Министерство образования РФ

Пермский государственный технический университет

Информационные технологи управления

**Базы данных и системы управления базами данных**

Электронное методическое пособие

для студентов заочного отделения

спец. ГМУ, ПГТУ

Составил доц. каф. ВМ и М О.Ю.Сметанников[[1]](#footnote-1)

ПЕРМЬ, 2002

СОДЕРЖАНИЕ

[1. Основные понятия баз данных. 3](#_Toc31273877)

[Базы данных и системы управления базами данных 3](#_Toc31273878)

[Структура простейшей базы данных 3](#_Toc31273879)

[Свойства полей базы данных 4](#_Toc31273880)

[Типы данных 5](#_Toc31273881)

[Безопасность баз данных 6](#_Toc31273882)

[2. Проектирование баз данных 6](#_Toc31273883)

[Режимы работы с базами данных 6](#_Toc31273884)

[Объекты базы данных 7](#_Toc31273885)

[Проектирование базы данных 10](#_Toc31273886)

[3. Работа с СУБД Access 2000 13](#_Toc31273887)

[Общие замечания 13](#_Toc31273888)

[Работа с таблицами 13](#_Toc31273889)

[Работа с запросами 17](#_Toc31273890)

[Работа с формами 19](#_Toc31273891)

[Работа со страницами доступа к данным 22](#_Toc31273892)

[Работа с отчетами. 23](#_Toc31273893)

# 1. Основные понятия баз данных.

## Базы данных и системы управления базами данных

*База данных — это организованная структура, предназначенная для хранения инфор­мации.* Внимательный читатель, знающий из первой главы этого пособия о том, что данные и информация — понятия взаимосвязанные, но не тождественные, дол­жен заметить некоторое несоответствие в этом определении. Его причины чисто исторические. В те годы, когда формировалось понятие баз данных, в них действи­тельно хранились только данные. Однако сегодня большинство *систем управле­ния базами данных (СУБД)* позволяют размещать в своих структурах не только данные, но и методы (то есть программный код), с помощью которых происходит взаимодействие с потребителем или с другими программно-аппаратными комплек­сами. *Таким образом, мы можем говорить, что в современных базах данных хра­нятся отнюдь не только данные, но и информация.*

Это утверждение легко пояснить, если, например, рассмотреть базу данных круп­ного банка. В ней есть все необходимые сведения о клиентах, об их адресах, кре­дитной истории, состоянии расчетных счетов, финансовых операциях и т. д. Доступ к этой базе имеется у достаточно большого количества сотрудников банка, но среди них вряд ли найдется такое лицо, которое имеет доступ *ко всей базе полностью* и при этом способно единолично вносить в нее произвольные изменения. Кроме дан­ных, база содержит *методы и средства,* позволяющие каждому из сотрудников оперировать только с теми данными, которые входят в его компетенцию. В результате взаимодействия данных, содержащихся в базе, с методами, доступными конкрет­ным сотрудникам, образуется информация, которую они потребляют и на основа­нии которой в пределах собственной компетенции производят ввод и редактиро­вание данных.

С понятием *базы данных* тесно связано понятие *системы управления базой данных.* Это комплекс программных средств, предназначенных для создания структуры новой базы, наполнения ее содержимым, редактирования содержимого и визуализации информации. Под *визуализацией информации* базы понимается отбор отображае­мых данных в соответствии с заданным критерием, их упорядочение, оформление и последующая выдача на устройство вывода или передача по каналам связи.

В мире существует множество систем управления базами данных. Несмотря на то что они могут по-разному работать с разными объектами и предоставляют пользова­телю различные функции и средства, большинство СУБД опираются на единый устоявшийся комплекс основных понятий. Это дает нам возможность рассмотреть одну систему и обобщить ее понятия, приемы и методы на весь класс *СУБД.* В качестве такого учебного объекта мы выберем *СУБД* Microsoft Access, входящую в пакет Microsoft Office наряду с рассмотренными ранее пакетами Microsoft Word и Microsoft Excel. В тех случаях, когда конкретные приемы операций зависят от используемой версии программы, мы будем опираться на последнюю версию Microsoft Access 2000, хотя в основном речь будет идти о таких обобщенных понятиях и методах, для которых различия между конкретными версиями программ второстепенны.

## Структура простейшей базы данных

Сразу поясним, что если в базе нет никаких данных *(пустая база),* то это все равно полноценная база данных. Этот факт имеет методическое значение. Хотя данных в базе и нет, но информация в ней все-таки есть — это *структура базы.* Она определяет методы занесения данных и хранения их в базе. Простейший вне компьютерный» вариант базы данных — деловой ежедневник, в котором каждому календарному дню выделено по странице! Даже если в нем не записано ни строки» он не перестает быть ежедневником, поскольку имеет структуру, четко отличающую его от записных книжек, рабочих тетрадей и прочей писчебумажной продукции,

Базы данных могут содержать различные объекты, но, забегая вперед, скажем, что основными объектами любой базы данных являются ее таблицы. Простейшая база данных имеет хотя бы одну таблицу. Соответственно, структура простейшей базы данных тождественно равна структуре ее таблицы.

Мы знаем, что структуру двумерной таблицы образуют столбцы и строки. Их анало­гами в структуре простейшей базы данных являются *поля* и *записи.* Если записей в таблице пока нет, значит, ее структура образована только набором полей. Изменив состав полей базовой таблицы (или их свойства), мы изменяем структуру базы данных и, соответственно, получаем новую базу данных.

## Свойства полей базы данных

Поля базы данных не просто определяют структуру базы — они еще определяют групповые свойства данных, записываемых в ячейки, принадлежащие каждому из полей. Ниже перечислены основные свойства полей таблиц баз данных на примере СУБД Microsoft Access.

• Имя поля — определяет, как следует обращаться к данным этого поля при авто­матических операциях с базой (по умолчанию имена полей используются в качестве заголовков столбцов таблиц).

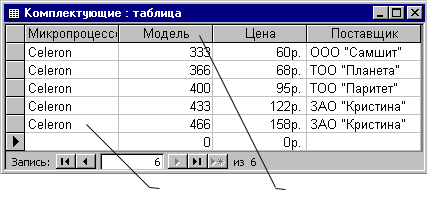
• Тип поля — определяет тип данных, которые могут содержаться в данном поле.

• Размер поля — определяет предельную длину (в символах) данных, которые могут размещаться в данном поле.

Рис. 1. Простейшая таблица базы данных.

Поля базы

Записи базы



• Формат поля — определяет способ форматирования данных в ячейках, принад­лежащих полю.

• Маска ввода — определяет форму, в которой вводятся данные в поле (средство автоматизации ввода данных).

• Подпись — определяет заголовок столбца таблицы для данного поля (если подпись не указана, то в качестве заголовка столбца используется свойство Имя поля).

• Значение по умолчанию — то значение, которое вводится в ячейки поля авто­матически (средство автоматизации ввода данных).

• Условие на значение — ограничение, используемое для проверки правильности ввода данных (средство автоматизации ввода, которое используется, как пра­вило, для данных, имеющих числовой тип, денежный тип или тип даты).

• Сообщение об ошибке — текстовое сообщение, которое выдается автоматически при попытке ввода в поле ошибочных данных (проверка ошибочности выпол­няется автоматически, если задано свойство Условие на значение).

• Обязательное поле — свойство, определяющее обязательность заполнения дан­ного поля при наполнении базы;

• Пустые строки — свойство, разрешающее ввод пустых строковых данных (от свойства Обязательное поле отличается тем, что относится не ко всем типам данных, а лишь к некоторым, например к текстовым).

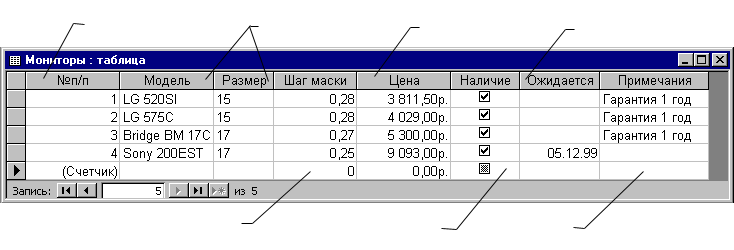
• Индексированное поле — если поле обладает этим свойством, все операции, связанные с поиском или сортировкой записей по значению, хранящемуся в данном поле, существенно ускоряются. Кроме того, для индексированных полей можно сделать так, что значения в записях будут проверяться по этому полю на наличие повторов, что позволяет автоматически исключить дублирование данных.

Здесь мы должны обратить особое внимание читателя на то, что поскольку в разных полях могут содержаться данные разного типа, то и свойства у полей могут разли­чаться в зависимости от типа данных. Так, например, список вышеуказанных свойств полей относится в основном к полям текстового типа. Поля других типов могут иметь или не иметь эти свойства, но могут добавлять к ним и свои. Например для данных, представляющих действительные числа, важным свойством является количество знаков после десятичной запятой. С другой стороны, для полей, исполь­зуемых для хранения рисунков, звукозаписей, видеоклипов и других объектов *OLE,* большинство вышеуказанных свойств не имеют смысла.

