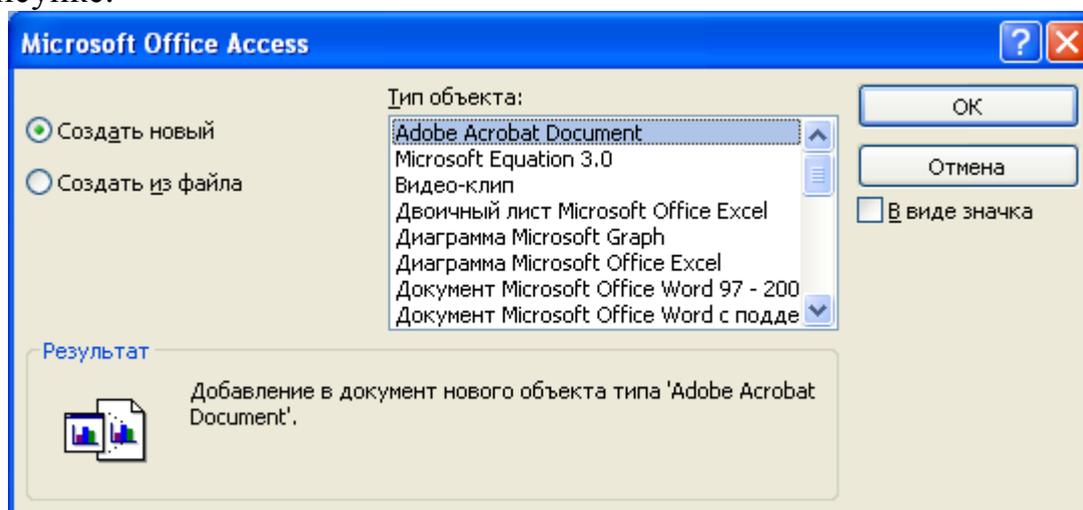
Перемещение по таблице. В строке состояния указывается общее число записей в таблице и номер *текущей* записи. Текущая запись отмечается стрелкой в левой части окна (в области маркировки записей). Для перемещения по таблице служат **кнопки переходов** в строке состояния (слева направо: переход к первой записи таблицы, к предыдущей записи, к следующей записи и к последней записи таблицы).

Чтобы переместить текстовый курсор в произвольную ячейку таблицы, можно просто щелкнуть на ячейке мышью.

Кроме того, по таблице можно перемещаться с помощью клавиш **Tab**, **Shift + Tab**, стрелок курсора.

Редактирование таблицы. При вводе данных используется основной стандарт редактирования. Закончив ввод или модификацию данных в конкретном поле, нажмите **Tab** или **Enter** (или щелкните мышью в другой ячейке таблицы).

Для ввода (внедрения) объекта OLE надо щелкнуть правой кнопкой мыши на его поле в таблице и выбрать OLE-сервер из списка, как показано на рисунке.



После внедрения OLE-объекта, отображаемым в таблице значением его поля будет название соответствующего OLE-сервера (например, **Точечный рисунок BMP**). Чтобы просмотреть или отредактировать объект (или, скажем, чтобы воспроизвести звукозапись) надо дважды щелкнуть на этом названии.

Операции с записями и столбцами.

Изменим структуру таблицы, вставив перед столбцом **Дата рождения** столбец с полем **Пол**. Для этого дважды щелкнем на заголовке пустого столбца **Добавить поле** и зададим столбцу имя **Пол**.

Для определения свойств нового поля переключим таблицу в режим Конструктора, выбрав команду **Конструктор**, в ленте **Главная**. Определим для поля **Пол** логический тип данных, а на вкладке **Общие** в области описания Свойства поля выберем формат **Да/Нет** и отредактируем его, записав как **Муж/Жен**.

Переключим отображение таблицы в **Режим таблицы** и в поле **Пол** выставим флажки в записях с мальчиками.

Переместим столбец **Пол**, установив его справа от поля **Дата рождения**. Для этого выделим столбец **Пол**, щелкнув область выделения поля со словом **Пол**, затем отпустим кнопку мыши. Снова нажмем кнопку мыши и, удерживая кнопку мыши в области выделения поля, перетащим столбец **Пол** в нужное положение.

Отсортируем записи в таблице по алфавиту фамилий, для чего, щелкнув поле **Фамилия**, нажмем кнопку **Сортировка по убыванию** в раскрывающемся списке у заголовка поля или воспользуемся командой **Сортировка** в закладке **Сортировка и фильтр** ленты **Главная**.

Код	Фамилия	Имя	Отчество	Пол	Дата рождения	Домашний адрес	Класс	№ личного
1	Ушаков	Дмитрий	Александрович	<input checked="" type="checkbox"/>	8 ноября 1998 г.	ул. Зеленая, 33-66	5в	У-55
5	Сидоров	Василий	Сидорович	<input checked="" type="checkbox"/>	3 декабря 1999 г.	Ул. Зеленая, 35-64	4б	С-97
8	Селезнев	Федор	Максимович	<input checked="" type="checkbox"/>	15 января 1999 г.	Ул. Синяя, 15-5	4в	С-24
2	Петров	Иван	Васильевич	<input checked="" type="checkbox"/>	12 марта 1999 г.	Ул. Красная, 22-11	4а	П-69
6	Окунев	Олег	Карпович	<input checked="" type="checkbox"/>	6 июля 1998 г.	Ул. Малиновая, 34-12	5а	О-67
3	Лыкова	Екатерина	Михайловна	<input type="checkbox"/>	3 августа 1998 г.	Ул. Синяя, 44-88	5а	Л-32
7	Карпов	Семен	Семенович	<input checked="" type="checkbox"/>	9 февраля 1998 г.	Ул. Красная, 23-11	5а	К-35
4	Епишев	Павел	Николаевич	<input checked="" type="checkbox"/>	3 декабря 1998 г.	Ул. Серая, 35-55	5б	Е-54

Использование фильтра для отбора данных в таблице. Работая с таблицей в Оперативном режиме, можно установить *фильтр*, т.е. задать логическое выражение, которое позволит выдавать на экран только записи, для которых это выражение принимает значение **True** («Истина»).

Выделив в поле **Класс** значение 5а, выберем в раскрывающейся кнопке **Выделение** на вкладке **Сортировка и Фильтр** равно 5а. После этого таблица будет выглядеть, как показано на рисунке.

Чтобы снять фильтр и увидеть все записи в таблице, щелкнем кнопку

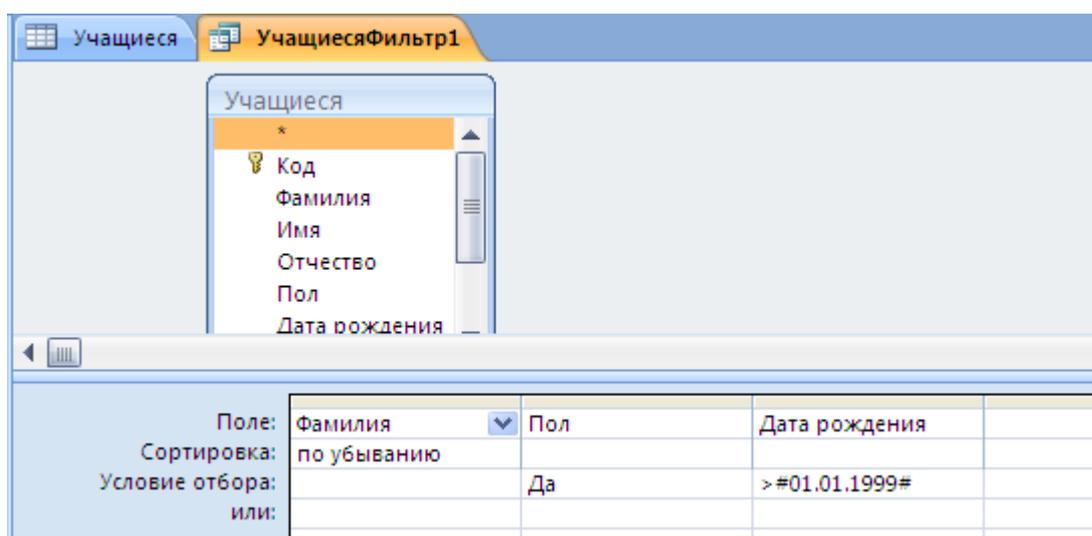
Код	Фамилия	Имя	Отчество	Пол	Дата рождения	Домашний адрес	Класс	№ личного
6	Окунев	Олег	Карпович	<input checked="" type="checkbox"/>	6 июля 1998 г.	Ул. Малиновая, 34-12	5а	О-67
3	Лыкова	Екатерина	Михайловна	<input type="checkbox"/>	3 августа 1998 г.	Ул. Синяя, 44-88	5а	Л-32
7	Карпов	Семен	Семенович	<input checked="" type="checkbox"/>	9 февраля 1998 г.	Ул. Красная, 23-11	5а	К-35

Применить фильтр на вкладке **Сортировка и фильтр**.

Для отбора записей, удовлетворяющих более сложным условиям отбора, используется **расширенный фильтр**. Например, создадим фильтр, который отбирает из всех записей таблицы только те, которые удовлетворяют условиям: учащиеся - мальчики, родившиеся после 1-го января 1999г.

Для этого выберем на вкладке **Сортировка и Фильтр** команду **Дополнительно** и подкоманду **Расширенный фильтр**. После этого в верхней области окна Access откроется список полей таблицы Учащиеся, а в нижней области окна будет раскрыт бланк записи фильтра.

Добавим в бланк поле **Пол**, затем в ячейке **Условие отбора** для этого поля запишем значение **Да**. Чтобы указать порядок расположения отфильтрованных записей таблицы, выберем ячейку **Сортировка**, щелкнем стрелку и выберем порядок сортировки **по возрастанию**. В строке поля зададим еще одно поле **Дата рождения** и в ячейке **Условие отбора** для этого поля введем условие отбора в виде логического выражения **>#01.01.99#**.



Для применения созданного расширенного фильтра щелчком кнопки **Применить фильтр** и посмотрим результат действия расширенного фильтра. Как видно, в таблице отображается три записи, удовлетворяющие заданным в фильтре условиям.

Код	Фамилия	Имя	Отчество	Пол	Дата рождения	Домашний адрес	Класс
5	Сидоров	Василий	Сидорович	<input checked="" type="checkbox"/>	3 декабря 1999 г.	Ул. Зеленая, 35-64	4б
8	Селезнев	Федор	Максимович	<input checked="" type="checkbox"/>	15 января 1999 г.	Ул. Синяя, 15-5	4в
2	Петров	Иван	Васильевич	<input checked="" type="checkbox"/>	12 марта 1999 г.	Ул. Красная, 22-11	4а

Примечание. Фильтры сохраняются автоматически при сохранении таблицы или формы. Таким образом, при повторном открытии таблицы или формы можно будет снова применить сохраненный фильтр.

Создание и использование формы

Нами рассмотрен универсальный способ представления в окне всех полей конкретной таблицы. Основные недостатки этого способа заключаются в следующем.

- Если полей слишком много, они не умещаются на экране и приходится прибегать к различным манипуляциям, чтобы отрегулировать представление: убирать некоторые столбцы, изменять ширину столбцов, перемещаться по таблице с помощью полосы прокрутки.
- Если в таблице имеются какие-то коды, таблица теряет информативность: приходится иметь под рукой классификатор, чтобы понять, какому экземпляру объекта соответствует тот или иной код.

Чтобы упростить просмотр, ввод и модификацию данных в конкретной таблице, можно создать для нее одну или несколько форм. **Форма** — это документ, в окне которого отображается, как правило, одна запись таблицы, причем пользователь имеет возможность по своему усмотрению разместить поля на форме. Таблица и форма — основные объекты в современных информационных системах. Они неотделимы друг от друга и размещены в одном окне документа MS Access на разных вкладках.

Формы используются для следующих целей:

- ввода/редактирования данных, помещенных в таблицу;

- организации диалога выбора, предварительного просмотра и печати нужного отчета;
- открытия других форм и отчетов с помощью кнопок данной формы.

