Глава 1. Введение в Access

1.1. Понятие базы данных

База данных (БД) — это систематизированное хранилище информации, которая может относиться к различным сферам человеческой деятельности. Типичные примеры такой информации: телефонный справочник, сведения о студентах вуза, записи о заказах товаров и т.д.

До появления компьютеров вся эта информация хранилась в папках или картотеках. На каждом листе бумаги или на карточке был напечатан бланк формы, в котором были оставлены пустые места для заполнения данными. Например, в личной карточке студента нужно было заполнить графы для указания фамилии, имени и отчества, даты рождения и других сведений. Информация, относящаяся к отдельному студенту, хранилась на нескольких карточках. Это обстоятельство доставляло немало неудобств сотрудникам деканата, так как простая смена фамилии при невнимательности сотрудника, вносившего изменения в БД, могла привести к появлению «фиктивного» студента. Весьма затруднителен был и поиск нужной информации. Нередко для получения справки приходилось перебирать сотни личных карточек.

Использование компьютеров позволило устранить многие проблемы, характерные для некомпьютерных БД. При правильном проектировании компьютерной БД добавление в нее новой информации и модификация уже существующих данных перестает быть трудной задачей, чреватой ошибками. С помощью компьютера можно быстро найти нужные сведения, причем критерий поиска может быть весьма сложным. Резко упростились подготовка и печатание различных отчетов и информационных справок.

Но, чтобы возможности компьютера при работе с БД были использованы в полной мере, необходимо при ее создании соблюдать определенные правила организации информации и пользоваться программным обеспечением, специально предназначенным для этих целей. Часто пользователи хранят данные в виде документов Word или таблиц Excel. Однако работа с информацией, содержащейся в изолированных файлах, весьма затруднительна. Как текстовые редакторы, так и электронные таблицы имеют свою сферу применения и не в состоянии обеспечить полноценную поддержку таких традиционных функций БД, как:

- хранение больших массивов информации;
- исключение или сведение к минимуму дублирования данных;
- установление и поддержка связей между данными;

- защита целостности[1] данных;
- быстрый доступ к нужной информации;
- обеспечение секретности;
- простота внесения изменений;
- возможность одновременного доступа к информации для нескольких пользователей.

Чтобы обеспечить выполнение этих требований, данные в БД должны иметь определенную структуру. В зависимости от того, какой способ описания структуры данных используется при создании БД, различают сетевые, иерархические, реляционные и объектно-ориентированные БД (см. [1]). В настоящее время наибольшее распространение получили *реляционные* БД.

1.2. Реляционные БД

Информация в реляционных БД хранится в виде двумерных таблиц. В каждой таблице содержатся сведения о наборе объектов определенного типа (людях, товарах и т.д.).

Студенты : таблица

Код студента	Группа	Фамилия	Имя
871255	9701	Асылгажин	Талгат
881308	9701	Булгакова	Татьяна
891271	9701	Быкова	Наталья
891276	9701	Воробьева	Евгения
891278	9701	Вохмякова	Любовь
891280	9701	Гапеев	Иван
891285	9701	Горовая	Галина
881273	9701	Домащенко	Оксана

Рис. 1.1. Пример таблицы Access

Строки таблицы называются *записями*. Запись — это компьютерный аналог той информации, которая обычно хранилась на карточке или бланке. Любая запись в таблице содержит информацию об отдельном объекте (человеке, товаре).

[1] Под защитой целостности данных здесь понимается набор средств, предотвращающих появление в БД некорректных данных.

Столбцы таблицы называются *полями.* Поле — это компьютерный аналог графы карточки или бланка. В нем содержится информация о каком-либо свойстве описываемых объектов. Все записи состоят из одинакового набора полей. Например, если таблица содержит сведения о студентах, то в каждой ее записи хранится информация о конкретном студенте (см. рис. 1.1). В одно поле помещается его код, в другое — номер учебной группы, в третье — фамилия и т.д.

Значения в каждом поле относятся к одному типу данных: числа, строки символов, даты. Пересечение отдельной записи и отдельного поля называется *ячейкой*, а сами данные в отдельной ячейке называются *значением в поле* или *элементом таблицы*.

В простейшем случае БД состоит из одной таблицы, но обычно она включает несколько взаимосвязанных таблиц. Связь (relation) между таблицами осуществляется через общие поля. Установление связи между таблицами в реляционной БД позволяет извлекать и объединять информацию сразу из нескольких таблиц.

1.3. СУБД Access

Для взаимодействия пользователя с БД используются *системы управления базами данных* (СУБД). Одной из наиболее распространенных СУБД для персонального компьютера является СУБД Access, входящая в состав пакета Microsoft Office. В отличие от других СУБД, рассчитанных на профессиональных программистов, освоить Access и эффективно использовать его в своей работе вполне по силам и обычному пользователю, не знающему программирования.

Access позволяет пользователю решать следующие задачи:

- создавать БД и вводить в нее данные;
- просматривать и редактировать содержимое таблиц;
- устанавливать связи между таблицами;
- обеспечить защиту целостности и секретность данных;
- выполнять различные запросы к данным;
- представлять информацию в виде форм и отчетов;
- вставлять в формы и отчеты рисунки и графики;
- осуществлять операции импорта и экспорта данных;
- публиковать БД на web-страницах в Internet;
- создавать собственные программы для работы с БД, содержащие меню, диалоговые окна и командные кнопки;

- обеспечивать многопользовательский режим доступа к информации, хранящейся в БД.

В настоящем пособии описана версия Microsoft Access 97 (см. также [2-5]).

1.4. Объекты Access

Отдельные компоненты БД, которые используются для хранения и представления информации, называются *объектами*. Каждый объект имеет имя, которое может содержать до 64 символов, включая пробелы. В Access основными объектами являются: таблицы, запросы, формы, отчеты, макросы и модули. Все объекты одной БД хранятся в общем файле с расширением *mdb*.

Таблица используется для хранения информации в БД.

Запрос позволяет выбрать нужные данные из одной или нескольких таблиц. С помощью запросов можно модифицировать существующие таблицы, а также создавать новые таблицы.

Форма используется для ввода данных в таблицу и для просмотра в заданном формате данных из таблицы или запроса. С ее помощью можно также запустить на выполнение макрос или процедуру.

Отчет предназначен для создания документа на основе данных из таблицы или запроса. Этот документ можно распечатать или включить в документ другого приложения, например, Word или Excel.

Макрос представляет собой описание стандартных действий, которые нужно выполнить в ответ на определенное событие. Например, можно определить макрос, который в ответ на выбор некоторого элемента в одной форме открывает другую форму.

Модуль — это программа, написанная на языке Visual Basic for Applications (VBA). Использование модулей позволяет автоматизировать выполнение сложных действий, которые нельзя описать с помощью макросов.

Для создания таких объектов, как таблицы, запросы, формы или отчеты можно использовать специальные средства — *мастера.* Мастер задает пользователю вопросы и создает объект в соответствии с его ответами.

1.5. Запуск и завершение работы

Для запуска Access достаточно сделать двойной щелчок по ярлыку программы . Обычно при установке ярлык помещается в подменю **Программы** стартового меню Windows. Запустить Access можно также, сделав двойной щелчок по ярлыку файла с расширением *mdb*, содержащего БД Access. В этом случае БД будет автоматически загружена в Access.

Завершить работу в Access можно одним из следующих способов:

- нажать сочетание клавиш **Alt+F4;**
- выбрать команду **Выход** в меню **Файл**;
- щелкнуть по кнопке **Закрыть** окна Access.

В отличие от Word и Excel пользователю не нужно заботиться о сохранении БД перед выходом из Access. Все сделанные в ней изменения будут автоматически сохранены.

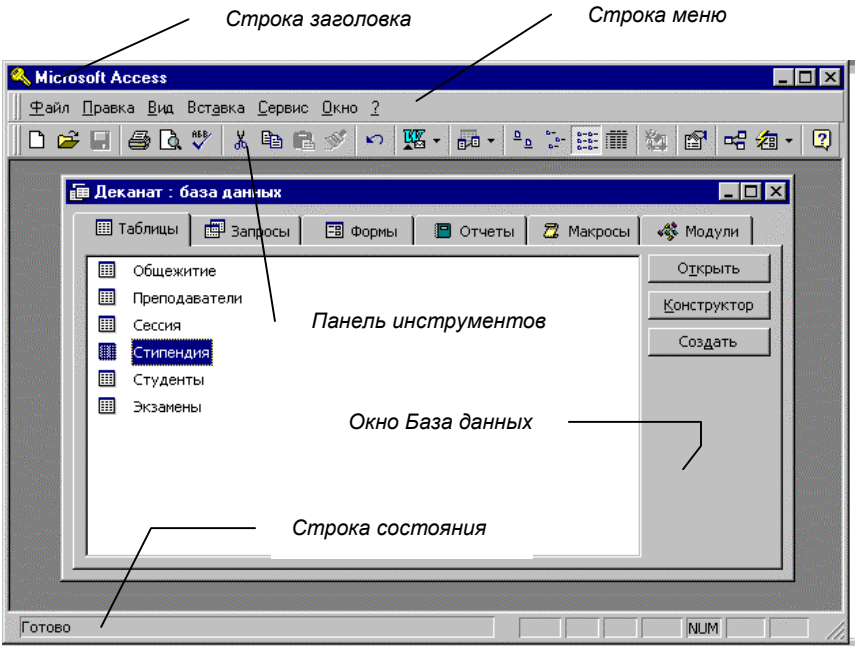


Рис. 1.2. Окно Microsoft Access

1.6. Структура окна Access

Окно Access содержит строку заголовка, строку меню, панель инструментов и строку состояния (см. рис. 1.2).

Строка заголовка содержит системное меню и три кнопки, предназначенные для сворачивания  , восстановления размеров  и закрытия  окна Access.

Строка состояния является последней строкой экрана. В ней появляется информация о назначении выбранного пункта меню или кнопки на панели инструментов.

Строка меню содержит имена раскрывающихся меню команд.

Панель инструментов содержит набор кнопок, которые предназначены для быстрого выполнения определенной команды. Access имеет несколько панелей инструментов, соответствующих различным режимам его работы. Для получения справки о той или иной кнопке достаточно подвести к ней и немного задержать указатель мыши. Имя кнопки будет выведено под указателем мыши.

Внутри окна Access могут находиться другие окна, содержащие его различные объекты: таблицы, формы, запросы и т.д.

1.7. Открытие и закрытие БД

После загрузки Access в оперативную память появляется диалоговое окно (см. рис. 1.3) и пользователю предлагается выбрать один из двух вариантов: создать новую БД или открыть существующую БД.

Вопросы, связанные с созданием новой БД, будут рассмотрены в следующей главе.

Для открытия БД, содержащейся в списке последних открывавшихся БД, нужно сделать двойной щелчок по соответствующему элементу списка. Если нужной БД в этом списке нет, то следует выбрать элемент списка *Другие файлы…,* найти в появившемся диалоговом окне **Открытие файла базы данных** файл, содержащий искомую БД, и щелкнуть по кнопке **ОК.**

Для открытия другой БД во время сеанса работы с Access нужно выбрать команду **Открыть** в меню **Файл** или щелкнуть по кнопке **Открыть БД**  . Затем следует найти в появившемся диалоговом окне нужный файл БД и щелкнуть по кнопке **ОК**. Открытой может быть только одна БД.

После открытия БД на экране появляется окно **База данных** (см. рис. 1.2), содержащее вкладки для каждого типа объектов Access. При щелчке по корешку какой-либо вкладки на экране появляется список имен имеющихся в БД объектов данного типа. При открытии БД первоначально активизируется вкладка **Таблица** и на экран выводится список таблиц. Чтобы открыть нужный объект, достаточно сделать двойной щелчок по его имени или щелкнуть по нему, а затем — по кнопке **Открыть**. Для создания новых объектов следует использовать кнопку **Создать**, а для модификации существующих объектов — кнопку **Конструктор**.

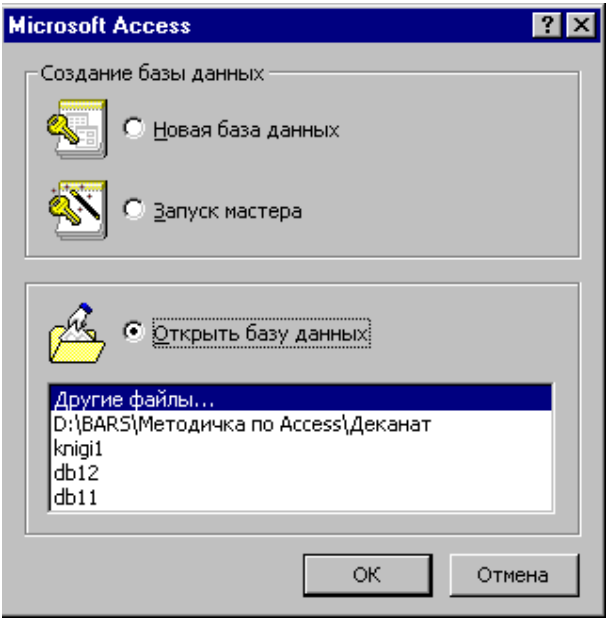


Рис. 1.3. Окно, появляющееся при запуске Access

Каждый из открытых объектов Access появляется в собственном окне. Для переключения между окнами можно использовать комбинацию клавиш **Ctrl+F6** или выбрать нужное окно в меню **Окно.** Окно **База данных** в любой момент можно активизировать нажатием клавиши **F11**. Для закрытия активного окна достаточно нажать комбинацию клавиш **Ctrl+F4** или щелкнуть по кнопке **Закрыть** этого окна.

Кроме операции открытия с каждым объектом, в зависимости от конкретной ситуации, можно совершать те или иные действия (редактировать, копировать, сохранять и т.д.). Перечень допустимых действий содержится в *контекстном меню*, которое вызывается щелчком правой кнопки мыши по объекту в окне БД или по окну объекта, если он уже открыт.

Для закрытия БД нужно выбрать команду **Закрыть** в меню **Файл** или щелкнуть по кнопке **Закрыть** окна БД.

1.8. Получение справки в Access

Для получения справки по интересующей теме нужно использовать справочную систему Access. В ней можно найти описание основных возможностей программы или получить подсказку относительно того, как выполнить то или иное действие. Для вызова справки нужно выбрать пункт меню  , а затем команду **Вызов справки**. На экране появится основное справочное окно, содержащее несколько вкладок. Вкладка **Содержание** содержит окно, в котором можно получить подробную информацию о работе с Access. Вкладка **Указатель** дает доступ к предметному указателю, с помощью которого можно по ключевому слову получить список разделов справки, относящихся к данной теме. Вкладка **Поиск** позволяет осуществить поиск разделов справки, содержащих заданные слова.

Кроме того, можно вызвать контекстную справку с помощью клавиши F1. В этом случае появится либо конкретный раздел справки, либо **Помощник,** который даст совет по возникшей ситуации или предложит перечень связанных с ней тем справки. Чтобы убрать его с экрана, достаточно щелкнуть по кнопке закрытия его окна.

Для получения справки о какой-либо кнопке или другом элементе экрана или диалогового окна нужно нажать **Shift+F1**. К указателю мыши добавится знак вопроса  . После этого нужно щелкнуть по интересующему элементу экрана. На экране появится окно справочной системы с информацией об указанном элементе экрана.

1.9. Учебные базы данных

Большинство примеров в пособии использует учебные БД **Деканат** и **Книги**.

Таблица 1.1. Перечень таблиц БД *Деканат*

Название	*Содержимое*
Студенты	сведения о студентах
Преподаватели	сведения о преподавателях
Сессия	информация об итогах сессии
Общежитие	информация об адресах студентов
Экзамены	коды и названия экзаменов
Стипендия	информация о стипендии

Структура таблиц БД **Деканат** приведена в приложении 2.

Таблица 1.2. Перечень таблиц БД *Книги*

Название	*Содержимое*
Покупатели	сведения о покупателях
Продавцы	сведения о продавцах
Книги	информация об имеющихся книгах
Заказы	информация о заказах

Структура таблиц БД **Книги** приведена в приложении 3.

Глава 2. Построение базы данных

2.1. Основные принципы проектирования БД

Создание БД должно начинаться с ее проектирования. Процесс проектирования БД включает следующие основные этапы:

1. *Определение назначения БД.* На первом этапе проектирования БД необходимо определить список задач, решаемых с ее помощью, и какие данные для этого нужны.

2. *Определение структуры таблиц.* Это один из наиболее сложных этапов в процессе создания БД. Правильная структуризация БД позволяет быстро извлечь нужные данные, исключает их дублирование и обеспечивает целостность хранящейся информации. Процедура разделения сложных данных на несколько таблиц называется *нормализацией*. Специалистами по проектированию БД была разработана теория нормализации БД (см. [1,2]). Она рекомендует при проектировании таблиц руководствоваться следующими основными принципами:

- Каждая из таблиц должна содержать информацию о наборе однотипных объектов, например, сведения о студентах или итоги сдачи сессии.
- Каждому из таких наборов данных должна соответствовать отдельная таблица. Например, сведения о студентах и о полученных ими оценках в сессию должны храниться в разных таблицах. Тогда при удалении сведений об оценках студента информация о нем останется в БД.
- Информация в таблице не должна дублироваться. Не должно быть повторений и между таблицами. Это исключает возможность несовпадения информации в разных таблицах и делает работу с БД более эффективной.

3. *Определение полей.* Каждая таблица содержит информацию о наборе объектов одного типа. При разработке полей для таблицы необходимо помнить следующее:

- В таблице должна присутствовать вся необходимая информация о данном наборе объектов.
- Каждое поле должно содержать сведения о том или ином свойстве именно этого, а не других наборов объектов. Исключением могут являться поля, используемые для связи с другими таблицами.

- Не рекомендуется включать в таблицу данные, которые являются результатом вычисления значения некоторого выражения, так называемые *вычисляемые поля*.
- Информацию следует разбивать на наименьшие логические единицы. Например, в таблицу со сведениями об адресах студентов лучше включить поля **Общежитие** и **Комната**, а не общее поле **Адрес**. Это даст возможность осуществить в БД поиск студентов, живущих в данной комнате, или произвести сортировку записей по номерам комнат.

4. *Определение ключевых полей.* Для того чтобы Access мог связать данные из разных таблиц, каждая таблица должна содержать *первичный ключ*, или просто *ключ*. Это одно или несколько полей, совокупность значений которых однозначно определяет каждую запись в таблице. Наличие ключа в таблице исключает возможность появления в ней двух одинаковых записей. При создании новой таблицы Access предлагает построить ключ путем добавления дополнительного поля с уникальными значениями.

5. *Определение связей между таблицами*. После распределения данных по таблицам и определения ключевых полей необходимо выбрать схему для связи данных в разных таблицах. Для этого нужно определить связи между таблицами. Эти связи Access будет использовать при создании многотабличных запросов, форм и отчетов.

2.2. Создание новой базы данных

Для создания БД нужно выбрать в меню **Файл** команду **Создать** или нажать кнопку **Создать базу данных** на панели инструментов. Будет открыто диалоговое окно **Создание.** В этом окне следует выбрать диск и каталог для сохранения БД и ввести имя файла новой БД. Access автоматически добавит к нему расширение *mdb*. В этом файле хранятся данные, а также описания структуры таблиц, запросов, форм, отчетов и других объектов создаваемой БД. На экране появится окно новой «пустой» базы данных. Она постепенно заполняется содержимым по мере создания с помощью соответствующих мастеров или «вручную» нужных таблиц, а затем и других объектов.

> *С помощью специального мастера можно быстро создать БД определенного типа со всеми необходимыми таблицами, формами и отчетами. Для этого в окне* ***Создание*** *нужно щелкнуть по корешку вкладки* ***Базы данных*** *и выбрать из списка нужную БД.*

2.3. Создание таблиц

Для создания таблицы нужно щелкнуть по корешку вкладки **Таблица** окна БД, а затем — по кнопке **Создать.** Access открывает окно **Новая таблица** и предлагает несколько способов создания таблицы.

2.3.1. Режим таблицы

При выборе варианта **Режим таблицы** появляется заготовка таблицы, содержащая 30 строк (записей) и 20 столбцов (полей) со стандартными именами *Поле1, Поле2* и т.д. После заполнения таблицы Access автоматически определит тип полей в зависимости от внесенной в них информации. Для изменения имени поля нужно сделать двойной щелчок по заголовку соответствующего столбца, ввести новое имя и нажать клавишу **Enter**.

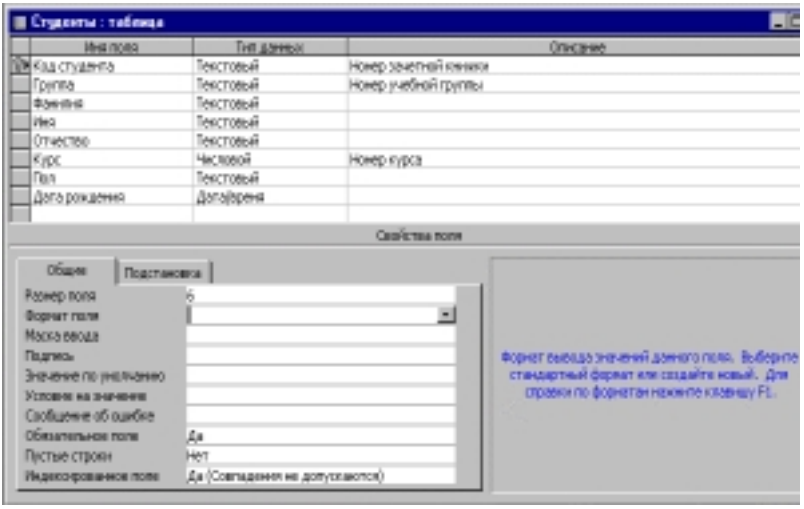


Рис. 2.1. Окно режима конструктора таблицы

После завершения ввода информации в таблицу нужно щелкнуть по кнопке **Сохранить**  и дать имя созданной таблице. Затем Access предложит добавить в нее ключевое поле. Если это предложение будет принято, то в таблицу будет добавлено поле **Код** типа *Счетчик*, содержащее номера введенных записей.

Использование режима таблицы не позволяет установить нужные свойства полей. Для этого следует перейти в режим конструктора таблицы, вызываемый нажатием кнопки **Конструктор** окна БД.

2.3.2. Режим конструктора таблицы

При создании таблицы в этом режиме открывается окно таблицы в режиме конструктора (см. рис. 2.1). В верхней части окна находится пустой бланк таблицы, содержащей три графы, и для создания таблицы нужно заполнить, по крайней мере, две из них: **Имя поля** и **Тип данных**. Для каждого из полей будущей таблицы следует выполнить следующие действия.

1. Определить имя поля таблицы

Чтобы определить поле таблицы, нужно ввести в ячейку столбца **Имя Поля** имя создаваемого поля (до 64 алфавитно-цифровых символов, включая пробелы[1]).

2. Определить тип данных поля

Для этого следует нажать клавишу Tab и перейти в столбец **Тип данных**. В этом столбце можно оставить выводящееся по умолчанию значение *Текстовый* или, раскрыв список, выбрать нужный тип данных.

Тип данных определяет вид информации, сохраняемой в поле. Например, если поле имеет тип *Числовой*, то Access не позволит ввести в это поле текст. По типу данных поля Access определяет не только, какие данные могут сохраняться в этом поле, но и сколько места для них надо зарезервировать. Для хранения данного типа *Дата/время* требуется 8 байт памяти, текстовое поле требует 1 байт для каждого символа.

Основные типы данных:

- *текстовый* — алфавитно-цифровые данные, до 255 байт;
- *поле MEMO* — комментарии и пояснения, до 64 Кбайт;
- *числовой* — числовые данные;
- *дата/время* — значения даты и времени;
- *денежный* — денежные значения;
- *счетчик* — автоматически вставляющиеся последовательные номера;
- *логический* — логические значения типа *Да/Нет*, *Истина/Ложь* или *Вкл/Выкл*;
- *объект OLE* — рисунок, таблица Excel, документ Word или другие данные в двоичном формате;
- *гиперссылка* — строка, состоящая из букв и цифр и представляющая адрес гиперссылки.

[1] Access разрешает включать пробелы в имя поля. Однако не все СУБД поддерживают эту возможность. Поэтому, если предполагается в дальнейшем использовать таблицы БД в других СУБД, то целесообразно создавать имена полей без пробелов, а пробелы включать в подпись поля.

3. Ввести описание поля (необязательно)

Для более подробного описания полей можно ввести пояснительный текст о содержащихся в поле данных в ячейку столбца **Описание**. Этот текст будет появляться в строке состояния при выборе данного поля в режиме таблицы или формы.

4. Задать свойства поля (необязательно)

Каждое поле имеет набор характеристик, называемых *свойствами*, которые задают условия хранения и отображения данных. Этот набор определяется типом данных поля. После указания типа данных Access выводит список свойств в нижней части окна конструктора и дает возможность задать нужные значения свойств поля. Настройка свойств поля позволяет, в частности, проконтролировать правильность вводимых данных, а нередко упростить сам процесс ввода.

Ниже приводится перечень и краткое описание свойств поля. Для получения подробной справки нужно щелкнуть по интересующему свойству и нажать **F1**.

Размер поля. Задает максимальный размер данных, которые будут храниться в поле. Текстовое поле может иметь размер от 1 до 255 символов (по умолчанию — 50 символов). Размер числового поля зависит от выбранного типа числовых данных. Access допускает следующие диапазоны значений для числовых данных.

- *Байт* — целые числа в пределах от 0 до 255.
- *Целое (2 байта)* — целые числа в пределах от -32768 до 32767.
- *Длинное целое (4 байта)* — целые числа в пределах от -2 147 483 648 до 2 147 483 647.
- *С плавающей точкой (4 байта)* — числа в пределах от $-3{,}4*10^{38}$ до $3{,}4*10^{38}$ с точностью до 6 разрядов.
- *С плавающей точкой (8 байт)* — числа в пределах от $-1{,}8*10^{308}$ до $1{,}8*10^{308}$ с точностью до 10 разрядов.

Формат поля. Это свойство поля задает формат представления его значений при выводе на экран или печать. Формат, установленный для поля при создании таблицы, используется по умолчанию в содержащей это поле форме или запросе.

Для числовых, денежных, логических полей, а также поля типа дата/время Access предоставляет список стандартных форматов. Этот список вызывается щелчком мыши (см. рис. 2.2). Пользователь может создать свой собственный формат поля (см. справку Access).

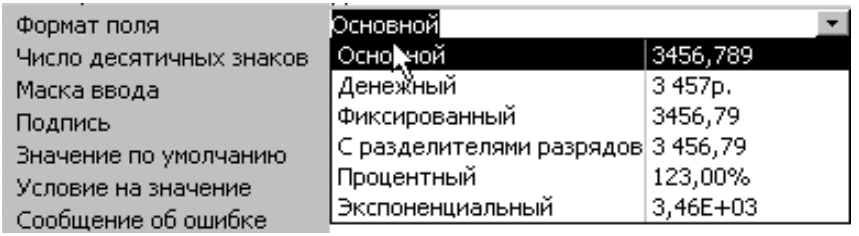


Рис. 2.2. Список стандартных числовых форматов

Дробная часть. Задает количество знаков в числах справа от запятой.

Таблица 2.1. Символы, используемые при задании маски ввода

Символ	*Назначение*
0	цифра от 0 до 9
9	может быть цифра или пробел
#	может быть цифра, пробел или символы плюс и минус
L	буква
?	может быть буква
A	буква или цифра
A	может быть буква или цифра
&	любой символ или пробел
C	может быть любой символ или пробел
<	перевод всех последующих символов на нижний регистр
>	перевод всех последующих символов на верхний регистр
. , : ; - /	разделители, сохраняющие свой вид в строке данных
!	указывает, что маску нужно заполнять справа налево
\	указывает, что следующий символ надо воспринимать не как символ маски, а буквально
«текст»	ввод постоянных текстовых символов

Маска ввода. Устанавливает символы форматирования для их автоматического заполнения во время ввода данных, например, добавления скобок и дефисов в полях номеров телефонов. Маску ввода можно использовать для преобразования вводимых символов к нужному регистру. В нее можно также включить строку символов, которая будет сохраняться в этом поле или использоваться для форматирования при выводе на экран.

Перечень символов, используемых при задании маски ввода, приведен в таблице 2.1. Символы 0, A, L и & предполагают обязательный ввод в поле.

Описание маски ввода состоит из трех разделов, разделенных точкой с запятой.

Первый раздел — это собственно маска ввода, состоящая из символов маски и постоянных символов.

Второй раздел указывает, следует ли сохранять постоянные текстовые символы маски в поле. Если постоянные символы нужно включить в значение поля, то в этом разделе следует указать **0**; в противном случае в этом разделе указывается **1**. Если этот раздел отсутствует, то сохраняются только введенные символы.

Третий раздел задает символ — указатель позиций, заполняемых при вводе. Допускается использование любого символа. Если этот раздел отсутствует, то для представления пустых символов используется символ подчеркивания (_).

Ниже приводятся описания некоторых масок ввода и примеры значений, которые могут быть в них введены (см. табл. 2.2).

Для создания маски ввода можно воспользоваться услугами мастера, который вызывается при нажатии кнопки **Построить** ...

Таблица 2.2. Примеры масок ввода

Описание маски	*Примеры значений*
>L<??????????????	Смирнов; Мария (маска для ввода слова длиной до 15 букв; первая буква становится заглавной).
(999) 900-00-00;0	(381) 23-45-67; (812) 321-43-55 (маска для ввода телефонных номеров; в поле хранятся все символы маски).
(\0\95) 000-00-00	(095) 123-54-87; (095) 321-55-88 (маска для ввода московских телефонных номеров; в поле хранятся только семь цифр номера без кода и дефисов).

Подпись. Задает альтернативное имя, которое будет появляться в качестве заголовка поля при просмотре таблицы или созданных на ее основе запросах, формах или отчетах. Например, если установить значение *№ зачетки* для подписи поля **Код студента** таблицы **Студенты**, то при переходе в режим таблицы это название будет выведено в качестве заголовка данного поля.

Значение по умолчанию. Задает значение, автоматически вводящееся в поле при создании новой записи. Для числового поля значение по

умолчанию равно 0, а для текстового или Memo поля — значение *Null* (данные в поле неизвестны**).**

Условие на значение. Задает логическое выражение, определяющее условие для ввода или редактирования данных. Выражение принимает значение «Истина» (*True*), если значение в поле удовлетворяет заданному условию, и «Ложь» (*False*) — в противном случае. Access не позволит ввести в поле значение, не удовлетворяющее условию, и выдаст сообщение об ошибке.

Таблица 2.3. Примеры условий на значение поля

Условие	*Комментарий*
>= 0	значение должно быть больше или равно 0
< #10.03.98#	дата, предшествующая 10 марта 1998 года
>10 Or Is Null	значение должно быть больше 10 или пусто
<= 20 And >= 10	значения должны находиться между 10 и 20
Not «Москва»	значение, отличное от слова «Москва»
Between 0 And 100	значения должны находиться между 0 и 100
"Омск" Or "Томск"	любое значение из списка городов
In ("Омск"; "Томск")	любое значение из списка городов
In (1; 3; 5; 11; 17)	любое значение из списка чисел
Like "A*"	любой текст, начинающийся с буквы А

Условие на значение задается выражением, состоящим из операторов сравнения (>, >=, <, <= и т.д.) и *операндов* — значений, используемых для сравнения. Если выражение не содержит оператора, то Access будет использовать оператор «равно» (=). Условие может содержать несколько сравнений, разделенных логическими операторами *Or* (логическое сложение) или *And* (логическое умножение). Для отрицания некоторого условия используется логический оператор *Not* (логическое отрицание).

Текстовые значения должны заключаться в кавычки. Если в качестве операнда используется дата, то ее нужно заключить в символы номера (#).

Для проверки, попадает ли значение в заданный интервал, можно использовать логический оператор *And* или оператор *Between*. Чтобы проверить, содержится ли значение в заданном списке, нужно использовать логический оператор *Or* или оператор *In*. Проверку соответствия вводимого значения текстового поля или поля Memo некоторому шаблону можно осуществить с помощью оператора *Like*.

Описание основных операторов сравнения и логических операторов приведено в главе 4.

Для создания условия на значение поля можно использовать построитель выражений, который вызывается при нажатии кнопки **Построить** ... . Он содержит перечень операторов, функций и констант, которые можно добавлять в создаваемое выражение.

Сообщение об ошибке. Содержит текст сообщения, которое будет выведено на экран при вводе в поле данных, нарушающих условие на значение. Например, если задать для поля **Оценка** условие *In (2; 3; 4; 5)* и ввести текст сообщения: *Введена неверная оценка*, то при попытке ввода любого символа, отличного от вышеперечисленных цифр, Access откажется занести его в таблицу **Сессия** и выдаст соответствующее сообщение.

От значений следующих двух свойств поля зависит, как Access будет интерпретировать отсутствие значения в поле.

Пустые (Null) значения и пустые строки

При вводе информации в таблицу некоторые поля могут остаться незаполненными из-за отсутствия соответствующих данных. В Access различаются следующие ситуации:

- *Значение в поле существует, но неизвестно*. В этом случае в поле ничего не вводится, и Access будет хранить в нем так называемое *пустое значение*, обозначаемое *Null*. Значение *Null* может использоваться в выражениях и вводиться в качестве критерия отбора в запросы. Ключевые поля не могут содержать значений *Null*.
- *Известно, что значение в поле отсутствует*. В этом случае в поле вводятся две кавычки и его значение — *пустая строка* (""). Пустую строку можно ввести только в текстовое и Memo поле.

Обязательное поле. Позволяет указать, обязателен ли ввод данных в это поле. Если задать для этого свойства значение *Нет* и оставить поле пустым (ввести пробелы или сразу нажать клавишу **Enter**), то Access будет считать, что в это поле введено значение *Null*[1]. Если же это свойство имеет значение *Да*, то значения *Null* в этом поле не допускаются и при вводе новой записи в это поле обязательно должно быть введено значение.

Пустые строки. Если установить значение этого свойства равным *Да,* то для текстового или Memo поля будет разрешен ввод пустых строк. В

[1] Если определено условие на значение поля, то для разрешения ввода в поле пустых значений недостаточно задания для свойства Обязательное поле значение *Нет*. В этом случае само условие должно иметь вид: *условие_на_значение Or Is Null* (см. третий пример в таблице 2.3).

противном случае Access преобразует введенные пустые строки, а также строки, состоящие только из пробелов, в строки, содержащие значение *Null* (при условии, что свойство **Обязательное поле** имеет значение ***Нет***).

Отметим, что если свойство **Пустые строки** имеет значение *Да*, то пустые строки являются допустимыми значениями данного поля вне зависимости от значения свойства **Обязательное поле**. Если же для свойства **Обязательное поле** задано значение *Да*, а для свойства **Пустые строки** значение *Нет*, то Access не разрешит ввод в поле пустой строки и потребует ввести непустое значение.

Пример 2.1. Предположим, что мы хотим различать следующие две ситуации:

а) неизвестно, есть ли у преподавателя домашний телефон;

б) известно, что у преподавателя нет домашнего телефона.

Тогда в таблице **Преподаватели** нужно установить для поля **Домашний телефон** свойству **Обязательное поле** значение *Нет,* а свойству **Пустая строка —** значение ***Да.*** **П**ри заполнении данными соответствующей записи таблицы **Преподаватели** в этом поле **Домашний телефон** следует ввести пустую строку (""), если известно, что у преподавателя нет домашнего телефона**.** Если же такая информация отсутствует, то в поле вводить ничего не нужно (нажать клавишу **Enter**). В этом случае его значением будет *Null*. При необходимости с помощью запроса можно будет получить список преподавателей, не имеющих домашнего телефона, либо список преподавателей, относительно которых такая информация в БД отсутствует (см. пример 5.7).

Индексированное поле. Создается *индекс* по данному полю. Индекс указывает местоположение записей таблицы на диске и помогает Access быстро находить нужные данные. Если таблица содержит много записей и часто проводится операция поиска или сортировки по определенным полям, то ее выполнение можно значительно ускорить, создав индекс по этим полям. Access также использует индексы для установления связей между таблицами. Ключевые поля таблицы индексируются автоматически.

При индексировании поля есть две возможности. Выбор варианта *Да (Совпадения не допускаются)* означает, что создается *уникальный индекс*. В этом случае таблица не может иметь в этом поле повторяющиеся значения. При выборе варианта *Да (Совпадения допускаются*) создается индекс, учитывающий возможность повторения значений в этом поле.

Для создания индекса, основанного на нескольких полях (*составного индекса*) нужно щелкнуть по кнопке **Индексы**  панели инструментов и ввести в столбце **Индекс** окна диалога имя индекса. Затем в столбце **Имя поля следует** указать первое поле индекса и добавить дополнительные

поля в расположенные ниже строки, не вводя других имен индексов. В столбце **Порядок сортировки** можно изменить порядок сортировки для любого поля, входящего в создаваемый индекс.

Для удаления индекса нужно щелкнуть по кнопке **Индексы**, выделить строки, определяющие удаляемый индекс, и нажать клавишу **Delete.**

Задание свойства поля

Для задания свойства поля нужно щелкнуть мышью по изменяемому свойству и выполнить одно из следующих действий:

- если в ячейке появится кнопка раскрытия списка, нажать эту кнопку и выбрать значение из списка.
- если рядом с ячейкой появится кнопка **Построить**  , нажать эту кнопку. После открытия окна построителя выражений ввести нужное значение или выражение.

Установленные свойства поля таблицы автоматически переносятся на использующие это поле запросы, формы и отчеты.

Создание ключевого поля

Как уже говорилось, обычно таблица содержит ключевое поле (ключ). Использование ключа дает следующие преимущества:

- *уникальность записей* — Access не позволяет вводить и хранить в таблице записи, имеющие одинаковое значение в ключевом поле;
- *связи* — используя ключ, Access устанавливает связи между таблицами;
- *скорость* — Access создает индекс по ключевому полю, ускоряющий поиск нужных записей и выполнение запросов;
- *упорядочение* — Access автоматически сортирует и отображает записи таблицы в порядке возрастания или убывания значений в ключевом поле.

Простейший способ создания ключевого поля заключается в создании поля типа *Счетчик* и объявлении его ключевым. Если до сохранения созданной таблицы ключ не был определен, то Access создает его именно таким способом.

Если в таблице имеется поле, содержащее значения, уникальные для каждой записи, то это поле можно объявить ключевым. Пример поля такого типа — поле **Код студента** в таблице **Студенты**, содержащее номера зачетных книжек студентов. Такое поле называется *простым ключом*.

В том случае, когда нельзя гарантировать уникальность значений ни одного из полей, можно создать ключ, состоящий из нескольких полей. Такое ключевое поле называется *составным ключом*. Пример составного ключа — совокупность полей **Код студента и Код экзамена** в таблице **Сессия.** Ни одно из этих полей по отдельности не может использоваться в

этой таблице в качестве ключевого, так как каждое из них содержит повторяющиеся значения. Однако комбинация значений этих полей уникальна, поэтому их совокупность может служить ключом.

Чтобы объявить одно или несколько полей ключевыми, нужно выделить эти поля, щелкнув по ним мышью. Если полей несколько, то их выделение следует производить, держа нажатой клавишу **Ctrl.** Затем нужно щелкнуть по кнопке **Ключевое поле**  панели инструментов.

Для удаления ключа достаточно выделить составляющие его поля и повторно щелкнуть по кнопке **Ключевое поле**. Другой способ — щелкнуть по кнопке **Индексы**  и затем удалить индекс *PrimaryKey*.

Подстановка данных

Подстановка данных является очень полезной возможностью в Access. Ее использование во многих случаях позволяет существенно упростить процедуру ввода данных. Фактически подстановка сводится к созданию *столбца подстановки* — списка, обычно содержащего несколько столбцов. Значения одного из них, так называемого *присоединенного столбца*, заносятся в поле таблицы, называемое *полем подстановки*. Остальные столбцы служат комментариями, обеспечивая выбор нужного элемента списка.

Пусть, например, нужно ввести в таблицу **Сессия** результаты сдачи какого-либо экзамена. Экзаменационная ведомость содержит следующую информацию: фамилию, имя и отчество студента, его оценку и некоторые другие данные. Так как в таблице **Сессия** студент представлен не фамилией, а кодом, то при вводе информации приходится всякий раз по фамилии студента отыскивать его код в таблице **Студенты** и затем вводить найденное значение в БД.

Ясно, что при такой организации ввода значительная часть времени и усилий затрачивается не на саму операцию ввода, а на поиск нужных кодов, причем вполне возможно занесение ошибочного кода. С помощью подстановки удается заменить поиск и ввод кода студента следующей существенно более удобной процедурой: из общего списка студентов, созданного Access на базе таблицы **Студенты**, выбирается нужный студент, после чего его код автоматически заносится в таблицу **Сессия**.

Мастер подстановок

Чтобы создать подстановку для поля таблицы, проще всего использовать соответствующий мастер. Для этого нужно выбрать в качестве типа данных поля значение: *Мастер подстановок*. Опишем работу мастера на примере создания подстановки для поля **Код студента** в таблице **Сессия**.

1. На первом шаге мастер предлагает указать источник данных для столбца подстановки:
   - значения из таблицы или запроса;
   - фиксированный список значений.

   По умолчанию используется первый пункт. Поэтому, так как мы собираемся использовать в качестве источника данных таблицу **Студенты**, нужно просто щелкнуть по кнопке **Далее**.
2. Мастер просит указать таблицу или запрос, содержащие столбец подстановки. В списке таблиц нужно выбрать таблицу **Студенты** и щелкнуть по кнопке **Далее**.
3. Затем нужно двойным щелчком мыши отобрать поля, используемые в подстановке: **Код студента, Фамилия** и **Имя**. Последнее поле добавлено, чтобы иметь возможность различать студентов-однофамильцев.
4. Так как **Код студента** — ключевое поле, то Access автоматически выбирает его в качестве источника данных (присоединенного столбца) для поля подстановки. На этом шаге можно изменить ширину полей, используемых в подстановке, и указать, надо ли скрыть присоединенный столбец. Оставим без изменения установку по умолчанию: *Скрыть ключевой столбец*.
5. На последнем шаге зададим подпись для столбца подстановки: *Студент* и нажмем кнопку **Готово**. Access попросит сохранить таблицу, и операция создания подстановки завершена.

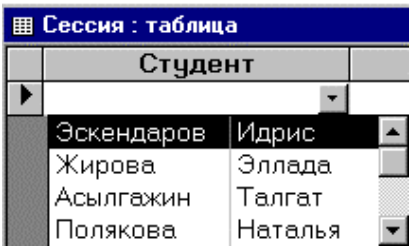


Рис. 2.3. Столбец подстановки

Для просмотра полученных результатов перейдем в режим таблицы, щелкнув по кнопке **Вид** на панели инструментов. В первом столбце **Студент** вместо кодов студентов появятся их фамилии. Перейдем на последнюю (пустую) запись таблицы, щелкнем по ячейке в поле **Студент**, а затем по появившейся справа кнопке. Откроется список, содержащий фамилии и имена студентов (см. рис. 2.3). Если выбрать щелчком мыши одного из студентов, то в ячейке появится его фамилия, а в таблицу **Сессия** будет занесен его код.

В качестве источника данных для столбца подстановки может использоваться фиксированный список значений. Например, можно создать

столбец подстановки для поля **Оценка**, указав на первом шаге работы мастера, что значения берутся не из таблицы, а из списка. Затем следует создать список, содержащий перечень возможных оценок: 2, 3, 4 и 5. После завершения создания столбца подстановки значения в поле **Оценка** можно будет не вводить с клавиатуры, а выбирать из списка.

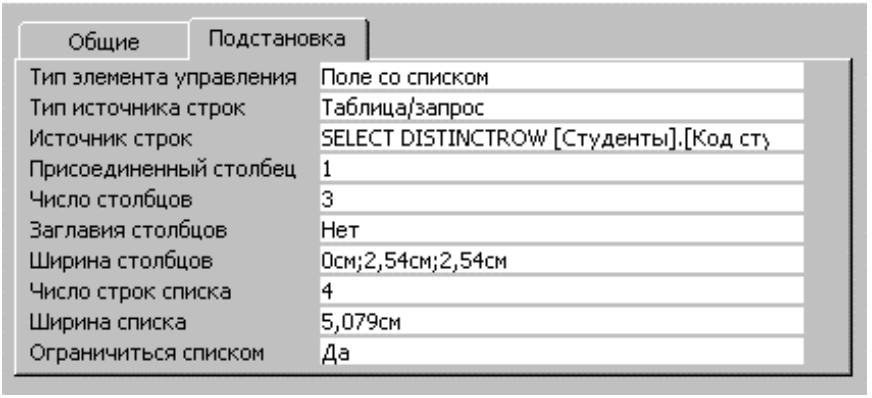


Рис. 2.4. Свойства поля подстановки

Чтобы изменить свойства поля подстановки, созданного с помощью мастера, нужно щелкнуть по ярлычку **Подстановка.** На рис. 2.4 видны свойства поля **Код Студента**. Ниже дается их краткое описание:

- *Тип элемента управления* — задается представление этого поля в форме.
- *Тип источника строк* — указывается тип источника данных для поля. Им может быть *таблица/запрос* (по умолчанию) или *список значений*.
- *Источник строк* — определяет источник данных. Если тип источника строк — *таблица/запрос*, то здесь указывается имя таблицы/запроса или инструкция SQL. Если тип источника строк — *список значений*, то указывается список элементов, разделяемых точкой с запятой.
- *Присоединенный столбец* — указывается номер столбца, значения из которого заносятся в поле.
- *Число столбцов* — задает число выводящихся столбцов.
- *Заглавия столбцов* — выводятся (*Да*) или нет (*Нет*) в качестве заголовков столбцов имена полей или первые элементы списка значений.
- *Ширина столбцов* — задается ширина выводимых столбцов (через точку с запятой). Чтобы скрыть столбец, нужно установить его ширину, равной 0.

- *Число строк списка* — задает максимальное число строк, выводящихся в раскрывающемся списке.
- *Ширина списка* — задает ширину раскрывающегося списка.
- *Ограничиться списком* — указывает, что в поле вводятся только значения, принадлежащие списку (*Да*), или разрешен ввод любых других значений (*Нет*). Access разрешит вводить в поле значения, не принадлежащие списку, лишь в том случае, если присоединенный столбец является первым отображаемым столбцом в списке.

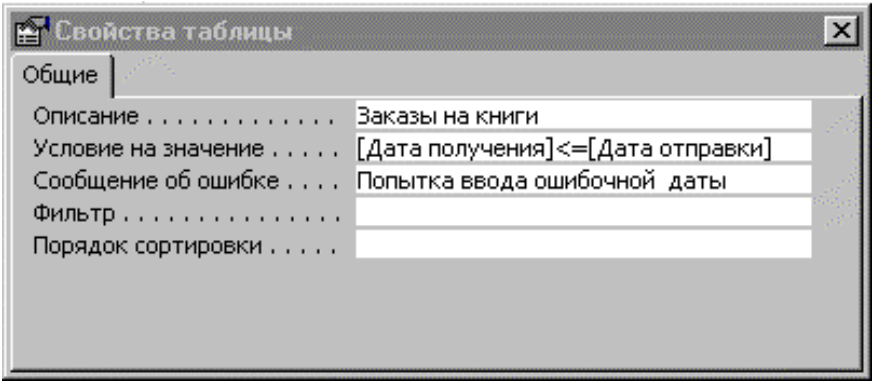


Рис. 2.5. Установка свойств таблицы Заказы

Установка свойств таблицы

Кроме задания свойств полей пользователь имеет возможность задать свойства таблицы. Эти свойства относятся ко всей таблице и всем ее записям. В частности, можно задать условия на значения в записях, относящиеся не к одному, а нескольким полям таблицы, и указать текст выводимого сообщения при нарушении этих условий.

Для установки свойств таблицы нужно щелкнуть правой кнопкой мыши в окне конструктора и выбрать в появившемся контекстном меню пункт **Свойства**. В появившемся диалоговом окне **Свойства** таблицы (см. рис. 2.5) следует задать нужные свойства таблицы.

Пример 2.2. Зададим следующее условие на значения в записях таблицы **Заказы: д**ата получения заказа не превосходит дату его отправки.

Для этого вызовем окно **Свойства таблицы**, щелкнем по ячейке **Условие на значение** и с помощью построителя выражений введем выражение: *[Дата получения]<=[Дата отправки]*. Затем введем в ячейку **Сообщение об ошибке** текст*: Попытка ввода ошибочной даты*.

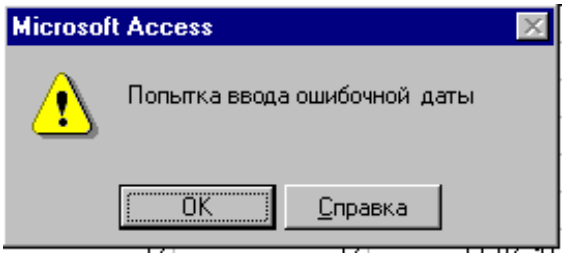


Рис. 2.6. Сообщение об ошибке

Если теперь попытаться ввести в таблицу новую запись или изменить уже существующую запись таким образом, что введенное условие окажется нарушенным, то Access откажется выполнить эту операцию и выдаст предупреждающее сообщение (см. рис. 2.6).

2.3.3. Мастер таблиц

Чтобы создать таблицу с помощью мастера, нужно в окне диалога **Создание таблицы** нажать кнопку **Мастер.** Мастер **Создание таблиц** предлагает на выбор различные образцы таблиц, каждая из которых содержит набор полей. Пользователь должен отобрать те поля, которые он собирается включить в свою таблицу. После этого мастер автоматически создает таблицу, предоставляя на завершающем этапе возможность пользователю создать ключевое поле и определить связи с уже существующими таблицами.

После завершения работы мастера можно перейти в режим конструктора и произвести необходимые изменения в структуре таблицы и свойствах ее полей.

2.3.4. Импорт таблиц

Таблицу можно создать, импортируя данные из файлов других форматов. Если выбран этот способ создания таблицы, то открывается окно диалога **Импорт (см.** рис. 2.7). Пользователь должен указать тип файла, из которого будут импортированы данные, и найти этот файл.

Проще всего производится операция импорта таблицы из другой БД Access, а также таблиц, созданных в формате других СУБД (dBASE, FoxPro, Paradox). Она осуществляется автоматически без участия пользователя.

При импорте электронных таблиц (файла Excel или Lotus 1-2-3) можно прочитать целый лист или именованный диапазон. При этом импортируемая таблица должна иметь стандартный формат баз данных, когда столбцы являются полями, и каждая строка представляет собой отдельную запись.

После выбора файла электронной таблицы вызывается мастер **Импорт электронной таблицы** и пользователю предлагается ответить на ряд

вопросов. Сначала он должен выбрать вариант импорта данных: лист или именованный диапазон. Затем указать, содержит ли первая строка источника данных имена полей, и выбрать вариант сохранения данных: в новой или существующей таблице. На следующем шаге пользователь имеет возможность изменить названия полей, создать индексы и отметить поля, не включаемые в таблицу. В заключение мастер предлагает создать ключевое поле и дать таблице имя.

По сходной схеме производится импорт данных из текстового файла.

Для осуществления операции импорта можно также вызвать щелчком правой кнопки мыши по окну базы данных контекстное меню и выбрать в нем пункт **Импорт**.

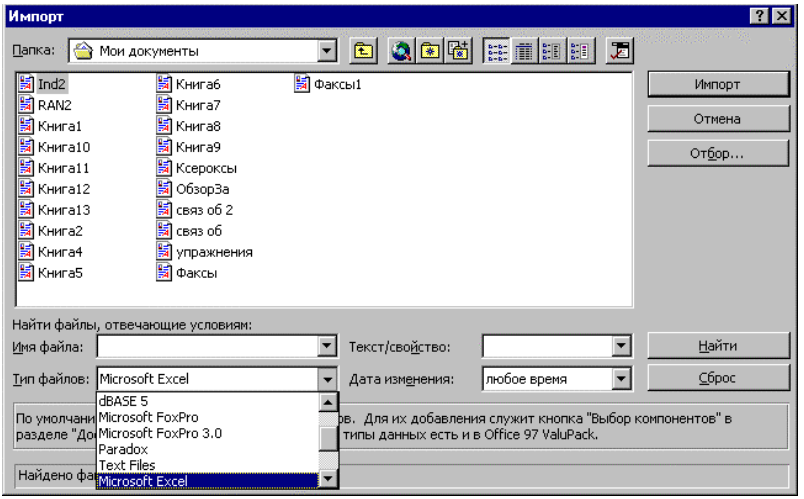


Рис. 2.7. Окно для выбора файла при импорте данных

2.3.5. Связывание таблиц

Альтернативным способом использования в Access данных из других источников (баз данных, электронных таблиц или текстовых файлов) является установление связи с внешними данными — *связывание* данных.