## Типы данных

С основными типами данных мы уже знакомы. Так, например, при изучении электрон­ных таблиц Microsoft Excel мы видели, что они работают с тремя типами данных: текстами, числами и формулами. Таблицы баз данных, как правило, допускают работу с гораздо большим количеством разных типов данных. Так, например, базы данных Microsoft Access работают со следующими типами данных (рис. 13.2)

.



Поле счетчика Текстовые поля Поле денежного типа Поле даты

Числовое поле Логическое поле Поле МЕМО

Рис. 2. Таблица с полями некоторых типов



*•* Текстовый — тип данных, используемый для хранения обычного неформатиро­ванного текста ограниченного размера (до 255 символов).

• Поле Мемо — специальный тип данных для хранения больших объемов текста (до 65 535 символов). Физически текст не хранится в поле. Он хранится в другом месте базы данных, а в поле хранится указатель на него, но для пользователя такое разделение заметно не всегда.

• Числовой — тип данных для хранения действительных чисел.

• Дата/время — тип данных для хранения календарных дат и текущего времени.

• Денежный — тип данных для хранения денежных сумм. Теоретически, для их записи можно было бы пользоваться и полями числового типа, но для денежных сумм есть некоторые особенности (например, связанные с правилами округления), которые делают более удобным использование специального типа данных, а не настройку числового типа.

• Счетчик — специальный тип данных для уникальных (не повторяющихся в поле) натуральных чисел с автоматическим наращиванием. Естественное использо­вание — для порядковой нумерации записей.

• Логический — тип для хранения логических данных (могут принимать только два значения, например Да или Нет).

• Поле объекта OLE — специальный тип данных, предназначенный для хранения объектов *OLE,* например мультимедийных. Реально, конечно, такие объекты в таблице не хранятся. Как и в случае полей MEMO, они хранятся в другом месте внутренней структуры файла базы данных, а в таблице хранятся только указа­тели на них (иначе работа с таблицами была бы чрезвычайно замедленной).

• Гиперссылка — специальное поле для хранения адресов *URL* Web-объектов Интернета. При щелчке на ссылке автоматически происходит запуск броузера и воспроизведение объекта в его окне.

• Мастер подстановок — это не специальный тип данных. Это объект, настройкой которого можно автоматизировать ввод в данных поле так, чтобы не вводить их вручную, а выбирать из раскрывающегося списка.

## Безопасность баз данных

Базы данных — это тоже файлы, но работа с ними отличается от работы с файлами других типов, создаваемых прочими приложениями. Выше мы видели, что всю работу по обслуживанию файловой структуры берет на себя операционная система. Для баз данных предъявляются особые требования с точки зрения безопасности, поэтому в них реализован другой подход к сохранению данных.

При работе с обычными приложениями для сохранения данных мы выдаем соответ­ствующую команду, задаем имя файла и доверяемся операционной системе. Если мы закроем файл, не сохранив его, то вся работа по созданию или редактированию файла пропадет безвозвратно.

Базы данных — это особые структуры. Информация, которая в них содержится, очень часто имеет общественную ценность. Нередко с одной и той же базой (напри­мер, с базой регистрации автомобилей в ГИБДД) работают тысячи людей по всей стране. От информации, которая содержится в некоторых базах, может зависеть благополучие множества людей. Поэтому целостность содержимого базы не может и не должна зависеть ни от конкретных действий некоего пользователя, забывшего сохранить файл перед выключением компьютера, ни от перебоев в электросети.

Проблема безопасности баз данных решается тем, что в СУБД для сохранения информации используется двойной подход. В части операций, как обычно, участвует операционная система компьютера, но некоторые операции сохранения происходят в обход операционной системы.

Операции изменения структуры базы данных, создания новых таблиц или иных объектов происходят при сохранении файла базы данных. Об этих операциях СУБД предупреждает пользователя. Это, так сказать, глобальные операции. Их никогда не проводят с базой данных, находящейся в коммерческой эксплуатации, — только с ее копией. В этом случае любые сбои в работе вычислительных систем не страшны.

С другой стороны, операции по изменению содержания данных, не затрагивающие структуру базы, максимально автоматизированы и выполняются без предупреждения. Если работая с таблицей данных мы что-то в ней меняем в составе данных, то изменения сохраняются немедленно и автоматически.

Обычно, решив отказаться от изменений в документе, его просто закрывают без сохранения и вновь открывают предыдущую копию. Этот прием работает почти во всех приложениях, но только не в СУБД. Все изменения, вносимые в таблицы базы, сохраняются на диске без нашего ведома, поэтому попытка закрыть базу «без сохранения» ничего не даст, так как все уже сохранено. Таким образом, редактируя таблицы баз данных, создавая новые записи и удаляя старые, мы как бы работаем с жестким диском напрямую, минуя операционную систему.

Но по указанным выше причинам нельзя заниматься учебными экспериментами на базах данных, находящихся в эксплуатации. Для этого следует создавать специальные учебные базы или выполнять копии структуры реальных баз (без фактического наполнения данными).

# 2. Проектирование баз данных

## Режимы работы с базами данных

Обычно с базами данных работают две категории исполнителей. Первая катего­рия — *проектировщики.* Их задача состоит в разработке структуры таблиц базы данных и согласовании ее с заказчиком. Кроме таблиц проектировщики разраба­тывают и другие объекты базы данных, предназначенные, с одной стороны, для автоматизации работы с базой, а с другой стороны — для ограничения функциональ­ных возможностей работы с базой (если это необходимо из соображений безопас­ности). Проектировщики не наполняют базу конкретными данными (заказчик может считать их конфиденциальными и не предоставлять посторонним лицам). Исклю­чение составляет экспериментальное наполнение модельными данными на этапе отладки объектов базы.

Вторая категория исполнителей, работающих с базами данных, — *пользователи.* Они получают исходную базу данных от проектировщиков и занимаются ее наполне­нием и обслуживанием. В общем случае пользователи не имеют средств доступа к управлению структурой базы — только к данным, да и то не ко всем, а к тем, работа с которыми предусмотрена на конкретном рабочем месте.

Соответственно, система управления базами данных имеет два режима работы: *проектировочный* и *пользовательский.* Первый режим предназначен для создания или изменения структуры базы и создания ее объектов. Во втором режиме проис­ходит использование ранее подготовленных объектов для наполнения базы или получения данных из нее.

## Объекты базы данных

Мы уже упомянули о том, что кроме таблиц база данных может содержать и другие типы объектов. Привести полную классификацию возможных объектов баз данных затруднительно, поскольку каждая система управления базами данных может реали­зовать свои типы объектов. Однако основные типы объектов мы можем рассмотреть

на примере СУБД Microsoft Access. В версии Microsoft Access 2000 эта СУБД позво­ляет создавать и использовать объекты семи различных типов.

**Таблицы.** Как мы уже говорили, это основные объекты любой базы данных. Во-первых, в таблицах хранятся все данные, имеющиеся в базе, а во-вторых, таблицы хранят и структуру базы (поля, их типы и свойства).

**Запросы.** Эти объекты служат для извлечения данных из таблиц и предоставления их пользователю в удобном виде. С помощью запросов выполняют такие опера­ции как отбор данных, их сортировку и фильтрацию. С помощью запросов можно выполнять преобразование данных по заданному алгоритму, создавать новые таблицы, выполнять автоматическое наполнение таблиц данными, импортированными из других источников, выполнять простейшие вычисления в таблицах и многое другое.

Начинающие пользователи не сразу понимают роль запросов, поскольку все те же операции можно делать и с таблицами. Да, действительно, это так, но есть соображе­ния удобства (в первую очередь быстродействия) и соображения безопасности.

Из соображений безопасности, чем меньше доступа к базовым таблицам имеют конечные пользователи, тем лучше. Во-первых, снижается риск того, что неуме­лыми действиями они повредят данные в таблицах. Во-вторых, предоставив раз­ным пользователям разные запросы, можн о эффективно разграничить их доступ к данным в строгом соответствии с крутом персональных обязанностей. В банках, например, одни сотрудники имеют доступ к таблицам данных о клиентах, другие — к их расчетным счетам, третьи — к таблицам активов банка. Если и есть специаль­ные службы, имеющие доступ ко всем информационным ресурсам банка (с целью контроля и анализа), то они лишены средств для внесения изменений — все сделано так, чтобы один человек не мог совершить фиктивную операцию, независимо от того, какую должность он занимает. В базе данных, имеющей правильно организо­ванную структуру, для совершения противоправных действий необходим сговор нескольких участников, а такие действия пресекаются не программными, а традици­онными средствами обеспечения безопасности.