По структуре форма похожа на окно диалога. Связь между формой и источником данных для нее создается при помощи графических объектов, называемых элементами управления. Наиболее часто используемым для вывода и ввода данных элементом управления является поле. В зависимости от природы поля вы можете сохранить для него обычное представление (поле ввода, как в таблице), или исключить поле, или описать группой кнопок-переключателей (если поле имеет несколько допустимых значений), или флажком (для логических данных), или полем ввода со списком и т.п.

Заголовок формы

Учащиеся2

Область данных

Код		Фамилия		Имя		Отчество	
Код		Фамилия		Имя		Отчество	
Пол	Дата рождения			Домашний адрес			Класс
<input checked="" type="checkbox"/>	Дата рождения			Домашний адрес			Класс
№ личного дела							
№ личного дела							
Фотография							

В форме имеются следующие разделы.

1. Заголовок, который отображается вверху и содержит общие сведения, например название формы.
2. Колонтитулы, в которых отображаются сведения для вывода форм на печать, например, название столбцов, дата и номер страницы.
3. Область данных формы, которая включает поясняющий текст, данные, вычисленные значения, графические элементы (рисунки).
4. Примечание формы, в котором содержатся сведения, общие для всех записей, инструкции по работе с формой.

Как и любой объект, MS Access может создать форму вручную или воспользоваться услугами **Мастера форм**. Форма создается для конкретной таблицы или конкретного запроса. Подробные сведения о создании и использовании форм можно получить, выбрав в справке Access тему **Формы**. Для создания формы необходимо в окне базы данных щелкнуть значок **Формы** в

ленте **Создать**. В диалоговом окне **Новая форма** выбрать строку **Конструктор** и выбрать имя таблицы, на которой нужно основать форму, например **Учащиеся**. Если форма не будет содержать данные (например, если нужно создать кнопочную форму для открытия других форм или отчетов), не выбирайте ничего из этого списка. Нажмите кнопку **ОК**.

Как показано на рисунке, форма будет открыта в режиме **Конструктора**. Для размещения поля таблицы в форме следует взять его мышью в таблице и перетащить в область данных.

При выборе любого из компонентов в формах и отчетах Microsoft Access отображает маркеры перемещения для поля и подписи, а также маркеры изменения размеров для выбранного компонента. Подпись находится слева от поля и перемещается вместе с ним. Для выделения отдельно подписи или поля нужно взять объект мышью за левый верхний маркер и перетащить.

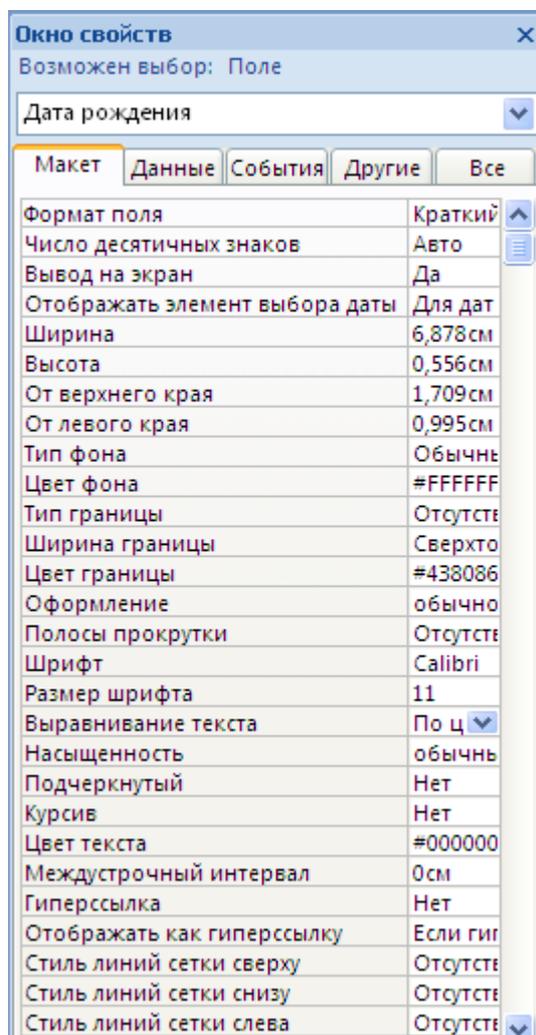
Для одновременного перемещения нескольких объектов выделите элементы управления, удерживая нажатой клавишу **Shift**.

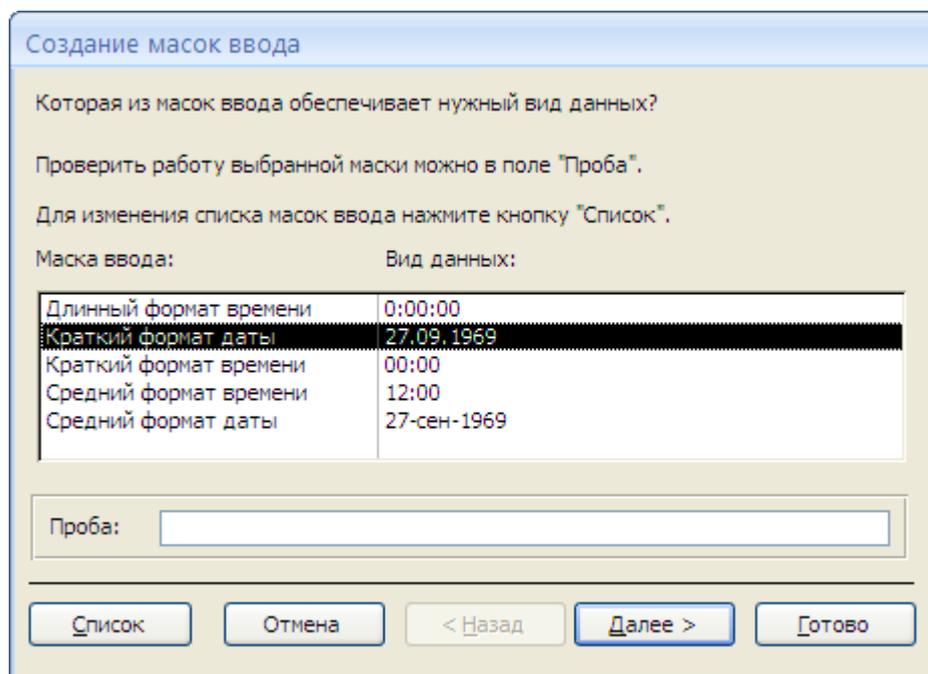
Наведите указатель на границу любого выделенного элемента управления и, когда указатель примет форму руки, переместите элементы управления в нужную позицию.

Для изменения форматирования элемента формы можно использовать кнопку **Режимы** **Режим макета** ⇒ кнопки в ленте **Работа с макетами форм** ⇒ **Формат**; **Работа с макетами форм** ⇒ **Упорядочить** или команды контекстного меню.

Для изменения свойств поля, выбрав объект **формы**, щелкнем правой кнопкой мыши и в контекстном меню выберем команду **Свойства**. Выбирая вкладки, можно изменить свойства объекта, например, цвет, отображаемые в поле формы данные, связать с этим объектом определенные события и т.п..

Зададим маску ввода даты, для чего, указав поле **Дата рождения**, нужно вызвать контекстное меню и в нем выбрать команду **Свойства**. В окне **Поле: Дата рождения** на вкладке **Данные** щелкнем строку **Маска ввода** для вызова диалогового окна. В окне **Создание масок ввода** выберем вариант **Краткий формат даты**, в поле **Проба** зададим вариант ввода даты для проверки избранной маски ввода. Щелкнув кнопку **Далее**, отредактируем маску ввода. Для применения созданной маски ввода щелкнем кнопку **Готово**.





Создадим заголовок формы, для чего выберем в меню **Макеты** ⇒ **Формат** команду **Заголовок**. После этого выберем место в зоне заголовка формы, и введем ее в текст «Учащиеся», параметры надписи можно задать через **Свойства**.

После сохранения макета формы данные таблицы будут представлены, как показано.

Код	Фамилия	Имя	Отчество
5	Сидоров	Василий	Сидорович

Пол Дата рождения: 3 декабря 1999 г. Домашний адрес: Ул. Зеленая, 35-64

Класс: 46

№ личного дела: С-97

Фотография:

Используя поле номера записи, можно просматривать записи таблицы. Щелкнув кнопку **Последняя запись**, затем **Следующая запись** откроем форму для создания новой записи и введем данные о новом учащемся. Выбрав команду **Режим таблицы**, просмотрим изменения данных в таблице **Учащиеся**.

Создание и использование запроса

Что такое запрос? В спроектированной нами таблице **Учащиеся** содержится вся информация, необходимая для решения поставленной нами задачи. Но как этой информацией пользоваться? Как узнать, например, сколько человек учится в 4А классе или у скольких учащихся день рождения в апреле? Не сидеть же перед компьютером с калькулятором, ручкой и бумагой! Решить эту проблему поможет запрос.

В общем случае **запрос** — это **вопрос о данных**. Существуют разные типы запросов (на выборку, запрос с параметрами, перекрестные запросы, запрос на изменение таблицы, запросы SQL).

Запрос-выборка в MS Access.

Простейший из запросов — **запрос-выборка** — это **производная таблица**, которая содержит те же структурные элементы, что и обычная таблица (столбцы-поля и строки), и формируется на основе фактических данных базы данных. Запрос на выборку отбирает данные из одной или более таблиц по заданным условиям, а затем отображает их в нужном порядке. Запрос можно создать с помощью мастера или самостоятельно в режиме конструктора, выбрав таблицы или запросы, содержащие нужные данные, и заполнив бланк запроса.

При создании макета запроса (т.е. производной таблицы) в общем случае нам необходимо выполнить следующие базовые операции:

- указать системе, какие поля и из каких таблиц мы хотим включить в запрос;
- описать вычисляемые поля, т.е. поля, значения которых являются функциями значений существующих полей (например, средняя успеваемость — это среднее арифметическое значение всех оценок);
- описать групповые операции над записями исходных таблиц (например, нужно ли объединить группу записей и указать условие отбора (мальчики из 5Б класса) в одну и просуммировать значение их роста для расчета среднего значения роста учащихся класса).

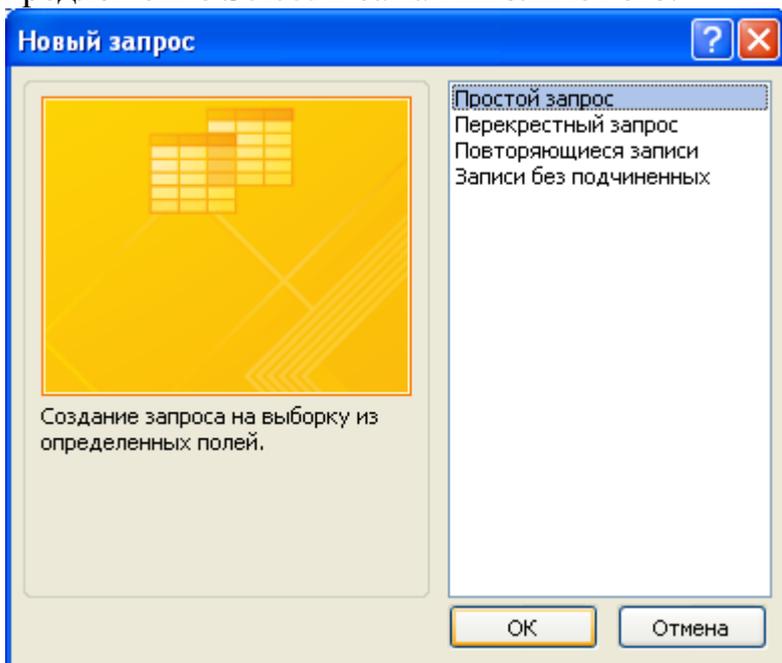
При разработке конкретного запроса допускается любое сочетание базовых операций.

Создание запроса-выборки.

В общем случае для создания произвольного запроса используется универсальный язык SQL. В предложении этого языка (**Select** — **Выбрать**) можно описать все базовые операции: какие поля и откуда выбрать, какие вычислить, как их сгруппировать (просуммировать, пересчитать, найти среднее и т.п.) и при каких условиях включить записи в выборку. Однако в реальности пользоваться этим языком могут только специалисты (или очень грамотные пользователи).

