

**Юрий Бекаревич  
Нина Пушкина**

**Самоучитель  
ACCESS  
2010**

Санкт-Петербург  
«БХВ-Петербург»  
2011

УДК 681.3.06  
ББК 32.973.26-018.2  
Б42

**Бекаревич, Ю. Б.**

Б42      Самоучитель Access 2010 / Ю. Б. Бекаревич, Н. В. Пушкина. — СПб.:  
БХВ-Петербург, 2011. — 432 с.: ил. + CD-ROM

ISBN 978-5-9775-0651-9

Самоучитель позволяет изучить новый интерфейс и инструменты Microsoft Office Access 2010 для разработки баз данных и приложений пользователя. Рассмотрены возможности быстрой адаптации начинающего пользователя к среде Access 2010 для решения практических задач, в том числе по созданию таблиц в процессе ввода данных, упрощенному получению форм или отчетов. Опытные пользователи найдут описание усовершенствованных средств создания таблиц, схемы базы данных, запросов для решения задач пользователя, форм документов, отчетов и сводных таблиц для многоаспектного анализа данных и интерфейса приложения. Показано, как повысить эффективность разработки приложений, используя новый конструктор макросов и макросы данных, подключаемые при обработке событий в таблицах. Прилагаемый CD содержит примеры баз данных из разных предметных областей, демонстрирующие основные приемы работы с инструментарием Access 2010, а также дополнительные материалы по работе с базами данных Microsoft SQL Server и программированию на VBA.

*Для широкого круга пользователей и программистов*

УДК 681.3.06  
ББК 32.973.26-018.2

**Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Наталья Смирнова</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 30.09.10.

Формат 70×100<sup>1/16</sup>. Печать офсетная. Усл. печ. л. 34,83.

Тираж 2000 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.60.953.Д.005770.05.09  
от 26.05.2009 г. выдано Федеральной службой по надзору  
в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов  
в ГУП "Типография "Наука"  
199034, Санкт-Петербург, 9 линия, 12

# Оглавление

ПРЕДИСЛОВИЕ .....	1
<b>ГЛАВА 1. ВВЕДЕНИЕ В ACCESS 2010.....</b>	<b>5</b>
СУБД Microsoft Access и ее реляционная база данных .....	6
Система управления базами данных.....	6
Требования к системе для Access 2010 .....	9
Реляционная база данных .....	10
Определения и понятия .....	10
Схема данных.....	13
Объекты Access.....	13
Сводные таблицы и сводные диаграммы .....	15
Размещение базы данных.....	17
Шаблоны баз данных .....	18
Мастера Access.....	19
Средства конструирования объектов.....	21
Средства программирования .....	22
Интеграция и использование внешних данных .....	24
Поддержка технологий корпоративных сетей.....	25
Многопользовательская база данных Access .....	26
Работа Access с базой данных SQL Server.....	27
Интернет-технологии.....	29
Начало работы в Microsoft Access 2010.....	31
Запуск Access .....	31
Интерфейс пользователя Access.....	33
Представление Backstage .....	35
Лента .....	35
Коллекция .....	36
Диалоговые окна .....	36
Контекстное меню .....	38
Пользовательская панель инструментов быстрого доступа .....	38
Область навигации.....	40
Вкладки документов .....	43
Строка состояния .....	43
Панель сообщений .....	44
Мини-панель инструментов .....	44
Технология Drag and Drop.....	45
Смарт-теги .....	45
Справка Access.....	46
Защита баз данных.....	48
Контрольные вопросы .....	53
Ответы.....	54

<b>ГЛАВА 2. ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ .....</b>	<b>55</b>
Этапы проектирования и создания базы данных .....	55
Построение информационно-логической модели данных .....	57
Информационные объекты .....	58
Функциональные зависимости реквизитов .....	58
Требования нормализации .....	59
Выделение информационных объектов предметной области.....	60
Информационный анализ и определение логической структуры информации ..	61
Выделение информационных объектов на примере предметной области "Поставка товаров" .....	64
Связи информационных объектов .....	74
Тип связи информационных объектов.....	75
Определение связей между информационными объектами.....	77
Информационно-логическая модель предметной области.....	78
Логическая структура реляционной базы данных.....	79
Контрольные вопросы .....	81
Ответы.....	82
<b>ГЛАВА 3. СОЗДАНИЕ БАЗЫ ДАННЫХ .....</b>	<b>85</b>
Создание файла базы данных Access.....	87
Окно Access .....	89
Лента .....	89
Представление Backstage .....	90
Панель быстрого доступа .....	90
Область навигации .....	91
Область документов .....	93
Строка состояния .....	94
Создание таблицы базы данных.....	95
Определение первичного ключа.....	100
Создание таблицы в режиме таблицы .....	101
Добавление полей .....	102
Определение структуры таблицы в режиме конструктора .....	106
Создание таблиц базы данных <i>Поставка товаров</i> .....	107
Создание структуры таблицы .....	107
Определение составного первичного ключа.....	113
Использование данных типа <i>Поле объекта OLE</i> .....	114
Использование данных типа <i>Вложение</i> .....	115
Использование данных типа <i>Поле МЕМО</i> .....	115
Использование данных типа <i>Гиперссылка</i> .....	116
Использование данных типа <i>Дата/время</i> .....	118
Маска ввода для даты и текста .....	119
Непосредственный ввод данных в таблицы .....	119
Макет таблицы .....	121
Ввод данных в таблицы базы данных.....	122
Размещение объекта OLE.....	124
Размещение вложений.....	125
Ввод логически связанных записей .....	127
Отображение записей подчиненных таблиц в главной таблице .....	128

Использование поля со списком при вводе записей .....	133
Создание поля со списком в режиме таблицы .....	133
Создание поля со списком в режиме конструктора.....	137
Схема данных в Access .....	140
Создание схемы данных.....	140
Включение таблиц в схему данных.....	141
Создание связей между таблицами схемы данных.....	141
Обеспечение целостности данных .....	143
Каскадное обновление и удаление связанных записей .....	145
Контрольные вопросы.....	146
Ответы.....	147
<b>ГЛАВА 4. ЗАПРОСЫ .....</b>	<b>149</b>
Однотабличные запросы на выборку .....	151
Конструирование запросов на выборку с условиями отбора .....	151
Вычисляемые поля в запросах.....	155
Параметры в запросах .....	159
Групповые операции в запросах .....	160
Запрос с функцией <i>Sum</i> .....	161
Запрос с функцией <i>Count</i> .....	163
Условия отбора в запросах с групповыми операциями .....	164
Отображение строки итогов по столбцу.....	165
Просмотр инструкции SQL в запросе .....	166
Контрольные вопросы .....	168
Ответы.....	169
Многотабличные запросы на выборку данных .....	170
Объединение записей в многотабличном запросе .....	171
Параметры объединения в инструкциях SQL.....	175
Ссылки на имена полей различных таблиц в условии отбора .....	176
Представление данных нарастающим итогом .....	178
Решение задачи, требующей выполнения нескольких запросов .....	182
Контрольные вопросы .....	188
Ответы.....	189
Запросы на изменение .....	189
Конструирование запроса на создание таблицы.....	191
Конструирование запроса на обновление .....	192
Обновление полей значениями, рассчитанными в запросе с группировкой .....	194
Конструирование запроса на добавление .....	196
Добавление данных в связанные таблицы.....	197
Добавление данных из нескольких таблиц .....	199
Конструирование запроса на удаление .....	201
Контрольные вопросы .....	203
Ответы.....	204
<b>ГЛАВА 5. ФОРМЫ.....</b>	<b>207</b>
Однотабличные формы .....	208
Создание однотабличной формы .....	210

Редактирование формы в режиме макета.....	211
Макеты элементов управления .....	213
Условное форматирование элементов управления .....	214
Свойства формы.....	216
Добавление полей в форму .....	217
Работа с данными таблицы в режиме формы .....	217
Создание формы на основе запроса.....	219
Создание разделенной формы .....	220
Вычисления в форме .....	222
Вычисления в каждой записи формы .....	222
Вычисление итоговых значений.....	224
Многотабличные формы.....	225
Создание многотабличной формы с помощью мастера .....	227
Создание одиночной многотабличной формы .....	233
Создание и редактирование формы в режиме конструктора .....	236
Создание новой формы конструктором .....	236
Добавление подчиненной формы.....	237
Вычисление итогового значения в подчиненной форме и вывод его в текущей записи основной формы.....	241
Ограничения доступа к данным через форму.....	243
Защита данных поля от изменений .....	243
Установка ограничений на корректировку записей через форму .....	243
Защита данных подчиненной формы от изменений.....	245
Разработка интерфейса для ввода, просмотра и корректировки документов .....	245
Этапы разработки интерфейса.....	246
Определение последовательности загрузки таблиц с документов .....	247
Справочная информация .....	248
Плановая информация .....	248
Оперативно-учетная информация .....	248
Проектирование интерфейса для ввода и корректировки документа .....	249
Определение подсхемы данных .....	250
Разработка макета .....	251
Создание интерфейса для ввода и корректировки документа .....	253
Доработка интерфейса .....	256
Создание кнопок .....	259
Ограничение доступа к данным таблиц базы.....	260
Создание полей со списком .....	261
Вычисления в документе.....	267
Работа с документами .....	268
Выборка документа по его идентификатору.....	270
Выборка документа по диапазону дат .....	273
Выборка документов с помощью фильтрации .....	274
Контрольные вопросы.....	276
Ответы.....	279
<b>ГЛАВА 6. СВОДНЫЕ ТАБЛИЦЫ И ДИАГРАММЫ. АНАЛИЗ ДАННЫХ .....</b>	<b>283</b>
Режим сводной таблицы .....	284
Разработка сводной таблицы для таблицы базы данных.....	284

Размещение полей в макете сводной таблицы .....	285
Вычисление итоговых значений .....	287
Разработка сводной таблицы для запроса.....	289
Работа с датами в сводных таблицах .....	291
Использование нескольких полей в областях сводной таблицы .....	292
Добавление полей в источник записей сводной таблицы .....	292
Форматирование элементов сводной таблицы .....	293
Вычисляемые итоги и поля в сводной таблице .....	293
Режим сводной диаграммы.....	297
Контрольные вопросы.....	301
Ответы.....	301
<b>ГЛАВА 7. ОТЧЕТЫ.....</b>	<b>303</b>
Основы конструирования отчетов .....	304
Однотабличные отчеты .....	306
Доработка отчета в режиме макета .....	308
Группировка и сортировка данных отчета .....	309
Просмотр и печать отчета .....	313
Представление отчета.....	313
Предварительный просмотр.....	315
Печать отчета.....	317
Изменение источника записей отчета.....	318
Многотабличные отчеты.....	320
Разработка отчета с помощью мастера .....	322
Выбор таблиц для отчета .....	322
Источник записей отчета.....	328
Доработка отчета в режиме конструктора.....	328
Вычисляемые поля в отчете .....	329
Определение параметров в отчете .....	331
Анализ данных отчета средствами фильтрации .....	332
Составные отчеты.....	333
Создание главного отчета .....	335
Создание подчиненного отчета .....	336
Сортировка и группировка записей отчета .....	337
Включение подчиненного отчета.....	339
Доработка составного отчета.....	340
Добавление текущей даты и номера страницы .....	341
Просмотр отчета.....	343
Вывод значений нарастающим итогом .....	344
Контрольные вопросы.....	344
Ответы.....	346
<b>ГЛАВА 8. РАЗРАБОТКА ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ МАКРОСОВ.....</b>	<b>349</b>
Конструирование макроса .....	351
Формирование макрокоманд в окне макроса .....	352
Формирование макроса с помощью мыши .....	354
Использование в макросах ссылок на объекты .....	355
Ссылки на объекты и их элементы управления.....	355

Ссылки на свойство объекта.....	356
Ссылки на свойство элемента управления.....	356
Ссылка на подчиненную форму или отчет.....	357
Создание ссылок построителем выражений .....	358
Вложенные макросы .....	358
Внедренный макрос.....	362
Управление последовательностью выполнения макрокоманд .....	364
Примеры условных выражений.....	368
Организация выполнения макросов .....	369
Запуск макроса.....	369
Выполнение макроса с наступлением события .....	370
Порядок выполнения макросов, вызываемых событиями.....	373
Вызов макроса из другого макроса.....	374
Создание кнопки запуска макроса в форме .....	375
Создание кнопки запуска макроса с помощью мыши.....	375
Создание кнопки запуска макроса мастером .....	375
Макросы данных.....	376
Именованные макросы.....	378
Использование макросов для решения задач.....	379
Регистрация событий.....	386
Контрольные вопросы.....	387
Ответы.....	388
<b>ГЛАВА 9. РАЗРАБОТКА ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ .....</b>	<b>389</b>
Управление приложением на основе форм.....	389
Диалоговое окно входа в приложение.....	390
Главная форма управления приложением.....	395
Использование форм с вкладками при разработке интерфейса.....	400
Панель навигации формы с вкладками.....	406
Формы навигации .....	407
Контрольные вопросы.....	409
Ответы.....	409
<b>ПРИЛОЖЕНИЕ 1. СТРУКТУРА ТАБЛИЦ БАЗЫ ДАННЫХ <i>ПОСТАВКА ТОВАРОВ</i> .....</b>	<b>411</b>
Таблицы справочных данных.....	411
Таблицы плановых данных.....	413
Таблицы оперативно-учетных данных .....	414
<b>ПРИЛОЖЕНИЕ 2. ПРИМЕР ЗАПОЛНЕННЫХ ДОКУМЕНТОВ ДЛЯ ЗАГРУЗКИ В БАЗУ ДАННЫХ <i>ПОСТАВКА ТОВАРОВ</i> .....</b>	<b>415</b>
Справочная информация.....	415
Плановая информация.....	416
Оперативно-учетная информация .....	417
<b>ПРИЛОЖЕНИЕ 3. ОПИСАНИЕ КОМПАКТ-ДИСКА .....</b>	<b>418</b>
<b>ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ .....</b>	<b>420</b>

# Предисловие

Эта книга предназначена для самостоятельного изучения новой версии приложения по управлению реляционными базами данных — СУБД Microsoft Office Access 2010.

Access 2010 позволяет создавать как простые, так и очень сложные приложения баз данных. В новой версии стало еще проще разработать приложение, не зная языков программирования и не имея предварительных навыков проектирования баз данных. С помощью этого приложения отдельные пользователи, работающие с информацией, могут в кратчайшие сроки начать выполнять повседневные задачи, работая с документами и реализуя деловые операции бизнес-процессов. В то же время, используя функциональные возможности Access, специалисты отделов информационных технологий смогут обеспечить разработку и поддержку приложений Access, создаваемых на уровне компаний. Access 2010 для любых пользователей имеет необходимый набор функций.

Широкий круг пользователей в значительной степени знаком с возможностями популярного приложения Microsoft Office Excel. Однако если объемы данных значительны, то для организации пользовательских таблиц Excel, так называемых баз данных, целесообразно переходить к использованию специализированных средств создания и использования баз данных Access. СУБД обладает мощным и удобным аппаратом распределения данных по нескольким таблицам, реализуя практически любую модель данных предметной области. При этом обеспечивается совместная обработка и корректировка данных средствами форм и отчетов, обеспечивающими эффективный доступ и представление данных. В основе этих средств лежит мощный реляционный язык структурированных запросов SQL. С помощью этого языка пользователь может сформулировать практически любой запрос к базе данных и решать разнообразные задачи обработки данных. Благодаря этому обеспечивается возможность без помощи опытных программистов и администраторов разрабатывать практические приложения базы данных в различных сферах.

Новая версия Access 2010, продолжая линию на модернизацию интерфейса пользователя, заложенную в Access 2007, делает еще доступнее и удобнее использование средств СУБД для начинающих пользователей. Современный интуитивно понятный интерфейс, многочисленные готовые решения, простые средства создания таблиц базы данных, форм и отчетов в полной мере реализуют эту цель.

Работа с усовершенствованным интерфейсом, обеспечивающим быстрый поиск необходимых инструментов, позволяет сразу включиться в подготовку профессиональных документов, повышает эффективность, качество работы и сокращает сроки ее выполнения. Команды стало легче находить благодаря тому, что они организованы в виде набора вкладок, которые четко соответствуют функциям, реализуемым в Access.

Готовые приложения, доступные через шаблоны, поставляемые вместе с Access или на веб-сайтах Microsoft, помогут пользователю, обладающему ограниченными знаниями в области баз данных, познакомиться с самыми важными функциями и во многих случаях начать работу с первого дня использования Access 2010. Можно использовать эти стандартные приложения без каких-либо модификаций или настройки, а можно взять их за отправную точку и адаптировать в соответствии с задачами и способами обработки информации.

Access 2010 предоставляет гибкие средства для работы пользователя с разным уровнем подготовки. Новичок может начать создавать таблицы с любым числом полей непосредственным вводом данных без предварительного определения ее структуры. Тип введенных данных определится автоматически. Тут же прямо в режиме таблицы можно вмешаться в этот процесс: присвоить имя, уточнить тип данных, его формат, создать столбцы подстановки.

Кроме создания новых таблиц в базе данных, Access позволяет организовать связи с внешними данными с помощью интерфейса ODBC (Open Database Connectivity). Связанные таблицы отображаются и функционируют аналогично собственным таблицам. Возможность создания связанных таблиц обеспечивает легкость, с которой пользователи Access могут перемещаться, объединять и обновлять данные из нескольких разных источников. Через связанные таблицы возможно взаимодействие и со всеми распространенными серверами реляционных баз данных, включая Microsoft SQL Server, Oracle, IBM DB2, Informix и Sybase.

Интерактивные средства обеспечивают простоту разработки форм и отчетов. Построить форму или отчет можно одним щелчком кнопки мыши. Режим представления форм и отчетов — режим макета позволяет легко настроить их в соответствии с потребностями пользователя, наблюдая изменения этих объектов в реальном времени. Разделенные формы с одной частью для ввода данных и другой для быстрого просмотра данных в режиме таблицы существенно повышают удобства в работе. Простота изменения дизайна форм, отчетов, электронных документов, сохраняемых в базе данных, обеспечивается объединением их элементов в макеты. Такие макеты обеспечивают настройку всех элементов в целом, например макет в столбец позволяет изменять размеры сразу всех его элементов, а табличный макет обеспечивает смещение всех элементов строки при изменении размера одного из элементов.