При импорте данных создается их копия в новой таблице текущей БД. После этого всякая связь с внешними данными теряется. Связывание позволяет использовать данные из внешнего файла, не импортируя их в Access, и в этом случае пользователь всегда имеет дело с самой «свежей» информацией. Формат данных файла-источника не меняется и его можно продолжать использовать в приложении, в котором он был создан.

Поэтому в тех случаях, когда внешний файл часто изменяется или используется в режиме коллективного доступа, целесообразно вместо операции импорта использовать связывание. Правда, это может привести к определенному снижению быстродействия при работе с данными, так как Access максимально эффективно работает со своей копией данных и в «родном» формате.

Операция связывания производится при помощи мастера **Связь**, который начинает работу при выборе пункта **Связь с таблицами** контекстного меню окна БД. Операция связывания таблицы из другой БД осуществляется автоматически без участия пользователя. При связывании листа или именованного диапазона электронной таблицы мастер запрашивает, содержит ли первая строка источника данных имена полей, и затем устанавливает связь.

Для связанных таблиц изменение некоторых свойств полей становится невозможным. В связанную таблицу нельзя добавить новое поле или удалить существующее поле, но можно добавить записи, а также изменить значения отдельных полей. Если связь установлена с таблицей Access, то доступна также операция удаления записей.

В Access для обозначения связанных таблиц и таблиц, хранящихся в текущей БД, используются разные значки. Для удаления связи нужно щелкнуть по значку связанной таблицы и нажать клавишу **Delete**.

2.4. Сохранение таблицы

После определения всех полей таблицы нужно нажать кнопку **Сохранить** на панели инструментов или выбрать в меню **Файл** команду **Сохранить** для сохранения макета таблицы.

Выводится окно диалога **Сохранение**, в которое следует ввести имя таблицы и нажать кнопку **OK**.

2.5. Определение связей между таблицами

После того как в БД созданы основные таблицы, следует указать, как они связаны друг с другом. Эти связи Access будет использовать в запросах, формах и отчетах при отборе информации из нескольких таблиц. Задание связей между таблицами позволяет также обеспечить защиту целостности данных в БД.

Связь между двумя таблицами, одна из которых является *главной*, а другая *подчиненной,* устанавливает соответствие между записями этих таблиц. Для установления связи нужно, чтобы в главной таблице существовало поле или группа полей, совокупность значений которых однозначно определяла бы запись (была уникальной). Обычно в качестве

поля (группы полей) связи выбирается ключевое поле таблицы, но достаточно, чтобы оно имело уникальный индекс. В соответствие ему ставится поле (группа полей) подчиненной таблицы, называемое *внешним ключом*. Связь между записями устанавливается по совпадению значений в полях связи. Access анализирует значение поля связи (ключа) любой записи главной таблицы и связывает ее со всеми записями подчиненной таблицы, имеющими такое же значение внешнего ключа.

Поля связи могут иметь разные имена, но они должны иметь один тип данных и иметь однотипное содержимое. Исключение из этого правила: поле типа *Счетчик* можно связывать с числовым полем, имеющим в свойстве **Размер поля** значение «*Длинное целое*». Кроме того, связываемые поля числового типа должны иметь одинаковые значения свойства **Размер поля**.

Например, для получения информации о студентах и полученных ими оценках следует определить связь по полям **Код Студента** в таблицах **Студенты** и **Сессия**. Главной в этой связи будет таблица **Студенты**, а подчиненной — таблица **Сессия**. Каждой записи в главной таблице — данным о студенте — соответствует несколько (или ни одной) записей из подчиненной таблицы — информация об его оценках. Поле **Код Студента** в таблице **Сессия** является внешним ключом.

2.5.1. Типы связей между таблицами

1. Связь типа «один-ко-многим»

Описанная выше связь между таблицами **Студенты** и **Сессия** — пример связи типа «один-ко-многим». Это наиболее распространенный тип связи. При таком типе связи каждой записи в главной таблице могут соответствовать одна, несколько или ни одной записи в подчиненной таблице, а каждая запись в подчиненной таблице не может иметь более одной соответствующей ей записи в главной таблице. Если запись в подчиненной таблице не имеет соответствующей ей записи в главной таблице, то значение поля связи в такой записи должно быть пусто.

2. Связь типа «один-к-одному»

При связи типа «один-к-одному» каждой записи в главной таблице может соответствовать не более одной записи в подчиненной таблице, и наоборот, каждая запись в подчиненной таблице не может иметь более одной соответствующей ей записи в главной таблице. Как и в предыдущем случае, если запись в подчиненной таблице не имеет соответствующей ей записи в главной таблице, то значение поля связи в такой записи должно быть пусто.

Этот тип связи применяется реже, так как такие данные могут быть помещены в одну таблицу. Связь типа «один-к-одному» обычно используют для разделения таблиц, имеющих много полей, а также для сохранения сведений, относящихся к подмножеству записей в главной таблице. Например, такой тип связи использован при установлении связей между таблицами **Студенты** и **Общежитие**.

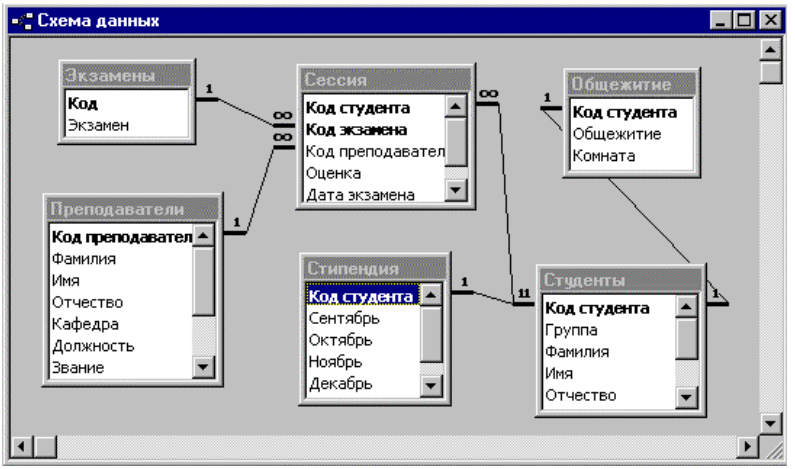


Рис. 2.8. Связи между таблицами в БД ***Деканат***

3. Связь типа «многие-ко-многим»

При установлении связи между таблицами возможна ситуация, когда между ними нельзя установить отношение «главная-подчиненная» из-за того, что любой записи в одной таблице может соответствовать несколько записей из другой таблицы. Примером могут служить таблицы **Студенты** и **Преподаватели**, так как каждый студент сдавал экзамены нескольким преподавателям, а каждый преподаватель принимал экзамен у нескольких студентов. Поэтому между этими таблицами нельзя установить ни одну из описанных выше связей.

Это пример связи типа «многие-ко-многим». Access непосредственно не поддерживает такой тип связи. Но ее можно реализовать в виде двух связей типа «один-ко-многим» с помощью третьей (связующей) таблицы. В качестве такой связующей таблицы в БД **Деканат** используется таблица **Сессия** (см. рис. 2.8). Она связана как с таблицей **Студенты** по полю **Код студента**, так и с таблицей **Преподаватели** по полю **Код преподавателя**. После того как эти связи установлены, можно легко определить с помощью

соответствующих запросов, у каких студентов принимал экзамены данный преподаватель или кому сдавал экзамены данный студент.

2.5.2. Создание связей

Для создания связей между таблицами нужно вернуться в окно БД, закрыть все открытые таблицы и щелкнуть по кнопке **Схема данных**  панели инструментов либо вызвать щелчком правой кнопки мыши контекстное меню и выбрать в нем пункт **Схема данных**. Если связи в БД определяются впервые, то будет открыто пустое окно **Схема данных.** В это окно нужно добавить таблицы, между которыми устанавливается связь.

Для добавления таблиц следует вызвать щелчком правой кнопки мыши контекстное меню окна **Схема данных** и выбрать в нем пункт **Добавить таблицу**. Откроется диалоговое окно **Добавление таблицы,** содержащее список таблиц БД (см. рис. 2.9). Для добавления таблицы нужно щелкнуть по ее имени, а затем — по кнопке **Добавить**. После того как все таблицы отобраны, нужно закрыть это окно и вернуться в окно **Схема данных.**

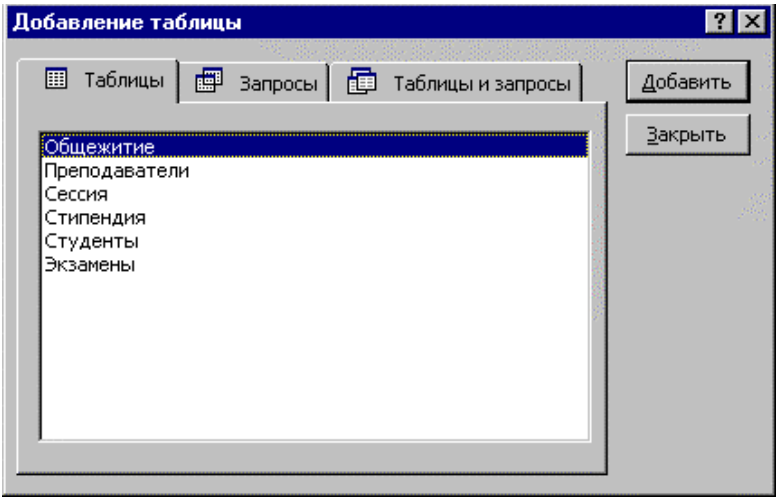


*Рис. 2.9. Окно **Добавление таблицы***

> *Диалоговое окно **Добавление таблицы** дает возможность добавить как таблицы, так и запросы. Иногда нужно определить связи между таблицами и запросами или только между запросами, чтобы Access знал, как правильно объединять эти объекты.*

Чтобы определить связь между таблицами, находящимися в окне **Схема данных**, следует перенести с помощью мыши поле связи главной

таблицы и поместить его на поле связи подчиненной таблицы. Откроется диалоговое окно **Связи (**см. рис. 2.10).

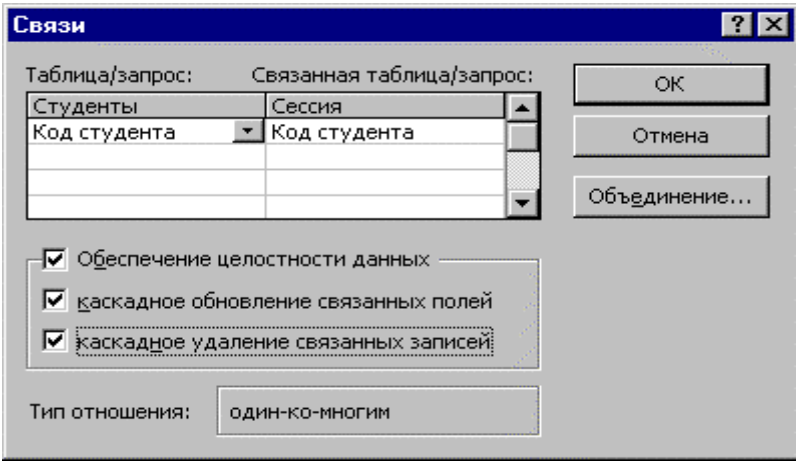


Рис. 2.10. Окно ***Связи***

В левом столбце выводятся имена главной таблицы и ключа, используемого для связи, а в правом столбце — имена подчиненной таблицы и внешнего ключа.

Для изменения поля следует открыть список полей справа от его имени. Если связь производится по нескольким полям, то их имена можно добавить, используя пустые строки.

Обычно Access сам определяет тип создаваемой связи, проводя анализ полей, для которых определяется связь. Если только одно из полей является ключевым или имеет уникальный индекс, создается ·связь «один-ко-многим». Связь «один-к-одному» создается в том случае, когда оба связываемых поля являются ключевыми или имеют уникальные индексы.

Создавая связь, нужно настроить режим *обеспечения целостности данных*. Обеспечение целостности позволяет избежать ситуации, когда в подчиненной таблице имеются записи, не связанные с записями главной таблицы. Если этот режим включен, то Access не разрешит добавить в подчиненную таблицу запись, для которой не найдется связанной с ней записи из главной таблицы. Нельзя будет также удалить из главной таблицы запись, имеющую связанные с ней записи в подчиненной таблице.

Например, нельзя будет добавить в таблицу **Сессия** запись с кодом студента, отсутствующим в таблице **Студенты**. Соответственно, из таблицы

Студенты нельзя удалить запись о студенте, пока в таблице **Сессия** содержатся сведения о его оценках.

> *Режим обеспечения целостности данных этой связи можно включить, если выполнены следующие условия:*
> - *поле связи главной таблицы является первичным ключом или имеет уникальный индекс;*
> - *связанные поля имеют один и тот же тип данных;*
> - *обе связанные таблицы принадлежат одной базе данных Access.*

Если для связи включен этот режим, то можно дополнительно указать, следует ли автоматически выполнять для связанных записей операции *каскадного обновления* и *каскадного удаления*. Если включить режим **Каскадное обновление связанных полей**, то при изменении значения ключа в главной таблице будут автоматически обновлены соответствующие значения в связанных записях подчиненной таблицы. При включении режима **Каскадное удаление связанных записей** при удалении записи из главной таблицы будут автоматически удалены связанные с ней записи в подчиненной таблице.

В том случае, когда эти режимы не включены, а режим обеспечения целостности данных включен, Access не позволит изменить значение в ключевом поле главной таблицы, а также удалить запись в главной таблице, если в подчиненной таблице имеются данные, связанные с этой записью.

После завершения операции создания связи нужно нажать кнопку **ОК**. Связь отображается в виде линии, соединяющей две таблицы. Если включен режим обеспечения целостности данных, то Access изобразит на конце линии, соответствующей главной таблице, цифру **1**. На другом конце линии, соответствующем подчиненной таблице, будет изображен символ бесконечности ∞ для связи типа «один-ко-многим» и цифра **1** для связи типа «один-к-одному».

> *Если перенести с помощью мыши поле, не являющееся ключевым или не имеющее уникального индекса, на другое поле, которое также не является ключевым или не имеет уникального индекса, будет создана связь неопределенного типа. Режим обеспечения целостности данных в этом случае включить нельзя.*

Любую связь можно изменить или удалить. Для изменения связи нужно сделать двойной щелчок по линии связи, и отредактировать ее в открывшемся окне **Связи**. Чтобы удалить связь, следует щелкнуть по ней и нажать клавишу **Delete.**

2.6. Модификация БД

Access позволяет достаточно просто внести изменения в БД. Однако, прежде чем вносить в БД изменения, нужно иметь в виду следующее:

- Access не вносит автоматически сделанные в таблицах изменения в использующие эти таблицы объекты (запросы, формы и т.д.).
- Нельзя изменять тип данных для поля, которое используется при определении связи между таблицами. Сначала нужно удалить соответствующую связь.
- Прежде чем открывать таблицу в режиме конструктора для внесения изменений, нужно закрыть все объекты, которые на нее ссылаются.

2.6.1. Операции над таблицами

Мы рассмотрим следующие операции над таблицей: копирование, переименование, удаление и экспорт. Все эти операции производятся в окне БД.

Копирование таблицы

Чтобы создать копию таблицы в БД, нужно вызвать ее контекстное меню и выбрать пункт **Копировать (**другой способ — выделить щелчком мыши таблицу и нажать клавиши **Ctrl+C** или щелкнуть по кнопке **Копировать**  панели инструментов). Затем нужно нажать клавиши **Ctrl+V** или щелкнуть по кнопке **Вставить**  панели инструментов. Access откроет диалоговое окно **Вставка таблицы** и предложит ввести имя новой таблицы; выбрать режим копирования. В зависимости от того, какой режим выбран, Access создаст пустую таблицу, имеющую такую же структуру, что и исходная таблица; создаст точную копию исходной таблицы (этот режим задан по умолчанию) или добавит данные из исходной таблицы в указанную таблицу.

Переименование таблицы

Для переименования таблицы нужно вызвать ее контекстное меню и выбрать пункт **Переименовать.** Затем нужно ввести новое имя таблицы и щелкнуть по кнопке **ОК**.

Удаление таблицы

Чтобы удалить ставшую ненужной таблицу в БД, нужно вызвать ее контекстное меню и выбрать пункт **Удалить (**другой способ — выделить щелчком мыши таблицу и нажать клавишу **Delete**). Access предложит подтвердить ваше желание удалить таблицу и в случае получения положительного ответа удалит ее.

Экспорт таблицы

Для экспорта таблицы Access в формат Excel или Word проще всего выполнить такие действия:

1. Щелчком мыши выделить нужную таблицу.
2. Выбрать пункт меню **Сервис, а** затем — **Связи с Office** или щелкнуть мышью по правому краю кнопки **Связи с Office**  .
3. Выбрать команду **Анализ в MS Excel** или соответственно команду **Публикация в MS Word**.

Access скопирует таблицу в электронную таблицу Excel (таблицу Word в формате RTF) с таким же именем. Ее первая строка будет содержать названия (подписи) полей экспортируемой таблицы. Если файл с таким именем уже существует, то Access предложит заменить этот файл или изменить имя нового файла.

Можно использовать более универсальный способ экспорта, позволяющий экспортировать таблицу Access в другую БД (Access, dBase, FoxPro, Paradox) или файл другого формата (текстовый, документ Word, таблица Excel, документ HTML). Для этого нужно выполнить следующие действия:

1. Вызвать контекстное меню таблицы или открыть меню **Файл** и выбрать пункт **Сохранить как/ экспорт…**
2. В диалоговом окне **Сохранение объекта** следует выбрать параметр **Во внешнем файле или базе данных** и нажать кнопку **OK**.
3. В поле со списком **Тип файла** выбрать формат БД или файла.
4. В поле со списком **Папка** выбрать диск и папку, в которую следует поместить экспортируемую таблицу.
5. В поле **Имя файла** нужно указать имя, которое получит экспортируемая таблица, и нажать кнопку **Экспорт**.

Экспорт в другую БД или в формат HTML Access осуществляет автоматически, создавая файл в заданном формате. Если этот формат не поддерживает длинные имена таблиц или кириллицу в названиях полей (FoxPro), то перед осуществлением операции следует внести изменения в структуру исходной таблицы.

Если задать экспорт таблицы в текстовый файл, то будет вызван мастер **Экспорт текста**, который позволит выбрать формат текста (Windows или DOS), настроить формат вывода дат и чисел и указать, какие поля нужно экспортировать. Мастер дает возможность создать текстовый файл с полями фиксированной ширины или с разделителями полей.

2.6.2. Модификация структуры таблицы

Все описанные ниже операции по модификации структуры таблицы производятся в режиме конструктора таблицы.

Вставка полей

Чтобы вставить в таблицу новое поле, нужно установить курсор на строку определения поля, перед которым вы хотите вставить новое поле. Затем следует щелкнуть по кнопке **Добавить строки**  или выбрать команды **Вставка/Строки**. Access вставит пустую строку, которую можно использовать для определения нового поля. Введите его имя, укажите тип и задайте нужные свойства.

Удаление полей

Для удаления поля нужно выделить строку его определения, щелкнув по области маркировки строки. Область выделения можно расширить, используя клавиши **Ctrl** и **Shift**. Затем нужно нажать клавишу **Delete** или щелкнуть по кнопке **Удалить строки**  . Если таблица содержит данные в удаляемом поле (полях), то Access и их потерю. Поэтому перед такой операцией имеет смысл на всякий случай создать копию таблицы.

Перемещение полей

Чтобы переместить поле, нужно выделить строку его определения и, держа нажатой мышь, переместить эту строку в другое место области определения полей.

Можно поступить иначе: с помощью клавиш **Ctrl+F8** включить режим **Сдвиг** (в строке состояния появится индикатор **СДВ**) и затем для перемещения поля использовать клавиши ↓ и ↑. Для выключения этого режима нужно нажать на клавишу **Esc**.

Аналогично можно переместить выделенную группу полей.

Копирование полей

Если несколько полей таблицы имеют близкие определения, то можно создать одно поле, а затем скопировать его нужное число раз. Для выполнения операции копирования поля нужно выделить целиком строку определения поля и скопировать в буфер обмена, используя клавиши **Ctrl+C** или кнопку **Копировать**  панели инструментов. Затем вставить пустую строку (см. пункт **Вставка полей**) в то место определения таблицы, где должна находиться копия поля, и выполнить операцию вставки из буфера обмена, используя клавиши **Ctrl+V** или кнопку **Вставить**  панели инструментов. После этого вносятся необходимые изменения в название и свойства нового поля.

Изменение типа данных и размера поля

Для изменения типа данных нужно щелкнуть по ячейке типа данных изменяемого поля. Затем щелкнуть по кнопке справа, открывающей список возможных типов данных, и выбрать новый тип данных. С некоторыми ограничениями Access может успешно преобразовать любой тип данных (кроме типа *поле объекта OLE*). Например, при преобразовании из текстового в числовой формат поле должно содержать только цифры и допустимые разделители.

Для текстового и числового поля задается его размер. Установленный размер поля можно затем изменить. Если он будет уменьшен, то Access выдаст предупреждающее сообщение о возможной потере некоторых данных. Если вы подтвердите свое желание сохранить сделанные изменения в определении таблицы, то операция будет выполнена, но это может повлечь за собой ошибки в преобразовании данных и их потерю. Поэтому перед такой операцией имеет смысл на всякий случай создать копию таблицы.

Глава 3. Работа с таблицей

Ввод, редактирование и просмотр информации в таблице может производиться либо в так называемом *режиме таблицы,* либо с помощью специально созданных для этих целей форм. В этой главе рассматривается работа в режиме таблицы. Для быстрого перехода в этот режим из режима конструктора достаточно нажать кнопку **Вид**  на панели инструментов, которая служит для переключения между этими режимами. В режиме таблицы записи располагаются одна под другой по строкам, а поля отображаются одно рядом с другим по столбцам (см. рис. 1.1).

3.1. Ввод данных

После создания структуры таблицы можно приступать к ее заполнению. Заметим сразу, что изменять данные в таблице можно в любой момент, поэтому не нужно заранее заботиться о полноте данных, их последовательности и т.д.

Если данных в таблице еще нет, то при ее открытии пользователь получает готовый для заполнения пустой бланк следующей структуры (см. рис. 3.1).

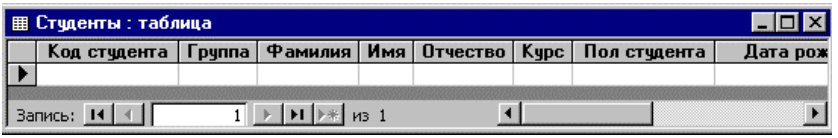


Рис. 3.1. Ввод данных в пустую таблицу

В первой строке бланка расположены имена полей таблицы в том порядке, в котором они были введены при построении структуры таблицы. В некоторых полях имеется дополнительная информация, зависящая от свойств поля, установленных по умолчанию или измененных пользователем. Пустая строка предназначена для ввода новой (в данном случае первой) записи в таблицу. Внизу бланка расположена информация о числе записей, имеющихся в таблице на данный момент, и расположении курсора. Крайний слева столбец бланка является индикаторным. В нем отражается информация о состоянии записей таблицы, а находящийся в нем в данный момент значок  означает текущую запись. Этот столбец также используется для маркировки записей. Щелкнув в нем, можно выделить всю запись целиком.

Такой же пустой бланк появится и при добавлении новых записей в таблицу, уже содержащую данные, если выполнить команды **Записи/Ввод**

данных. После завершения операции ввода для просмотра всех записей нужно выполнить команды **Записи/Удалить фильтр**. Для добавления новой записи — в таблицу, содержащую данные, можно также щелкнуть по кнопке **Новая запись**  или выполнить команды **Вставка/ Новая запись**.

Обычно данные в таблицу вводятся от записи к записи, а внутри записи по ее полям слева направо. Этот порядок не является обязательным.

При вводе данных в поле пользуются обычными правилами корректировки данных. В каждый момент времени вставка данных происходит в поле, в котором находится курсор. После ввода данных в поле можно перейти к следующему полю записи одним из следующих способов (при соответствующей настройке параметров Access):

- нажав клавишу перемещения курсора;
- нажав клавишу **Tab**;
- нажав клавишу **Enter**.

Если одна из этих клавиш нажата, когда курсор находится в последнем поле вводимой записи, осуществляется переход к следующей записи. При этом Access автоматически сохраняет на диске введенную запись. В дальнейшем любую из введенных записей можно редактировать.

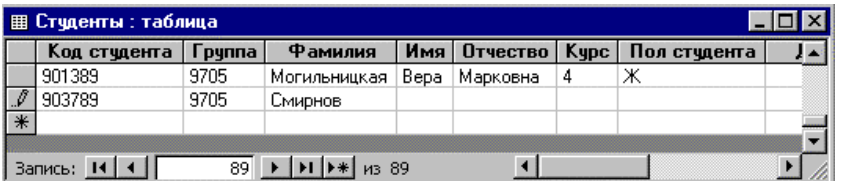


Рис. 3.2. Ввод новой записи

Если тип введенного значения не соответствует типу поля или нарушены условия на значение, заданные в свойствах поля, то Access не разрешит выйти из этого поля. Такая ситуация, например, возникает, если в поле типа **Дата** введено значение, которое Access не может интерпретировать как дату. В этом случае можно исправить допущенную ошибку или отменить ввод нажатием клавиши **Esc** или комбинации клавиш **Ctrl + Z**.

Access также не разрешит перейти к новой записи, если при вводе в ключевое поле нарушена уникальность его значений или не введены данные в поле, требующее обязательного ввода значения. Нужно исправить допущенные ошибки или отменить ввод новой записи двойным нажатием клавиши **Esc** или комбинации клавиш **Ctrl + Z**.

Рис. 3.2 характеризует ситуацию, когда в таблице уже есть введенные записи и осуществляется ввод очередной записи. Об этом говорит значок карандаша на столбце маркировки записи. Звездочкой помечена пустая запись.

После того как в таблицу введены все необходимые записи, таблицу можно просто закрыть, так как введенные записи и вся таблица сохраняются автоматически.

3.2. Перемещение по таблице

Для перемещения по таблице с помощью мыши можно использовать горизонтальную и вертикальную полосы прокрутки. В левой нижней части окна таблицы находится поле номера текущей записи (см. рис. 3.3), которое позволяет быстро перейти к нужной записи путем указания ее номера. Это поле содержит несколько кнопок перехода на первую, последнюю, следующую, предыдущую и новую запись. Для этой цели можно также воспользоваться командами **Правка**/**Перейти,** а затем выбрать нужный пункт открывающегося подменю (см. рис. 3.4).

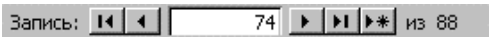


Рис. 3.3. Поле номера записи

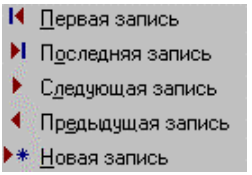


Рис. 3.4. Переход по записям

Для перемещения по таблице можно использовать и клавиатуру. Перечень клавиш прокрутки таблицы приводится в таблице 3.1, а клавиш перемещения между полями — в таблице 3.2.

Таблица 3.1. Клавиши прокрутки таблицы

Действие	*Клавиша*
Вверх на одну страницу	PgUp
Вниз на одну страницу	PgDn
Влево на одну страницу	Ctrl + PgUp
Вправо на одну страницу	Ctrl + PgDn

Таблица 3.2. Клавиши перемещения между полями таблицы

Действие	*Клавиша*
Переход на следующее поле	Tab, Enter
Переход на предыдущее поле	Shift + Tab
Переход на первое поле текущей записи	Home
Переход на последнее поле текущей записи	End
Переход на первое поле первой записи	Ctrl + Home
Переход на последнее поле последней записи	Ctrl + End
Переход на текущее поле предыдущей записи	↑
Переход на текущее поле следующей записи	↓
Переход на текущее поле первой записи	Ctrl + ↑
Переход на текущее поле последней записи	Ctrl + ↓
Переход на поле номера записи	F5

3.3. Редактирование данных в таблице

Таблица в любой момент доступна для редактирования. Движение курсора по полям и записям таблицы осуществляется с помощью отдельных клавиш или их комбинаций (см. табл. 3.2). При таком перемещении текст в поле выделяется, и при наборе новый текст в выделенном поле замещает уже имеющийся. После завершения набора текста следует нажать клавишу **Tab** или **Enter**.

Таблица 3.3. Клавиши, используемые в режиме редактирования

Действие	*Клавиши*
Удаление выделенного фрагмента или символа слева от позиции курсора	Backspace
Удаление выделенного фрагмента или символа справа от позиции курсора	Del
Вставка текущей даты	Ctrl + ; (точка с запятой)
Вставка в поле значения по умолчанию	Ctrl + Alt + [Пробел]
Вставка в поле значения этого поля в предыдущей записи	Ctrl + Э (' апостроф)
Добавление новой записи	Ctrl + + (плюс)
Удаление текущей записи	Ctrl + - (минус)
Сохранение изменений в текущей записи	Shift + Enter
Переключение режимов редактирования и перемещения	F2
Вставка новой строки	Ctrl + Enter

Чтобы внести изменения внутри содержимого поля, нужно перейти в режим редактирования. Для этого достаточно нажать клавишу **F2** или щелкнуть в нужном месте поля мышью. Ниже приведены основные комбинации клавиш, используемые в режиме редактирования поля (табл. 3.3), а также для перемещения внутри поля в этом режиме (табл. 3.4).

Таблица 3.4. Клавиши перемещения курсора в режиме редактирования

Перемещение курсора в поле	*Клавиши*
На один символ вправо	→
На одно слово вправо	Ctrl + →
На один символ влево	←
На одно слово влево	Ctrl + ←
В конец поля, содержащего одну строку	End
В конец поля, содержащего несколько строк	Ctrl + End
В начало поля, содержащего одну строку	Home
В начало поля, содержащего несколько строк	Ctrl + Home

Выделение данных в таблице

Часто редактированию данных предшествует процедура выделения данных, с которыми будет произведена та или иная операция редактирования. Существует множество способов выделения данных, часть из которых приведена ниже.

С помощью мыши выделение осуществляется с использованием действий, указанных в следующей таблице.

Таблица 3.5. Выделение полей и записей с помощью мыши

Выделение	*Установка указателя и нажатие кнопки мыши*
Данные в поле	В месте начала выделения. Для расширения области выделения перемещайте указатель при нажатой кнопке мыши.
Целое поле	На левой границе поля, где указатель принимает вид  .
Соседние поля	На левой границе поля. Для расширения области выделения перемещайте указатель при нажатой кнопке мыши.
Столбец	На заголовке столбца. Мышь должна принять вид жирной направленной вниз стрелки.
Соседние столбцы	На заголовке столбца. Для расширения области выделения перемещайте указатель при нажатой кнопке мыши.

Выделение	*Установка указателя и нажатие кнопки мыши*
Запись	На маркере выделения записи (левее первого поля). Мышь должна принять вид жирной направленной слева направо стрелки.
Несколько записей	На маркере выделения первой записи. Для расширения области выделения перемещайте указатель при нажатой кнопке мыши.
Все записи	Выберите команду **Правка/Выделить все записи.**

Те же действия можно осуществить с использованием клавиатуры.

Таблица 3.6. Выделение полей и записей с помощью клавиатуры

Область выделения	*Действие*
Данные в поле	Поместите курсор в начало выделяемого текста и, удерживая клавишу Shift, нажимайте клавиши перемещения курсора для перехода в конец выделяемого текста.
Целое поле	Поместите курсор в поле и нажмите клавишу F2.
Соседние поля	Выделите поле, нажмите клавишу Shift и, не отпуская ее, нажмите подходящую клавишу перемещения курсора.
Текущий столбец	Нажмите клавиши Ctrl + [Пробел].
Текущая запись	Нажмите клавиши Shift + [Пробел].
Несколько записей	Нажмите клавиши Shift + [Пробел], а затем клавиши Shift + ↑ или Shift + ↓

Таблица 3.7. Сдвиг границы выделения в поле с помощью клавиатуры

Сдвиг границы выделения	*Клавиши*
На один символ вправо	Shift + →
На одно слово вправо	Ctrl + Shift + →
На один символ влево	Shift + ←
На одно слово влево	Ctrl + Shift + ←

Таблица 3.8. Выделение полей и записей с помощью клавиатуры

Выделение текстового поля или записи	*Клавиши*
Выделение следующего поля	Tab
Переключение между режимом редактирования (с выведенным курсором) и режимом перемещения.	F2
Переключение из режима выбора текущей записи и первого поля текущей записи в режим перемещения	Shift + [Пробел]
Расширение границы выделения на предыдущую запись, если выбрана текущая запись	Shift + ↑
Расширение границы выделения на следующую запись, если выбрана текущая запись	Shift + ↓
Выделение всех записей	Ctrl + Ф (A)

Иногда требуется ввести в поле пустую строку, указывающую на отсутствие данных. Для этого в поле вводятся двойные кавычки (""). Поле сохранит пустое значение, хотя символы кавычек исчезают при переходе к другому полю или объекту. Если появится сообщение «*Поле не допускает ввод пустых строк*», то для этого поля необходимо изменить значение свойства **Пустые строки**.

Если видимая на экране часть поля не позволяет редактировать его с удобствами, можно нажатием комбинации клавиш **Shift+F2** открыть окно **Область ввода**. В этом окне текст, находящийся в поле, будет виден целиком.

Отмена изменений

Изменения, внесенные в содержимое полей текущей записи, можно отменить с помощью комбинации клавиш **Ctrl + Z,** а также команды **Правка**/**Отменить ввод** или кнопки на панели инструментов **Отменить ввод**  . Отмена сделанных изменений остается возможной и после перехода на другую запись до начала ее редактирования.

Использование буфера обмена Windows

При добавлении в таблицу данных, содержащихся в других таблицах БД, можно воспользоваться стандартным средством — буфером обмена Windows. При работе с буфером обмена применяются стандартные команды **Вырезать**, **Копировать** и **Вставить**, находящиеся в меню

Правка, а также соответствующие им комбинации клавиш **Ctrl + X**, **Ctrl + C, Ctrl + V** и кнопки  на панели инструментов.

Чтобы добавить в таблицу-приемник записи из таблицы-источника, нужно выполнить следующие действия:

1. Выделить копируемые записи в таблице-источнике и скопировать их в буфер обмена.
2. Открыть таблицу-приемник и выполнить команду **Правка/Добавить из буфера** или выделить последнюю (пустую) запись и выполнить команду **Вставить.**
3. После проверки возможности осуществления этой операции Access предложит подтвердить ее и затем добавит записи в таблицу.

Для замены записей таблицы-приемника записями из таблицы-источника нужно выполнить операцию копирования, а затем выделить заменяемые записи и выполнить команду **Вставить.** Access заменит выделенные записи записями из буфера обмена. Значения полей строки-источника будут вставлены в соответствии с порядком столбцов таблицы-**приемника**, независимо от имен полей. Поэтому, если таблицы имеют разную структуру, следует во избежание появления ошибок вставки (см. ниже) перед осуществлением операции внести необходимые изменения в очередность полей и их свойства.

Иногда требуется перенести данные, содержащиеся в прямоугольном блоке ячеек одной таблицы, в другую таблицу. В этом случае нужно выделить блок ячеек в таблице-источнике и выполнить команду **Копировать.** Затем следует выделить в таблице-приемнике такой же прямоугольный блок, как и в таблице-источнике, и выполнить команду **Вставить.** Если будет выделен блок меньших размеров, то Access вставит лишь часть данных из буфера обмена.

В ряде случаев Access отказывается выполнить операцию вставки данных из буфера обмена или выполняет ее частично. Перечислим типичные причины невыполнения этой операции:

- таблица-приемник имеет ключевое поле или поле с уникальным индексом, и осуществление операции приводит к нарушению уникальности его значений;
- таблица-приемник связана с другими таблицами и в результате операции нарушается целостность связей в БД;
- сделана попытка вставить содержимое поля или полей без указания места их назначения;
- вставка не может быть осуществлена из-за несоответствия типов данных (вставляется текст в числовое поле), нарушения условий на

значения полей (сделана попытка вставить в текстовое поле слишком длинный текст) и т.д.

Невставленные данные Access помещает в специальную таблицу **Ошибки вставки**. После корректировки они могут быть позднее вставлены в таблицу-приемник.

> ***Таблицу Ошибки вставки*** *можно использовать для быстрого создания таблицы, содержащей нужную информацию. Например, чтобы создать таблицу, содержащую сведения о студентах группы 9701, достаточно скопировать в буфер обмена соответствующие записи таблицы* ***Студенты****, предварительно отсортировав ее по полю* ***Группа****. Затем нужно выполнить операцию их вставки в эту же таблицу. Из-за нарушения условия уникальности значений ключевого поля* ***Код студента*** *Access откажется выполнить эту операцию и поместит записи из буфера обмена в таблицу* ***Ошибки вставки****, которая будет иметь такую же структуру, как и таблица* ***Студенты****.*

Буфер обмена можно использовать и при вставке в поля таблицы данных из других приложений Windows, например фрагмента таблицы Excel, текста, подготовленного в Word, или графического изображения.

3.4. Настройка внешнего вида таблицы

При работе с таблицей у различных пользователей базы данных могут возникнуть различные потребности, связанные с представлением содержащихся в них данных на экране. Например, разным пользователям может потребоваться различный порядок представления столбцов (полей таблицы), их ширина, представление данных в полях и т.д. Все эти элементы настройки (форматирования) таблицы находятся в руках пользователя. При изменении настроек сама структура таблицы не меняется, меняется лишь внешний вид ее представления на экране (макет). При желании новый макет таблицы может быть сохранен.

Наиболее часто встречающаяся операция при настройке внешнего вида таблиц — это выбор представления отдельных ее полей.

Что в основном меняет пользователь для отдельного поля:

- имя поля;
- представление данных в поле (в том числе тип поля);
- ширину поля;
- местонахождение (порядковый номер) поля.

Эти операции могут быть осуществлены различными способами. Имя поля и представление данных в поле лучше всего менять в конструкторе таблиц, тогда как ширину полей и порядок их следования— в режиме просмотра таблицы.

О задании имен полей и их характеристиках (типе данных, формате данных и др.) подробно говорилось в предыдущей главе, поэтому остановимся кратко на изменении ширины поля и порядке следования полей.

Для изменения ширины поля достаточно в режиме просмотра таблицы установить курсор мыши на правую границу имени поля, нажать левую клавишу мыши и переместить границу в заданном направлении для увеличения или уменьшения ширины поля. Можно также использовать соответствующую команду меню **Формат** (см. рис. 3.5).

Аналогичным образом может быть изменена высота строк таблицы. При этом меняется высота сразу всех строк таблицы.

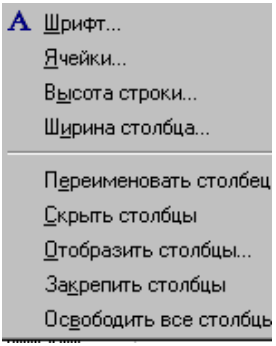


Рис. 3.5. Меню Формат

Для изменения порядка следования полей необходимо сначала выделить поле, щелкнув мышью на имени поля. Поле (весь столбец) окажется выделенным. После этого надо «ухватиться» мышью за имя поля и перетащить поле на новое место, удерживая нажатой левую клавишу мыши. Можно также с помощью клавиш **Ctrl+F8** включить режим **Сдвиг** и затем для перемещения поля использовать клавиши ↓ и ↑. Для выключения этого режима нужно нажать на клавишу **Esc**.

В режиме просмотра таблицы можно также скрыть показ отдельных столбцов. Для этого достаточно выбрать столбец (столбцы), щелкнуть правой клавишей мыши и из контекстного меню выбрать опцию **Скрыть столбцы**. Можно также использовать команды **Формат/Скрыть столбцы**. Для показа скрытых столбцов нужно использовать команды **Формат/Отобразить столбцы**.

Иногда при просмотре данных может понадобиться, чтобы при прокрутке содержимого таблицы вправо или влево один или несколько столбцов оставались на экране. Например, при просмотре сведений о студентах удобно, чтобы столбец с фамилией студента не исчезал с экрана. Для этого нужно выделить один или несколько (смежных) фиксируемых столбцов и выполнить команды **Формат/Закрепить столбцы.** Для отмены фиксации столбцов следует выполнить команды **Формат/Освободить все столбцы.**

3.5. Поиск и замена данных

В Access существует множество способов отобрать только требуемые данные при выполнении поиска конкретного значения, одной записи или группы записей.

- С помощью диалогового окна **Поиск** легко найти конкретные записи или определенные значения в полях. При обнаружении каждого вхождения требуемого элемента выполняется перемещение по записям. Если нужно заменить конкретные обнаруженные при поиске значения, следует воспользоваться диалоговым окном **Замена**.
- Фильтры позволяют временно выбрать и просмотреть конкретный набор записей в режиме таблицы.
- Запросы дают возможность работать с конкретным набором записей, которые удовлетворяют условиям, заданным для одной или нескольких таблиц базы данных. При выполнении запроса становится возможной независимая работа с этим набором записей в конкретной форме или в объекте в режиме таблицы.

О запросах будет подробно сказано в главе 5. В этой главе мы сосредоточимся на выполнении операций поиска и замены данных, а также на построении фильтров.

Для нахождения (поиска) данных служит специальное диалоговое окно поиска (см. рис. 3.6), открыть которое можно с помощью команд меню **Правка/Найти** или нажатия комбинации клавиш **Ctrl + A (F)**.

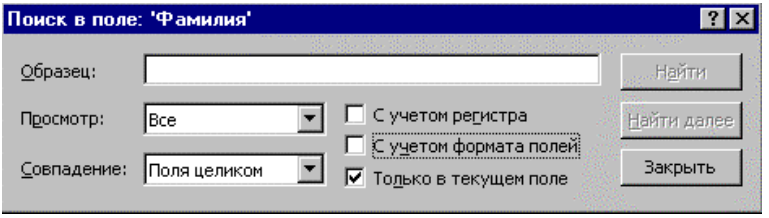


Рис. 3.6. Окно поиска

При этом можно найти конкретное вхождение образца в поле, а также возможность найти сразу все вхождения образца в поле или искать каждое вхождение по отдельности. Если не требуется выполнять поиск по всем полям, в режиме таблицы требуется выбрать поле (столбец) для поиска. значения, которое требуется найти. Само значение вводится в поле **Образец**.

Если точное значение неизвестно, можно задать образец поиска с помощью обычных знаков подстановки.

Чтобы найти первое вхождение указанного значения, необходимо нажать кнопку **Найти**. Для поиска следующих вхождений этого значения нажимается кнопка **Найти далее**.

Для замены данных служит диалоговое окно замены (см. рис. 3.7), открыть которое можно с помощью элемента головного меню **Правка/Заменить** или нажатия комбинации клавиш **Ctrl + P (H).**

Для замены найденного значения (значений) требуется задать информацию в поле **Заменить на.**

Существует возможность сразу заменить все вхождения образца в поле или заменять каждое вхождение по отдельности. Однако поиск пустых значений и пустых строк необходимо выполнять в диалоговом окне **Поиск** с последующей заменой этих значений вручную.

Для того чтобы выполнять замену каждого вхождения по отдельности, нужно нажимать **Найти далее**, а затем кнопку **Заменить**. Для пропуска текущего вхождения и поиска следующего нажимается кнопка **Найти далее**.

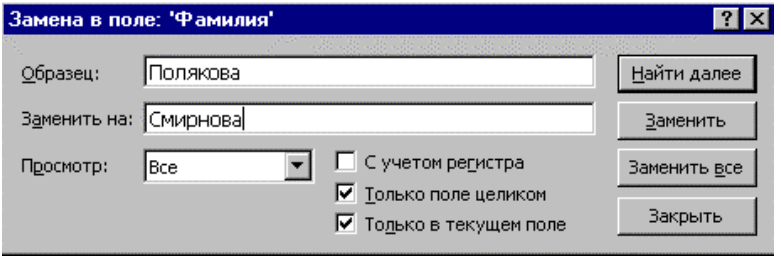


Рис. 3.7. Окно замены

Быстрее выполнить замену большого количества данных или произвести вычисления с данными (например, повысить все оклады сотрудникам) можно с помощью запроса на обновление (см. главу о запросах), а не с помощью диалогового окна **Замена.** Однако при этом невозможно выполнять замену отдельных вхождений с подтверждением. Кроме того, запрос на обновление при выполнении поиска и замены данных в нескольких полях становится достаточно громоздким.

Кроме поиска по образцам данных существуют и другие возможности поиска, например, записи по ее номеру.

3.6. Сортировка и фильтрация данных

Сортировка данных в таблице позволяет изменить порядок следования записей. При этом структура таблицы не меняется. Сортировка используется в тех случаях, когда требуется, например, выдать список сотрудников, упорядоченный по алфавиту, возрасту или окладу. Сортировка касается всех записей таблицы одновременно.

Для сортировки по отдельному столбцу достаточно в режиме таблицы установить в нем курсор мыши и выбрать команду сортировки по возрастанию (от А до Я) или по убыванию (от Я до А) из меню **Записи**. Эту же команду можно выполнить с использованием кнопок **Сортировка по возрастанию**  и **Сортировка по убыванию**  .

Если мы хотим произвести сортировку по группе рядом стоящих столбцов, то выделяется вся эта группа столбцов и выбирается команда сортировки аналогично вышеописанному. При этом вначале сортируются данные в самом левом столбце, затем в соседнем справа и т.д. Все столбцы сортируются в одном направлении. Понятно, что такая процедура сортировки имеет свои неудобства — сортируются только соседние столбцы и вся сортировка ведется в одном направлении. Если с первым недостатком легко справиться (достаточно переставить столбцы, чтобы они оказались рядом), то сортировка в различном направлении по различным полям возможна только при фильтрации или конструировании запросов.

Фильтрация данных позволяет не только выбрать различный порядок сортировки по различным полям (это побочный эффект фильтрации данных), но, что самое главное, выбрать из всех записей таблицы только те, которые удовлетворяют заданным критериям.

Принципиальное отличие фильтров от запросов состоит в том, что фильтры позволяют построить временные наборы данных и сами являются временными конструкциями, создаваемыми на момент работы с конкретной таблицей. Запросы являются объектами базы данных и хранятся вместе с таблицами и другими объектами. Любой последующий фильтр удаляет предыдущий, тогда как запросы могут создаваться независимо и в большом числе.

Фильтр создается только для одной таблицы, и выбираемые записи включаются в результирующий набор со всеми своими полями; тогда как запрос может быть построен для группы взаимосвязанных таблиц, а в результат запроса могут включаться лишь отдельные поля записей.

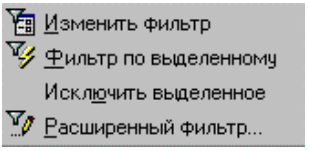


Рис. 3.8. Подменю ***Фильтр***

Задать критерии фильтрации можно несколькими способами. В пункте **Записи** главного меню есть пункт **Фильтр** (см. рис. 3.8) Его команда **Фильтр по выделенному** обеспечивает выполнение самого простого варианта фильтра. Для этой же цели служит кнопка **Фильтр по выделенному**  на панели инструментов.

Критерием отбора записей в этом варианте фильтра являются значения, содержащиеся в одной или нескольких (смежных) выделенных ячейках таблицы. Можно также произвести фильтрацию и по выделенной части значения ячейки. Если фильтр производится по полному значению ячейки, то ее даже не нужно выделять, а просто щелкнуть по ней курсором. Результатом фильтрации будет набор записей, содержащий в соответствующих полях значения, совпадающие с выделенными элементами данных.

Команда **Исключить выделенное** также производит фильтрацию записей, но в этом случае критерий отбора противоположный: ищутся все записи со значениями полей, несовпадающими со значениями выделенных ячеек.

К набору, полученному в результате фильтрации, при необходимости можно снова применить фильтр.

Пусть, например, требуется отобрать в таблице **Студенты** сведения о девушках 9703 группы. Для этого нужно найти в поле **Группа** ячейку со значением *"9703"* и щелкнуть по кнопке **Фильтр по выделенному**. На экране останутся записи, относящиеся к студентам 9703 группы. Затем следует щелкнуть по ячейке поля **Пол** со значением *"Ж"*, и снова щелкнуть по этой же кнопке. (Можно поступить иначе: щелкнуть по ячейке поля **Пол** со значением "*М*", затем щелчком правой кнопки мыши вызвать контекстное меню и выбрать команду **Исключить выделенное.)** В итоге на экране останутся лишь требуемая информация.

Вернуть на экран данные исходной таблицы можно командой **Записи/Удалить фильтр** или щелкнув по кнопке **Удалить фильтр**  на панели инструментов. Отметим, что эта команда не удаляет из памяти использованный критерий фильтра, а лишь отменяет его действие.

Снова включить режим фильтрации можно, выполнив команду **Записи/Применить фильтр** или щелкнув по кнопке **Применение фильтра,** которая имеет такой же вид, как и кнопка **Удалить фильтр.**

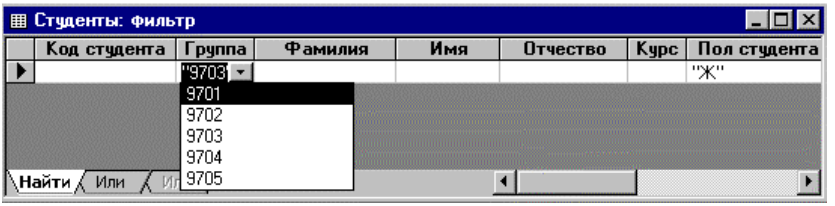


Рис. 3.9. Изменение обычного фильтра

В фильтр, созданный последним, можно внести изменения и затем для отбора записей использовать новый фильтр.

Команда **Изменить фильтр** или нажатие кнопки  с таким же именем вызывает следующий экран (см. рис. 3.9). В каждом поле можно нажать кнопку списка и в появившемся списке значений этого поля выбрать нужное для задания критерия. Перед значениями полей можно задать операции отношений (>, <, >=, <=, =, <>) или же удалить часть значения. После набранного таким образом критерия, нужно выполнить команду **Применить фильтр**.

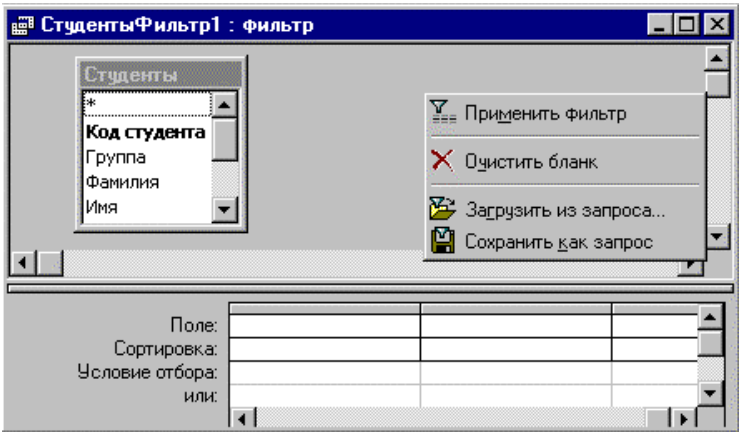


Рис. 3.10. Бланк расширенного фильтра с контекстным меню

Например, если нам понадобится после просмотра сведений о девушках 9703 группы вывести информацию о девушках 9701 группы, можно использовать только что созданный фильтр, внеся в него соответствующие изменения. Для этого нужно открыть список значений в поле **Группа**, выбрать нужный номер группы и выполнить новый фильтр.

Команда **Расширенный фильтр** позволяет задать критерии фильтрации с помощью бланка запроса и позволяет понять основы

построения запросов, да и сам фильтр может быть с помощью специальной команды преобразован в запрос. При выборе этой команды на экране появляется бланк построения фильтра, в верхней части которого находится список полей таблицы, а в нижней — бланк для построения критериев фильтрации (см. рис. 3.10).

При необходимости построить критерий выбора или просто отсортировать данные в таблице по какому-либо полю это поле переносится в бланк (например, с помощью перетаскивания мышью или двойным щелчком левой кнопки мыши по имени поля), а в строке **Условие отбора** строится критерий фильтрации. Построение критериев фильтрации полностью аналогично построению критериев в запросах (см. главу 5) и здесь не приводится. В строке сортировки можно задать сортировку в любом направлении. После заполнения бланка расширенного фильтра можно осуществить фильтрацию простым нажатием соответствующей кнопки на панели инструментов или вызвать выбором команды **Применить фильтр** контекстного меню.