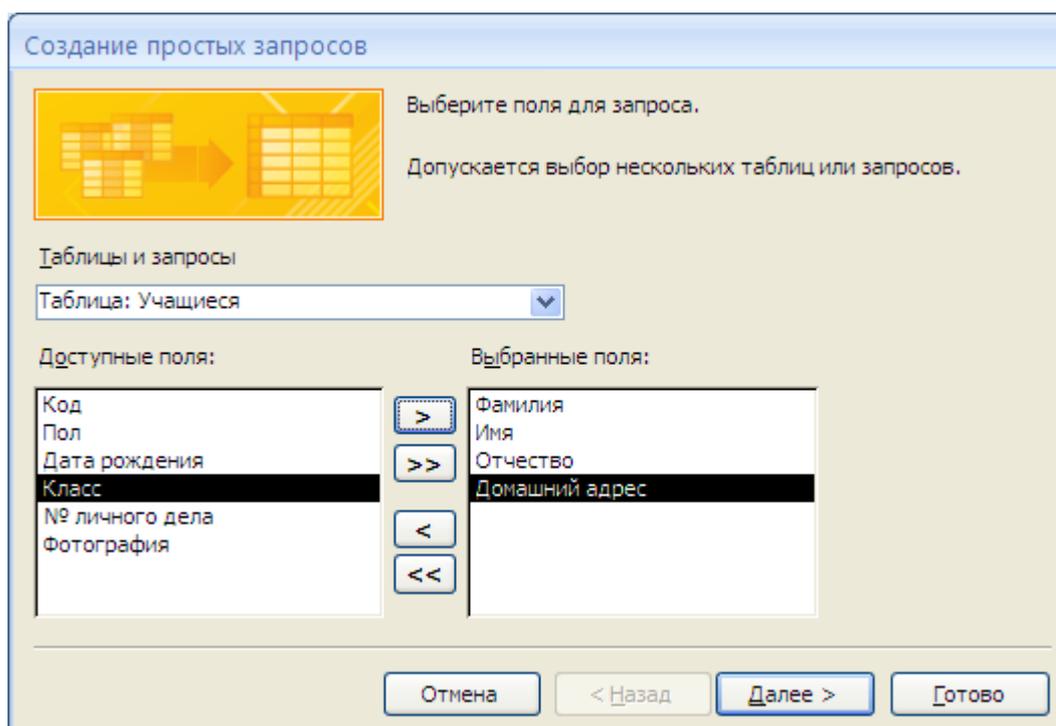
А для обычных людей разработчики придумали упрощенный механизм создания запроса, называемый **QBE (Query By Example — запрос по образцу)**. Вам предлагают **бланк QBE** — некую модель, заготовку запроса, и на этом бланке, пользуясь определенными правилами, вы сообщаете системе о

своих планах: помечаете поля, вводите выражения, значения и т.п. На основании заполненного вами бланка система сама создает соответствующее предложение **Select** и сама выполняет его.



Рассмотрим создание простого запроса на выборку с помощью мастера. Мастер простого запроса на выборку создает запросы для получения данных из полей, выбранных в одной или нескольких таблицах или запросах. С помощью мастера можно также вычислять суммы, количества и средние значения для всех записей или определенных групп записей, а также находить максимальное и минимальное

значение в поле. Однако нельзя ограничить количество записей, возвращаемых этим запросом, с помощью условий отбора.

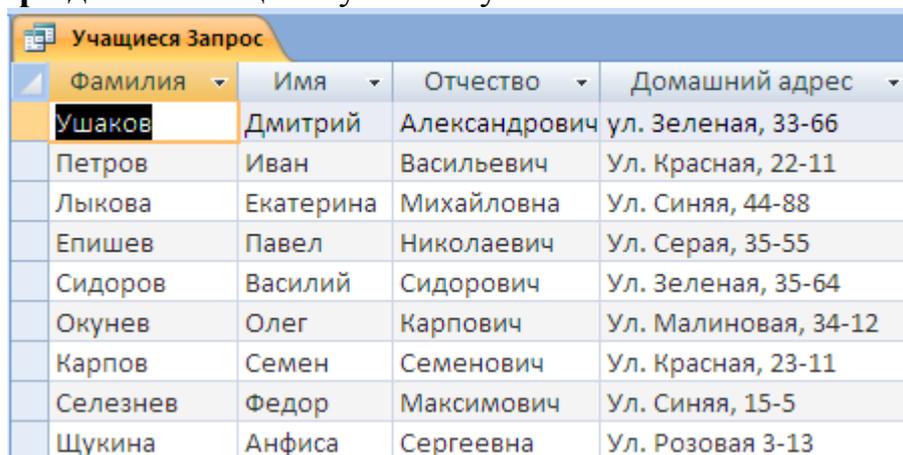


Для создания запроса выберите в окне базы данных **Группа** значок **Мастер Запросов** в ленте **Создать**. В диалоговом окне **Новый запрос** выберите мастера **Простой запрос** и нажмите кнопку **ОК**.

Укажите имя таблицы **Учащиеся**, на которой будет основан создаваемый запрос, а затем выберите поля, данные которых нужно использовать, как показано на рисунке.

Примечание. Если необходимо, укажите дополнительную таблицу или запрос и выберите нужные поля. Повторяйте этот шаг до тех пор, пока не будут выбраны все необходимые поля.

Следуя инструкциям Мастера, выберем вариант **подробный отчет** и щелкнем кнопку **Далее**. В последнем диалоговом окне зададим имя запроса **Запрос** и запустим полученный запрос, выбрав вариант **Открыть запрос для просмотра данных** и щелкнув кнопку **Готово**.



Фамилия	Имя	Отчество	Домашний адрес
Ушаков	Дмитрий	Александрович	ул. Зеленая, 33-66
Петров	Иван	Васильевич	Ул. Красная, 22-11
Лыкова	Екатерина	Михайловна	Ул. Синяя, 44-88
Епишев	Павел	Николаевич	Ул. Серая, 35-55
Сидоров	Василий	Сидорович	Ул. Зеленая, 35-64
Окунев	Олег	Карпович	Ул. Малиновая, 34-12
Карпов	Семен	Семенович	Ул. Красная, 23-11
Селезнев	Федор	Максимович	Ул. Синяя, 15-5
Щукина	Анфиса	Сергеевна	Ул. Розовая 3-13

Выбрав команду **Режим SQL** в ленте **Главная** ⇒ **Режимы**, мы можем увидеть текст созданного запроса на языке SQL:

```
SELECT Учащиеся.[Фамилия], Учащиеся.[Имя], Учащиеся.[Отчество], Учащиеся.[Домашний адрес]  
FROM Учащиеся;
```

Как видно из текста, в запросе на языке SQL записана команда выбора из таблицы **Учащиеся** и описан порядок размещения полей таблицы в таблице-результате действия запроса.

Если получился не тот запрос, который был нужен, можно изменить этот запрос в режиме конструктора. Откроем созданный запрос в режиме **Конструктор**. В верхней части этого окна показана схема данных выбранных таблиц с указанием связей и имен всех полей, у нас это таблица **Учащиеся**. В нижней части окна размещается бланк **QBE**, который представляет собой макет некоей таблицы. Столбцы этой таблицы соответствуют полям создаваемого запроса, а число строк переменное и зависит от количества таблиц и операций сортировки и отбора. В строке **Поле:** указываются имена столбцов (полей) создаваемого запроса. Существующее имя можно выбрать из раскрывающегося списка (щелкнув мышью на поле) или просто перенести в ячейку **Поле:** методом «Drag-and-Drop» из таблиц в верхней части окна запроса.

Если бы в строке **Поле:** размещалось вычисляемое поле, тогда в ячейке этой строки нужно было бы ввести формулу: **<Имя поля>:<Выражение>**

Например: **Стоимость: [Количество]* [Цена]**

В строке **Сортировка:** можно указать порядок вывода на экран записей (по возрастанию, по убыванию).

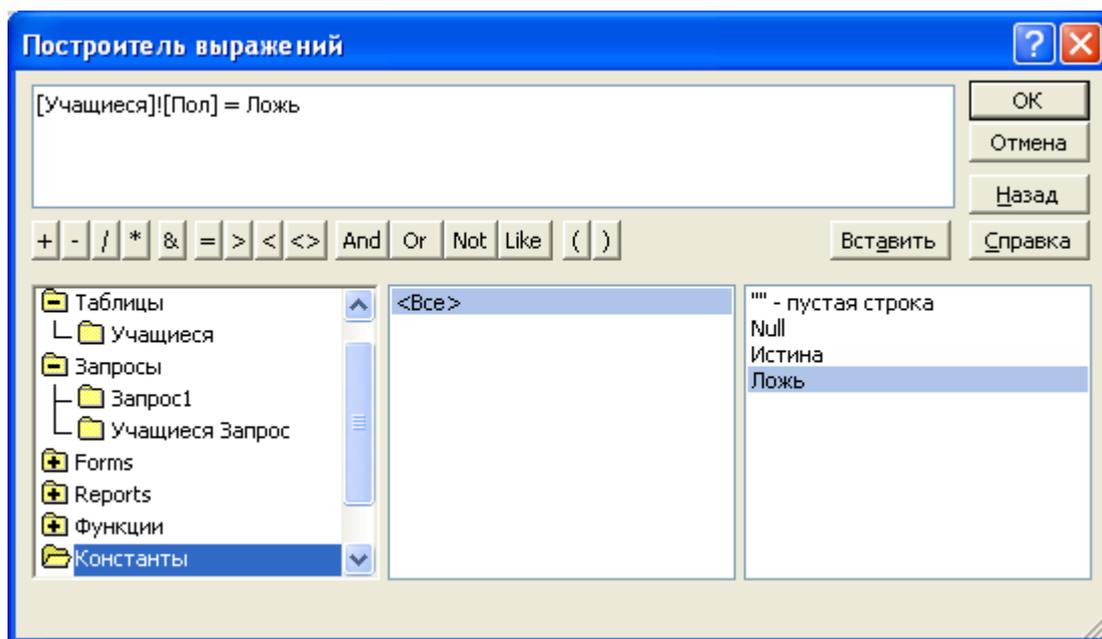
В поле **Вывод на экран:** находятся *флажки*, установив или сбросив флажок, мы разрешим или запретим вывод данного столбца на экран.

В строках **Условие отбора:** и **или:** можно указать условные или логические выражения, которые позволят нам отобрать для запроса только записи, удовлетворяющие заданному условию. Обратите внимание на следующие соглашения, предусмотренные в Access.

1. Условные выражения, набранные в разных столбцах строки **Условие отбора**, по умолчанию соединяются между собой знаком **AND**. Например, если соседние столбцы имеют имена **Пол** и **Класс** и вы набрали в них **=True** и **=5Б**, то тем самым вы сформулировали логическое выражение: **Пол = True** и **Класс = 5Б**.

2. Условные выражения, набранные в соседних строках одного и того же столбца, соединяются между собой знаком **OR**. Например, если столбец имеет имя **Класс** и мы набрали в строке **Условие отбора:** **= 4А**, а в строке **или:** **=5А**, то тем самым мы сформулировали логическое выражение: **[Класс] = 4А Or [Класс] = 5А**.

Изменим запрос, добавив условие отбора только тех учащихся, которые не имеют мужской пол. Для этого установим курсор в столбце **Пол** на строке **Условие отбора** и, щелкнув правую кнопку мыши, вызовем контекстное меню, а в меню выберем команду **Построить**. В списке папок построителя выражений, щелкнув папку **Запрос1**, раскроем список полей. Дважды щелкнув поле **Пол**, включим э то поле в область выражений. Щелкнув оператор «**=**», включим его в выражение.



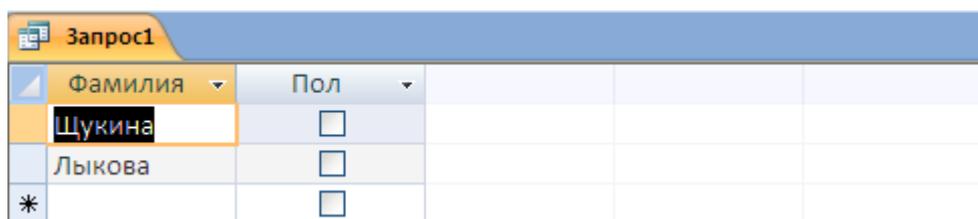
Выбрав папку **Константы**, в списке констант дважды щелкнем на значении «**Ложь**». Щелкнув **ОК**, завершим построение логического выражения.