Простые в использовании и разнообразные средства фильтрации данных позволяют, не перестраивая макет, получить отчеты, представляющие данные в самых разных разрезах, и использовать их для анализа данных.

Задача Access 2010 для баз данных, созданных в новом формате (файлы с расширением accdb или accde), построена на доверии к происхождению ее выполняемых объектов. Некоторые объекты Access, такие как запросы на изменение, макросы, изменяющие базу данных, код VBA и элементы управления ActiveX, полученные из ненадежных источников, выполняясь, могут повредить базу данных. Поэтому база данных открывается для работы с полным набором функциональных возможностей только, если ее компоненты подписаны надежными издателями или пользователь придал ей состояние доверенной и берет на себя ответственность за безо-

пасность. Средства обеспечения безопасности баз данных нового формата не предусматривают защиты на уровне пользователей.

Access 2010 позволяет без значительных затрат создавать практически любые приложения баз данных для индивидуальных пользователей и небольших рабочих групп. Если критерии масштабируемости, бесперебойной работы, надежности и безопасности все-таки диктуют необходимость использования сервера баз данных, например Microsoft SQL Server, то Access может быть использован для разработки клиентского приложения и создания его объектов.

Подключение к таблицам в базе данных SQL Server напрямую через интерфейс OLE DB осуществляется из проекта Access (расширение adp). В проекте открывается возможность не только разрабатывать приложение, используя формы, отчеты и другие средства Access, а также создавать и модифицировать объекты серверной базы данных.

В Access 2010 обеспечивается возможность сбора в базе данных через электронную почту Office Outlook. С помощью функции "Отправить по электронной почте" форма отправляется клиенту, который, щелкнув на кнопке **Ответить**, заполняет ее и отправляет обратно. При поступлении формы в папку "Входящие" введенные клиентом данные будут автоматически внесены в соответствующую таблицу базы.

Новые технологии в Access 2010 позволяют создавать специальные веб-базы данных и публиковать их на сайтах Microsoft SharePoint Server, на котором выполняются службы Access. Базу данных можно опубликовать как на собственном сервере SharePoint в интрасети, так и в Интернете. Пользователи могут использовать базу данных с помощью стандартного браузера, не устанавливая приложение Access на компьютере. Это делает простым совместное использование корпоративной информации в среде настольных систем. При этом пользователи могут с помощью браузера открывать веб-формы и отчеты.

Вследствие возможности Access удовлетворять потребности самых разных групп пользователей система получила на рынке значительно большее признание, чем любое другое приложение для работы с базами данных. С помощью мастеров и графических инструментов Access даже пользователи, не владеющие специальными навыками, могут весьма успешно разрабатывать приложения баз данных.

В процессе подготовки самоучителя по Access 2010 перед авторами стояла задача ознакомить пользователей с особенностями новой версии, сформулировать и осмыслить тенденции развития. Вместе с тем самоучитель ориентирован на получение целостного представления о программном продукте, изучение во взаимосвязи всех компонент от традиционных инструментальных средств до перспективных технологий. Важнейшей целью книги является демонстрация функционирования всех рассматриваемых средств на конкретных примерах.

В качестве сквозного примера рассматривается база данных, содержащая сведения о плановых и фактических поставках товаров гипотетического предприятия. Представлены примеры работы с данными справочных, плановых и оперативно-учетных документов и решение экономических задач. Такой выбор продиктован

тем, что одним из характерных применений СУБД является хранение и обработка экономической информации.

Система взаимосвязанных примеров позволяет более наглядно показать возможности инструментальных средств конструирования объектов, средств программирования при разработке приложений, организации диалога при решении экономических задач. Показана возможность использовать среду Access при разработке профессиональных приложений, обеспечивающих подготовку системы взаимосвязанных документов, их сохранение в базе данных с возможностью оперативно получать аналитические оценки при дополнении базы новыми данными.

Выполнение приведенных примеров на компьютере в процессе изучения Access 2010 позволит успешно освоить его инструментарий, приобрести навыки работы с базой данных и использовать их в любой профессиональной предметной области.

Книга рассчитана на широкий контингент пользователей от студента, изучающего основы технологии баз данных, до разработчиков практических приложений в среде СУБД. Самоучитель сопровождает читателя-пользователя в процессе создания файла базы данных, ее таблиц, запросов, последующего конструирования объектов приложения: форм, отчетов и макросов, вплоть до последнего этапа, когда отдельные компоненты объединяются в единое приложение.

Для новичка, впервые открывшего книгу из этой серии, она будет не только источником сведений по новым инструментальным средствам Access 2010<sup>1</sup>, но станет удобным практическим пособием при их освоении на компьютере, так как все примеры содержат подробное описание действий пользователя в процессе выполнения конкретных заданий. Проверить степень освоения изученного материала позволяют контрольные вопросы с ответами, приведенные в конце каждого раздела.

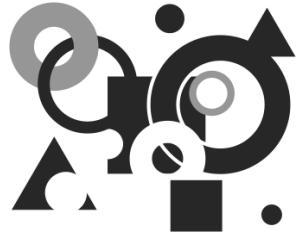
Применительно к тематике настоящего издания важно не только изучить возможности программных средств Access 2010 и сформировать практические навыки, но самое главное — научиться использовать эти средства для разработки баз данных и приложений пользователя в соответствии с требованиями конкретной предметной области.

Прилагаемый к книге компакт-диск содержит базу данных предметной области Поставка товаров, используемой в основных примерах по созданию объектов Access 2010, рассматриваемых в соответствующих главах книги. Наличие такой базы данных позволит использовать ее для изучения разделов книги в любой последовательности в зависимости от подготовленности и желания читателя. Кроме того, в диск включена база данных из сферы учебного процесса, что может быть полезным для широкого круга читателей самоучителя, далеких от особенностей предметной области основного примера.

В диск включены материалы по программированию приложений в среде VBA и разработке клиентского приложения Access для многопользовательской базы данных на SQL Server.

---

<sup>1</sup> Для операционных систем и программ, кроме Microsoft Access, перевод интерфейса не приводится. — Прим. ред.



## ГЛАВА 1

# Введение в Access 2010

Подавляющее большинство деловых операций сегодня записывается, отслеживается и анализируется в виде данных, хранящихся в реляционных системах управления базами данных.

Система управления реляционными базами данных Microsoft Office Access удовлетворяет потребности самых разных групп пользователей. С помощью мастеров и графических инструментов Access даже пользователи, не владеющие специальными навыками, могут весьма успешно разрабатывать полезные приложения баз данных. Исследования малых и средних предприятий, проведенные различными службами, показали, что Access является одной из самых популярных программ для работы с базами данных.

Приложения для автоматизации работы с электронными таблицами, такие как Microsoft Office Excel, используются на персональных компьютерах с момента их появления и, реализуя мощные вычислительные функции, средства анализа данных и построения диаграмм, позволяют выполнять многие стандартные задачи по обработке табличных данных. Современные приложения для совместной работы, такие как Windows SharePoint, также поддерживают создание и обслуживание списков, доступных через интерфейс Web-обозревателя. Но следует заметить, что ни один из продуктов, поддерживающих таблицы, не обладает всеми достоинствами настоящей реляционной базы данных.

По мере усложнения приложений на основе таблиц или списков возникает вопрос о перемещении данных в стандартные реляционные таблицы, из которых можно выбирать и обрабатывать данные с помощью языка SQL (Structured Query Language).

После превращения списков в реляционные данные Access позволяет быстро создавать приложения для решения самых разнообразных задач. Для решения стандартных задач в Access включен ряд шаблонов баз данных. Кроме того, многочисленные шаблоны баз данных представлены на веб-узле Office Online.

Мастера и удобные средства конструирования обеспечивают простоту создания в Access реляционных структур данных, а также запросов, форм и отчетов, необходимых для работы с данными. Контекстно-зависимый интерфейс всегда предоставит пользователю элементы, необходимые в данный момент времени.

Улучшенный интерфейс пользователя и интерактивные средства разработки в составе Access 2010 делают разработку приложений в среде этого программного продукта доступной для начинающих пользователей. Любой сотрудник, не имея опыта программирования и обладая ограниченными знаниями в области баз данных, может, используя Access, самостоятельно решать задачи по обработке данных.

В то же время Access удовлетворяет требованиям профессиональных разработчиков и позволяет за незначительное время разрабатывать сложные бизнес-системы.

Благодаря тесной интеграции с Microsoft Windows SharePoint стало удобно создавать приложения для совместной работы с базой данных в масштабах всего предприятия. Размещение данных базы в списках на веб-узле SharePoint обеспечивает общий доступ к ним через веб-обозреватель и сохраняет возможность доступа через Access. Задавая с помощью Windows SharePoint разрешения на доступ к спискам, можно предоставлять или запрещать доступ определенным пользователям, предоставить разрешение только на чтение или все разрешения на изменение. При использовании узла SharePoint для доступа к базам данных пользователи могут подписаться на уведомления по электронной почте, отправляемые при добавлении, удалении или изменении записей в списке SharePoint.

Интеграция с Office Outlook позволяет использовать электронную почту для сбора или обновления данных в таблице Access, не прибегая к вводу данных вручную.

## **СУБД Microsoft Access и ее реляционная база данных**

### **Система управления базами данных**

СУБД (система управления базами данных) является универсальным программным инструментом создания и обслуживания баз данных и приложений пользователя в самых разных предметных областях. СУБД обеспечивает создание, многоаспектный доступ к данным и использование одних и тех же данных различными задачами и приложениями пользователей.

СУБД поддерживаются различные модели данных. *Модель данных* — это метод (принцип) логической организации данных, используемый СУБД. Наиболее известными являются иерархическая, сетевая и реляционная модели.

В СУБД для персональных компьютеров (настольных СУБД) поддерживается преимущественно *реляционная модель*, которую отличает простота и единообразие представления данных простейшими *двумерными таблицами*. Реляционная модель обеспечивает возможность использования в разных СУБД операций обработки данных, имеющих единую основу — *алгебру отношений* (реляционную алгебру), и универсального языка структурированных запросов — SQL (Structured Query Language).

Основной логической структурной единицей манипулирования данными является *строка таблицы — запись*. Структура записи определяется составом входящих в нее *полей*. Совокупность полей записи соответствует логически связанным реквизитам, характеризующим некоторую сущность предметной области.

Типовыми функциями СУБД по манипулированию данными являются выборка, добавление, удаление, изменение данных.

*Выборка данных* — выборка записей из одной или нескольких взаимосвязанных таблиц в соответствии с заданными условиями.

*Добавление и удаление данных* — добавление новых записей в таблицы и удаление существующих.

*Изменение данных* — изменение значений данных в полях существующих записей.

Данные из одной или нескольких взаимосвязанных таблиц могут подвергаться обработке. К операциям *обработки* относятся, например, расчеты в пределах каждой записи, группировка записей в соответствии с заданным критерием группировки и обработка записей выделенных групп с помощью статистических функций, таких как суммирование, определение максимального, подсчет числа записей в группе и т. п.

СУБД *Microsoft Access* является системой управления реляционной базой данных, включающей все необходимые инструментальные средства для создания локальной базы данных, общей базы данных в локальной сети с файловым сервером или базы данных на SQL-сервере, а также для создания приложения пользователя, работающего с этими базами данных. База данных Access, создаваемая на локальном компьютере, хранит в файле не только все таблицы с данными, но и объекты приложения — формы, отчеты, а также программный код. Благодаря этому можно создать приложение, целиком хранящееся в одном accdb-файле, что существенно упрощает как создание, так и распространение приложений баз данных.

СУБД Access включают разнообразные и многочисленные относительно автономные инструментальные средства, ориентированные на создание объектов базы данных и приложений пользователя.

*Средства графического конструирования* позволяют создавать объекты базы данных и объекты приложения с помощью многочисленных графических элементов, не прибегая к программированию.

*Разнообразные мастера* в режиме ведения диалога с пользователем позволяют создавать объекты и выполнять разнообразные функции по реорганизации и преобразованию баз данных.

Среди многочисленных средств графического конструирования и диалоговых средств Access следует выделить средства для создания:

- ❖ таблиц и схем баз данных, отображающих их связи;
- ❖ запросов выборки, отбирающих и объединяющих данные нескольких таблиц в виртуальную таблицу, которая может использоваться во многих задачах приложения;
- ❖ запросов на изменение данных базы;

- ◆ экранах форм, предназначенных для ввода, просмотра и обработки данных в диалоговом режиме;
- ◆ отчетов, предназначенных для просмотра и вывода на печать данных из базы и результатов их обработки в удобном для пользователя виде.

*Средства программирования* СУБД включают язык структурированных запросов SQL, язык макрокоманд и язык объектно-ориентированного программирования для приложений Microsoft Visual Basic for Applications (VBA). VBA является частью семейства Microsoft Visual Basic, которое входит в состав Visual Studio.

VBA является базовым компонентом Microsoft Office: он интегрирован в Access, Excel, FrontPage, Outlook, PowerPoint и Word. Все эти приложения, в том числе и локализованные на русском языке, используют англоязычный вариант VBA (включая справочную систему). VBA входит во все варианты поставок Microsoft Office.

VBA представляет собой базовую платформу программирования не только в среде Microsoft Office, но и многих других приложений. VBA содержит средства доступа не только к базам данных Access, но и к базам данных клиент-серверной архитектуры, таким как Microsoft SQL Server, Oracle и др.

*Система доступа к данным*, начиная с Access 2007, построена на основе ядра базы данных Access Database Engine, заменившего прежнюю версию ядра Microsoft Jet 4.0. Ядро базы данных выполняет загрузку, сохранение и извлечение данных в пользовательских и системных базах данных. Обеспечивает высокую производительность и улучшенные сетевые характеристики, поддержку двухбайтового представления символов — Unicode, позволяющего использовать символы нескольких национальных алфавитов. Для компенсации возрастающего при Unicode объема памяти применяет сжатие данных, сохраняемых в Unicode. Для лучшей совместимости Microsoft Jet и Microsoft SQL Server и соответствия языку SQL спецификации SQL 92 были внесены изменения еще в реализацию Jet 4.0 SQL.

Ядро базы данных Access 2010 настроено для приложений системы Microsoft Office 2010 и обеспечивает интеграцию со службами Microsoft Windows SharePoint Services 3.0 и Microsoft Office Outlook 2010.

В Access активно развиваются технологические направления, составляющие основу совместного использования корпоративных баз данных.

- ◆ Сохраняя реализованную в предыдущих версиях возможность создания клиентских приложений — *проектов*, в Access обеспечивается подключение к многопользовательским базам данных SQL Server, поддерживается технология "клиент-сервер". Подключение к серверу реализуется с помощью интерфейса OLE DB без использования ядра баз данных Access Database Engine. Благодаря возможности разрабатывать клиентские приложения для доступа к данным сервера, Access стал активно применяться на средних и крупных предприятиях. Для мягкого перехода к клиент-серверной технологии в Access включены многочисленные и разнообразные средства. Они обеспечивают преобразование объектов локальной базы данных и приложения в объекты базы данных сервера и проекта, удобный графический интерфейс создания объектов базы данных сервера и сохраняют интерфейс создания объектов приложения. Кроме того, бесплатная версия Microsoft SQL Server — SQL Server Express — обеспечивают

как локальное хранение данных в формате, совместимом с Microsoft SQL Server, так и удаленное, рассчитанное на небольшое число пользователей. Установка такого сервера не требует лицензирования и позволяет освоить клиент-серверную технологию, работая на компьютере, не подключенному к сети.

- ❖ Интернет-технология позволяет эффективно распространять и получать доступ к разнородной информации в глобальных и корпоративных сетях. Эта технология обеспечивает унифицированный доступ к данным различных приложений в разнородных сетях. При этом веб-браузер используется как универсальный интерфейс для доступа и работы с данными баз из внешней среды вне зависимости от аппаратно-программной платформы компьютера пользователя и компьютера — источника данных. Веб-страницы приложения могут использоваться подобно формам Access для ввода и редактирования данных, подобно отчетам Access — для отображения иерархически сгруппированных данных. Новая технология в Access 2010 предоставляет платформу для создания баз данных, которые можно совместно использовать в пределах организации или в Интернете. С помощью Access 2010 и службы Access (компонент сервера SharePoint Server) веб-базы данных разрабатываются и могут публиковаться, как на собственном сервере SharePoint в интрасети, так и в Интернете. При публикации базы данных службы Access создают сайт SharePoint, содержащий базу данных. Все данные базы перемещаются в списки SharePoint на этом сайте. Access 2010 и службы Access позволяют создавать веб-приложения баз данных. Формы, отчеты и большинство макросов таких приложений выполняются пользователями, обладающими необходимыми разрешениями, в браузере. При этом на компьютере не требуется устанавливать Access. Безопасность доступа к данным и управление им обеспечивается средствами SharePoint. После публикации базы данных в Интернете можно открыть файл ACCDB в Access для внесения изменений в структуру базы данных и затем синхронизировать эти изменения с опубликованной версией.

## Требования к системе для Access 2010

Access 2010 входит в состав Microsoft Office 2010 (в наборы приложений выпусков Профессиональный, Профессиональный Плюс и Корпоративный).

Для работы Access 2010 рекомендуется использовать компьютер с частотой процессора не ниже 500 МГц и RAM не менее 256 Мб. Компьютер должен работать под управлением одной из операционных систем:

- ❖ Windows XP с пакетом обновления 3 (SP3) (32-разрядная);
- ❖ Windows Vista с пакетом обновления 1;
- ❖ Windows 7;
- ❖ для ряда дополнительных возможностей по совместной работе требуется Windows Server 2003 R2 с установленным MSXML 6.0 с выполняемыми службами Microsoft Windows SharePoint Services или Windows Server 2008 (32- или 64-разрядная).

Для сбора данных с использованием электронной почты требуется Microsoft Outlook.

Более насыщенные функциональными возможностями новые системы семейства Windows в сочетании с пакетом Office 2010 дают пользователю возможность наиболее эффективно применять компьютер, но их установка не является обязательным условием для работы с пакетом Office 2010. При переходе с Microsoft Office 2007 на версию 2010 оборудование обновлять не нужно, хотя может потребоваться выполнить обновление операционной системы.

Практический минимум, предъявляемый Access 2010 к персональному компьютеру (процессор с частотой 233 МГц и 128 Мб оперативной памяти), возрастает при одновременном выполнении нескольких приложений Office 2010, так как для каждого приложения требуются дополнительные ресурсы.