Глава 4. Выражения в Access

При работе с различными объектами в Access широко используются выражения — аналог формул в Excel**.** *Выражение* — это любая комбинация операторов, констант, функций и идентификаторов[1], результатом которой является некоторое значение. Константы, функции и идентификаторы, используемые в выражениях, называются *операндами*.

Выражения часто употребляются для проверки различных условий и проведения вычислений в таблицах, запросах, формах и отчетах. Они позволяют выполнять действия с числами, датами и текстовыми значениями в каждой записи, используя данные из одного или нескольких полей. Например, с помощью выражения можно перемножить значения двух числовых полей или объединить несколько текстовых значений.

Несколько примеров выражений было приведено ранее в главе 2 при описании условий на значения полей таблицы. Кроме примеров, содержащихся в этой главе, различные выражения часто встречаются и в последующих главах пособия.

4.1. Операторы

В выражениях применяются следующие типы операторов:

- *арифметические* операторы — используются для выполнения математических вычислений;
- операторы *конкатенации* — используются для слияния строк;
- операторы *сравнения* — используются для выполнения операций сравнения;
- *логические* операторы — используются для выполнения логических операций;
- операторы *идентификации* — создают однозначные имена объектов БД.

4.1.1. Арифметические операторы

Операнды должны быть выражениями, имеющими числовое значение. Для изменения приоритета арифметических операций используются круглые скобки. Если хотя бы один из операндов является выражением со значением *Null*, то результат имеет значение *Null.*

---

[1] *Идентификатор* — это элемент выражения, определяющий ссылку на значение поля, элемента управления или свойства.

Оператор деления нацело «\» округляет оба операнда до целых значений, а затем делит первый на второй. Результат округляется до целого, например, 11 \ 2 = 5; 7,6 \ 2,5 = 4.

Оператор *Mod* также округляет оба операнда до целых значений и делит первый на второй. Результат — остаток от деления. Например, 9 Mod 2 = 1, а 7,6 Mod 4 =0.

Таблица 4.1. Арифметические операторы

Оператор	*Описание*	*Пример*
+	Складывает два операнда	[Цена] + 10
–	Вычитает из первого операнда второй или меняет знак операнда	[Дата1] – [Дата2] –111
*	Перемножает два операнда	[Цена]*[Вес]
/	Делит один операнд на второй	[Сумма] / 10
\	Делит один операнд на второй нацело	[Месяц] \ 4
^	Возводит первый операнд в степень, задаваемую вторым операндом	[Число] ^ [Степень]
Mod	Возвращает остаток от деления нацело	[Месяц] mod 4

4.1.2. Операторы слияния строк (конкатенации)

Операторы **&** (амперсанд) или **+** создают текстовую строку, присоединяя содержимое второй строки к концу первой. Если один из операндов — число, то он преобразуется перед проведением операции слияния в строку символов.

Для объединения строк лучше использовать оператор &, а не +, так как если одна из участвующих в операции строк, например <строка 2>, имеет значение *Null*, то результат операции <строка 1> + <строка 2> равен *Null*, , а результат операции <строка1> & <строка 2> равен <строка 1>.

Например, в выражении "*Число заказов = " & [Число заказов]* объединяются строка символов и значение поля **Число заказов**. Если число заказов равно 100, то результатом выполнения операции будет строка "*Число заказов = 100"*.

4.1.3. Операторы сравнения

Оператор сравнения сравнивает значения двух операндов и возвращает в качестве результата одно из логических значений: *True* или *False*. Если хотя бы один из операндов является выражением со значением *Null*, то результат имеет значение *Null*.

Кроме стандартных операторов сравнения, приведенных в таблице 4.2, в Access имеется еще четыре дополнительных оператора сравнения: *Is*, *In*, *Between* и *Like*, которые обычно используются для проверки условия на значение в поле или в условиях отбора записей в запросе.

Оператор Is

При использовании вместе с *Null* определяет, является ли значение *Null* или *Not Null*. Например, *Is Null* применяется для отбора записей, имеющих в данном поле значение Null, а *Is Not Null* — для отбора записей, имеющих в данном поле непустое значение (см. пример 5.6).

Таблица 4.2. Операторы сравнения

Оператор	*Описание*	*Пример*	*Результат*
<	Меньше, чем	1+2 < 3+4	True
<=	Меньше или равно	1 <= 3/5	False
>	Больше, чем	1 > 0	True
>=	Больше или равно	0 >= 1	False
=	Равно	1 = 1	True
<>	Не равно	1 <> 1	False

Оператор In

Осуществляет проверку значения на совпадение с элементом из заданного списка. Элементы списка отделяются друг от друга точкой с запятой. Текстовые значения должны браться в кавычки, например, *In("Москва";"Киев";"Минск")* или *In(2;4;6;8).*

Оператор Between

Осуществляет проверку, находится ли числовое значение внутри заданного диапазона. Например, *Between 10 And 20* означает, что значение должно находиться в интервале [10, 20].

Таблица 4.3. Спецсимволы, используемые в операторе Like

Спецсимвол	*Совпадающие символы*
?	Любой одиночный символ;
#	Любая одиночная цифра (0-9);
*	Любое число символов или их отсутствие;
[список]	Любой одиночный символ, входящий в список;
[!список]	Любой одиночный символ, не входящий в список.

Оператор Like

Осуществляет проверку значения на соответствие заданному шаблону.

Таблица 4.4. Примеры использования оператора *Like*

Условие	*Комментарий*
Like "А*ов"	Любой текст, начинающийся с буквы «А» и заканчивающийся буквами «ов»;
Like "К??#"	Значение должно содержать четыре символа; начинаться с буквы К и заканчиваться цифрой.
Like "[А-ВК]*"	Любой текст, начинающийся с букв А, Б, В и К;
Like "[!П-СЯ]*"	Любой текст, не начинающийся с букв П, Р, С и Я.

В таблице 4.3 перечислены специальные символы, используемые в шаблоне, и соответствующие им символы в сравниваемом выражении. При проверке совпадения символов их регистр роли не играет. Чтобы включить в список диапазон символов, нужно указать первый символ, знак дефиса и затем последний символ, например [К-Р].

4.1.4. Логические операторы

Обычно применяются для объединения двух или нескольких условий в единое целое. Ниже приведены наиболее часто используемые логические операторы.

Таблица 4.5. Логические операторы

Оператор	*Назначение*
And	Должны выполняться все условия;
Or	Должно выполняться хотя бы одно из условий;
Not	Не должно выполняться данное условие.

Результат выполнения операции логического умножения *And* равен *True*, если значения всех операндов равны *True*; в противном случае он равен *False*.

Результат выполнения операции логического *Or* равен *True*, если значение хотя бы одного операнда равно *True;* в противном случае он равен *False*.

Результат выполнения операции логического отрицания *Not* равен *True*, если значение операнда равно *False;* в противном случае он равен *False.*

4.1.5. Операторы идентификации. Идентификаторы

Часто в выражениях используются значения полей таблиц, элементов управления форм и других объектов БД. Имена полей в разных таблицах или элементов управления в формах могут совпадать. Чтобы Access правильно вычислил значение выражения, необходимо обеспечить однозначность ссылок в выражениях на объекты БД и их свойства.

Access использует два оператора идентификации "!" (восклицательный знак) и "." (точка).

Оператор "!"

Чаще в идентификаторах встречается оператор "!". Он используется для ссылок на объекты[1]. При ссылке на поле таблицы он служит для отделения имени поля от имени таблицы. Сами имена заключаются в квадратные скобки, и ссылка имеет следующий вид:

[<имя таблицы>]![<имя поля>].

Это так называемая полная форма записи идентификатора поля таблицы. Если нет неопределенности в ссылке, то допустима и неполная форма записи идентификатора поля таблицы в виде *[<имя поля>]*. Так, в запросах, использующих одну таблицу, обычно используется неполная ссылка на поле. Например, полная ссылка на поле **Фамилия** в таблице **Студент** имеет вид *[Студент]![Фамилия]*, а неполная — *[Фамилия]*.

Соответственно, ссылка на элемент управления формы (главной формы, если она содержит подчиненную форму) имеет такую полную форму записи:

Forms![<имя формы>]![<имя элемента управления>].

Здесь имени формы предшествует имя семейства открытых форм *Forms*, которому принадлежит данная форма. Эта «добавка» вызвана тем обстоятельством, что БД вполне может содержать таблицу и форму с одинаковыми именами, имеющих к тому же одноименные поля.

В общем случае синтаксис оператора «!» таков:

<класс объекта>!<имя объекта>

Оператор "."

Оператор "." (точка) обычно используется для ссылок на свойства форм, отчетов и элементов управления. В частности, идентификатор поля со списком в форме выглядит так:

---

[1] В SQL и VBA для этой цели обычно используется оператор ".".

Forms![<имя формы>]![<имя поля со списком>].[Text].

Здесь точка "." используется для отделения имени поля со списком от его свойства *Text,* которое возвращает текущее значение поля. В общем случае синтаксис оператора "." таков:

<класс объекта>!<имя объекта>.<свойство объекта>

Подробную информацию о синтаксисе идентификаторов различных объектов можно найти в соответствующих разделах справки Access. Отметим лишь еще, что разработчики Access рекомендуют всегда ссылаться на объект или свойство, используя полную форму идентификатора. Если при создании выражения пользоваться построителем выражений, то эта рекомендация обычно автоматически выполняется.

4.2. Константы

В выражениях встречаются константы следующих типов:

- *Числовые* константы (числа) — последовательность цифр, содержащая, если нужно, знак числа и разделитель целой и дробной части числа. В качестве разделителя в зависимости от установок Windows обычно используются "," (запятая) или "." (десятичная точка). Числа могут содержать символ **E** или **e** и знак порядка, например, 1,2E+04 = 12 000.
- *Текстовые* константы (строки) — могут содержать любые символы из набора символов кодовой таблицы ANSI. В выражениях строки нужно с обеих сторон заключать в прямые кавычки ("). Обычно Access добавляет их сам.
- Константы типа *Дата/время.* Они должны быть заключены в знаки номера "#". Обычно Access добавляет их сам, если опознает, что вводится дата в одном из стандартных форматов "дд.мм.гг" или "дд/мм/гг".

4.3. Функции

В состав выражений часто входят различные функции. Всего в Access и VBA определено более 160 функций. Здесь приводится краткое описание лишь части из них. В основном даны функции, используемые в примерах пособия и заданиях по Access (см. [6]).

4.3.1. Функции для работы с датами

- *Date()* — текущая дата. Может использоваться в формах и отчетах, а также задавать условие отбора для запроса. Например, *Date() – 1* задает дату, предшествующую текущей дате.
- *Day(дата* [1]*)* — день месяца, целое число от 1 до 31. Например, *Day(#10.08.99#)* возвращает номер дня, равный 10.
- *DatePart(интервал* [2]*; дата)* — возвращает указанный в аргументе <*интервал*> компонент даты, целое число. Например, *Datepart("q"; #15.09.98#)* возвращает число 3 — номер третьего квартала.
- *DateAdd(интервал; число; дата)* — возвращает значение, содержащее дату, вычисляемую по формуле <*дата*> + <*число*> * <*интервал*>. Аргумент <*интервал*> принимает такие же значения, как и в функции *DatePart*. Например, *DateAdd("m";2;[Дата1])* возвращает дату, отстоящую от значения даты в поле **Дата1** на два месяца.
- *Format(дата, формат* [3]*)* — дает дополнительные возможности для использования в выражениях дат и возвращает строку символов. Строка <*формат*> может объединять несколько базовых форматов и символьных строк, которые заключаются в двойные кавычки. Например, *Format(#22.03.73#; "dddd ""," d mmmm yy ""года""")* возвращает строку *четверг, 22 марта 73 года*.
- *Month(дата)* — месяц, целое число от 1 до 12. Например, *Month(#10.08.99#)* возвращает номер месяца, равный 8 (август).
- *Now()* — дата и время компьютера. Часто используется в отчетах, созданных с помощью мастеров Access.
- *Weekday(дата)* — день недели, целое число от 1 до 7, воскресенье равно 1. Например, *Weekday(#22.03.73#)* возвращает число 5 (четверг).
- *Year(дата)* — год, целое число. Например, *Year([Студенты]![Дата рождения])* возвращает год рождения студента.

---

[1] Аргумент <*дата*> должен иметь значение, представляющее дату.

[2] Аргумент <*интервал*> заключается в кавычки и должен иметь значение, определяющее тип возвращаемого временного интервала: *yyyy* — год, *q* — квартал, *m* — месяц, *y* — день года, *d* — день месяца, *w* — день недели, *ww* — неделя.

[3] Аргумент <*формат*> заключается в кавычки и в дополнение к значениям, приведенным в предыдущей сноске, может иметь следующие значения: *yy* — последние две цифры номера года, *ddd* — сокращенное название дня недели, *dddd* — полное название дня недели, *mm* — номер месяца из двух цифр, *mmm* — сокращенное название месяца, *mmmm* — полное название месяца.

4.3.2. Функции для работы со строками

- *Chr(код_символа)* — возвращает символ, соответствующий указанному коду символа из кодовой таблицы Windows ANSI. Например, *Chr(100)="*d", *Chr(200)*="И".
- *Left(строка, число_символов)* — указанное число первых символов строки. Например, *Left([Студенты]![Имя],1*) возвращает первую букву имени студента.
- *Len(строка)* — число символов в строке. Например, *Len([Фамилия])* дает число символов в фамилии, содержащейся в поле **Фамилия.**
- *Mid(строка; нач_символ; число_символов)* — возвращает подстроку, содержащую указанное число символов строки, начиная с указанного символа. Последний аргумент необязателен. Если он отсутствует, то возвращаются все символы начиная с указанного символа до конца строки. Например, *Mid("Студент Петров";9;4)* возвращает «Петр», а *Mid("Студент Петров";9)* возвращает «Петров».
- *Right(строка, число_символов)* — указанное число последних символов строки. Например, *Right([Код студента],3*) возвращает последние три символа кода студента.
- *Trim(строка)* — удаляет пробелы в начале и конце строки символов.

4.3.3. Математические функции

- *Abs(выражение)* — возвращает абсолютное значение числового аргумента *выражение.* Например, *Abs(-10) = 10.*
- *Int(выражение)* — возвращает целую часть числового аргумента <*выражение*>. Например, *Int*(5,2) = 5, а *Int*(-5,2) = -6.
- *Rnd()* — возвращает случайное число между 0 и 1.

4.3.4. Статистические функции

Возвращают в качестве значения результат соответствующей статистической операции над данными, содержащимися в указанном поле запроса, формы или отчета. Записи с пустыми (*Null*) значениями в поле в вычислениях не участвуют. Обычно используются в итоговых запросах, при создании вычисляемых полей и инструкциях SQL.

- *Avg(выражение*[1]*)* — вычисляет среднее арифметическое значений, содержащихся в указанном поле. Например, *Avg([Стипендия]![Сентябрь])* находит среднюю стипендию в сентябре.
- *Count(выражение)* — определяет число записей, возвращаемых запросом. Например, *Count([Стипендия]![Октябрь])* вычисляет число студентов, получивших стипендию в октябре.
- *First(выражение)* — возвращает значение, содержащееся в указанном поле первой записи результата запроса. Обычно результат запроса предварительно подвергается сортировке. Например, *First([Студенты]![Фамилия])* находит фамилию самого молодого студента, если записи в запросе отсортированы по убыванию в поле [**Дата рождения**].
- *Last(выражение)* — возвращает значение, содержащееся в указанном поле последней записи результата запроса. Например, *Last([Студенты]![Фамилия])* находит фамилию самого молодого студента, если записи в запросе отсортированы по возрастанию в поле **Дата рождения**.
- *Max(выражение)* — подсчитывает максимальное из значений, содержащихся в указанном поле. Например, *Max([Студенты]![Дата рождения])* находит дату рождения самого молодого студента.
- *Min(выражение)* — подсчитывает минимальное из набора значений, содержащихся в указанном поле. Например, *Min([Книги]![Цена])* находит минимальную из цен на книги.
- *StDev(выражение)* — возвращает значение несмещенной оценки стандартного отклонения значений, содержащихся в указанном поле.
- *Sum(выражение)* — подсчитывает сумму значений, содержащихся в указанном поле. Например, *Sum([Заказы]![Количество]*[Книги]![Цена])* вычисляет суммарную стоимость заказов.
- *Var(выражение)* — возвращает значение несмещенной оценки дисперсии значений, содержащихся в указанном поле.

[1] Во всех приведенных в этом пункте функциях аргумент <*выражение*> идентифицирует поле с данными для статистической операции. Им может быть как уже существующее поле таблицы (запроса), так и вычисляемое поле. В последнем случае выражение, задающее правила вычисления его значений, не должно содержать других статистических функций.

4.3.5. Статистические функции по подмножеству

Также позволяют вычислить статистические характеристики данных, содержащихся в указанном поле набора записей (таблицы или запроса). Однако синтаксис описываемых ниже функций дает возможность отобрать из этого набора с помощью логического условия подмножество записей и тем самым сузить область действия статистической операции. Другое важное отличие от функций, рассмотренных в предыдущем пункте, состоит в том, что статистические функции по подмножеству можно использовать для отбора нужных записей в выражении, содержащемся в строке условий запроса, а также в макросах и модулях.

- *DAvg(выражение; набор[; условие]*[1]*)* — подсчитывает среднее арифметическое значений, содержащихся в указанном поле заданного подмножества записей. Например, *DAvg(“[Ноябрь]”; ”Стипендия”)* вычисляет среднюю стипендию в ноябре.
- *DCount(выражение; набор[;[условие])* — определяет число записей в указанном подмножестве записей. Так, *DCount("[Код студента]"; "Студенты"; "[Группа] = '9701' ")* находит количество студентов в учебной группе 9701.
- *DLookUp(выражение*; *набор[; условие])* — возвращает значение указанного поля в заданном подмножестве записей. Например, *DLookUp("[Оценка]"; "Сессия";"[Код студента] = '881375' and [Код экзамена] = '1' ")* возвращает оценку студента Иванова по информатике.
  Обычно задаются условия, обеспечивающие уникальность значения, возвращаемого функцией *DLookup*. Если условиям отбора удовлетворяет несколько записей, то *DLookup* возвращает значение

---

[1] Все приведенные в этом пункте статистические функции по подмножеству имеют следующие аргументы:

- <*выражение*> — текстовое выражение, идентифицирующее поле с данными для статистической операции. Им может быть как уже существующее поле таблицы (запроса), так и вычисляемое поле. В последнем случае выражение, задающее правила вычисления его значений, не должно содержать других статистических функций.
- <*набор*> — текстовое выражение, определяющее набор (подмножество) записей (таблица или запрос),
- <*условие*> — текстовое выражение, задающее диапазон записей, для которых должна производиться статистическая операция. Текстовые значения в аргументе <*условие*> нужно заключать в одинарные кавычки (').
  Этот аргумент может отсутствовать. Если аргумент <*условие*> опущен, то статистическая операция выполняется над полем, заданным в аргументе <*выражение*>, для всего набора записей, указанного в аргументе <*набор*>.

заданного поля первой из них. Если же ни одна из записей набора не удовлетворяет аргументу <*условие*>, то функция *DLookup* возвращает значение *Null.*

- *DSum(выражение; набор[; условие])* — подсчитывает сумму значений, содержащихся в указанном поле заданного подмножества записей. Так, *DSum("[Количество]"; "Заказы"; "[Код книги]=" & [введите код])* находит число заказанных экземпляров данной книги.
- *DMax(выражение; набор[; условие])* — подсчитывает максимальное из набора значений, содержащихся в указанном поле заданного подмножества записей. Например, *DMax("[Цена]";"Книги"; "[Серия] ='В подлиннике'"),* возвращает максимальную из цен на книги серии *"В подлиннике".*
- *DMin(выражение; набор[; условие])* — подсчитывает минимальное из набора значений, содержащихся в указанном поле заданного подмножества записей.
- *DFirst(выражение; набор[; условие])* — возвращает значение, содержащееся в указанном поле первой записи заданного подмножества записей.
- *DLast(выражение; набор[; условие])* — возвращает значение, содержащееся в указанном поле последней записи заданного подмножества записей.
- *DStDev(выражение; набор[; условие])* — возвращает значение несмещенной оценки стандартного отклонения значений, содержащихся в указанном поле заданного подмножества записей.
- *DVar(выражение; набор[; условие])* — возвращает значение несмещенной оценки дисперсии значений, содержащихся в указанном поле заданного подмножества записей.

4.3.6. Другие полезные функции

- *IsNull(выражение)* — возвращает *True,* если <*выражение*> имеет значение *Null;* в противном случае функция возвращает значение *False.* Например, значение *IsNull([Стипендия]![Сентябрь])* равно *True*, если текущее значение поля пусто (данный студент не получает стипендию), и *False* в противном случае.
- *IIF(условие; выражение1; выражение2)* — возвращает значение <*выражение1*>, если *условие* равно *True* и <*выражение2*>, если *условие* равно *False.* Например, *IIF([Пол]="м"; "студент"; "студент-*

ка") имеет значение «*студент*», если в поле **Пол** содержится буква "м", и значение «*студентка*» — в противном случае.

- *Nz(выражение[; представление])* — возвращает 0 (нуль), пустую строку ("") или другое указанное в аргументе <*представление*> значение, если <*выражение*> имеет значение *Null*. Например, *Nz(([Стипендия]! [Сентябрь];"нет")* возвращает значение стипендии студента за сентябрь, если он в сентябре получал стипендию, или слово «*нет*» в противном случае.

  Аргумент <*представление*> необязателен. Если он отсутствует, то функция *Nz* возвращает нуль или пустую строку в зависимости от контекста, требующего числовое или текстовое значение.

4.4. Построитель выражений

При создании выражений для таблиц, запросов и других объектов Access следует использовать *построитель выражений*. Для вызова построителя нужно сначала щелкнуть по ячейке, в которую будет вводиться выражение, а затем по кнопке **Построить**  на панели инструментов или по кнопке , обычно появляющейся справа от ячейки ввода. На экране появится окно **Построитель выражений (**см. рис**.** 4.1**)**, содержащее четыре поля.

В верхнем поле располагается создаваемое выражение, а три нижних используются для выбора нужных элементов. Для выбора любого элемента в этих полях нужно сделать двойной щелчок по соответствующему имени.

Левое поле отображает иерархию папок, содержащих основные типы компонентов выражений. После выбора элемента (папки) из левого поля в среднем поле будет выведен либо список его элементов (поля таблицы или запроса), либо список подтипов (элементы управления формы, категории функций и т.п.). При выборе подтипа в правом поле появится список его элементов (поля, функции, свойства элементов управления).

Чтобы ввести в формируемое выражение ссылку на имя поля таблицы или запроса, нужно выбрать в левом поле таблицу или запрос, а затем в среднем поле нужное поле.

Для ввода функции следует выбрать в левом поле папку **Функции**, а затем **Встроенные функции.** В среднем поле нужно выбрать категорию или вариант <**Все**>, а затем, прокрутив список в правом поле, — нужную функцию.

Для ввода оператора (+, >, And и др.) щелкните по соответствующей кнопке в окне построителя. Если требуемого оператора на кнопках нет, следует открыть в левом поле папку **Операторы**. Затем в среднем поле выбрать категорию или вариант <Все>, а в правом поле — нужный оператор.

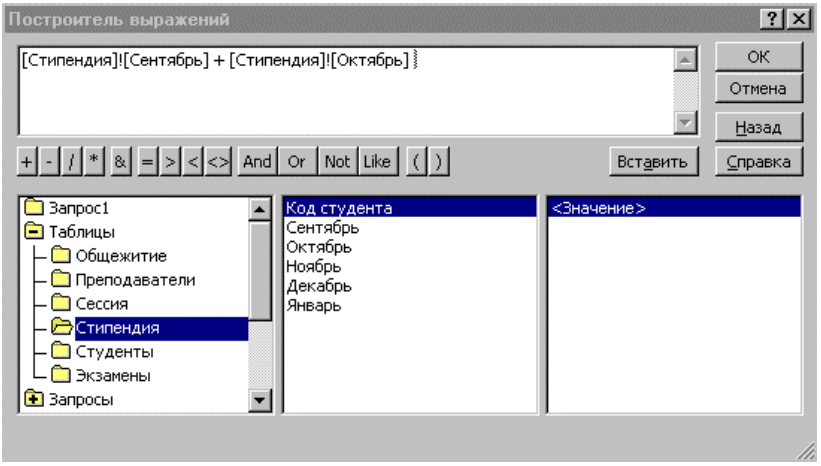


Рис. 4.1. Окно ***Построитель выражений***

Access часто вставляет в создаваемое выражение вместе с выбранным элементом один или несколько прототипов, заключенных в кавычки («выражение», «number» и т.п). В этом случае нужно либо ввести вместо прототипа соответствующее значение, либо выделить прототип и заменить его элементом из правого списка, либо просто удалить его.

Для вставки элемента в выражение можно использовать кнопку **Вставить**. Чтобы отменить ошибочный ввод, нужно щелкнуть по кнопке **Назад**. Создание выражения завершается нажатием кнопки **ОК**.

Пример 4.1. Нужно создать выражение, которое подсчитывает суммарную стипендию, полученную каждым студентом за сентябрь и октябрь. Информация о стипендии содержится в таблице **Стипендия**. Само выражение имеет следующий вид:

[Стипендия]![Сентябрь] + [Стипендия]![Октябрь][1]

Его можно ввести вручную, но лучше воспользоваться построителем выражений. После вызова построителя нужно выбрать в левом поле папку **Таблицы,** а затем — **Стипендия**. После щелчка по этой папке в среднем поле появится список полей таблицы **Стипендия**. Теперь нужно последовательно вставить в создаваемое выражение следующие элементы: поле **Сентябрь, знак** "+" и поле **Октябрь,** а затем щелкнуть по кнопке **ОК**.

[1] Это выражение не будет давать правильный результат в ситуации, когда студент в одном из месяцев не получал стипендию. Поэтому более корректная форма его записи должна иметь следующий вид:

Nz([Стипендия]![Сентябрь]) + Nz([Стипендия]![Октябрь])

Следует отметить, что функция *Nz*, заменяющая пустые значения нулевыми, отсутствует в списке функций построителя выражений Access 97 и ее нужно вводить вручную.

Глава 5. Создание запросов

5.1. Общая характеристика запросов

При работе с БД часто возникает необходимость выбрать из нее информацию, удовлетворяющую определенным условиям, или произвести различные вычисления. Например, нужно извлечь из учебной БД **Деканат** сведения о студентах, сдавших сессию на «отлично», либо определить, каков средний балл экзамена по английскому языку в данной группе.

Для решения таких задач предназначены *запросы*. Запрос сообщает Access, какая именно информация интересует пользователя в настоящий момент. В нем можно указать, какие

- таблицы содержат нужную информацию;
- записи нужно выбрать из таблиц БД и порядок их сортировки;
- поля должны быть выданы на экран;
- вычисления следует выполнить над выбранными данными.

Запросы могут быть использованы для выделения в таблицах различных групп данных и выполнения над ними вычислений и сравнений. С их помощью можно также выполнить следующие операции:

- добавление новых записей в таблицу;
- удаление записей из таблицы;
- изменение содержимого полей таблицы.

Запрос часто используется в качестве источника данных при создании формы или отчета. Открытие такой формы или отчета автоматически приводит к выполнению соответствующего запроса. Поэтому на экране всегда присутствует самая «свежая» информация об объекте.

Результат запроса можно распечатать или передать с помощью буфера обмена в другое приложение Windows, например, вставить в виде таблицы в документ Word.

Для создания запросов обычно используется метод, суть которого заключается в следующем. Заполняется специальная форма *бланк запроса.* В нее включаются нужные поля и выражения, а также указываются условия отбора записей и порядок их сортировки. Тем самым создается образец желаемого результата запроса.

Access анализирует этот образец и сохраняет в виде инструкций *языка структурированных запросов SQL* (Structured Query Language). Именно в таком виде запрос затем используется в качестве источника данных для форм и отчетов. Отметим, что некоторые важные виды запросов нельзя получить путем заполнения бланка запроса. Для их создания необходимо написать инструкцию на SQL.

5.1.1. Типы запросов в Access

Access позволяет создавать следующие типы запросов:

Запрос-выборка. Используется для отбора информации из таблиц и других запросов БД. При выполнении этого запроса Access создает *динамический набор* записей, содержащий выбранные данные. С этим набором в большинстве случаев можно работать как с обычной таблицей. Его можно просматривать и редактировать, выбирать нужную информацию и т.д. Однако в отличие от таблицы динамический набор записей физически в базе данных не существует и хранится в памяти ЭВМ лишь во время выполнения запроса. При изменении данных в полях динамического набора Access обычно автоматически вносит сделанные изменения в таблицы, на базе которых был построен запрос.

Перекрестный запрос. Представляет данные в формате, подобном электронной таблице, на основе условий, определенных в запросе. С его помощью можно сгруппировать большой объем информации и представить его в виде, удобном для восприятия. Этот вид запроса часто применяется при построении диаграмм.

Запрос на изменение. За одну операцию выбирает записи на основе указанных условий отбора и вносит в них изменения. Существует четыре типа запросов на изменение:

- *запрос на добавление* — добавляет группу записей из одной таблицы в другую таблицу.
- *запрос на удаление* — позволяет удалить группу записей из одной или нескольких таблиц.
- *запрос на обновление* — вносит изменения в группу записей одной или нескольких таблиц.
- *запрос на создание таблицы* — позволяет создать таблицу на основе данных, содержащихся в других таблицах БД.

Запрос SQL — запрос, который может быть создан только с помощью SQL. Существует три типа таких запросов:

- *запрос-объединение* — позволяет объединить поля из нескольких таблиц или запросов в один набор данных;
- *запрос к серверу* — передает инструкции SQL удаленной базе данных;
- *управляющий запрос* — создает, изменяет или удаляет таблицы или индексы базы данных Access.

Запросы являются гибким, интерактивным и итеративным средством. Если запрос сформулирован неточно, его можно легко изменить и выполнить еще раз. Составляя запросы, построенные на результатах предыдущих запросов, вы можете работать с базой данных, задавая ей вопросы типа «А что, если ...?».

5.1.2. Режимы окна запроса

Окно запроса может находиться в одном из трех режимов: конструктора, таблицы и SQL.

Режим конструктора предназначен для создания новых и изменения существующих запросов. Для открытия существующего запроса в режиме конструктора нужно открыть вкладку **Запросы** окна БД, выбрать нужный запрос из списка запросов и щелкнуть по кнопке **Конструктор.** В окне конструктора запросов появится выбранный запрос.

Режим таблицы служит для предварительного просмотра данных, отобранных запросом, или, в случае запроса на изменение, для просмотра данных, которые могут измениться в результате выполнения запроса.

Окно запроса может также находиться в *режиме SQL*, который используют для ввода или просмотра инструкций SQL при создании или изменении запроса. При создании запроса в режиме конструктора Access автоматически создает в режиме SQL эквивалентную инструкцию SQL.

Для переключения между этими режимами следует использовать кнопку **Вид**  **,** расположенную слева на панели инструментов. Нажатие этой кнопки переводит окно в режим, значок которого выведен на ней. Если нажать стрелку рядом с кнопкой, то раскроется список доступных режимов и можно выбрать нужный режим.

5.1.3. Создание запроса

Прежде чем приступить к созданию запроса, следует продумать ответы на следующие вопросы:

- какая таблица или таблицы содержит нужную информацию,
- как связать эти таблицы,
- какой тип запроса использовать,
- каким условиям должны удовлетворять отбираемые записи,
- каким должен быть порядок их сортировки,
- какие вычисления нужно выполнить над отобранными данными,
- какое имя должен получить создаваемый запрос.

Для создания запроса нужно щелкнуть по корешку **Запросы** окна БД, а затем по кнопке **Создать** над списком запросов. На экране появится диалоговое окно **Новый запрос,** и Access предоставит вам выбор: создать запрос самому в режиме конструктора или воспользоваться помощью одного из мастеров по разработке запросов.

С помощью мастеров можно создать запросы следующих типов:

- *Простой запрос.* Создается запрос на выборку из указанных полей.

- *Перекрестный запрос.* Выводит данные в формате электронной таблицы.
- *Поиск повторяющихся записей.* Осуществляется поиск повторяющихся записей в указанной таблице или запросе.
- *Поиск записей, не имеющих подчиненных.* Находит все записи в главной таблице, не имеющие связанных с ними записей в подчиненной таблице.

Использование мастера Простой запрос

Для быстрого создания запроса можно воспользоваться мастером **Простой запрос**. Этот мастер позволяет не только отобрать нужные поля из таблиц или ранее созданных запросов, но также суммировать, вычислять средние значения и находить другие статистические характеристики определенных групп записей.

После выбора этого варианта появится диалоговое окно **Создание простых запросов**. В раскрывающемся списке **Таблицы/запросы** нужно выбрать таблицу или запрос и с помощью двойного щелчка отобрать поля, которые будут содержаться в создаваемом запросе. Если в запросе используются поля из нескольких таблиц или запросов, то эту операцию отбора полей следует повторить нужное число раз. Отметим, что мастер не разрешит использовать таблицы, между которыми не были установлены связи.

Затем нужно выбрать тип создаваемого запроса. Если нужно, чтобы отображались все записи, щелкните на переключателе **Подробный**. Для вывода только общей информации о записях щелкните на переключателе **Итоговый** и затем по кнопке **Итоги**. После этого укажите, какие итоговые значения необходимо вычислять, и щелкните по кнопке **ОК.**

В последнем диалоговом окне нужно ввести имя запроса и выбрать дальнейшие действия: просмотр результатов выполнения запроса в режиме таблицы или изменение его структуры в режиме конструктора. Затем следует щелкнуть по кнопке **Готово**.

Использование режима Конструктор

Больше возможностей для создания запроса предоставляет конструктор запросов. Чтобы создать запрос с его помощью, нужно выбрать пункт **Конструктор** в диалоговом окне **Новый запрос**. На экране появится окно **Запрос на выборку**, а поверх него окно **Добавление таблицы** со списком таблиц, хранящихся в текущей БД. По мере того как вы будете выбирать нужные таблицы, Access помещает списки их полей в верхней части окна запроса. Для выбора таблицы достаточно дважды

щелкнуть мышью по ее имени или щелкнуть сначала по ее имени, а затем по кнопке **Добавить**. Если в создаваемом запросе вы хотите использовать поля из ранее созданного запроса, то нужно щелкнуть по ярлыку **Запросы** или **Запросы и таблицы**. В этом случае список таблиц будет заменен или, соответственно, расширен списком уже существующих запросов, из которого можно выбрать нужный запрос.

После завершения отбора нужных таблиц и запросов следует нажать кнопку **Закрыть.** Чтобы снова открыть окно **Добавление таблицы**, нужно щелкнуть по кнопке **Добавить таблицу**  , расположенной на панели инструментов, или выбрать команду **Добавить таблицу** в меню **Запрос.**

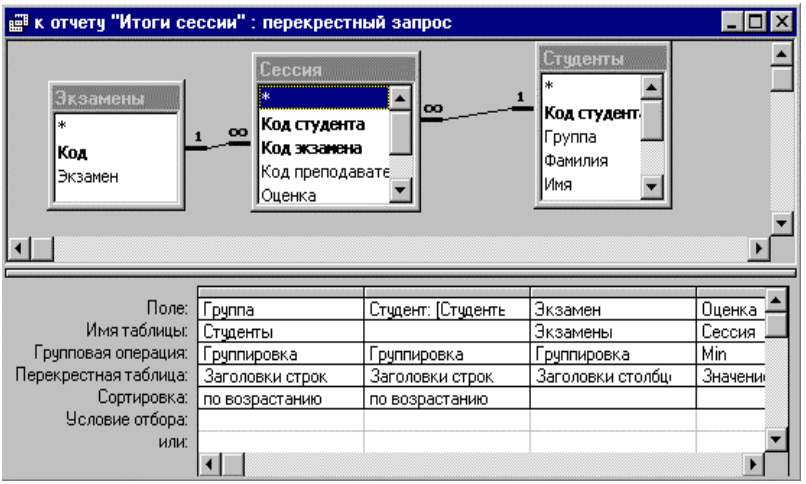


Рис. 5.1. Окно запроса в режиме конструктора

Если вы включили в запрос несколько таблиц, и между ними были установлены связи, то Access автоматически показывает их, рисуя *линии объединения* между связанными полями. Линию объединения можно создать самостоятельно, переместив ключевое поле главной таблицы на связываемое с ним поле подчиненной таблицы.

5.2. Работа в окне конструктора запросов

Окно конструктора запросов разделено на две части (см. рис. 5.1). Верхняя часть содержит списки полей таблиц или запросов, включенных в данный запрос. В нижней части находится бланк запроса, заполняя который вы указываете параметры создаваемого запроса. Каждый столбец бланка запроса соответствует одному из полей запроса, которое может быть полем из списка, а также вычисляемым или итоговым полем.

Для каждого поля можно указать в соответствующих строках бланка порядок сортировки записей по этому полю; будет ли оно включено в динамический набор — результат запроса, а также можно определить условия отбора его значений. Внизу окна находится полоса прокрутки полей запроса.

Бланк запроса содержит еще две строки, которые изначально обычно не видны на экране. Это строки **Имя таблицы** и **Групповая операция**. Чтобы отобразить их на экране, нужно выбрать в меню **Вид** соответствующие команды.

5.2.1. Добавление или удаление таблицы/запроса

Чтобы добавить таблицу или запрос в окно конструктора, достаточно щелкнуть правой кнопкой мыши в пустом месте верхней части окна запроса и выбрать в контекстном меню пункт **Добавить таблицу.** Затем выбрать в появившемся окне **Добавление таблицы** нужную таблицу или запрос.

В окне конструктора запросов не должны находиться посторонние (не используемые в создаваемом запросе) таблицы. Даже если такая таблица не связана с другими таблицами, само ее наличие в окне конструктора приведет к созданию неверного запроса.

Для удаления таблицы или запроса из окна конструктора нужно выделить удаляемую таблицу или запрос, щелкнув по списку полей и нажать клавишу **Delete**. Поля, добавленные в бланк путем переноса с помощью мыши из списка полей, также будут удалены из запроса.

5.2.2. Включение полей в бланк запроса

В бланк запроса следует включать только те поля, данные из которых выводятся на просмотр, обновляются или используются для задания условий отбора, группировки и сортировки.

Чтобы включить поле из списка полей таблицы или запроса в бланк запроса, достаточно дважды щелкнуть мышью по этому полю. Можно также перетащить его в нужное место бланка.

Для добавления сразу нескольких полей нужно выделить в списке эти поля, держа нажатой клавишу **Ctrl**, и затем перенести всю выделенную группу с помощью мыши в бланк запроса. Если добавляемые поля в списке идут подряд, то для их выделения можно, держа нажатой клавишу **Shift**, щелкнуть мышью по первому полю, а затем по последнему полю.

Если в запрос нужно включить все поля из списка, то в этом случае можно просто перенести в бланк запроса символ звездочки (*), расположенный над списком полей. Преимущество этого способа состоит в том, что при использовании * изменения в структуре полей базовой

таблицы автоматически отражаются в запросе, созданном таким способом. В частности, все поля, удаленные из базовой таблицы, будут автоматически удалены из запроса.

Для полей, добавленных в запрос с помощью *, нельзя задавать условия отбора или сортировки. В этом случае следует явно добавить требуемые поля в запрос, отменив для них вывод на экран.

5.2.3. Добавление столбца в бланк запроса

Чтобы вставить пустой столбец в бланк запроса, нужно выбрать любое место в столбце, слева от которого требуется добавить новый столбец, и выбрать пункт С**толбцы** в меню Вставк**а.**

5.2.4. Удаление столбца из бланка запроса

Нужно выделить удаляемый столбец. Для этого следует установить курсор мыши в область выделения (над заголовком поля), добиться, чтобы он принял форму жирной направленной вниз стрелки, и щелкнуть мышью. Удаляемый столбец окрасится в черный цвет. Затем нужно нажать клавишу **Delete**. Для удаления нескольких идущих подряд столбцов достаточно распространить на них выделение перемещением мыши с нажатой левой кнопкой и затем нажать клавишу **Delete**. Можно также использовать команду **Удалить столбцы** в меню **Правка.**

5.2.5. Перемещение поля в бланке запроса

Включите режим **Сдвиг** одновременным нажатием клавиши **Ctrl** и **F8**. Затем выделите поле, которое требуется переместить. Для перемещения выделенного поля используются клавиши ← и →.Чтобы выключить режим **Сдвиг**, нужно нажать на клавишу **Esc**. Аналогично можно переместить группу идущих подряд полей.

5.2.6. Добавление вычисляемого поля

В бланк запроса можно добавить вычисляемые поля. *Вычисляемые поля* — это временные поля, создаваемые в динамическом наборе записей при выполнении запроса. В них заносятся результаты вычислений над данными из полей таблиц, включенных в запрос. Правила вычисления значения поля задаются с помощью выражения.

Для создания вычисляемого поля нужно щелкнуть на пустой ячейке строки **Поле.** Затем следует ввести в эту ячейку имя создаваемого поля, двоеточие и выражение, вычисляемое в этом поле. Если элементом выражения является поле, его имя нужно заключить в квадратные скобки. Арифметические операторы записываются обычным образом, а для объединения текстовых значений нужно использовать оператор &.

После ввода выражения следует щелкнуть за пределами ячейки. Если Access обнаружит ошибку, то будет выведено соответствующее сообщение.

Для модификации выражения нужно щелкнуть по содержащей его ячейке и внести изменения. Чтобы просмотреть введенное выражение целиком, нужно нажать клавиши **Shift+F2** или щелкнуть правой кнопкой мыши по содержащей его ячейке и выбрать в контекстном меню пункт **Масштаб**. Содержимое ячейки появится в отдельном окне **Область ввода**.

Примеры вычисляемых полей:

ФИО: [Фамилия] & " " & [Имя] & " " & [Отчество];

*Стоимость заказа: [Цена] * [Количество];*

Год рождения: Year([Дата рождения]);

*Цена со скидкой: [Цена]*0,9.*

Если при создании вычисляемого поля не указать его имя, а ограничиться только вводом выражения, то Access сам даст имя новому полю. Это имя можно легко изменить (см. п. 5.2.7).

При создании сложных выражений следует использовать построитель выражений. Для вызова построителя нужно щелкнуть правой кнопкой мыши на ячейке Поле, куда необходимо поместить вычисляемое поле и выбрать пункт **Построить**, или щелкнуть по ячейке Поле, а затем по кнопке **Построить**  на панели инструментов. Работа в окне построителя выражений описана в главе 4.

5.2.7. Изменение имени поля в запросе

Поле запроса можно переименовать, чтобы точнее описать его содержимое. Обычно эта операция производится при определении нового вычисляемого поля, а также при расчетах различных итоговых значений по данным существующего поля. В таких случаях Access по умолчанию использует имена типа «*ВыражениеN*» или «*Sum_Количество*», которые можно легко заменить более содержательными именами.

Чтобы переименовать поле, включенное в запрос, нужно установить указатель слева от первой буквы имени этого поля и ввести новое имя, а затем двоеточие. Например, для замены старого имени поля **Дата рождения** на новое имя **Год рождения** нужно ввести в строку **Поле** соответствующего столбца бланка запроса выражение *Год рождения: Дата рождения.*

Для изменения имени вычисляемого или итогового поля нужно просто заменить расположенное левее двоеточия старое имя на новое имя.

Другим способом изменения имени поля является указание нового имени в свойстве **Подпись** данного поля.

При изменении имени поля Access покажет новое имя как заголовок столбца запроса в режиме таблицы. Новое имя будет также использовано в любой форме (отчете), созданной на основе этого запроса.

5.2.8. Отображение/скрытие поля в результатах запроса

В бланк запроса можно добавить поле, используемое для задания условия отбора или порядка сортировки, без отображения данных из этого поля в результатах запроса. Для этого нужно установить флажки в строке *«Вывод на экран»* бланка для тех полей, которые требуется отображать, и снять флажки для неотображаемых полей. Последние поля не будут включены в результирующий динамический набор.

5.2.9. Изменение ширины столбца в бланке запроса

Перемещайте указатель к границе области выделения столбца, ширину которого нужно изменить, до тех пор пока указатель не примет вид двусторонней стрелки. Затем переместите границу столбца в нужном направлении.

Чтобы быстро установить оптимальную ширину столбца (по размеру данных), достаточно установить указатель на правую границу изменяемого столбца и дважды щелкнуть мышью. Если было выделено несколько столбцов, то после выполнения этой операции для каждого из них будет установлена оптимальная ширина.

5.2.10. Сортировка результатов запроса

Обычно результаты запроса появляются на экране в том же порядке, в котором записи находились в базе данных. Для удобства просмотра их можно отсортировать по алфавиту или в числовом порядке в зависимости от типа поля сортировки.

Чтобы выбрать порядок сортировки, установите указатель мыши в ячейку **Сортировка** под полем, значения в котором нужно отсортировать, щелкните мышью и раскройте появившийся список. Вы можете выбрать из этого списка тип сортировки *«По возрастанию»* (0 – 9, А – Я) или *«По убыванию»* (9 – 0, Я – А).

Сортировку можно задать сразу для нескольких полей. В этом случае порядок сортировки определяется их положением в бланке запроса. Сначала сортируются данные в крайнем левом поле и далее слева направо.

5.2.11. Вставка или удаление строки условий отбора

Для того чтобы вставить строку условий отбора в бланк запроса, нужно щелкнуть мышью по строке, над которой должна появиться новая строка, а затем выбрать в меню **Вставка** пункт С**троки**.

Для удаления строки условий отбора нужно щелкнуть мышью по удаляемой строке, а затем выбрать в меню **Правка** пункт **Удалить строки.**

5.2.12. Установка свойств запроса

Каждый запрос имеет свойства, которые определяют его поведение и вид полученного результата. Чтобы задать свойства запроса, нужно вызвать *бланк свойств*. Для этого следует щелкнуть правой кнопкой мыши, установив ее указатель в окне запроса вне бланка запроса и списков полей, и выбрать в контекстном меню пункт **Свойства**. На экране появится бланк свойств запроса. Ниже описаны некоторые из них.

а) Вывод всех полей

Указывает, какие поля должны быть включены в результат запроса: все поля из базовых таблиц или только те, для которых установлен флажок **Вывод на экран** в бланке запроса. Если вы создаете запрос для формы и хотите, чтобы все поля из всех таблиц были в ней доступны, то установите для свойства **Вывод всех полей** значение «**Да**».

б) Набор значений

Указывает, сколько записей должно быть возвращено запросом. Значением свойства **Набор значений** является целое число или число процентов. Например, для вывода 10 первых записей в качестве значения свойства **Набор значений** следует ввести число 10, а для вывода 10 процентов от общего числа записей — 10%.

Обычно это свойство используется, если нужен не весь результат запроса, а лишь записи, имеющие самые большие или самые маленькие значения в данном поле. В этом случае в ячейке **Сортировка**, соответствующей этому полю, следует выбрать «*по убыванию*», если нужно отобразить наибольшие значения, и «*по возрастанию*» — при выводе на экран наименьших значений.

в) Уникальные значения

Указывает, следует ли возвращать повторяющиеся значения полей. Если значение этого свойства «**Да**», то Access возвращает только те записи, у которых значения всех полей, изображаемых в режиме таблицы, являются уникальными. Если значение «**Нет**», то возвращаются все записи (используется по умолчанию).

5.2.13. Установка свойств полей запроса

Кроме свойств запроса можно задать свойства любого из его полей, кроме * и полей, для которых не установлен **флажок Вывод на экран.** Для вызова бланка свойств поля следует щелкнуть по нему правой кнопкой мыши, а затем выбрать пункт **Свойства** в контекстном меню.

Чаще всего бланк свойств поля запроса используется для изменения формата изображения данных в запросе. По умолчанию поле в запросе наследует формат соответствующего ему поля базовой таблицы. Чтобы

изменить этот формат, нужно щелкнуть по ячейке **Формат поля**, открыть список стандартных форматов и выбрать нужный формат или задать пользовательский формат.

В качестве формата для текстового поля можно ввести символ > или <, чтобы его содержимое выводилось прописными или строчными буквами. Для вывода дополнительных символов в ячейку **Формат поля** следует ввести @ и нужные символы.

С помощью задания соответствующего формата можно обеспечить вывод сообщения в том случае, когда в поле отсутствует значение. Например, чтобы при выводе сведений о преподавателях в случае пустого (*Null*) значения в поле **Домашний телефон** появлялось слово *неизвестно*, а в случае пустой строки ("") слово *нет*, следует задать формат: *@; "неизвестно"; "нет"*.

Для поля типа *Дата/время* можно, в частности, использовать следующие форматы вывода его значений:

ddd	—	сокращенное название дня недели;
dddd	—	полное название дня недели;
mmm	—	сокращенное название месяца;
mmmm	—	полное название месяца;
q	—	номер квартала в году;
yy	—	последние две цифры номера года;
yyyy	—	полный номер года.

Свойство **Подпись позволяет** определить новое имя поля. Оно заменит при просмотре результатов запроса старое имя. Новое имя будет также использовано при создании формы (отчета), основанной на этом запросе.

Пример 5.1. Для того чтобы при отборе записей из таблицы **Студенты** появлялась информация не о дате, а о годе рождения, достаточно внести следующие изменения в свойства поля **День рождения**: в ячейку **Формат поля** ввести формат даты «*yyyy*», а в ячейку **Подпись** — текст «*Год рождения*».

5.2.14. Предварительный просмотр результатов запроса

Чтобы просмотреть результаты запроса, находясь в режиме конструктора, нужно выбрать в меню **Вид** пункт **Режим таблицы** или нажать кнопку **Вид**  на панели инструментов. На экране появится созданный динамический набор данных.

Чтобы снова вернуться в режим конструктора, нужно нажать кнопку **Вид** на панели инструментов или выбрать в меню **Вид** пункт **Конструктор.**

5.2.15. Выполнение запроса

Чтобы выполнить ранее созданный запрос, нужно найти его в списке запросов, находящихся во вкладке **Запросы** окна БД, и сделать двойной щелчок мышью по его значку или имени.

Для выполнения запроса, находящегося в окне конструктора, нужно нажать кнопку Запуск  на панели инструментов. Чтобы остановить выполнение запроса, следует нажать клавиши **Ctrl+Break**.

5.2.16. Сохранение запроса

Для сохранения запроса нужно выбрать команду **Сохранить** в меню **Файл** или нажать кнопку **Сохранить**  на панели инструментов**.** Если сохраняется новый запрос, Access автоматически предлагает для него стандартное имя. Лучше заменить его более содержательным именем и нажать кнопку **ОК**. Сохраненный запрос становится частью БД. Его имя не должно совпадать с именем какой-либо таблицы БД.

5.3. Отбор записей в однотабличных запросах

Обычно в результате выполнения запроса нужно найти только те записи, которые удовлетворяют некоторым условиям отбора. Ввод условия отбора в запросе похож на ввод условия на значение для поля (см. соответствующий пункт главы 2). Сначала рассмотрим случай, когда в создании запроса участвует одна таблица.

5.3.1. Точное совпадение значений полей

Простейший критерий отбора предполагает извлечение из исходной таблицы всех записей, имеющих одинаковые значения в одном или нескольких полях. В этом случае для создания соответствующего условия отбора нужно включить в бланк запроса поля, значения в которых должны совпадать, и напечатать в них значения-шаблоны в строке **Условие отбора**. Тем самым будет создан образец, с которым Access при выполнении запроса будет сравнивать все записи исходной таблицы. В результирующий динамический набор данных попадут лишь те записи, значения соответствующих полей которых совпали со значениями полей этого образца.

При вводе текстовых значений регистр символов несущественен. После завершения ввода текста Access автоматически заключает его в кавычки.

Пример 5.2. Требуется извлечь из таблицы **Студенты** сведения о девушках группы 9701. Для этого достаточно перенести в первое поле бланка запроса звездочку *, задающую вывод всех полей. В поле **Группа**

бланка запроса нужно ввести в строке **Условия отбора** значение-шаблон «9701», а в поле **Пол** — значение-шаблон "ж" и отменить вывод на экран содержимого этих полей. Созданный бланк запроса будет иметь следующий вид (см. рис. 5.2).