Особенность запросов состоит в том, что они черпают данные из базовых таблиц и создают на их основе *временную результирующую таблицу.* Если хотят подчеркнуть факт «временности» этой таблицы, то ее еще называют *моментальным снимком.* Когда мы работаем с основными таблицами базы, мы физически имеем дело с жест­ким диском, то есть с очень медленным устройством (напомним, что это связано с особенностью сохранения данных, описанной выше). Когда же на основании запроса мы получаем результирующую таблицу, то имеем дело с электронной таблицей, не имеющей аналога на жестком диске, — это только *образ* отобранных полей и запи­сей. Разумеется, работа с «образом» происходит гораздо быстрее и эффективнее — это еще одно основание для того, чтобы широко использовать запросы.

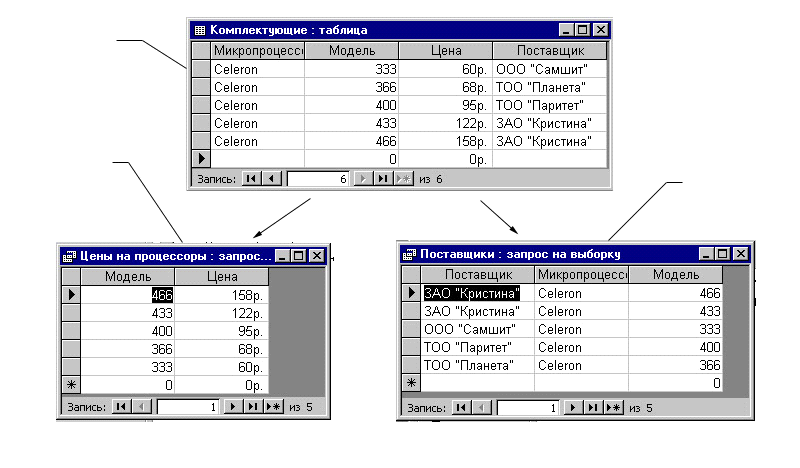
Когда в главе 1 мы обсуждали основные структуры данных, то отметили, что недо­статком упорядоченных табличных структур является сложность их обновления, поскольку при внесении новых записей нарушается упорядоченность — приходится переделывать всю таблицу. В системах управления базами данных и эта проблема решается благодаря запросам.

Основной принцип состоит в том, что от базовых таблиц никакой упорядоченности не требуется. Все записи в основные таблицы вносятся только в *естественном* поряд­ке по мере их поступления, то есть в неупорядоченном виде. Если же пользователю надо видеть данные, отсортированные или отфильтрованные по тому или иному принципу, он просто использует соответствующий запрос. Если нужного запроса нет, он обращается к проектировщику и просит его такой запрос сделать и предоставить.

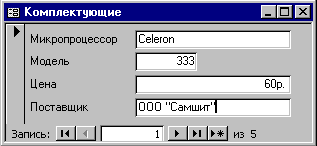
Результирующая таблица

Результирующая таблица

Базовая таблица



**Формы.** Если запросы — это специальные средства для отбора и анализа данных, то формы — это средства для ввода данных. Смысл их тот же — предоставить пользователю средства для заполнения только тех полей, которые ему заполнять положено.



Одновременно с этим в форме можно разместить специальные элементы управления (счетчики, раскрывающиеся списки, переключатели, флажки и прочие) для автоматизации ввода. Преимущества форм раскрываются особенно наглядно, когда происходит ввод данных с заполнен­ных бланков. В этом случае форму делают графическими средствами так, чтобы она повторяла оформление бланка - это замет­но упрощает работу наборщика, снижает его утомление и предотвращает появление печатных ошибок. На сопроводительном рисунке приведен пример простейшей формы для ввода данных.

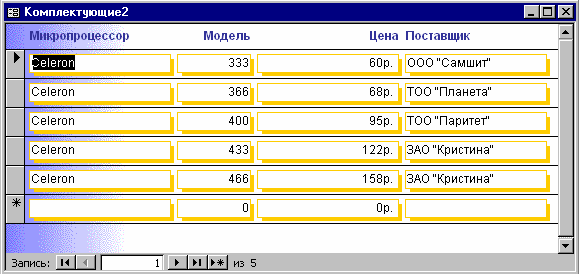


Рис. 4. Форма для оформленного вывода данных.

С помощью форм данные можно не только вводить, но и отображать. Запросы тоже отображают данные, но делают это в виде результирующей таблицы, не имеющей

почти никаких средств оформления. При выводе данных с помощью форм можно применять специальные средства оформления.

**Отчеты.** По своим свойствам и структуре отчеты во многом похожи на формы, но предназначены только для вывода данных, причем для вывода не на экран, а на печатающее устройство (принтер). В связи с этим отчеты отличаются тем, что в них приняты специальные меры для группирования выводимых данных и для вывода специальных элементов оформления, характерных для печатных докумен­тов (верхний и нижний колонтитулы, номера страниц, служебная информация о времени создания отчета и т. п.)

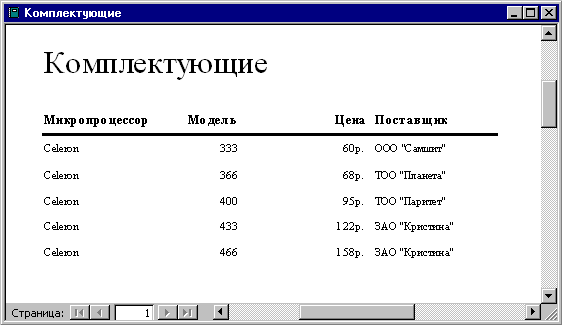


Рис. 5. Пример простейшего отчета.

**Страницы.** Это специальные объекты баз данных, реализованные в последней вер­сии СУБД Microsoft Access (Access 2000). Правда, более корректно их называть *страницами доступа к данным.* Физически это особый объект, выполненный в коде *HTML,* размещаемый на Web-странице и передаваемый клиенту вместе с ней. Сам по себе этот объект не является базой данных, но содержит компоненты, через кото­рые осуществляется связь переданной Web-страницы с базой данных, остающейся на сервере. Пользуясь этими компонентами, посетитель Web-узла может просматривать записи базы в полях страницы доступа. Таким образом, страницы доступа к данным осуществляют интерфейс между клиентом, сервером и базой данных, размещенной на сервере. Эта база данных не обязательно должна быть базой данных Microsoft Access. Страницы доступа, созданные средствами Microsoft Access, позволяют работать также с базами данных Microsoft SQL Server.

Панель управления просмотром

Поля вывода данных



Рис. 6. Пример простейшей страницы доступа.



Макросы и модули. Эти категории объектов предназначены как для автоматизации повторяющихся операций при работе с системой управления базами данных, так и для создания новых функций путем программирования. В СУБД Microsoft Access *макросы* состоят из последовательности внутренних команд СУБД и являются одним из средств автоматизации работы с базой. *Модули* создаются средствами внеш­него языка программирования, в данном случае языка Visual Basic for Applications. Это одно из средств, с помощью которых разработчик базы может заложить в нее нестандартные функциональные возможности, удовлетворить специфические требо­вания заказчика, повысить быстродействие системы управления, а также уровень ее защищенности.

## Проектирование базы данных

Мы лишь рассмотрели основные понятия баз данных и еще не знакомы ни с одной системой управления базами данных, но, тем не менее, поднимаем вопрос о проек­тировании базы. Это не случайно. Методически правильно начинать работу с каран­дашом и листом бумаги в руках, не используя компьютер. На данном этапе он просто не нужен. Неоптимальные решения и прямые ошибки, заложенные на этапе проек­тирования, впоследствии очень трудно устраняются, поэтому этот этап является основополагающим.

Разработка технического задания. Техническое задание на проектирование базы данных должен предоставить заказчик. Однако для этого он должен владеть соот­ветствующей терминологией и знать, хотя бы в общих чертах, технические воз­можности основных систем управления базами данных. К сожалению, на практике такое положение встречается не всегда. Поэтому обычно используют следующие подходы:

• демонстрируют заказчику работу аналогичной базы данных, после чего согласо­вывают спецификацию отличии;

• если аналога нет, выясняют крут задач и потребностей заказчика, после чего помогают ему подготовить техническое задание.

При подготовке технического задания составляют:

• список исходных данных, с которыми работает заказчик;

• список выходных данных, которые необходимы заказчику для управления структурой своего предприятия;

• список выходных данных, которые не являются необходимыми для заказчика, но которые он должен предоставлять в другие организации (в вышестоящие структуры, в органы статистического учета, прочие административные и контро­лирующие организации).