При стандартной установке Access 2010 профессиональных выпусков требуется примерно 2 Гб свободного дискового пространства. Часть места на диске будет освобождена после установки при удалении с него исходного загрузочного пакета. В процессе установки на диске необходимо иметь дополнительное пространство примерно такого же объема.

Рекомендуется монитор с разрешением не менее 1024×768 или с более высоким разрешением с поддержкой 256 цветов.

При установке приложений Office 2010 на локальном компьютере требуется дисковод CD-ROM или DVD-дисков.

## Реляционная база данных

### Определения и понятия

*База данных* является организованной на машинном носителе совокупностью взаимосвязанных данных и содержит сведения о различных сущностях одной предметной области — реальных объектах, процессах, событиях или явлениях.

Реляционная база данных представляет собой множество взаимосвязанных двумерных таблиц — *реляционных таблиц*, называемых также *отношениями*, в каждой из которых содержатся сведения об одной сущности автоматизируемой предметной области.

Логическую структуру реляционной базы данных образует совокупность реляционных таблиц, между которыми установлены логические связи.

В таблицах базы должны сохраняться все данные, необходимые для решения задач предметной области. Причем каждый элемент данных должен храниться в базе только в одном экземпляре. Для создания таблиц, соответствующих реляционной модели данных, используется процесс, называемый нормализацией данных. *Нормализация* — это удаление из таблиц повторяющихся данных путем их переноса в новые таблицы, записи которых не содержат повторяющихся значений.

Минимальное дублирование данных в реляционной базе обеспечивает высокую эффективность поддержания базы данных в актуальном и непротиворечивом состоянии, одинократный ввод и корректировку данных.

Структура реляционной таблицы определяется составом полей. Каждое *поле* отражает определенную характеристику сущности. Для поля указывается тип и размер элементарного данного, размещаемого в нем, и ряд других свойств. Содержимое поля отображается в столбце таблицы. Столбец таблицы содержит данные одного типа.

Содержание таблицы заключено в ее строках, однотипных по структуре. Каждая строка таблицы содержит данные о конкретном экземпляре сущности и называется *записью*.

Для однозначного определения (*идентификации*) каждой записи таблица должна иметь *уникальный (первичный) ключ*. По значению ключа таблицы отыскивается единственная запись в таблице. Ключ может состоять из одного или нескольких полей таблицы. Значение уникального ключа не может повторяться в нескольких записях.

Логические связи между таблицами дают возможность объединять данные из разных таблиц. Связь каждой пары таблиц обеспечивается одинаковыми полями в них — *ключом связи*. Таким образом, обеспечивается рациональное хранение не дублированных данных и их объединение в соответствии с требованиями решаемых задач.

В нормализованной реляционной базе данных связь двух таблиц характеризуется отношениями записей типа "один-к-одному" (1 : 1) или "один-ко-многим" (1 : M). Отношение 1 : 1 предполагает, что каждой записи одной таблицы соответствует одна запись в другой. Отношение 1 : M предполагает, что каждой записи первой таблицы соответствует много записей во второй, но каждой записи второй таблицы соответствует только одна запись в первой.

Для двух таблиц, находящихся в отношении типа 1 : M, связь устанавливается по уникальному ключу таблицы, представляющей в отношении сторону "один", — *главной таблицы* в связи. Во второй таблице, представляющей в отношении сторону "многие" и называемой *подчиненной*, этот ключ связи может быть либо частью уникального ключа, либо не входить в состав ключа. В подчиненной таблице ключ связи называется еще *внешним ключом*.

На рис. 1.1 показаны две таблицы со списком покупателей и перечнем заключенных договоров, которые находятся в отношении типа 1 : M и логически связаны с помощью общего поля (столбца) Код покупателя — ключа связи. Это поле является уникальным ключом в главной таблице — ПОКУПАТЕЛЬ и неключевым полем в подчиненной таблице — ДОГОВОР.

Размещение сведений о каждой сущности в отдельной таблице и связывание таблиц позволяет избежать повторения описательных данных в разных таблицах. При этом обеспечивается однократный ввод данных при загрузке и корректировке базы данных. Если данные двух таблиц в приведенном примере разместить в одной таблице, то каждая запись должна соответствовать одному договору. Причем данные о покупателе (наименование, ИНН, адрес и др.) будут повторяться во всех записях о договорах одного покупателя, что усложнит ввод, корректировки и обеспечение актуального состояния базы данных. При хранении данных в двух табли-

цах сведения о покупателе хранятся в единственном экземпляре, а в таблице договоров повторяются только значения ключевого поля с кодом покупателя.

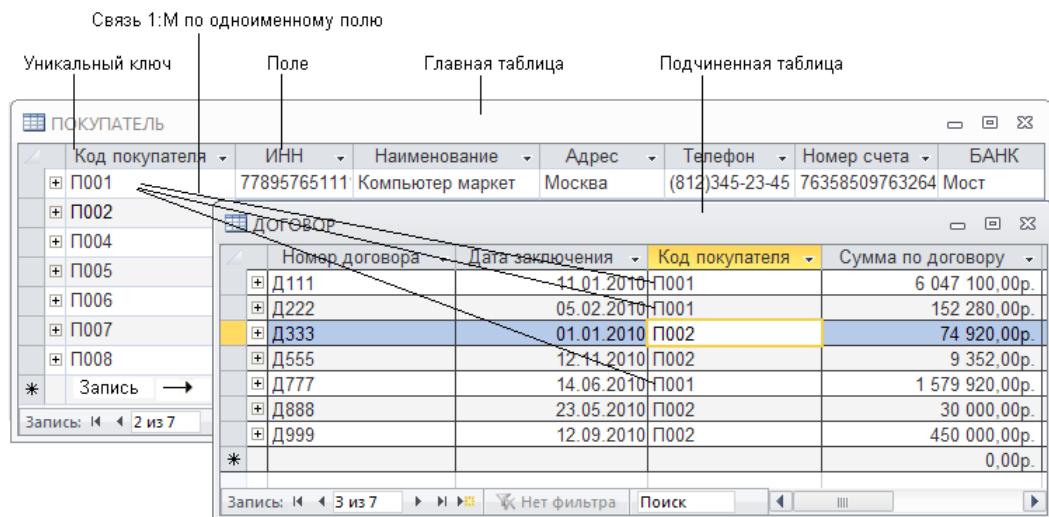


Рис. 1.1. Взаимосвязанные таблицы реляционной базы данных

ПОКУПАТЕЛЬ									
<b>ПОКУПАТЕЛЬ</b>	Код покупателя	ИИНН Наименование Адрес Телефон Номер счета	Подчиненная таблица						
			ДОГОВОР						
			Номер договора	Дата заключения	Сумма по дог.	Код исполнителя			
			Д111	11.01.2010	6 047 100,00р.	6			
			Д222	05.02.2010	152 280,00р.	1			
			Д777	14.06.2010	1 579 920,00р.	1			
			*		0,00р.				
			<b>ПОКУПАТЕЛЬ</b>	Код покупателя	ИИНН Наименование Адрес Телефон Номер счета	Подчиненная таблица			
						ДОГОВОР			
Номер договора	Дата заключения	Сумма по дог.				Код исполнителя			
Д333	01.01.2010	74 920,00р.				3			
Д555	12.11.2010	9 352,00р.				3			
Д888	23.05.2010	30 000,00р.				6			
Д999	12.09.2010	450 000,00р.				7			
*		0,00р.							
П004	45657567567	Монитор				Санкт-Петербург	( )123-45-67	585	
П005	77124356782	Компьютер лэнд	Саратов	( )123-56-23	597				
П006	58787987912	Компьютерная техника	Ярославль	( )123-45-67	763				
Запись: 1 из 7 Нет фильтра Поиск									

Рис. 1.2. Отображение в записях главной таблицы связанных записей подчиненной таблицы

В Access реализовано средство просмотра и редактирования связанных записей нескольких таблиц. При этом данные отображаются в иерархическом виде. При

раскрытии одного уровня иерархии рядом с записью главной таблицы отображаются связанные записи подчиненной. Для записи подчиненной таблицы также могут быть открыты связанные записи и т. д. Например, для таблиц ПОКУПАТЕЛЬ, ДОГОВОР (рис. 1.2), связанных отношением 1 : M, для каждой записи таблицы ПОКУПАТЕЛЬ могут быть отображены и отредактированы связанные записи в таблице ДОГОВОР.

## Схема данных

В СУБД Access процесс создания реляционной базы данных включает создание *схемы данных*. Схема данных наглядно отображает логическую структуру базы данных: таблицы и связи между ними, а также обеспечивает использование установленных в ней связей при обработке данных.

Для нормализованной базы данных, основанной на одно-многозначных и одно-однозначных отношениях между таблицами, в схеме данных для связей таких таблиц по первичному ключу или уникальному индексу главной таблицы могут устанавливаться параметры обеспечения *связной целостности*.

При поддержании целостности взаимосвязанных данных не допускается наличия записи в подчиненной таблице, если в главной таблице отсутствует связанная с ней запись. Соответственно при первоначальной загрузке базы данных, а также корректировке, добавлении и удалении записей система допускает выполнение операции только в том случае, если она не приводит к нарушению целостности.

Связи, определенные в схеме данных, автоматически используются для объединения таблиц при разработке многотабличных форм, запросов, отчетов, существенно упрощая процесс их конструирования.

В схеме данных связи могут устанавливаться для любой пары таблиц, имеющих одинаковое поле, позволяющее объединять эти таблицы.

## Объекты Access

База данных Access включает следующие сохраняемые в одном accdb-файле объекты:

- ❖ *таблицы, запросы, схемы данных*, непосредственно имеющие отношение к базе данных;
- ❖ *формы, отчеты, макросы и модули*, называемые объектами приложения.

Формы и отчеты предназначены для типовых процессов обработки данных — просмотра, обновления, поиска по заданным критериям, получения отчетов. Эти объекты приложений конструируются из графических элементов, называемых элементами управления. Основные элементы управления служат для отображения полей таблиц, являющихся источниками данных объекта.

Для автоматизации доступа к объектам и их взаимодействия используется программный код. Только с помощью программного кода получается полноценное приложение пользователя, функции которого доступны через меню, панели инст-

рументов и формы. Для создания программного кода используются модули на языке VBA и макросы.

Каждый объект и элемент управления имеет свои свойства, определяя которые, можно настраивать их. С каждым объектом и элементом управления связывается набор событий, которые могут обрабатываться макросами или процедурами обработки событий на VBA, входящими в состав модулей форм, отчетов.

Объекты представлены в области навигации окна базы данных Access. Все операции по работе с объектами собственно базы данных и приложений начинаются в этом окне.

*Таблицы* (Tables) создаются пользователем для хранения данных об одной сущности — одном информационном объекте модели данных предметной области. Таблица состоит из полей (столбцов) и записей (строк). Каждое поле содержит одну характеристику информационного объекта предметной области. В записи собраны сведения об одном экземпляре информационного объекта.

База данных Access может включать до 32 768 объектов (в том числе формы, отчеты и т. д.). Одновременно может открываться до 2048 таблиц.

*Запросы* (Queries). Запросы на выборку служат для выборки нужных данных из одной или нескольких связанных таблиц. Результатом выполнения запроса является виртуальная таблица. В запросе можно указать, какие поля исходных таблиц следует включить в запись таблицы запроса и как отобрать нужные записи. Таблица запроса может быть использована наряду с другими таблицами базы при обработке данных. Запрос может формироваться с помощью конструктора запросов или инструкции языка SQL. Запросы на изменение позволяют обновлять, удалять или добавлять данные в таблицы, а также создавать новые таблицы на основе существующих.

*Схема данных* (Relationships) определяет, с помощью каких полей таблицы связываются между собой, как будет выполняться объединение данных этих таблиц, нужно ли проверять связную целостность при добавлении и удалении записей, изменении ключей таблиц. Схемы данных в области навигации в окне базы данных отображаются только в проектах Access, работающих с базами данных сервера. Для отображения схемы данных в базах данных Access используется команда **Схема данных** (Relationships), размещенная на вкладке ленты **Работа с базами данных** (Database Tools) в группе **Отношения** (Relationships).

*Формы* (Forms) являются основным средством создания диалогового интерфейса приложения пользователя. Форма может создаваться для работы с электронными документами, сохраняемыми в таблицах базы данных. Вид таких документов может соответствовать привычному для пользователя бумажному документу. Форма используется для разработки интерфейса по управлению приложением. Включаемые в форму процедуры обработки событий позволяют управлять процессом обработки данных в приложении. Такие процедуры хранятся в модуле формы. В формы могут вставляться рисунки, диаграммы, звуковые фрагменты, видео. Возможна разработка форм с набором вкладок, с каждой из которых связано выполнение той или иной функции приложения.

*Отчеты* (Reports) предназначены для формирования на основе данных базы выходных документов любых форматов, содержащих результаты решения задач пользователя, и вывода их на печать. Как и формы, отчеты могут включать процедуры обработки событий. Использование графических объектов позволяет дополнять данные отчета иллюстрациями. Отчеты обеспечивают возможность анализа данных при использовании фильтрации, агрегирования и представления данных источника в различных разрезах.

*Макросы* (Macros) являются программами, состоящими из последовательности макрокоманд, которая выполняется по вызову или при наступлении некоторого события в объекте приложения или его элементе управления. Макросы позволяют автоматизировать некоторые действия в приложении пользователя. Создание макросов осуществляется в диалоговом режиме путем выбора нужных макрокоманд и задания параметров, используемых ими при выполнении. В Access 2010 обновлен конструктор макросов. Его новые возможности упрощают создание, редактирование макросов, позволяют сокращать количество ошибок кода и более эффективно создавать надежные приложения. В Access 2010 появилась новая возможность — макросы данных, позволяющие изменять данные на основе событий в исходных таблицах. Макросы данных используются для добавления логики к данным и со средоточения ее в исходных таблицах. В Web-приложениях Access, базирующихся на базах данных, опубликованных в SharePoint, для программирования необходимо использовать только макросы, так как код VBA несовместим со средствами Web-публикации.

*Модули* (Modules) содержат процедуры на языке Visual Basic for Applications. Могут создаваться процедуры-подпрограммы, процедуры-функции, которые разрабатываются пользователем для реализации нестандартных функций в приложении пользователя, и процедуры для обработки событий. Использование процедур позволяет создать законченное приложение, которое имеет собственный графический интерфейс пользователя, позволяющий запросить выполнение всех функций приложения, обработать все ошибки и нестандартные ситуации.

В Access для удобства пользователя объекты в области навигации базы данных могут быть объединены в пользовательские группы по функциональному или иному признаку. Группы содержат ссылки на объекты базы данных различных типов. Группы, в свою очередь, объединяются в категории. С помощью такой организации объектов базы данных может быть разработан интерфейс пользовательского приложения, полностью заменяющий существующие ранее кнопочные формы и обеспечивающий доступность только к категориям и группам, наглядно и понятно представляющим функциональность приложения.

## Сводные таблицы и сводные диаграммы

Сводная таблица представляет собой интерактивную таблицу, с помощью которой можно анализировать данные, быстро объединяя большие объемы данных и рассчитывая итоги (рис. 1.3). С помощью сводных таблиц выполнение сложного анализа данных делается просто.

Наименование покупателя ▾		Номер договора ▾					
Компьютер маркет		Перспектива		Общие итоги			
Д111	Д222	Итоги	Д333	Итоги			
+/-	+/-	+/-	+/-	+/-	+/-		
Наименование товара ▾	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"
CD-ROM Panasonic IDE	4	16	20				20
DIMM 64М PC100	5		5	1	1		6
FDD 3,5		15	15				15
HDD Maxtor 20GB		8	8	3	3		11
Зв. Карта Genius Liv		10					13
Монитор 17LG	2	7					13
СканерAcer		10					22

Значение: 8  
Итог: Сумма "Количество отгружено"  
Компонент строки: HDD Maxtor 20GB  
Компонент столбца: Компьютер маркет - Итоги  
Фильтр: Дата отгрузки по месяцам = 2007

Рис. 1.3. Сводная таблица для анализа суммарного количества отгруженного по любому из товаров, по различным покупателям и договорам, по всем или некоторым месяцам, кварталам, годам

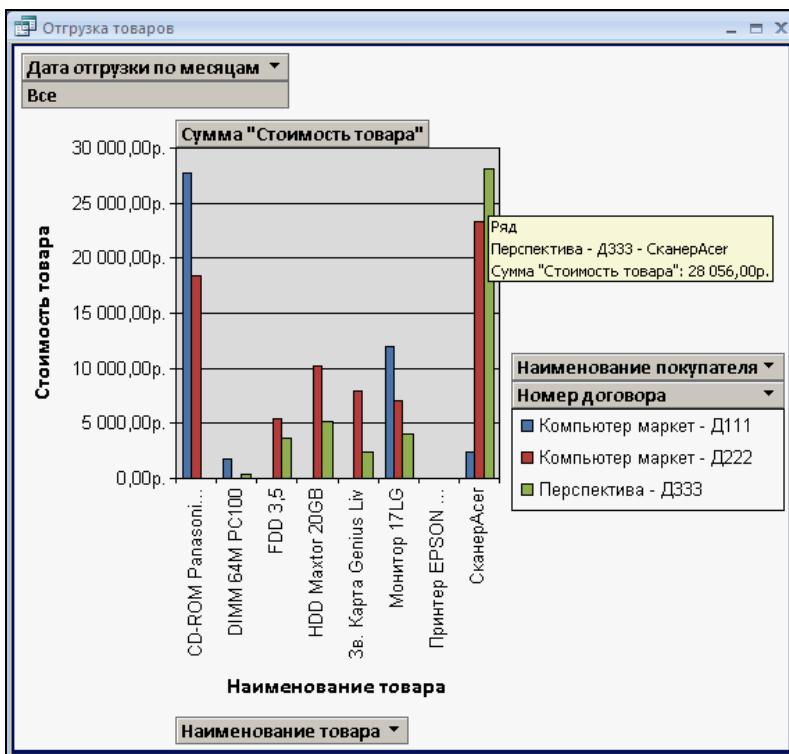


Рис. 1.4. Сводная диаграмма для анализа суммарного количества отгруженного по любому из товаров, по различным покупателям и договорам, по всем или некоторым месяцам, кварталам, годам

Для получения различных итогов по исходным данным достаточно в созданном макете сводной таблицы выбирать значения в поле строк (**Наименование товара ▼**), поле столбцов (**Наименование покупателя ▼**, **Номер договора ▼**) и поле страницы (фильтра) (**Дата отгрузки по месяцам ▼**).