Выбрав в меню **Вид** команду **Режим SQL**, мы можем просмотреть запись созданного запроса на языке SQL:

```
SELECT Учащиеся.Фамилия, Учащиеся.Пол
FROM Учащиеся
WHERE (([Учащиеся]![Пол]=False));
```

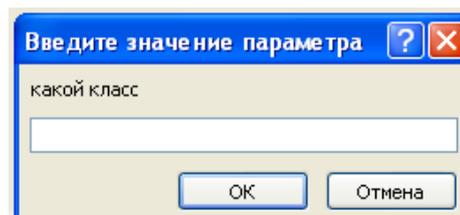
Как видно из текста, в описание запроса добавлено выражение **WHERE**

((Пол = False)), которое отбирает в таблице те записи об учащихся, в которых поле **Пол** не является мужским. После просмотра записи выберем в закладке **Результаты** команду **Выполнить** и просмотрим таблицу-результат запроса.



Фамилия	Пол			
Щукина	<input type="checkbox"/>			
Лыкова	<input type="checkbox"/>			
*	<input type="checkbox"/>			

Создание запроса с параметром. В частном случае, проектируя запрос, вы можете создать из него своеобразную микропрограмму, которая будет работать по-разному, в зависимости от вводимого вами *параметра*. Запрос с параметрами выводит одно или несколько predetermined диалоговых окон, в которых запрашивается ввод условий отбора при каждом запуске запроса. В ответ на запрос пользователь должен ввести значения параметров (условия отбора). Создадим новый запрос на выборку **Запрос2** в режиме конструктора и выберем поля **Фамилия, Имя, Класс**.



Для каждого поля, которое предполагается использовать как параметр, следует ввести в ячейку строки **Условие отбора** текст приглашения, заключенный в квадратные скобки. Это приглашение будет выводиться при запуске запроса. «Снимем» флажок вывода столбца **Класс** на экран, а в поле **Условие отбора:** (в этом столбце!) наберем текст приглашения: **[Какой класс]** (квадратные скобки обязательны).

Для просмотра результатов нажмем кнопку **Выполнить**. После этого на экран будет выведено окно с приглашением ввести параметр. Введем параметр **5а** и щелкнем ОК для выполнения выборки записей из таблицы в соответствии с заданным параметром-условием.

Чтобы вернуться в режим конструктора запроса, нажмем кнопку **Режим конструктора**.

Изменим условие отбора данных, чтобы в запросе отображались данные об учащихся 5а класса, родившихся летом 1998 г. Для этого в запрос добавим поле **Дата рождения**. В ячейку строки **Условие отбора** поля **Дата рождения** введем текст условия: **>#01.06.98#And<#01.09.98#**.

Для проверки действия созданного запроса выберем команду **Режим таблицы**.

Создание и использование отчета

Отчет представляет собой эффективный способ представления данных в печатном формате. Имея возможность управлять размером и внешним видом всех элементов отчета, пользователь может отобразить сведения желаемым образом. Как правило, для формирования отчета создают запрос, в котором

собирают данные из разных таблиц с включением вычисляемых полей, группировкой, условиями отбора (любая операция необязательна). Далее, по общим правилам MS Access, на базе такого запроса проектируют отчет, который позволяет:

- представить данные в удобной для чтения и анализа форме;
- сгруппировать записи (по нескольким уровням) с вычислением итоговых и средних значений;
- включить в отчет и напечатать графические объекты (например, диаграммы).

Сведения, отображаемые в отчете:

- заголовков отчета и столбцов;
- данные, определяемые выражением, которое задается в макете отчета;
- данные из полей базовой таблицы, запроса или инструкции SQL;
- итоговые значения, вычисляемые с помощью выражений, заданных в макете отчета.

Для создания связи между отчетом и его исходными данными применяются элементы управления: поля, содержащие имена или числа, надписи для заголовков, декоративные линии для графического оформления отчета.

Структура отчета. Структура отчета аналогична структуре формы. Сведения в отчете могут быть разбиты на разделы. Каждый раздел имеет определенное назначение и печатается на странице и в отчете в заданном порядке.

Как в форме, так и в отчете могут присутствовать следующие разделы:

- заголовок;
- верхний и нижний колонтитулы;
- область данных;
- примечание отчета.

Заголовок отчета выводится один раз в начале отчета и содержит наиболее общие сведения: название и логотип фирмы, название отчета.

Верхний колонтитул печатается вверху каждой страницы и может содержать подписи столбцов — графы отчета. Нижний колонтитул печатается внизу каждой страницы и может содержать номер страницы, дату создания отчета.

Основной раздел отчета — область данных, в котором размещаются данные отчета из каждой записи базового источника (таблицы, запроса). Если в отчете используются сгруппированные записи, то в основном разделе отчета в каждой группе может использоваться заголовок группы. В нем отображаются сведения, общие для всей группы (название группы).

Примечание отчета выводится один раз в конце отчета и содержит итоговые данные отчета.

Размеры разделов можно изменить в режиме конструктора отчета.

В Access 2007 возможны следующие способы создания отчета: Автоотчет, Мастер отчетов, создание отчета в режиме Конструктора.

Создание отчета с помощью мастера. В ленте **Создание** выберем ко-

манду **Отчеты** ⇒ **Мастер отчетов**. В диалоговом окне **Создание отчетов** выберем таблицу **Учащиеся**, на которой должен быть основан отчет. Включим в список **Выбранные поля** поля из таблицы. Для этого, указав поле в списке **Доступные поля**, щелкнем кнопку **>**. Если нужно включить в список **Выбранные поля** все поля из списка **Доступные поля**, то нужно щелкнуть кнопку **>>**.

Примечание. Порядок расположения полей таблицы, в списке **Выбранные поля** определяет порядок расположения столбцов будущего отчета. Используйте кнопку **<** и **>** для формирования списка полей в том порядке, как вам нужно.

Завершив формирование списка **Выбранные поля**, щелкнем кнопку **Далее**. На следующем шаге диалога с **Мастером отчетов** определим уровни группировки полей в отчете. Пусть данные в нашем отчете будут сгруппированы по классам, для этого, выбрав поле **Класс**, щелкнем кнопку **>**.

Щелкнув кнопку **Далее**, зададим сортировку записей в группах по фамилиям в алфавитном порядке. На следующем шаге создадим макет отчета, выбрав ступенчатый вариант отчета на странице с книжной ориентацией. Включим опцию **Настроить ширину полей для размещения на одной странице**. Щелкнув кнопку **Готово**, завершим создание отчета и посмотрим результат на экране.



Адреса школьников				
Класс	Фамилия	Имя	Отчество	Домашний адрес
4а	Петров	Иван	Васильевич	Ул. Красная, 22-11
4б	Сидоров	Василий	Сидорович	Ул. Зеленая, 35-64
4в	Селезнев	Федор	Максимович	Ул. Синяя, 15-5
5а	Окунев	Олег	Карпович	Ул. Малиновая, 34-12
	Лыкова	Екатерина	Михайловна	Ул. Синяя, 44-88
	Карпов	Семен	Семенович	Ул. Красная, 23-11

Автоматизация выполнения задач обработки данных с помощью макрокоманд

СУБД Microsoft Access для целей автоматизации операций с объектами баз данных использует два типа средств: макросы и модули.

Макросом называют набор из одной или более макрокоманд, выполняющих определенные операции, такие как открытие форм или печать отчетов. Например, при нажатии пользователем определенной кнопки можно за-

пустить макрос, который распечатает отчет.

Макрос может быть как собственно макросом, состоящим из последовательности макрокоманд, так и группой макросов. В некоторых случаях для решения, должна ли в запущенном макросе выполняться определенная макрокоманда, может применяться условное выражение.

Модулем называют набор описаний и процедур на языке Visual Basic for Applications (VBA), собранных в одну программную единицу.

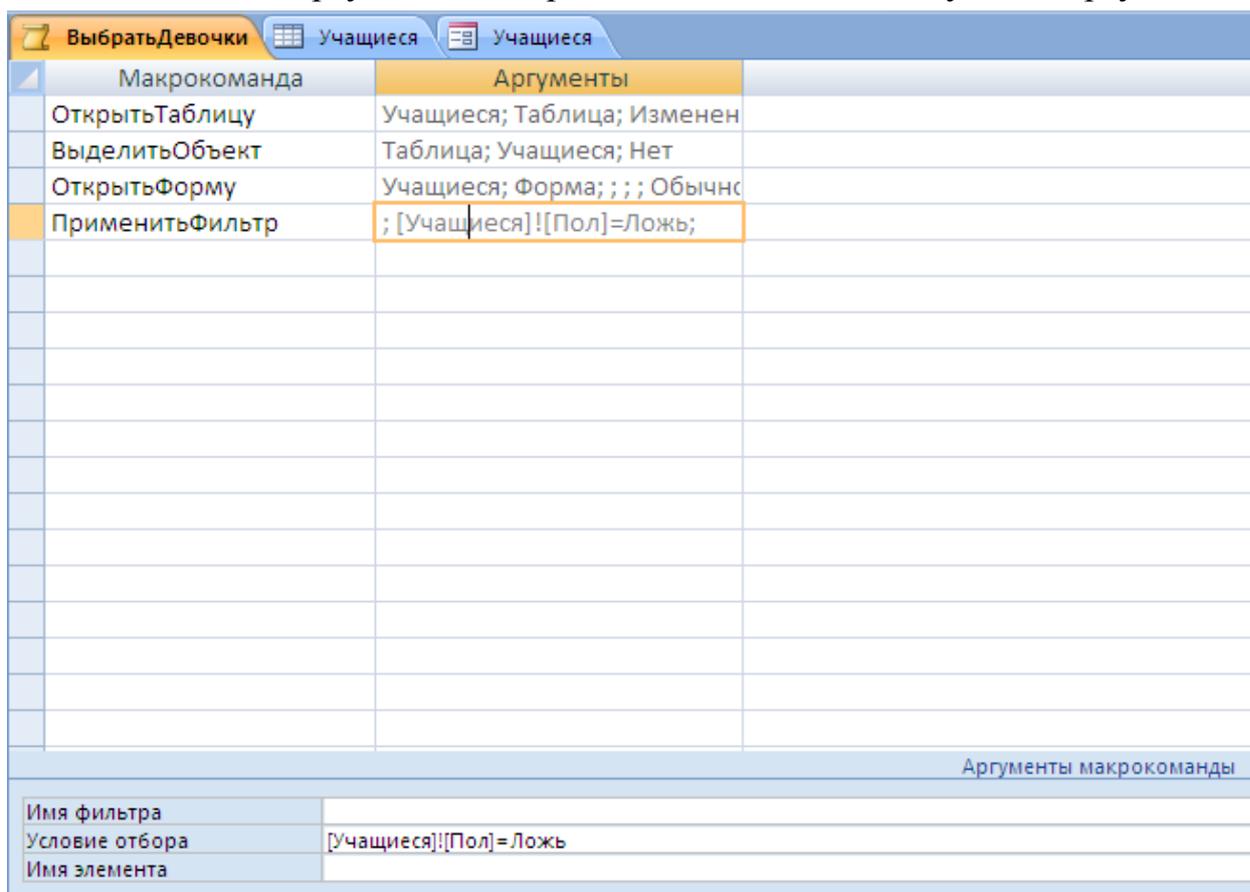
Макрос является удобным средством выполнения простых задач:

- открытие и закрытие форм;
- вывод на экран и скрытие панелей инструментов или запуск отчетов;
- связывание различных объектов базы данных;
- определение общих назначенных клавиш;
- выполнение макрокоманды или набора макрокоманд при открытии базы данных.

Создание макроса

Создадим макрос, который открывает таблицу **Учащиеся**, выделяет эту таблицу, открывает для нее форму **Форма1**, затем применяет к ней фильтр, позволяющий просмотреть записи об учащихся женского пола.