При этом очень важно не ограничиваться взаимодействием с головным подразделе­нием заказчика, а провести обсуждение со всеми службами и подразделениями, которые могут оказаться поставщиками данных в базу или их потребителями. Так, например, при подготовке базы данных для учета абитуриентов и студентов в высшем учебном заведении, необходимо не только изучить документооборот ректората и всех деканатов, но и понять, что хотели бы получить от базы данных службы. Сле­дует изучить работу подразделений, распределяющих учебную нагрузку преподава­телей, отвечающих за распределение аудиторного фонда, за проживание студентов в общежитии и других. В расчет должны приниматься и такие службы, как библио­тека, отдел кадров и прочие. В любой момент может выясниться, например, что администрация библиотеки должна периодически поставлять кому-то отчеты, харак­теризующие читательскую активность студентов в зависимости от пола, возраста и социального положения. К возможным пожеланиям заказчика следует готовиться на этапе проектирования, до создания базы.

Разработка структуры базы данных. Выяснив основную часть данных, которые заказчик потребляет или поставляет, можно приступать к созданию структуры базы, то есть структуры ее основных таблиц.

1. Работа начинается с составления генерального списка полей — он может насчи­тывать десятки и даже сотни позиций.

2. В соответствии с типом данных, размещаемых в каждом поле, определяют наиболее подходящий тип для каждого поля.

3. Далее распределяют поля генерального списка по базовым таблицам. На первом этапе распределение производят по функциональному признаку. Цель — обеспечить, чтобы ввод данных в одну таблицу производился, по возможности, в рамках одного подразделения, а еще лучше — на одном рабочем месте.

Наметив столько таблиц, сколько подразделений охватывает база данных, при­ступают к дальнейшему делению таблиц. Критерием необходимости деления является факт множественного повтора данных в соседних записях. На рис. показана таблица, у которой в поле Адрес наблюдается повтор данных. Это явное свидетельство того, что таблицу надо поделить на две взаимосвязанные таблицы.

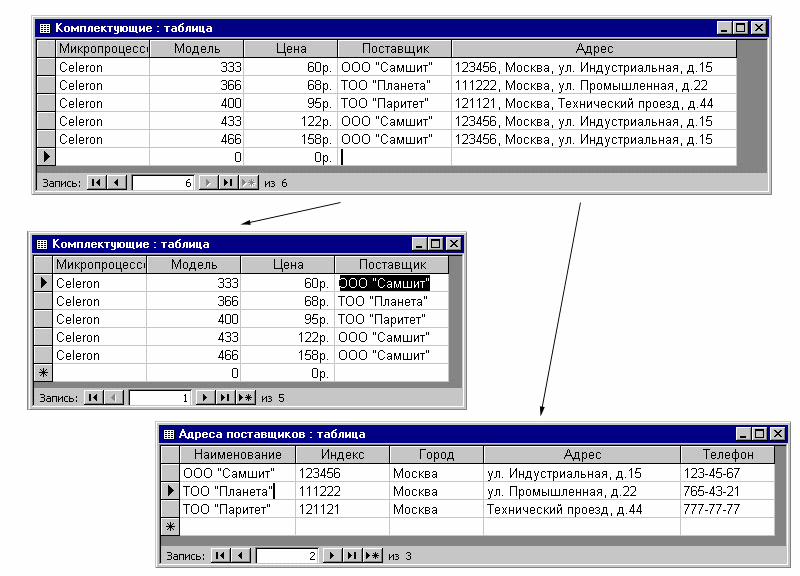


Рис. 7. Если данные в поле начинают повторяться, это признак того, что таблицу стоит поделить

4. В каждой из таблиц намечают *ключевое поле.* В качестве такового выбирают поле, данные в котором повторяться не могут. Например, для таблицы данных о студентах таким полем Может служить индивидуальный шифр студента. Для таблицы, в которой содержатся расписания занятий, такого поля можно и не найти, но его можно создать искусственным комбинированием полей «Время занятия» и «Номер аудитории». Эта комбинация неповторима, так как в одной аудитории в одно и то же время не принято проводить два различных занятия.

Если в таблице вообще нет никаких полей, которые можно было бы использовать как ключевые, всегда можно ввести дополнительное поле типа Счетчик — оно не может содержать повторяющихся данных по определению.

5. С помощью карандаш и бумаги расчерчивают связи между таблицами. На рис. 13.8 показан пример взаимосвязи между группой таблиц, составляющих одну базу данных. Такой чертеж называется *схемой данных.*

Существует несколько типов возможных связей между таблицами. Наиболее распространенными являются связи «один ко многим» и «один к одному». Связь между таблицами организуется на основе общего поля, причем в одной из таблиц оно обязательно должно быть ключевым, то есть на стороне «один»

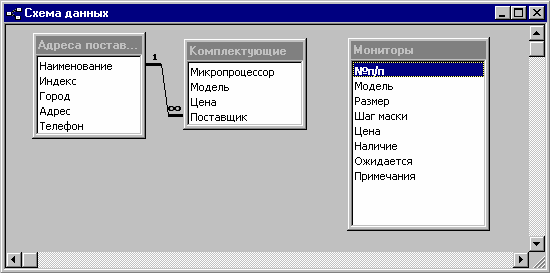


Рис. 8. Схема связей между таблицами

должно выступать ключевое поле, содержащее уникальные, неповторяющиеся значения. Значения на стороне «многие» могут повторяться.

Рассмотрим таблицу Клиенты. Здесь поле Код клиента является ключе­вым. Это понятно, поскольку у каждого клиента должен быть свой уникальный код, идентифицирующий его однозначно. Если мы рассмотрим таблицу Заказы, то увидим, что в ней код клиента не может быть уникальным, поскольку каж­дый клиент мог сделать сколь угодно много заказов. На схеме данных эти поля соединены *линией связи. С* одной стороны эта линия маркирована знаком «1>, с другой стороны — значком «бесконечность». Это графический метод изображе­ния связи «один ко многим».

Ключевым полем в таблице заказов является Код заказа — он однозначно иден­тифицирует, кто, когда, что заказал и на какую сумму. Здесь же можно узнать, какой сотрудник принял заказ к исполнению. Поскольку один сотрудник может принять множество заказов, поле Код сотрудника в таблице заказов не является ни уникальным, ни ключевым, зато в таблице Сотрудники это поле уникально.

Про подобные таблицы говорят, что они *связаны реляционными отношениями.* Соответственно, системы управления, способные работать со связанными табли­цами, называют *системами управления реляционными базами данных,* а схему данных в технической литературе могут называть *схемой реляционных отно­шений.*

6. Разработкой схемы данных заканчивается «бумажный» этап работы над техни­ческим предложением. Эту схему можно согласовать с заказчиком, после чего приступать к непосредственному созданию базы данных.

Следует помнить, что по ходу разработки проекта заказчику непременно будут приходить в голову новые идеи. На всех этапах проектирования он стремится охва­тить единой системой все новые и новые подразделения и службы предприятия. Возможность гибкого исполнения его пожеланий во многом определяется квалифи­кацией разработчика базы данных. Если схема данных составлена правильно, под­ключать к базе новые таблицы нетрудно. Если структура базы нерациональна, разра­ботчик может испытать серьезные трудности и войти в противоречия с заказчиком.

Противоречия исполнителя с заказчиком всегда свидетельствуют о недостаточной квалификации исполнителя. Именно поэтому этап предварительного проектиро­вания базы данных следует считать основным. От его успеха зависит, насколько база данных станет удобной и будут ли с ней работать пользователи. Если отмечается, что пользователи базы «саботируют» ее эксплуатацию и предпочитают работать традиционными методами, это говорит не о низкой квалификации пользователей, а о недостаточной квалификации разработчика базы.

На этом этапе завершается предварительное проектирование базы данных, и на следующем этапе начинается ее непосредственная разработка. С этого момента следует начать работу с системой управления базами данных. В нашем примере мы рассмотрим СУБД Microsoft Access 2000.

# 3. Работа с СУБД Access 2000

## Общие замечания

Ниже мы рассмотрим, как в программе Microsoft Access 2000 реализованы средства разработки основных объектов базы данных, и в упражнениях познакомимся с конкрет­ными приемами работы. Однако прежде чем приступать к освоению системы, следует учесть ряд важных замечаний, связанных с особенностями ее автоматизации.

СУБД Microsoft Access 2000 предоставляет несколько средств создания каждого из основных объектов базы. Эти средства можно классифицировать как:

• ручные (разработка объектов в режиме Конструктора);

• автоматизированные (разработка с помощью программ-мастеров);

• автоматические — средства ускоренной разработки простейших объектов.

Соотношения между этими средствами понятны: ручные средства являются наибо­лее трудоемкими, но обеспечивают максимальную гибкость; автоматизированные и автоматические средства являются наиболее производительными, но и наименее гибкими. Методической особенностью изучения программы Microsoft Access явля­ется тот факт, что в учебных целях для создания разных объектов целесообразно пользоваться разными средствами.