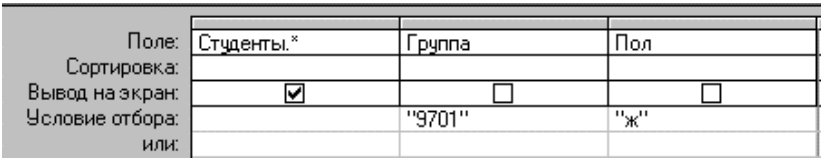


Рис. 5.2. Пример бланка запроса с использованием шаблонов

5.3.2. Шаблоны общего вида. Оператор Like

Часто условием отбора записей является не полное, а частичное совпадение значений в указанных полях исходной таблицы, например, если значения в текстовом поле содержат заданные символы. В этом случае при записи в строке условий соответствующего шаблона используются уже известные спецсимволы ?, # и *.

Условие отбора имеет следующий вид: *Like <шаблон>*. Как правило, вводить оператор *Like* не нужно. В большинстве случаев Access самостоятельно добавит его после завершения ввода строки, содержащей любой из вышеприведенных символов шаблона.

Таблица 5.5. Примеры условий с шаблонами

Условие	*Комментарий*
Like "п*в"	любой текст, начинающийся с буквы «п» и заканчивающийся буквой «в»;
Like "*запрос*"	любой текст, содержащий слово «запрос»;
Like "1?.05.72"	любая дата между 10 и 19 мая 72 года;
Like "*.01.98"	любая дата в январе 98 года;
Like "*.03.*"	любая дата в марте;
Like "*.*.98"	любая дата в 98 году;
Like "*.0[4-6].00"	любая дата во втором квартале 2000 года.

Например, чтобы получить информацию о студентах, фамилия которых начинается с буквы М, нужно ввести в поле **Фамилия** два символа: "м" и "*". После нажатия клавиши **Enter** Access заключит введенные символы в кавычки и добавит слева оператор *Like*. В итоге условие отбора нужных записей примет следующий вид: *Like "м*"*.

Оператор *Like* обычно используется при задании шаблонов для текстовых или Memo полей. Однако с его помощью можно создавать шаблоны для отбора записей по содержимому поля даты (см. табл. 5.5).

5.3.3. Диапазон значений. Операторы And и Between

Иногда возникает необходимость отобрать записи, у которых значения в указанном поле лежат в некотором диапазоне значений. Диапазон значений можно определить, если перед значением указать один из операторов сравнения (см. гл. 4), например, *> 100* или *<= 10*.

Операторы сравнения могут использоваться в текстовых и числовых полях, а также в полях дат. Если диапазон имеет две границы, то оба граничных условия должны быть помещены на одной строке и разделены логическим оператором *And*.

Пример 5.3. Допустим, что нужно получить информацию о студентах, родившихся в первой половине августа 1972 года. В этом случае в поле **Дата рождения** бланка запроса нужно ввести условие *>=#01.08.72# And <=#15.08.72#*.

Другим способом задания диапазона значений является использование оператора *Between*. Например, с его помощью приведенное выше условие отбора можно записать так: *Between #01.08.72# And #15.08.72#*.

5.3.4. Список значений. Операторы Or и In

Для проверки содержится ли значение в заданном списке, нужно использовать логический оператор *Or* или оператор *In*. Если список невелик (не более 9 элементов), то требуемый результат можно получить, введя каждое значение из списка в отдельной ячейке строки условий **или**.

Пример 5.4. Для отбора записей таблицы **Студенты**, содержащих сведения о студентах групп 9701, 9702 и 9705, нужно в поле **Группа** бланка запроса ввести условие *"9701" Or "9702" Or "9705"*. Это условие можно заменить условием *In ("9701";"9702";"9705")* либо ввести каждое из названий групп в отдельной строке условий **или**.

5.3.5. Отрицание образца. Оператор NOT

Чтобы найти записи, которые не удовлетворяют определенному условию, перед условием выбора нужно поставить логический оператор *Not*. Этот оператор можно использовать вместе со всеми другими рассмотренными выше условиями выбора.

Пример 5.5. Чтобы получить информацию обо всех студентах кроме тех, кто учится в группе 9701, в поле **Группа** бланка запроса к таблице **Студенты** надо ввести текст *not 9701*.

5.3.6. Отбор записей с Null значениями и пустыми строками

Иногда могут понадобиться записи, в полях которых отсутствуют значения. Вы можете найти записи в таблице, не содержащие значения в определенном поле, если наберете слово *null* в этом поле в бланке запроса. Access создаст условие отбора вида *Is Null,* и в таблицу попадут только те записи, которые не содержат значения (имеют пустое значение) в этом поле. Соответственно, вы можете найти только те записи, которые содержат значения в определенных полях, если наберете *not null* (*Is Not Null*) в этих полях в бланке запроса.

Пример 5.6. Чтобы определить, какие студенты получили (не получили) стипендию в сентябре, нужно создать запрос к таблице **Стипендия** и ввести в поле **Сентябрь** бланка запроса в качестве условия *Is Not Null (Is Null).*

В том случае, когда поле таблицы может содержать пустые строки (""), для выборки записей, содержащих в этом поле пустую строку, нужно ввести в качестве условия отбора две идущих подряд кавычки "".

Пример 5.7. Описанные ниже варианты условий относятся к запросам, извлекающим информацию из таблицы **Преподаватели**.

1. Для получения списка преподавателей, не имеющих домашнего телефона (см. пример 2.1 на стр. 21), нужно ввести в поле **Домашний Телефон** в качестве условия отбора пустую строку ("").

2. Для получения списка преподавателей, о которых неизвестно, имеют ли они домашний телефон, нужно ввести в поле **Домашний Телефон** в качестве условия отбора слово *null*.

3. Для получения списка преподавателей, имеющих домашний телефон, нужно ввести в поле **Домашний Телефон** в качестве условия отбора текст *Is Not Null And <>""*.

5.3.7. Использование нескольких строк условий

При создании сложных запросов может возникнуть ситуация, когда описание критерия отбора занимает не одну, а несколько строк в бланке запроса. В этом случае Access последовательно анализирует каждую из строк и включает в итоговый набор те записи, для которых эти условия выполняются. Если строка содержит несколько ячеек с условиями отбора, то в набор попадают лишь те записи, которые удовлетворяют одновременно всем условиям в этой строке. В результате в итоговый набор будут включены все записи, удовлетворяющие хотя бы одной из строк условий.

Пример 5.8. Предположим, что нам нужно получить списки студентов групп 9701 и 9703, родившихся в 1972 году. Это можно сделать, создав запрос следующего вида:

Первая строка условия отбора включает в результат запроса список студентов 9701 группы, родившихся в 1972 году, а вторая строка — список студентов 9703 группы того же года рождения. Задание сортировки позволяет получить итоговый список, отсортированный сначала по группам, а внутри каждой группы — по фамилиям студентов. Отметим, что условие *Like "*.*.72"* необходимо указать в обеих строках, так как, если его не повторить во второй строке, то в итоговый список попадут все студенты 9703 группы.

Поле:	Группа	Фамилия	Имя	Дата рождения
Имя таблицы:	Студенты	Студенты	Студенты	Студенты
Сортировка:	по возрастанию	по возрастанию		
Вывод на экран:	☑	☑	☑	☑
Условие отбора:	"9701"			Like "*.*72"
или:	"9703"			Like "*.*72"

Рис. 5.3. Запрос, использующий две строки условий

В данном примере получить такой же результат можно, используя лишь одну строку условия отбора и один из операторов Or или In. Например, в поле **Группа** можно ввести условие *"9701" Or "9703"*, а в поле **Дата рождения** — условие *Like "*.*.72"*. Однако если поставлена задача получить списки студентов этих групп, родившихся в разные годы, то соответствующий запрос к таблице **Студенты** обязательно будет содержать более одной строки условий.

5.3.8. Использование вычисляемых полей

В ряде случаев для отбора записей необходимо включить в запрос вычисляемое поле и ввести условие, использующее значения этого поля.

Пример 5.9. Таблица **Заказы** содержит сведения о заказанных товарах. В частности, поле **Количество** содержит информацию о числе заказанных товаров, а поле **Цена** — цену товара. Чтобы выбрать из таблицы сведения о заказах, стоимость которых не ниже 500 руб., следует включить в запрос нужные поля, а также создать вычисляемое поле *Стоимость: [Количество]*[Цена]* и ввести в него условие *>=500*.

5.3.9. Использование ссылки на имя поля

Иногда необходимо использовать значение одного поля таблицы в качестве условия отбора для другого поля. При создании соответствующего выражения нужно заключить имя поля в квадратные скобки.

Пример 5.10. Таблица **Заказы** содержит поля **Дата получения** и **Дата отправки**. Чтобы найти все заказы, в которых от получения до отправки заказа прошло не более трех дней, следует в поле **Дата отправки** ввести условие <= *[**Дата получения**]* + *3*.

5.3.10. Использование ссылки на элемент управления

В условии отбора можно сослаться на текущее значение элемента управления открытой формы.

Пример 5.11. Пусть форма **Список жильцов** содержит поле с именем **Номер комнаты**, значениями которого являются номера комнат в студенческом общежитии. Чтобы создать запрос, выбирающий из таблицы **Общежитие** коды студентов, живущих в данной комнате, можно поступить так: поместить в бланк запроса поля: **Код студента** и **Комната,** затем ввести в поле **Комната** условие

Forms![Список жильцов]![Номер комнаты].

Если открыть форму **Список жильцов** и выбрать номер какой-либо комнаты, а затем запустить на выполнение созданный запрос, то на экране появятся коды студентов, живущих в этой комнате.

Если информация для запроса содержится в поле со списком **Номер комнаты,** то условие отбора запроса должно иметь следующий вид:

Forms![Список жильцов]![Номер комнаты].[Text].

5.3.11. Использование параметров в условиях отбора

В условиях отбора можно использовать не только конкретные значения, но и параметры. Перед выполнением запроса, содержащего один или несколько параметров, Access каждый раз будет запрашивать конкретные условия отбора и затем произведет отбор нужной информации. Используя параметры, можно заменить несколько однотипных запросов к данным одним запросом.

Чтобы задать параметр, нужно ввести в строку **Условие отбора** вместо конкретного значения произвольный текст, заключенный в квадратные скобки [ ]. Этот текст Access рассматривает как имя параметра и выводит его в специальном диалоговом окне при выполнении запроса. Поэтому в качестве имени параметра лучше использовать содержательную фразу, причем имя каждого параметра должно быть уникальным.

Для каждого параметра запроса можно указать тип данных. Эту информацию Access использует для проверки введенного значения. Например, если параметр определен как числовой, то Access не разрешит

ввести значение, содержащее буквы. По умолчанию все параметры запроса имеют текстовый тип.

Для задания типа данных параметра нужно выбрать пункт меню **Запрос**, а затем **Параметры** или вызвать щелчком правой кнопки мыши контекстное меню запроса и выбрать в нем пункт **Параметры. Access выведет на экран** диалоговое окно **Параметры запроса**. В этом окне нужно ввести в столбце **Параметр** имя того параметра, тип которого следует изменить, точно в том же виде, в каком оно было введено в бланк запроса, но без квадратных скобок. В столбце **Тип данных** следует выбрать из раскрывающегося списка нужный тип данных параметра. После того как определены типы всех параметров, нужно нажать кнопку **ОК.**

При выполнении запроса Access попросит ввести одно за другим значения для каждого из параметров, используя для ввода диалоговые окна **Введите значение параметра**. После ввода всех значений запрос будет выполнен и на экране появится результирующий набор записей.

Поле:	Группа	Фамилия	Имя
Сортировка:			
Вывод на экран:	☑	☑	☑
Условие отбора:	[Введите номер группы]		
или:			

Рис. 5.4. Пример запроса с параметром

Пример 5.12. Чтобы получить список студентов любой учебной группы, используем запрос с параметром. Для его создания нужно включить в бланк запроса к таблице **Студенты** поля **Группа**, **Фамилия** и **Имя**. В строке **Условие отбора** в поле **Группа** следует ввести текст *Введите номер группы* и заключить его в квадратные скобки **(см. рис**. 5.4).

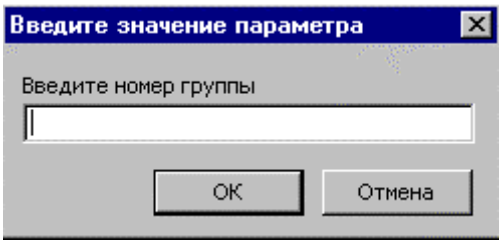


Рис. 5.5. Окно для ввода параметра

При выполнении запроса появится диалоговое окно с приглашением ввести номер группы **(см. рис**. 5.5). После ввода номера и нажатия кнопки **ОК** Access выполнит запрос и создаст требуемый список группы.

Параметры могут использоваться в условиях отбора совместно с шаблонами. Например, для нахождения всех студентов,

фамилия которых начинается с данной буквы, нужно включить в бланк запроса к таблице **Студенты** в поле **Фамилия** условие *Like [введите букву] & "*"*.

5.3.12. Использование в условиях отбора подчиненного запроса

Иногда условие отбора использует значение, которое можно получить в результате выполнения другого запроса, называемого *подчиненным*. В этом случае можно поступить следующим образом:

1. Создать подчиненный запрос.
2. Перейти в режим SQL и скопировать инструкцию SQL, описывающую подчиненный запрос в буфер обмена.
3. Создать основной запрос и перейти в нужную ячейку строки **Условие отбора**.
4. Ввести требуемый условный оператор (например, > или <), а затем вставить содержимое буфера обмена, заключив его в круглые скобки.

Отметим, что подчиненный запрос должен иметь в качестве результата значение, т.е. одну запись, содержащую одно поле. Подчиненный запрос может использоваться в выражении, определяющем вычисляемое поле.

Пример 5.13. Пусть нужно найти студентов, которые не старше студента 9702 группы Иванова, причем известно, что у него в группе нет однофамильцев. В этом случае можно создать подчиненный запрос (см. рис. 5.6) к таблице **Студенты**, результат которого — дата рождения Иванова (27 марта 72 года)[1].

Поле:	Группа	Фамилия	Дата рождения
Сортировка:			
Вывод на экран:	☐	☐	☑
Условие отбора:	"9702"	"иванов"	

Рис. 5.6. Подчиненный запрос, дающий дату рождения Иванова

Соответствующая инструкция SQL SELECT выглядит так:

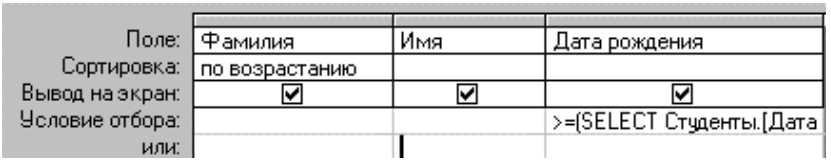
SELECT Студенты.[Дата рождения] FROM Студенты WHERE (((Студенты.Группа)=«9702») AND ((Студенты.Фамилия) =«иванов»));

Она содержит следующую информацию:

- список выбираемых полей (находится после ключевого слова SELECT);

[1] Для нахождения даты рождения Иванова можно также использовать функцию *DLookUp*.

- таблицу, используемую в запросе (указывается в предложении FROM);
- условие отбора записей (указывается в предложении WHERE).



Поле:	Фамилия	Имя	Дата рождения
Сортировка:	по возрастанию		
Вывод на экран:	☑	☑	☑
Условие отбора:			>=(SELECT Студенты.[Дата
или:			

Рис. 5.7.Основной запрос, использующий инструкцию SQL

Эту инструкцию нужно скопировать в буфер обмена и создать еще один запрос к таблице **Студенты,** содержащий поля **Фамилия**, **Имя** и **Дата рождения.** В поле **Дата рождения** следует вставить из буфера обмена сохраненную инструкцию SELECT, заключив ее в круглые скобки и поставив в начале условия знак >=.

Условие отбора в поле **Дата рождения** будет выглядеть так:

>= (SELECT Студенты.[Дата рождения] FROM Студенты WHERE (((Студенты.Группа)="9702") AND ((Студенты.Фамилия) ="иванов"));

Список студентов, младше Иванова : запрос на выборку

Фамилия	Имя	Дата рождения
Амосов	Дмитрий	26 сентября 1974 г.
Болотов	Константин	24 июля 1974 г.
Бродников	Александр	24 июля 1974 г.
Брянская	Надежда	11 мая 1974 г.
Бутаков	Андрей	9 августа 1974 г.
Вохмякова	Любовь	19 сентября 1974 г.

Запись: 1 из 19

Рис. 5.8. Итоговый список студентов

В результате выполнения этого запроса будет получен требуемый список студентов (см. рис. 5.8). Отметим, что этот запрос даст правильный результат лишь в том случае, если у Иванова нет однофамильцев. Если это условие не выполнено, то необходимо использовать для отбора даты рождения не фамилию, а код студента.

5.4. Многотабличные запросы

Часто возникает необходимость в создании запроса, отбирающего информацию из нескольких таблиц или ранее сохраненных запросов. В этом случае списки полей всех используемых таблиц/запросов должны быть включены в окно конструктора создаваемого запроса. Процедура добавления таблиц/запросов в запрос, создаваемый в режиме конструктора, была описана выше (см. стр. 72).

Далее рассматриваются различные типы установления связи (объединения) таблиц/запросов, находящихся в окне конструктора, и соответствующие им принципы отбора записей в запросе. Задание связей между таблицами — важный этап конструирования запроса и сделанная на этом этапе ошибка, как правило, приводит к тому, что запрос дает неверный результат.

Отметим, что обычно Access устанавливает связи добавленной таблицы с другими таблицами автоматически. Это происходит в следующих случаях:

- таблицы были связаны между собой в окне **Схема данных**; информация о связи между ними берется из схемы данных;
- связь (внутреннее объединение) устанавливается между таблицами, имеющими поля с одинаковыми именами, если одно из них ключевое.

Однако нередко возникает ситуация, когда приходится либо задавать связь самостоятельно, либо переопределять тип связи, которую установил Access.

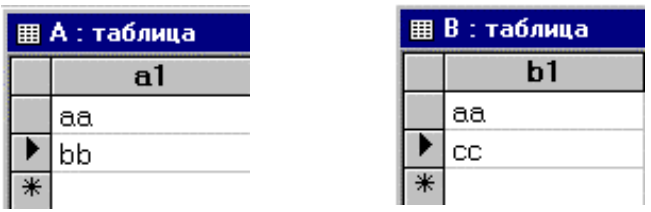


Рис. 5.9. Содержимое таблиц A и B

Рассмотрим основные принципы отбора записей, соответствующие различным типам объединения таблиц, на примере двух таблиц A и B. Каждая из них содержит по одному текстовому полю и две записи (см. рис. 5.9).

5.4.1. Случай несвязанных таблиц

Начнем с рассмотрения ситуации, когда таблицы A и B включены в запрос, но между ними не установлена связь (см. рис. 5.10). В этом случае Access формирует динамический набор данных, записи которого являются декартовым произведением[1] записей исходных таблиц. Его можно вывести

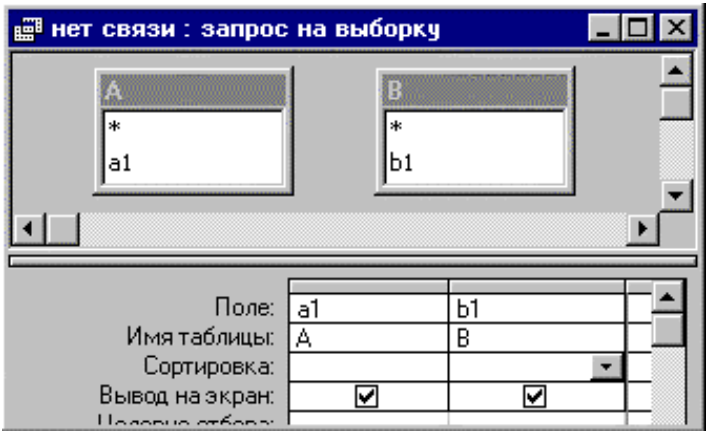


Рис. 5.10. Запрос с несвязанными таблицами

на экран, включив поля обеих таблиц в бланк запроса (см. рис. 5.11). Инструкция SQL, соответствующая этому запросу, имеет вид

SELECT A.a1, B.b1 FROM A, B;

нет связи : запрос на выборку

a1	b1
aa	aa
bb	aa
aa	cc
bb	cc

Рис. 5.11. Результат выполнения запроса

Если включить в бланк запроса поле только одной таблицы, например A (см. рис. 5.12), то результат выполнения запроса будет следующим: итоговый набор содержит записи таблицы A, но они продублированы: их не две, как можно было бы ожидать, а четыре (см. рис. 5.13).

Таким образом, сам факт присутствия в окне конструктора посторонней таблицы (B), не связанной с основной таблицей (A), используемой в запросе, приводит к дублированию отбираемых записей.

Появление лишних записей вызвано тем, что Access в этом случае создает такой же динамический набор, как и в предыдущем запросе (декартово произведение таблиц A и B). Однако на этот раз выводятся на экран не оба

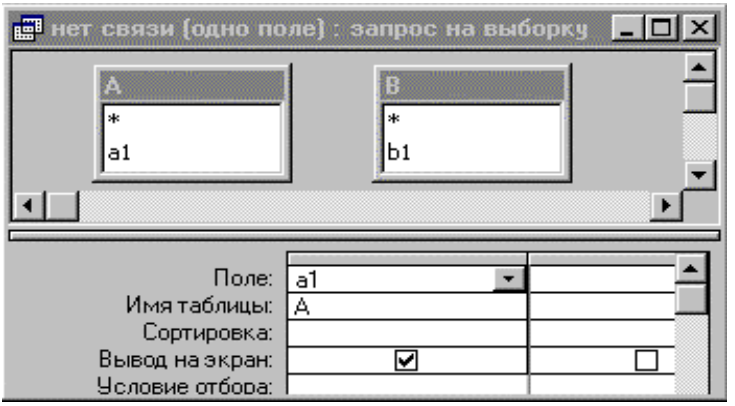


Рис. 5.12. Еще один запрос с несвязанными таблицами

столбца набора, а лишь первый столбец. Ясно, что число дубликатов записей таблицы A в итоговом наборе равно количеству записей, содержащихся в таблице B.

Инструкция SQL для запроса, представленного на рис. 5.12, имеет вид

SELECT A.a1 FROM A, B;

Она отличается от инструкции SQL предыдущего запроса лишь списком выводимых полей.

Рис. 5.13. Результат запроса

Итак, если в окне конструктора находятся две несвязанные таблицы, то Access рассматривает в качестве исходной таблицы их декартово произведение и число обрабатываемых им логических записей равно числу строк первой таблицы, умноженному на число строк второй таблицы.

[1] *Декартовым произведением* множеств X и Y называется множество Z, состоящее из элементов z=(x,y), где x — элемент множества X, а y — элемент множества Y.

Обычно наличие несвязанных таблиц в окне конструктора — следствие невнимательности пользователя, создающего запрос. Однако можно добавить таблицу в запрос, не связывая ее с другими таблицами, для того чтобы иметь возможность ссылаться на значения ее полей.

Например, для получения списка студентов младше студента Иванова можно поступить следующим образом (см. также пример 5.13):

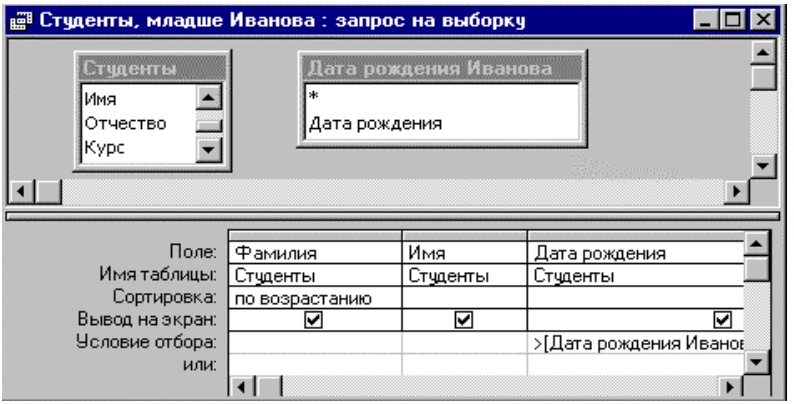


Рис. 5.14. Использование несвязанных таблиц

1. Создать и сохранить запрос, дающий дату рождения Иванова.
2. Поместить в окно конструктора таблицу **Студенты** и сохраненный запрос, не связывая их.
3. Поместить в бланк запроса поля **Фамилия**, **Имя**, **Дата рождения** и в поле **Дата рождения** задать условие отбора:

   >[Дата рождения Иванова]![Дата рождения]
4. Выполнить запрос.

Присутствие в окне конструктора запроса **Дата рождения Иванова** позволяет использовать его значение в условии отбора (см. рис. 5.14).

5.4.2. Внутреннее объединение таблиц

Чтобы установить связь между таблицами A и B, нужно переместить поле одной таблицы на связываемое с ним поле другой таблицы. Access покажет установленную связь в виде тонкой линии.

Включим оба связанных поля в бланк запроса и выполним созданный запрос (см. рис. 5.15). Результирующий набор данных содержит лишь одну запись, причем значения полей совпадают (см. рис. 5.16).

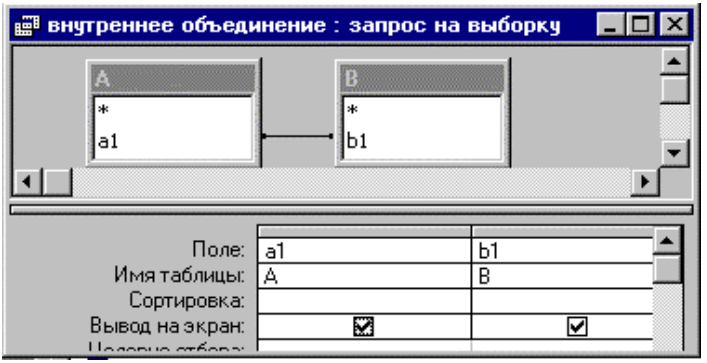


Рис. 5.15. Запрос со связанными таблицами

Мы видим, что при создании связи между таблицами A и B Access объединяет в динамическом наборе записи, имеющие одинаковые значения в связанных полях. Такой тип объединения называется *внутренним*. Конечно, если запрос содержит какие-либо условия на отбор записей, то в итоговый набор попадут лишь те записи, которые удовлетворяют этим условиям.

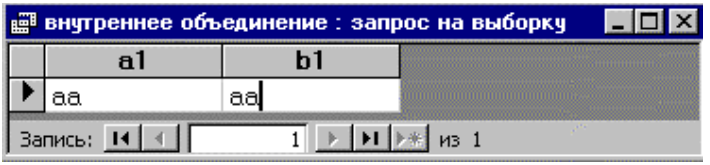


Рис. 5.16. Результат запроса со связанными таблицами

Инструкция SQL для созданного нами запроса имеет вид

SELECT A.a1, B.b1 FROM A INNER JOIN B ON A.a1 = B.b1;

Предложение FROM в ней выглядит не так, как в случае несвязанных таблиц. В нем добавилась операция INNER JOIN (внутреннее объединение), указаны объединяемые таблицы A и B и правило объединения записей *A.a1 = B.b1* — совпадение значений в связанных полях.

Итак, для установления связи между таблицами в режиме конструктора запроса нужно переместить с помощью мыши поле из списка полей одной таблицы в соответствующее поле из списка полей другой таблицы. Обычно

«перетаскивается» ключевое поле главной таблицы, выделенное жирным шрифтом, на поле внешнего ключа подчиненной таблицы. Связываемые поля должны иметь одинаковый или совместимый тип данных.

По умолчанию Access использует *внутреннее объединение*, при котором из обеих таблиц отбираются лишь записи, содержащие одинаковые значения в полях связи. Если значения в этих полях одинаковы, запрос объединяет две соответствующие записи из обеих таблиц и отображает их как одну запись в динамическом наборе данных. Если для записей одной таблицы нет соответствующих записей из другой таблицы, то динамический набор будет пустым.

Пример 5.14. Нужно узнать, как студенты 9701 группы сдали экзамен по информатике.

Фамилии студентов хранятся в таблице **Студенты**, а их оценки — в таблице **Сессия.** Поэтому создание запроса нужно начать с включения в окно конструктора этих таблиц. После того как списки полей таблиц появятся в окне конструктора, Access автоматически соединит их линией по полю **Код студента,** так как связь между этими таблицами типа «один-ко-многим» содержится в схеме данных.

Добавим в бланк запроса поле **Группа** из таблицы **Студенты** и поля **Оценка** и **Код экзамена** из таблицы **Сессия**. Для отбора нужных записей введем условие на значения в поле **Группа:** *9701* и условие на значения в поле **Код экзамена:** *1* (информатика). Отменим вывод этих полей на экран снятием флажков в строке **Вывод на экран.**

Для вывода фамилий и имен студентов создадим вычисляемое поле, введя в любую пустую ячейку строки **Поле** выражение:

Студент: [Студенты]![Фамилия] & " " & [Студенты]![Имя].

Для его создания удобно использовать построитель выражений.

Чтобы список появлялся на экране в отсортированном виде, установим для поля **Студент** в строке **Сортировка** значение *по возрастанию*. На этом создание запроса завершено (см. рис. 5.17).

В том случае, когда две таблицы непосредственно связать нельзя, для их объединения нужно использовать дополнительные таблицы или запросы.

Пример. 5.15. Требуется создать запрос, позволяющий определять, у каких студентов принял экзамен тот или иной преподаватель.

Так как таблицы **Студенты** и **Преподаватели** не имеют общих полей, то для установления связи между ними нужно добавить в окно конструктора таблицу **Сессия**, имеющую общие поля с обеими таблицами.

Связь между таблицами **Студенты** и **Сессия** устанавливается по полю Код студента, а между таблицами **Преподаватели** и **Сессия** — по полю **Код преподавателя**.

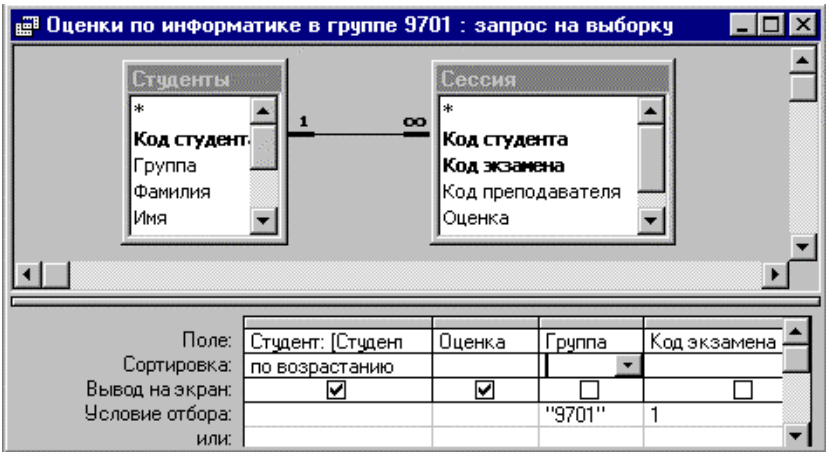


*Рис. 5.17. Внутреннее объединение таблиц **Студенты и Сессия***

Запрос должен выдавать список студентов для любого преподавателя. Поэтому следует использовать запрос с параметром. Параметр лучше задать в поле **Код преподавателя**, чтобы обеспечить получение правильного результата и в том случае, когда у экзаменатора есть однофамилец. Для этого нужно ввести в этом поле в строке **Условие отбора** текст в квадратных скобках примерно такого содержания: *Введите код преподавателя*. В бланк запроса следует также добавить поле **Фамилия** из таблицы **Преподаватели** и поля **Группа** и **Фамилия** из таблицы **Студенты.** Так как в запрос включены два поля с одинаковым именем, имеет смысл их переименовать, введя перед старым именем новое имя и отделив имена двоеточием. Затем созданный запрос (см. рис. 5.18) нужно сохранить. При его запуске Access попросит ввести код преподавателя и после того как это будет сделано, выдаст на экран фамилию преподавателя и список его студентов.

Созданный запрос обычно можно легко изменить. Например, если возникло желание узнать, какие оценки получили студенты, то нужно открыть запрос в режиме конструктора, добавить в бланк поле **Оценка** и сохранить сделанные изменения.

Access позволяет также объединять таблицы по совпадению значений в нескольких полях связи (см. пример 5.19).

5.4.3. Внешнее объединение таблиц

При внутреннем объединении запись таблицы включается в динамический набор лишь в том случае, если в связанной с ней таблице найдется запись, имеющая совпадающее значение в поле связи. Часто возникает необходимость включить в результат запроса и те (а иногда только те) записи таблицы, для которых в таблице, связанной с ней, отсутствуют соответствующие им записи.

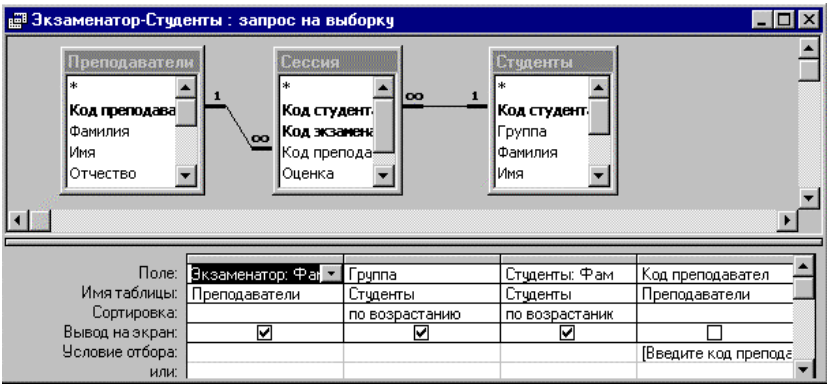


Рис. 5.18. Запрос к примеру 5.15

Пусть, например, нужно создать запрос, содержащий результаты сдачи экзамена в группе, причем итоговый список должен включать и фамилии студентов, по каким-то причинам не сдававших экзамен. Если использовать для связи таблиц **Студенты** и **Сессия** внутреннее объединение, то в итоговый список попадут лишь студенты, сдававшие экзамен (см. пример 5.14). Поэтому нужно использовать другой тип объединения таблиц.

Access позволяет изменить тип объединения таким образом, что из одной таблицы/запроса будут выбраны все записи, независимо от того, содержит ли вторая таблица/запрос соответствующие им записи. Для этого нужно сделать двойной щелчок мышью на линии объединения между таблицами, а в появившемся окне **Параметры объединения** выбрать подходящий тип объединения и нажать кнопку **OK**.

Например, если мы хотим выбрать все записи таблицы A и те записи таблицы B, которые имеют совпадающие значения в полях связи, нужно выполнить следующие действия:

1. Создать связь между таблицами A и B путем «перетаскивания» поля *a1* на поле *в1*.

2. Двойным щелчком по линии объединения открыть окно **Параметры объединения** и установить нужный тип объединения (см. рис. 5.19).
3. Щелкнуть по кнопке **ОК** и добавить поля *a1* и *в1* в окно запроса (см. рис. 5.20).
4. Щелкнуть по кнопке **Запуск**  и выполнить запрос (см. рис. 5.21).

Для внешнего объединения Access добавляет стрелку в конце линии объединения. В нашем примере стрелка направлена слева направо от таблицы A к таблице B, а выбранный второй тип объединения называется *левым внешним объединением*.

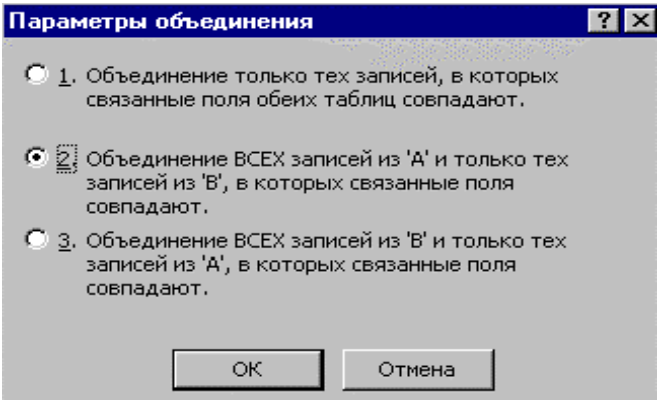


Рис. 5.19. Окно Параметры объединения

Если нужно отобрать все записи таблицы B и записи таблицы A, имеющие совпадающие значения в полях связи, то следует выбрать в окне **Параметры объединения** третий тип, называемый *правым внешним объединением*. При выборе этого типа стрелка в конце линии объединения будет направлена справа налево от таблицы B к таблице A.

Левому внешнему объединению (LEFT JOIN) в нашем примере соответствует инструкция SQL:

SELECT A.a1, B.b1 FROM A LEFT JOIN B ON A.a1 = B.b1;

а правому внешнему объединению (RIGHT JOIN) — инструкция SQL:

SELECT A.a1, B.b1 FROM A RIGHT JOIN B ON A.a1 = B.b1;

Отметим, что любой из описанных выше типов объединения таблиц может быть задан в окне **Связи** (см. рис. 2.6 на стр. 33) при определении связей между таблицами. Для этого следует щелкнуть по кнопке **Объединение.** Установленный тип связи будет использован по умолчанию Access при объединении таблиц в окне конструктора запросов.

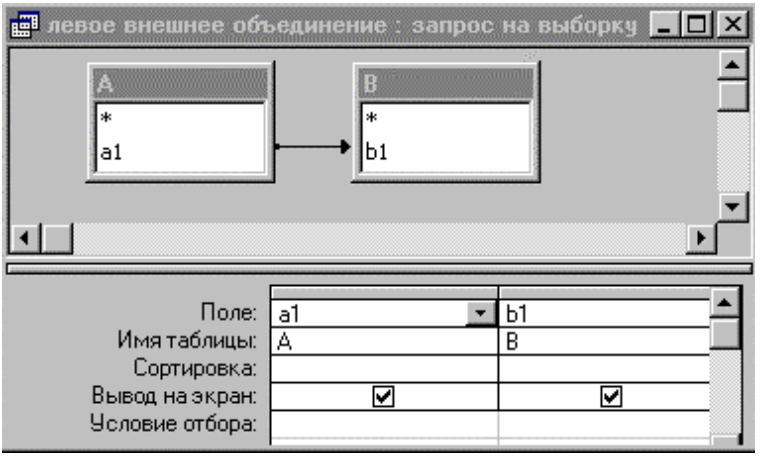


Рис. 5.20. Запрос с левым внешним объединением

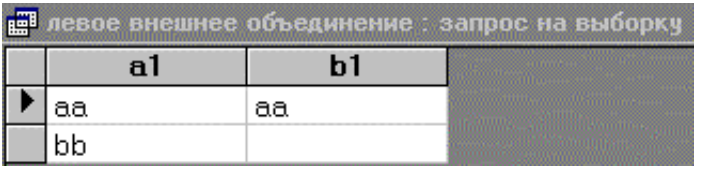


Рис. 5.21. Его результат

Пример 5.16. Нужно создать запрос **Адреса студентов**, содержащий список студентов курса, причем для проживающих в общежитии должен быть указан их адрес: номер общежития и номер комнаты.

Для этого следует добавить в окно конструктора две таблицы: **Студенты** и **Общежитие**, содержащие нужную информацию. Так как поле **Код студента** является ключевым в таблице **Студенты**, между ними будет автоматически установлена связь по этому полю, имеющая внутренний тип объединения. Если оставить эту связь без изменения, то Access отберет лишь записи с информацией о студентах, живущих в общежитии. Нам же нужен полный список студентов, поэтому этот тип объединения следует изменить на тип объединения, отбирающий все записи из таблицы **Студенты** и соответствующие им записи из таблицы **Общежитие**.

Затем нужно включить в бланк запроса поля с нужной информацией и сохранить в БД. После выполнения запроса появится список всех студентов курса с их адресами (см. рис. 5.22). Для студентов, не живущих в общежитии, поля **Общежитие** и **Адрес** остаются пустыми (содержат значение *Null*).

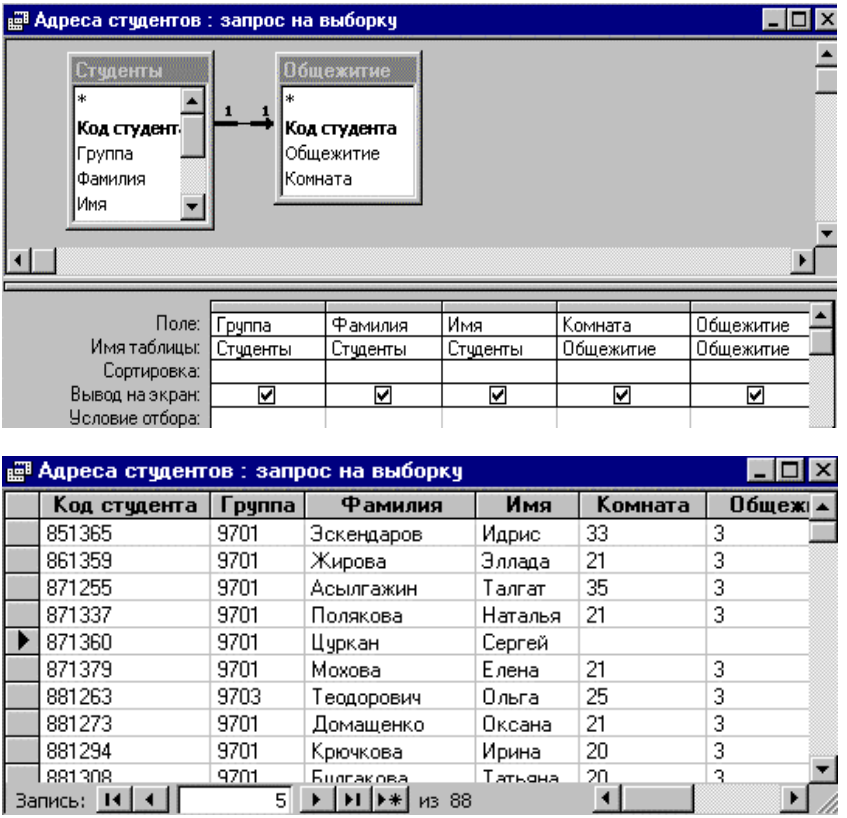


*Рис. 5.22. Запрос **Адреса студентов***

Использование мастера Записи без подчиненных

Созданный запрос можно легко превратить в запрос, который находит студентов, не живущих в общежитии. Для этого нужно лишь добавить в качестве условия отбора в поле **Комната** или **Общежитие** выражение *Is Null*. Для создания такого рода запросов можно также использовать мастера **Записи без подчиненных.**

Пример 5.17. Создадим с помощью этого мастера запрос, который находит студентов, не сдававших сессию.

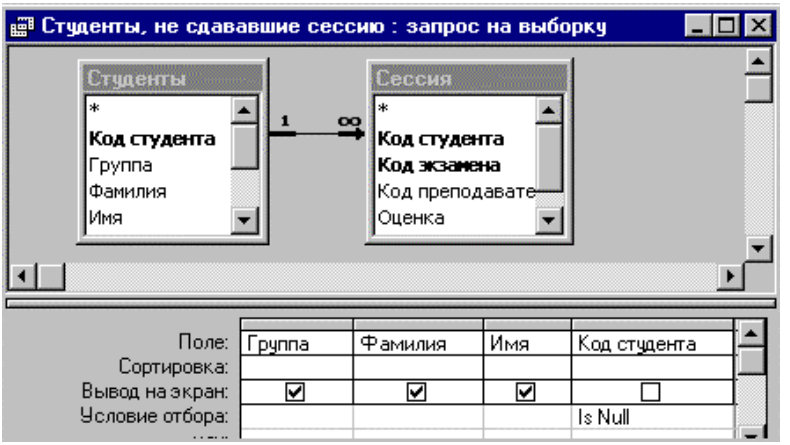


Рис. 5.23. Запрос, созданный мастером Записи без подчиненных

Таким студентам в таблице **Студенты** соответствуют записи, не имеющие связанных с ними записей в подчиненной таблице **Сессия**. После вызова мастера путем выбора соответствующего пункта в окне **Новый запрос** нужно последовательно ответить на следующие вопросы (ответ набран курсивом):

1. Указать таблицу, не имеющую связанных записей с таблицей, указываемой в следующем пункте: *Студенты*.
2. Указать таблицу с подчиненными записями: *Сессия*.
3. Указать поля, используемые для связи таблиц: в обеих таблицах *Код студента*.
4. Выбрать поля, отображаемые в результате выполнения запроса: *Группа, Фамилия, Имя.*
5. Задать имя запроса: *Студенты, не сдававшие сессию*.

Группа	Фамилия	Имя
9703	Теодорович	Ольга
9704	Марычева	Ольга

Рис. 5.24. Студенты, не сдававшие сессию

В результате выполнения созданного мастером запроса (см. рис. 5.23) на экране появятся сведения о студентах, не сдававших сессию (рис. 5.24).

5.4.4. Самообъединение

Иногда для получения нужной информации следует объединить таблицу саму с собой, создавая тем самым *самообъединение* таблицы. Такая ситуация, в частности, возникает, если в таблице есть поле, которое ссылается на другое поле той же таблицы.

Служащие : таблица

Код	Сотрудник	Подчиняется
1	Иванов	3
2	Петрова	4
3	Смирнов	
4	Сидоров	3

Рис. 5.25. Таблица Служащие

Пример 5.18. Пусть имеется таблица **Служащие**, содержащая поле **Подчиняется** с идентификационными номерами начальников, причем сведения о них хранятся в той же таблице. Если нужно, чтобы вместо номера стояла фамилия начальника, этого можно добиться, создав запрос, использующий копию таблицы **Служащие** (см. рис. 5.26).

Для создания этого запроса нужно включить в окно конструктора таблицу **Служащие** дважды. Второй раз она будет добавлена под именем **Служащие**-1, которое во избежание ошибок лучше заменить другим. Для

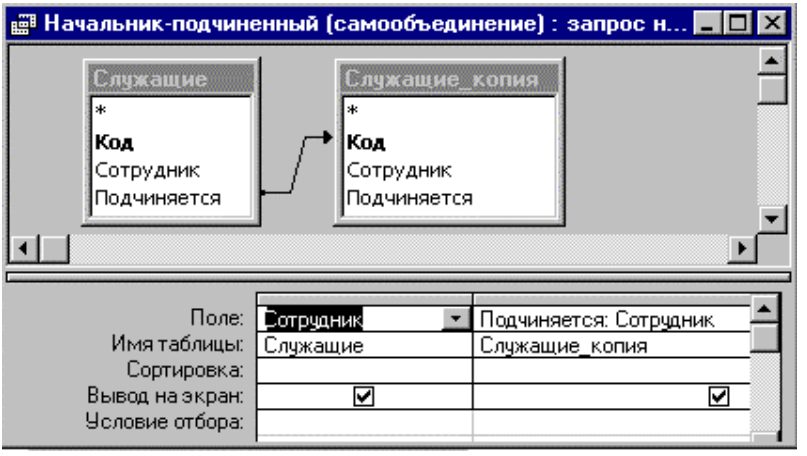


Рис. 5.26. Запрос, использующий самообъединение таблицы

этого нужно щелкнуть по копии таблицы правой кнопкой мыши, выбрать пункт **Свойства,** а затем в поле **Псевдоним** ввести текст ***Служащие_копия***.

Затем следует установить связь между таблицами, соединив поле **Подчиняется** таблицы **Служащие** с полем **Код** ее копии.

Сотрудник	Подчиняется
Иванов	Смирнов
Петрова	Сидоров
Смирнов	главный начальник
Сидоров	Смирнов

Рис. 5.27. Структура подчиненности

В качестве типа соединения лучше выбрать левое внешнее объединение, так как при внутреннем объединении в результат запроса не будет включена информация о самом главном начальнике. После этого следует включить в бланк запроса поле **Сотрудник** из обеих таблиц, изменив название этого поля из таблицы **Служащие_копия** на **Подчиняется**. Результат запроса представлен на рис. 5.27[1].

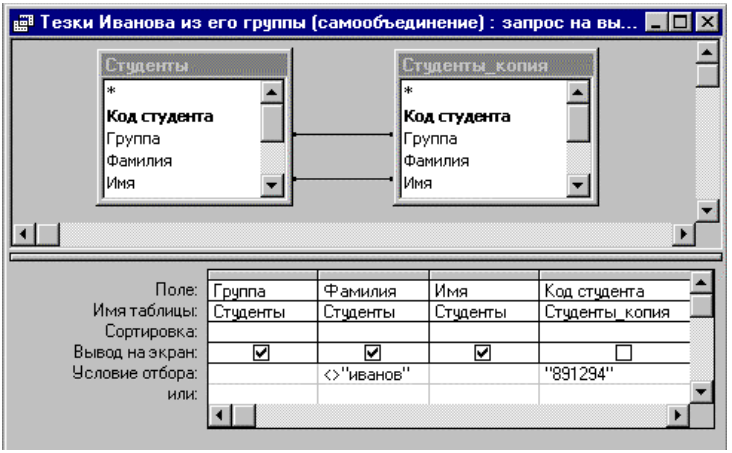


Рис. 5.28. Еще один пример на самообъединение

Самообъединение можно использовать и в других случаях.

Пример 5.19. Пусть нужно найти студентов, которые учатся в той же группе, что и студент Иванов (код 891294), и являются его тезками. Одним из возможных путей решения этой задачи является создание запроса, использующего самообъединение таблицы **Студенты** (см. рис. 5.28)[2].

[1] Чтобы в пустом поле напротив фамилии *Смирнов* появился текст, достаточно задать в свойствах поля Подчиняется такой формат: @;"главный начальник".

[2] Можно также использовать функцию *DLookUp*.

Группа	Фамилия	Имя
9702	Кривоносов	Евгений
9702	Мелентьев	Евгений
9702	Пентегов	Евгений

Рис. 5.29. Тезки Иванова

В этом запросе Access объединяет пары записей, имеющие одинаковые значения в полях **Группа** и **Имя.** После выполнения запроса в динамическом наборе остаются лишь три записи (см. рис. 5.29).

5.4.5. Объединение таблиц по отношению неравенства

В большинстве случаев таблицы объединяются по условию равенства значений в полях связи, но допустимо объединение таблиц и по условию «неравенства». В этом случае условием связи может быть любой из операторов сравнения: >, >=, <, <=, <>.

Для создания запроса, использующего такие условия связи, можно поступить следующим образом:

1. Создать в окне конструктора связь типа равенства.
2. Перейти в режим SQL и заменить знак "=" другим оператором сравнения**.** Однако такой запрос уже нельзя представить в режиме конструктора.

Пример 5.20. Заменив знак "=" на "<>", в приведенной на стр. 93 инструкции SQL, можно создать запрос, объединяющий записи таблиц A и B с несовпадающими значениями в связанных полях. Его результат виден на рис. 5.30. Соответствующая инструкция SQL имеет вид

SELECT A.a1, B.b1 FROM A INNER JOIN B ON A.a1 <> B.b1;

объединение с <> : запрос на выборку

a1	b1
bb	aa
aa	cc
bb	cc

Рис. 5.30. Связь по неравенству

Можно поступить иначе: не связывать таблицы в окне конструктора, а записать условие на несовпадение значений полей в строке условий. В нашем примере это условие накладывается на значения в поле *a1* и выглядит так: *<> [b1]*.

Соответствующая инструкция SQL имеет вид

SELECT A.a1, B.b1 FROM A, B WHERE (((A.a1)<>[b1]));

5.5. Подведение итогов

Иногда требуется, чтобы результатом выполнения запроса были не значения отдельных записей таблиц БД, а итоговые значения по группам записей. Например, может понадобиться определить средние баллы по экзаменам в каждой учебной группе или найти суммарную стипендию, полученную студентами в каком-то месяце. Получить ответ на такие вопросы можно с помощью *итогового запроса*.

5.5.1. Группировка записей в итоговом запросе

Для вычисления в запросе итоговых значений нужно щелкнуть по кнопке **Групповые операции** Σ на панели инструментов. В бланке запроса появится строка **Групповая операция** и для всех полей в этой строке будет установлено значение **Группировка**. Это же значение будет по умолчанию устанавливаться и для добавляемых в бланк запроса полей. Необходимо помнить, что в итоговом запросе Access группирует (объединяет в одну группу) записи с одинаковыми значениями в полях, имеющих в строке **Групповая операция** значение **Группировка**. Поэтому, если выполнить запрос, в бланке которого все поля имеют значения **Группировка**, то будет создан набор записей, включающий по одной записи из каждой группы.

Например, если включить в бланк запроса поле **Группа** таблицы **Студенты**, щелкнуть по кнопке **Групповые операции и** выполнить запрос, то Access выдаст на экран список номеров групп[1].