1. В ленте **Создать** базы данных выберем команду **Макрос**.
2. В столбце **Макрокоманда**, нажав кнопку раскрытия списка, открывающую список макрокоманд, выберем команду **Открыть таблицу**. В области задания аргументов макрокоманды зададим следующие аргументы:



Макрокоманда	Аргументы
ОткрытьТаблицу	Учащиеся; Таблица; Изменен
ВыделитьОбъект	Таблица; Учащиеся; Нет
ОткрытьФорму	Учащиеся; Форма; ; ; ; Обычн
ПрименитьФильтр	; [Учащиеся]![Пол]=Ложь;

Аргументы макрокоманды	
Имя фильтра	
Условие отбора	[Учащиеся]![Пол]=Ложь
Имя элемента	

- Имя таблицы **Учащиеся**
- Режим **Таблица**
- Режим данных **Изменение**

3. Введем текст комментария к макрокоманде (необязательно). Комментарии облегчают понимание и сопровождение макросов.

4. Для добавления в макрос других макрокоманд перейдем на следующую строку и повторим шаги 2 и 3.

Использование условий в макросах. В некоторых случаях требуется выполнять макрокоманду или серию макрокоманд только при выполнении некоторых условий. Например, если в макросе проверяется соответствие данных в форме условиям на значение, то для одних значений может потребоваться вывести одно сообщение, а для других значений другое сообщение. В подобных случаях условия позволяют определить порядок передачи управления между макрокомандами в макросе.

Условие задается логическим выражением. В зависимости от значения логического выражения управление может передаваться разным макрокомандам.

Условное выражение вводится в ячейку **Условие отбора** в окне макроса. Если условие истинно, выполняется макрокоманда, содержащаяся в данной строке.

Например, в макрокоманде **Применить фильтр** в качестве аргумента зададим условие отбора записей таблицы:

[Учащиеся]! [Пол] = Ложь.

Сохраним макрос с именем **Выбрать Девочки**. В результате получим макрос, состоящий из четырех макрокоманд.

Запуск макроса. При запуске макроса выполнение макрокоманд начинается с первой строки макроса и продолжается до конца макроса.

Выполнение макроса может начинаться по команде пользователя, при вызове из другого макроса или процедуры обработки события, а также в ответ на событие в форме, отчете или элементе управления. Например, можно связать макрос с кнопкой в форме или отчете, в результате чего макрос будет запускаться при нажатии кнопки. Допускается также создание специальной команды меню или кнопки панели инструментов, запускающей макрос, определение сочетания клавиш, нажатие которых запускает макрос, а также автоматический запуск макроса при открытии базы данных.

Запустим созданный макрос **Выбрать Девочки**, щелкнув кнопку **Выполнить** в ленте **Работа с макросами**. Макрос выполнится, и в окне Access будет раскрыта форма **Учащиеся** с отфильтрованными записями.

Связь между таблицами и целостность данных

До сих пор мы рассматривали операции с объектами на примере базы данных, состоящей из одной таблицы, но на практике, как правило, в составе базы данных имеется несколько таблиц.

Для координации таблиц между ними устанавливается связь. Связь между таблицами устанавливает отношения между совпадающими значениями

в ключевых полях. В большинстве случаев с ключевым полем одной таблицы, являющимся уникальным идентификатором каждой записи, связывается внешний ключ другой таблицы.

При установлении связи между таблицами возможны три типа связей.

Отношение «один-ко-многим». Это наиболее часто используемый тип связи между таблицами. В отношении «один-ко-многим» каждой записи в таблице А могут соответствовать несколько записей в таблице В, но запись в таблице В не может иметь более одной соответствующей ей записи в таблице А.

Отношение «многие-ко-многим». При этом типе связи одной записи в таблице А могут соответствовать несколько записей в таблице В, а одной записи в таблице В — несколько записей в таблице А. Этот тип связи возможен только с помощью третьей (связующей) таблицы, первичный ключ которой состоит из двух полей, являющихся внешними ключами таблиц А и В. Отношение «многие-ко-многим» по сути дела представляет собой два отношения «один-ко-многим» с третьей таблицей. Например, отношение «многие-ко-многим» между таблицами «Заказы» и «Товары» определяется путем создания двух отношений «один-ко-многим» с таблицей «Заказано».

Отношение «один-к-одному». При отношении «один-к-одному» запись в таблице А может иметь не более одной связанной записи в таблице В и наоборот. Отношения этого типа используются не очень часто, поскольку большая часть сведений, связанных таким образом, может быть помещена в одну таблицу. Отношение «один-к-одному» может использоваться для разделения очень широких таблиц, для отделения части таблицы по соображениям защиты, а также для сохранения сведений, относящихся к подмножеству записей в главной таблице.

Тип отношения в создаваемой Microsoft Access связи между таблицами зависит от способа определения связываемых полей.

Отношение «один-ко-многим» создается в том случае, когда только одно из полей является ключевым или имеет уникальный индекс.

Отношение «один-к-одному» создается в том случае, когда оба связываемых поля являются ключевыми или имеют уникальные индексы.

Отношение «многие-ко-многим» фактически является двумя отношениями «один-ко-многим» с третьей таблицей, первичный ключ которой состоит из полей — внешних ключей двух других таблиц.

Целостность данных

Для поддержания связей между записями в связанных таблицах, а также обеспечения защиты от случайного удаления или изменения связанных данных в Microsoft Access используется механизм **поддержки целостности данных**.

Целостность данных означает:

- в связанное поле **подчиненной** таблицы можно вводить только те значения, которые имеются в связанном поле **главной** таблицы;
- из **главной** таблицы нельзя удалить запись, у которой значение свя-

занного поля совпадает хотя бы с одним значением того же поля в **подчиненной** таблице.

Установить целостность данных можно, если выполнены следующие условия:

1. Связанное поле главной таблицы является ключевым полем или имеет уникальный индекс. В большинстве случаев связывают первичный ключ одной таблицы с соответствующим ему полем второй таблицы, которое называют полем внешнего ключа.

2. Связанные поля не обязательно должны иметь одинаковые имена, но они должны иметь одинаковые типы данных (из этого правила существуют два исключения) и иметь содержимое одного типа. Кроме того, связываемые поля числового типа должны иметь одинаковые значения свойства **Размер поля**.

Исключения. Поле счетчика можно связывать с числовым полем, свойство **Размер поля** которого имеет значение **Длинное целое**; также поле счетчика можно связать с числовым, если для обоих полей в свойстве **Размер поля** задано значение **Код репликации**.

3. Обе таблицы принадлежат одной базе данных Microsoft Access. Если таблицы являются связанными, то они должны быть таблицами Microsoft Access. Для установки целостности данных база данных, в которой находятся таблицы, должна быть открыта. Для связанных таблиц из баз данных других форматов установить целостность данных невозможно.

Установив целостность данных, необходимо следовать следующим правилам:

Невозможно ввести в поле внешнего ключа связанной таблицы значение, не содержащееся в ключевом поле главной таблицы. Однако в поле внешнего ключа возможен ввод значений **Null**, показывающих, что записи не являются связанными. Например, нельзя сохранить запись, регистрирующую заказ, сделанный несуществующим клиентом, но можно создать запись для заказа, который пока не отнесен ни к одному из клиентов, если ввести значение **Null** в поле «КодКлиента».

Не допускается удаление записи из главной таблицы, если существуют связанные с ней записи в подчиненной таблице. Например, невозможно удалить запись из таблицы «Сотрудники», если в таблице «Заказы» имеются заказы, относящиеся к данному сотруднику.

Невозможно изменить значение первичного ключа в главной таблице, если существуют записи, связанные с данной записью. Например, невозможно изменить код сотрудника в таблице «Сотрудники», если в таблице «Заказы» имеются заказы, относящиеся к этому сотруднику.

Чтобы наложить эти правила на конкретную связь, при ее создании следует установить флажок **Обеспечение целостности данных**. Если данный флажок установлен, то любая попытка выполнить действие, нарушающее одно из перечисленных выше правил, приведет к выводу на экран предупреждения, а само действие выполнено не будет.

Чтобы преодолеть ограничения на удаление или изменение связанных

записей, сохраняя при этом целостность данных, следует установить флажки **Каскадное обновление связанных полей** и **Каскадное удаление связанных полей**.

Режимы каскадного обновления и каскадного удаления. Для отношений, в которых проверяется целостность данных, пользователь имеет возможность указать, следует ли автоматически выполнять для связанных записей операции каскадного обновления и каскадного удаления. Если включить данные параметры, станут возможными операции удаления и обновления, которые иначе запрещены условиями целостности данных. Чтобы обеспечить целостность данных при удалении записей или изменении значения первичного ключа в главной таблице, автоматически вносятся необходимые изменения в связанные таблицы.

Если при определении отношения установить флажок **Каскадное обновление связанных полей**, любое изменение значения первичного ключа главной таблицы приведет к автоматическому обновлению соответствующих значений во всех связанных записях. Например, при изменении кода клиента в таблице «Клиенты» будет автоматически обновлено поле «КодКлиента» во всех записях таблицы «Заказы» для заказов каждого клиента, поэтому целостность данных не будет нарушена. Microsoft Access выполнит каскадное обновление без ввода предупреждающих сообщений.

Примечание. Если в главной таблице ключевым полем является поле счетчика, то установка флажка **Каскадное обновление связанных полей** не приведет к каким-либо результатам, так как изменить значение поля счетчика невозможно.

Если при определении отношения установить флажок **Каскадное удаление связанных записей**, любое удаление записи в главной таблице приведет к автоматическому удалению связанных записей в подчиненной таблице. Например, при удалении из таблицы «Клиенты» записи конкретного клиента будут автоматически удалены все связанные записи в таблице «Заказы» (а также записи в таблице «Заказано», связанные с записями в таблице «Заказы»). Если записи удаляются из формы или таблицы при установленном флажке **Каскадное удаление связанных записей**, Microsoft Access выводит предупреждение о возможности удаления связанных записей. Если же записи удаляются с помощью запроса на удаление, то Microsoft Access удаляет записи автоматически без вывода предупреждения.

Определение связей между таблицами

Связь между таблицами определяется путем добавления связываемых таблиц в окно «Схема данных» с последующим перетаскиванием ключевого поля из одной таблицы в другую. Создавать или изменять связи между открытыми таблицами нельзя. Для определения связей между таблицами закройте все открытые таблицы.

Создание связей между таблицами. Установление связей между таблицами рассмотрим на конкретном примере — базе данных **Группа**. В таблицу **Учащиеся** добавим поле **Книга** и создадим таблицу **Книги**.

Имя поля	Тип данных	Размер поля, формат	Примечание
----------	------------	---------------------	------------

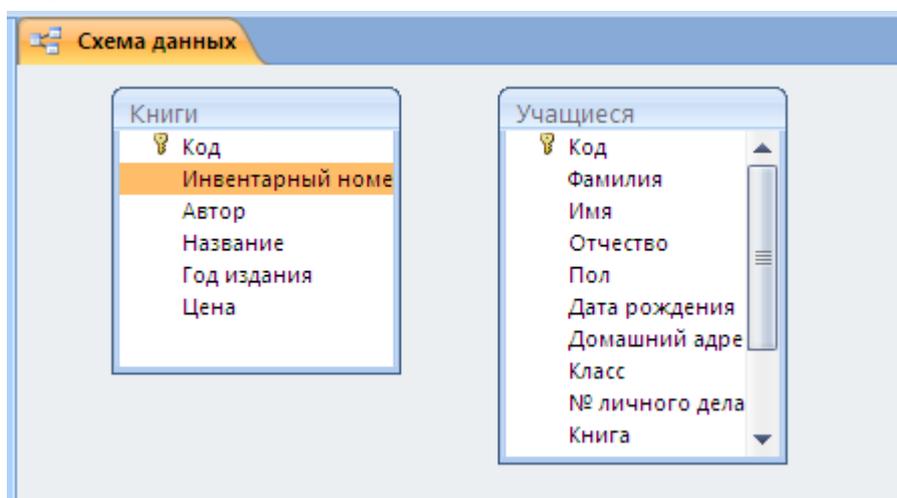
Таблица **Книги**

Код	Счетчик	Длинное целое	
Инвентарный номер	Числовой	Длинное целое	
Автор	Текстовый	30 символов	
Название	Текстовый	50 символов	
Год издания	Числовой	Целое число	
Цена	Числовой	Одинарное с плавающей точкой	