Сводные таблицы позволяют динамически изменять макет для всестороннего анализа данных. Существует возможность изменять заголовки строк, столбцов, а также полей, определяющих страницу. Создавать и быстро модифицировать макет можно, выбирая и перетаскивая поля из раскрывающегося списка полей в рабочую область. При каждом изменении макета сводная таблица немедленно выполняет вычисления заново в соответствии с новым расположением данных.

Сводные диаграммы служат для наглядного графического представления анализируемой информации, облегчая для пользователей сравнение и выявление тенденций и закономерностей в данных (рис. 1.4).

Access позволяет открывать в режимах сводной таблицы и сводной диаграммы таблицы, запросы и формы.

Таким образом, источником записей для режима сводной таблицы и режима сводной диаграммы может быть не только таблица, но и базовый источник данных для формы. В базе данных Access источником записей может быть таблица, запрос или инструкция SQL; в проекте Access — таблица, представление, инструкция SQL, хранящая процедура или табличная функция.

## Размещение базы данных

Все таблицы, а также другие объекты базы данных Access — запросы, формы, отчеты, макросы и модули, построенные для этой базы, и внедренные объекты могут размещаться на диске в одном файле формата ACCDB. Это упрощает технологию ведения базы данных и приложения пользователя. Обеспечивается высокая компактность размещения всех объектов базы данных на диске и эффективность обработки данных.

Когда база данных открыта, для корректного внесения изменений требуются блокировки данных разных уровней. Контроль над ними осуществляется с помощью файла блокировки. Если в MS Access 2010 открыт mdb-файл, для контроля блокирования создается файл с расширением ldb и тем же именем, что у mdb-файла. Для файлов в формате ACCDB блокирование управляет файлом с расширением laccdb. Как ldb-файлы, так и laccdb-файлы уничтожаются автоматически, когда база данных будет закрыта всеми пользователями.

Введение отдельных блокирующих файлов для файлов Access 2010 и файлов, созданных в более ранних версиях Access, обеспечивает одновременное открытие в Access 2010 файлов mdb и accdb с одинаковым именем, и это не приведет к возникновению конфликтов в блокирующем файле, поскольку будут созданы два разных блокирующих файла. Также можно открывать один и тот же файл mdb в Access 2010 и в более ранней версии Access одновременно, обе версии используют один и тот же блокирующий файл ldb.

База данных Access 2010 может быть превращена в базу данных, доступную только для выполнения, не доступную для изменений и скрывающую свой код. Для этого она должна быть скомпилирована и сохранена в файле формата ACCDE, заменившем файлы с расширением mde предыдущих версий. В процессе преобразования из базы данных удаляется весь исходный текст программ на VBA, база сжимается, что значительно сокращает размер файла. В базе данных формата ACCDE код VBA может только выполняться, но просматривать и изменять его нельзя. При этом у пользователей нет разрешений на изменение структуры форм, отчетов или модулей. Для преобразования файла базы данных ACCDB в формат ACCDE на вкладке **Файл** (File) нажмите кнопку **Сохранить и опубликовать** (Save & Publish) и выберите в списке **Сохранить базу данных как** (Save Database As) пункт **Создать ACCDE** (Make ACCDE) и в окне **Сохранить как** (Save As) нажмите кнопку **Сохранить** (Save).

Проекты Access, являясь клиентскими приложениями пользователя, позволяют подключаться к базам данных SQL Server, размещенным на вашем компьютере или в сети. Проект размещается в adp-файле на компьютере пользователя. В проекте пользователь может создавать базу данных на сервере SQL или использовать существующую. Файл проекта, как и файл базы данных Access, может быть преобразован в исполняемый файл, который приобретет расширение ade.

Начиная с Access 2007 появилось новое расширение файлов — accdr, позволяющее открывать базу данных в режиме выполнения. С помощью простой замены расширения файла базы данных с accdb на accdr можно создать исполняемую версию базы данных Access 2007, закрытую для изменений. Чтобы восстановить полную функциональность, можно просто вернуть файлу старое расширение — accdb.

## Шаблоны баз данных

Сразу после запуска Access отображается новый компонент пользовательского интерфейса Access 2010 — представление Backstage. Представление Backstage — это место, где можно управлять файлами. В момент открытия здесь наряду с командами **Создать** (New), **Открыть** (Open) базу данных, установить **Параметры** (Options) и списком последних использованных баз данных, представлены многочисленные шаблоны для создания различных типовых баз данных (см. рис. 1.8). Для открытой базы данных при выборе вкладки **Файл** (File) отображается представление Backstage с командами, применимыми ко всей базе данных, но естественно шаблоны баз данных при этом не отображаются.

После закрытия базы данных в окне представления Backstage восстанавливается библиотека готовых шаблонов для приложений баз данных. Шаблоны типовых баз данных включают все необходимые таблицы, формы, запросы и отчеты, для предметных областей различных сфер деловой и личной жизни. Эти стандартные приложения можно использовать без какой-либо модификации и настройки, либо взять их за основу и адаптировать в соответствии с характером информации, которую требуется сохранять и обрабатывать. Выбрав нужный шаблон, достаточно оп-

ределить место, где следует сохранить создаваемую базу данных, и нажать кнопку **Создать** (Create). Автоматически создается и открывается база данных, соответствующая выбранному шаблону, и пользователю остается только ввести данные.

Типовые базы данных позволяют начинающему пользователю познакомиться с основными принципами построения базы данных и приложения пользователя и получить навыки практической работы в среде Access. Работая с типовой базой, пользователь научится просматривать и изменять данные через формы, делать запросы для получения сведений из связанных таблиц, готовить отчеты.

Новые шаблоны приложений можно загрузить с сайта [Microsoft Office.com](http://Microsoft Office.com). В разделе шаблонов Office предлагается на выбор множество разнообразных шаблонов для простых и сложных задач, которые можно использовать в коммерческих целях и для собственного использования.

Просматривать шаблоны из Интернета так же легко, как если бы они находились на локальном компьютере. Имеющиеся шаблоны упорядочены по логическим категориям для удобства обзора. При выборе шаблона для просмотра отображается размер шаблона, приблизительная длительность загрузки и, если необходимо, дополнительные системные требования. Одним щелчком можно загрузить шаблон и открыть его.

Шаблоны приложений Access 2010 позволяют быстрее создавать приложения и приступать к работе с ними — даже если пользователь не имеет навыков проектирования баз данных и не знает языков программирования.

Однако, используя типовую базу данных, трудно рассчитывать, что она в полной мере удовлетворит потребности пользователя. Базу данных, созданную по шаблону, можно изменить и расширить, но эта работа требует от пользователя практически тех же знаний, что и создание новой базы данных.

## Мастера Access

Access располагает разнообразными диалоговыми средствами, которые позволяют создавать объекты базы данных и приложения, не прибегая к программированию.

Множество мастеров Access позволяет автоматизировать процесс создания таблиц базы данных, запросов, форм, отчетов; анализировать таблицы базы данных и выполнять многие другие работы. Практически для любых работ имеется *мастер* (*Wizard*), который поможет их выполнить.

*Мастера по созданию форм и отчетов* упрощают и ускоряют процесс создания однотабличных и многотабличных форм и отчетов. Так, выбрав таблицу или запрос в области навигации базы данных, одним щелчком можно создать форму, отображающую поля только одной записи в столбец или отображающую несколько записей в виде таблицы или смешанное представление записей. В диалоге с другим мастером пользователю достаточно выбрать таблицы и поля, которые необходимо включить в форму, выбрать источник основной и подчиненной части формы. Мастера по разработке форм и отчетов автоматически создают инструкцию SQL, опи-

сыгающую источник записей для формы или отчета, что избавляет пользователя от подготовки запроса.

*Мастер подстановок* (Lookup Wizard) создает в поле таблицы или формы раскрывающийся список значений из полей другой таблицы или запроса для выбора и ввода в поле нужного значения. Созданные в полях таблиц списки наследуются при включении этих полей в форму (поле со списком).

*Мастера запросов* позволяют создавать простые запросы на выборку или запросы на выборку, в которых выполняются групповые операции над данными из одной или нескольких таблиц.

*Мастер перекрестных запросов* (Crosstab Query Wizard) формирует из взаимосвязанных таблиц или запросов базы данных таблицу, подобную электронной, в который одно поле используется в качестве заголовков строк, второе — столбцов, а на их пересечении размещаются итоговые данные, рассчитываемые по значениям третьего поля.

*Мастер по созданию диаграмм* (Chart Wizard) обеспечивает создание в формах и отчетах диаграмм, базирующихся на данных в таблицах или запросах. На рис. 1.5 приведена форма, позволяющая просматривать в справочнике данные о товаре и синхронно отображать диаграмму с количеством заказанного по договорам (в примере Д111, Д222, Д333). Вызывается мастер с помощью элемента управления **Диаграмма** (Chart), расположенного на вкладке ленты конструктора форм в группе **Элементы управления** (Controls). Ответив на ряд элементарных вопросов, легко можно получить диаграмму, связанную с выбранными в форме данными.

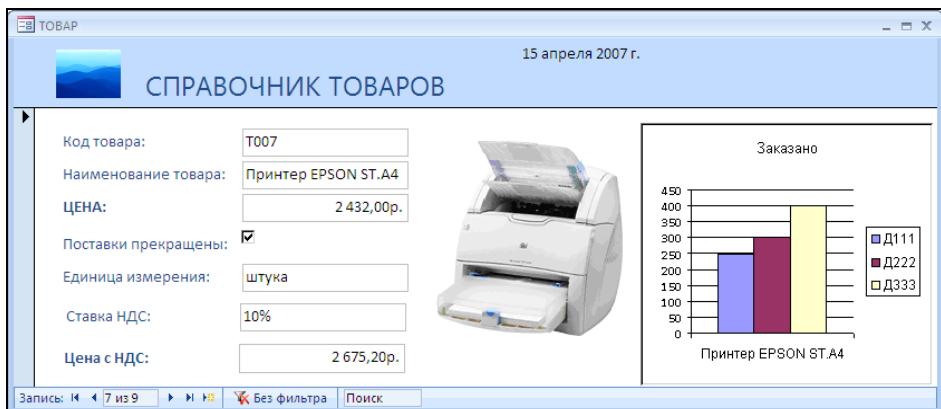


Рис. 1.5. Форма с диаграммой, построенной мастером

После добавления диаграммы или открытия существующей диаграммы с помощью двойного щелчка мышью может быть выполнен переход к ее редактированию с помощью Microsoft Graph.

*Мастер кнопок* (Command Button Wizard и Page Command Button Wizard) создает командные кнопки — элементы управления в форме. При построении кнопки мастер предлагает большой набор действий, которые могут быть выполнены при

щелчке на кнопке. Для выбранного действия мастер создает встроенный макрос и связывает его с событием **Нажатие кнопки** (On Click). Формы с командными кнопками, связанными с макросами, позволяют реализовать удобный интерфейс пользователя для управления приложением.

*Мастер по анализу таблиц* (Table Analyzer Wizard) позволяет выполнить нормализацию данных базы. Мастер разделяет ненормализованную таблицу с дублированными данными на две или несколько таблиц меньшего размера, в которых данные сохраняются без повторений. Вызывается мастер с помощью элемента управления **Анализ таблицы** (Table Analyzer), расположенного на стандартной вкладке ленты **Работа с базами данных** в группе **Анализ** (Analyzer).

*Мастер сводных таблиц и диаграмм* (PivotTable Wizard) позволяет открыть существующие таблицы, запросы, формы в режиме сводной таблицы и создать макет.

*Служебная программа* (Database Utilities) *Преобразовать базу данных* (Convert Database), которая ранее позволяла конвертировать базу данных из предыдущих версий в текущую и, наоборот в Access 2010, заменена командами **Сохранить базу данных как** (Save Database As), доступными в разделе **Сохранить и опубликовать** (Save & Publish) на вкладке **Файл** (File). Если база данных Access 2007/2010 использует возможности, не поддерживаемые более ранними версиями, выполнение преобразования невозможно.

*Служебная программа* (Database Utilities) *Сжать и восстановить базу данных* (Compact & Repair Database) заменена одноименной командой, доступной в разделе **Сведения** (Info) на вкладке **Файл** (File). Команда реорганизует базу данных на диске с целью улучшения объемно-временных характеристик. Перезаписывает фрагментированный в результате удаления и добавления объектов файл базы данных и сокращает его размер, а также восстанавливает поврежденную базу данных.

## Средства конструирования объектов

Для автоматизации процесса создания объектов базы данных — таблиц, схемы базы данных, запросов — и объектов приложения — форм, отчетов, макросов используются специализированные диалоговые графические средства, называемые **конструкторами** (Design). Конструктор предоставляет пользователю набор элементов, с помощью которых быстро создается и модифицируется объект. Для конструирования макета формы, отчета используются **элементы управления**, которые доступны в режиме макета и режиме конструктора на вкладках ленты **Работа с макетами форм/отчетов** (Form Layout Tools) или **Инструменты конструктора форм/отчетов** (Form Design Tools).

Дополнительные возможности по изменению структуры форм и отчетов в режиме макета были включены уже в Access 2007. В режиме макета форма или отчет пользователь, изменяя макет, одновременно видит данные почти в том же виде, в каком они отобразятся в режиме формы или после вывода отчета на печать. Режим макета представляет собой наиболее наглядный режим для изменения макета форм и отчетов. Его можно использовать для внесения практически любых изменений.

Поскольку при внесении изменений можно просматривать данные, в этом режиме очень удобно задавать размеры элементов управления и выполнять другие задачи, связанные с внешним видом и удобством формы или отчета. Некоторые задачи невозможно выполнить в режиме макета и тогда необходимо переключиться в режим конструктора. Access выдает сообщение о том, что для внесения изменений следует переключиться в режим конструктора.

В Access 2010 формирование источника данных для формы или отчета может быть выполнено с помощью *списка доступных полей*. При этом оно сводится к простому перетаскиванию полей из таблиц базы данных в нужное место формы/отчета в режиме конструктора или макета.

В проекте Access, связанном с сервером баз данных MS SQL Server, конструкторы таблиц, представлений, хранимых процедур, определяемых пользователем функций позволяют работать с объектами базы данных сервера в простом привычном для пользователя локальных баз данных графическом режиме.

Для упрощения внесения изменений в объекты базы данных Access предлагает технологию *интеллектуальной замены имен объектов в базе данных*. При этом автоматически исправляются ошибки, вызванные переименованием таблиц, полей, форм, отчетов, запросов, текстовых блоков или других элементов управления. Реализуется эта технология за счет того, что каждый именуемый объект (или элемент) базы данных имеет внутренний уникальный идентификатор, а имя является только псевдонимом. При переименованиях изменяется лишь псевдоним, и при необходимости в других объектах корректируются все ссылки на объект. Для применения этой технологии следует установить параметры **отслеживать автозамену имен** (Track name AutoCorrect info) и **выполнять автозамену имен** (Perform name AutoCorrect) в окне **Параметры Access** (Access Options) на вкладке **Текущая база данных** (Current Database) в разделе **Параметры автозамены имен** (Name AutoCorrect Options). Окно **Параметры Access** (Access Options) открывается соответствующей командой в представлении Backstage, открываемом на вкладке **Файл** (File).

## Средства программирования

Наряду с диалоговыми средствами создания объектов базы данных и объектов приложения, которые позволяют решить многие задачи пользователя, Access располагает мощными средствами программирования. Эти средства могут использоваться как для доработки приложений, созданных диалоговыми средствами, так и для реализации сложных задач и создания приложений с необходимым пользователю интерфейсом в целом. Без использования программного кода практически невозможно получить законченное автоматизированное приложение пользователя.

Одним из средств программирования в Access является язык макрокоманд. Программы, созданные на этом языке, называемые *макросами*, позволяют легко связывать отдельные действия, реализуемые с помощью форм, запросов, отчетов. Макросы управляются событиями, которые вызываются действиями пользователя

при диалоговой работе с данными через формы, например, нажатие кнопки, изменение данного в поле, или системой.

Простой язык макрокоманд и новый конструктор макросов с ясным и понятным интерфейсом, поддерживающий функции IntelliSense, позволяют при малой трудоемкости интегрировать объекты приложения и организовать процесс обработки данных. Конструктор макросов упрощает создание, редактирование и автоматизацию логики базы данных, позволяет сокращать количество ошибок кода.

Помимо обычных макросов новый конструктор макросов позволяет создавать макросы данных, которые являются новшеством в данной версии.

Макросы данных связываются с событиями в таблице и позволяют выполнять определенные действия при изменении, вставке или удалении записи. Например, с помощью макросов можно проверять данные или выполнять вычисления. Макросы данных позволяют добавлять логику к данным и сосредотачивать логику в исходных таблицах. По сути макросы данных позволяют реализовать триггеры в любой базе данных Access 2010.

С помощью усовершенствованного конструктора макросов и макросов данных можно распространить автоматизацию за пределы клиентского приложения Access на веб-базы данных SharePoint и другие приложения, обновляющие таблицы Access.

Наряду с языком макрокоманд Access включает развитую интегрированную среду объектно-ориентированного программирования Visual Basic for Applications (VBA), позволяющую реализовать любые программные решения. Программы на VBA реализуются процедурами, которые объединяются в объектах, называемых *модулями*.

В VBA база данных рассматривается как совокупность объектов (таблиц, форм, отчетов, их элементов и т. д.), имеющих свойства и методы, реализующие заранее определенные действия над объектами. Структурированность объектов базы данных упрощает освоение этого языка и создание приложений. Управление выполнением программ в диалоговых приложениях VBA осуществляется событиями, вызываемыми действиями пользователя или системы.

Среда VBA объединяет разнообразные наглядные графические инструменты: редактор VBA, окно разрабатываемого проекта, окно свойств объектов проекта, окно просмотра объектов, отладчик и др. Все инструменты унифицированы и являются общими для всех приложений Microsoft Office, для Visual Basic, а также продуктов ряда других фирм.

Приложения, разрабатываемые на VBA, могут выполняться только в той среде, в которой поддерживается VBA, в то время как Visual Basic ориентирован на полностью самостоятельную разработку автономно выполняющихся приложений. Язык VBA является производным от самостоятельной системы программирования Visual Basic и имеет с ним много общего. Их синтаксис и интерфейс практически одинаковы.