Однако никаких итогов Access не подведет. Для выполнения этой операции ему нужна следующая информация:

- по каким полям нужно подводить итоги;
- какие итоги нужно подводить по данному полю.

Поэтому в бланке запроса наряду с *групповыми полями*, по которым производится группировка, должны находиться *итоговые поля*, причем для каждого итогового поля должен быть указан тип подводимых итогов (сумма, среднее, максимум и т.п.).

Итоговым полем может быть как поле таблицы (запроса), находящейся в окне конструктора, так и специально созданное для этой цели вычисляемое поле. Для каждого итогового поля в строке **Групповая операция** должна быть указана статистическая функция, с помощью которой будет проводиться групповая операция — вычисление итогов в

[1] Фактически использование группировки по полю — быстрейший способ создания списка уникальных значений в этом поле.

этом поле. Имя нужной функции (*Sum*, *Avg*, *Max* и т.д.) можно ввести в соответствующую ячейку **в** строке **Групповая операция** или выбрать из раскрывающегося списка.

Пример 5.21. Найти число студентов в каждой учебной группе.

Добавим в окно конструктора таблицу **Студенты,** затем щелкнем по кнопке **Групповые операции** и включим в бланк запроса поле **Группа**. Оно будет использовано для группировки записей.

Поле:	Группа	Число студентов: Код студента
Имя таблицы:	Студенты	Студенты
Групповая операция:	Группировка	Count
Сортировка:	по возрастанию	
Вывод на экран:	☑	☑
Условие отбора:		

Рис. 5.31. Бланк итогового запроса (пример 5.21)

В качестве итогового поля выберем поле **Код студента**, а в качестве итоговой функции — функцию *Count.* Дадим итоговому полю имя **Число студентов.** Результат выполнения запроса представлен на рис. 5.32.

Группа	Число студентов
9701	19
9702	16
9703	19
9704	18
9705	15

Рис. 5.32. Его результат

Для подсчета числа студентов можно было бы взять в качестве итогового поля любое другое поле таблицы **Студенты.** Единственное условие, которому оно должно удовлетворять, — отсутствие пустых значений, так как функция *Count* игнорирует записи со значением *Null* в итоговом поле. Это условие заведомо выполнено для поля **Код студента,** так как оно является ключевым. Если же ни для одного поля таблицы нельзя гарантировать отсутствие пустых значений, то в этом случае для подсчета числа всех записей в каждой группе нужно создать вычисляемое поле, введя в строку **Поле** выражение *Count(*)* (см. рис.5.37 на стр. 108).

Часто для подведения итогов используется информация, содержащаяся в разных таблицах. В этом случае Access группирует записи результирующего набора.

Пример 5.22. Найти суммарные баллы, полученные студентами в сессию. Итоговый набор должен содержать следующие сведения: код студента, его фамилию, номер группы и итоговую сумму оценок по сданным экзаменам. Информация должна быть отсортирована по сумме набранных баллов.

Поле:	Код студента	Группа	Фамилия	Оценка
Имя таблицы:	Студенты	Студенты	Студенты	Сессия
Групповая операция:	Группировка	Группировка	Группировка	Sum
Сортировка:				по убыванию
Вывод на экран:	☑	☑	☑	☑
Условие отбора:				

Рис 5.33. Итоги сдачи сессии

Для построения этого запроса добавим в окно конструктора таблицы **Студенты** и **Сессия** и щелкнем по кнопке **Групповые операции**. Затем включим в бланк запроса поля **Код студента**, **Фамилия**, **Группа** и **Оценка**. У первых трех полей оставим тип операции **Группировка,** а в поле **Оценка** выберем в качестве групповой операции функцию Sum и зададим тип сортировки *по убыванию* (см. рис. 5.33). Построенный запрос сохраним под именем **Итоги сдачи сессии**.

Группировку записей Access производит для динамического набора, полученного в результате внутреннего объединения таблиц **Студенты** и **Сессия.** Фактически в качестве группового поля используется поле **Код студента**, задающее самое дробное разбиение на группы: в одну группу объединяются записи об оценках одного студента. Отметим, что даже если бы нам не нужно было выводить на экран код студента, это поле все равно необходимо включить в запрос в качестве группового. Иначе Access объединит в одну группу записи с оценками однофамильцев, и запрос выдаст неверный результат.

5.5.2. Отбор записей, формирующих группы

В том случае, когда из групп должны быть исключены некоторые записи, нужно добавить в бланк запроса поле или поля, которые будут использованы для отбора нужных записей. Для создания условия отбора по данному полю нужно выбрать в строке **Групповая операция** значение **Условие** и ввести условие в строку **Условие отбора.** Access автоматически отключит вывод на экран значений этого поля.

Пример 5.23. Определить число студентов в каждой учебной группе, родившихся в 1973 году или позднее.

Для этого нужно в бланк запроса из предыдущего примера (см. рис. 5.31) включить дополнительно поле **День рождения,** выбрать в строке **Групповая операция** значение **Условие и** ввести условие отбора >

31.12.72. Access добавит к дате с двух сторон # и бланк запроса будет иметь вид, представленный на рис. 5.34**.**

Поле:	Группа	Число студентов: I	Дата рождения
Имя таблицы:	Студенты	Студенты	Студенты
Групповая операция:	Группировка	Count	Условие
Сортировка:	по возрастанию		
Вывод на экран:	☑	☑	☐
Условие отбора:			>#31.12.72#

Рис 5.34. Отбор записей в группах

После добавления этого условия в Access включит в формируемые группы лишь те записи, которые относятся к студентам, родившимся позднее 31 декабря 1972 года.

Поле:	Группа	Оценка	Пол	Код экзамена
Имя таблицы:	Студенты	Сессия	Студенты	Сессия
Групповая операция:	Группировка	Avg	Условие	Условие
Сортировка:				
Вывод на экран:	☑	☑	☐	☐
Условие отбора:			"ж"	"1"

Рис. 5.35. Бланк запроса к примеру 5.23

Пример 5.24. Определить средние баллы по информатике девушек в каждой учебной группе.

Этот запрос использует данные из двух таблиц: **Студенты** и **Сессия**. В его бланк следует включить следующие поля (см. рис. 5.35):

Группа	Средний балл
9701	4,11
9702	3,67
9703	4,17
9704	4,00
9705	3,91

Рис 5.36. Итоговый результат

- **Группа** — используется для группировки записей;
- **Пол** и **Код экзамена** — служит для отбора нужных записей в группах;
- **Оценка** — по этому полю производится групповая операция *Avg* (вычисление среднего значения среди оценок по информатике).

Результат выполнения запроса представлен на рис. 5.36. Чтобы получить значения средних баллов в таком **(**отформатированном) виде необходимо настроить свойства итогового поля Оценка**.** Для этого нужно щелкнуть правой кнопкой мыши по полю **Оценка**, открыть список его

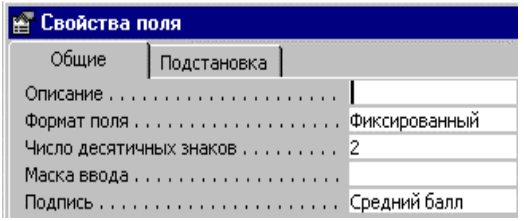


Рис 5.37. Установка свойств поля Оценка

свойств и установить для свойства **Формат поля** значение *Фиксированный*, для свойства **Число десятичных знаков** — *2*, а для свойства **Подпись** — *Средний балл* (см. рис. 5.37).

5.5.3. Отбор групп

В итоговый динамический набор можно включать не все группы. Для того чтобы исключить ненужные группы, следует ввести условия отбора в любое поле, содержащее в строке **Групповая операция** значение *Группировка* или одну из статистических функций.

Пример 5.25. Определить группы, в которых учится более 16 студентов.

Для этого достаточно добавить в поле **Число студентов** бланка запроса из примера 5.21 (см. рис. 5.31) условие отбора: *> 16*. В результате выполнения этого запроса будут найдены номера учебных групп.

Поле:	Группа	Число студентов: Count(*)
Имя таблицы:	Студенты	
Групповая операция:	Группировка	Выражение
Сортировка:	по возрастанию	
Вывод на экран:	☑	☑
Условие отбора:		>16

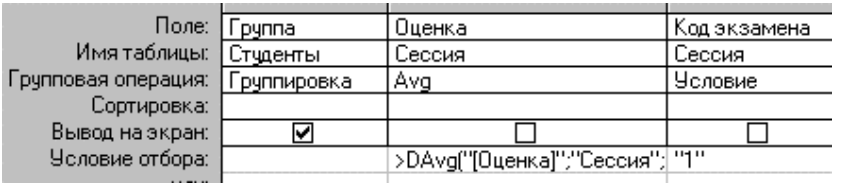
Рис. 5.38. Использование функции Count(*)

Для подсчета числа студентов в учебной группе можно также использовать функцию *Count(*)*. В этом случае бланк запроса будет иметь вид, представленный на рис. 5. 38.

Строка условий может содержать выражение, в состав которого входит результат некоторой статистической операции. Для вычисления этого результата должна использоваться не обычная статистическая функция, а статистическая функция по подмножеству.

Пример 5.26. Определить учебные группы, имеющие по информатике более высокий средний балл, чем на курсе в целом.

Запрос использует информацию из таблиц **Студенты** (номера групп) и **Сессия** (оценки). В бланк итогового запроса следует включить следующие поля (см. рис. 5.39):



Поле:	Группа	Оценка	Код экзамена
Имя таблицы:	Студенты	Сессия	Сессия
Групповая операция:	Группировка	Avg	Условие
Сортировка:			
Вывод на экран:	☑	☐	☐
Условие отбора:		>DAvg("[Оценка]";"Сессия";	"1"

Рис. 5.39. Использование функции Davg в условии отбора

- **Группа —** используется для группировки записей;
- **Оценка** — используется как итоговое поле для подсчета средних оценок в группах, а также для отбора нужных групп записей;
- **Код экзамена** — это поле служит для отбора записей в группах (код экзамена по информатике равен 1).

Условие отбора групп выглядит так:

>DAvg("[Оценка]";"Сессия";"[Код экзамена]='1'")

Результат вычисления значения функции *DAvg* — средний балл по информатике на курсе.

Для нахождения среднего балла на курсе можно также использовать подчиненный запрос. В этом случае условие отбора в поле **Оценка** будет таким:

>(SELECT DISTINCTROW Avg(Сессия.Оценка) AS Avg_Оценка
FROM Сессия WHERE (((Сессия.[Код экзамена])="1"));)

Пример 5.27. Определить общую сумму стипендии, полученной студентами групп 9701-9703 в сентябре, октябре и за оба месяца.

Так как информация о распределении студентов по группам содержится в таблице **Студенты**, а информация о стипендии — в таблице **Стипендия**, то создание запроса начнем с включения этих таблиц в окно конструктора. Они связываются по полю **Код студента**. Затем нужно щелкнуть по кнопке **Групповые операции** и включить в бланк запроса поле **Группа**, которое будет использоваться для группировки записей. Для отбора нужных групп следует ввести условие *< 9704* или другое равносильное ему условие, например, *9701 Or 9702 Or 9703.*

Следующий шаг — создание итоговых полей, подсчитывающих суммарную стипендию за каждый месяц. Для этого нужно включить соответствующие поля в бланк запроса, выбрать в качестве групповой

функции *Sum* и заменить названия, которое Access дает итоговым полям (*Sum_<месяц>)* на более подходящие.

На завершающем этапе построения запроса добавим итоговое поле, подсчитывающее суммарную стипендию за оба месяца. Для этого создадим вычисляемое поле, введя в пустую ячейку строки **Поле** выражение

Nz([Стипендия]![Сентябрь])+Nz([Стипендия]![Октябрь])[1]

и выберем в качестве групповой функции *Sum*. Это поле тоже нужно переименовать и, вызвав его свойства, установить ему в качестве формата вывода значений *Денежный*. Результаты запроса представлены на рис. 5.40.

Группа	Сентябрь	Октябрь	Всего
9701	280 000р.	295 000р.	575 000р.
9702	235 000р.	250 000р.	485 000р.
9703	305 000р.	305 000р.	610 000р.

Рис. 5.40. Стипендия за два месяца

Интересно, что Access модифицировал созданный нами запрос, внеся следующие изменения в итоговое поле **Всего** (см. рис. 5.41)**:**

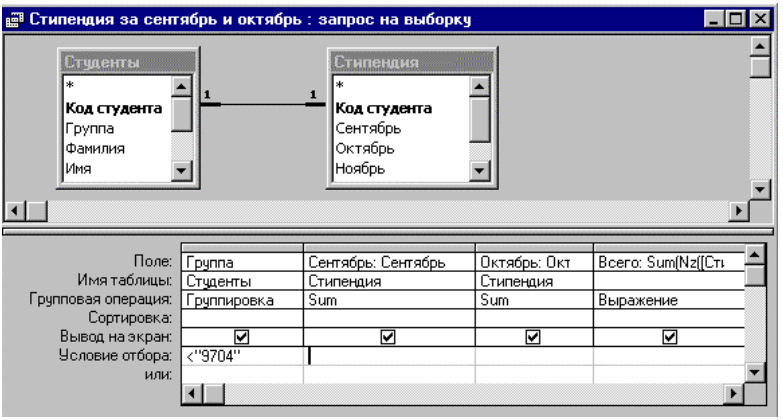


Рис 5.41. Отбор групп в итоговом запросе

- выражение, которое вычисляется в этом поле, заменено выражением *Sum(Nz([Стипендия]![Сентябрь])+Nz([Стипендия]![Октябрь]))*
- в строке **Групповая операция** групповая функция *Sum* заменена групповой операцией *Выражение*[2].

[1] О том, как это сделать с помощью построителя выражений, и почему использована функция *Nz,* см. пример 4.1 и замечание к нему.

[2] Эта групповая операция обычно выбирается, если в выражении в строке **Поле** используются одна или несколько групповых функций**.**

В запрос можно включить дополнительные итоговые поля. Например, для подсчета числа студентов, получающих стипендию в сентябре, нужно добавить в бланк поле **Сентябрь** и выбрать групповую функцию *Count*.

5.6. Перекрестные запросы

Перекрестный запрос — это итоговый запрос специального типа, выводящий результат в виде *перекрестной таблицы*, похожей на электронную таблицу.

Для создания перекрестной таблицы нужно указать:

- *заголовки строк* — одно или несколько полей, значения которых будут использованы в качестве заголовков строк таблицы;
- *заголовки столбцов* — поле, значения которого будут использованы в качестве заголовков столбцов;
- *значения* — числовое поле, значения которого будут использованы для проведения итоговых расчетов;
- *групповую операцию* — групповую функцию, задающую тип итоговых расчетов, или выражение, содержащее групповые функции.

Построение перекрестной таблицы производится примерно по такой схеме. Access производит группировку данных по групповым полям, которые также служат заголовками ее строк и столбцов. Групповая операция проводится над числами, содержащимися в поле значений. Итоговое значение для данной группы помещается в ячейку таблицы, находящуюся на пересечении строк и столбца, названия которых берутся из соответствующих групповых полей. Часть ячеек перекрестной таблицы могут оказаться пустыми (содержащими значение *Null)*. Это обстоятельство необходимо учитывать при создании выражений, включающих поля перекрестной таблицы.

Перекрестные запросы обычно используются при создании диаграмм и отчетов. Сохранить перекрестную таблицу в виде обычной таблицы Access с помощью запроса на создание таблицы (см. стр. 124) нельзя.

5.6.1. Мастер *Создание перекрестных таблиц*

Для создания перекрестного запроса можно воспользоваться соответствующим мастером. При этом нужно иметь в виду, что он создает перекрестный запрос на базе полей одной таблицы/запроса, причем группировка проводится по всем записям. Если для создаваемого перекрестного запроса такая таблица/запрос не существует, то следует сначала создать запрос, содержащий всю необходимую информацию, и

лишь затем воспользоваться услугами мастера. Рассмотрим работу мастера на следующем примере из БД **Книги**.

Пример 5.28. Требуется создать запрос, содержащий сведения о количестве книг, проданных продавцами в каждом квартале и за весь 1997 год.

Так как нужная информация находится в разных таблицах, создадим запрос, который будет использован мастером в качестве источника данных при построении перекрестной таблицы. Для этого поместим в окно конструктора таблицы **Продавцы** и **Заказы**. Access свяжет их по полю **Код продавца**. В бланк запроса включим поле **Код продавца**, а также поля **Количество** и **Дата отправки** со сведениями о количестве книг в заказе и дате его отправки. Создадим вычисляемое поле **Продавец**, содержащее фамилии и имена продавцов. Для этого в пустую ячейку строки **Поле** введем выражение *Продавец: [Фамилия] & " " & [Имя]*. В поле **Дата отправки** введем условие отбора заказов, выполненных в 1997 году, и сохраним запрос под именем «*Заказы 1997 года*» (см. рис. 5.42).

Поле:	Код продавца	Продавец: [Фамилия] & "	Дата отправки	Количество
Имя таблицы:	Продавцы		Заказы	Заказы
Сортировка:				
Вывод на экран:	☑	☑	☑	☑
Условие отбора:			Like "*.*.97"	

Рис. 5.42. Бланк запроса с информацией для мастера

Для вызова мастера нужно выбрать пункт **Перекрестный запрос** в окне **Новый запрос.** Работа мастера состоит из следующих шагов.

1. Мастер предлагает указать таблицу или запрос, поля которых будут использованы в создаваемом запросе. В списке запросов нужно найти запрос «**Заказы 1997 года»** и щелкнуть по кнопке **Далее** >.
2. Мастер просит указать поля, значения которых будут использованы в качестве заголовков строк (не более трех). Выберем в левом окне, содержащем список доступных полей, поля **Код продавца** и **Продавец**, перебросим их с помощью кнопки > в правое окно и перейдем к следующему окну диалога.
3. Мастер просит указать поле, значения которого будут использованы в качестве заголовков столбцов. Выберем поле **Дата отправки**.
4. Так как выбранное поле имеет тип **Дата**/**Время**, то мастер предлагает выбрать временной интервал, с которым нужно сгруппировать данные в этом поле. Выберем *Квартал*.
5. На этом этапе мастер просит указать итоговое поле и групповую функцию. Выберем поле **Количество**, а в качестве групповой функции

укажем *Sum*. Здесь же предоставляется возможность указать, нужно ли включать в таблицу итоговое значение по каждой строке. Поскольку мы хотим включить в таблицу данные об общем числе книг, проданных каждым продавцом, нужно оставить «галочку» во флажке с подписью *Да*. В противном случае «галочку» нужно убрать.

6. В заключение мастер предлагает дать имя созданному запросу и указать действия после его создания. Назовем запрос «**Работа продавцов в 97 году**» и выберем просмотр результатов запроса.

Созданная таблица имеет лишь один недостаток: столбец с итогами работы каждого продавца за год получил название **Итоговое значение Количество**. Для его устранения перейдем в режим конструктора и заменим имя соответствующего поля в бланке запроса на имя **За год**. Затем перейдем в режим таблицы и переместим этот столбец в конец таблицы. Сохраним все сделанные изменения. Теперь итоговая таблица имеет такой вид (см. рис. 5.43):

Работа продавцов в 97 году : перекрестный запрос

Код	Продавец	Кв 1	Кв 2	Кв 3	Кв 4	За год
1	Ильина Елена	4	50	18	43	115
2	Иванова Наталья	11	18	12	47	88
3	Петров Андрей	14	17	15	56	102
4	Костин Сергей	20	19	20	28	87
5	Сидоров Андрей	9	16	20	16	61

Рис. 5.43. Итоги работы продавцов в 1997 году

> *При создании перекрестной таблицы на базе многотабличного запроса возможна ситуация, когда в него включены одноименные поля, принадлежащие различным таблицам, например, поля **Фамилия** из таблиц **Продавцы** и **Покупатели**. В этом случае перед использованием мастера такие поля следует переименовать. Если этого не сделать, то Access создаст неработоспособный перекрестный запрос.*

5.6.2. Создание перекрестного запроса без помощи мастера

Для создания перекрестного запроса вручную необходимо:

1. Выбрать в окне **Новый запрос** пункт **Конструктор.**
2. Включить в верхнюю часть окна конструктора все таблицы и запросы, данные которых будут использованы в создаваемом запросе.

3. На панели инструментов нажать кнопку Тип запроса  и выбрать **Перекрестный**. В бланке запроса появятся две дополнительные строки: **Групповая операция** и **Перекрестная таблица.**
4. Добавить поля в строку **Поле** в бланке запроса и создать, если нужно, дополнительно вычисляемые поля. Все поля вначале будут получать в строке **Групповая операция** значение **Группировка**.
5. Для полей, значения которых будут использованы в качестве заголовков строк, нужно в строке **Перекрестная таблица** указать значение **Заголовки строк** и оставить в строке **Групповая операция** значение **Группировка**.
6. Для поля, значения которого будут использованы в качестве заголовков столбцов, нужно в строке **Перекрестная таблица** задать значение **Заголовки столбцов** и оставить в строке **Групповая операция** значение **Группировка**.
7. Для поля, значения которого будут использованы при создании перекрестной таблицы, нужно в строке **Перекрестная таблица** задать значение **Значения,** а в строке **Групповая операция** выбрать групповую функцию, используемую для вычисления значений перекрестной таблицы (например, *Sum* или *Count*).
8. Чтобы отобрать нужные заголовки строк или столбцов, следует ввести выражение, задающее условие отбора в строку **Условие отбора** для соответствующих полей-заголовков в ячейке строки **Перекрестная таблица**.

   Если нужно исключить некоторые записи до вычисления значений перекрестной таблицы, то следует добавить в бланк запроса поле или поля, которые будут использованы для отбора нужных записей. Для создания условия отбора по данному полю нужно выбрать в строке **Групповая операция** значение **Условие** и ввести условие в строку **Условие отбора. Я**чейка в строке **Перекрестная таблица** должна быть пустой.

   Если при отборе записей запроса используются параметры, то для них обязательно должен быть указан их тип.

Пример 5.29. Создадим перекрестный запрос **Средние баллы по группам**. Его результатом должна быть таблица, столбцами которой являются средние баллы в группах по данному экзамену, а заголовками столбцов служат названия экзаменов (см. рис. 5.44).

Группа	Английский	Информатика	История	Философия
9701	3,88	4,24	4,44	4,05
9702	3,94	4,12	4,81	4,31
9703	4,33	4,22	4,44	4,35
9704	3,81	4,00	4,06	4,47
9705	3,67	3,80	4,53	4,13

Рис .5.44. Перекрестная таблица ***Средние баллы по группам***

В этом запросе используются данные из таблиц **Студенты**, **Сессия** и **Экзамены**. После их добавления в окно конструктора укажем тип создаваемого запроса: **Перекрестный**. Затем включим в бланк запроса следующие поля: **Группа**, **Экзамен** и **Оценка.** Каждое из них получит в строке **Групповая операция** значение **Группировка**. Так как значения из поля **Группа** будут служить заголовками строк, выберем для него в строке **Перекрестная таблица** значение **Заголовки строк.** Соответственно для

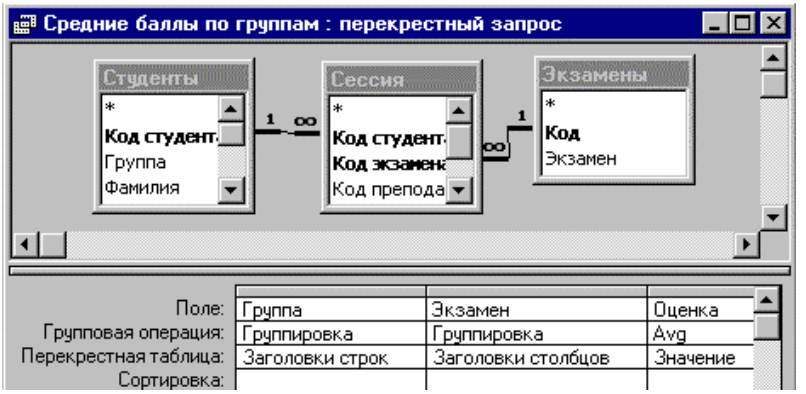


Рис. 5.45. Бланк запроса ***Средние баллы по группам***

поля **Экзамен** выберем для него в этой строке значение *Заголовки столбцов*. Поле **Оценка** используется для подсчета среднего балла. Поэтому выберем для него в строке **Групповая операция** значение *Avg*, а в строке **Перекрестная таблица —** *Значение*. Чтобы значения таблицы выдавались в отформатированном виде, зададим следующие свойства этого поля: для **Формат поля** — *Фиксированный* и для **Число десятичных знаков** — *2*.

Сохраним созданный запрос под именем **Средние баллы по группам** (см. рис. 5.45). В дальнейшем (в главе 6) мы используем его для построения диаграммы.

Использование вычисляемых полей в перекрестном запросе

В перекрестном запросе наравне с обычными полями в качестве заголовков могут использоваться вычисляемые поля.

Пример 5.30. Создадим перекрестный запрос **Итоги сессии на курсе**. Итоговая таблица должна содержать ФИО студентов, номера их учебных групп, номера зачетных книжек и оценки по экзаменам. Информация в ней должна быть отсортирована по группам, а затем по фамилиям.

Группа	Код студента	ФИО: [Студенты]	Экзамен	Оценка
Студенты	Сессия		Экзамены	Сессия
Группировка	Группировка	Группировка	Группировка	First
Заголовки строк	Заголовки строк	Заголовки строк	Заголовки столбц	Значение
по возрастанию		по возрастанию		

*Рис. 5.46. Бланк запроса **Итоги сессии на курсе***

В этом запросе также используются данные из таблиц **Студенты**, **Сессия** и **Экзамены**. Добавим их в окно конструктора и укажем тип создаваемого запроса: *Перекрестный.* Затем включим в бланк запроса следующие поля: **Группа**, **Код студента, Экзамен** и **Оценка.** Зададим для первых двух полей в строке **Перекрестная таблица** значение *Заголовки строк***,** а для поля **Экзамен** — *Заголовки столбцов***.** Для поля **Оценка** укажем в качестве групповой операции *First*, а для строки **Перекрестная таблица —** *Значение.* Выбор групповой операции в данном случае особой роли не играет (можно взять *Sum* или *Avg*), так как после группировки записей в поле значений каждой группы содержится не более одного числа (оценки студента по экзамену).

Создадим вычисляемое поле **ФИО,** и также выберем для него в строке **Перекрестная таблица** значение *Заголовки строк***.** Разместим это поле после поля **Группа** и зададим в полях **Группа** и **ФИО** сортировку по возрастанию. Для поля **Код студента** зададим в качестве свойства **Подпись** значение *№ зач_кн*. На этом создание запроса завершено. Сохраним его под именем **Итоги сессии на курсе.**

Добавление итогового столбца в перекрестную таблицу

В перекрестную таблицу можно добавить один или несколько столбцов, содержащих итоговые сведения по строке. Для добавления итогового столбца следует еще раз включить в бланк запроса поле, значения которого используются при создании перекрестной таблицы. В

строке **Групповая операция** нужно выбрать подходящую итоговую функцию, а в строке **Перекрестная таблица** — значение *Заголовки строк*.

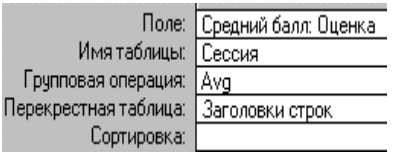

Итоги сессии на курсе : перекрестный запрос

Группа	№ зач_кн	ФИО	Английский	Информатика
9701	891302	Касымбеков Мурат Сарсе	4	5
9701	891303	Кенжебаева Шолпан Саду	4	5
9701	891307	Коняев Сергей Николаеви	3	5
9701	881294	Крючкова Ирина Алексее	4	4
9701	891311	Куликов Сергей Владимир	4	5
9701	891314	Лукьянов Александр Ген	5	5
9701	871379	Мохова Елена Иосифовна	2	4
9701	871337	Полякова Наталья Геннад	3	
9701	871360	Цуркан Сергей Павлович	4	4
9702	851365	Эскендаров Идрис Яхулов	4	4
9702	891255	Андрина Наталья Альберт	4	2
9702	891266	Борисов Дмитрий Юрьевич	4	5

Запись: 1 из 86

Рис. 5.47. Итоги сессии на курсе

Например, чтобы включить в перекрестную таблицу **Итоги сессии на курсе** столбец со средними баллами студентов и сделать его последним в таблице, достаточно выполнить следующие действия:



Поле:	Средний балл: Оценка
Имя таблицы:	Сессия
Групповая операция:	Avg
Перекрестная таблица:	Заголовки строк
Сортировка:	

Рис. 5.48. Создание итогового столбца

1. Включить в бланк запроса поле **Оценка** из таблицы **Сессия**.
2. Выбрать в строке **Групповая операция** значение *Avg*, а в строке **Перекрестная таблица** — значение *Заголовки строк*.
3. Дать этому столбцу новое название **Средний балл** и установить в его свойствах нужный формат вывода значений.
4. Перейти в режим просмотра и сделать столбец **Средний балл** последним в итоговой таблице.

Отметим, что добавление итоговой строки в перекрестную таблицу невозможно, так как результат запроса может содержать строки либо с детализированной информацией, либо с агрегированной. Однако в формах и отчетах это вполне допустимо.

Использование свойства «Заголовки столбцов»

Иногда требуется изменить порядок следования столбцов в итоговой таблице. Для этого нужно, находясь в режиме конструктора, вызвать щелчком мыши по кнопке **Свойства** на панели инструментов свойства запроса. Затем щелкнуть на свойстве **Заголовки столбцов** и ввести текст заголовков в нужном порядке, отделяя их друг от друга точкой с запятой или запятой в зависимости от установок Windows.

> *1. В перекрестной таблице появятся лишь те столбцы, чьи имена указаны в свойстве **Заголовки столбцов.** Это обстоятельство можно использовать для отбора нужных столбцов.*
> *2. Если какое-то имя введено с ошибкой, то в таблице появится пустой столбец с этим именем.*

Пример 5.31. Нужно **с**оздать перекрестный запрос с информацией о заказах покупателей, живущих в данном городе, за последние три месяца 1997 года. Название города — параметр запроса. Заголовками строк должны быть фамилии и инициалы покупателей, заголовками столбцов — названия месяцев в формате «ммм-гг», а значениями таблицы — суммарная стоимость заказов, сделанных покупателем в данном месяце.

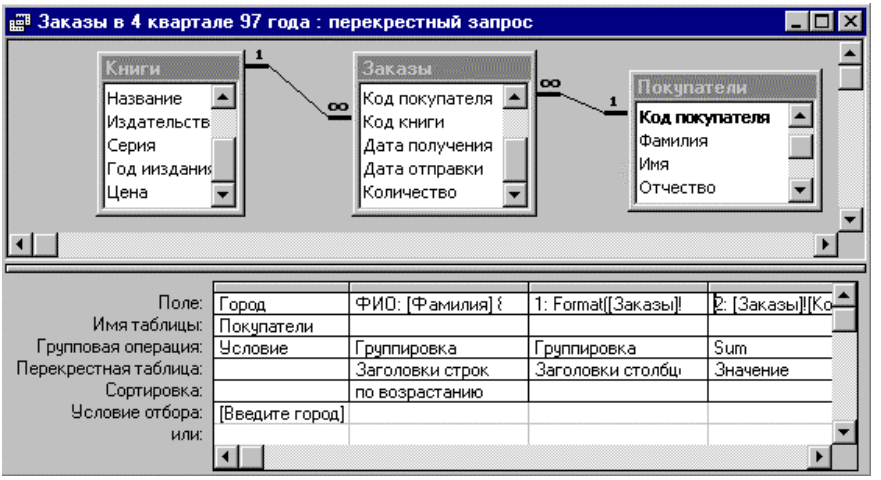


Рис. 5.49. Перекрестный запрос с параметром

В этом запросе используется информация из таблиц **Покупатели**, **Заказы** и **Книги**. После их добавления в окно конструктора укажем тип создаваемого запроса: *Перекрестный.*

Затем включим в бланк запроса поле **Город** и введем в строке **Условие отбора** в качестве имени параметра текст *Введите город* и заключим его в

квадратные скобки. Выберем пункт меню **Запрос**, а затем **Параметры**. В окне **Параметры запроса** введем имя этого параметра и укажем его тип: *Текстовый*. В строке **Групповая операция заменим** значение *Группировка* на *Условие*.

Создадим вычисляемое поле ФИО, которое будет использоваться в качестве заголовков строк. Для этого введем в строку **Поле** выражение

ФИО: [Фамилия] & " " & Left([Имя];1) & ". " & Left([Отчество];1) & ".",

в строке **Перекрестная таблица** укажем значение *Заголовки строк*, а в строке **Сортировка** — *по возрастанию*.

Для заголовков столбцов также создадим вычисляемое поле *Format ([Заказы] ![Дата отправки];«mmm-yy»)*. В строке **Перекрестная таблица** укажем значение *Заголовки столбцов*. Затем вызовем свойства запроса и в **свойстве Заголовки столбцов** введем текст заголовков *окт-97; ноя-97; дек-97* (см. рис. 5.50).

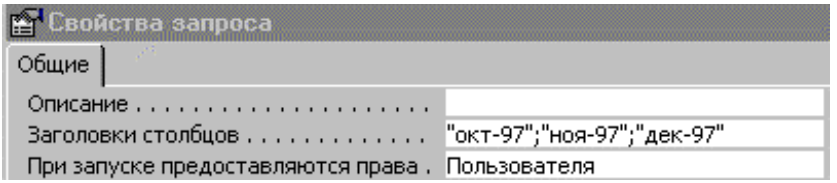


Рис. 5.50. Ввод заголовков столбцов

Для вычисления значений итоговой таблицы создадим еще одно поле, введя в строку **Поле** выражение *[Заказы]![Количество]*[Книги]![Цена]*. Для этого поля нужно в строке **Перекрестная таблица** выбрать *Значение*, а в строке **Групповая операция** — *Sum*.

На этом создание запроса завершено. При его выполнении Access попросит указать город, в котором живут покупатели. После ввода текста *новосибирск* появится таблица со сведениями о заказах покупателей, живущих в Новосибирске (см. рис. 5.51).

ФИО	окт-97	ноя-97	дек-97
Буланов П. С.		225р.	43р.
Конев С. Л.	186р.		
Кузнецов С. И.		124р.	381р.
Лукьянов Д. В.			
Смирнов И. В.			115р.

Рис. 5.51. Заказы покупателей из Новосибирска

Задание фиксированных заголовков столбцов позволяет уменьшить время выполнения запроса. Если перекрестный запрос используется как

базовый для подчиненной формы, то явное указание имен столбцов в этом свойстве необходимо для установления связи между подчиненной формой и запросом.

5.7. Запросы на изменение

Запросы этого типа позволяют автоматизировать изменения данных в таблицах, а также сохранить результат запроса в виде таблицы Access.

5.7.1. Запрос на добавление

С помощью запроса на добавление можно скопировать данные из одной таблицы (источника) и поместить их в другую таблицу (получатель). Причем можно добавить в таблицу-получатель как записи целиком, так и содержимое отдельных полей таблицы-источника.

Для создания запроса на добавление нужно выполнить следующие действия:

1. Включить в окно конструктора таблицу-источник. Затем выбрать пункт меню **Запрос,** а потом **Добавление.** На экране появится диалоговое окно **Добавление** (см. рис 5.52).
2. В поле **Имя таблицы** указать имя таблицы-получателя и затем нажать кнопку **OK**. Если таблица находится в другой БД, то нужно перед нажатием кнопки выбрать параметр **В другой базе данных** и ввести полное имя этой БД.

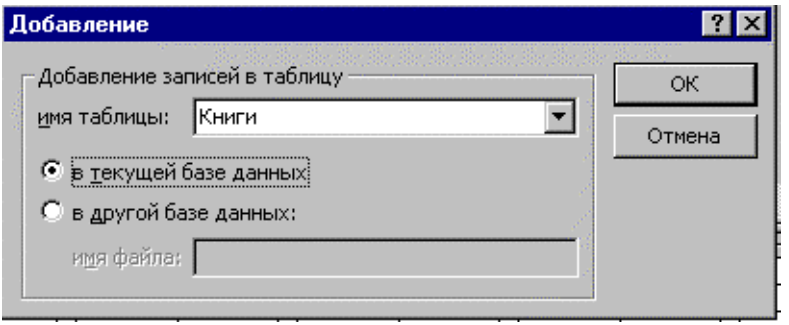


*Рис. 5.52. Диалоговое окно **Добавление***

3. В бланк запроса следует включить:
   - поля, участвующие в добавлении;
   - поля, для которых задаются условия отбора;
   - поле, соответствующее ключевому полю таблицы-получателя[1].

[1]Если ключевое поле в таблице-получателе имеет тип *Счетчик*, то этого делать не нужно.

Если все поля в обеих таблицах имеют одинаковые имена, то можно просто переместить символ «звездочка» (*) в бланк запроса. Если имя добавляемого поля совпадает с именем соответствующего поля таблицы-получателя, то имя поля таблицы-получателя будет автоматически введено в строку **Добавление**. Если же имена полей таблиц отличны друг от друга, то в строке **Добавление** нужно указать имя поля таблицы-получателя, которое соответствует добавляемому полю таблицы-источника. Затем следует ввести условия отбора записей из таблицы-источника в строку **Условие отбора**.

4. Для просмотра добавляемых записей нужно щелкнуть по кнопке **Вид**  на панели инструментов. Затем следует вернуться в режим конструктора, повторно щелкнув по этой кнопке.

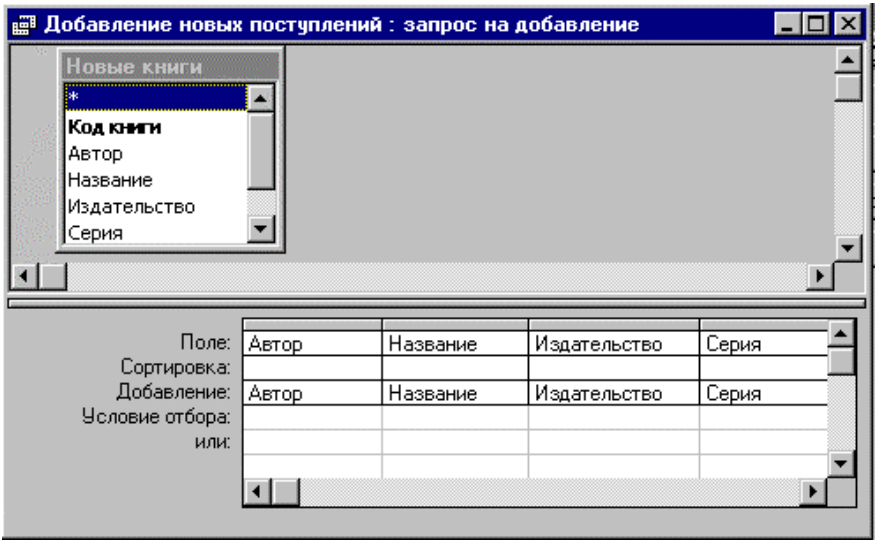


Рис. 5.53. Запрос на добавление в режиме конструктора

5. Для добавления записей нужно щелкнуть по кнопке **Запуск**  на панели инструментов. Access сообщит число добавляемых записей и попросит подтвердить выполнение операции. После подтверждения записи будут добавлены в таблицу-получатель.

Пример 5.32. Нужно добавить в таблицу **Книги** сведения о новых поступлениях книг. Эти сведения содержатся в таблице **Новые книги,** имеющей такую же структуру, что и таблица **Книги,** кроме поля **Цена.**

Создание запроса начнем с включения в окно конструктора таблицы-источника **Новые книги**. Затем вызовем контекстное меню запроса и

выберем сначала **Тип запроса,** а потом пункт **Добавление**. В появившемся окне **Добавление** укажем имя таблицы-получателя *Книги*.

В бланк запроса поместим все поля таблицы **Новые книги** за исключением ключевого поля **Код книги**, так как оно имеет тип *Счетчик*. Поскольку имена полей в обеих таблицах совпадают, Access сам сопоставляет с каждым полем таблицы-источника соответствующее поле таблицы-получателя (см. рис. 5.53).

После выполнения запроса в конец таблицы **Книги** будут добавлены записи с информацией о новых книгах. Access автоматически введет значения в поле **Код книги**. Поле **Цена** в добавленных записях будет пустым, так как сведения о ценах книг отсутствовали в таблице-источнике.

5.7.2. Запрос на удаление

Запрос на удаление позволяет удалить ненужные записи из таблицы. С его помощью можно удалять только всю запись целиком, а не отдельные поля внутри нее.

Для создания запроса на удаление нужно выполнить следующие действия:

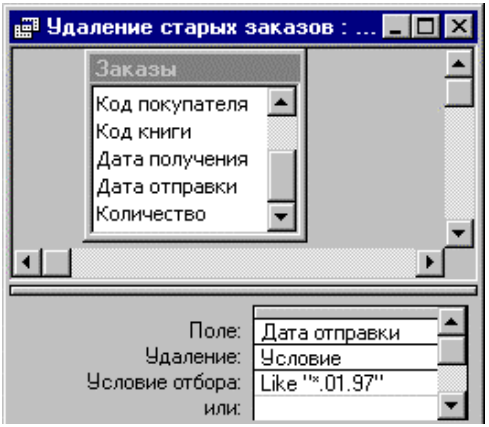


Рис. 5.54. Удаление заказов за январь 97 года

1. Включить в окно конструктора таблицу, из которой предстоит удалить записи. Затем **выбрать пункт меню Запрос,** а потом **Удаление.**
2. Ввести условия отбора удаляемых записей в строку **Условие отбора**. Эти условия появятся под словом **Условие** строки **Удаление**.
3. Для просмотра удаляемых записей нужно щелкнуть по кнопке **Вид** на панели инструментов. Затем следует вернуться в режим конструктора, повторно нажав эту кнопку.
4. Для удаления записей нужно нажать кнопку **Запуск** на панели инструментов. Access сообщит число удаляемых записей и попросит подтвердить выполнение операции. После подтверждения записи будут удалены из таблицы.

Пример 5.33. Нужно удалить из таблицы **Заказы** сведения о заказах, отправленных в январе 1997 года.

Для создания запроса нужно включить в окно конструктора таблицу **Заказы** и выбрать в **меню Запрос** тип запроса **Удаление.** Затем следует добавить в бланк запроса поле **Дата отправки** и ввести условие отбора: **.01.97* (см. рис. 5.54).

Чтобы просмотреть удаляемые записи, щелкнем по кнопке **Вид**. На экране появятся четыре записи (см. рис. 5.55). После выполнения запроса информация об этих заказах будет удалена из БД.

При удалении записей с помощью запроса на удаление отменить эту операцию нельзя. Поэтому рекомендуется перед выполнением запроса сделать резервную копию таблицы. Тогда, если в результате выполнения запроса будут удалены не те записи, то их можно будет восстановить из резервной копии.

Дата отправки
24.01.97
25.01.97
26.01.97
29.01.97

Рис. 5.55. Удаляемые записи

При выполнении запроса на удаление могут быть удалены некоторые записи из таблиц, не включенных в запрос. Это произойдет, если в запрос включена таблица, находящаяся на стороне «один» отношения «один-ко-многим», и для этого отношения установлен режим каскадного удаления. При удалении записей из этой таблицы будут также удаляться связанные с ними записи из подчиненных таблиц.

Например, если в связи «один-ко-многим» между таблицами **Студенты** и **Сессия** установлен режим каскадного удаления, то при удалении из таблицы **Студенты** сведений об исключенном студенте с помощью соответствующего запроса Access автоматически удалит из таблицы **Сессия** информацию о его оценках.

Если режим каскадного удаления не включен, то прежде чем удалить записи из главной таблицы, нужно выполнить запрос на удаление связанных с ними записей из подчиненных таблиц.

Пример 5.34. Нужно удалить из таблицы **Покупатели** записи, относящиеся к покупателям из Омска. Известно, что эта таблица является главной в связи «один-ко-многим» между таблицами **Покупатели и Заказы,** для которой установлен режим обеспечения целостности данных, но не включен режим каскадного удаления.

Для выполнения этой операции с помощью запроса на удаление нужно включить в окно конструктора обе связанные таблицы и указать тип запроса — **Удаление**. Затем следует добавить в бланк запроса поле **Город** из таблицы **Покупатели** и ввести в него условие отбора: *омск.* После этого

нужно перенести в бланк запроса звездочку (*) из таблицы **Заказы**. В ячейке **Удаление** в этом поле появляется значение **Из** (см. рис. 5.56).

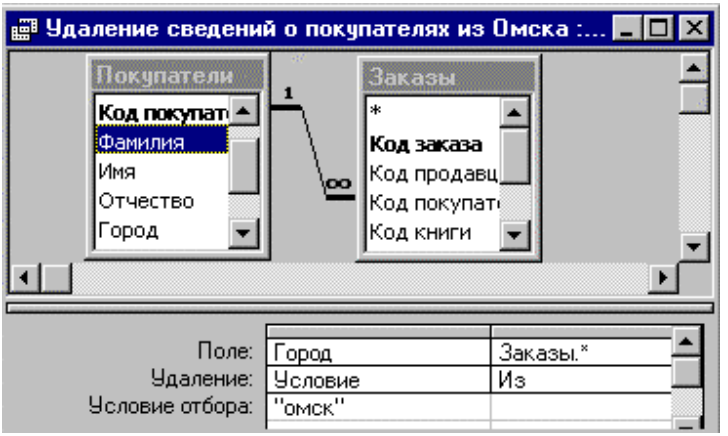


Рис. 5.56. Удаление заказов из Омска

Если выполнить этот запрос, то из подчиненной таблицы **Заказы** будут удалены записи, относящиеся к омским покупателям. Для их предварительного просмотра достаточно щелкнуть по кнопке **Вид**.

После удаления записей о заказах омичей нужно удалить саму таблицу **Заказы** из окна конструктора (см. рис. 5.57). Для этого следует выделить щелчком мыши список ее полей и нажать клавишу **Delete**. Затем нужно еще раз выполнить запрос. Теперь будут удалены записи из главной таблицы **Покупатели**, указанные в условии отбора.

5.7.3. Запрос на создание таблицы

Запрос на создание таблицы позволяет создать таблицу на основе данных, содержащихся в других таблицах БД. Он часто используется для создания таблицы, предназначенной для экспорта в другую БД Access или архивной копии таблицы. Например, можно создать таблицу со сведениями обо всех старых заказах на книги прежде чем удалить их из таблицы **Заказы**.

Для создания запроса этого типа нужно выполнить следующие действия:

1. Включить в окно конструктора таблицы или запросы, содержащие записи, которые следует поместить в новую таблицу.

2. Поместить в бланк запроса поля, которые должна содержать новая таблица. Добавить к ним поля, для которых задаются условия отбора. В строке **Условие отбора** ввести условия выбора нужных записей.

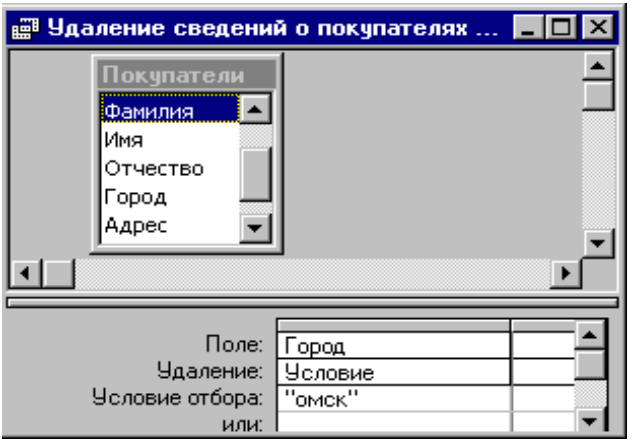


Рис. 5.57. Удаление сведений о покупателях из Омска

3. Выбрать пункт меню **Запрос,** а затем **Создание таблицы**. На экране появится диалоговое **Создание таблицы (**см. рис. 5.58).

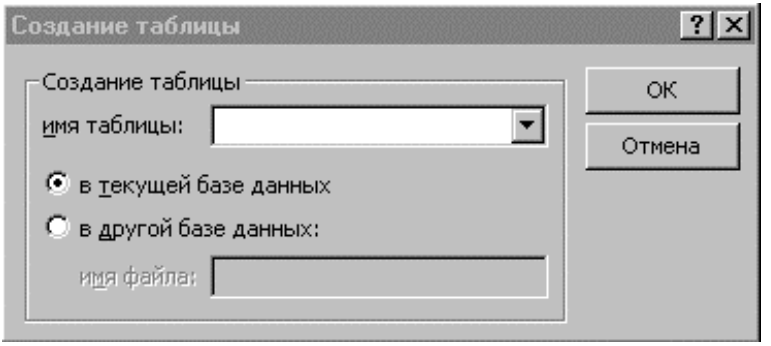


*Рис. 5.58. Диалоговое окно **Создание таблицы***

4. В поле **Имя таблицы** ввести имя создаваемой таблицы и затем щелкнуть по кнопке **OK**. Если таблица находится в другой БД, то нужно перед нажатием кнопки выбрать параметр **В другой базе данных** и ввести полное имя этой БД.

5. Для предварительного просмотра новой таблицы нужно щелкнуть по кнопке **Вид** на панели инструментов. Затем следует вернуться в режим конструктора, повторно щелкнув по этой кнопке.
6. Для создания таблицы нужно щелкнуть по кнопке **Запуск** на панели инструментов. Если в БД уже существует таблица с таким именем, то Access попросит разрешения удалить ее перед началом операции. Затем он сообщит число записей и попросит подтвердить выполнение операции. После подтверждения будет создана таблица.

Пример 5.35. Построим запрос, создающий таблицу, в которой будет храниться информация о «старых» заказах.

Для этого добавим в окно конструктора таблицу **Заказы** и перенесем в бланк запроса звездочку (*) и поле **Дата отправки**. Введем в это поле условие в виде параметра (см. рис. 5.59) и установим для него тип *Дата/время*. Затем укажем тип запроса **Создание таблицы** и в окне **Создание таблицы** введем имя создаваемой таблицы: **Старые заказы.**

Поле:	Заказы.*	Дата отправки
Сортировка:		
Вывод на экран:	☑	☐
Условие отбора:		<[конечная дата ?]

Рис. 5.59. Отбор «старых» заказов

Выполним созданный запрос. Access попросит ввести значение параметра и после ввода даты создаст таблицу **Старые заказы**, содержащую заказы, дата отправки которых не превосходит введенной даты. Ее структура идентична структуре таблицы **Заказы**. При повторном выполнении запроса старая таблица будет удалена и ее заменит новая таблица с тем же именем.

5.7.4. Запрос на обновление

Этот запрос позволяет внести изменения в группу записей одной или нескольких таблиц. Для создания запроса на обновление нужно выполнить следующие действия:

1. Включить в окно конструктора таблицу или запрос, содержащие записи, которые следует обновить.
2. Выбрать пункт меню **Запрос,** а затем **Обновление.** В бланке запроса появится дополнительная строка **Обновление**.
3. Поместить в бланк запроса обновляемые поля и добавить к ним поля, для которых задаются условия отбора.

4. В строке **Условие отбора** ввести условия выбора нужных записей, а в строке **Обновление** ввести выражения, которые должны быть использованы для изменения полей.
5. Для предварительного просмотра обновляемых записей щелкнуть по кнопке **Вид**  на панели инструментов. Выводимый список будет содержать старые значения. Затем следует вернуться в режим конструктора, повторно щелкнув по этой кнопке.
6. Для обновления записей щелкнуть по кнопке **Запуск**  на панели инструментов. Access сообщит число обновляемых записей и попросит подтвердить выполнение операции. После подтверждения в записи будут внесены новые значения.

Пример 5.36. Построим запрос, вносящий изменения в цены книг издательства «Питер». Предполагается, что цена всех книг этого издательства должна увеличиться на 20%.

Поле:	Цена	Издательство
Имя таблицы:	Книги	Книги
Обновление:	[Цена]*1,2	
Условие отбора:		"Питер"

Рис. 5.60. Изменение цен на книги

Для создания запроса добавим в окно конструктора таблицу **Книги** и выберем пункт меню **Запрос,** а затем пункт **Обновление.** Поместим в бланк запроса обновляемое поле **Цена** и поле **Издательство**, в котором зададим условие отбора *Питер* (см. рис. 5.60). После выполнения запроса Access обновит значения в шести записях.