Таблица **Учащиеся**

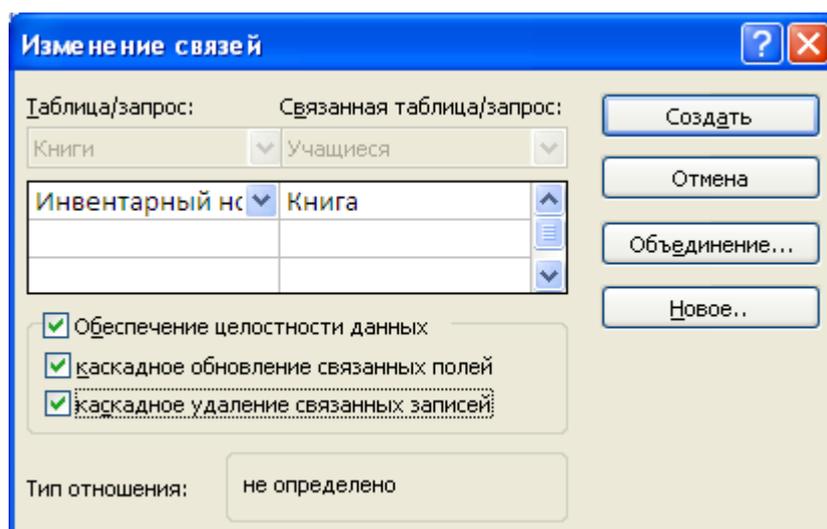
Книга	Числовой	Длинное целое	
--------------	----------	---------------	--

Установим связь между таблицами **Книги** и **Читатели**. Выберем команду **Схема данных** в ленте **Работа с базами данных**. После этого раскроется пустое окно **Схема данных**. Выбирая из списка всех таблиц открытой базы данных **Группа** и щелкая кнопку **Добавить**, добавим в окно схемы данных таблицы **Книги** и **Учащиеся**. Закроем окно **Добавление таблицы**, щелкнув кнопку **Заккрыть**.



Как показано на рисунке, после этого в окне **Схема данных** будут представлены две выбранных нами таблицы, между которыми устанавливается связь.

Для установления связи между двумя таблицами можно методом «Drag-and-Drop» переместить имя поля *главной* таблицы (**Инвентарный номер**) на поле **Книга** *подчиненной* таблицы.

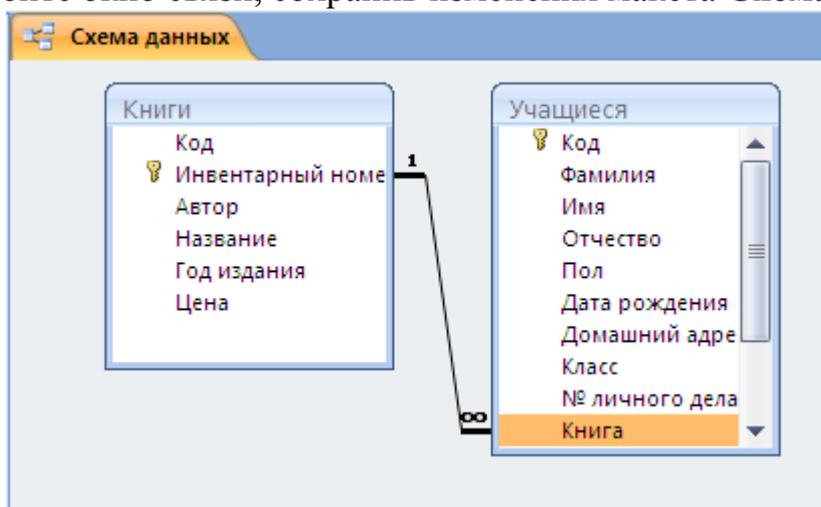


Как только вы отпустите левую кнопку мыши, на экране появится диалоговое окно **Изменение связей**. Для включения механизма поддержки целостности данных в связываемых таблицах установите флажок **Обеспечение целостности данных**.

После активизации флажка **Обеспечение целостности данных** становятся доступными два флажка **каскадных** операций. Включим переключатели каскадной модификации — обновления и удаления связанных записей.

В группе **Тип отношений** выберем **один-ко-многим**. Завершим создание связи, щелкнув кнопку **Создать**. Как показано на рисунке, в окне **Схема данных** появится графическое изображение установленной связи. Пометки у концов линии связи **1—∞** означают, что *одна* запись таблицы **Книги** может иметь *сколько угодно* связанных записей в таблице **Учащиеся**.

Закройте окно связи, сохранив изменения макета **Схема данных**.



Изменение или удаление существующих связей. Изменять связи между открытыми таблицами нельзя. Закройте все открытые таблицы. Переключитесь в окно базы данных и нажмите кнопку **Схема данных** на панели инструментов. Если таблиц, связи которых нужно изменить, нет на экране, нажмите кнопку **Отобразить таблицу** на панели инструментов и дважды щелкните таблицы, которые нужно добавить. Дважды щелкните линию связи, которую необходимо изменить, и установите параметры связи.

Для удаления выделите линию связи, которую необходимо удалить (выделенная линия становится более толстой), а затем нажмите клавишу **Delete** и подтвердите удаление.

Удаление таблицы из окна «Схема данных». Выберите таблицу, которую необходимо удалить, а затем нажмите клавишу **Delete**. При этом из окна **Схема данных** удаляется таблица и ее линии связи. Это действие затрагивает только отображение в окне схемы данных. Сама таблица и ее связи остаются в базе данных.

Создание объектов (форм, отчетов, макросов) для связанных таблиц отличается от ранее рассмотренных нами приемов тем, что пользователю необходимо выполнять выбор полей из нескольких таблиц и учитывать действие механизма защиты целостности данных.

Создание формы для связанных таблиц

Выберем создание формы с помощью мастера (Создание ⇒ **Формы** ⇒ **Другие формы** ⇒ **Мастер форм**).

На первом шаге диалога **Мастера форм** выберем таблицы **Книги**, а затем и **Учащиеся**, включим в форму все поля таблицы **Книги**, а также все поля таблицы **Учащиеся**, кроме поля **Книга** (это поле дублирует поле **Инвентарный номер** таблицы **Книги**), и щелкнем кнопку **Далее**.

На следующем шаге диалога с мастером выберем вид представления

Код	
Инвентарный номер	1818
Автор	Крылов И.А.
Название	Басни
Год издания	2003
Цена	110

Код	Фамилия	Имя	Отчество	Пол	Дата рождения	Домашний адрес	Класс	№ личного дела	Фотография
1	Ушаков	Дмитрий	Александрович	<input checked="" type="checkbox"/>	8 ноября 1998 г.	ул. Зеленая, 33-66	5в	У-55	

данных **Подчиненные формы**. Щелкнув кнопку **Далее**, выберем внешний вид подчиненной формы **Стандартная**. На следующих этапах диалога с **Мастером форм** зададим имя для каждой из связанных форм, выберем в качестве дальнейших действий вариант **Открыть форму для просмотра и ввода данных**. Завершим создание форм, щелкнув кнопку **Готово**.

Для запуска щелкнем ярлычок главной формы **Книги**. После этого на экране раскроется окно формы **Книги** с подчиненной формой **Учащиеся**.

Как видно на рисунке, у главной и подчиненной форм имеются отдельные строки перехода по записям. При переходе в главной форме к другой книге в подчиненной форме выводятся данные о читателях этой книги. Если в такой форме ввести данные о книге и читателе, то они будут записаны в соответствующие таблицы.

Можно изменить форму, разработанную с помощью **Мастера форм**. Для этого закроем форму и, указав главную форму **Книги**, щелкнем кнопку **Конструктор**. В режиме конструктора переместим и изменим размеры элементов управления, их свойства. Изменим надписи и другие параметры макета формы. Завершим редактирование, выбрав команду **Режим формы**.

Создание базы данных, операции с таблицами

Запустим Access и создадим базу данных **Автоматин**, состоящую из одной таблицы **Склад**.

Структура базы данных

Имя поля	Тип данных	Размер поля, формат
Код	Счетчик	Длинное целое
Марка	Текстовый	30 символов
Объем двигателя	Числовой	Одинарное с плавающей точкой
Цвет	Текстовый	20 символов
Тип кузова	Текстовый	20 символов
Год выпуска	Числовой	Целое
Номер кузова	Текстовый	30 символов, ключевое поле

1. Запустим **Microsoft Access**, щелкнув кнопку **Пуск** и выбрав в меню пункт **Программы**, а затем команду **Microsoft Access**.

2. В диалоговом окне при старте Access выберем опцию — **Новая база данных**. В окне **Имя Файла** выберем свою папку и зададим имя базы данных **Автоматин**.

3. Вызвав справку Access, на вкладке **Оглавление** выбрать тему **Структура баз данных**. Изучить разделы справки: **Основные сведения о создании баз данных**, **Создание таблиц в базе данных**. Закрыть окно справки.

4. В ленте СУБД Access **Создание** выберем закладку **Таблицы**, в правой области закладки выберем команду **Конструктор таблиц**.

5. В режиме конструктора таблицы в столбце **Имя поля** введем имя **Код**. В столбце **Тип данных** зададим тип **Числовой**, перейдем в бланк **Свойства поля** в нижней части окна и зададим значения **Размер поля: Длинное целое**. Во второй строке в графу **Имя поля** введем имя **Марка**. В столбце **Тип данных** оставим тип **Текстовый**. В столбце **Описание** введем описание данных, которые будет содержать это поле, например, **марка автомобиля**. Текст описания будет выводиться в строке состояния при добавлении данных в поле, а также будет включен в описание объекта таблицы. Вводить описание не обязательно. Перейдем в бланк **Свойства поля** в нижней части окна и зададим значения **Размер поля: 30 символов**. Действуя аналогично, зададим названия, укажем тип и свойства данных для остальных полей.

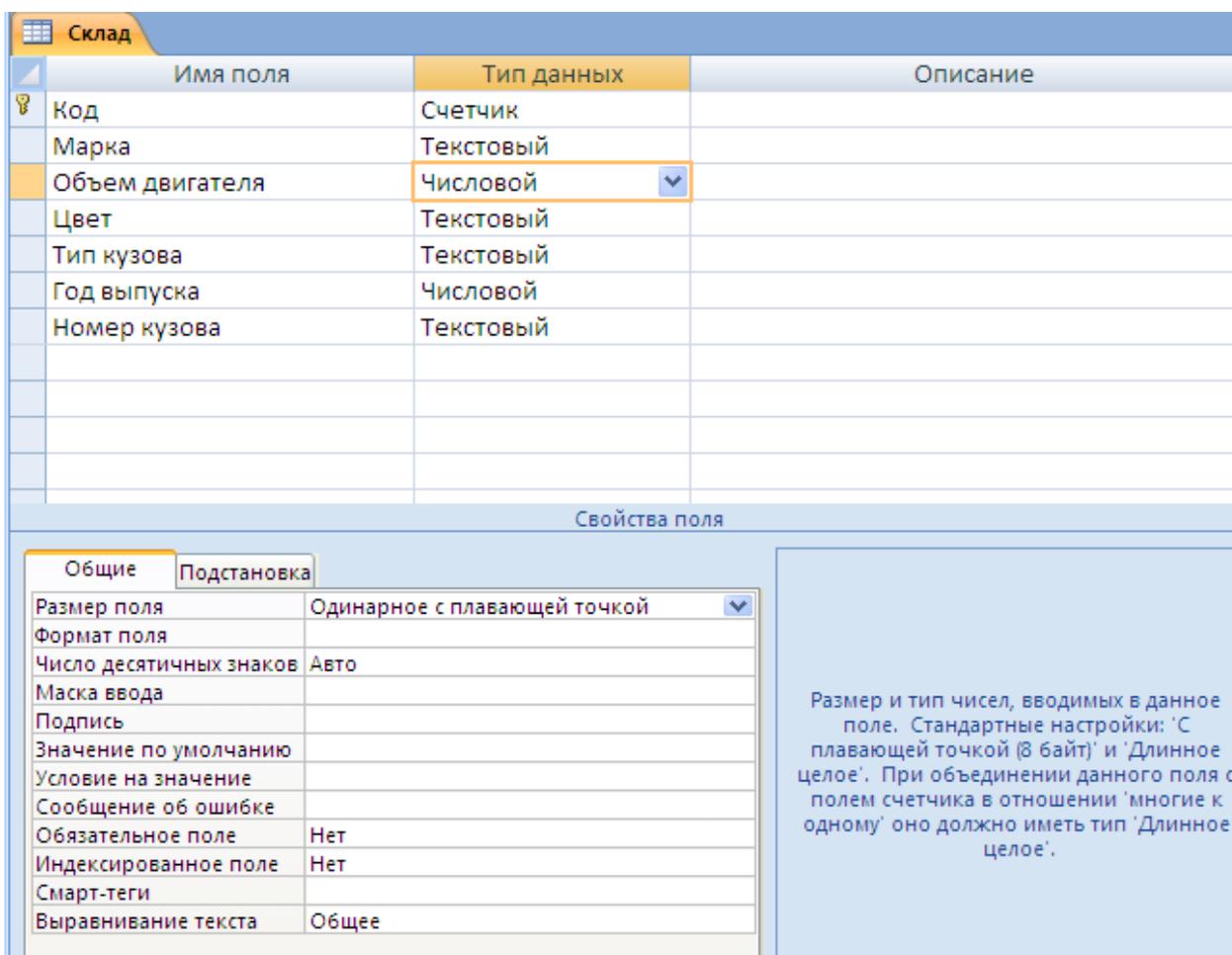
6. После ввода описания всех полей таблицы укажем ключевое поле, для чего щелкнув область выделения строки с записью поля **Код**, нажмем кнопку **Ключевое поле** на панели инструментов или в контекстном меню. После этого в области выделения поля **Код** появится знак ключевого поля — **пиктограмма ключ**.