1. При разработке учебных таблиц и запросов рекомендуется использовать ручные средства — работать в режиме Конструктора. Использование мастеров ускоряет работу, но не способствует освоению понятий и методов.

2. При разработке учебных форм, отчетов и страниц доступа наоборот лучше пользо­ваться автоматизированными средствами, предоставляемыми мастерами. Это связано с тем, что для данных объектов большую роль играет внешний вид. Дизайн этих объектов весьма трудоемок, поэтому его лучше поручить программе, а учащемуся сосредоточиться на содержательной части работы.

3. Разработку макросов и модулей в данном пособии мы не рассматриваем. Эти средства ориентированы на профессиональных разработчиков баз данных, поэтому в рамках общетехнического курса «Информатики» для них недостаточно места.

## Работа с таблицами

**Создание таблиц.** Работа с любыми объектами начинается с окна База данных (рис. 13.9). На левой панели данного окна сосредоточены элементы управления для вызова всех семи типов объектов программы. Создание таблиц начинается с выбора элемента управления Таблицы.

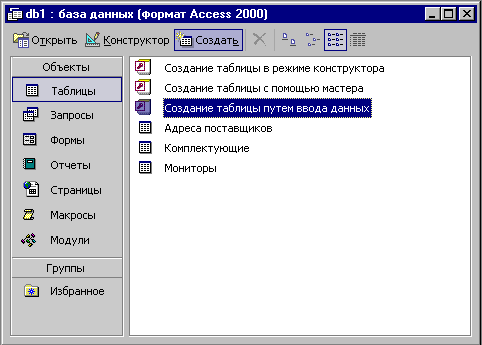


Рис. 9. Окно База данных является исходным элементом управления программы Microsoft Access

На правой панели представлен список таблиц, уже имеющихся в составе базы, и приведены элементы управления для создания новой таблицы. Чтобы создать таблицу вручную, следует использовать значок Создание таблицы в режиме конструктора.

Окно Конструктора таблиц представлено на рис. То, что мы видим в этом режиме, фактически является графическим бланком для создания и редактирования структуры таблиц. В первом столбце вводят имена полей. Если свойство Подпись для поля не задано, то Имя поля станет одновременно и именем столбца будущей таблицы. Тип для каждого поля выбирают из раскрывающегося списка, открывае­мого кнопкой выбора типа данных (см. рис. 13.10). Эта кнопка — скрытый элемент управления. Она отображается только после щелчка на поле бланка. Это надо иметь в виду — в Microsoft Access очень много таких скрытых элементов управления, кото­рые не отображаются, пока ввод данных не начат.

При изучении приемов работы с программой Microsoft Access целесообразно специально «прощелкивать» пустые поля ее бланков левой кнопкой мыши в поисках «скры­тых» элементов управления.

Нижняя часть бланка содержит список свойств поля, выделенного в верхней части. Некоторые из свойств уже заданы по умолчанию. Свойства полей не являются обязательными. Их можно настраивать по желанию, а можно и не трогать.

При создании таблицы целесообразно (хотя и не обязательно) задать ключевое поле. Это поможет впоследствии, при организации связей между таблицами. Для задания ключевого

поля достаточно щелкнуть на его имени правой кнопкой мыши и в открывшемся контекстном меню выбрать пункт Ключевое поле.

Рис. 10. Проектирование структуры таблицы

Кнопка выбора поля

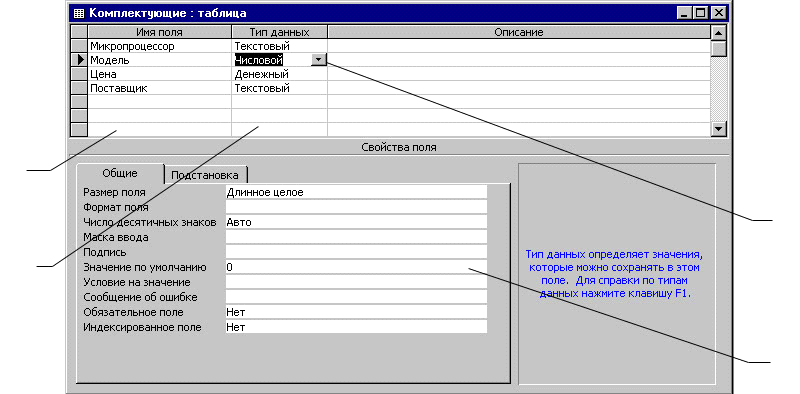
Панель редактирования свойств полей

Список полей

Список

Типов

полей



Выше мы говорили о том, что если первичный ключ необходим для связи с другими таблицами, но ни одно из полей не является уникальным, то первичный ключ можно создать на базе двух (или более полей). Эта операция выполняется точно так же, через контекстное меню, надо только уметь выделить сразу несколько полей. Группо­вое выделение выполняют при нажатой клавише SHIFT щелчками на квадратных маркерах слева от имен полей.

Закончив создание структуры таблицы, бланк закрывают (при этом система выдает запрос на сохранение таблицы), после чего дают таблице имя, и с этого момента она доступна в числе прочих таблиц в основном окне База данных. Оттуда ее и можно открыть в *случае* необходимости.

Созданную таблицу открывают в окне База данных двойным щелчком на ее значке. Новая таблица не имеет записей — только названия столбцов, характеризующие структуру таблицы. Заполнение таблицы данными производится обыч­ным порядком. Курсор ввода устанавливается в нужную ячейку указателем мыши. Переход к следующей ячейке можно выполнить клавишей TAB. Переход к очеред­ной записи выполняется после заполнения последней ячейки.

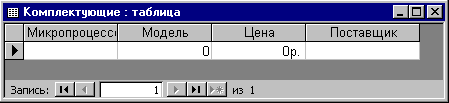


Рис. 11. Пример новой таблицы

В нижней части таблицы расположена Панель кнопок перехода. Ее элементами управ­ления удобно пользоваться при навигации по таблице, имеющей большое число записей.

Начинающим пользователям Microsoft Access доставляет неудобство тот факт, что данные не всегда умещаются в ячейках таблицы. Шириной столбцов можно управ­лять методом перетаскивания их границ. Удобно использовать автоматическое форматирование столбцов «по содержимому». Для этого надо установить указатель мыши на границу между столбцами (в строке заголовков столбцов), дождаться, когда указатель сменит форму, и выполнить двойной щелчок. Это общесистемный прием Windows 98, и им можно пользоваться в данной программе, как и во многих других.

После наполнения таблицы данными сохранять их не надо — все сохраняется авто­матически. Однако если при работе с таблицей произошло редактирование ее макета (например, изменялась ширина столбцов), СУБД попросит подтвердить сохранение этих изменений.

Если возникнет необходимость изменить структуру таблицы (состав полей или их свойства), таблицу надо открыть в режиме Конструктора. Для этого ее следует выделить в окне База данных и щелкнуть на кнопке Конструктор.

Если на этапе проектирования базы данных была четко разработана структура таб­лиц, то создание таблиц с помощью Конструктора происходит очень быстро и эффек­тивно. Даже без использования автоматизированных средств создание основы для достаточно крупных проектов происходит в считанные минуты — это ценное свойство СУБД Microsoft Access, но оно реализуется при непременном условии тщательной предварительной подготовки.

Создание межтабличных связей. Если структура базы данных продумана зара­нее, а связи между таблицами намечены, то создание реляционных отношений между таблицами выполняется очень просто. Вся необходимая работа происходит в специальном окне Схема данных и выполняется с помощью мыши. Окно Схема данных открывают кнопкой на панели инструментов или командой Сервис *>* Схема данных (если в меню Сервис не видно соответствующего пункта, следует раскрыть расширенную часть меню). Порядок создания межтабличных свя­зей рассмотрен подробно в упражнении 13.2.

Образовавшаяся межтабличная связь отображается в окне Схема данных в виде линии, соединяющей два поля разных таблиц. При этом одна из таблиц считается *главной,* а другая — *связанной.* Главная — это та таблица, которая участвует в связи своим ключевым полем (название этого поля на схеме данных отображается полу­жирным шрифтом).

Здесь мы подходим к важному вопросу: «А зачем вообще нужна связь между таб­лицами?» У связи два основных назначения. Первое — обеспечение целостности данных, а второе — автоматизация задач обслуживания базы. Представим себе, что в таблице Клиенты, где каждый клиент уникален, кто-то удалит запись для одного из клиентов, но не сделает этого в таблице Заказы. Получится, что согласно таблице

Заказы некто, не имеющий ни имени, ни адреса, а только абстрактный код, делал заказы. Узнать по коду, кто же это был на самом деле, будет невозможно — про­изошло нарушение целостности данных.

В данном случае владелец базы может применить три подхода: либо вообще ничего не делать для защиты целостности данных, либо запретить удаление данных из ключевых полей главных таблиц, либо разрешить его, но при этом адекватно обра­ботать и связанные таблицы. Вручную сделать это чрезвычайно трудно, поэтому и нужны средства автоматизации.