5.7.5. Ошибки при выполнении запросов на изменение

Перед внесением изменений в БД Access анализирует запрос и изменяемые данные. При обнаружении ошибки он выдает соответствующее сообщение и предоставляет пользователю возможность отменить операцию. Access различает следующие типы ошибок:

1. *Ошибки преобразования типа* возникают при добавлении данных в существующую таблицу, если тип данных полей-получателей не совпадает с типом данных полей-источников и добавляемые данные не могут быть преобразованы к типу данных полей-получателей.
2. *Нарушение уникальности ключа.* Такие ошибки возникают при попытке добавления или изменения записи в таблице, в результате чего появляется дубликат первичного ключа или уникального индекса.

Access не будет добавлять или изменять такие записи. Поэтому перед выполнением такой операции нужно изменить значения первичного ключа в исходной таблице.

3. *Нарушение блокировки* возникает при работе с таблицей, к которой имеют доступ другие пользователи в сети. Access не может выполнить операцию обновления записи, если в нее в это же время вносит изменения другой пользователь.
4. *Нарушение условий на значение.* Если добавляемая или обновляемая запись не удовлетворяет условию на значение для одного из полей или таблицы, то Access не произведет операцию добавления или обновления для этой записи.

Глава 6. Формы

6.1. Общие сведения

Формы предназначены для просмотра, ввода, редактирования и управления данными. При использовании формы Access предоставляет пользователю существенно больше возможностей для работы с информацией по сравнению с ее стандартным представлением в режиме таблицы. Удобство применения форм для работы с данными заключается в следующем:

- Формы обычно позволяют отобразить на экране всю запись целиком, причем порядок следования полей может быть изменен, а часть из них вообще не включена в форму.
- В формах кроме редактируемых полей, содержащих данные из таблиц БД, можно размещать и нередактируемые (вычисляемые) поля.
- В формы можно добавлять комментарии, рисунки, графики, изменять их внешний вид, подбирая подходящие шрифты, фон и стиль оформления. Это позволяет в случае необходимости разработать форму, имеющую большое сходство с бумажной формой.
- Формы позволяют существенно упростить, а зачастую полностью автоматизировать ввод новых данных.
- В формах можно размещать различные кнопки (*кнопочные формы*), нажатие на которые приводит к открытию других форм, выполнению запросов, печати отчетов и т. д.
- Формы могут использоваться в качестве *диалоговых окон* и содержать собственное меню пользователя.

Для разработки форм в Access существует специальный инструментарий. Прежде чем приступать к описанию процедур создания форм, обсудим несколько общих моментов.

Основным *источником данных* для формы являются таблицы и запросы. Информация может быть также включена в форму в результате импорта различных объектов либо проведения вычислений. Если через форму осуществляется обновление данных, то они обновляются и в источнике.

Построение формы — итеративный процесс. После создания макета формы нужно просмотреть его, чтобы убедиться в его пригодности. Если необходимо внести изменения, можно вернуться к корректировке макета.

Для создания макета формы нужно в окне БД перейти на вкладку **Формы** и нажать клавишу **Создать**. После выбора источника данных для

формы, а также способа ее создания, при переходе к следующему этапу автоматически обновляются панели инструментов. Появляется панель инструментов **Конструктор форм**.

Создание, корректировка и просмотр формы осуществляется в различных режимах:

- в *режиме конструктора* форма создается и корректируется;
- *в режиме формы* или *режиме таблицы* форма используется для работы с данными;
- в *режиме предварительного просмотра* форма просматривается перед печатью.

Все эти режимы выбираются с помощью кнопки **Вид**  панели инструментов **Конструктор форм** или через меню **Вид**.

6.1.1. Структура формы

Форма состоит из нескольких *разделов* (см. рис. 6.1), причем обязательным является только один из них — *область данных*. Кроме этого раздела в форме могут присутствовать следующие разделы: *заголовок* и *примечание формы,* а также *верхний* и *нижний колонтитулы*. Информация в них вводится разработчиком формы. Эти разделы имеют следующие назначения:

Рис. 6.1. Структура формы

Заголовок формы может содержать название формы, инструкции по работе с ней, а также другую информацию (текущую дату, время и т.д.). В режиме формы он находится в верхней части экрана, а при печати — в верхней части первой страницы. В режиме таблицы этот раздел не отображается.

Верхний колонтитул может содержать заголовки и любые другие сведения, помещаемые в верхней части каждой страницы формы при печати.

Нижний колонтитул также используется при печати и может содержать номер страницы, дату и другую информацию.

Примечание формы может содержать инструкции по использованию формы, кнопки и свободные элементы управления для ввода или вывода данных. В режиме формы область примечаний находится в нижней части экрана, а при печати печатается на последней странице.

При создании формы с помощью конструктора она содержит лишь один раздел — область данных. Остальные разделы можно добавить, используя соответствующие пункты контекстного меню формы или меню Вид.

Форма может включать *подчиненные формы,* внутри которых могут отображаться три раздела: заголовок, область данных и примечание.

Как и все объекты БД, форма и ее разделы имеют *свойства*. В свойствах задаются режимы работы с формой, вид формы, источник данных, использование фильтра. Свойства разделов используются при выводе их на экран или печать.

6.1.2. Виды форм

Различные виды форм создаются обычно мастером форм или пользователем с помощью конструктора. Кроме того, существуют мастер *диаграмм* и мастер *сводная таблица*.

Возможно создание трех видов автоформ: *в один столбец*, *табличной* и *ленточной формы*. При создании автоформы все поля таблицы или запроса выводятся в форму автоматически.

Форма в один столбец отображает поля, расположенные в один столбец. На экране отображается одна запись.

Табличная форма отображает данные в виде нескольких строк и столбцов. Одновременно отображается несколько записей. По внешнему виду и способам перемещения по записям она ничем не отличается от обычной таблицы Access.

Ленточная форма похожа на табличную форму и отличается лишь внешним оформлением.

В отличие от автоформ при использовании мастера форм поля в форму можно выбирать. Мастер может создать кроме вышеперечисленных видов форм *составную форму*. Она состоит из *главной* формы и *подчиненной*. В этих формах отображаются данные из разных таблиц, причем эти таблицы чаще всего связаны отношением «один-ко-многим». Данные главной формы отображаются в один столбец, а подчиненная форма обычно имеет табличный формат.

Диаграмма представляет данные в графическом виде и создается мастером диаграмм с использованием приложения Microsoft Graph.

Форма *Сводная таблица* создается на основании данных из сводных таблиц Excel.

Пользователь может создавать формы так, как он захочет, и включать в них любые данные. Он может начать с пустой формы (с «нуля») или воспользоваться мастером форм, а затем в режиме конструктора внести в форму любые изменения.

6.1.3. Содержимое формы

Вся информация в форме размещается внутри *элементов управления (ЭУ)*. Это графические объекты, предназначенные для изображения данных, выполнения вычислительных операций, вывода на экран вспомогательной информации, оформительских эффектов (рамок, линий и др.). Характер ЭУ зависит от источника размещаемой в нем информации.

Присоединенные ЭУ связаны с полями источника данных (таблицы или запроса) для формы. Они используются для просмотра, ввода или обновления значений из полей таблиц БД.

Для *свободного ЭУ* источник данных не определен. ЭУ этого типа обычно используются для вывода на экран надписей, линий и рисунков.

Для *вычисляемого ЭУ* источником данных служит выражение, в котором могут быть использованы значения из полей источника данных для формы, а также значения, содержащиеся в других ЭУ формы. Эти ЭУ обычно используются для выполнения вычислений и вывода на экран их результатов. Результат вычислений не сохраняется в БД.

6.2. Создание формы

При нажатии кнопки **Создать** во вкладке **Формы** окна БД появляется диалоговое окно **Новая форма**. В процессе диалога нужно выбрать один из способов создания формы и таблицу или запрос, которые будут служить источником данных для формы (см. рис. 6.2).

Проще всего создать форму с помощью одной из автоформ. Нужно выбрать из списка источник данных (таблицу/запрос) и одну из предложенных автоформ. Автоформу можно также создать, находясь в окне БД во вкладке **Таблицы** или **Запросы**. Для этого нужно выбрать щелчком мыши таблицу/запрос и щелкнуть по кнопке **Автоформа**  на панели инструментов.

6.2.1. Использование *Мастера форм*

Если выбран Мастер форм и указан источник данных для новой формы (таблица/запрос), то все шаги по ее созданию будут определяться мастером с помощью последовательности диалоговых окон. Сначала рассмотрим случай, когда создаваемая форма основана на одной таблице/запросе. Создание с помощью мастера составной формы, использующей данные из нескольких таблиц/запросов, будет описано позднее (см. п. 6.6.1).

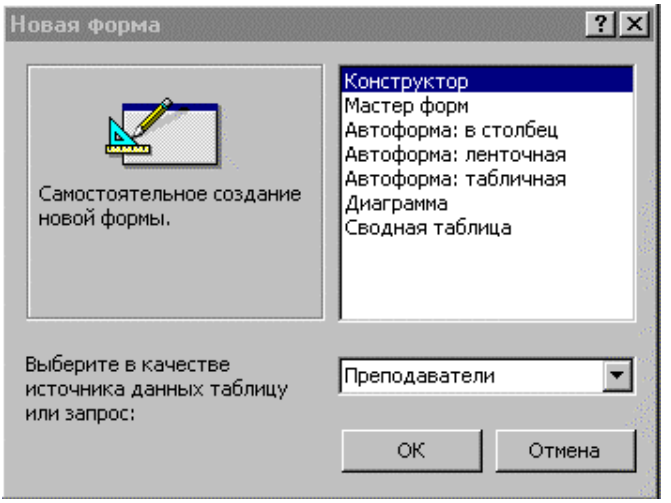


*Рис. 6.2. Окно **Новая форма***

1. В первом окне мастера форм нужно выбрать поля таблицы/запроса, которые будут присутствовать в форме, и переместить их из списка Доступные поля в список Выбранные поля. Для этого используются кнопки >>, >, <, <<. Кнопки >> и << перемещают все поля из одного списка в другой, а > и < — только одно поле. Поле можно также переместить из одного списка в другой с помощью двойного щелчка мыши.
2. В следующих двух окнах диалога задается внешний вид и стиль оформления формы. После выбора варианта в окне мастера появляется соответствующий образец.
3. Последнее диалоговое окно дает возможность изменить заданное по умолчанию имя формы. Закончить диалог можно либо просмотром созданной формы, либо переходом в режим конструктора для внесения в нее изменений.

6.2.2. Использование Конструктора форм

При создании формы с помощью режима **Конструктор** пользователю также предлагается указать имя ее источника данных. Если форма использует данные из нескольких таблиц, то обычно предварительно создается и сохраняется запрос, содержащий нужные поля из этих таблиц. Затем он указывается в качестве источника данных создаваемой формы. Конечно, если форма не предназначена для вывода информации, например является кнопочной формой, указывать источник ее данных не нужно.

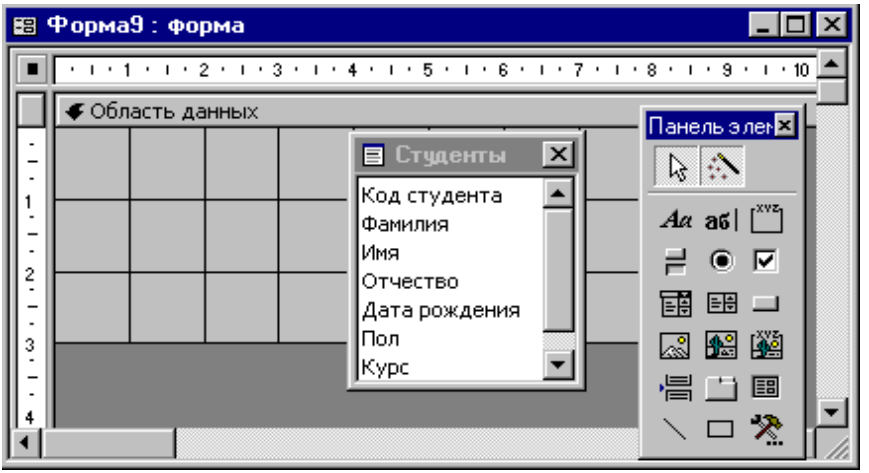


Рис. 6.3. Бланк конструктора с панелью элементов и списком полей

После выбора этого режима на экране появляется бланк конструктора формы (см. рис. 6.3). Кроме бланка конструктора формы появляются окно **Список полей** и **Панель элементов** с кнопками для ввода различных ЭУ. Если панель элементов отсутствует, то ее можно вызвать на экран с помощью пункта **Панель элементов** меню **Вид**, либо нажать кнопку **Панель элементов** на панели инструментов **Конструктор форм**.

Состав и размещение ЭУ пользователь определяет самостоятельно. На бланке конструктора формы имеется сетка для удобного размещения ЭУ. Размещению ЭУ помогают также горизонтальная и вертикальная линейки. Если сетка или линейки отсутствуют, то их также можно вызвать на экран с помощью соответствующих пунктов меню **Вид**. Для размещения ЭУ в форме нужно нажать соответствующую кнопку на панели элементов, а затем щелкнуть мышью в том месте области данных, где предполагается поместить левый верхний угол ЭУ. Ниже приводится описание панели элементов.

Перечень кнопок панели элементов

Кнопка *Выбор объектов* — используется для выделения одного или нескольких ЭУ. По умолчанию эта кнопка нажата. Для отключения этой кнопки следует нажать ее еще раз.

Кнопка *Мастера элементов* — включает и отключает мастеров по созданию ЭУ. Мастера помогают создавать такие сложные ЭУ, как список, поле со списком, группа переключателей, кнопка, диаграмма, подчиненная форма или подчиненный отчет. Для создания этих элементов управления с помощью мастеров кнопка **Мастера элементов** должна быть нажата.

Кнопка *Поле* — создает ЭУ **Поле,** используемый для отображения, ввода или изменения данных в источнике записей формы, а также для вывода результатов вычислений.

Кнопка *Надпись* — используется для создания ЭУ **Надпись**, содержащего неизменяемый текст, например, заголовок, подпись поля или пояснение. По умолчанию многие ЭУ, в частности **Поле**, имеют связанные с ними надписи.

Кнопка *Группа переключателей* — используется для размещения группы ЭУ, состоящей из флажков, переключателей или выключателей, представляющих набор альтернативных значений.

Кнопка *Выключатель* — используется для создания ЭУ **Выключатель**, связанного с логическим полем. Если щелкнуть по выключателю, то он будет изображен нажатым и его значение станет равно *Вкл (Истина***)**. При повторном щелчке его значение станет равно *Выкл (Ложь)*. Выключатель можно включить в группу переключателей. Если щелкнуть по выключателю, входящему в группу переключателей, переводя его в состояние *Вкл*, то любой ранее нажатый элемент группы (выключатель, переключатель или флажок) будет переключен в состояние *Выкл*.

Кнопка *Переключатель* — используется для создания ЭУ **Переключатель,** называемого также радиокнопкой. Если щелкнуть по переключателю, то в центре его появится темный кружок, а его значение станет равно *Вкл (Истина***)**. При повторном щелчке его значение станет равно *Выкл (Ложь)*. Если щелкнуть по переключателю, входящему в группу переключателей, переводя его в состояние *Вкл*, то любой ранее нажатый элемент группы (выключатель, переключатель или флажок) будет переключен в состояние *Выкл*.

Кнопка *Флажок* — используется для создания ЭУ **Флажок**, связанного с логическим полем. Если щелкнуть по флажку, то в центре его появится «галочка», а его значение станет равно *Вкл (Истина*). При повторном щелчке его значение станет равно *Выкл (Ложь*), а «галочка» исчезнет. Если щелкнуть по флажку, входящему в группу переключателей, переводя его в состояние *Вкл*, то любой ранее нажатый элемент группы (выключатель, переключатель или флажок) будет переключен в состояние *Выкл*.

Кнопка *Поле со списком* — используется для создания ЭУ **Поле со списком**, объединяющего два ЭУ: поле и раскрывающийся список. Для ввода значения в поле таблицы, связанное с этим ЭУ, можно ввести значение в поле или выбрать значение в списке.

Кнопка *Список* — создает ЭУ **Список,** содержащий список потенциальных значений. В режиме формы выбранное в списке значение можно ввести в новую запись или использовать для изменения значения в существующей записи.

Кнопка *Кнопка* — создает ЭУ **Кнопка**, используемый для выполнения таких действий, как поиск или печать записи, работа с формой или отчетом, запуск запроса или макроса.

Кнопка *Рисунок* — используется для включения в форму или отчет неизменяемого рисунка.

Кнопка *Свободная рамка объекта* — используется для отображения в форме или отчете свободного объекта OLE, например, фрагмента электронной таблицы Excel или документа Word. Этот объект остается неизменным при переходе от записи к записи.

Кнопка *Присоединенная рамка объекта* — создает рамки для отображения в форме или отчете объектов OLE, таких как набор рисунков. Этот элемент управления предназначен для объектов, сохраненных в поле источника записей формы. При переходе от записи к записи в форме выводятся разные объекты.

Кнопка *Конец страницы* — создает ЭУ **Конец страницы**, указывающий принтеру начало новой страницы в печатной форме или отчете. Этот ЭУ не появляется в форме в режиме формы.

Кнопка *Набор вкладок* — вставляет ЭУ **Набор вкладок** для создания многостраничной формы. Этот ЭУ выглядит как набор страниц, которые вы видели в окнах свойств различных ЭУ. Страницы ЭУ **Набор вкладок** могут содержать присоединенные или свободные ЭУ, включая ЭУ **Подчиненная форма** или **Подчиненный отчет.**

Кнопка *Подчиненная форма/отчет* — добавляет в основную форму или основной отчет соответственно подчиненную форму или подчиненный отчет.

Кнопка *Прямоугольник* — создает прямоугольник, который можно перемещать и размеры которого можно изменять. Цвет рамки и цвет фона определяются с помощью палитры.

Кнопка *Линия* — создает прямую линию, которую можно перемещать и размеры которой можно изменять. Цвет и толщину линии можно изменить с помощью кнопок панели инструментов **Панель форматирования** или окна свойств.

Кнопка *Дополнительные элементы* — выбор этой кнопки открывает список дополнительных ЭУ ActiveX, которые можно использовать в формах. ЭУ выбираются из списка **Дополнительные элементы**, который является частью Access 97. Они поддерживаются библиотеками OCX из Office 97 и Visual Basic и различными библиотеками независимых поставщиков.

Фиксация кнопки на панели элементов

Если нужно создать несколько одинаковых ЭУ, то имеет смысл зафиксировать соответствующую кнопку на панели элементов. Для этого нужно по ней дважды щелкнуть. Когда кнопка зафиксирована, нет необходимости каждый раз нажимать ее при создании нового ЭУ. Для отмены фиксации кнопки следует нажать клавишу **Esc**.

Изменение стандартных свойств ЭУ

Каждый ЭУ имеет набор *стандартных свойств*, которые он получает по умолчанию при его создании. Эти свойства определяют внешний вид и характеристики ЭУ. Так, например, ЭУ **Поле** создается по умолчанию с присоединенной подписью (см. рис. 6.4). Для того чтобы изменить стандартные свойства ЭУ, следует щелкнуть по соответствующей кнопке на панели элементов, затем по кнопке **Свойства**  на панели инструментов и в появившемся окне задать нужные значения стандартных свойств.

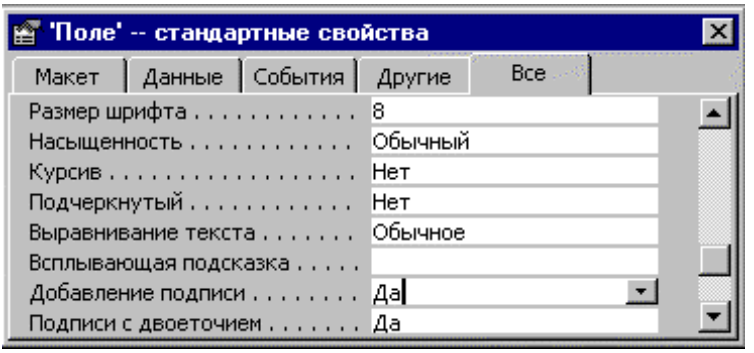


Рис. 6.4. Стандартные свойства ЭУ ***Поле***

После изменения стандартных свойств ЭУ добавляемые в форму ЭУ этого типа по умолчанию получат новые значения свойств.

В качестве стандартных свойств ЭУ данного типа можно указать свойства существующего ЭУ. Для этого нужно щелкнуть по ЭУ со свойствами, которые должны стать стандартными, и выбрать в меню **Формат** команду **Задать стандартные свойства**. Свойства выбранного ЭУ станут стандартными свойствами для элементов этого типа.

Пример 6.1. Нужно создать форму, содержащую несколько одинаковых полей без подписей шириной 2 см.

Чтобы поля в форму добавлялись с такими характеристиками, следует вызвать окно стандартных свойств ЭУ **Поле** (см. рис. 6.4) и задать для свойства **Добавление подписи** значение *Нет*, а для свойства **Ширина** — значение *2*.

> *Изменения в стандартных свойствах ЭУ формы не влияют на стандартные свойства ЭУ, создаваемые в других формах. Чтобы использовать измененные стандартные свойства ЭУ в других формах, нужно сначала сохранить текущую форму. Затем следует выбрать в меню* ***Сервис*** *команду* ***Параметры*** *и во вкладке* ***Формы/отчеты*** *указать в поле* ***Шаблон формы*** *ее имя в качестве шаблона.*

Настройка внешнего вида формы

При создании формы у пользователя может возникнуть желание изменить положение и размеры уже размещенных в форме ЭУ. Перед выполнением этих операций ЭУ нужно выделить. Чтобы выделить ЭУ, укажите на него и щелкните кнопкой мыши. Вокруг выделенного ЭУ

появляются *маркеры размеров* и *маркер перемещения* (см. рис. 6.5). Если этот ЭУ — поле, то выделяется и поле и его подпись.

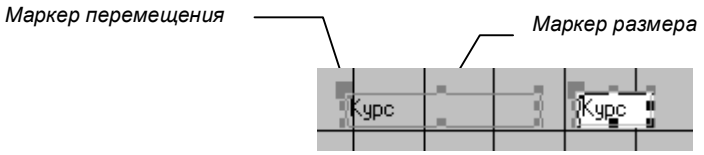


Рис. 6.5. Маркеры размера и перемещения на выделенном ЭУ

Можно проделывать оформительские операции сразу с группой ЭУ. Группу ЭУ можно выделять несколькими способами.

Если нужно выделить ЭУ, расположенные рядом, установите указатель мыши в верхнем левом углу области расположения ЭУ и при нажатой кнопке мыши переместите его в нижний правый угол области.

Если нужно выделить ЭУ, расположенные не рядом или поверх других ЭУ, нажмите клавишу **Shift** и, не отпуская ее, по очереди выделите нужные ЭУ.

Можно выделять ЭУ, занимающие одинаковые горизонтальные или вертикальные позиции, с помощью линеек бланка формы. Установите указатель на нужную позицию горизонтальной или вертикальной линейки и нажмите кнопку мыши. Все ЭУ, расположенные на этом уровне, будут выделены. Если нужно увеличить область выделения, нужно, не отпуская кнопку мыши, выделить необходимую область на горизонтальной или вертикальной линейке.

Выделенные ЭУ можно удалить, переместить, изменить размеры и выровнять их положение.

Удалить выделенные ЭУ можно, нажав клавишу **Del.**

Для *перемещения* ЭУ надо поместить указатель на рамку выделенных ЭУ вне маркеров. Он должен принять вид раскрытой руки. При нажатой кнопке мыши переместить ЭУ на нужное место. Если нужно переместить один из выделенных ЭУ, нужно указатель установить на маркер перемещения этого ЭУ. Он примет вид руки с указательным пальцем. При нажатой кнопке мыши перетащить ЭУ на нужное место. Для повышения точности расположения ЭУ используйте линейки.

Для *изменения размеров* ЭУ нужно установить указатель на маркер размера одного из выделенных ЭУ и изменить его размеры. Все выделенные ЭУ изменят свои размеры пропорционально первому. Если при

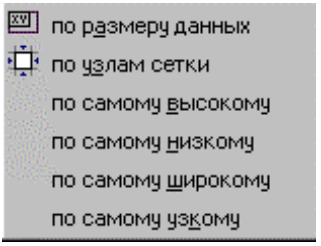


Рис. 6.6. Изменение размера

этом в пункте **Размер** меню **Формат** задано значение **По узлам сетки**, размеры будут изменяться так, чтобы их углы совпадали с углами сетки. В этом же пункте **Размер** можно задать другой вариант изменения размеров в выделенной группе ЭУ (см. рис. 6.6).

Выровнять положение ЭУ на бланке формы можно их перемещением, но удобнее воспользоваться пунктом Выровнять меню Формат. Предусмотрено выравнивание по самому правому, левому, нижнему или верхнему ЭУ либо по ячейкам сетки. С помощью пункта **Интервал по вертикали** (**горизонтали**) можно быстро установить равные интервалы по вертикали и горизонтали между выделенными ЭУ.

Для оформительских эффектов используются кнопки панели инструментов **Формат** или аналогичные пункты меню **Формат**. Это кнопки **Цвет заливки/фона**, **Цвет текста**, **Оформление**. С их помощью можно менять цвет ЭУ, цвет текста, цвет границ, различные эффекты изображения (вдавленное, выпуклое и т. д.). Можно также применять автоформаты. Они облегчают создание оформительских эффектов. Автоформаты вызываются кнопкой **Автоформат** на панели инструментов Конструктор формы.

> *Можно скопировать характеристики форматирования из одного ЭУ в другой. Для этого следует щелкнуть сначала по ЭУ, формат которого копируется, затем по кнопке* ***Формат по образцу*** *на панели конструктора, и в заключение — по ЭУ, на который нужно перенести скопированный формат. Если зафиксировать двойным щелчком кнопку* ***Формат по образцу****, то скопированный формат можно использовать для форматирования нескольких ЭУ.*

6.3. Создание основных ЭУ

Как уже говорилось выше, ЭУ могут быть присоединенными, свободными или вычисляемыми. Каждому типу ЭУ соответствует свой способ его создания.

Для создания присоединенного ЭУ проще всего использовать *список полей* источника данных, который обычно автоматически появляется при создании формы. Если на экране его нет, то его можно открыть кнопкой **Список полей** или командой **Список полей** меню Вид.

Чтобы создать присоединенный ЭУ, нужно выбрать связанное с ним поле в списке полей и, держа нажатой левую кнопку мыши, перетащить его

в нужное место формы. Access создаст ЭУ, соответствующий выбранному полю, и задаст значения его свойств, отвечающие типу данных и свойствам поля.

Можно перенести, предварительно выделив, сразу несколько полей из списка. В этом случае в форме появится группа ЭУ, присоединенных к этим полям. При желании тип созданного ЭУ можно изменить.

Свободные ЭУ создаются с помощью панели элементов. Для создания элемента нужно нажать соответствующую кнопку на панели, а затем щелкнуть мышью в том месте области данных, где должен находиться левый верхний угол создаваемого ЭУ. Будет создан свободный ЭУ стандартного размера.

Обычно вычисляемым ЭУ является поле. Описание правил создания этого ЭУ, а также более детальная информация о создании наиболее часто используемых ЭУ приводятся ниже.

6.3.1. Создание надписи

Для создания надписи, не связанной ни с каким ЭУ, нужно нажать кнопку Надпись  , указать место размещения надписи, а затем ввести текст. Чтобы перейти в надписи на новую строку, надо нажать клавиши **Ctrl+Enter**. Для форматирования содержимого надписи можно использовать кнопки форматов текста.

При изменении формата текста надпись может перестать помещаться в отведенной рамке. В этом случае нужно привести в соответствие надпись и размер рамки. Для этого достаточно выделить надпись и выполнить команду **По размеру данных** в пункте **Размер** меню **Формат**.

Чтобы создать надпись, присоединенную к ЭУ (такая надпись называется *подписью*), достаточно создать этот ЭУ. Access автоматически присоединит к нему подпись. После этого можно изменить и отформатировать содержащийся в ней текст.

6.3.2. Создание присоединенного поля

Для создания в форме *присоединенного поля* (т.е. получающего информацию из поля таблицы/запроса) проще всего выбрать это поле в списке полей и перетащить его в нужное место формы. В форме появятся два ЭУ: *поле* и его *подпись*. Подпись содержит имя (или подпись, если она существует) выбранного поля таблицы/запроса. Само поле в форме наследует имя и свойства связанного с ним поля. Это имя можно использовать для ссылки на текущее значение поля в выражении (см. пример 6.2).

Поле и его подпись связаны между собой. Если попытаться выделить (щелкнуть мышкой) один из них, выделяются оба, причем маркеры размера

и кнопка перемещения находятся на том элементе, на котором щелкнули мышью. При переходе в режим формы подпись остается без изменений, а в окне поля появляется информация из связанного с ним поля источника данных.

Можно перенести в форму сразу группу полей, предварительно выделив их, либо все поля (выделяются двойным щелчком мыши на заголовке **Списка полей**).

Для изменения свойств поля можно вызвать команду **Свойства** из контекстного меню, либо из меню **Вид**. Свойства также вызываются кнопкой  или двойным щелчком мыши внутри ЭУ.

6.3.3. Создание вычисляемого поля

Для создания ЭУ *вычисляемое поле* нужно нажать кнопку **Поле**  панели элементов и указать место размещения нового элемента. Появится свободный ЭУ **Поле** и связанная с ним подпись. Затем нужно создать выражение, являющееся источником данных для этого поля.

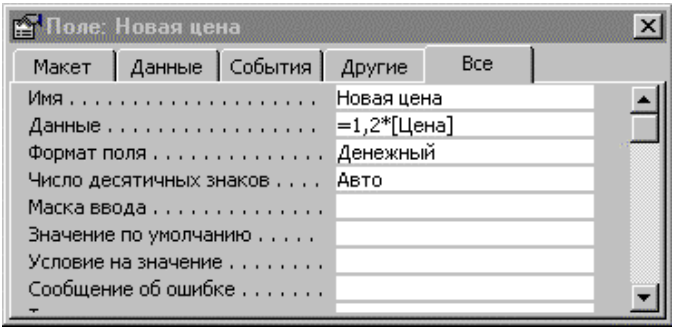


Рис. 6.7. Окно свойств ЭУ ***Поле***

Если выражение достаточно простое, то его можно ввести непосредственно в поле. Для этого нужно щелкнуть мышкой внутри поля и затем ввести формулу вычисления значения поля, содержащую знак равенства (=) и вычисляемое выражение. У вычисляемого поля следует изменить надпись. Можно также ввести формулу вычисления значения в ячейку свойства **Данные,** открыв окно свойств поля (см. рис. 6.7).

Если выражение имеет сложную структуру, то лучше использовать построитель выражений. Для этого нужно щелчком мыши выделить поле, для которого создается выражение, а затем вызвать окно его свойств. После щелчка по ячейке свойства **Данные** появится кнопка  , щелчок по которой вызывает построитель выражений. Созданное с его помощью выражение появится в окне свойств поля в качестве значения свойства **Данные**. Затем

можно настроить формат вывода значения поля на экран с помощью свойства **Формат поля** и, если нужно, свойства **Число десятичных знаков**.

Пример 6.2. В связи с изменением цен на книги (подорожанием на 20%) в БД **Книги** нужно создать форму, в которой для каждой книги выводятся сведения об ее старой и новой цене, а также о величине изменения цены.

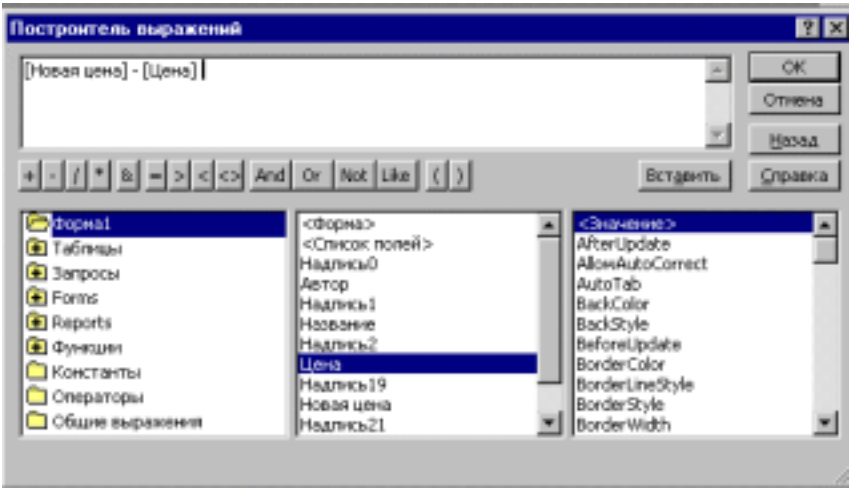


Рис. 6.8. Создание вычисляемого поля с помощью построителя

Для создания формы выберем в окне **Новая форма** пункт **Конструктор** и укажем в качестве источника данных таблицу **Книги**. Выделим в списке полей поля **Автор**, **Название**, **Цена** и «перетащим» их в область данных формы. Там появятся три ЭУ **поле**, присоединенные к соответствующим полям таблицы и наследующие их имена, а также ряд свойств. Слева от этих полей Access размещает связанные с ними подписи, содержащие названия полей.

Чтобы добавить в форму сведения о новой цене и величине изменения цены, нужно разместить в ней два ЭУ **вычисляемое поле**. Для их создания зафиксируем двойным щелчком кнопку **Поле** на панели элементов. Затем, щелкая мышью под уже размещенными в форме полями, создадим два новых свободных поля с подписями. Для отмены фиксации кнопки нажмем клавишу **Esc**.

Далее откроем окно свойств первого из этих полей и введем в ячейку **Данные** формулу: *= 1,2*[Цена]*. Дадим этому полю имя *Новая цена* и установим путем выбора из списка значение свойства **Формат поля:**

Денежный. Затем, щелкнув по подписи к этому полю, вызовем окно ее свойств и введем в ячейку **Подпись** значение *Новая цена*.

Вызовем окно свойств второго поля. Чтобы подсчитать изменение цены, нужно ввести в ячейку **Данные** формулу *=[Новая цена] – [Цена]*. Для ее создания используем построитель выражений.

Его средняя колонка содержит список имен ЭУ, размещенных в создаваемой форме. Сделаем двойной щелчок по имени **Новая цена**, затем щелчок по кнопке со знаком минус (-) и снова двойной щелчок по имени **Цена**. В верхнем поле появилось нужное выражение (см. рис. 6.8). Для возврата в окно свойств поля нажмем кнопку **ОК**.

Также установим для этого поля формат *Денежный*. Щелкнув по подписи к этому полю, вызовем окно ее свойств и введем в ячейку **Подпись** значение *Изменение*. Затем увеличим размеры обеих подписей, чтобы в них помещались новые названия полей. Для этого выделим обе подписи и в пункте **Размер** меню **Формат** выберем команду **По размеру данных.**

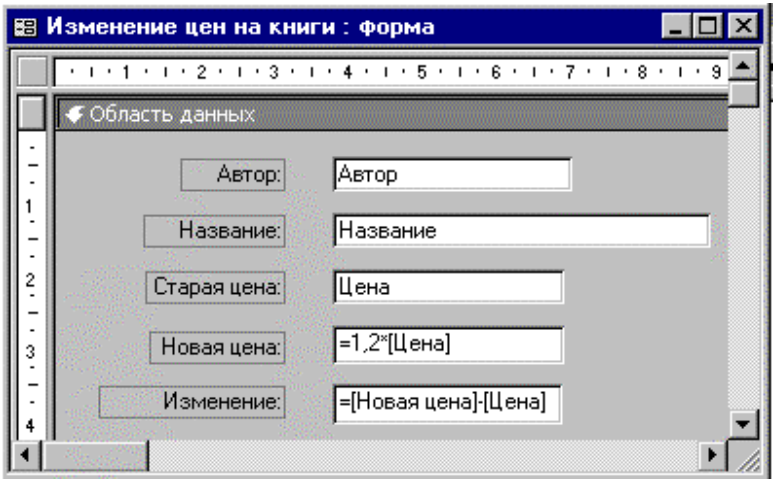


*Рис. 6.9. Форма **Изменение цен на книги** в режиме конструктора*

Выровняем размещенные в форме ЭУ. Для этого выделим все подписи и выберем в меню **Формат** пункт **Выровнять**, а затем пункт **по правому краю**. Аналогичную операцию проделаем для всех полей, но выберем для них выравнивание по левому краю. Для выравнивания числовых значений выделим последние три поля и щелкнем по кнопке **По левому краю** панели инструментов **Формат**. В завершение процедуры выравнивания

выделим все ЭУ, выберем в меню **Формат** пункт **Интервал по вертикали**, а затем — пункт **сделать равным**.

После выравнивания полей и их подписей созданная форма в режиме конструктора приобретает вид, представленный на рис. 6.9, а в режиме формы — вид, представленный на рис. 6.10.

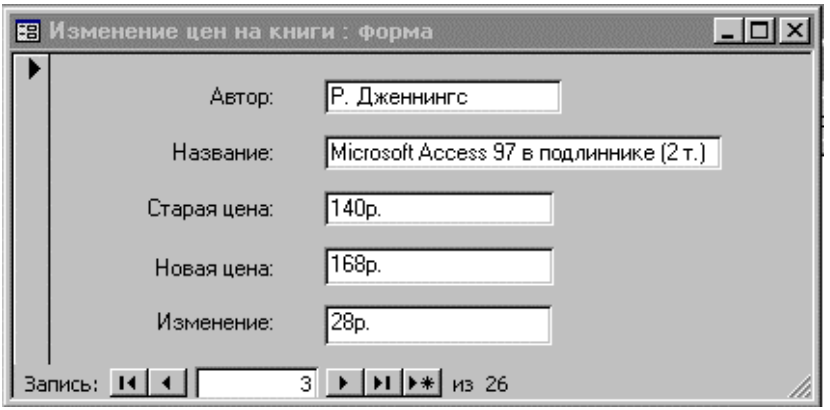


*Рис. 6.10. Форма **Изменение цен на книги** в режиме формы*

Отметим, что в вычисляемых полях формы могут использоваться ссылки только на поля, содержащиеся в ее источнике данных (таблице или запросе). Эти поля могут и не быть размещены в форме. В построителе выражений для получения доступа к таким полям нужно щелкнуть по элементу списка в средней колонке <*Список полей*>. В выражении можно также использовать ссылки на ЭУ, содержащиеся в других открытых *(«загруженных»)* формах.

6.3.4. Создание списка или поля со списком

При вводе значения в поле таблицы или запроса через форму быстрее и проще выбрать нужное значение из списка, чем вводить его с клавиатуры. При этом исключаются ошибки ввода. Существуют два ЭУ, с помощью которых можно организовать прокручиваемый список значений — поле со списком и список.

Список изображается в форме постоянно, в качестве вводимого значения может быть выбрано только значение из списка.

В *поле со списком* изображение списка появляется лишь при щелчке по кнопке, размещенной в правом конце поля. Кроме того, пользователь может разрешить ввод новых значений, не содержащихся в поле со списком.

Для создания этих ЭУ нужно нажать кнопку **Список**  или кнопку **Поле со списком**  на панели элементов, затем выделить нужное поле в списке полей и перенести его в нужное место формы. Чтобы определить, какие строки должны быть включены в список или поле со списком, следует использовать свойства ЭУ **Источник строк** и **Тип источника строк**.

Значения списка могут быть заданы непосредственно в строке **Источник строк**, они разделяются точками с запятой. В этом случае в свойство **Тип источника строк** помещается значение *Список значений*.

В качестве значений списка можно использовать строки из таблицы или запроса. В свойстве **Источник строк** задается имя этой таблицы или запроса, а в свойстве **Тип источника строк** устанавливается значение *таблица/запрос*.

В качестве значений списка может также использоваться список полей таблицы/запроса. В свойстве **Источник строк** задается имя таблицы/запроса, а в свойстве **Тип источника строк** — значение *Список полей*.

При использовании списка или поля со списком для просмотра и обновления данных пользователь выделяет в списке строку и вводит ее в поле таблицы или запроса, присоединенное к этому ЭУ. Имя этого поля содержится в свойстве **Данные**. Список, являющийся источником строк, часто состоит из одного столбца. В этом случае значение свойства **Число столбцов** в свойствах ЭУ равно 1 и свойство **Присоединенный столбец** тоже равно 1, но можно задать список, содержащий нескольких столбцов. Один из столбцов (присоединенный) будет содержать вводимые значения, а остальные столбцы — пояснения для них.

Для определения числа столбцов в списке используется свойство **Число столбцов**. Например, если источником строк списка является таблица или запрос и значение свойства **Число столбцов** равно 3, то в список будут включены первые три поля таблицы или запроса.

Свойство **Ширина столбцов** определяет ширину каждого столбца в единицах, которые заданы с помощью панели управления Windows. Эти величины разделяются в списке точками с запятой. Если ширина столбца не указывается (оставлена пустой), то используется стандартная ширина 2,54 см (1 дюйм), а если задана нулевая ширина, то столбец вообще не отображается в списке.

После выбора нужной строки из списка, состоящего из нескольких столбцов, в поле вводится значение, которое содержится в столбце, указанном с помощью свойства **Присоединенный столбец**. В этом свойстве задается порядковый номер в списке присоединенного столбца. В свойствах **Поля со списком** есть свойство **Ограничиться списком**. Если

оно имеет значение *Нет*, то пользователь может вводить в это поле, кроме значений из списка, любые другие.

Пример 6.3. В Форме **Итоги сессии**, созданной на базе таблицы **Сессия**, размещено поле со списком **Экзамен**. В качестве списка используется таблица **Экзамены**, состоящая из двух полей: **Код** и **Экзамен**. В ЭУ **Экзамен** в режиме формы при открытии списка появляется список названий экзаменов, содержащийся в поле **Экзамен** таблицы **Экзамены**. При выборе нужного экзамена из этого списка в поле **Код экзамена** таблицы **Сессия**, к которому присоединен этот ЭУ, вводится значение кода выбранного экзамена, взятое из поля Код таблицы **Экзамены**.

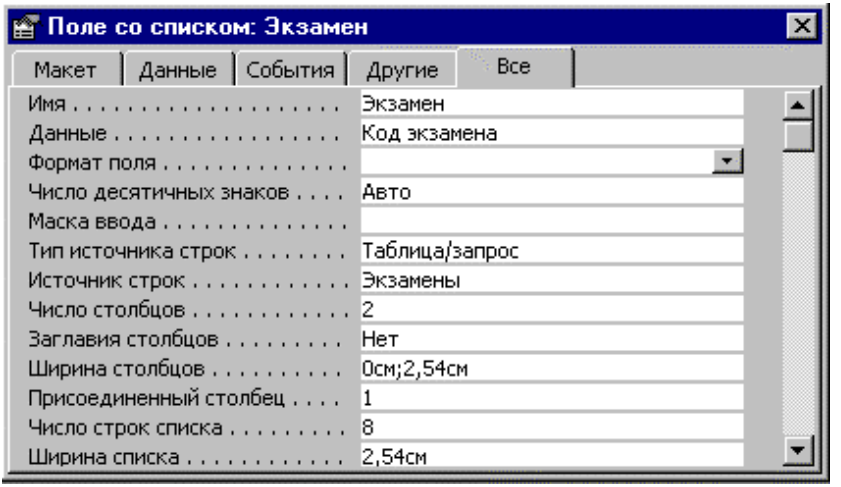


*Рис. 6.11. Список свойств ЭУ **Поле со списком***

Значения свойств ЭУ **Экзамен** в форме **Итоги сессии** приведены на рис. 6.11, а вид этого ЭУ на экране в режиме формы — на рис. 6.12.

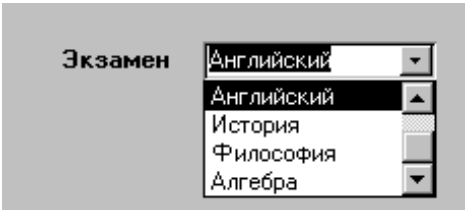


*Рис. 6.12. Вид ЭУ **Поле со списком***

При создании ЭУ **Поле со списком** или **Список** можно использовать мастера по созданию списков. Проиллюстрируем его работу на примере создания поля со списком из примера 6.3. Для вызова мастера нужно, чтобы кнопка **Мастера элементов** на панели элементов была с нажатом состоянии. Тогда после щелчка по кнопке **Поле со списком** и последующего щелчка по тому месту в форме, где

предполагается разместить этот ЭУ, начинает свою работу мастер **Создание полей со списком.**

1. На первом шаге мастер предлагает указать источник данных для поля со списком. По умолчанию считается, что значения берутся из таблицы или запроса. Так как в качестве источника данных будет взята таблица **Экзамены**, то нужно щелкнуть по кнопке **Далее**.
2. Мастер просит указать таблицу или запрос со значениями, которые будет содержать поля со списком. В списке таблиц нужно выбрать таблицу **Экзамены** и щелкнуть по кнопке **Далее**.
3. Затем нужно, щелкнув по кнопке >>, отобрать оба поля таблицы, используемые в списке: **Код** и **Экзамен**.
4. Так как **Код** — ключевое поле, то Access автоматически выбирает его в качестве источника данных (присоединенного столбца) для поля со списком. На этом шаге можно изменить ширину полей, используемых в подстановке, и указать, надо ли скрыть присоединенный столбец. Оставим без изменения установку по умолчанию: скрыть.
5. Мастер спрашивает, нужно ли сохранить в БД выбранное значение из списка. В качестве ответа надо в списке полей таблицы **Сессия** выбрать поле **Код экзамена**.
6. На последнем шаге зададим подпись поля со списком **Экзамен** и нажмем кнопку **Готово**.

Просмотр списка свойств созданного с помощью мастера ЭУ показывает, что он практически не отличается от списка свойств, представленного на рис. 6.11. Лишь в свойстве **Источник строк** вместо имени таблицы **Экзамены** содержится инструкция SQL, задающая отбор обоих полей этой таблицы.

Вопросы мастера по созданию списка фактически совпадают с вопросами, задаваемыми мастером подстановок. Это совпадение не случайно, так как столбец подстановки не что иное, как поле со списком (список) данных для поля таблицы. При переносе поля подстановки из списка полей в область данных формы Access создает присоединенный к нему ЭУ **Поле со списком** или **Список.** Конкретный вид присоединенного ЭУ определяется тем, какой тип ЭУ для вывода значений поля подстановки в форме указан в его свойстве **Тип элемента управления,** находящемся на вкладке **Подстановка** (см. рис. 2.4).

6.3.5. Создание кнопки

Кнопки в форме используются для активизации действия или последовательности действий. С помощью кнопок можно, например, организовать меню для работы приложения. Чтобы указать, что должна делать кнопка, необходимо связать с ней макрос или *процедуру обработки*

события[1], написанную на языке VBA. Кнопки можно создавать с помощью мастера и самостоятельно. Мастер **Создание кнопок** создает кнопки, выполняющие стандартные действия (число стандартных действий в Access 97 равно 32). С помощью таких кнопок можно открыть или распечатать отчет или форму, переместиться к новой записи таблицы, удалить запись и т. д. Мастер сам создает процедуры обработки событий, выполняющиеся при нажатии кнопки. Чтобы создать кнопку с помощью мастера, нужно:

1. Находясь в режиме конструктора формы, убедиться, что включена кнопка **Мастера** на панели элементов.
2. Щелкнуть по кнопке **Кнопка**  на панели элементов, а затем по тому месту в форме, куда нужно поместить кнопку. На экране появится первое окно диалога **Создание кнопок**.
3. Далее нужно выбрать действие, которое должно быть выполнено при нажатии кнопки, и ответить на вопросы мастера, связанные со сделанным выбором.
4. Ввести текст или выбрать нужный рисунок для размещения на кнопке.
5. В последнем окне диалога нужно задать понятное имя кнопки и нажать кнопку **Готово**.

Проверить действие кнопки можно, щелкнув по ней в режиме формы. Для просмотра и редактирования процедуры обработки события нужно щелкнуть правой кнопкой мыши на созданной кнопке и выбрать в контекстном меню пункт **Обработка событий**. Access сделает доступным текст процедуры *ИмяКнопки_Click*, запускаемой при нажатии на кнопку.

Пример 6.4. Нужно создать в форме кнопку, нажатие на которую позволит запустить Excel.

Для этого щелчком по кнопке **Кнопка** на панели элементов вызовем мастера **Создание кнопок**. Выберем в левом окне **Категории** пункт *Приложение*, а в правом окне **Действия** из списка возможных действий — *Запуск MS Excel*. Затем выберем для размещения на кнопке предложенный мастером рисунок «MS Excel» и нажмем кнопку **Готово**. Access создаст кнопку  , нажатие на которую в режиме формы приведет к запуску Excel. При установке указателя мыши на эту кнопку появится всплывающая подсказка *Запуск* Excel.

[1] *Процедура обработки события* — это процедура, автоматически выполняемая в ответ на событие, возникающее в результате различных действий пользователя (нажатие кнопки, открытие или закрытие формы и т.п.), выполнения программы или генерируемое Access.

6.3.6. Создание ЭУ Подчиненная форма/отчет

ЭУ **Подчиненная форма/отчет** предназначен для размещения в одной (главной) форме другой (подчиненной) формы. Подчиненная форма обычно применяется для просмотра и редактирования информации в связанных таблицах (см. п. 6.6). В этом качестве можно использовать ранее созданную форму.

Для создания ЭУ **Подчиненная форма/отчет** проще всего воспользоваться услугами специального мастера. Для этого нужно выполнить следующие действия:

1. Находясь в режиме конструктора формы, убедиться, что включена кнопка **Мастера**  на панели элементов.
2. Затем нажать кнопку **Подчиненная форма/отчет**  на панели элементов и установить указатель мыши на то место в форме, куда нужно поместить подчиненную форму, и нажать левую кнопку. На экране появится первое окно диалога **мастера Создание подчиненных форм и отчетов.**
3. Если уже существует форма, которую можно использовать в качестве подчиненной, то следует выбрать значение переключателя формы, а затем указать форму из списка. В противном случае нужно выбрать значение таблица или запрос.
4. Если готовой подчиненной формы нет, то во втором окне диалога нужно выбрать таблицу (запрос), являющуюся источником данных для подчиненной формы, и перебросить нужные поля из списка *Доступные поля* в список *Выбранные поля*. Если предполагается установление связи с главной формой, то в список выбранных полей должны быть обязательно включены поля, используемые для связи.
5. Далее мастер предлагает установить связь между главной и подчиненной формами. Имеется две возможности: либо воспользоваться списком возможных связей, либо самостоятельно выбрать поля связи между главной и подчиненной формами.
6. В последнем окне диалога задается имя подчиненной формы, которая сохраняется Access, как отдельная форма.

Проверить полученный результат можно, перейдя в режим формы.

6.3.7. Создание набора вкладок

ЭУ **Набор вкладок** является новым элементом Access 97. С его помощью можно создавать многостраничные формы, подобные окну свойств ЭУ (см. рис. 6.11). Этот ЭУ можно использовать для экономии места на экране и отображения информации из нескольких таблиц. На

страницах набора вкладок разрешается размещать любые ЭУ, кроме другого набора вкладок.

Для создания этого ЭУ нужно нажать кнопку **Набор вкладок**  на панели элементов и щелкнуть левой кнопкой мыши в области данных формы для создания нового ЭУ. При отпускании кнопки мыши Access создает ЭУ **Набор вкладок**, состоящий из двух страниц. В зависимости от того, какие данные и как Вы хотите отображать, может понадобиться включить в ЭУ **Набор вкладок** дополнительные страницы.

Для добавления страницы в ЭУ Набор вкладок нужно:

1. Щелкнуть правой кнопкой мыши на этом ЭУ. Access выведет контекстное меню.
2. Выбрать пункт Добавить вкладку. Access вставит новую страницу за последней страницей.

Для изменения порядка страниц в ЭУ Набор вкладок нужно:

1. Щелкнуть правой кнопкой мыши на ЭУ Набор вкладок. Access выведет контекстное меню.
2. Выбрать пункт **Последовательность вкладок**. Access выведет диалоговое окно **Порядок страниц** с перечнем страниц.
3. В этом списке нужно выбрать страницу, чью позицию вы хотели бы изменить и нажимать кнопки **Вверх** или **Вниз**, пока страница не окажется в нужной позиции.

Для изменения названия страницы в ЭУ Набор вкладок нужно:

1. Щелкнуть правой кнопкой мыши на корешке нужной страницы ЭУ Набор вкладок. Access выведет контекстное меню.
2. Выбрать пункт Свойства. Access выведет диалоговое окно свойств данной страницы.
3. В строке Подпись ввести новое название страницы.