7. Сохраним структуру таблицы командой **Сохранить**. В диалоговом окне **Сохранение** зададим имя таблицы **Склад** и щелкнем ОК для сохранения. Закроем окно конструктора таблицы. После этого в окне базы данных **Автоматин** на вкладке **Таблицы** появится новый объект — таблица **Склад**.

8. Открыть таблицу **Склад**, дважды щелкнув по ее имени в области пе-

реходов и ввести данные (для перехода к следующему полю нажимать клавишу **Tab**).

9. Открыть таблицу **Склад** и выполнить сортировку записей по объему двигателя в порядке убывания. Для этого, установив курсор в столбец **Объем двигателя**, щелкнуть кнопку **Сортировка по убыванию** в ленте **Главная** закладка **Сортировка и фильтр**.



Отсортировать записи по году выпуска в порядке возрастания, для чего, установив курсор в столбец **Год выпуска**, щелкнуть кнопку **Сортировка по возрастанию**.

Склад						
Код	Марка	Объем двигателя	Цвет	Тип кузова	Год выпуска	Номер кузова
1	ВАЗ-2115	1,6	желтый	седан	2006	FS6789B3
2	ГАЗ-31105	2,4	белый	седан	2008	GH6549KL
3	Лада Калина	1,6	желтый	хэтчбэк	2007	UP2093BN
4	Toyota Corolla Fielder	1,8	розовый	универсал	2005	LN8194h7
5	Audi A5	3,2	гранат	купе	2009	SE3671U9
6	Ford C-Max	1,8	металлик	минивэн	2006	WN6921K2
7	Nissan Note	1,4	голубой	хэтчбэк	2009	GF8788T2
8	Renault Clio	1,6	зеленый	хэтчбэк	2007	KV2873U5
9	Ока	1,2	сиреневый	лимузин	2008	JK8899JL
10	Ford Focus	1,8	ржавый	хэтчбэк	2008	SD7766FG

10. Используя фильтр, отобрать в таблице **Склад** записи об автомобилях с кузовом «седан». Для этого в поле **Тип кузова** найдем экземпляр значения «седан». Выделив это значение, щелкнем кнопку **Выделение** в ленте **Главная**. Для отмены фильтра щелкнем кнопку **Удалить фильтр**.

11. Используя расширенный фильтр, отобрать в таблице **Склад** записи об автомобилях с кузовом «седан», год выпуска которых старше 2007 г. Для этого выберем в закладке **Сортировка и Фильтр** — опцию **Расширенный фильтр**. После этого на экране будет раскрыт бланк создания расширенного фильтра.

Добавим в бланк поля **Тип кузова** и **Год выпуска**. Затем, установив курсор в строке **Условие отбора** в поле **Год выпуска** зададим условие отбора **[Склад]![Год выпуска]>2007** (В квадратных скобках вводится название таблицы и название поля). В этой же строке в поле **Тип кузова** зададим условие отбора «седан». Чтобы указать порядок сортировки, выберем ячейку **Сортировка** в поле **Год выпуска** и, щелкнув стрелку, выберем порядок сортировки **по возрастанию**. Чтобы применить фильтр, нажмем кнопку **Применение фильтра** в ленте или в контекстном меню. Для отмены фильтра щелкнем кнопку **Удалить фильтр** в ленте.

12. Закроем таблицу с сохранением и завершим работу СУБД MS Access.

Модификация базы данных. Использование связанных таблиц. Создание форм и отчетов

Создадим в базе данных **Автомагазин** таблицу **Поставщики**, в таблицу **Склад** добавим столбец **Поставщик** и создадим связь таблиц.

1. Загрузим программу Access и откроем созданную базу данных **Автомагазин**.

2. Откроем таблицу **Склад** в режиме конструктора, для чего в списке объектов базы данных **Автомагазин** откроем таблицу **Склад** и щелкнем кнопку **Конструктор** в закладке **Режимы**

3. Вставим в эту таблицу новое поле, для чего, выделив поле **Объем двигателя**, выберем в контекстном меню команду **Вставка строки**. Введем в новой строке следующее описание:

Имя поля	Тип данных	Размер, формат	Описание
Поставщик	Текстовый	30 символов	фирма-поставщик автомобиля

4. Сохраним изменения в структуре таблицы, для чего щелкнем кнопку **Сохранить** на панели быстрого запуска.

5. Создадим таблицу **Поставщики**, описав ее поля следующим образом:

Имя поля	Тип данных	Размер поля, формат поля	Описание
Код	Числовой	Длинное целое	
Фирма	Текстовый	30 символов,	Название фирмы
ФИО	Текстовый	50 символов	Фамилия имя отчество руководителя
Телефон	Текстовый	12 символов, маска ввода, (9999)-999-99-99	Номер телефона
Адрес	Текстовый	50 символов	Почтовый адрес

Для создания таблицы выберем в ленте **Создание** вкладку **Таблицы** и щелкнем кнопку **Создание таблицы в режиме конструктора**.

В режиме конструктора таблицы в столбце **Имя поля** введем имя **Фирма**. В столбце **Тип данных** оставим тип **Текстовый**. В столбце **Описание** введем описание данных, которые будет содержать это поле, например, **Название фирмы**. Перейдем в бланк **Свойства поля** в нижней части окна и зададим значения **Размер поля**: 30 символов, **Индексированное поле** – Да (Совпадения не допускаются). Действуя аналогично, зададим названия, укажем тип и свойства данных для остальных полей.

Для поля **Телефон** в бланке **Свойства поля** зададим маску ввода, которая обеспечит контроль ввода телефонного номера с кодом города, например, (8341)-256-75-98. Для этого введем в строке **Маска ввода** текст маски **(9999)-999-99-99**.

Поставщики				
Код	Фирма	Ф И О	Телефон	Адрес
1	ООО Бампер	Ушаков Дмитрий Александрович	(8341)-245-54-37	г. Ижевск ул. Зеленая, 33
2	ООО Кузовок	Окунев Олег Карпович	(8245)-365-88-44	г. Пермь ул. Красная, 67
3	ООО Руль и педали	Селезнев Федор Максимович	(8651)-284-33-11	г. Кукуев ул. Сосновая, 12
4	ООО По тормозам	Сидоров Василий Сидорович	(8332)-518-98-89	г. Мухино ул. Небесная, 13
5	Ездовые собаки и др транспорт	Окунев Олег Карпович	(3322)-556-66-21	Ижевск ул. Барабулькина 13

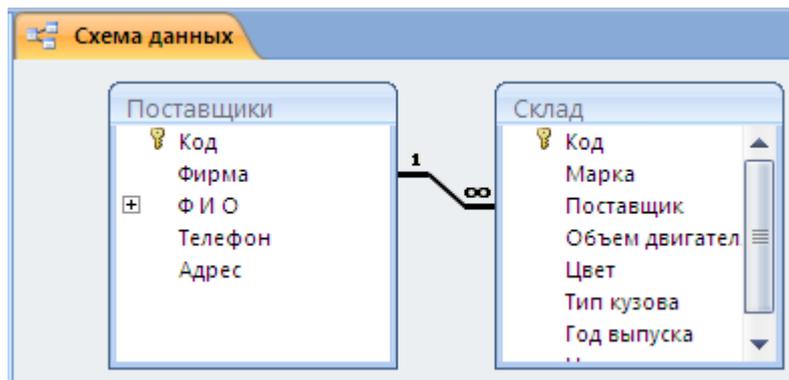
В качестве ключевого поля укажем поле **Код**, значения которого в таблице являются уникальными. Закроем таблицу **Поставщики** с сохранением структуры.

Введем в таблицу **Поставщики** четыре названия фирм и все остальные данные, а также заполним поле **Поставщик** в таблице **Склад**.

6. Установим связь между таблицами **Склад** и **Поставщики**. Для этого выберем команду **Схема данных** в ленте **Работа с базами данных**. После этого раскроется пустое окно **Схема данных**, и появится диалоговое окно **Добавить таблицу**. В диалоговом окне **Добавление таблицы** выберем вкладку **Таблицы**. Выбирая из списка таблиц открытой базы данных **Автомагазин** и щелкая кнопку **Добавить**, добавим в окно схемы данных таблицы

Склад и Поставщики. Закроем окно **Добавление таблицы**, щелкнув кнопку **Заккрыть**.

Для установления связи между двумя таблицами методом «Drag-and-Drop» переместим имя поля **главной** таблицы (**Фирма**) на поле **Поставщик** **подчиненной** таблицы. Как только вы отпустите левую кнопку мыши, на экране



появится диалоговое окно **Изменение связей**. Для включения механизма поддержки целостности данных в связываемых таблицах установите флажок **Обеспечение целостности данных**, а затем включим переключатели каскадной модификации — обновления и удаления связанных записей. Завершим создание связи, щелкнув кнопку **ОК**. В окне **Схема данных** появится графическое изображение установленной связи. Пометки у концов линии связи 1—∞ означают, что **одна** запись таблицы **Поставщики** может иметь **сколько угодно** связанных записей в таблице **Склад**.

7. Создадим форму для связанных таблиц. Для этого щелкнем в ленте **Создание** кнопку **Создание формы с помощью мастера**.

Код	Марка	Объем двигателя
1	ВАЗ-2115	1,6
6	Ford C-Max	1,8
7	Nissan Note	1,4
9	Ока	1,2
* (№)		

На первом шаге диалога мастера **Создание форм**, выбрав таблицы **По-**

ставщики, а затем и **Склад**, включим в форму все поля таблицы **Поставщики**, а также все поля таблицы **Склад**, кроме поля **Поставщик** (это поле дублирует поле **Фирма** таблицы **Поставщики**), и щелкнем кнопку **Далее**.

На следующем шаге диалога с мастером выберем вид представления данных, включив опцию **Подчиненные формы**. Щелкнув кнопку **Далее**, выберем внешний вид подчиненной формы — **табличный**, далее выберем стиль оформления **Стандартная**.

На следующих этапах диалога с мастером **Создание форм** зададим имя для каждой из связанных форм и выберем в качестве дальнейших действий вариант **Открыть форму для просмотра и ввода данных**. Завершим создание форм, щелкнув кнопку **Готово**. После этого на экране раскроется окно формы **Поставщики** с подчиненной формой **Склад**.

8. Если размер поля в форме мал для представления данных, закроем окно формы, укажем главную форму **Поставщики** и щелкнем кнопку **Конструктор** на панели инструментов. Изменим размеры элементов управления формы и закроем режим конструктора, сохранив изменения макета формы.

9. Введем новые данные в таблицу **Поставщики** и **Склад**. Закроем окно формы и, открыв таблицы **Поставщики** и **Склад**, просмотрим внесенные записи и убедимся, что в обеих таблицах внесены связанные записи.