Заметим, что код VBA несовместим со средством веб-публикации, поэтому если планируется опубликовать приложение как веб-приложение Access для выполнения задач программирования, необходимо использовать только макросы.

## Интеграция и использование внешних данных

Access продолжает поддерживать технологию OLE (Object Linking and Embedding, связь и внедрение объектов), обеспечивающую возможность интеграции данных различных приложений в составном документе. С помощью OLE пользователь может внедрять объекты другого приложения в базу данных или устанавливать с ними связи. Активизация внедренного объекта запускает программу, которая его создала, и пользователь может изменить объект. При установлении связи с объектом он по-прежнему сохраняется в файле другого приложения, а не в базе данных. За счет этого он может обновляться независимо, а в базе данных всегда будет представлена последняя версия объекта. Надо иметь в виду, что при изменении местоположения файла связь с ним должна обновляться пользователем.

Внедряемыми или связываемыми объектами могут быть документы различных приложений Windows — рисунки, графики, электронные таблицы, звуковые или видеофайлы. Например, в таблице наряду с обычными реквизитами, характеризующими информационный объект, может храниться любая графическая информация о нем — схемы, чертежи, диаграммы, рисунки и т. п. Таким образом, в Access расширяется традиционное понятие данных, хранимых в базе и представляющих информационные объекты.

В Access 2007/2010 обеспечивается хранение одного или нескольких файлов разных типов — документов Word, презентаций PowerPoint, изображений и т. п. — в поле записи базы данных, имеющем тип данного **Вложение** (Attachment). Вложения позволяют хранить данные более рационально.

Access может использовать данные различных внешних источников. Внешними источниками данных могут служить таблицы других баз данных Access, dBase, Paradox, Oracle и Microsoft SQL Server, электронные таблицы Microsoft Excel, Lotus 1-2-3, таблицы и списки HTML и файлы XML, списки Windows SharePoint Services, текстовые файлы и др., находящиеся на локальном компьютере или на сервере в локальной корпоративной или интернет-сети.

Для взаимодействия с внешними источниками данных применяется специальное программное обеспечение — интерфейсы. Широко используемыми интерфейсами являются ODBC (Open Database Connectivity, открытый интерфейс подключения к базам данных), который служит, прежде всего, для доступа к базам данных, и интерфейсы модели составного объекта (Component Object Model), называемые OLE DB, разработанные как средство универсального доступа к данным по сети. OLE DB позволяет подсоединяться к источникам данных многих типов, в том числе реляционным источникам данных, почтовым файлам, неформатированным текстовым файлам и электронным таблицам.

К источникам данных ODBC могут отправляться запросы, таблицы таких источников данных могут использоваться в базе данных Access, как связанные. Связанные таблицы отображаются в области навигации базы данных, оставаясь в исходном файле вне файла Access. Их можно использовать при создании запросов, форм, отчетов, причем можно объединять их данные с данными из локальных объектов Access. Связывание позволяет использовать данные другой программы, не

импортируя их. В этом случае можно просматривать и изменять данные, как в исходной программе, так и в базе данных Access.

Данные внешних источников могут импортироваться в базу данных Access.

На вкладке **Внешние данные** (External Data) в группе **Импорт и связи** (Import & Link) представлены доступные для этой операции источники данных. Импорт и связывание могут быть выполнены для данных из таких источников как Access, Excel, база данных ODBC, текстовый файл, XML-файл, список SharePoint, документ HTML, папка Outlook, файл dBase.

Возможен экспорт таблиц, запросов, форм и отчетов, выделенной части объекта в режиме таблицы из базы данных Access в форматы других приложений. Операции экспорта представлены в одноименной группе на вкладке **Внешние данные** (External Data).

Выполнение операций импорта или связывания данных, как и экспорта данных из базы для большинства форматов, требует лишь указания, где расположены данные, и выбора способа их хранения в базе данных.

## Поддержка технологий корпоративных сетей

Корпоративные сети являются сетями уровня предприятия, которые базируются на клиент-серверных и интернет-технологиях. Эти сети могут подключаться или не подключаться к Интернету (рис. 1.6).

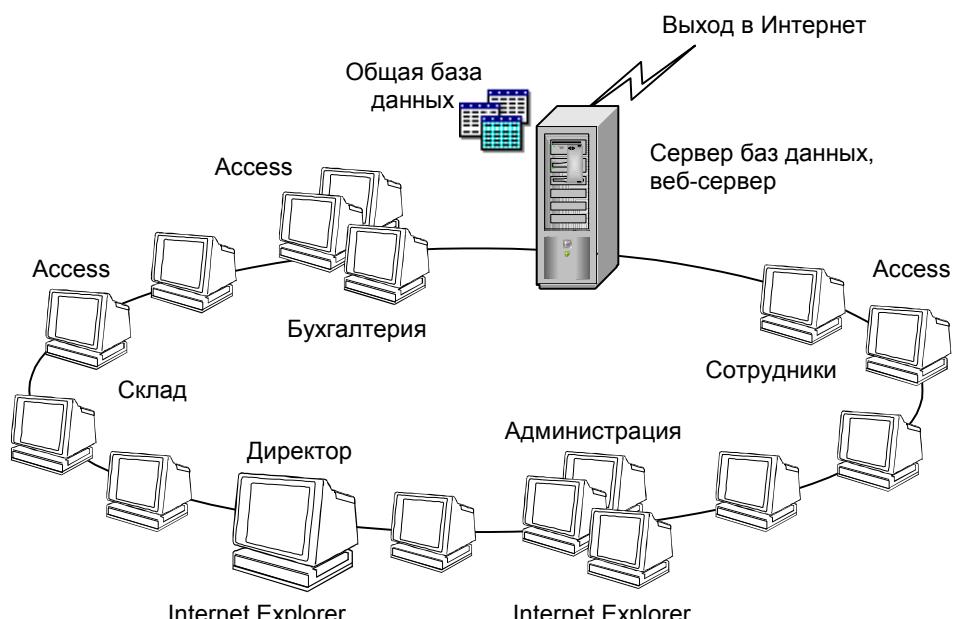


Рис. 1.6. Корпоративная сеть с SQL- и веб-серверами

Технологии Интернета позволяют получить доступ к информации всего предприятия со своего рабочего места, не заботясь о совместимости аппаратных и программных платформ, используя обычную программу просмотра — обозреватель Internet Explorer.

Access включает развитые средства, ориентированные на клиент-серверные технологии, которые позволяют создавать клиентские приложения для работы с общими базами данных SQL Server.

Для организации совместной работы с данными и взаимодействия пользователей с помощью единой веб-среды могут быть использованы служба Windows SharePoint Services, специальные технологии взаимодействия и коммуникаций, которые легко интегрируются с приложениями Microsoft Office. Служба Windows SharePoint Services является компонентом Microsoft Windows Server. Обеспечивает пользователей набором инструментов для организации информации, управления документами и эффективного взаимодействия.

С помощью Windows SharePoint Services можно создавать веб-сайты для хранения общих данных и совместной работы с ними. Доступ пользователей к содержимому Web-сайтов осуществляется через веб-обозреватель или приложения Microsoft Office из любой точки корпоративной сети.

Access предоставляет разнообразные простые и удобные средства размещения данных из баз на одном или нескольких сайтах SharePoint для доступа к ним пользователей корпоративной сети, а также средства использования списков SharePoint в приложениях Access.

## **Многопользовательская база данных Access**

База данных, как правило, содержит данные, необходимые многим пользователям. Создание многопользовательской базы данных Access и получение одновременного доступа нескольких пользователей к общей базе данных возможно в одноранговой сети персональных компьютеров или в сети с файловым сервером.

Под одноранговой понимается сеть, каждый компьютер которой может предоставлять остальным подключенным к сети компьютерам доступ ко всем или некоторым своим ресурсам, т. е. являться сервером и рабочей станцией одновременно. Одноранговая сеть может управляться встроенной сетевой операционной системой Windows 9x или Windows NT/2000/XP/Vista/Windows 7.

Сети больших масштабов используют выделенные файловые серверы. В сети, поддерживающей концепцию файлового сервера, база данных Access размещается на компьютере, выделенном в качестве файлового сервера. СУБД Access может быть установлена или на файловом сервере, или на каждой рабочей станции, но выполняется она всегда на рабочей станции пользователя. Обработка данных базы в обоих случаях также осуществляется на рабочих станциях пользователей. Поэтому по сети передаются с сервера на рабочие станции большие объемы данных, что сильно загружает ее и делает невозможным одновременное обслуживание большого числа пользователей.

Работа локальной сети с файловым сервером обеспечивается рядом сетевых операционных систем. Наиболее популярными являются Microsoft Windows Server и Novell NetWare. ОС Windows имеют версию Server, предназначенную для управления файловым и другими серверами сети, и версию 2000/XP/Vista/Windows 7, которая устанавливается на рабочей станции и под управлением которой могут выполняться различные приложения, в том числе и Access.

Сеть обеспечивает аппаратную и программную поддержку обмена данными между компьютерами. Сетевая операционная система обеспечивает защиту базы данных, размещенной в сети, предоставляя пользователям доступ к определенным сетевым ресурсам, например папкам, файлам только по предъявлении имени и пароля.

Access не следит за разграничением доступа разных пользователей к базе данных, но автоматически обеспечивает защиту данных от одновременной их корректировки несколькими пользователями. Для обеспечения защиты данных от одновременной их корректировки несколькими пользователями в Access предусматривается блокировка на уровне страниц, при которой блокируются все записи одной страницы. Дополнительно к блокировке страничного уровня Access поддерживает блокировку на уровне записи. Блокирование не допускает изменения записи другими пользователями, пока текущее изменение не будет закончено.

Выбор режима блокировки на уровне записей по умолчанию в текущей базе данных производится параметром **Открытие баз данных с использованием блокировки на уровне записей** (Open databases using record level locking) в окне **Параметры Access** (Access Options) на вкладке **Параметры клиента** (Client Settings) в группе **Дополнительно** (Advanced). Если флажок этого параметра сброшен, по умолчанию будет использована блокировка на уровне страницы. Выбранный вариант применяется к данным в формах, таблицах и программах, использующих объект Recordset для перебора записей. Этот параметр не применяется к запросам на изменение или программам, выполняющим массовые операции с использованием инструкций SQL.

## Работа Access с базой данных SQL Server

Работа с общей базой данных в сети с файловым сервером становится неэффективной уже при одновременной работе 15 пользователей. На обеспечение эффективной работы большого числа пользователей с общей базой данных ориентирована технология "клиент-сервер". В этой технологии пользователь-клиент передает со своего компьютера запрос на компьютер сервера, там СУБД обрабатывает запрос и обратно посыпает только результат выполнения запроса. Таким образом, значительно снижается объем передаваемых по сети данных.

Приложение пользователя разрабатывается и выполняется под управлением СУБД Access на компьютере клиента. Общая база данных размещается на мощном компьютере, где функционирует сервер баз данных, управляемый СУБД SQL Server (см. рис. 1.6). Эта СУБД выполняет обработку данных, размещенных на сер-

вере, и отвечает за их целостность и сохранность. Для доступа к данным базы на сервере используется язык структурированных запросов SQL.

Широко известны серверы баз данных — SQL Server фирмы Microsoft и Oracle Server фирмы Oracle. SQL-серверы баз данных являются самыми мощными приложениями для сетевой обработки данных.

Подключение из Access к серверам баз данных SQL может быть осуществлено с помощью драйверов ODBC. Каждому серверу баз данных соответствует свой драйвер ODBC. В комплект поставки MS Access включены драйверы ODBC для MS SQL Server и Oracle SQL Server.

Использование унифициированного языка запросов SQL позволяет работать с одной и той же базой данных сервера разным пользователям из различных приложений. Данные из базы могут получать Access, Excel, FoxPro и многие другие приложения, использующие протокол ODBC, посылая запросы на языке SQL серверу баз данных.

Приложение Access взаимодействует с данными, расположенными на сервере, несколькими способами. Можно посылать на сервер запросы на языке SQL-сервера. С помощью запроса можно получать необходимые данные в виде виртуальной таблицы — таблицы запроса и далее использовать эту таблицу в качестве источника данных в форме или отчете. Можно посылать на сервер обновленные данные, а также создавать или изменять таблицы в базе данных сервера. Кроме того, в локальной базе данных могут быть созданы связанные таблицы, отображающие данные из таблиц SQL-сервера. Эти таблицы могут обрабатываться в приложении наряду с локальными таблицами базы.

В Access реализована возможность создания *приложения-проекта*, в котором хранятся только объекты, составляющие приложение пользователя, а база данных, с которой работает приложение, размещается на SQL-сервере. Причем в проекте Access обеспечивается работа с объектами базы данных (таблицами, представлениями, хранимыми процедурами, схемами данных) в диалоговом режиме через интерфейс, аналогичный интерфейсу SQL-сервера. Проект Access, являющийся клиентом SQL-сервера, позволяет напрямую, не используя ядро баз данных Access Database Engine, подключаться к базам данных Microsoft SQL Server с помощью интерфейсов OLE DB. Из проекта Access посредством OLE DB обеспечивается доступ не только к базам данных SQL Server, но и к базам данных Access, файлам электронной почты и источникам данных многих других типов.

Создается проект на вкладке **Файл** (File) в окне представления Backstage, где при выборе местоположения файла новой базы данных в окне **Файл новой базы данных** (File New Database) задается имя файла и в поле **Тип файла** (Save as type) из списка выбирается **Microsoft Access Проекты (\*.adp)** (Microsoft Access Projects).

MS SQL Server, начиная с версии 2005, предоставляет бесплатную версию SQL Server Express, которая позволяет создать проект, предназначенный для работы с базой данных SQL Server, работая на локальном компьютере, не подключенном к сети. При этом нет необходимости приобретать лицензию и устанавливать сетевой вариант сервера баз данных. Работа этой версии основывается на том же ядре базы

данных, что SQL Server. Созданное с использованием этих версий приложение будет полностью пригодно для работы с базой данных на большом сетевом SQL-сервере. После завершения разработки, переместив базу с локального компьютера на сервер и модифицировав информацию о соединении, проект можно подключить к удаленному SQL-серверу.

Такой подход может быть удобен при разработке проекта для небольшого числа пользователей, когда в дальнейшем предполагается эксплуатация базы данных на SQL-сервере. Кроме того, использование локального варианта SQL-сервера позволяет изучить работу с базами данных сервера при наличии лишь одного компьютера.

В Access 2010 возможно преобразование базы данных в формат SQL Server. Для этого предназначена команда **SQL Server** в группе **Перемещение данных** (Move Data) на вкладке **Работа с базами данных** (Database Tools). Мастер может создать новую базу данных SQL Server или воспользоваться существующей. При этом указываются имя SQL Server, имя базы данных и способ соединения.

## Интернет-технологии

Пользователи баз данных все больше ориентируются на уникальные возможности быстрого сбора и совместного использования информации, предоставляемые интернет-технологиями. Базы данных широко используются как в интернет-публикациях, так и в электронной коммерции. Новые технологии в Access 2010 позволяют создавать специальные веб-приложения — веб-базы данных и публиковать их на сайтах Microsoft SharePoint Server, на которых выполняются службы Access. Базу данных можно опубликовать как на собственном сервере SharePoint в интрасети, так и в Интернете. Пользователи могут использовать базу данных с помощью стандартного браузера, не устанавливая приложение Access на компьютере. Это делает простым совместное использование корпоративной информации в среде настольных систем. При этом пользователи могут с помощью браузера открывать веб-формы и отчеты. В то же время сохраняется возможность работы в Access с веб-базами данных автономно, можно вносить изменения в макеты и данные, а затем, восстановив подключение, синхронизировать их с сервером Microsoft SharePoint Server 2010.

Создается веб-база данных, как и база данных для настольных компьютеров, в окне представления Backstage после открытия Access или в любой момент после закрытия активной базы данных выбором **Пустая веб-база данных** (Blank web database). Для создания таблицы в веб-базе данных используется режим таблицы, режим конструктора отсутствует. Использовать связанные таблицы в веб-базе данных нельзя, схема данных недоступна. Для создания связи в веб-базе данных используется поле подстановки. Поле подстановки создается с помощью мастера в таблице на стороне связи "многие" и указывает на таблицу на стороне "один". Обеспечение целостности данных в веб-базе данных можно реализовать с помощью макросов данных.

При публикации веб-базы данных службы Access создают сайт SharePoint, содержащий базу данных. Публикация веб-базы данных выполняется на вкладке **Файл** (File) выбором раздела **Сохранить и опубликовать** (Save & Publish) и пункта **Опубликовать в Access Services** (Publish to Access Services). Здесь заполняются поле **URL-адрес сервера** (Server URL) и поле **Имя сайта** (Site Name), в которое вводится имя веб-базы данных. Это имя будет добавлено к URL-адресу сервера для получения URL-адреса приложения.

Все таблицы становятся списками SharePoint, а записи — элементами списка. Это позволяет использовать разрешения SharePoint для управления доступом к веб-базе данных, а также другие возможности SharePoint.

После публикации базы данных посетители сайта SharePoint, обладающие необходимыми разрешениями, могут с ней работать. Одни пользователи, имея полный доступ, могут вносить изменения в данные и структуру базы данных, другие только изменять данные, и наконец, ряду пользователей разрешается только читать данные.

В браузере выполняются формы, отчеты и большинство макросов. Формы являются основным средством ввода и редактирования, а также просмотра данных в веб-базе данных. Отчеты предназначены для просмотра и печати данных из веб-базы данных. При открытии формы/отчета браузер получает необходимые данные с сервера SharePoint. Данные форм и отчетов можно фильтровать и сортировать без их повторного получения с сервера. Вся обработка SQL-кода выполняется на сервере, что повышает производительность сети за счет ограничения обмена данными с результирующими наборами.

В веб-браузере недоступна область навигации. Чтобы упростить работу с объектами базы данных, необходимо предоставить им средства навигации. Для этого можно создать форму навигации и указать, что она должна отображаться при открытии приложения в веб-браузере.

В веб-базе данных можно создать множество клиентских объектов, но их нельзя использовать в браузере. Тем не менее эти объекты являются частью веб-базы данных, и пользователи, открыв веб-базу данных в приложении Access на настольном компьютере, могут использовать клиентские объекты.

Некоторые функции, которые можно использовать в базах данных для настольных компьютеров, отсутствуют в службах Access. Однако ряд новых функций поддерживает многие из тех сценариев, которые доступны для настольных компьютеров.