6.3.8. Создание специальных эффектов

При разработке форм можно использовать различные цвета, шрифты и другие специальные эффекты для придания форме привлекательного вида. Прямоугольники и линии можно использовать в форме для привлечения внимания к важной информации, группирования логически связанных элементов, проведения различных границ.

Чтобы нарисовать прямоугольник, нажмите кнопку **Прямоугольник** на панели элементов, поместите указатель мыши в точку, где будет находиться верхний левый угол прямоугольника, и при нажатой левой кнопке мыши переместите указатель в нижний правый угол прямоугольника. Если прямоугольник закрывает размещенные внутри него ЭУ, в меню **Формат** выполните команду **На задний план**. Для изменения размеров

прямоугольника выделите его и измените с помощью маркеров. Для перемещения прямоугольника используется маркер перемещения.

Для рисования линии нажмите кнопку **Линия** и проведите нужную линию. Чтобы создать строго горизонтальную или вертикальную линию, нужно при ее проведении удерживать нажатой клавишу **Shift**.

Рельефное оформление прямоугольника или линии можно осуществить с помощью кнопок **Цвет линии/границы**  , **Толщина линии/границы**  , **Обычное оформление**  на панели форматирования.

6.3.9. Размещение рисунков и других объектов OLE

Для улучшения вида форм и отчетов Access в них можно размещать рисунки, диаграммы и прочие объекты, созданные другими приложениями. Например, в заголовок формы можно добавить эмблему фирмы, а объемы продаж в отчете представить в виде диаграммы. Вставка таких объектов в формы и отчеты осуществляется с помощью протокола OLE (протокола связывания и внедрения объектов).

Краткое описание протокола OLE

Протокол OLE — это метод передачи информации в виде объектов (объектов OLE) между различными приложениями Windows. Этот метод похож на копирование текста и графики в буфер обмена Windows с последующей вставкой в другие приложения. Все программы Microsoft Office поддерживают протокол OLE.

Объект OLE — это произвольная порция данных, созданных приложением Windows, поддерживающим протокол OLE. В качестве объекта OLE может фигурировать как целый документ (например, документ Word или электронная таблица Excel), так и его часть (фрагмент (текста или блок ячеек электронной таблицы). Можно также использовать различные графические изображения (фотографии, рисунки, диаграммы), видеоклипы, звуковые файлы и др. Приложение, которое используется для создания объекта OLE, называется *исходным приложением,* или *приложением-сервером OLE*, а файл, который содержит этот объект, — *исходным файлом*. Объект OLE кроме самих данных также содержит информацию об исходном файле.

Объект OLE может быть *связан* или *внедрен* в форму или отчет Access. *Связанные* и *внедренные объекты* отличаются местоположением данных и способом их изменения после помещения в форму.

При связывании создается ссылка на объект, которая помещается в форму. Сам объект остается на своем месте — в исходном файле. Связанный объект будет автоматически обновляться при обновлении исходного файла. Связывание объекта удобно применять при работе с

большими файлами, которые нежелательно включать в файл БД, а также с объектами, используемыми в нескольких формах и отчетах. Если связанный файл объекта перемещен, необходимо повторно установить связь.

При внедрении объекта создается его копия, которая вставляется в форму. Внедренный объект становится частью формы и теряет связь с исходным файлом. При двойном щелчке по внедренному объекту он открывается с помощью создавшего его приложения-сервера. Все вносимые в него изменения отображаются в содержащей его форме.

Технология OLE не только обеспечивает доступ к объектам, созданным другими приложениями, но и существенно упрощает процедуру их изменения. Пользователь может открыть исходное приложение и внести изменения в объект OLE, не прекращая разработку формы или отчета.

Объект OLE может быть *присоединенным* или *свободным*.

Присоединенные объекты хранятся в файле БД. Для изменения и даже создания такого объекта не требуется покидать Access. При изменении присоединенного объекта из Access изменяется только объект в БД, а в исходном файле внесенные изменения не отражаются.

Свободный объект можно просматривать и изменять, находясь в форме или отчете. Однако внесенные в него изменения сохраняются в исходном файле, а не в БД Access. Кроме того, исходный объект может быть изменен в отсутствие Access. При этом внесенные изменения будут отражены в форме или отчете при их последующем открытии.

Для отображения объектов OLE применяются два типа ЭУ: **Присоединенная рамка объекта** и **Свободная рамка объекта**. Элемент **Присоединенная** рамка объекта позволяет отобразить в форме или отчете рисунки, диаграммы и другие объекты OLE, хранимые в полях БД Access. Элемент **Свободная рамка объекта** применяется для отображения объектов, хранимых вне таблиц.

Использование ЭУ Присоединенная рамка объекта

Присоединенные рамки объектов следует использовать для размещения в форме объектов OLE, хранящихся в полях таблиц. В режиме формы эти ЭУ используются для изображения, ввода и изменения объектов в текущей записи таблицы точно так же, как поля используются для изображения, ввода и изменения текста. При этом каждая запись таблицы содержит (или не содержит) свой объект.

Например, поле **Изображение** таблицы **Типы**, включенной в учебную базу данных Борей, содержит рисунок для каждого включенного в эту таблицу типа товара. Для изображения этих рисунков в форме или отчете можно использовать присоединенную рамку объекта.

Чтобы создать присоединенную рамку объекта:

1. Откройте форму или отчет в режиме конструктора;
2. Выберите **Список полей** в меню **Вид** (или нажмите кнопку **Список полей** на панели инструментов);
3. Перенесите поле, предназначенное для хранения объектов OLE, в макет формы или отчета. (Это поле должно быть OLE-полем.).

Access создаст рамку объекта, связанную с указанным полем. В режиме конструктора связанная рамка объекта изображается пустой; объекты из указанного поля изображаются в ней в режиме формы и предварительного просмотра или печатаются при печати формы.

Пользователь может изменить размеры и пропорции объектов, размещенных в форме.

Чтобы создать объект и внедрить его в присоединенную рамку объекта:

1. Откройте форму в режиме формы (или откройте таблицу, форму или отчет в режиме таблицы) и найдите запись, в которую следует добавить объект.
2. Выделите присоединенную рамку объекта (или поле таблицы), в которую следует добавить объект.
3. Выберите **Вставить объект** в меню **Правка**. На экране появится окно диалога **Вставка объекта** со списком доступных приложений-серверов OLE.
4. Выберите **Создать новый** или **Создать из файла** а затем выделите тип объекта, который следует внедрить.
5. Если в форме должен изображаться не сам объект, а заменяющий его значок, установите флажок В виде значка.
6. Нажмите кнопку **ОК**. Access откроет исходное приложение.
7. Создайте объект.
8. Для возврата в Access выберите **Выход** в меню **Файл** исходного приложения, а затем ответьте утвердительно на предложение обновить документ.

Access внедрит созданный объект в присоединенную рамку объекта и отобразит или сам объект (в режиме формы) или текст, указывающий тип объекта OLE, например, *Рисунок Paint* (в режиме таблицы).

6.3.11. Использование ЭУ Свободная рамка объекта

Свободные рамки объектов следует использовать для размещения в формах и отчетах объектов OLE, которые не требуется заносить в таблицу (в таком случае рамка объекта не будет связана ни с каким полем таблицы). Например, отчет **Счет**, включенный в базу данных **Борей**, содержит эмблему фирмы. Эта эмблема была разработана профессиональным

художником с помощью графического редактора Paint, а затем сохранена в отдельном файле. Для того чтобы иметь возможность изобразить в отчете другую эмблему или изменить существующую, следует разместить в макете отчета свободную рамку объекта для эмблемы.

Чтобы создать свободную рамку объекта и поместить в нее существующий объект:

1. Откройте форму или отчет в режиме конструктора;
2. Нажмите кнопку **Свободная Рамка объекта** на панели элементов;
3. Установите указатель в то место формы или отчета, куда следует поместить верхний левый угол рамки объекта, и нажмите кнопку мыши, чтобы создать рамку стандартных размеров, или укажите нужные размеры рамки с помощью мыши. На экране появится окно диалога **Вставка объекта** со списком OLE-приложений, зарегистрированных в Windows;
4. Выберите значение **Из файла**;
5. В поле **Файл** введите полное имя файла, содержащего объект, который следует внедрить или связать. Или нажмите кнопку **Поиск** и выберите нужный файл;
6. Если объект следует связать, а не внедрить, установите флажок **Связь**;
7. Если в форме должен изображаться не сам объект, а заменяющий его значок, установите флажок **В виде значка**;
8. Нажмите кнопку **ОК**. Access создаст свободную рамку объекта и изобразит в ней указанный объект.

Если в форму нужно вставить рисунок, который в дальнейшем не требует изменений, лучше использовать ЭУ **Рисунок**, так как в этом случае форма будет загружаться быстрее. Для этого следует щелкнуть по ЭУ **Рисунок** и указать местоположение графического файла, который содержит вставляемый рисунок. Можно также выбрать пункт меню **Вставка**, а затем **Рисунок**. Рисунок выбирается из указанного файла и помещается в рамку. После этого можно изменить пропорции и размеры рисунка, но редактировать его нельзя.

Вставленный рисунок встраивается в форму и не сохраняет связь с исходным файлом. Однако если предполагается использовать рисунок в нескольких формах или отчетах, то следует его связать. Для связывания рисунка следует задать для свойства **Тип рисунка** ЭУ **Рисунок** значение *Связанный*. В этом случае рисунок будет храниться не в БД, а в исходном файле.

Рисунок можно включить в форму (отчет) и как фоновый рисунок, занимающий все ее окно. Если требуется добавить фоновый рисунок, следует использовать свойство **Рисунок** формы или отчета.

6.4. Настройка формы

Настройка формы производится путем изменения свойств как самой формы, так и ее размещенных в ней ЭУ.

6.4.1. Изменение свойств ЭУ

Свойства ЭУ, размещенного в форме, вызываются через контекстное меню ЭУ или нажатием кнопки **Свойства**  на панели инструментов. Изменяя некоторые свойства ЭУ, можно настроить форму для более удобного применения.

Присоединенный ЭУ **Поле** наследует по умолчанию такие свойства поля таблицы или запроса, как формат поля, число десятичных знаков, маска ввода, текст строки состояния, условие на значение, сообщение об ошибке и др. Эти свойства ЭУ можно изменить. Например в свойстве **Текст строки состояния** можно задавать короткую инструкцию типа *"Вводится полностью (без сокращения)"*. Эта инструкция будет появляться в строке состояния, когда наступит ввод значения в это поле в режиме формы.

Для ускорения ввода данных через форму можно для некоторых ЭУ задать свойство **Значение по умолчанию**, а в свойстве **Условие на значение** задать проверку вводимых значений. Чтобы запретить ввод или изменение данных в режиме формы для присоединенного ЭУ, нужно установить для его свойства **Блокировка** значение «*Да*». Для полного запрета изменений, выводящихся в форме данных, проще установить для свойства формы **Разрешить изменения** значение «*Нет*».

Иногда требуется, чтобы размещенный в форме ЭУ был невидим в режиме формы. В этом случае следует установить для его свойства **Вывод на экран** значение «*Нет*».

При размещении в форме каждый ЭУ получает стандартное имя, например, «Поле3» или «Надпись2». Это имя можно изменить на более содержательное, указав новое имя в свойстве **Имя**. Обычно это делается, если предполагается использовать имя данного элемента в каком-либо выражении или программе. Давая ЭУ новое имя, нужно следить за тем, чтобы оно не совпало с именем другого ЭУ, размещенного в форме.

Размеры ЭУ можно регулировать с помощью маркеров размера. Если ЭУ отражает данные текстового поля, содержащего большой текст, то помимо увеличения размера ЭУ, можно воспользоваться свойством **Полосы прокрутки**.

Свойство **Расширение** и **Сжатие** используется при печати формы. Если в этих свойствах задано значение *«Да»*, то размеры ЭУ будут при печати изменяться автоматически так, чтобы содержимое было напечатано полностью.

Можно также изменить выражение для вычисляемого поля (свойство **Данные**) и формат вывода на экран его значения (свойство **Формат поля**).

6.4.2. Изменение последовательности перехода

Кроме изменения свойств ЭУ формы можно изменить последовательность перехода по полям формы. В режиме формы перемещение по полям формы осуществляется нажатием клавиши **Tab** или **Shift+Tab** (в обратной последовательности). Последовательность этих перемещений задается в пункте **Последовательности перехода** меню **Вид**. В режиме конструктора формы при добавлении в форму нового ЭУ он добавляется и в **Последовательность перехода**. С помощью команды **Последовательность перехода** из меню **Вид** пользователь может вмешаться в этот порядок и поменять его на нужную ему последовательность. В диалоговом окне вызванной команды нужно указать на расположенную слева от названия ЭУ кнопку, нажать левую кнопку мыши и «перетащить» ЭУ в нужное место в списке **Последовательность**.

6.4.3. Добавление разделов

Добавление или удаление разделов в форму осуществляется через меню **Вид**. В любом разделе формы можно поменять, добавить или удалить любые ЭУ. Можно изменять и размеры раздела. У разделов формы также есть свойства, которые можно менять.

Так, раздел **Заголовок формы** можно отображать на экране в режиме формы, если установить значение Только на экран в свойстве **Режим вывода**. При печати формы в этом случае раздел не печатается.

Можно запретить вывод примечаний на экран в свойстве **Режим вывода**, но разрешить их печать или, наоборот, разрешить вывод примечаний на экран, но не их распечатку.

Размеры разделов в форме можно изменять, перемещая их нижнюю границу вверх или вниз нажатой левой кнопкой мыши.

Свойства раздела вызываются двойным щелчком мыши на заголовке раздела или через контекстное меню, когда курсор мыши находится внутри раздела вне каких - либо ЭУ.

6.4.4. Изменение свойств формы

Сама форма также имеет свойства (см. рис. 6.13), которые вызываются двойным щелчком мыши из бланка формы, если указатель находится на серой поверхности формы вне всех разделов или на пересечении горизонтальной и вертикальной линеек.

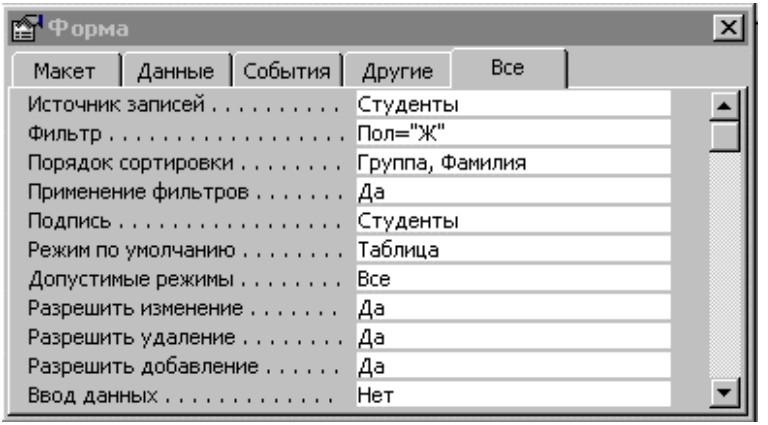


Рис. 6.13. Окно свойств формы

В свойствах задаются режимы работы с формой, ее внешний вид, источник данных и другие характеристики. Одним из основных свойств формы является **Режим по умолчанию**. В этом свойстве устанавливается режим окна формы при ее открытии. Перечислим возможные режимы:

- *простая форма* — на экране отображается одна запись;
- *ленточная форма* — отображение нескольких записей;
- *таблица* — показ записей в табличном формате.

В свойствах формы можно задать порядок сортировки ее записей. Для этого следует указать в свойстве **Порядок сортировки** имена полей, по которым производится сортировка, разделив их запятыми. Если сортировка производится по одному полю, то проще всего задать ее порядок в режиме формы или режиме таблицы, щелкнув мышью сначала по этому полю, а затем по одной из кнопок **Сортировка по возрастанию**  или **Сортировка по убыванию**  , задающих нужное направление сортировки.

Требуемый порядок сортировки можно задать непосредственно в источнике данных формы. Для этого нужно щелкнуть по свойству **Источник записей**, а затем по появившейся справа кнопке построителя запросов. Откроется окно конструктора запросов, содержащее базовую таблицу или запрос. Нужно создать и сохранить запрос, задающий нужный вид сортировки. Его инструкция SQL заменит в свойстве **Источник записей** имя базовой таблицы/запроса. Свойство **Источник записей** можно использовать и для другой модификации базового источника данных формы, например отбора подмножества записей в соответствии с каким-либо критерием.

В свойствах формы задается и режим работы с записями. Режим может быть четырех типов, задается значениями **Да/Нет**:

- *Разрешить изменения* — можно просматривать, корректировать и вводить новые записи.
- *Разрешить добавление* — после перехода к последней записи будет представлена пустая строка для ввода новой записи.
- *Разрешить удаление* — можно удалять записи.
- *Ввод данных* — позволяет начать работу с записями с пустой строки, т. е. с ввода новой записи.

По умолчанию свойство *Ввод данных* имеет значение «*Нет*». В этом случае при открытии формы в ней выводятся существующие записи. Если форма предназначена для ввода новых записей, то нужно установить значение этого свойства — «*Да*». Тогда при открытии формы будет выводиться только пустая запись.

6.5. Создание диаграмм

В формы и отчеты Access можно включать диаграммы. При создании формы/отчета после выбора источника данных задается способ создания **Диаграмма**, вызывающий мастера **Создание диаграмм** (приложение Microsoft Graph 97). Лучше всего в качестве источника данных при создании диаграмм использовать перекрестные запросы. В них данные уже подытожены и отражены в виде таблицы с шапкой.

После выбора источника данных и способа создания формы/отчета — **Диаграмма** последует цепочка диалоговых окон мастера **Создание диаграмм**.

1. В первом окне нужные поля из списка **Доступные поля** переправляются в **Список полей диаграммы**.
2. Во втором окне выбирается тип диаграммы: областная, график, гистограмма и т.д.
3. В третьем окне мастера **Создание диаграмм** появляется образец выбранного типа диаграммы и предлагается выбрать поля, которые будут отображаться в различных областях диаграммы (см. рис. 6.14). На образце в прямоугольниках, расположенных в областях, видны названия полей, которые выбраны на первом этапе. В правой части окна расположены кнопки полей. Если Вас не устраивают данные, расположенные в прямоугольниках, можно заменить их, «перетаскивая» из одного прямоугольника или кнопки в другой. В области диаграммы часто отображаются названия полей с добавлением названия операции группирования, которую мастер провел с данными. Если Вы хотите изменить выбор мастера, щелкните по такому полю

дважды и в появившемся списке операций выберите нужную группировку, например, вместо *Сумма — Среднее*. Если никакая операция не нужна, выберите значение *Отсутствует*.

4. В четвертом диалоге задается заголовок и система обозначений областей диаграммы. После этого можно нажать кнопку **Готово**.

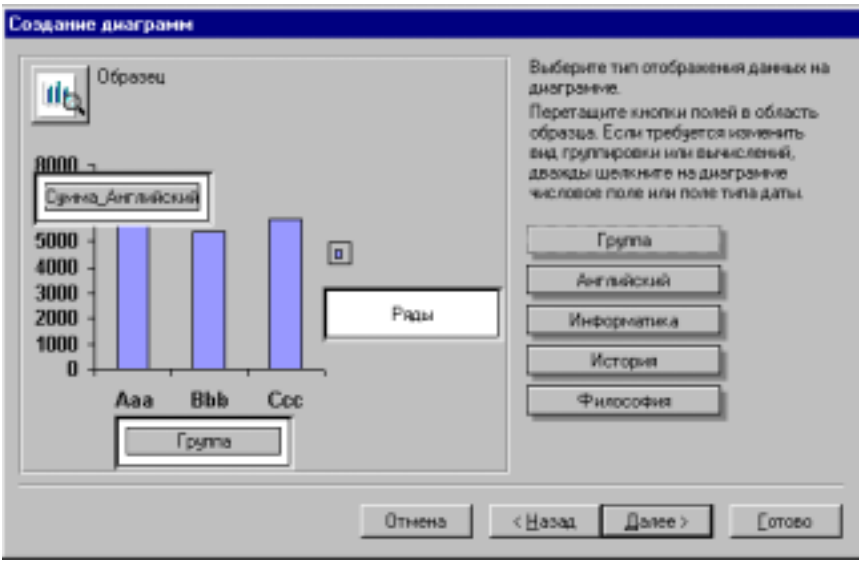


Рис. 6.14. Окно Мастера с макетом диаграммы

Диалоги мастера диаграмм закончены, но почти наверняка Вас не устроит вид созданной диаграммы. Посмотрите на результат работы в режиме формы и войдите в режим конструктора. Дважды щелкните по диаграмме для активизации Microsoft Graph. Диаграмма будет выделена в рамку, а также изменится главное меню Access, в нем добавится пункт **Диаграмма**. В рамке будет отражена не Ваша диаграмма, а похожий образец. Вы можете выделять любые его части, вносить в них изменения, форматировать их. Эти же действия можно выполнять через пункты меню **Диаграмма**.

Пример 6.5. Создадим диаграмму **Итоги сессии** на базе перекрестного запроса **Средние баллы по группам** из БД **Деканат** (см. пример 5.29).

Для построения диаграммы перейдем на вкладыш **Форма** и нажмем кнопку **Создать**. Выберем из списка запрос **Средний балл по группам** и способ создания формы — **Диаграмма**.

В первом окне перебросим все поля из списка **Доступные поля** в **Список полей диаграммы**. Затем выберем тип диаграммы — **Гистограмма**.

Третье окно диалога будет иметь вид, представленный на рис. 6.14.

На нем видно, что в области диаграммы отображается гистограмма только для одного предмета — английского языка. Название поля в прямоугольнике — **Сумма_Английский**, т.е. Мастер диаграмм выбрал для полей со средним баллом функцию группирования **Суммирование**.

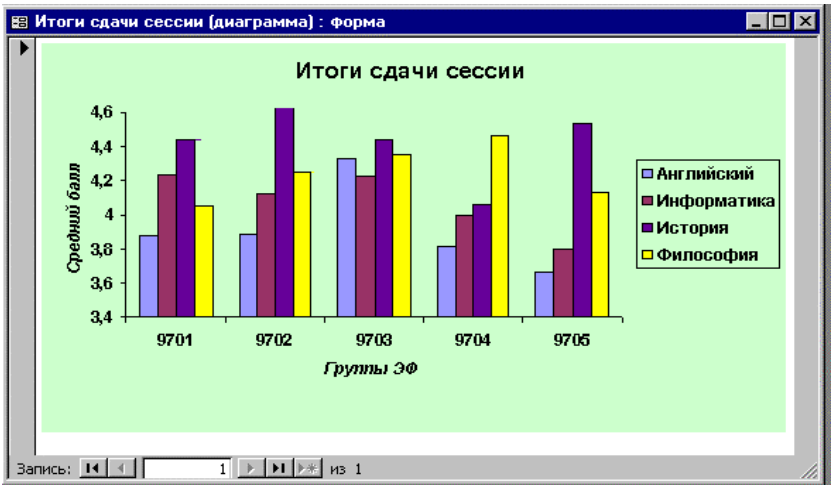


Рис. 6.15. Диаграмма в режиме формы

У нас средний балл уже подсчитан в запросе, поэтому операция группирования нам не нужна. Для ее отмены следует сделать двойной щелчок по полю **Сумма_Английский.** Появится список групповых операций. Из него нужно выбрать значение **Отсутствует**. После этого название поля изменится на **Английский**. Добавим поля **История**, **Информатика** и **Философия** «перетаскиванием» их из правой части окна в прямоугольник области диаграммы. Закроем диаграмму и дадим ей имя **Итоги сдачи сессии.**

Откроем созданную форму в режиме формы. Мы увидим, что нужно изменить размер диаграммы, добавить подписи по осям, изменить шкалу оси значений, а также формат текста осей и легенды. Для внесения изменений войдем в режим конструктора. На бланке формы будет находиться не наша диаграмма, а похожий образец. Щелкнем по нему дважды. Диаграмма выделится рамкой и изменится главное меню Access. В нем добавится пункт **Диаграмма**.

Выберем этот пункт меню, а затем пункт **Параметры диаграммы**. Введем название диаграммы *Итоги сдачи сессии*, подпись оси X — *Группы ЭФ* и подпись по оси Y — *Средний балл*. Выйдем из команды **Параметры диаграммы**. Установим размеры шрифта для названия диаграммы — 12 пунктов и шрифт подписей — 8 пунктов.

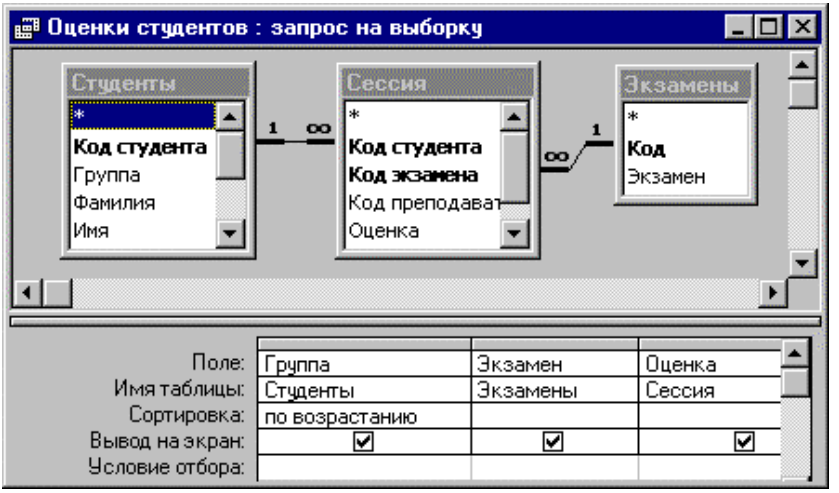


Рис.6.16. Бланк запроса ***Оценки студентов***

Выделим ось значений и вызовем контекстное меню. Выберем команду **Формат оси**. Установим минимальное значение шкалы — 3,4, максимальное значение — 4,6 и точку пересечения с осью X — 3,4.

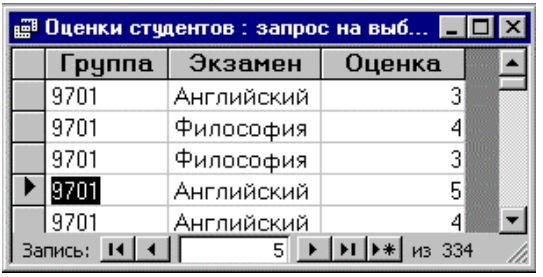


Рис. 6.17. Результат выполнения запроса ***Оценки студентов***

Выделим легенду и вызовем контекстное меню. Выберем команду **Формат легенды.** Зададим размер шрифта легенды — 8 пунктов. Для просмотра полученного результата снова вернемся в режим формы (см. рис. 6.15).

Мастер диаграмм может сам создавать итоговые перекрестные запросы для построения диаграммы.

Для создания приведенной выше диаграммы другим способом в качестве источника данных можно взять запрос **Оценки студентов**. Бланк запроса и результат его выполнения приведены на рис. 6.16 и 6.17.

Мастер диаграмм сам построит итоговый перекрестный запрос и предложит описанную выше цепочку диалогов. На третьем шаге, когда на экране появится макет диаграммы, нужно поменять групповую итоговую операцию по полю **Оценка**. Для этого следует дважды щелкнуть по названию поля **Сумма_Оценка** и выбрать в появившемся списке операцию *Среднее*. Дальнейшие действия такие же, как в вышеописанном процессе.

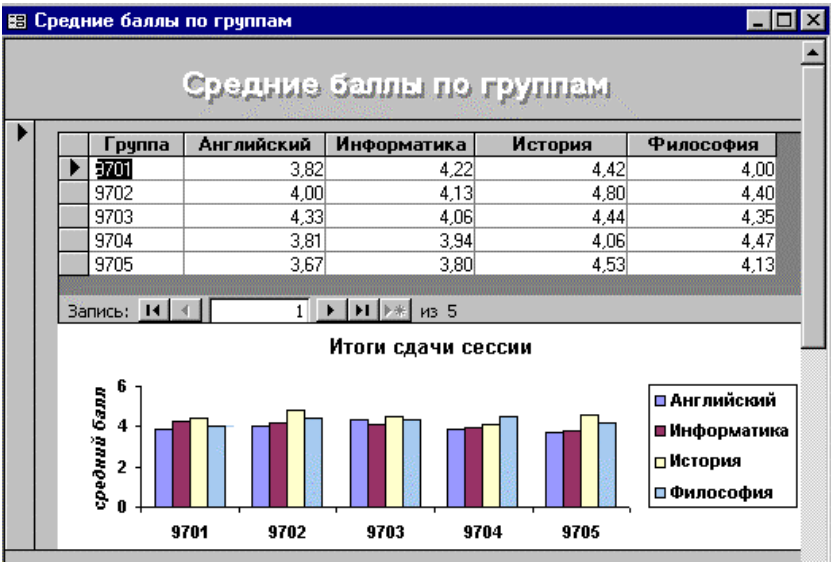


Рис. 6.18. Объединение таблицы и диаграммы в одной форме

Кроме внесения изменений во внешний вид диаграммы можно изменить ее источник данных. Для этого нужно открыть окно свойств диаграммы и щелкнуть по ячейке **Источник строк**. Затем можно или выбрать из списка новый источник данных или внести изменения в текущий источник, нажав кнопку построителя. В окно конструктора запросов будет загружен запрос — источник данных диаграммы. После внесения изменений, закрытия конструктора запросов и перехода в режим формы Access автоматически построит диаграмму по новым данным.

Можно объединить в одной форме числовое и графическое отображение информации из БД. Например, чтобы поместить в форму сведения об итогах сдачи сессии в виде таблицы и диаграммы, нужно сначала разместить в пустой форме подчиненную форму, связав ее с

запросом **Средние баллы по группам**. Затем следует вставить в форму диаграмму, основанную на этом же запросе, выбрав в меню пункт **Вставка**, а затем — **Диаграмма**. Будет запущен мастер **Создание диаграмм** и после настройки диаграммы форма приобретет вид, представленный на рис. 6.18.

6.6. Составные формы

Составная форма обычно используется для работы с информацией, находящейся в связанных таблицах. Она состоит из главной формы и одной или нескольких подчиненных форм. Форма называется *подчиненной*, если она содержится внутри другой формы. Обычно «контейнером» для подчиненной формы служит главная форма. Но Access позволяет создавать составные формы с двумя уровнями вложенности. В них главная форма может включать подчиненную форму, в свою очередь содержащую другую подчиненную форму.

Главная и подчиненная формы связаны между собой. Эту связь в ряде случаев Access может создать автоматически, используя информацию о связях и структуре таблиц или запросов, являющихся источниками данных для связываемых форм. Но чаще связь устанавливается самостоятельно самим пользователем в процессе создания составной формы или позднее заданием нужных значений соответствующих свойств подчиненной формы.

Обычно составная форма используется для отображения данных из таблиц или запросов, связанных отношением «один ко многим», причем главная форма содержит данные со стороны «один», а подчиненная форма — со стороны «многие». Если связь определена правильно, то просмотр информации в созданной форме будет синхронизирован: в подчиненной форме появятся лишь записи, связанные с текущей записью в главной форме. При этом подчиненная форма может быть выведена в любом режиме, а главная форма — только как простая форма.

Составная форма является также удобным средством для редактирования и добавления данных в таблицы, связанные отношением «один ко многим». При вводе новой записи в подчиненную форму она автоматически связывается с текущей записью в главной форме.

Создать составную форму можно одним из следующих способов:

- главная и подчиненная формы создаются одновременно с помощью мастера **Создание форм**;
- подчиненная форма создается в главной форме с помощью ЭУ **Подчиненная форма/отчет**;
- существующая форма добавляется в другую (главную) форму в качестве подчиненной путем «перетаскивания» ее значка мышью.

6.6.1. Использование мастера *Создание форм*

Это самый простой способ создания составной формы. Его можно использовать в том случае, когда между таблицами, являющимися источниками данных для главной и подчиненной форм, в схеме данных установлена связь «один ко многим».

Использование этого мастера для создания простой формы рассматривалось в п. 6.2. Чтобы создать с его помощью составную форму, нужно проделать следующие действия.

1. Находясь во вкладке **Формы**, щелкнуть по кнопке **Создать. В** открывшемся окне **Новая форма** (см. рис. 6.2) выбрать пункт **Мастер форм** и щелкнуть по кнопке **ОК**.
2. В первом окне мастера в списке **Таблицы/запросы** последовательно выбрать таблицы/запросы, поля которых будут присутствовать в форме, и переместить используемые поля из списка Доступные поля в список Выбранные поля с помощью кнопок >> и > или двойного щелчка мыши. Отметим, что в создаваемых главной и подчиненной формах эти поля будут размещены в таком же порядке, как и в списке Выбранные поля.
3. В следующем окне (см. рис. 6.21) Access предлагает указать тип представления данных в создаваемой форме путем выбора из списка в левой части окна имени таблицы/запроса, служащей источником для главной формы. Эта таблица/запрос должна находиться на стороне «один» в связях между отобранными таблицами/запросами, заданными в схеме данных БД.

   Проанализировав эти связи, Access разбивает отобранные поля на две группы, относя часть из них к главной, а остальные — к подчиненной форме. Это разбиение он выводит в правой части окна. Для создания обычной составной формы нужно выбрать значение переключателя **Подчиненная форма**. Если задать значение переключателя Связанная форма, то будет создана главная форма, содержащая отобранные Access поля и кнопку, нажатие которой приведет к открытию подчиненной формы.

   Если окно не появляется, то это означает, что Access не может создать с помощью мастера на основе имеющейся информации о связях требуемую составную форму и для ее создания следует использовать другие средства, описываемые ниже.

4. В последующих окнах диалога задается внешний вид и стиль оформления формы. После выбора варианта в окне мастера появляется соответствующий образец.
5. Последнее диалоговое окно дает возможность изменить заданные по умолчанию имена главной и подчиненной форм и указать режим, в котором должна появиться на экране созданная форма.

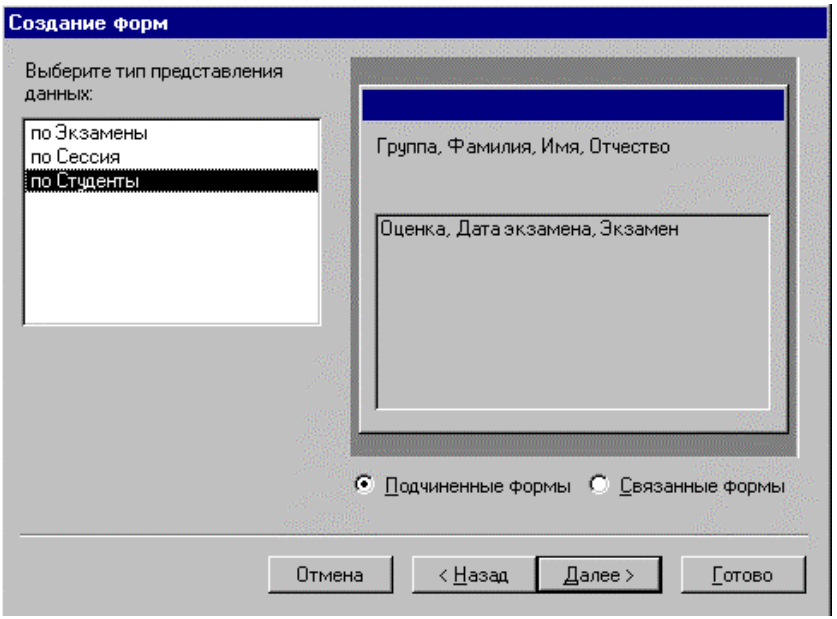


Рис. 6.21. Выбор типа представления данных

Мастер всегда создает составную форму, если источниками данных главной и подчиненной формы являются две таблицы, между которыми установлена связь типа «один ко многим», причем главная форма содержит данные из таблицы со стороны «один», а подчиненная форма — данные из таблицы со стороны «многие».

Мастер часто позволяет создать составную форму и в том случае, когда подчиненная форма использует информацию из нескольких связанных таблиц. При этом в зависимости от типа их связи может быть создана как обычная составная форма, так и составная форма с двумя уровнями вложенности.

Пример 6.6. Нужно создать составную форму **Оценки студентов,** причем главная форма должна содержать информацию о студенте (группа, фамилия, имя и отчество), а подчиненная — итоги сдачи сессии (название экзамена, полученная оценка и дата экзамена).

В создаваемой форме в качестве источника данных главной формы можно использовать таблицу **Студенты,** а подчиненной — таблицы **Сессия** и **Экзамены.** Между этими таблицами установлены связи, поэтому для создания формы можно воспользоваться мастером **Создание форм.**

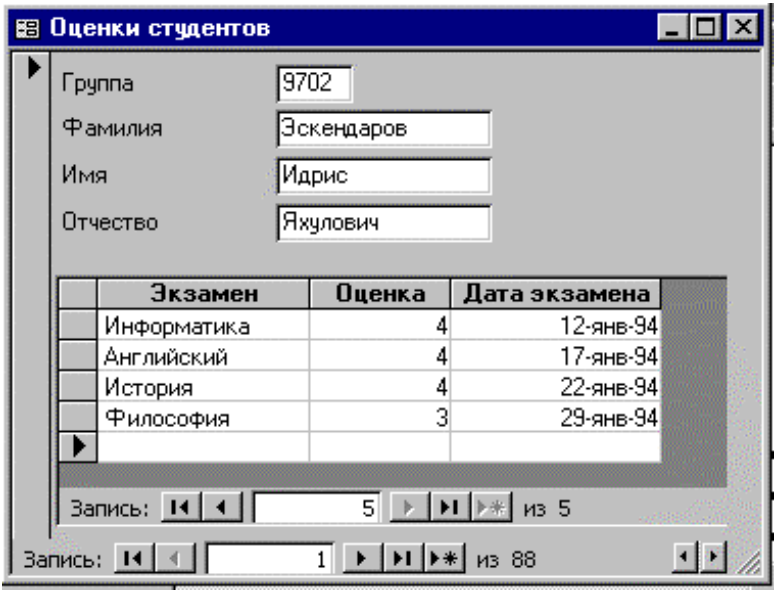


*Рис. 6.22. Форма **Оценки студентов** в режиме формы*

В первом диалоговом окне этого мастера следует выбрать:

- из таблицы **Студенты** — поля **Группа**, **Фамилия**, **Имя** и **Отчество**;
- из таблицы **Сессия —** поля **Оценка** и **Дата экзамена**;
- из таблицы **Экзамены —** поле **Экзамен**.

Используя информацию о связях таблиц в схеме данных, мастер выводит в левой части следующего окна список базовых таблиц, а в правой части — схему создаваемой формы (см. рис. 6.21). Выберем из левого списка элемент ***по Студенты***, сообщая тем самым мастеру, что источник данных в главной форме — таблица **Студенты**. Если же выбрать элемент списка ***по Экзамены,*** то мастер предложит создать составную форму, главная форма в которой будет содержать единственное поле **Экзамен**, а остальные поля будут помещены в подчиненную форму.

В следующих диалоговых окнах укажем, что подчиненная форма должна быть табличной формой, и выберем в качестве стиля формы *Обычный*.

В заключение введем в качестве названия главной формы **Оценки студентов,** подчиненной формы — **подчиненная к форме Оценки студентов** и укажем в качестве завершающего действия открытие формы.

Мастер создаст и сохранит обе формы, а затем откроет созданную форму в режиме формы (см. рис. 6.22).

В этом режиме можно изменить внешний вид подчиненной формы: поменять местами поля, изменить их ширину и скрыть лишние поля. Для внесения изменений в главную форму и для изменения размеров подчиненной формы следует перейти в режим конструктора.

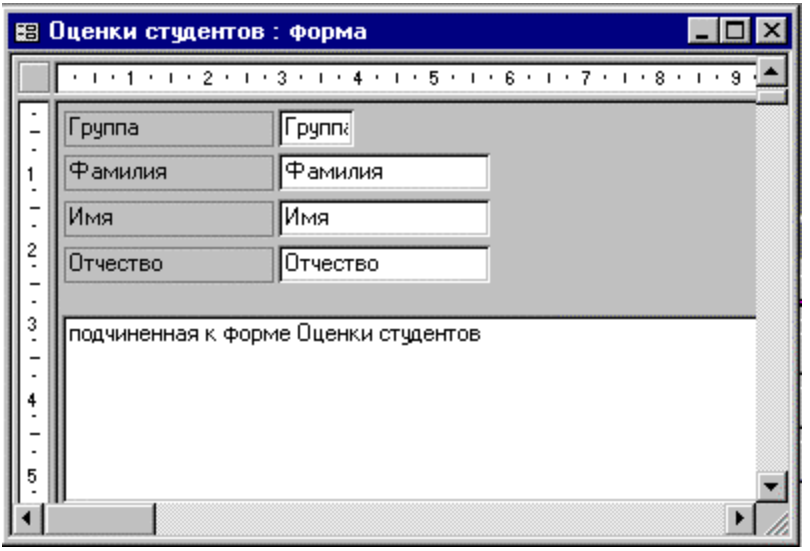


*Рис. 6.23. Форма **Оценки студентов** в режиме конструктора*

Чтобы осуществить дальнейшую настройку подчиненной формы (формат вывода информации, порядок сортировки, возможность внесения изменений и т.д.), нужно внести в режиме конструктора изменения в свойства ее ЭУ или самой формы. Открыть подчиненную форму в этом режиме можно, находясь в режиме конструктора в составной форме. Для этого достаточно сделать двойной щелчок по ЭУ **подчиненная к форме Оценки студентов**, в который мастер помещает подчиненную форму (см. рис. 6.23). После задания новых значений свойств и закрытия подчиненной

формы внесенные изменения можно просмотреть, открыв составную форму в режиме формы.

В этом примере Access связывает по полю **Код студента** главную и подчиненные формы, используя информацию о связях таблиц **Студенты** и **Сессия**. Сведения о полях связи между формами содержатся в свойствах подчиненной формы: **Подчиненные поля** и **Основные поля**. Для получения этой информации нужно щелчком правой кнопки мыши вызвать свойства ЭУ **подчиненная к форме Оценки студентов** (см. рис. 6.24).

Появление на экране новой записи в главной форме приводит к автоматическому обновлению содержимого подчиненной формы: она всегда будет содержать записи с оценками текущего студента.

Имя	подчиненная к форме Оценки студентов
Объект-источник	подчиненная к форме Оценки студентов
Подчиненные поля	Код студента
Основные поля	Код студента

Рис. 6.24. Информация о полях связи

Недостатком созданной формы является то обстоятельство, что она выводит на экран записи о студентах в порядке, определяемом значениями ключевого поля **Код студента**. Сортировку по другому полю, например **Фамилия**, проще всего задать в режиме формы или таблицы, щелкнув по этому полю, а затем по соответствующей кнопке сортировки на панели инструментов.

Для задания сортировки по нескольким полям, например **Группа** и **Фамилия**, следует указать их через запятую в свойстве главной формы **Порядок сортировки** (см. рис. 6.13). Другим способом является изменение источника данных главной формы. Для этого нужно вызвать окно свойств главной формы, щелкнуть по свойству **Источник записей**, а затем по появившейся справа кнопке построителя запросов. Откроется окно конструктора запросов, содержащее таблицу **Студенты**. Следует создать и сохранить запрос, задающий нужный вид сортировки. Его инструкция SQL заменит в свойстве **Источник записей** имя базовой таблицы **Студенты**.

Таблица **Экзамен** играет в этом примере вспомогательную роль, позволяя заменить в подчиненной форме коды экзаменов их названиями, и не мешает работе мастера. Однако при отборе полей для главной формы можно использовать поля только из одного источника данных. Так, попытка построить аналогичную форму, содержащую дополнительно информацию об адресах студентов (таблица **Общежитие)**, не приведет к желаемому результату. Мастер не сможет создать составную форму.

Тем не менее нужный результат можно получить, создав при помощи мастера составную форму без информации об адресах студентов и добавив в нее эту информацию позднее, после завершения работы мастера. Для этого, как и в случае изменения порядка сортировки, следует внести изменения в свойство **Источник записей** главной формы. В верхнюю часть окна конструктора запросов, содержащего таблицу **Студенты,** нужно добавить таблицу **Общежитие** и изменить тип связи этих таблиц (см. пример 5.16). Затем нужно включить в состав отбираемых полей все поля из таблицы **Студенты** (проще всего с помощью *), а также поля **Комната** и **Общежитие** из таблицы **Общежитие**. После сохранения запроса добавленные поля с информацией об адресах будут включены в список полей, доступных в форме, и могут быть размещены в ней непосредственно либо использованы для создания вычисляемого поля, содержащего адреса студентов.

В ряде случаев мастер автоматически синхронизирует главную и подчиненную формы, источниками которых являются не таблицы, а запросы, при условии, что между базовыми таблицами этих запросов установлена связь нужного типа.

Пример 6.7. Нужно создать составную форму, содержащую информацию об оценках по информатике и английскому языку девушек 9701 группы.

Для этого достаточно выполнить следующие действия.

1. Создать и сохранить запрос **Девушки 9701 группы**, содержащий записи из таблицы **Студенты** со сведениями о девушках 9701 группы.
2. Создать и сохранить запрос **Оценки по информатике и английскому языку,** основанный на таблицах **Сессия** и **Экзамены** и содержащий записи с оценками по нужным предметам.
3. Вызвать мастера **Создание форм** и отобрать из созданных запросов нужные поля.
4. Указать в качестве источника данных главной формы запрос **Девушки 9701 группы.**
5. Ответить на остальные вопросы мастера. Используя информацию о связях базовых таблиц **Студенты, Сессия** и **Экзамены**, он создаст форму, аналогичную приведенной в примере 6.6.

Эту форму можно также получить из составной формы примера 6.6 путем внесения соответствующих изменений в свойства **Источник записей** ее главной и подчиненной форм.

Иногда использовать мастера для создания составной формы сразу не удается, так как в БД отсутствуют нужные таблицы и/или не установлены связи между ними. В этом случае вызову мастера должно предшествовать

создание недостающих таблиц, а также установление их связей с другими таблицами в БД.

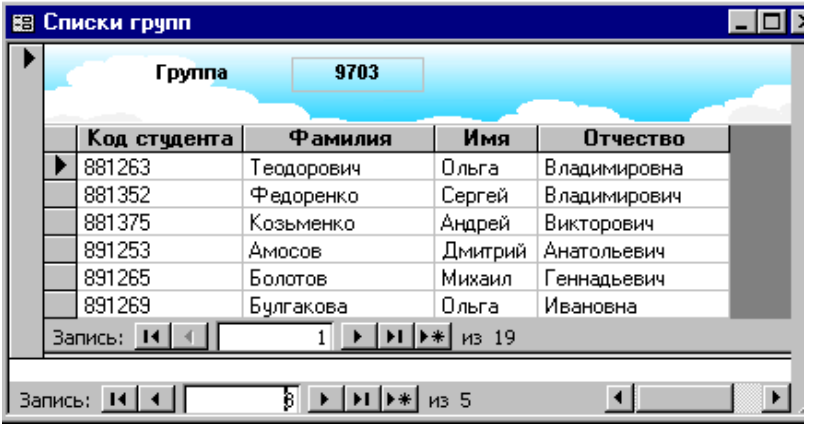


Рис. 6.25. Составная форма со списками учебных групп

Пример 6.8. Нужно создать составную форму **Списки групп**, которая выводит на экран информацию о студентах по учебным группам.

БД **Деканат** не содержит таблицу с номерами групп, которую можно было бы использовать в качестве базовой для главной формы. Поэтому для создания формы с помощью мастера следует выполнить следующие действия.

1. Создать таблицу **Группы**, содержащую единственное текстовое поле **Группа**, и сделать это поле ключевым.
2. Занести в эту таблицу номера учебных групп: 9701, 9702,…, 9705.
3. Установить связь «один ко многим» между таблицами **Группы** и **Студенты** по полю **Группа**. Таблица **Группы** будет находиться на стороне «один» этого отношения.
4. Вызвать мастера **Создание форм** и отобрать из таблицы **Группы** поле **Группа,** а из таблицы **Студенты** — поля **Код студента**, **Фамилия**, **Имя** и **Отчество.**
5. Указать в качестве главной таблицу **Группы** и нажать кнопку **Готово**.

Будет создана составная форма, главная форма которой содержит поле **Группа** с номером группы, а подчиненная — список студентов этой группы (см. рис. 5.25)

Мастер позволяет создавать составные формы с двумя уровнями вложенности. Проиллюстрируем эту возможность на примере создания формы, также предназначенной для просмотра итогов сдачи сессии.

Пример 6.9. Нужно создать составную форму **Оценки студентов по группам**, выводящую на экран информацию об оценках студентов по учебным группам и включающую две подчиненные формы. Главная форма должна содержать номер группы, первая подчиненная форма — список студентов текущей группы, а вторая подчиненная форма — оценки текущего студента из первой подчиненной формы.

Эта форма использует данные из четырех таблиц: созданной в предыдущем примере таблицы **Группы,** являющейся базовой для главной формы**,** а также таблиц **Студенты, Сессия** и **Экзамены,** информация из которых должна размещаться в подчиненных формах. Главная форма связана отношением «один ко многим» (**Группы** – **Студенты**) с первой подчиненной формой, которая в свою очередь связана отношением «один ко многим» со второй подчиненной формой (**Студенты** – **Сессия**).

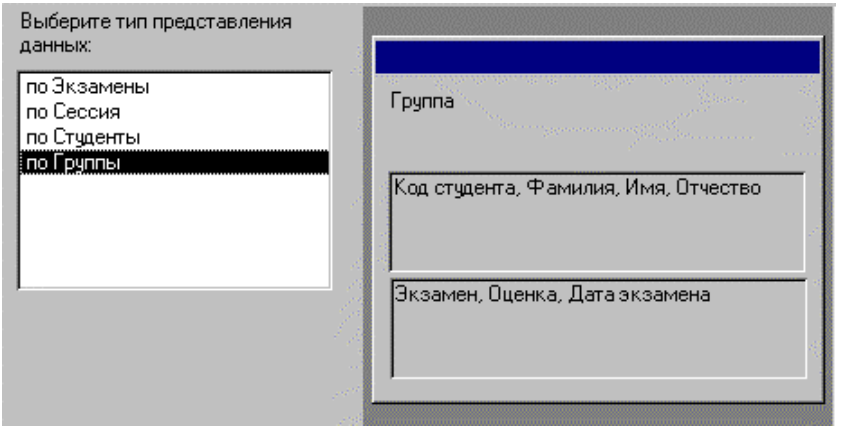


Рис. 6.26. Создание формы с двумя уровнями вложенности

Для создания формы следует вызвать мастера и выполнить следующие действия.

1. В первом диалоговом окне этого мастера отобрать требуемые поля:
   - из таблицы **Группы** — поле **Группа;**
   - из таблицы **Студенты** — поля **Код студента**, **Фамилия**, **Имя** и **Отчество**;
   - из таблицы **Сессия —** поля **Оценка** и **Дата экзамена**;
   - из таблицы **Экзамены —** поле **Экзамен**.
2. В следующем окне нужно указать, что источником данных главной формы является таблица **Группы.** Проанализировав связи между таблицами, Access предложит разместить отобранные поля не в одной, а в двух подчиненных формах. В первую он поместит поля таблицы

Студенты, а во вторую — поля из таблицы **Сессия** и **Экзамены** (см. рис. 6.26).

3. После получения ответов на остальные вопросы мастер создаст форму, содержащую два ЭУ **Подчиненная форма/отчет,** а также две формы, которые будут выводиться в этих ЭУ.

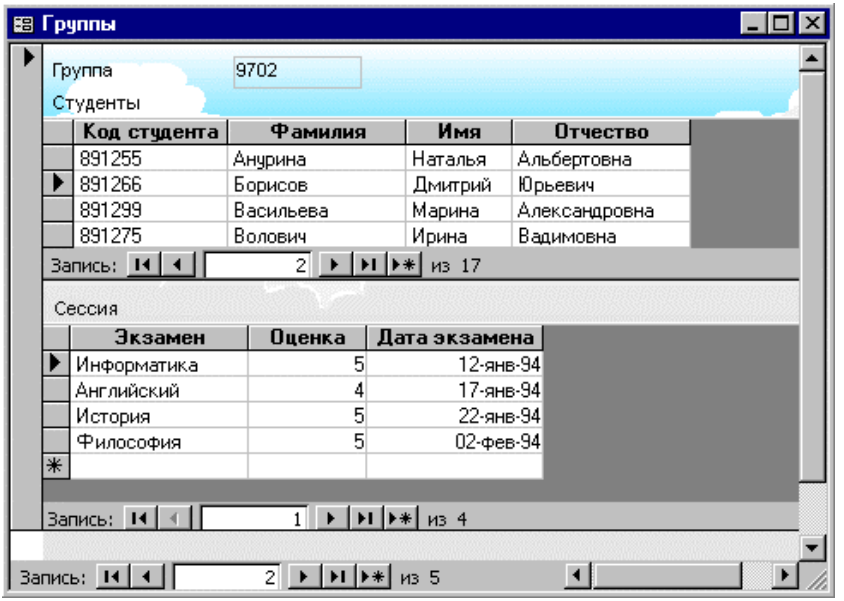


Рис. 6.27. Просмотр формы с двумя уровнями вложенности

Если в главной форме будет осуществлен переход от одной записи к другой, то в подчиненных формах автоматически обновится информация. В первой подчиненной форме будет выведен список студентов текущей группы, а во второй — оценки первого студента из этого списка. Соответственно переход в первой подчиненной форме от одной записи к другой приведет к обновлению информации во второй подчиненной форме: на экране всегда будут находиться оценки текущего студента.