10. Создадим отчет, для чего, выбрав в ленте **Создание** ⇒ **Отчеты**, команду **Мастер отчетов**. На первом шаге мастера **Создание отчетов**, выбрав таблицу **Поставщики**, включим в отчет поля **Фирма** и **Телефон**. Выбрав таблицу **Склад**, включим в отчет поля **Марка**, **Объем двигателя**, **Цвет**, **Тип кузова**, **Год выпуска**. Щелкнув кнопку **Далее**, выберем в качестве главной таблицы таблицу **Поставщики**. На следующем шаге диалога с мастером **Создание отчетов** добавим уровень группировки, выбрав поле **Фирма**. Щелкнув кнопку **Далее**, выберем сортировку по возрастанию по полю **Год**

Поставщики

Фирма	Телефон	Год выпуска	Марка	Объем двигателя	Цвет	Тип кузова
Ездовые собаки и др транспорт (3322)-556-66-21						
		2008	Ford Focus	1.8	ржавый	хэтчбэк
Итоги для 'Фирма' = Ездовые собаки и др транспорт (1 запись)						
Мах				1.8		
ООО Бампер (8341)-245-54-37						
		2006	Ford C-Max	1.8	металлик	минивэн
		2006	ВАЗ-2115	1.6	желтый	седан
		2008	Ока	1.2	сиреневый	лимузин
		2009	Nissan Note	1.4	голубой	хэтчбэк
Итоги для 'Фирма' = ООО Бампер (4 записей)						
Мах				1.8		
ООО Кузовок (8245)-365-88-44						
		2008	ГАЗ-31105	2.4	белый	седан
Итоги для 'Фирма' = ООО Кузовок (1 запись)						
Мах				2.4		
ООО По тормозам (8332)-518-98-89						

выпуска. Щелкнув кнопку **Итоги**, включим опцию **Мах** в поле **Объем двигателя**. Щелкнув кнопку **Далее**, выберем вид макета **ступенчатый** и включим опцию настройки ширины полей для размещения их на одной странице.

Затем выберем стиль оформления создаваемого отчета — **Официальная**. На заключительном этапе создания отчета зададим имя **Поставщики** и, выбрав **Изменить макет отчета**, настроим созданный отчет. Для завершения создания отчета и просмотра полученного отчета щелкнем кнопку **Готово**.

Работа с данными при помощи запросов

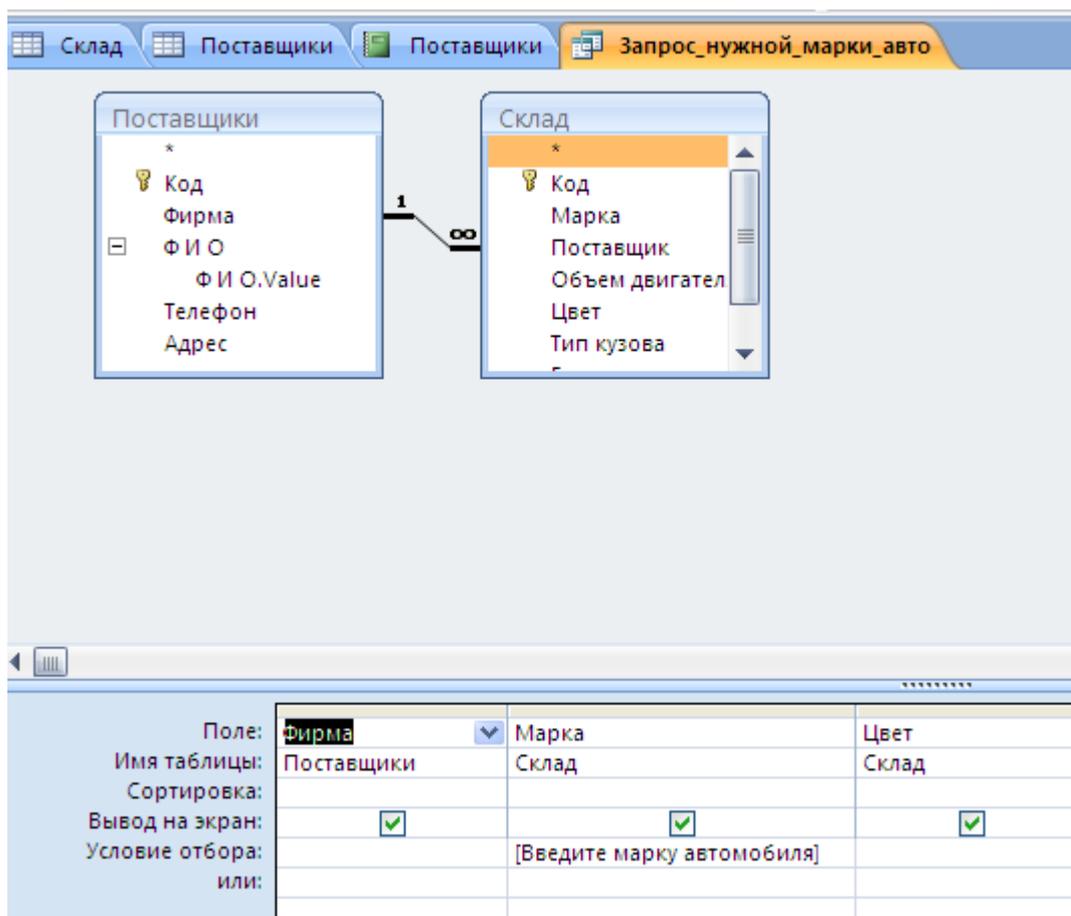
Создадим запрос к таблицам базы данных **Автомагазин**, который отберет данные об автомобилях, произведенных не ранее 2007 г. и поставленных одной из фирм, внесенных в таблицу, например ООО Бампер.

1. Выбрав вкладку **Другие**, в ленте **Создание** щелкнем кнопку **Конструктор запросов**.

2. В окне **Добавление таблицы**, выделяя таблицы **Поставщики**, а затем — **Склад** и щелкая кнопку **Добавить**, добавим обе таблицы базы данных **Автомагазин**, после чего закроем окно **Добавление таблицы**.

3. Перетаскивая поля из таблиц **Склад** и **Поставщики** в бланк запроса, определим поля таблиц для запроса и порядок их размещения. В строке **Вывод на экран** включим флажки отображения полей. В строке **Условие отбора** в столбце **Фирма** зададим условие отбора «ООО Бампер», а в столбце **Год выпуска** зададим условие отбора **> 2007**.

4. Перейдем в режим таблицы и посмотрим записи базы данных, отобранные согласно созданному запросу. Закроем окно запроса, сохранив макет запроса под именем **Запрос ООО Бампер не старше 2007**.



5. Создадим запрос с параметром **Поиск автомобилей по марке**. Для

4. Проверим действие макроса, для чего запустим его щелчком кнопки ! **Запуск.** После старта макроса на экране открывается окно ввода значения параметра с приглашением **Введите марку автомобиля.** Задав марку искомого автомобиля, просмотрим таблицу-результат действия вызванного макросом запроса. Если в таблице **Склад** такая марка есть, то данные о ней будут отображены в таблице. Если таких автомобилей нет, то таблица будет пуста.

5. Создадим макрос, который открывает отчет **Поставщики**, отбирает в нем данные об автомобилях какого-то поставщика и просматривает отчет.

6. Выбрав вкладку **Макросы**, выберем из списка в столбце **Макрокоманда** команду **ОткрытьОтчет.** В бланке **Аргументы макрокоманды** в поле **Имя отчета** зададим имя отчета **Поставщики.** В поле **Режим** зададим **Печать**, в поле **условие отбора** зададим **[Склад]![Поставщик]=[Введите название фирмы].**

7. Закроем окно конструктора, сохранив макрос под именем **Печать_отчета.** Проверим действие макроса, для чего запустим его щелчком кнопки ! **Выполнить.** После этого будет сформирован отчет, в который будут включены записи, отобранные из базы данных по заданному условию.

8. Закроем окно Access.

Контрольные вопросы

1. В чем недостатки текстового файла как базы данных?
2. Что такое структурирование информации?
3. Что такое база данных? В чем преимущества использования баз данных для организации данных?
4. Какими свойствами обладает реляционная таблица?
5. Чем отличаются поля и записи таблицы? Какие характеристики используются для описания полей баз данных?
6. Что такое «поле объекта OLE»?
7. Какое поле базы данных называют ключом?
8. Каково назначение и функции систем управления базами данных?
9. Какие типы могут принимать данные в информационных системах?
10. Чем отличается имя данного от значения данного?
11. Опишите возможности СУБД MS Access.
12. Какие объекты входят в состав файла базы данных MS Access?
13. Какие ограничения на имена полей, элементов управления и объектов действуют в MS Access?
14. Чем отличаются режимы работы с объектами базы данных в MS Access: режим таблицы, режим конструктора?
15. Опишите, какие типы данных могут иметь поля в MS Access? Каков их предельный размер?
16. Что такое выражения в MS Access? Какие бывают выражения и для чего они используются?
17. Какие особенности в записи различных операндов выражений: имя поля, число, текст?
18. Каково назначение построителя выражений?
19. С какой целью выполняется проектирование базы данных и в чем оно заключается?
20. Каково назначение сортировки данных в таблице? Какие бывают виды сортировки?
21. Что такое фильтр? Каковы особенности расширенного фильтра?
22. Зачем в базах данных используются формы? Какие разделы имеются в форме и для чего они предназначены? Какими способами можно создать форму?
23. Что такое запрос? Каково отличие запроса-выборки и запроса с параметром? Какими способами можно создать запрос?
24. Опишите назначение языка SQL.
25. Для чего нужен отчет? Какие сведения отображаются в отчете? Какова структура отчета? Какими способами можно создать отчет?
26. Какие средства используются в СУБД Microsoft Access для целей автоматизации операций с объектами баз данных? Чем они отличаются?
27. Зачем устанавливается связь между таблицами? Какие типы связей между таблицами возможны?
28. Зачем для связанных таблиц используется механизм поддержки целостности данных? В чем заключается его действие?

Задания

1 а. Создать реляционную базу данных Склад со структурой, позволяющей создать запрос:

Наименование материала	Количество поступления	Цена	Дата поступления	Дата выдачи	Остаток выдачи	Сумма

Остаток выдачи и Сумма должны быть вычисляемыми полями.

Заполнить созданную базу данных информацией о нескольких товарах. Отсортировать записи в данной базе данных по возрастанию даты поступления, по убыванию количества материала, по алфавиту наименований. Найти в базе данных Склад: материалы, в названии которых присутствует слово «трубы»; материалы, цена которых больше 2000,00р. и дата выдачи которых равна заданному значению в поле Дата.

1 б. Создать форму для вывода, просмотра данных в виде следующего бланка:

Склад материалов					
Наименование	<input type="text"/>				
Количество	<input type="text"/>	Цена	<input type="text"/>	Дата поступления	<input type="text"/>
Остаток	<input type="text"/>	Сумма	<input type="text"/>		

Просмотреть информацию базы данных «склад», используя созданную форму.

1 в. Создать отчет Ведомость инвентаризации склада в виде бланка следующей формы:

Наименование материала	Количество	Цена	Остаток	Сумма

2. Разработать подробный проект информационной системы Видеопрокат, включающей связанные таблицы Видеофильмы и Клиенты. Разработать форму ввода/ просмотра данных, запрос, выбирающий из базы данных клиентов, не возвративших видеофильм по истечении срока проката.

Оглавление

Введение	3
Базы данных	4
Реляционные базы данных	4
Что такое Microsoft Access?.....	6
Технология работы с MS Access	7
Основные понятия MS Access. Объекты MS Access	8
Технология создания базы данных в Access	10
Операции с данными в таблице	13
Создание и использование формы.....	16
Создание и использование запроса	20
Создание и использование отчета	24
Автоматизация выполнения задач обработки данных с помощью макрокоманд	26
Создание макроса	27
Связь между таблицами и целостность данных.....	28
Целостность данных.....	29
Определение связей между таблицами	31
Создание формы для связанных таблиц	34
Создание базы данных, операции с таблицами.....	35
Модификация базы данных. Использование связанных таблиц. Создание форм и отчетов.....	37
Работа с данными при помощи запросов.....	41
Использование макросов	42