SharePoint устраняет все проблемы, связанные с параллельным доступом к данным. Работа с приложением базы данных через браузер выполняется на основе разрешений SharePoint, определяющих возможность доступа к тем или иным объектам. Кроме того, средствами SharePoint можно воспользоваться для выполнения таких функций как запись версий данных, подписка на получение предупреждений при внесении изменений. Таким образом обеспечивается эффективный общий доступ к базе данных, а также предоставляются новые возможности для совместной работы через Интернет.

# Начало работы в Microsoft Access 2010

## Запуск Access

Для работы с Access 2010 на локальном компьютере пользователя должна быть установлена одна из настольных операционных систем Windows XP/Vista/7 и СУБД Access. Для того чтобы начать работу в СУБД Access, необходимо после загрузки операционной системы запустить ее. Это можно сделать, например, так: в нижней части рабочего стола Windows XP на панели задач нажать кнопку **Пуск** (Start), в открывшемся меню выбрать **Все программы** (All programmes). Из списка программ, установленных на данном компьютере, выбрать **Microsoft Office | Microsoft Access 2010** и запустить СУБД. Access может быть запущен из списка недавно использовавшихся программ, также представленного в меню **Пуск**.

Для быстрого запуска Access удобно иметь ярлык этой программы на рабочем столе Windows. Создать ярлык можно разными способами. Например, выберите программу Microsoft Access, как при ее запуске. Нажмите правую кнопку мыши. В контекстном меню выберите команду **Отправить** и в ее меню выполните команду **Рабочий стол (создать ярлык)**. Ярлык будет создан и отобразится на рабочем столе в виде, представленном на рис. 1.7.



Рис. 1.7. Ярлык для запуска Access 2010

Теперь запуск Access может быть выполнен двойным щелчком мыши на ярлыке. После запуска Access 2010 отображается представление Backstage — рис. 1.8.

### ЗАМЕЧАНИЕ

Можно запустить Microsoft Access, дважды щелкнув по файлу базы данных Access. При этом сразу будет открыта база данных.

На странице представления Backstage можно создать новую базу данных, открыть базу данных, создать базу с помощью одного из локальных шаблонов или просмотреть новейшее содержимое Web-сайта Office.com. Таким образом, эта страница обеспечивает быстрый доступ к средствам, позволяющим начать работу, в том числе с помощью комплектов профессионально разработанных шаблонов.

В левой части открытого окна представления Backstage выбран элемент **Создать** (New). При этом в правой части достаточно нажать кнопку **Создать** (Create), чтобы был создан файл базы данных с указанным именем и в предлагаемой по умолчанию папке. Очевидно, что и имя и место размещения файла могут быть здесь же изменены.

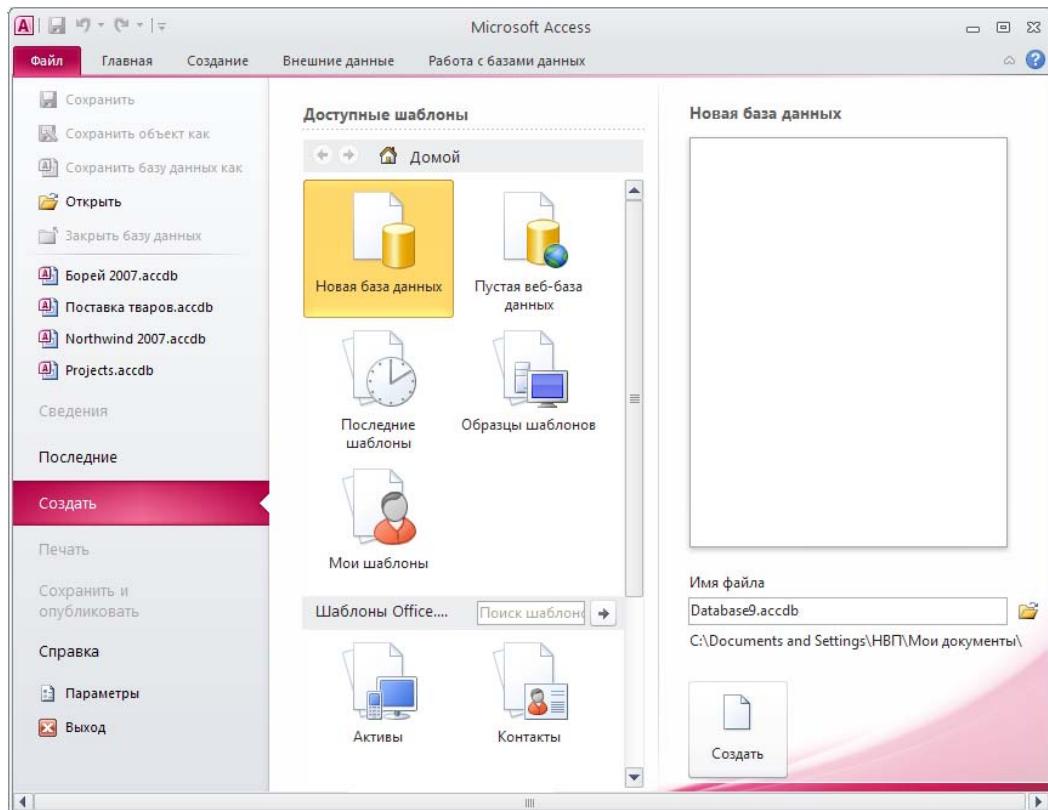


Рис. 1.8. Окно представления Backstage

Открыть существующую базу данных можно, выбрав в левой части окна представления соответствующую команду. Открыть одну из баз данных, с которыми работал пользователь, можно выбрав ее из списка последних баз данных для быстрого доступа. Число документов в списке последних файлов для быстрого доступа устанавливается соответствующим параметром на странице **Последние** (Recent) (по умолчанию 4). Число документов в списке последних файлов на странице **Последние** устанавливается в окне **Параметры Access** (Access Options) на вкладке **Параметры клиента** (Client Settings) в группе **Вывод на экран** (Display) (по умолчанию 17).

Представление Backstage можно открыть в любой момент щелчком на цветной вкладке **Файл** (File), заменившей кнопку **Office** (Office Button) предыдущих версий. При открытой базе данных представление содержит такие команды как: **Сохранить** (Save) базу данных; **Сохранить базу данных как** (Save Database As) — сохранить копию базы данных в текущем формате или формате другой версии Access, или сохранить текущий объект базы данных как новый объект, например таблицу как форму; выполнить **Печать** (Print) текущего объекта базы данных; **Закрыть базу данных** (Close Database). В разделе **Сведения** (Info) можно **Сжать** и

**восстановить базу данных** (Compact & Repair Database), **Зашифровать паролем** (Encrypt with Password) (рис. 1.9).

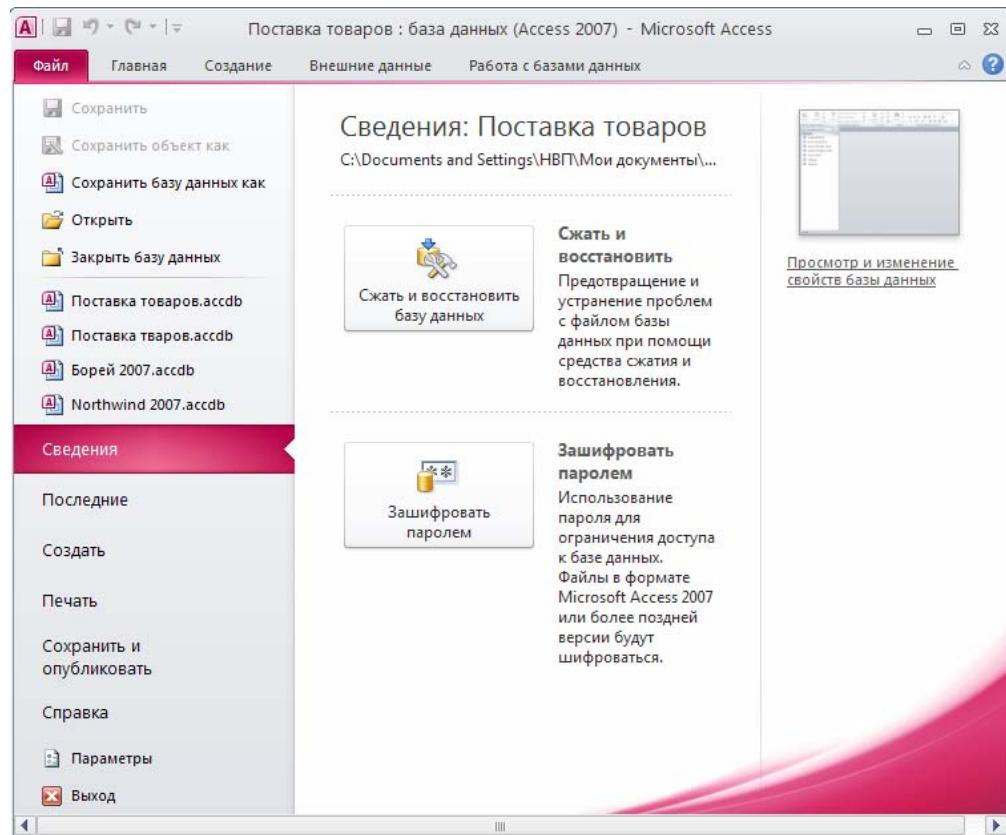


Рис. 1.9. Окно, открываемое на вкладке **Файл**

На странице представления можно перейти к настройке основных параметров Access, параметров текущей базы данных, безопасности документов и компьютера, открыть справку. Кнопка **Выход** (Exit Access) обеспечивает закрытие Access.

Чтобы при запуске Access вместо отображения представления открывалась последняя использованная база данных в окне **Параметры Access** (Access Options), на вкладке **Параметры клиента** (Client Settings) в разделе **Дополнительно** (Advanced) выберите **Открывать последнюю использовавшуюся базу данных при запуске Access** (Open last used database when Access starts).

## Интерфейс пользователя Access

В Access 2010 используется разработанный в Access 2007 интерфейс пользователя, который упрощает доступ к многочисленным функциональным возможно-

стям в процессе создания и работы с объектами базы данных и приложения пользователя.

Основу этого интерфейса составляют лента и область навигации. В Access 2010 лента изменена и добавлен еще один важный компонент пользовательского интерфейса — представление Backstage.

В версиях до Access 2007 команды были размещены в различных меню, панелях инструментов, в областях задач и других компонентах пользовательского интерфейса. Собранные на одной ленте команды четко соответствуют задачам, выполняемым в Access, что позволяет легко находить нужную команду. Интерфейс обеспечивает доступ практически к 1000 команд, но на экране отображаются только те, которые имеют отношение к задаче, выполняемой в данный момент.

Основные элементы интерфейса пользователя в Access 2010:

- ❖ представление Backstage предназначено для управления файлами баз данных. Оно отображается при запуске Access и позволяет открыть или создать новый файл базы данных. В процессе работы это представление доступно на вкладке **Файл (File)** и содержит команды для сохранения и публикаций, сжатия и восстановления базы данных;
- ❖ лента — широкая полоса, расположенная в верхней части окна Access. Она содержит стандартные вкладки с группами наиболее часто используемых команд, контекстные вкладки, которые появляются только тогда, когда их использование допустимо, и панель быстрого доступа — небольшую панель инструментов, на которую можно добавить самые нужные команды. Контекстные вкладки отображаются рядом со стандартными вкладками в зависимости от того, с каким объектом работает пользователь и какие действия он выполняет. В каждом режиме Access отображает свои вкладки и, соответственно, свой набор команд, представляющий всю его функциональность. Лента является основой интерфейса пользователя и обеспечивает быстрый доступ к набору команд, применимых к выполняемым в базе данных в текущий момент работам;
- ❖ коллекция (*галерея*) — элемент интерфейса, который не просто отображает команды, а показывает набор результатов выполнения этих команд с отображением внешнего вида вариантов выбора;
- ❖ диалоговые окна могут выводиться при выполнении команд для уточнения операции и передачи параметров. В некоторых группах вкладок ленты имеются кнопки вызова диалоговых окон;
- ❖ контекстное меню вызывается щелчком правой кнопкой мыши на элементе объекта. Содержит команды, зависящие от контекста — элемента объекта, с которым работает пользователь, или выполняемой задачи;
- ❖ панель быстрого доступа — единственная панель инструментов, предусмотренная в интерфейсе. Она обеспечивает доступ одним нажатием кнопки к наиболее часто используемым командам. Эта панель настраивается в соответствии с предпочтениями пользователя;
- ❖ область навигации расположена в левой части окна. В ней отображаются объединенные в группы объекты базы данных. Область переходов заменяет окно базы данных, использовавшееся в более ранних версиях Access;

- ❖ *вкладки документов* — таблицы, запросы, формы, отчеты и макросы отображаются на вкладках в рабочем пространстве окна Access — окне документов. Этот режим работы по умолчанию используется для всех баз данных, созданных с помощью Access 2010. Для того чтобы открывать каждый объект в отдельном окне, как в версиях до Access 2007, следует выбрать соответствующий параметр;
- ❖ *строка состояния* — полоса в нижней части окна программы, в которой отображаются сведения о состоянии и располагаются кнопки, позволяющие изменить режим представления объекта;
- ❖ *мини-панель инструментов* — прозрачный элемент, подключенный к объекту, который появляется над выбранным текстом и позволяет легко отформатировать его;
- ❖ *панель сообщений* — это единое средство вывода всех предупреждений системы безопасности. Отображается, когда в открываемой базе данных имеется любое потенциально опасное выполняемое содержимое.

## Представление Backstage

Сразу после запуска Access отображается новый компонент пользовательского интерфейса Access 2010 — представление Backstage с открытой вкладкой **Файл** (File). Представление Backstage — это место, где можно управлять файлами. В момент открытия здесь доступны команды **Создать** (New), **Открыть** (Open), **Параметры** (Options). Доступен список недавно использованных последних баз данных, который позволяет быстро выполнить открытие одной из них. Всплывающая подсказка каждой представленной в списке последних баз данных отображает местоположение ее файла — полный путь к нему. Представлены многочисленные шаблоны для создания различных типовых баз данных. Для открытой базы данных при выборе вкладки **Файл** (File) отображается представление Backstage с командами, применимыми ко всей базе данных, такими как **Сохранить и опубликовать** (Save & Publish), **Сжать и восстановить** (Compact & Repair), **Зашифровать паролем** (Encrypt with Password).

## Лента

В Access 2010 при открытии базы данных появляется лента (рис. 1.10). При этом на ней отображены четыре стандартные вкладки — **Главная** (Home), **Создание** (Create), **Внешние данные** (External Data) и **Работа с базами данных** (Database Tools). Вкладки включают логически связанные команды. Однотипные команды вкладки объединены в группы. Название группы представлено под набором команд, составляющих ее.

Название команды на вкладке и ее описание можно просмотреть во всплывающей подсказке, которая отобразится, если задержать на кнопке указатель мыши. Установив курсор на кнопках, с помощью всплывающих подсказок можно найти нужную операцию. Для того чтобы начать ее выполнение, достаточно щелкнуть на кнопке мышью.

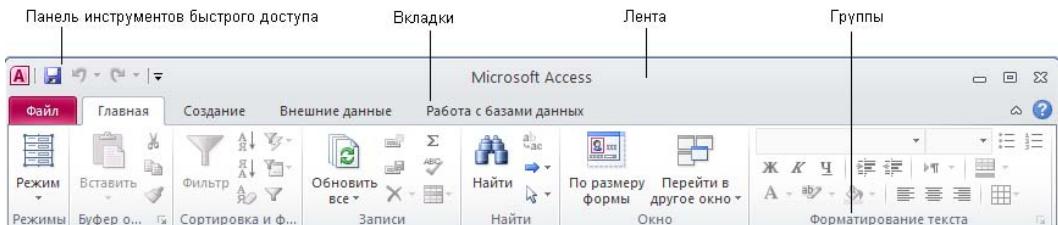


Рис. 1.10. Лента с открытой вкладкой Главная

Набор вкладок изменяется при переходе к другому объекту или режиму работы. Например, если открыть таблицу базы данных на ленте появится рядом со стандартными вкладками контекстная вкладка **Работа с таблицами** (Table Tools) с двумя вкладками второго уровня **Поля** (Fields) и **Таблица** (Datasheet). Команды стандартных вкладок по-прежнему остаются доступными. Например, щелкнув на команде **Форма** (Form) вкладки **Создание** (Create), вы создадите форму, источником записей которой будет выбрана открытая таблица.

Access, манипулируя набором встроенных вкладок, обеспечивает удобный интерфейс пользователя при выполнении работ в каждом из объектов базы данных. Эти вкладки выводятся в окне Access по умолчанию в соответствии с текущим режимом работы объекта.

## Коллекция

**Коллекция** — это элемент интерфейса, который отображает группу команд, показывая результат выполнения каждой из них. Смысл состоит в том, чтобы представить пользователю возможность по внешнему виду найти и выбрать нужные действия в Access 2010, сосредоточившись на результате, а не на поиске команды.

Коллекции различаются по форме и размерам. Примеры коллекций можно видеть на вкладках конструктора форм и отчетов. На рис. 1.11 приведена коллекция кнопки **Темы** (Themes).

Дополнительную более тонкую настройку внешнего вида формы или отчета можно выполнить, воспользовавшись командами вкладок **Формат** (Format) и **Упорядочить** (Arrange).

## Диалоговые окна

Диалоговые окна выводятся при выполнении некоторых команд для уточнения операции и передачи параметров. При открытом диалоговом окне нельзя перейти к выполнению других действий в данном приложении. Диалоговое окно имеет постоянные размеры, но может быть перемещено в другое место.

В некоторых группах вкладок ленты имеются маленькие значки — кнопки вызова диалоговых окон. Нажатие такой кнопки открывает соответствующее диалоговое окно или область задач, предоставляя дополнительные возможности, относящиеся к этой группе.