Связь между таблицами позволяет:

• либо исключить возможность удаления или изменения данных в ключевом поле главной таблицы, если с этим полем связаны какие-либо поля других таблиц;

• либо сделать так, что при удалении (или изменении) данных в ключевом поле главной таблицы автоматически (и абсолютно корректно) произойдет удаление или изменение соответствующих данных в полях связанных таблиц.

Для настройки свойств связи надо в окне Схема данных выделить линию, соединяю­щую поля двух таблиц, щелкнуть на ней правой кнопкой мыши и открыть контекст­ное меню связи, после чего выбрать в нем пункт Изменить связь — откроется диало­говое окно Изменение связи. В нем показаны названия связанных таблиц и имена полей, участвующих в связи (здесь же их можно изменить), а также приведены элементы управления для обеспечения условий целостности данных.

Если установлен только флажок Обеспечение целостности данных, то удалять данные из ключевого поля главной таблицы нельзя. Если вместе с ним включены флажки Каскадное обновление связанных полей и Каскадное удаление связанных записей, то, соответственно, операции редактирования и удаления данных в ключевом поле главной таблицы разрешены, но сопровождаются автоматическими изменениями в связанной таблице.

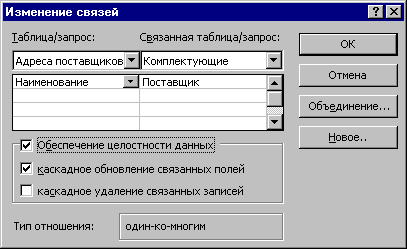


Рис. 12. Средство настройки межтабличной связи

Таким образом, смысл создания реляционных связей между таблицами состоит, с одной стороны, в защите данных, а с другой стороны — в автоматизации внесения изменении сразу в несколько таблиц при изменениях в одной таблице.

## Работа с запросами

Если структура базы данных предприятия хорошо продумана, то исполнители, работающие с базой, должны навсегда забыть о том, что в базе есть таблицы, а еще лучше, если они об этом вообще ничего не знают. Таблицы — слишком ценные объекты базы, чтобы с ними имел дело кто-либо, кроме разработчика базы.

Если исполнителю надо получить данные из базы, он должен использовать специ­альные объекты — запросы. Все необходимые запросы разработчик базы должен подготовить заранее. Если запрос подготовлен, надо открыть панель Запросы в окне База данных, выбрать его и открыть двойным щелчком на значке — откроется *резуль­тирующая таблица,* в которой исполнитель найдет то, что его интересует.

В общем случае результирующая таблица может не соответствовать ни одной из базовых таблиц базы данных. Ее поля могут представлять набор из полей разных таблиц, а ее записи могут содержать отфильтрованные и отсортированные записи таблиц, на основе которых формировался запрос. Лишь в тех случаях, когда испол­нитель не находит нужных данных в результирующей таблице, возникает необходи­мость готовить новый запрос — это задача разработчика базы.

В учебных целях запросы лучше готовить вручную, с помощью Конструктора. Как и в случае с таблицами, для этого есть специальный значок в окне База данных. Он называется Создание запроса в режиме конструктора и открывает специальный бланк, называемый *бланком, запроса по образцу.* За этим длинным названием скры­вается тот приятный факт, что, хотя запросы к таблицам баз данных пишутся на специальном языке программирования — *SQJL,* пользователям Microsoft Access изу­чать его не обязательно, а большинство операций можно выполнить щелчками кнопок мыши и приемом перетаскивания в бланке.

Бланк запроса по образцу представлен на рис. Как видно, он состоит из двух областей. В верхней отображается структура таблиц, к которым запрос адресован, а нижняя область разбита на столбцы — по одному столбцу на каждое поле будущей результирующей таблицы.

Идея формирования запроса по образцу чрезвычайно проста. С помощью контекст­ного меню на верхней половине бланка открывают те таблицы, к которым обращен запрос. Затем в них щелкают двойными щелчками на названиях тех полей, которые должны войти в результирующую таблицу. При этом автоматически заполняются столбцы в нижней части бланка. Сформировав структуру запроса, его закрывают, дают ему имя и в дальнейшем запускают двойным щелчком на значке в окне База данных.

Порядок действий, рассмотренный выше, позволяет создать простейший запрос, называемый *запросом на выборку.* Он позволяет выбрать данные из полей таблиц, на основе которых запрос сформирован.

**Упорядочение записей в результирующей таблице.** Если необходимо, чтобы дан­ные, отобранные в результате работы запроса на выборку, были упорядочены по какому-либо полю, применяют сортировку. В нижней части бланка имеется спе­циальная строка Сортировка. При щелчке на этой строке открывается кнопка раскрывающегося

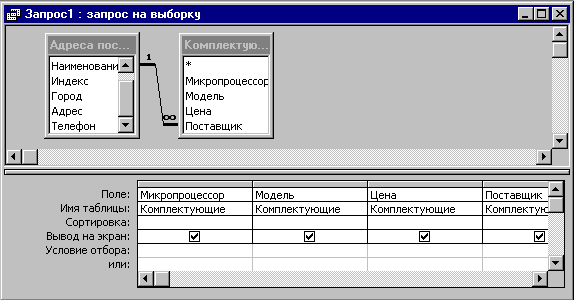


Рис. 13. Бланк запроса по образцу

списка, в котором можно выбрать метод сортировки: по возрастанию или по убыванию. В результирующей таблице данные будут отсортированы по тому полю, для которого задан порядок сортировки.

Возможна многоуровневая сортировка — сразу по нескольким полям. В этом случае данные сначала сортируются по тому полю, которое в бланке запроса по образцу находится левее, затем по следующему полю, для которого включена сортировка, и так далее слева направь. Соответственно, при формировании запроса надо рас­полагать поля результирующей таблицы не как попало, а с учетом будущей сорти­ровки. В крайнем случае, если запрос уже сформирован и надо изменить порядок следования столбцов, пользуются следующим приемом:

• выделяют столбец щелчком на его заголовке (кнопку мыши отпускают);

• еще раз щелкают на заголовке уже выделенного столбца (но кнопку не отпускают);

• перетаскивают столбец в другое место.

Управление отображением данных в результирующей таблице. В нижней части бланка запроса по образцу имеется строка Вывод на экран. По умолчанию предпо­лагается, что все поля, включенные в запрос, должны выводиться на экран, но это не всегда целесообразно. Например, бывают случаи, когда некое поле необходимо включить в запрос, например потому, что оно является полем сортировки, но, в то же время, нежелательно, чтобы пользователь базы видел его содержание. В таких случаях отображение содержимого на экране подавляют сбросом флажка Вывод на экран. Примером может быть запрос на вывод списка сотрудников предприятия, отсортированный по количеству дней, пропущенных по болезни. Он позволит каждому оценить свое положение в этом списке, но не позволит точно узнать, кто и сколько дней болел.

Использование условия отбора. Дополнительным средством, обеспечивающим отбор данных по заданному критерию, является так называемое Условие отбора. Соответствующая строка имеется в нижней части бланка запроса по образцу. Для

каждого поля в этой строке можно задать индивидуальное условие. На рис. 13.14 представлен пример, в котором мы хотим отобрать из прейскуранта торговой фирмы микропроцессоры, имеющие тактовую частоту свыше 333 МГц, но при этом стоящие менее восьмидесяти условных единиц.

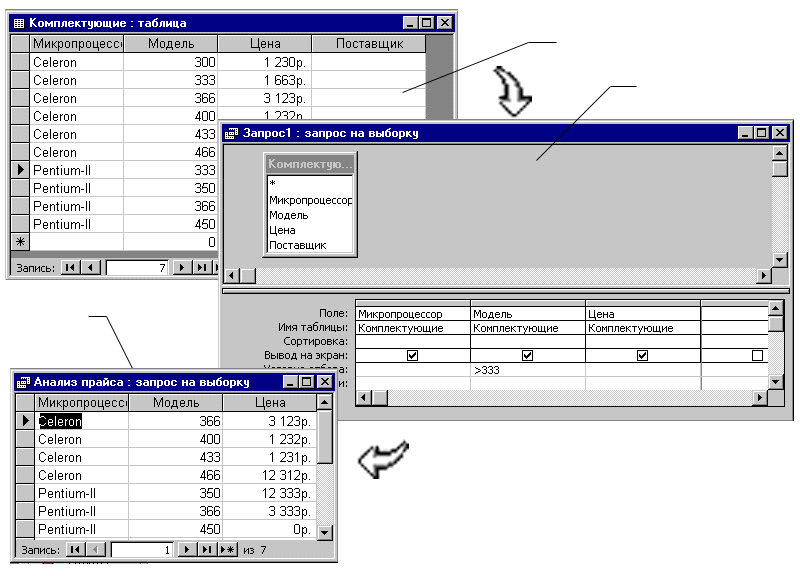
Рис. 14. Работа условия отбора

Базовая выборка

Запрос на

выборку

Результирующая таблица



**Другие виды запросов.** Мы рассмотрели запросы на выборку. Это самые простые и, в то же время, наиболее распространенные виды запросов. Однако существуют и другие виды запросов, некоторые их которых выполняются на базе предвари­тельно созданного запроса на выборку. К ним относятся прежде всего:

• *запросы с параметром* (интересны тем, что критерий отбора может задать сам пользователь, введя нужный параметр при вызове запроса);

• *итоговые запросы,* назначение которых отдаленно напоминает итоговые функции электронных таблиц (производят математические вычисления по заданному полю и выдают результат);

• *запросы на изменение —* позволяют автоматизировать заполнение полей таблиц;

• *перекрестные запросы,* позволяющие создавать результирующие таблицы на основе результатов расчетов, полученных при анализе группы таблиц;

\* *специфические запросы SQL —* запросы к серверу базы данных, написанные на языке запросов *SQL*

С радом видов запросов мы познакомимся в упражнениях.