Для синхронизации информации между главной и первой подчиненной формами Access использует поле связи **Группа**, а между первой и второй подчиненными формами — поле связи **Код студента** и процедуру обработки события **Текущая запись** в первой подчиненной форме, которая запускается при переходе от одной записи к другой.

6.6.2. Использование ЭУ *Подчиненная форма/отчет*

Хотя вышеописанный способ создания составной формы с помощью мастера **Создание форм** является самым простым, в ряде случаев он все же не позволяет достичь желаемого результата. Иногда установление связи между базовыми таблицами, необходимое для правильной работы мастера, по тем или иным причинам нежелательно или вообще невозможно. Но даже если все связи установлены, далеко не всегда мастер оказывается на высоте. Типичный пример — ситуация, когда в главной форме нужно поместить поля из нескольких таблиц (см. пример 6.6).

От всех недостатков и ограничений первого способа свободен способ создания составной формы, основанный на внедрении в главную форму подчиненной при помощи ЭУ **Подчиненная форма/отчет** с последующим связыванием этих форм. В этом случае также удобно использовать специального мастера, действуя по следующей схеме.

1. Определить состав таблиц/запросов и список их полей, нужных для создания как главной, так и подчиненной формы. Этот список должен включать поля, непосредственно размещаемые в форме или входящие в состав выражений вычисляемых полей, а также поля, используемые для связи между формами. Поля связи можно не размещать в формах, но они обязательно должны быть доступными в каждой из них.
2. Создать, если это необходимо, и сохранить запросы, содержащие отобранные на первом шаге поля. В этих запросах можно задать условия отбора нужной информации и порядок ее сортировки.
3. Создать главную форму на базе таблицы или сохраненного запроса и установить для нее режим по умолчанию *Простая форма*.
4. Включить, если она не включена, кнопку **Мастера**  на панели элементов. Затем нажать кнопку **Подчиненная форма/отчет**  на этой же панели, установить указатель мыши на то место в главной форме, куда нужно поместить подчиненную форму, и нажать левую кнопку. На экране появится первое окно диалога мастера **Создание подчиненных форм и отчетов.** Так как подчиненная форма создается на базе таблицы или запроса, нужно щелкнуть по кнопке **Далее** и перейти ко второму шагу.
5. Во втором окне диалога нужно выбрать таблицу (запрос), являющуюся источником данных для подчиненной формы, и перебросить нужные поля из списка *Доступные поля* в список *Выбранные поля*. В список выбранных полей должны быть обязательно включены поля, используемые для связи.

6. Затем мастер предлагает установить связь между формами. Можно воспользоваться списком возможных связей или самостоятельно выбрать поля связи между главной и подчиненной формами.
7. В последнем окне диалога задается имя подчиненной формы, которая сохраняется Access как отдельная форма. Для проверки полученного результата нужно перейти в режим формы.

Пример 6.10. Нужно создать составную форму **Итоги сессии по группам.** Главная форма должна содержать номер группы, а подчиненная — список студентов этой группы и их оценки. Форма также должна включать итоговую информацию: средние баллы в группах по каждому предмету.

1. Начнем с создания источника данных главной формы. Она должна содержать номера групп из поля **Группа** таблицы **Студенты**. Однако использовать саму таблицу в качестве источника данных нельзя, так как значения в поле **Группа** дублируются. Поэтому следует сначала создать форму, не указывая источника ее данных. Затем открыть окно ее свойств, щелкнуть по свойству **Источник записей**, а потом — по кнопке построителя запросов. Откроется окно конструктора запросов, в котором нужно создать и сохранить запрос к таблице **Студенты**, отбирающий уникальные номера групп. Для этого проще всего включить в бланк запроса поле **Группа** и щелкнуть по кнопке  (см. п. 5.5.1). После выхода из построителя запросов в свойстве **Источник записей** появится инструкция SQL: *SELECT Студенты.Группа FROM Студенты GROUP BY Студенты.Группа;*
2. Источник данных для подчиненной формы у нас фактически уже создан — это перекрестный запрос **Итоги сессии на курсе** (см. пример 5.30). Однако для использования в подчиненной форме его нужно немного изменить: вызвать в режиме конструктора список свойств и перечислить в свойстве **Заголовки столбцов** заголовки столбцов (названия экзаменов). Сохраним новый запрос под тем же именем.
3. Поместим поле **Группа** в главной форме и с помощью мыши увеличим размеры области данных. Далее щелкнем по кнопке **Подчиненная форма/отчет**  на панели элементов и выделим мышью место в главной форме, которое должна занимать подчиненная форма. Затем ответим на вопросы мастера **Создание подчиненных форм и отчетов.** Так как форма создается на базе запроса, щелкнем по кнопке **Далее**, а на следующем шаге выберем запрос **Итоги сессии на курсе** и

отберем все его поля. Затем требуется указать поля связи между формами. Можно выбрать связь из списка, но мы определим ее сами, указав в качестве поля связи поле **Группа** (см. рис. 6.28). Мастер дал подчиненной форме название **подчиненная форма Итоги сессии на курсе** и на этом его работа завершена.

Дальнейшую разработку формы будем проводить вручную. Проделаем следующие действия:

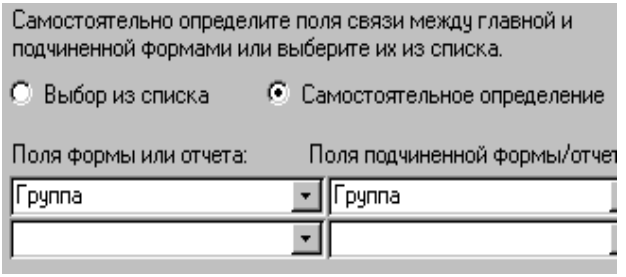


Рис. 6.28. Определение полей связи

1. С помощью команд **Вид** и затем **Заголовок/примечание формы** добавим в главную форму раздел **Заголовок формы.** В заголовок добавим надпись и введем в нее текст *Итоги сессии по группам,* установив размер шрифта — 18 и цвет текста — белый. Затем выделим надпись, скопируем ее в буфер обмена и вставим в заголовок формы. Изменим цвет текста у надписи-копии на темный и расположим ее чуть правее исходной надписи, создав тем самым эффект тени у букв.
2. Удалим созданную мастером надпись к подчиненной форме. Затем выделим поле **Группа** и его надпись. Установим для них полужирный шрифт 10, выравнивание по центру, размер — по размеру данных и разместим в центре формы.

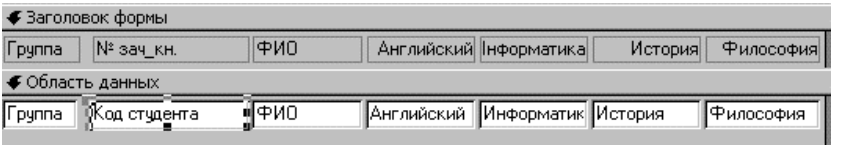


Рис. 6.29. Созданная мастером подчиненная форма

3. Перейдем в режим формы и скроем в подчиненной форме столбец **Группа.** Для этого щелкнем по этому столбцу и выполним команды **Формат/Скрыть столбцы.** Отметим, что для отображения скрытого столбца нужно открыть подчиненную форму в режиме таблицы и выполнить команды **Формат/Отобразить столбцы**. Затем настроим

ширину остальных столбцов, изменив при необходимости размеры подчиненной формы.

4. Чтобы изменить название столбца **Код студента** на **№ зач.кн.,** вызовем в режиме конструктора подчиненную форму. Как видно на рис. 6.29, мастер поместил надписи полей в заголовке формы, а сами поля — в области данных, причем в качестве заголовков в составной форме появляются именно названия полей, а не их надписей. Откроем свойства поля **Код студента** и заменим его старое имя, появляющееся в качестве заголовка столбца, новым именем**.** После закрытия формы внесенные изменения будут доступны для просмотра.

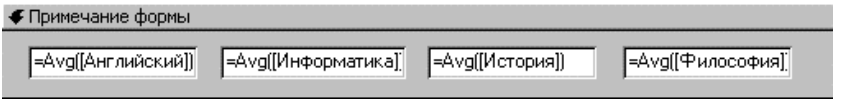


Рис. 6.30. Добавление итоговых полей в подчиненную форму

5. Последний этап создания формы — включение в нее итоговой информации. Сначала нужно добавить в раздел **Примечание формы** подчиненной формы четыре вычисляемых поля, содержащие средние баллы по экзаменам. Для этого следует щелкнуть мышью по ЭУ **Поле**, а затем — в области примечания. Access вставит в примечание формы свободное поле с надписью. Надпись нужно удалить, далее вызвать свойства поля и ввести в качестве значения свойства **Данные:** *=Avg([Английский])*. Затем с помощью операций копирования и вставки создадим три копии этого поля и внесем в их свойство **Данные** соответствующие изменения в названия итоговых полей (см. рис. 6.30). Пусть Access дал этим полям следующие имена: *Поле 14, Поле 15, Поле 16* и *Поле 17*. Они будут использованы в ссылках на значения итоговых полей из главной формы.

   После создания итоговых полей в подчиненной форме ее следует закрыть и вызвать в режиме конструктора главную форму. В ней создадим вычисляемое поле без надписи и введем в его свойство **Данные** ссылку на значение итогового поля в подчиненной форме в виде

   =[подчиненная форма Итоги сессии на курсе]![Поле 14]

   Установим значения его свойств **Формат поля** – *Фиксированный* и **Число десятичных знаков** – *2*. Затем создадим три копии этого поля, изменим в свойстве **Данные** их ссылки на поля подчиненной формы и разместим эти поля под соответствующими столбцами с оценками.

Добавим надпись *Средние баллы по предметам*. При открытии главной формы в режиме формы в добавленных полях появятся значения средних баллов текущей группы по предметам (см. рис. 6.31).

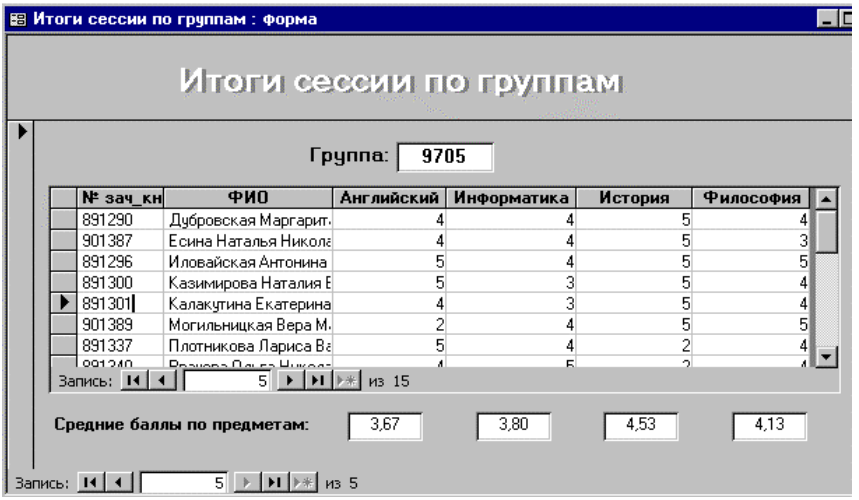


Рис. 6.31. Составная форма с итоговыми полями

6.6.3. Добавление готовой подчиненной формы

Можно создать составную форму путем «перетаскивания» с помощью мыши в главную форму уже существующей формы в качестве подчиненной. Для этого нужно выполнить следующие действия:

1. Открыть главную форму в режиме конструктора.
2. Вернуться в окно БД, выбрать подчиненную форму и «перетащить» ее в окно главной формы. Access создаст ЭУ **Подчиненная форма/отчет.**
3. Если это необходимо, следует изменить размеры и положение подчиненной формы.
4. Перейти в режим формы для просмотра полученного результата.

Access автоматически устанавливает связь между формами, если выполнено одно из следующих условий:

- Источники данных для обеих форм — связанные таблицы. В качестве полей связи будут использованы поля связи этих таблиц.
- Таблица – источник данных главной формы имеет ключевое поле, а таблица – источник данных подчиненной формы имеет одноименное поле с таким же или совместимым типом данных. В качестве полей связи будут использованы эти поля.

Если формы базируются на запросах, то этим условиям должны удовлетворять базовые таблицы для запросов.

Если Access не смог установить связь между формами, ее следует определить самостоятельно. Для этого нужно, находясь в главной форме, открытой в режиме конструктора, открыть бланк свойств ЭУ подчиненной формы и в свойстве **Подчиненные поля** задать имя поля (полей) связи подчиненной формы. В свойстве **Основные поля** задается имя поля (полей) связи главной формы. Тип данных и размер полей связи должны быть совместимыми. Если полей связи несколько, их имена нужно перечислить в одинаковом порядке, разделяя точками с запятой.

6.6.4. Использование для связи других ЭУ

Во всех рассмотренных выше примерах полями связи главной формы служили поля ее базовой таблицы/запроса. Их имена задавались в свойстве **Основные поля** подчиненной формы, а значения использовались Access для отбора выводимых на экран записей в подчиненной форме. Но в качестве поля связи в свойстве **Основные поля** можно указать также имя другого ЭУ, размещенного в главной форме. Обычно для этой цели используется поле со списком, реже — список или группа переключателей. Для отбора записей можно использовать несколько ЭУ.

В этом случае главную форму не нужно связывать с каким-либо источником данных, так как отбор записей в подчиненной форме производится на основе совпадения значений ее поля (полей) связи с выбранными значениями ЭУ.

Пример 6.11. Для просмотра заказов покупателей в БД **Книги** нужно создать составную форму **Заказы покупателей**, в которой главная форма содержит поле со списком покупателей (фамилия, имя, город), а подчиненная форма — список заказов текущего покупателя (автор и название книги, дата отправки заказа, количество экземпляров). Список покупателей должен быть отсортирован по фамилиям, а список заказов — по дате заказа.

1. Связь между полем со списком и подчиненной формой должна осуществляться по полю **Код покупателя.** Чтобы Access рассматривал его значения в качестве значений поля со списком, оно должно быть указано в свойстве списка **Присоединенный столбец**. Это поле также должно быть доступно в подчиненной форме.
2. Начнем с создания источника данных для поля со списком. На базе таблицы **Покупатели** создадим и сохраним запрос **Список покупателей**, содержащий четыре поля: **Код покупателя**, **Фамилия, Имя** и **Город** с сортировкой по возрастанию по полю **Фамилия**.

3. Создадим пустую форму, не связывая ее с каким-либо источником данных, и добавим в нее поле со списком. Сообщим мастеру, что поле со списком использует значения из запроса **Список покупателей,** и отберем все поля этого запроса. Затем укажем, что в качестве значений поля со списком (присоединенного столбца) нужно взять поле **Код покупателя** и зададим подпись *Покупатель***.**
4. Добавим в форму **ЭУ Подчиненная форма/отчет** и отберем для подчиненной формы следующие поля:
   - **Автор** и **Название** из таблицы **Книги**;
   - **Код покупателя**, **Дата отправки** и **Количество** из таблицы **Заказы**.

   Затем щелкнем по кнопке **Готово**.
5. Access добавит подчиненную форму, но ее связь с полем со списком нужно установить самостоятельно. Для этого двойным щелчком мыши по полю со списком откроем окно его свойств и скопируем в буфер обмена значение свойства **Имя**: *ПолеСоСписком0.* Затем щелкнем по подчиненной форме: откроется окно ее свойств. Вставим из буфера обмена в качестве значения его свойства **Основные поля** — имя ЭУ **поле со списком**, а в качестве значения свойства **Подчиненные поля** — *Код покупателя***.**

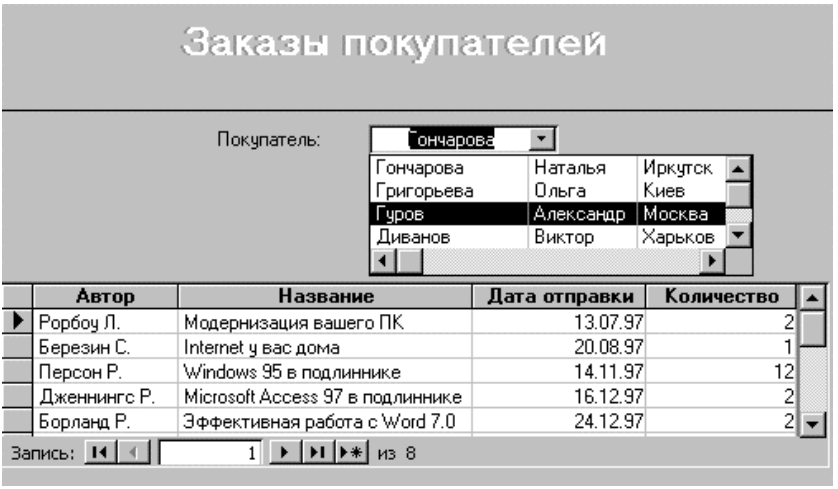


Рис. 6.32. Поле со списком в составной форме

6. Перейдем в режим формы и проверим правильность установки связи. Мы видим, что после выбора покупателя из списка в подчиненной форме содержатся именно его заказы. Об этом говорит совпадение значений кода выбранного покупателя со значениями в столбце **Код**

покупателя подчиненной формы. Следовательно, связь установлена верно.

7. Вернемся в режим конструктора и добавим в главную форму заголовок. Разместим в нем надпись *Заказы покупателей* и отформатируем ее. Затем вызовем свойства поля со списком и в свойстве **Ширина столбцов** укажем в качестве значения ширины первого столбца 0. В результате этой процедуры в закрытом поле будет выводиться не код, а фамилия покупателя. При открытии поля список будет также содержать имя и город проживания покупателя.
8. Перейдя в режим формы, скроем в подчиненной форме столбец **Код покупателя** и настроим порядок и ширину остальных столбцов. Щелкнем по столбцу **Дата отправки**, а затем по кнопке **Сортировка по убыванию**. Сохраним созданную форму.

Пример 6.12. Создадим форму для просмотра распределения числа различных оценок по данному экзамену по группам. Главная форма должна содержать группу из четырех переключателей, каждый из которых соответствует одному из экзаменов. После выбора переключателя в подчиненной форме должна появиться таблица с распределением оценок между группами по выбранному экзамену.

1. Для связи между группой переключателей и подчиненной формой будем использовать значения кодов экзаменов. Разработку формы начнем с размещения группы переключателей в главной форме. Для этого создадим пустую форму и щелкнем по кнопке **Группа переключателей**  на панели элементов, а затем по тому месту в форме, где мы хотим поместить эту группу. Начнет работу мастер **Создание группы переключателей**. Сначала он предложит создать подписи к переключателям. Введем в качестве подписей названия экзаменов: *Английский*, *Информатика*, *История* и *Философия*. На следующем шаге в качестве переключателя, используемого по умолчанию, выберем: *Информатика.* Затем укажем значения для каждого переключателя. При выборе переключателя его значение будет являться значением всей группы. Поэтому в качестве значений переключателей нужно ввести коды соответствующих экзаменов*: 2, 1, 3, 4*. Далее мастер предлагает выбрать тип ЭУ и оформление. Примем используемые по умолчанию тип *переключатели* и оформление *вдавленное*. В качестве подписи к группе введем текст: *Выберите экзамен*. На этом работа мастера завершена.
2. На базе таблиц **Студенты** и **Сессия** создадим перекрестный запрос **Распределение оценок**, содержащий в качестве заголовков строк поля **Группа** и **Код экзамена**, а в качестве заголовка столбцов — поле

Оценки. Значения итоговой таблицы — число полученных оценок (см. рис. 6.33). В свойстве запроса **Заголовки столбцов** нужно также перечислить названия столбцов*: 5; 4; 3; 2*.

Группа	Код экзамена	Оценка	Оценка
Студенты	Сессия	Сессия	Сессия
Группировка	Группировка	Группировка	Count
Заголовки строк	Заголовки строк	Заголовки столбцов	Значение

Рис. 6.33. Бланк запроса ***Распределение оценок***

3. Добавим в форму ЭУ **Подчиненная форма/отчет,** укажем, что он основан на созданном выше запросе и отберем все поля из запроса.
4. Для установления связи между группой переключателей и подчиненной формой в свойстве **Основные поля** укажем имя группы *Группа0*, а в свойстве **Подчиненные поля** — имя поля **Код экзамена**. После настройки созданная форма имеет вид, представленный на рис. 6.34.

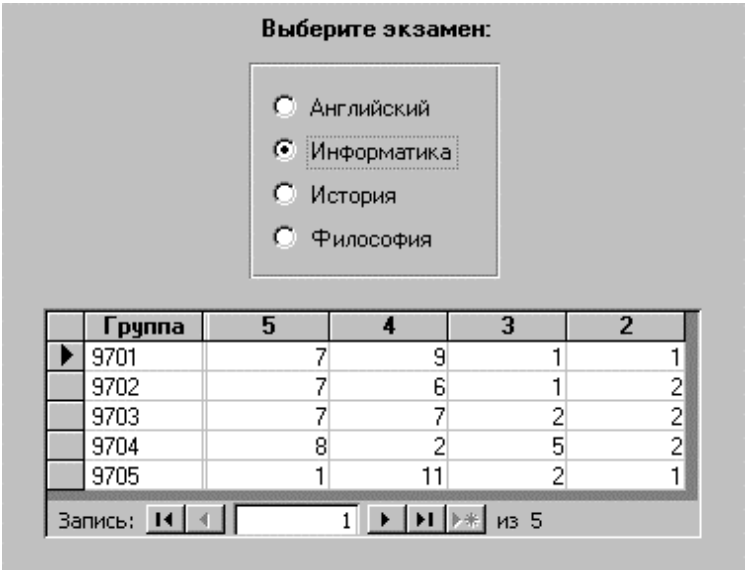


Рис. 6.34. Форма Распределение оценок по экзамену

Глава 7. Отчеты

7.1. Основные понятия

Отчеты — форма представления информации для использования и распространения. Отчеты — итоговые документы для лиц, которым была предназначена создаваемая БД. Если форма — документ разработчика и лиц, работающих с информацией в БД, то отчеты — инструмент «хозяев» БД, позволяющий им в нужный момент иметь информацию для личного использования или передачи другим лицам. Отчет нужен для подведения итогов деятельности за период подсчетов итоговых сумм и т.д.

Структура отчета (см. рис. 7.1) напоминает структуру формы за одним существенным исключением — возможностью добавления нескольких пар новых разделов, если возникает потребность группировки данных по каким-либо признакам.

Рис. 7.1. Структура отчета

Возможность группирования данных, - это, пожалуй, главное отличие создания отчета от создания формы. Как реализовать группирование, мы подробно рассмотрим ниже, а пока кратко рассмотрим саму идею группирования и ее влияние на структуру разделов.

Предположим, что мы строим отчет по таблице, описывающей реализацию товаров различным клиентам. Можно составить отчет по разным «срезам» такой таблицы. Например, составить отчет, характеризующий заказы каждого клиента. В этом случае основа

группирования данных — клиенты. Все заказы разбиваются на группы, относящиеся к одному клиенту.

Другой вариант — рассмотреть, как реализовывались товары различных видов. В этом случае основа группирования — отдельный товар, а все данные о заказах можно разбить на группы, относящиеся к конкретному товару. Группы могут быть вложенными. Например, в первом случае, внутри группирования по клиентам, можно сгруппировать заказы каждого клиента по отдельным товарам. Каждое группирование сопровождается, как правило, появлением двух разделов — заголовка группы и примечания группы. В заголовке обычно указывается информация о поле — источнике группирования (клиенты, товары) и др. В области примечания можно подвести итоги группирования — число заказов каждого клиента, сумму стоимости всех заказов каждого клиента, число заказов каждого товара и т. д.

Если группы вложенные, можно подвести промежуточные итоги, т. е. число заказов конкретного товара, сделанных клиентом. Общее число вложений групп — до 10. Свойства разделов отчета совпадают со свойствами разделов формы, хотя у форм и отчетов существуют различные режимы. Напомним, что форма могла находиться в трех режимах: конструктора, работы (формы) и предварительного просмотра.

У отчета тоже может быть три режима, причем два из них такие же: режим предварительного просмотра и режим конструктора. Кроме того, отчет может находиться в режиме *образца*. Этот режим похож на режим предварительного просмотра, но отличается от него тем, что в режиме образца показывается не весь отчет, а только его часть, с целью оценки того, как будет выглядеть весь отчет. Режим, подобный режиму формы, естественно отсутствует, так как с отчетом не работают как с формой или запросом.

Некоторые разделы отчета, такие как колонтитулы и примечания, имеют дополнительные свойства, более тонкие, чем у аналогичных разделов форм. Так, заголовок отчета может быть напечатан отдельно на первой странице, без колонтитулов.

Надо отметить, что в отчете дублируются и многие другие свойства различных объектов, изучавшихся нами в формах.

В отчетах, как и в формах, размещаются элементы управления: поля, надписи и т. д. Как и в формах, они могут быть связанными, свободными и вычисляемыми в зависимости от источника информации, реализуемой в элементе управления.

7.2. Создание отчета

Отчеты, как и формы, можно создавать с помощью мастеров или самостоятельно. Источником данных для отчета также являются таблицы и запросы. Если в отчете требуется представить данные из разных таблиц, имеет смысл предварительно создать многотабличный запрос, а затем строить отчет на его основе.

Чтобы создать отчет, нужно в окне БД перейти на вкладыш **Отчеты** и нажать кнопку **Создать**. Появится окно диалога **Новый отчет**. Оно аналогично окну создания формы. Ваши действия начинаются с выбора таблицы/запроса.

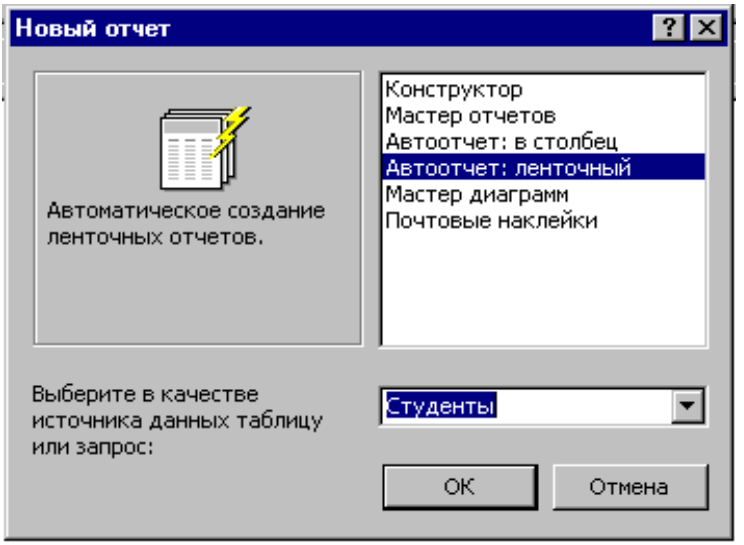


Рис. 7.1. Окно диалога ***Новый отчет***

Предусмотрено автоматическое создание двух типов отчетов: **ленточного** и **в один столбец**.

В этих отчетах имеется заголовок, содержащий имя отчета, совпадающее с названием таблицы или запроса, и дата создания отчета. Нижний колонтитул содержит номер страницы. В области данных содержатся все поля таблицы (запроса), расположенные в один столбец или в табличном виде (в ленточном отчете). На одной странице реализуется

несколько записей таблицы (запроса). Пример ленточного отчета приведен на рис. 7.2.

Студенты

Фамилия	Имя	Отчество	Дата рождения	Группа
Федоренко	Сергей	Владимирович	13.07.74	9703
Шкрабин	Дмитрий	Леонидович	20.06.72	9705
Козьменко	Андрей	Викторович	07.08.73	9703
Амосов	Дмитрий	Анатольевич	26.09.74	9703
Болотов	Константин	Рудольфович	24.07.74	9704
Болотов	Михаил	Геннадьевич	02.11.72	9703
Борисов	Дмитрий	Юрьевич	25.01.72	9702

Рис. 7.2. Пример ленточного отчета

7.2.1. Создание отчета с помощью Мастера отчетов

Использование мастера по разработке отчетов является быстрым и защищенным от ошибок способом создания нового отчета. Отчет, созданный мастером, можно использовать в том виде, в котором он был создан, или улучшить путем внесения изменений.

Чтобы создать отчет с помощью мастера по разработке отчетов, надо:

1. Находясь в окне базы данных, указать на корешок **Отчет** и нажать кнопку мыши (или выбрать **Отчеты** в меню **Вид**).
2. Нажмите кнопку **Создать** (или нажать кнопку **Новый объект** на панели инструментов, а затем выбрать **Отчет**. На экране появится окно диалога **Новый отчет**.
3. Из списка выбрать таблицу/запрос, содержащий данные, которые следует представить в отчете.
4. Выбрать способ создания отчета — **Мастер отчетов**. На экране появится первое окно диалога **Создание отчета**.
5. Переслать необходимые для отчета поля из списка **Доступных** полей в список **Выбранных**.
6. В следующем окне задать необходимые уровни группировки.
7. Данные в отчете можно отсортировать. Можно задать сортировку до четырех уровней, по убыванию или возрастанию.
8. На следующем этапе задается макет отчета. Он может быть табличный, в один столбец или выровненный. Вид выбранного макета появляется на экране. Здесь же можно задать ориентацию данных в отчете —

книжную или альбомную. Если в отчете выводится много полей, можно задать настройку ширины полей для размещения их на странице. Но при этом в ходе просмотра образца отчета можно ожидать что либо в заголовках полей, либо в информации будут выводиться неполные данные. Придется эти поля форматировать в режиме конструктора.

9. В следующем окне диалога выбирается стиль оформления. Он может быть выделенным, компактным, простым, строгим, формальным или черно-белым. Выбранный стиль отображается на экране.
10. В последнем окне нужно задать имя отчета.

Создание отчета мастером закончено. Вы можете просмотреть образец отчета. Для этого можно нажать кнопку  или выполнить пункт **Предварительный просмотр** меню **Файл**. Если отчет нужно несколько изменить, воспользуйтесь **Конструктором**.

7.2.2. Создание отчета с помощью Конструктора

Пользователь может начать с пустого отчета и самостоятельно разместить в нем все необходимые поля, надписи и другие элементы управления.

Для создания отчета без помощи мастера нужно, находясь в окне БД, щелкнуть по корешку **Отчет**, а затем по кнопке **Создать**.

На экране появится окно диалога **Новый отчет**. Из списка **Выбор таблицы/запроса** выберите таблицу/запрос, содержащую данные, которые следует представить в отчете. Если отчет не должен содержать данные, не делайте выбор из этого списка. Задайте способ создания отчета — **Конструктор**.

На экране появится окно конструктора отчетов. Чтобы вывести на экран список полей базовой таблицы (или запроса), нажмите кнопку **Список полей** на панели инструментов.

Рабочие экраны при создании отчета и формы практически совпадают. Мы не будем останавливаться на инструментарии, рассмотренном при создании форм (это создание полей, надписей, вычисляемых полей и т. д.). Например, для печати эмблемы фирмы можно в заголовке отчета создать **ЭУ Рисунок** или **Свободная рамка объекта**. Все ЭУ создаются в разделах отчета теми же способами, что и в формах. Остановимся на принципиальном отличии отчета от формы — создании групп.

7.3. Сортировка и группирование

Обычно записи в отчете требуется расположить в определенном порядке. В меню Вид имеется подкоманда **Сортировка и группировка.** С помощью этой команды можно просто провести сортировку отчетных данных или можно определить группирование данных. Сортировка задается по определенному полю (наиболее часто используемый способ) или по выражению, т. е. вычисляемому полю, либо по первой букве символьного поля. Полей сортировки может быть до 10. При выборе команды **Сортировка и группировка** на экране появляется окно следующего вида (см. рис. 7.3).

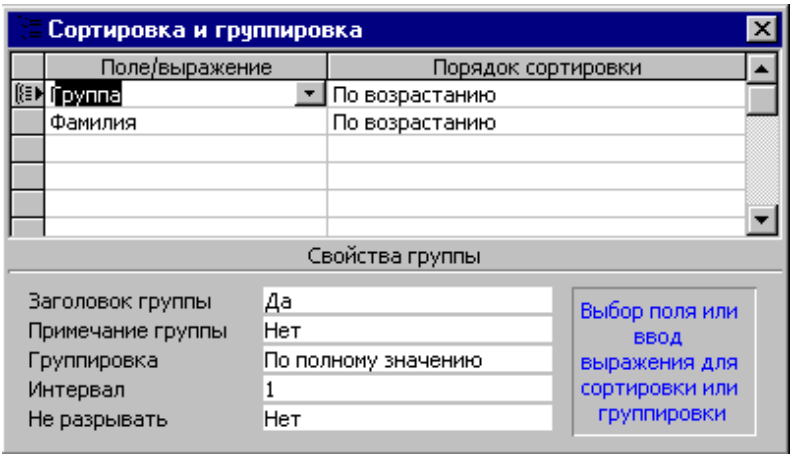


Рис.7.3. Окно Сортировка и группировка

Для задания просто сортировки не нужно обращаться к свойствам групп. Если же мы хотим сгруппировать данные по какому либо полю, нужно указать это поле в верхней части окна и установить «**Да**» в заголовке группы или примечании. Если нужно создать вложенную группировку, нужно внести несколько полей в верхнюю часть окна и открыть заголовки (примечания) в нижней части. Окно конструктора при этом будет содержать следующие разделы:

Заголовок отчета
Верхний колонтитул
Заголовок первой группы
Заголовок второй группы
.....
Область данных
....
Примечание второй группы
Примечание первой группы
Нижний колонтитул
Примечание отчета

Для того чтобы отчет выглядел хорошо, нужно разместить элементы управления в соответствующих разделах. Например, так, как это выглядит на рис. 7.4.

Как видно на этом рисунке, в бланке отчета отсутствует раздел примечаний группы **Курс**, т.к. в окне **Сортировка и группировка** в свойствах этой группы было задано значение «**Нет**» в строке **Примечание группы**.

После задания группировок по полям **Курс** и **Группа** в разделе **Примечания** группы **Группа** добавлено два ЭУ — надпись **В группе** и вычисляемое поле с выражением =*Count(Фамилия*). Кроме того надписи к полям, расположенным в области данных, обычно размещаются в области верхнего колонтитула. В этом отчете поля **Курс** и **Группа** размещены в областях заголовков групп вместе со своими надписями.

При создании групп важно правильно задать значения или диапазон значений, с которых начинается новая группа. Эти значения задаются в свойствах **Группировка** и **Интервал** в окне **Сортировка и группировка**. Свойство **Группировка** указывает, как должны быть сгруппированы данные. По полному значению можно группировать поля: текстовые, денежные, числовые, счетчики, даты/время. Текстовые поля, кроме того, можно группировать по определенному числу первых символов. Счетчики, денежные, числовые поля и даты/время можно группировать по диапазону значений.

В свойстве **Группировка** диапазон значений можно выбрать из списка, который предлагается для каждого типа поля группировки. Например, если поле группировки имеет тип дата/время, то будет предложен следующий список интервалов: по полному значению, по годам, по кварталам, по месяцам, по неделям, по дням, по часам, по минутам. Для счетчиков, денежных, числовых полей можно кроме группировки по

полному значению задавать группировку по интервалу. При этом нужно задавать значение в свойстве Интервал любым числом, допустимым для группируемого поля.

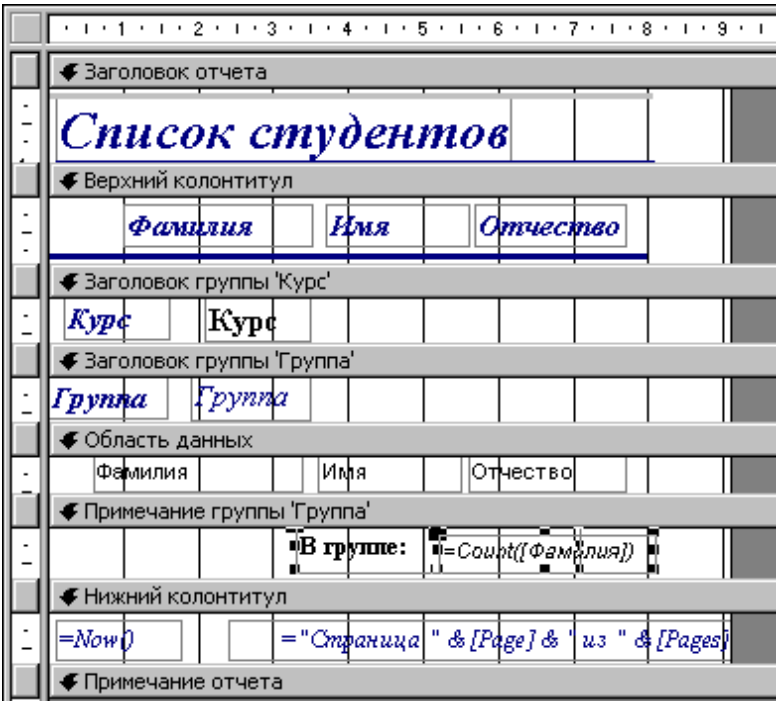


Рис. 7.4. Окно ***Конструктора отчета***

Например, для поля типа *Счетчик* свойство **Группировка** имеет значение *Интервал*, а в свойстве **Интервал** задано значение 5. Записи в отчете будут сгруппированы в следующие группы: 0-4,5-9,10-14 и т. д. Или для группового поля **Дата продажи** задано свойство **Группировка** — *По месяцам*, а значение в свойстве **Интервал** – 6. Продажи будут сгруппированы в отчете по интервалам в шесть месяцев.

Если в режиме конструирования отчета возникает необходимость переноса уже созданного ЭУ из одного раздела в другой, перемещение нужно начать тогда, когда указатель мыши примет вид раскрытой руки, либо можно выполнить это действие через буфер обмена.

Размеры разделов изменяются при перемещении их нижних границ, когда указатель принимает вид двунаправленной стрелки.

7.4. Режимы окна отчета

Окно отчета может находиться в одном из трех режимов: режиме конструктора, режиме просмотра образца и режиме предварительного просмотра:

Режим конструктора предназначен для создания новых и изменения существующих отчетов;

Режим просмотра образца предназначен для предварительной оценки правильности шрифтового оформления и расположения элементов управления в отчете. В этом режиме изображаются все разделы отчета, а также несколько записей в области данных. При этом выполняется сортировка и группирование данных, однако не используются ни условия отбора, ни объединение таблиц, определенные в базовом запросе.

В режиме предварительного просмотра отчет выглядит на экране так, как он выглядел бы, будучи напечатан. Напечатанный отчет может выглядеть иначе, чем на экране в режиме предварительного просмотра, если для его оформления используются немасштабируемые шрифты. Масштабируемыми являются шрифты Tгуе Type и некоторые другие шрифты.

При просмотре отчета в режиме просмотра образца или в режиме предварительного просмотра можно изменять масштаб изображения отчета.

Чтобы перейти в режим просмотра образца из режима конструктора, выберите **Образец** в меню **Вид** (или нажмите кнопку  **Образец** на панели инструментов). Для возврата в режим конструктора нажмите кнопку **Закрыть** на панели инструментов в окне образца.

Чтобы перейти в режим предварительного просмотра из режима конструктора, выберите **Предварительный просмотр** в меню **Файл** (или нажмите кнопку  **Предварительный просмотр** на панели инструментов). Для возврата в режим конструктора нажмите кнопку **Закрыть** на панели инструментов в окне предварительного просмотра.

Чтобы перейти в режим предварительного просмотра из окна базы данных, находясь в окне базы данных, укажите на корешок Отчет и нажмите кнопку мыши, а затем выберите нужный отчет, после чего выберите **Предварительный просмотр** в меню **Файл** (или нажмите кнопку **Просмотр** на панели инструментов).

7.5. Печать отчета

Чтобы напечатать готовый отчет, нужно выполнить команду **Печать** из меню **Файл**. Напечатанный отчет может содержать пустые страницы. Как правило, это является следствием того, что размеры отчета превышают размер бумаги, которая используется для печати.

Чтобы в отчете не было пустых страниц, сумма полной ширины отчета с шириной правого и левого полей не должна превышать ширину бумаги, указанную при настройке печати (смотрите следующую формулу):

Ширина отчета + Левое поле + Правое поле <= Ширина бумаги

Более подробные сведения об устранении пустых страниц можно найти в справочной системе по ключевому слову печать.

Если отчет содержит слишком много пустого пространства вокруг разделов и элементов управления, то его можно уменьшить, изменив размеры и определив некоторые свойства разделов и элементов управления. При печати отчета очень важно правильно задать свойства отчета, его разделов и некоторых ЭУ. Свойства отчета вызываются двойным щелчком мыши по прямоугольнику на пересечении горизонтальной и вертикальной линеек, либо из контекстного меню, вызванного, когда указатель мыши находился в серой области бланка отчета вне разделов. Свойства разделов вызываются двойным щелчком мыши по серой полосе с названием раздела. Свойства любого ЭУ вызываются двойным щелчком мыши по этому ЭУ.

Для печати всего отчета важны свойства **Верхний колонтитул** и **Нижний колонтитул**. Возможные значения:

- *Все страницы* — колонтитулы печатаются на всех страницах;
- *Без заголовка* — колонтитулы не печатаются на тех страницах, где печатался заголовок отчета;
- *Без примечания* — колонтитулы не печатаются на тех страницах, где печаталось примечание отчета;
- *Без заголовка/примечания* — колонтитулы не печатаются на тех страницах, где печатались заголовок или примечание отчета.

Для печати разделов отчета важны свойства **Конец страницы** и **Не разрывать**. Свойство **Конец страницы** может принимать значения:

- *Отсутствует* — расставлять страницы не обращая внимание на начало/конец раздела;
- *До раздела* — поставить конец страницы перед началом раздела;
- *После раздела* — поставить конец страницы после конца раздела;
- *До и после раздела* — поставить конец страницы до начала и после конца раздела.

Для того чтобы в приведенном в качестве примера отчете **Список студентов** каждая группа печаталась на отдельной странице, нужно задать значение **После раздела** в свойстве **Конец страницы** раздела **Примечание** группы **Группа**. Чтобы разделы не разрывались при печати, нужно задавать значение «**Да**» в свойстве **Не разрывать**.

При печати отчета важно правильно выделить место для ЭУ, особенно если это длинное символьное поле или поле типа Мемо. У ЭУ типа поле есть свойства **Расширение** и **Сжатие**. Если задать значение «**Да**» в свойстве **Расширение**, при необходимости будет автоматически увеличиваться высота области, выделенной полю на бланке отчета. Раздел, в котором находится такой ЭУ, автоматически получает значение «**Да**» в свойстве **Расширение**. По умолчанию вместо полей, содержащих пустые значения, на страницах отчета остается пустое пространство. Для удаления этого пустого пространства следует использовать свойство **Сжатие**. Поле, имеющее значение «**Да**»**,** в этом свойстве не занимает места при печати отчета, если содержит пустое значение или строку нулевой длины.

Приложение 1. Схемы данных учебных БД

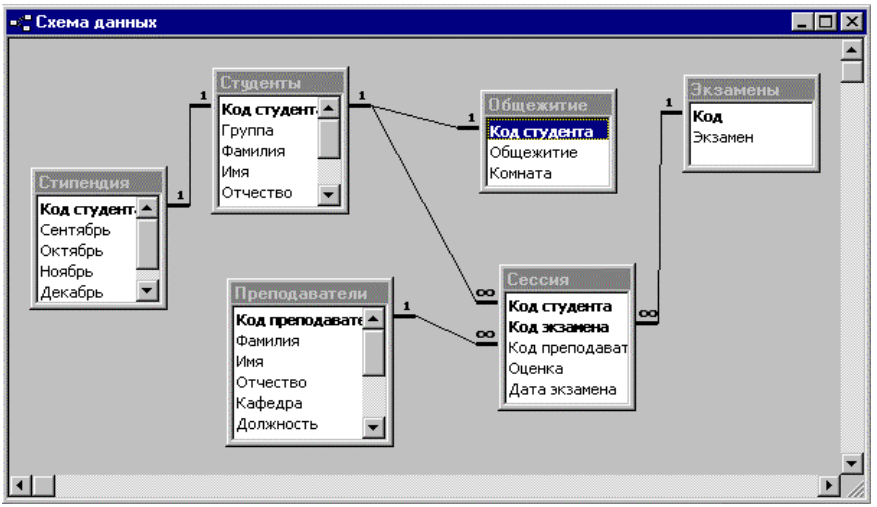


*Рис 1. Схема данных БД «**Деканат**»*

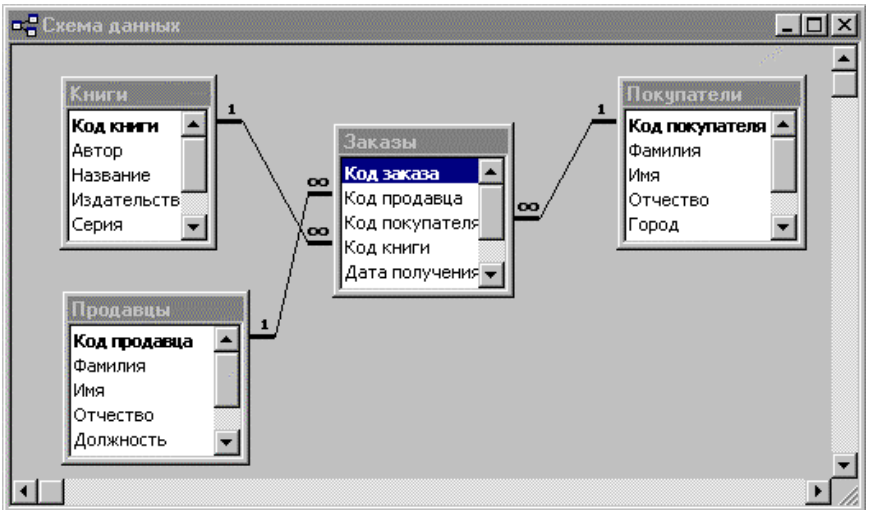


*Рис 2. Схема данных БД **«Книги»***

Приложение 2. Структура таблиц БД «Деканат»

Студенты

Имя поля	*Тип*	*Длина*	*Индекс*
Код студента	Текстовый	6	Да
Группа	Текстовый	4	
Фамилия	Текстовый	20	
Имя	Текстовый	20	
Отчество	Текстовый	20	
Курс	Числовой		
Пол	Текстовый	1	
Дата рождения	Дата/время		

Сессия

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

Экзамены

Имя поля	*Тип*	*Длина*	*Индекс*
Код экзамена	Счетчик		Да
Экзамен	Текстовый	20	

Общежитие

Имя поля	*Тип*	*Длина*	*Индекс*
Код студента	Текстовый	6	Да
Общежитие	Текстовый	20	
Комната	Текстовый	20	

Преподаватели

Имя поля	*Тип*	*Длина*	*Индекс*
Код преподавателя	Счетчик		Да
Фамилия	Текстовый	20	
Имя	Текстовый	20	
Отчество	Текстовый	20	
Кафедра	Текстовый	20	
Должность	Текстовый	20	
Звание	Текстовый	10	
Адрес	Текстовый	40	
Рабочий телефон	Текстовый	8	
Домашний телефон	Текстовый	8	
Дата рождения	Дата/время		

Стипендия

Имя поля	*Тип*	*Длина*	*Индекс*
Код студента	Текстовый	6	Да
Сентябрь	Денежный		
Октябрь	Денежный		
Ноябрь	Денежный		
Декабрь	Денежный		
Январь	Денежный		

Приложение 3. Структура таблиц БД «Книги»

Покупатели

Имя поля	*Тип*	*Длина*	*Индекс*
Код покупателя	Счетчик		Да
Фамилия	Текстовый	20	
Имя	Текстовый	20	
Отчество	Текстовый	20	
Город	Текстовый	20	
Адрес	Текстовый	40	
Страна	Текстовый	20	

Продавцы

Имя поля	*Тип*	*Длина*	*Индекс*
Код продавца	Счетчик		Да
Фамилия	Текстовый	20	
Имя	Текстовый	20	
Отчество	Текстовый	20	
Должность	Текстовый	30	
Дата приема	Дата/время		
Оклад	Денежный		

Книги

Имя поля	*Тип*	*Длина*	*Индекс*
Код книги	Счетчик		Да
Автор	Текстовый	30	
Название	Текстовый	40	
Издательство	Текстовый	30	
Серия	Текстовый	40	
Год издания	Числовой		
Цена	Денежный		

Заказы

Имя поля	*Тип*	*Длина*	*Индекс*
Код заказа	Счетчик		Да
Код продавца	Числовой		
Код покупателя	Числовой		
Код книги	Числовой		
Дата получения	Дата/время		
Дата отправки	Дата/время		
Количество	Числовой		

Литература

1. К. Дейт. Введение в системы баз данных, Спб.: Вильямс, 2000.
2. Р. Дженнингс. Microsoft Access 97 в подлиннике. В 2 т. Спб.: BHV, 1997.
3. Д. Вейскас. Эффективная работа с Microsoft Access. Спб.: Питер, 1997.
4. А. Гончаров. Microsoft Access 97 в примерах. Спб.: Питер, 1997.
5. Р. Винтер. Microsoft Access 97: справочник. Спб.: Питер, 1998.
6. С.Б. Барабаш, Н.С. Кощеева. Задания по Microsoft Access. Новосибирск, НГУ, 2000.

Оглавление

Глава 1. Введение в Access ... 3
1.1. Понятие базы данных ... 3
1.2. Реляционные БД ... 4
1.3. СУБД Access ... 5
1.4. Объекты Access ... 6
1.5. Запуск и завершение работы ... 7
1.6. Структура окна Access ... 8
1.7. Открытие и закрытие БД ... 8
1.8. Получение справки в Access ... 10
1.9. Учебные базы данных ... 11
Глава 2. Построение базы данных ... 12
2.1. Основные принципы проектирования БД ... 12
2.2. Создание новой базы данных ... 13
2.3. Создание таблиц ... 14
2.4. Сохранение таблицы ... 29
2.5. Определение связей между таблицами ... 29
2.6. Модификация БД ... 35
Глава 3. Работа с таблицей ... 39
3.1. Ввод данных ... 39
3.2. Перемещение по таблице ... 41
3.3. Редактирование данных в таблице ... 42
3.4. Настройка внешнего вида таблицы ... 47
3.5. Поиск и замена данных ... 49
3.6. Сортировка и фильтрация данных ... 51
Глава 4. Выражения в Access ... 55
4.1. Операторы ... 55
4.2. Константы ... 60
4.3. Функции ... 60
4.4. Построитель выражений ... 66

Глава 5. Создание запросов ... 69
5.1. Общая характеристика запросов ... 69
5.2. Работа в окне конструктора запросов ... 73
5.3. Отбор записей в однотабличных запросах ... 80
5.4. Многотабличные запросы ... 89
5.5. Подведение итогов ... 104
5.6. Перекрестные запросы ... 111
5.7. Запросы на изменение ... 120
Глава 6. Формы ... 129
6.1. Общие сведения ... 129
6.2. Создание формы ... 132
6.3. Создание основных ЭУ ... 140
6.4. Настройка формы ... 156
6.5. Создание диаграмм ... 159
6.6. Составные формы ... 164
Глава 7. Отчеты ... 183
7.1. Основные понятия ... 183
7.2. Создание отчета ... 185
7.3. Сортировка и группирование ... 188
7.4. Режимы окна отчета ... 191
7.5. Печать отчета ... 192
Приложение 1. Схемы данных учебных БД ... 194
Приложение 2. Структура таблиц БД «Деканат» ... 195
Приложение 3. Структура таблиц БД «Книги» ... 197
Литература ... 198