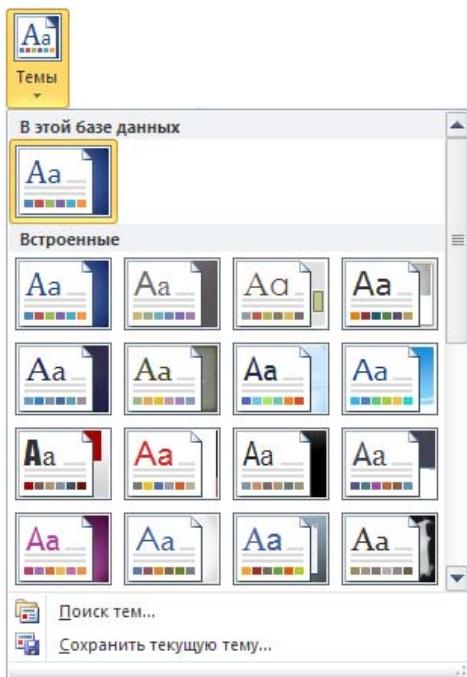


Рис. 1.11. Пример коллекции на вкладке конструктора при работе с макетом формы или отчета

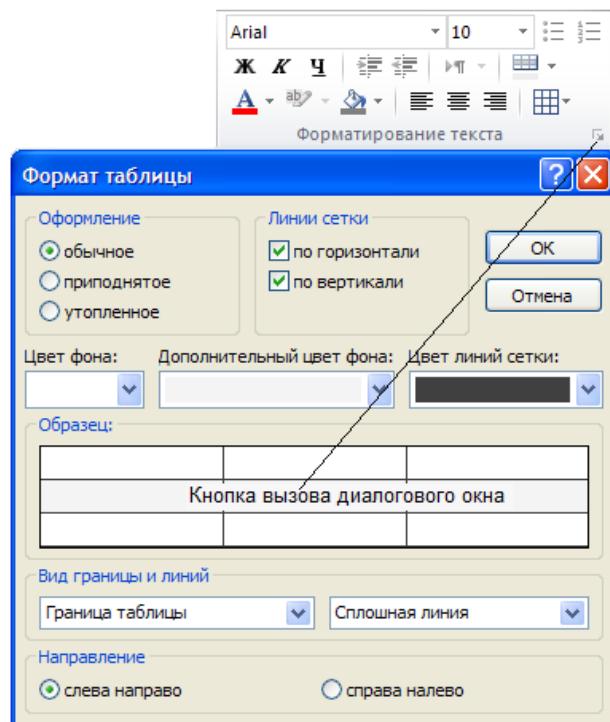
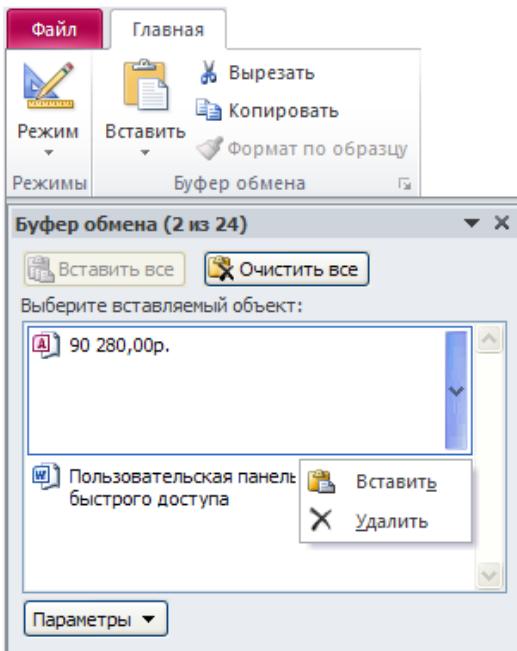


Рис. 1.12. Диалоговое окно Формат таблицы

Пример диалогового окна **Формат таблицы** (Datasheet Formatting) для выбора дополнительных параметров форматирования открытой таблицы базы данных на вкладке **Главная** (Home) приведен на рис. 1.12.

Пример области задач для отображения содержимого буфера обмена, открывающейся кнопкой в группе **Буфер обмена** (Clipboard) на вкладке **Главная** (Home), приведен на рис. 1.13.



**Рис. 1.13. Область задач Буфер обмена**

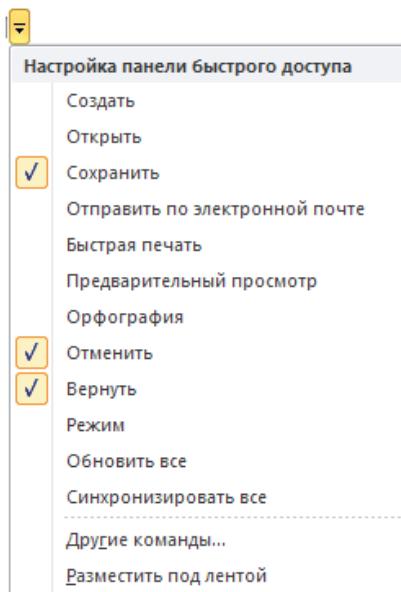
## Контекстное меню

Access практический каждый элемент объекта в любом режиме снабжает динамическим контекстным меню, вызываемым правой кнопкой мыши и содержащим набор команд, применимых в данной ситуации. Эти меню включают большое число команд, не доступных в других элементах интерфейса Access. Например, команды открытия существующего объекта (таблицы, формы, отчета) в режиме конструктора, режиме таблицы или макета представлены только в контекстном меню этого объекта, выбранного в области навигации, которая представляет все объекты базы данных.

## Пользовательская панель инструментов быстрого доступа

Пользовательская панель инструментов быстрого доступа (Customize Quick Access Toolbar) первоначально включает лишь кнопки **Сохранить** (Save), **Отме-**

**нить** (Undo), **Вернуть** (Redo) и кнопку **Настройка панели быстрого доступа** (Customize Quick Access Toolbar), открывающую список команд, которыми может быть дополнена панель (рис. 1.14).



**Рис. 1.14.** Список команд кнопки **Настройка панели быстрого доступа**

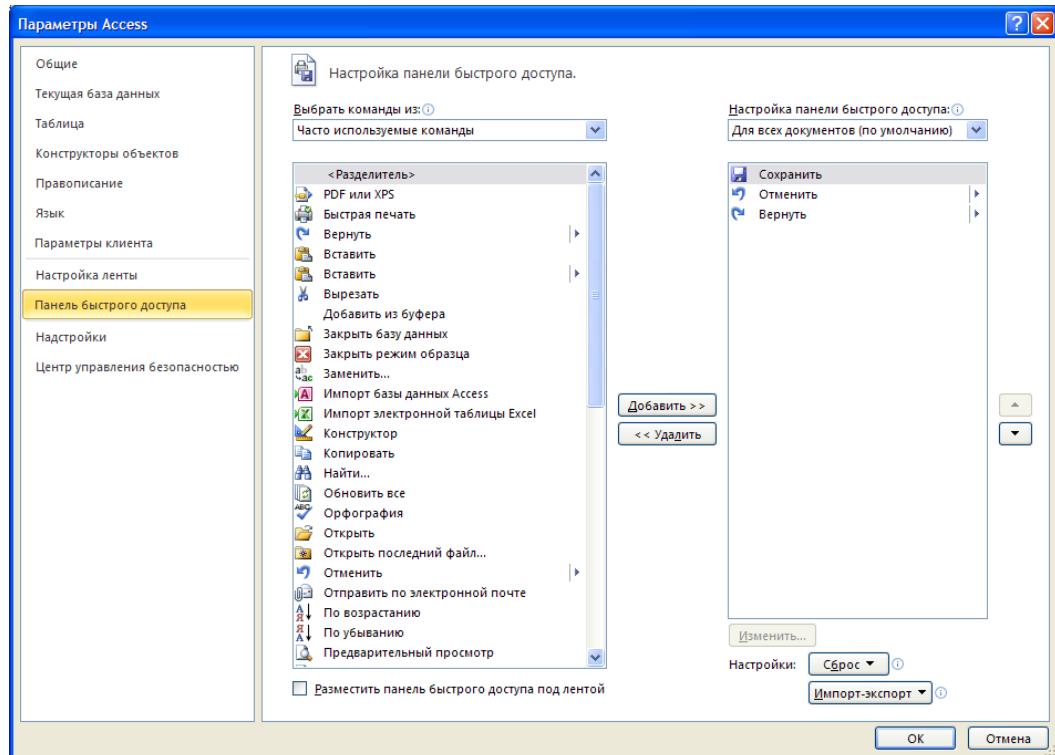
Команды списка **Разместить над лентой** (Place Quick Access Toolbar above the Ribbon) или **Разместить под лентой** (Place Quick Access Toolbar below the Ribbon) позволяют изменить местоположение этой панели, разместив ее ниже или выше ленты команд.

Строка списка **Другие команды** (More Commands) открывает окно **Параметры Access** (Access Options) на вкладке **Панель быстрого доступа** (Quick Access Toolbar) (рис. 1.15). В этом окне пользователь может по своему усмотрению дополнить панель быстрого доступа командами, выбирая их из полного списка команд Access. Команды списка разбиты по категориям. Отдельно представлена категория **Часто используемые команды** (Popular Commands). Выбор категории команд, из которой вы собираетесь добавить команды, выполняется в поле **Выбрать команды из** (Choose commands from) в левой части окна.

Для добавления или удаления выбранной команды предназначены кнопки **Добавить** (Add) и **Удалить** (Remove). Для упорядочения кнопок на панели инструментов быстрого доступа воспользуйтесь кнопками перемещения **Вверх** (Move Up) и **Вниз** (Move Down). Настройка может быть выполнена как для текущей базы данных, так и для всех баз данных. Этот выбор выполняется в списке **Настройка панели быстрого доступа** (Customize Quick Access Toolbar) в правой части окна.

Добавить команду на пользовательскую панель инструментов быстрого доступа можно, не открывая окна **Параметры Access** (Access Options). Контекстное меню любой кнопки на ленте включает команду **Добавить на панель быстрого доступа** (Add to Quick Access Toolbar). Контекстное меню любой кнопки на панели

инструментов быстрого доступа содержит команду **Удалить с панели быстрого доступа** (Remove from Quick Access Toolbar). Эти команды позволяют в оперативном режиме менять состав кнопок, настраивая панель на текущие потребности пользователя.



**Рис. 1.15.** Окно настройки пользовательской панели инструментов быстрого доступа

Используя контекстное меню, на панель инструментов быстрого доступа можно добавить (удалить) не только отдельную команду ленты, но и целую коллекцию, которая представляется одним значком.

При последующих открытиях Access добавленные на панель инструментов кнопки будут сохранены.

Выход команд на панель позволяет выполнить их быстрым и удобным способом из любого окна Access. Однако состав кнопок панели является контекстно-независимым, и поэтому на панели целесообразно размещать только те команды, которые могут быть выполнены в любой момент или используются достаточно часто.

## Область навигации

Начиная с Access 2007, объекты открытой базы данных представлены в разделе **Область навигации** (Display Navigation Pane). Эта область заменяет окно базы

данных предыдущих версий Access, а также кнопочные формы — панели управления приложением, используемые для переходов по объектам базы данных и выполнения различных задач, например запуска отчетов, открытия форм.

При открытии базы данных в Access 2010 все ее объекты — таблицы, формы, отчеты, запросы, макросы и модули — отображаются в области навигации в соответствующих группах. Она обеспечивает быстрый доступ к нужным объектам, которые для удобства пользователя могут быть объединены не только в стандартных категориях и группах, а и в созданных пользователем в соответствии с функциональностью приложения. Состав отображаемых в области навигации объектов может меняться. Можно вывести только нужные категории, отфильтровать группы объектов, выбрать объекты по датам создания или изменения. Для представления области навигации в виде, аналогичном Access предыдущих версий, щелкните на кнопке списка в заголовке области навигации и выберите категорию **Тип объекта** (Object Type) и фильтр **Все объекты Access** (All Access Objects) (рис. 1.16).

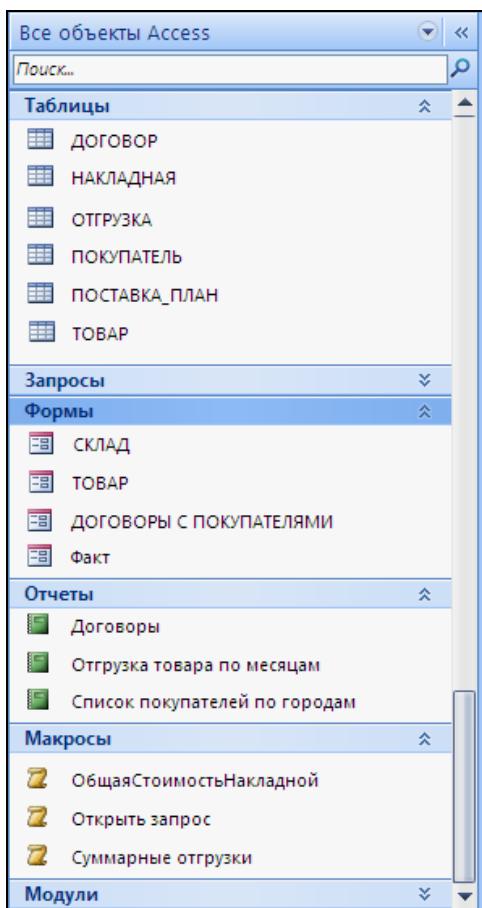


Рис. 1.16. Область навигации с группами объектов

Область навигации используется для того, чтобы открыть объект для работы с ним или для изменения его структуры в режиме конструирования или макета. От-

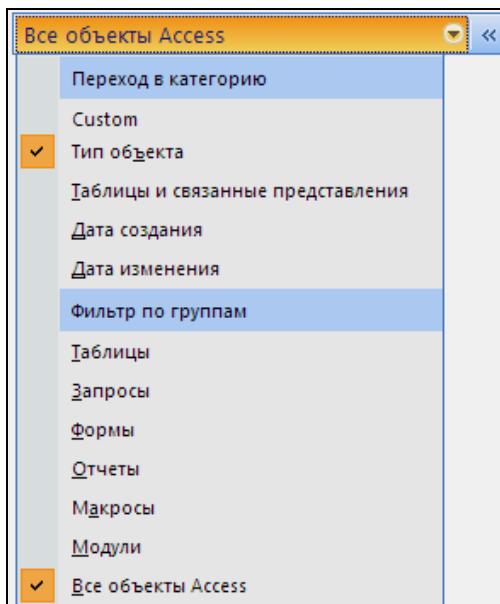
крытие объекта может быть выполнено двойным щелчком на нем или с помощью контекстного меню, в котором представлены основные режимы работы с объектом. Только для модуля при двойном щелчке запускается редактор Visual Basic, а не выполнение модуля. Кроме того, выбранный в области навигации объект можно использовать для экспорта в Excel, в RTF-файлы, файлы XML, списки SharePoint, в базы данных различных приложений. Можно также отправить объект по электронной почте в открывшемся окне Microsoft Office Outlook.

Если не нужно просматривать сразу все объекты базы данных, можно создать настраиваемые пользователем категории и группы объектов базы данных и отобразить в области навигации только эти категории и группы. Поскольку в таких группах доступны, например, только требуемые формы и отчеты и скрыты остальные объекты базы данных, эти группы можно использовать вместо кнопочных форм.

Область навигации всегда остается видимой на экране и не может быть закрыта другими окнами. Чтобы убрать ее с экрана, освободив место для выполнения других работ, например, конструирования формы, предназначена кнопка **Открыть/закрыть границу области переходов** (Shutter Bar Open/Close Button) , размещенная в правом верхнем углу заголовка, или клавиша <F11>.

Объекты в области навигации организованы по категориям (рис. 1.17). Раздел **Переход в категорию** (Navigate To Category) отображает стандартные категории **Настройка** (Custom), **Тип объекта** (Object Type), **Таблицы и связанные представления** (Tables and Related Views), **Дата создания** (Created Date), **Дата изменения** (Modified Date).

Выбор категории определяет список доступных фильтров. Для категории **Тип объекта** (Object Type) в разделе **Фильтр по группам** (Filter By Group) перечисляются все доступные типы объектов.



**Рис. 1.17.** Панель выбора категории и группы объектов для отображения в области навигации

При выборе категории **Таблицы и связанные представления** (Tables and Related Views) в разделе **Фильтр по группам** (Filter By Group) открывается возможность выбора таблицы, для которой нужно просмотреть все запросы, в которых она использована. Заметим, что в список представлений включаются не только запросы, но и формы и отчеты, в основе которых лежат запросы, в том числе созданные при их разработке.

В соответствии с названиями категорий может быть произведен отбор объектов по времени создания или изменения.

Чтобы по умолчанию запретить отображение области навигации при открытии базы данных, следует для текущей базы данных (Current Database) в окне параметров Access в разделе **Навигация** (Navigation) снять флажок **Область навигации** (Display Navigation Pane). Скрывать область навигации целесообразно только при наличии кнопочной формы или другого метода для запуска объектов в базе данных.

## Вкладки документов

Окно документов Access работает с новой моделью пользовательского интерфейса, называемой моделью однодокументного интерфейса (SDI). Эта модель размещает все объекты базы данных: формы, отчеты и т. д. в одном окне документов и образует для каждого объекта свою вкладку. Если открыто несколько объектов, для переключения между этими объектами используются вкладки. Рисунок 1.18 иллюстрирует обычный набор вкладок.

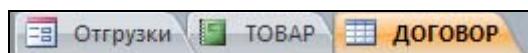


Рис. 1.18. Вкладки окна документов Access

Использование вкладок — это режим работы по умолчанию для всех баз данных, созданных с помощью Access 2010. Для того чтобы открывать каждый объект в отдельном окне, как это было в предыдущих версиях, в окне **Параметры Access** (Access Options) выберите **Текущая база данных** (Current Database), в разделе **Параметры приложений** (Application Options) в группе **Параметры окна документа** (Document Window Options) установите флажок **Перекрывание окон** (Overlapping Windows) вместо **Вкладки документов** (Tabbed Documents).

## Строка состояния

Вдоль нижней границы окна может отображаться строка состояния. Эта строка предназначена для вывода текста сообщения, связанного с текущим режимом, состоянием объекта, выполняющейся программой.

Элементы управления в правой части строки состояния позволяют быстро переключать различные режимы активного объекта (рис. 1.19). При просмотре объекта, который поддерживает изменение масштаба (например, отчета в режиме

предварительного просмотра), можно регулировать степень увеличения или уменьшения с помощью ползунка в строке состояния.