## Работа с формами

С одной стороны, формы позволяют пользователям вводить данные в таблицы базы данных без непосредственного доступа к самим таблицам. С другой стороны, они позволяют выводить результаты работы запросов не в виде скупых результирую­щих таблиц, а в виде красиво оформленных форм. В связи с таким разделением существует два вида формирования структуры форм: на основе таблицы и на основе запроса, хотя возможен и комбинированный подход, — это вопрос творчества.

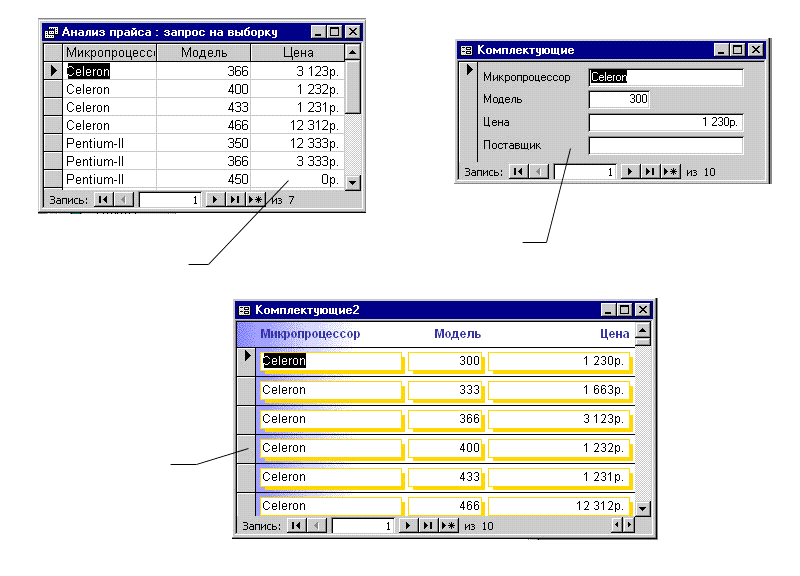
**Автоформы.** В отличие от таблиц и запросов, которые мы формировали вручную, формы удобнее готовить с помощью средств автоматизации. Полностью автома­тическими являются средства, называемые *автоформами.* Существует три вида автоформ: «в столбец», ленточные и табличные.

Автоформа

ленточная

Автоформа «в столбец»

Автоформа табличная



Автоформа

ленточная

Автоформа

«в столбец»

Автоформа

табличная

Рис.15. Три вида автоформ

Как видно из рисунков, автоформа <в столбец» отображает все поля одной записи — она удобна для ввода и редактирования данных. «Ленточная» автоформа отображает одновременно группу записей — ее удобно использовать для оформления вывода данных. Табличная автоформа по внешнему виду ничем не отличается от таблицы, на которой она основана.

Для создания автоформы следует открыть панель Формы в окне База данных и вос­пользоваться командной кнопкой Создать. В открывшемся диалоговом окне Новая форма выбирают тип автоформы и таблицу (или запрос), на которой она основывается. После щелчка на кнопке ОК автоформа формируется автоматически и немед­ленно готова к работе, то есть к вводу или отображению данных.

Обратите внимание на то, что автоформа основывается только на одном объекте. Иные средства создания форм позволяют закладывать в основу структуры формы поля нескольких таблиц или запросов. Если форма основывается только на одном объекте, она называется *простой формой.* Если форма основывается на полях из нескольких связанных таблиц, то она называется *сложной* и представляет собой композицию из нескольких форм.

**Создание форм с помощью мастера.** Автоматизированные средства предоставляет Мастер форм — специальное программное средство, создающее структуру формы в режиме диалога с разработчиком. Мастер форм можно запустить из окна База дан­ных щелчком на значке Создание формы с помощью мастера на панели Формы.

1. На первом этапе работы Мастера форм выбирают таблицы и поля, которые войдут в будущую форму.

2. На втором этапе выбирается внешний вид формы.

3. На третьем этапе выбирается стиль оформления формы.

4. На последнем этапе выполняется сохранение формы под заданным именем. Здесь же можно включить переключатель Изменить макет формы, который откры­вает только что созданную форму в режиме Конструктора. Этим удобно вос­пользоваться в учебных целях, чтобы рассмотреть структуру формы на готовом примере.

**Структура формы.** Как видно из рисунка, форма имеет три основных раздела: область заголовка, область данных и область примечания. Линии, разделяющие разделы, перетаскиваются по вертикали с помощью мыши — это позволяет изме­нять размеры разделов так, как требуется.

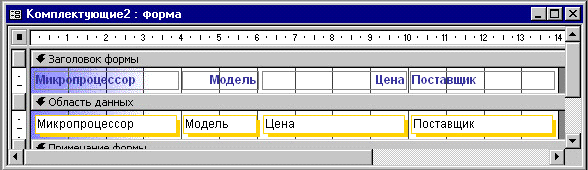
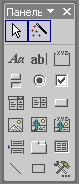


Рис. 13.16. Структура формы, созданной мастером.

Разделы заголовка и примечания имеют чисто оформительское назначение — их содержимое напрямую не связано с таблицей или запросом, на котором основана форма. Раздел данных имеет содержательное значение — в нем представлены эле­менты управления, с помощью которых выполняется отображение данных или их ввод. Разработчик формы может разместить здесь дополнительные элементы управ­ления для автоматизации ввода данных (переключатели, флажки, списки и дру­гие, типичные для приложений Windows).

**Элементы управления формы.** Элементы управления, которы­ми может пользоваться разработчик, представлены на Панели элементов. Ее открывают щелчком на соответствующей кнопке панели инструментов Microsoft Access или командой Вид > Панель элементов.

Выбор элемента управления выполняется одним щелчком на его значке в Панели элементов, после чего следующим щелчком в поле формы отмеча­ется место, куда он должен быть поставлен. Вместе с элементом в поле формы вставля­ется его *присоединенная надпись.* По умолчанию эта надпись стандартная, например для переключателей это Переключатель, Переключатель2 и т. д. Редактированием свойства элемента управления (доступ к свойствам открывается через контекстное меню) можно Дать элементу управления более содержательную подпись.



Основными элементами оформления формы являются текстовые надписи и ри­сунки. Для создания в форме текстовых надписей служат два элемента управле­ния — Надпись и Поле. В качестве надписи можно задать произвольный текст. Эле­мент Поле отличается тем, что в нем отображается содержимое одного из полей таблицы, на которой основана форма, то есть при переходе от записи к записи текст может меняться.

Для создания графических элементов оформления служат элементы управления Рисунок, Свободная рамка объекта и Присоединенная рамка объекта. Рисунок выбира­ется из графического файла и вставляется в форму. Элемент Свободная рамка объекта отличается тем, что это не обязательно рисунок — это может быть любой другой объект *OLE,* например мультимедийный. Элемент Присоединенная рамка объекта тоже в какой-то степени может служить для оформления формы, но его содержимое берется не из назначенного файла, а непосредственно из таблицы базы данных (если она имеет поле объекта *OLE).* Естественно, что при переходе между записями содер­жимое этого элемента будет меняться.

**Дизайн формы.** В то время как таблицы базы данных глубоко скрыты от посторон­них глаз, формы базы данных — это средства, с помощью которых с ней общаются люди. Поэтому к формам предъявляются повышенные требования по дизайну.

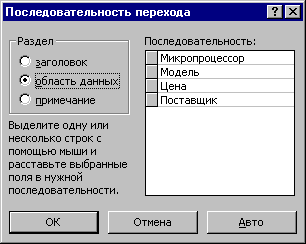
В первую очередь, все элементы управления форм должны быть аккуратно выров­нены. Это обеспечивается командой Формат *>* Выровнять. Если нужно равномерно распределить элементы управления по полю формы, используют средства меню Формат > Интервал по горизонтали или Формат > Интервал по вертикали.

Ручное изменение размеров и положения элементов управления тоже возможно, но редко приводит к качественным результатам. При работе вручную используют перетаскивание маркеров, которые видны вокруг элемента управления в тот момент, когда он выделен. Особый статус имеет маркер левого верхнего угла. Обычно эле­менты управления перетаскиваются вместе с присоединенными к ним надписями. Перетаскивание с помощью этого маркера позволяет оторвать присоединенную надпись от элемента.

Существенную помощь при разработке дизайна формы оказывает вспомогательная сетка. Ее отображение включают командой Вид > Сетка. Автоматическую привязку элементов к узлам сетки включают командой Формат > Привязать к сетке.

**Управление последовательностью перехода**. Пользователь, для которого, соб­ственно, и разрабатывается форма, ожидает, что ввод данных в нее должен происхо­дить по элементам управления слева направо и сверху вниз. Однако при проектиро­вании сложных форм, когда в процессе дизайна элементы управления многократно перемещаются с места на место, очень легко перепутать их последовательность и создать неудобный порядок ввода данных.

Физически последовательность перехода — это порядок перехода к следующему полю по окончании работы с предыдущим. Она легко проверяется с помощью клавиши TAB. Если при последовательных нажатиях этой клавиши фокус ввода «мечется» по всей форме, значит последовательность перехода нерациональна и ее надо править.



Для управления последовательностью перехода

служит диалоговое окно Последовательность

перехода. В нем представлен список элементов

управления формы. Порядок элементов в списке соответствует текущему порядку

перехода. Изменение порядка перехода выполняется перетаскиванием в два приема:

• щелчком на кнопке маркера слева от названия выделяется элемент управления (кнопка мыши отпускается);

• после повторного щелчка с перетаскиванием элемент перемещается на новое место.

Закончив разработку макета формы, ее следует закрыть и сохранить под заданным именем. После открытия формы в окне База данных, с ней можно работать: про­сматривать или редактировать данные из базовой таблицы. Проверку последова­тельности перехода выполняют клавишей TAB.

## Работа со страницами доступа к данным

*Страницы (страницы доступа к данным) —* новый объект баз данных, вошедший в последнюю версию Microsoft Access. Как и формы, этот объект служит для обес­печения доступа к данным, содержащимся в базе, но здесьречь идет об удаленном доступе, например о доступе через Интернет или через корпоративную сеть Intranet.

С помощью страниц доступа к данным решается вопрос передачи данных из базы удаленному потребителю. Обычно базы данных имеют очень большие размеры, и напрямую передавать их через медленные каналы связи непрактично. В то же время, большинство современных Web-броузеров пока не имеют функций для работы с базами данных, размещенными на серверах. Таким образом, страницы доступа выполняют как бы посредническую функцию. Они имеют небольшой размер, содержат удобные элементы управления для навигации в базе данных, могут быть запи­саны в формате кода *HTML,* переданы по медленным каналам связи и воспроизве­дены в стандартном броузере. В связи с тем, что по формату они являются Web-документами, их нетрудно встроить в любой Web-документ, например, разместить на Web-странице.

От прочих объектов базы данных страницы доступа отличаются тем, что имеют двойную природу. Прочие объекты базы являются *внутренними.* Так, например, мы не можем выделить ни таблицу, ни запрос, ни форму в виде самостоятельного файла. Эти объекты размещаются где-то внутри файла базы данных, но операцион­ная система компьютера работать с ними не может, поскольку это не файлы. С ними работает лишь сама система управления базой данных. Страница же представлена двумя объектами — внутренним объектом базы (его можно редактировать) и внешним объектом — файлом в формате *HTML* Запись этого файла происходит при сохране­нии спроектированной страницы доступа.

**Создание страницы доступа к данным**. Для страниц доступа, как и для форм, важную роль играет внешний вид, поэтому создавать их удобно с помощью мастера. Мастер страниц запускается щелчком на значке Создание страницы доступа к данным с помощью мастера.

1. На первом этапе работы Мастера форм выбирают таблицы (или запросы), в их составе — поля, к которым должна обеспечить доступ страница.

2. Второй этап работы мастера предназначен для управления группировкой данных. Эта возможность предусмотрена для доступа к базам, содержащим большие объемы данных. Если значения в некотором поле часто повторяются, имеет смысл объединить соответствующие им записи в группу. Так, например, для страницы, обеспечивающей доступ к таблице Комплектующие, в отдельные группы могут быть собраны записи, относящиеся к процессорам, материнским платам, жестким дискам и т. д.

В результате группировки образуется иерархическая структура. Она может иметь несколько уровней вложения. Так, например, внутри группы Мониторы могут быть созданы вложенные группы по производителям: Sony, Panasonic, Hitachi и т. д.

Вторая страница мастера предоставляет элементы управления для выбора полей, по которым производится группировка, и управления глубиной уровней группи­ровки. Если просмотреть в режиме Конструктора страницу, имеющую уровни группировки, то можно убедиться, что для каждого уровня группировки в струк­туре объекта образуется отдельный раздел, то есть различные уровни группи­ровки могут быть дополнены различными элементами управления экранной Web-формы.

3. На третьем этапе выбирается метод упорядочения отображаемых данных. Воз­можно задание до четырех полей сортировки, причем сортировка возможна как по возрастанию, так и по убыванию.

4. На последнем этапе выполняется сохранение страницы под заданным именем. Здесь же можно перейти в режим Конструктора, включив переключатель Изме­нить макет страницы. В случае изменения макета к странице можно применить одну из *тем оформления,* входящих в состав пакета Microsoft Office 2000. Темы оформления представляют собой совокупности стилей оформления текстов, фоновых узоров и специфических элементов оформления страницы (маркеров, линий и прочих).

Редактирование страницы доступа к данным. Редактирование созданной страницы доступа выполняется в режиме Конструктора теми же приемами, которые были описаны для форм. Основными отличиями являются:

• наличие большего количества разделов (связано с возможностью группировки);

• расширенный состав элементов управления на Панели элементов (Вид > Панель элементов);

• иной механизм перетаскивания элементов управления и присоединенных над­писей (элементы управления перетаскиваются вместе с присоединенными над­писями, но присоединенные надписи перетаскиваются отдельно от элементов управления).

## Работа с отчетами.

Отчеты во многом похожи на формы и страницы доступа к данным, но имеют иное функциональное назначение — они служат для форматированного вывода данных на печатающие устройства и, соответственно, при этом должны учитывать пара­метры принтера и параметры используемой бумаги.

Большая часть того, что было сказано о формах, относится и к отчетам. Здесь также существуют средства автоматического, автоматизированного и ручного проекти­рования. Средства автоматического проектирования реализованы автоотчетами (База данных *>* Создать > Новый отчет > Автоотчет в столбец). Кроме автоотчетов «в столбец» существуют «ленточные» автоотчеты. Разницу между ними нетрудно увидеть, поставив эксперимент.

Средством автоматизированного создания отчетов является Мастер отчетов. Он запускается двойным щелчком на значке Создание отчета с помощью мастера в окне База данных. Мастер отчетов работает в шесть этапов. При его работе выпол­няется выбор базовых таблиц или запросов, на которых отчет базируется, выбор полей, отображаемых в отчете, выбор полей группировки, выбор полей и методов сортировки, выбор формы печатного макета и стиля оформления.

Структура готового отчета отличается от структуры формы только увеличенным количеством разделов. Кроме разделов заголовка, примечания и данных, отчет может содержать разделы верхнего и нижнего колонтитулов. Если отчет занимает более одной страницы, эти разделы необходимы для печати служебной информации, например номеров страниц. Чем больше страниц занимает отчет, тем важнее роль данных, выводимых на печать через эти разделы. Если для каких-то полей отчета применена группировка, количество разделов отчета увеличивается, поскольку оформление заголовков групп выполняется в отдельных разделах.

Редактирование структуры отчета выполняют в режиме Конструктора (режим запуска­ется кнопкой Конструктор в окне База данных). Приемы редактирования те же, что и для форм. Элементы управления в данном случае выполняют функции элементов оформления, поскольку печатный отчет не интерактивный объект, в отличие от электронных форм и Web-страниц. Размещение элементов управления выполняют с помощью Панели элементов (Вид > Панель элементов), которая по составу практи­чески не отличается от Панели элементов формы. Важной особенностью отчетов является наличие средства для вставки в область верхнего или нижнего колонти­тула текущего номера страницы и полного количества страниц. Эту операцию выпол­няют с помощью диалогового окна Номера страниц (Вставка *>* Номера страниц).

1. По учебнику С.В.Симоновича Информатика: Базовый курс/ СПб.: Питер, 2001.-640 с. с исправлением и дополнением практического занятия. [↑](#footnote-ref-1)