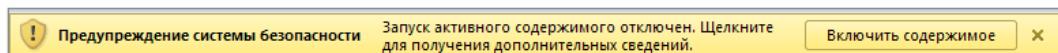


**Рис. 1.19.** Страна состояния при просмотре объекта, поддерживающего изменение масштаба

Включить или отключить отображение строки состояния можно в диалоговом окне **Параметры Access** (Access Options). В этом окне в левой области выберите **Текущая база данных** (Current Database). В открывшемся окне в разделе **Параметры приложений** (Application Options) снимите флажок **Страна состояния** (Status bar).

## Панель сообщений

В Access 2010, как и в других приложениях Microsoft Office этого выпуска, вывод всех предупреждающих сообщений системы безопасности производится не в отдельных окнах, а на панели сообщений. По умолчанию уже при открытии базы данных, если точно неизвестно, что можно доверять ее содержимому, Центр управления безопасностью отключает в открываемой базе данных такие потенциально опасные активные компоненты, как запросы на изменение (запросы, которые добавляют, удаляют или изменяют данные), макросы, элементы управления ActiveX, выражения (функции, возвращающие одно значение), программы на VBA, и выводит предупреждение об этом на панели сообщений (рис. 1.20).



**Рис. 1.20.** Предупреждающее сообщение системы безопасности при открытии базы данных

Если вы доверяете содержимому базы данных, включите отключенное содержимое, щелкнув **Включить содержимое** (Enable this content). Отключенное содержимое будет включено и база данных откроется с полным набором функциональных возможностей. В противном случае отключенные компоненты не будут работать.

## Мини-панель инструментов

В версиях Access, выпущенных до Access 2007, для форматирования текста использовалось меню или панель инструментов **Форматирование** (Format). В Access 2010 при выделении текста над ним автоматически появляется мини-панель инструментов для форматирования (рис. 1.21). При приближении указателя мини-панель становится более четкой, и ее можно использовать для применения полу-

жирного начертания или курсива, изменения размера и цвета шрифта и т. д. При удалении курсора мини-панель инструментов постепенно исчезает. Если использовать мини-панель инструментов для форматирования выделенного текста не требуется, просто немного отодвиньте указатель, и мини-панель исчезнет.



Рис. 1.21. Мини-панель инструментов форматирования текста

## Технология Drag and Drop

Реализованная в Access технология *Drag and Drop* позволяет пользователю переносить объекты базы данных и их элементы с помощью мыши. Например, любую таблицу или запрос можно перенести из области навигации в окно схемы данных. Для установления связи между объектами можно в схеме данных переместить поле из одной таблицы в другую. Для размещения подчиненной формы в главной достаточно перенести в нее ранее созданную форму или даже просто таблицу.

С помощью мыши можно переносить объекты между различными базами данных. При этом необходимо запустить две задачи Microsoft Access. Возможен перенос таблиц и запросов Access в другие приложения, например, в Microsoft Word и Microsoft Excel. Можно выделить нужные данные в форме или в объекте в режиме таблицы и перенести только их. Можно создать таблицу путем переноса с помощью мыши диапазона ячеек Microsoft Excel в область навигации базы данных Access. Объекты других приложений могут быть перенесены в поле объекта OLE в таблице или форме в режиме формы, а также в форму или отчет в режиме конструктора.

## Смарт-теги

Быстро выполнить некоторые задачи (действия), для которых обычно предназначены другие программы, в Access можно с помощью смарт-тегов. Добавить смарт-тег к полю таблицы, запроса или элементу управления формы, отчета можно в его свойстве **Смарт-теги** (Smart Tags). Причем достаточно выбрать тег действия из списка, который открывается щелчком на значке построителя в конце строки свойства. После добавления смарт-тега при активизации ячейки этого поля или элемента управления появляется кнопка, открывающая меню действий, которые доступны для смарт-тега.

Например, если в форме, отображающей сведения о покупателях, имеется поле с его электронным адресом, добавленный в это поле смарт-тег позволит перейти к подготовке письма покупателю в Outlook. С помощью смарт-тега имени покупателя можно добавить такие действия, как "Запланировать собрание" или "Показать мой календарь".

В Access представлена лишь малая часть смарт-тегов. Дополнительные смарт-теги, созданные в корпорации Microsoft или независимыми разработчиками, можно найти в Интернете.

## Справка Access

Справочная система Access позволяет получить недостающие сведения и отвечать на возникающие в ходе работы вопросы. Access, как каждое приложение в Microsoft Office и сама операционная система Windows, имеет отдельное окно справки **Справка: Access** (Access Help) и гипертекстовую структуру. Верхним уровнем этого гипертекстового документа является оглавление. Справочной системой обеспечивается поиск по ключевым словам и подготовка списка разделов, отвечающих на введенный запрос. Открыть окно справки можно, нажав кнопку справки  или клавишу <F1>.

Справка Access устанавливается на компьютере при инсталляции приложения. При подключении к Интернету открывается справка с веб-сайта **Office.com**, где размещено множество справочных статей, видеороликов и обучающих курсов. В правом нижнем углу отображается надпись **Подключение к сайту Office.com установлено**. Если в правом нижнем углу отображается надпись **Автономная работа** (Offline), поиск необходимых сведений будет выполняться в файлах, хранящихся на компьютере. В результатах поиска отображаются все найденные там статьи.

Для выбора источника необходимых сведений используется меню **Состояние подключения** (Connection Status) в правом нижнем углу окна справки (рис. 1.22). Например, для отображения данных только с локального компьютера необходимо выбрать параметр **Показать контент только с данного компьютера** (Show content only from this computer). Очевидно, для переключения на справку с сайта **Office.com** необходимо подключение к Интернету.

### ЗАМЕЧАНИЕ

Параметр меню **Состояние подключения** (Connection Status) сохраняется после закрытия окна справки. При следующем открытии справки статус подключения будет тем же, каким он был оставлен.

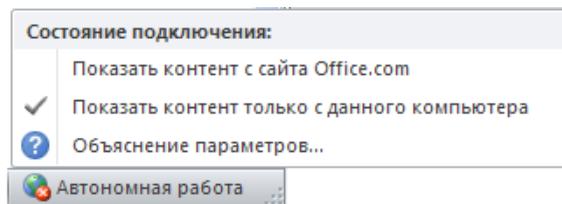
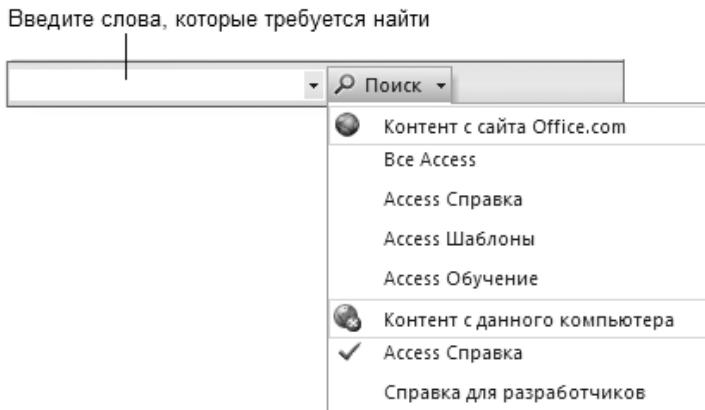


Рис. 1.22. Меню выбора источника справки Access

При получении справки можно не только ограничить область поиска автономным или интерактивным режимом, но и уточнить место для поиска необходимых

сведений выбором конкретной категории внутри приложения или на сайте **Office.com**. Выбор категории производится в списке **Поиск (Search)** (рис. 1.23).



**Рис. 1.23.** Меню выбора места поиска справки

В каждой программе пакета Microsoft Office имеется своя домашняя страница справки, отображающаяся при открытии справки. Эту страницу можно использовать для общего ознакомления с разделами справки. Открывшаяся домашняя страница будет соответствовать выбранной категории в списке кнопки **Поиск (Search)**.

Автономная справочная система имеет оглавление. Для отображения оглавления используйте кнопку панели инструментов **Показать/Скрыть оглавление** (Show/Hide Table of Contents) . Выбор гиперссылки оглавления приводит к отображению содержимого раздела в окне справки. Окно справки имеет кнопки перехода вперед и назад по просмотренным разделам и кнопку вывода раздела на печать.

Чтобы получить справку по возникшему вопросу, введите в качестве критерия поиска ключевые слова в поле **Введите слова, которые требуется найти** (Type words to search for), а затем нажмите кнопку **Поиск (Search)**. Место поиска ответа на вопрос зависит от настройки списка **Поиск (Search)**. Результатом поиска ответа на введенный вопрос является список гиперссылок на разделы справки. Ввод дополнительных ключевых слов сужает список найденных разделов. Ответы представлены в списке в порядке релевантности, т. е. первым отображается наиболее подходящий ответ на вопрос. Выбор любой из гиперссылок приводит к открытию содержимого раздела в окне справки.

Уже в приложениях системы Microsoft Office 2007 поле **Введите вопрос** (Type question) было перенесено из окна приложения в окно справки и переименовано в **Введите слова, которые требуется найти** (Type words to search for). Кроме того, новый интерфейс не включает в себя помощника Microsoft Office.

Для получения справки о применении доступных на ленте команд Access подведите указатель мыши к команде, появится всплывающая подсказка со сведениями о том, какие действия выполняет эта команда. Если наряду с этими сведениями

в подсказке отображается сообщение **Для получения дополнительных сведений нажмите клавишу F1** (Press F1 for more help), продолжайте удерживать указатель мыши на команде и нажмите клавишу <F1>.

Для получения общих сведений и рекомендаций по разработке процедур Visual Basic для приложений (VBA), позволяющих расширить возможности приложения, прежде чем вводить вопрос выберите в списке **Поиск (Search)** **Справочник разработчика** (Offline Developer Help).

Легкодоступные подсказки также помогут пользователю при работе в Access. Для получения подсказки в диалоговом окне, когда не ясно, как можно использовать тот или иной его параметр для выполнения задачи, установите курсор в поле параметра и нажмите клавишу <F1>. Краткие сведения о назначении этого параметра отобразятся в появляющейся подсказке. Тот же результат может быть получен, если нажать кнопку **Справка**  (Microsoft Office Access Help (F1)), расположенную в строке заголовка диалогового окна, перенести знак вопроса в поле нужного параметра и щелкнуть на нем. Для части диалоговых окон эти действия приводят к открытию справки с нужным разделом. Чтобы просмотреть название кнопки на панели инструментов, задержите указатель мыши на нужной кнопке, пока не отобразится всплывающая подсказка.

Веб-сайт **Office.com** обеспечит доступ к статьям, советам, коллекциям картинок, шаблонам, интерактивным средствам обучения, файлам для загрузки и службам.

Предлагаемые курсы обучения включают в себя изучение возможностей и обучение эффективному использованию продуктов Office. Небольшие курсы по 20—50 минут для самостоятельного обучения со звуком, графическими и анимационными возможностями позволяют приобрести практические навыки.

Для загрузки через Интернет на веб-сайте доступны многочисленные технические файлы, в том числе драйверы устройств, пакеты обновлений продуктов, файлы с исправлениями, а также полные продукты.

Сайт регулярно обновляется, а содержание узла определяется на основе обратной связи и определенных запросов пользователей Office.

## Защита баз данных

В предыдущих версиях Access базы данных имели защиту на уровне пользователя. Начиная с Access 2007 для баз данных, созданных в новом формате (файлы с расширением accdb или accde), не предусматривается такой защиты. Однако в базах данных предыдущих версий, имеющих защиту на уровне пользователя и открытых в Access 2007, защита будет продолжать работать. При преобразовании подобной базы данных в новый формат Access автоматически удалит все параметры защиты на уровне пользователя и будет применять новые правила защиты баз данных.

Защита баз данных в Access 2010 построена на доверии к происхождению ее выполнимого содержимого. Дело в том, что некоторые компоненты Access, полу-

ченные из ненадежных источников, выполняясь, могут повредить базу данных. Поэтому база данных открывается для работы с полным набором функциональных возможностей только, если она имеет состояние доверенной.

База данных Access в отличие от документа Word или книги Excel, хотя и хранится в одном файле, представляет собой набор объектов — таблиц, форм, запросов, отчетов, макросов и т. д., которые, как правило, являются взаимозависимыми и, следовательно, выполнение некоторых объектов может повлечь порчу базы данных.

К небезопасным компонентам относятся:

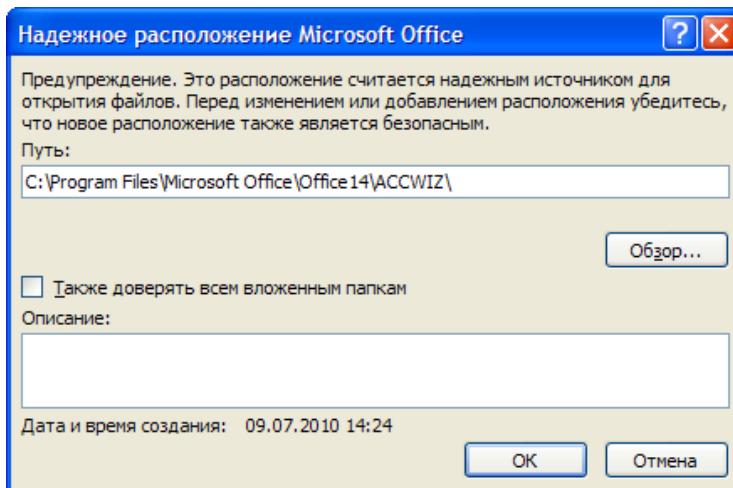
- ❖ запросы на изменение, которые добавляют, удаляют или изменяют данные в таблицах;
- ❖ макрокоманды, изменяющие базу данных или получающие доступ к ресурсам вне базы данных;
- ❖ управляющие запросы (DDL-запросы), предназначенные для создания или изменения объектов базы данных, таких как таблицы и процедуры;
- ❖ SQL-запросы к серверу, которые передаются на сервер баз данных и работают с таблицами непосредственно на нем, минуя ядро базы данных Access;
- ❖ элементы управления ActiveX;
- ❖ выражения (функции, возвращающие одно значение);
- ❖ код VBA.

База данных приобретет состояние доверенной, если поместить ее в надежное расположение либо применить цифровую подпись, воспользовавшись сертификатом от надежного издателя. Издатель — это разработчик макроса, элемента управления ActiveX, надстройки или других расширений приложений, используемых потребителями. Подпись подтверждает, что выполняемые компоненты базы данных созданы издателем, и никто другой не изменял их после подписи.

Защита баз данных в Access 2010 реализуется Центром управления безопасностью. Центр управления безопасностью отображает диалоговое окно, в котором можно задавать и менять параметры безопасности и конфиденциальности, в том числе определять надежное расположение, надежных издателей. Чтобы открыть это окно, воспользуйтесь кнопкой **Параметры** (Options) в представлении Backstage на вкладке **Файл** (File), затем в окне **Параметры Access** (Access Options) откройте группу **Центр управления безопасностью** (Trust Center) и нажмите кнопку **Параметры центра управления безопасностью** (Trust Center Settings). Параметры, установленные в Центре управления безопасностью, определяют поведение новых и существующих баз данных при их открытии в Access. Для установки параметров безопасности открывать базу данных не требуется.

Для создания надежного расположения и добавления туда папки с базами данных выберите **Надежные расположения** (Trusted Locations), нажмите кнопку **Добавить новое расположение** (Add new Location), а затем в диалоговом окне **Надежное расположение Microsoft Office** (Microsoft Office Trusted Location) укажите путь к папке надежного расположения на жестком диске или в сети (рис. 1.24). Для перемещения или копирования файла базы данных в надежное расположение используются любые привычные способы. Например, средства проводника Windows. После размещения

базы данных в надежном расположении при ее открытии не выполняется проверка средствами Центра управления безопасностью, не выводится предупреждение системы безопасности, а все выполнимое содержимое будет включено.



**Рис. 1.24.** Определение папки в качестве надежного источника для открытия файлов баз данных

Назначение в качестве надежного расположения всей папки Мои документы повышает риск атаки хакеров и увеличивает угрозу безопасности.

Любые внешние, такие как сетевые, папки являются менее безопасными. Не рекомендуется назначать в качестве надежного расположения папку общего пользования сетевого ресурса. При создании надежного расположения, не являющегося локальным ресурсом компьютера, в диалоговом окне **Надежное расположение** (Trusted Locations) нужно установить **Разрешить надежные расположения в моей сети (не рекомендуется)** (Allow Trusted Locations on my network (not recommended)), напоминающее о ненадежности расположения.

Для применения цифровой подписи (шифрованной электронной подписи) к компонентам базы данных, прежде всего, необходимо иметь цифровой сертификат. Если базы данных создаются для коммерческого распространения, нужно получить сертификат в коммерческом центре сертификации, например, VeriSign, Inc. или GTE. Центр сертификации наводит справки об издателе, чтобы удостовериться в его надежности.

Если базу данных планируется использовать в личных целях или в небольшой рабочей группе, можно воспользоваться предусмотренным в операционных системах Microsoft Windows средством создания сертификатов с собственной подписью, называемым SelfCert.exe. Этот сертификат следует добавить в список надежных источников, а затем подписать базу данных.

В Windows XP, Vista или Windows 7 для создания сертификата нажмите кнопку **Пуск** и выберите последовательно пункты **Все программы, Microsoft Office,**

**Средства Microsoft Office 2010** (Microsoft Office 2010 Tools) и **Средство создания цифровых сертификатов для проектов VBA** (Digital Certificate for VBA Projects). Откроется диалоговое окно **Создание цифрового сертификата** (Create Digital Certificate). В поле **Имя сертификата** (Your certificate's name) введите описательное имя сертификата. Нажмите **OK**. При появлении сообщения "SelfCert: успех" нажмите **OK**.

Для просмотра сертификата в хранилище личных сертификатов откройте браузер Internet Explorer. Выберите в меню **Сервис** команду **Свойства обозревателя** и откройте вкладку **Содержание**. Нажмите кнопку **Сертификаты** и откройте вкладку **Личные**.

При открытии accdb- или accde-файла Access 2010 сообщает расположение базы данных центру управления безопасностью. Если это расположение надежное, она работает с полным набором функциональных возможностей. При открытии базы данных из более ранней версии Access 2010 передает в Центр управления безопасностью расположение и цифровую подпись, если она имеется в базе данных.

Центр управления безопасностью проверяет подлинность этих сведений, чтобы определить, имеет ли база данных состояние доверенной, а затем информирует приложение Access о том, как следует ее открывать. Access либо отключает опасные компоненты базы данных, либо открывает с полным набором функциональных возможностей.

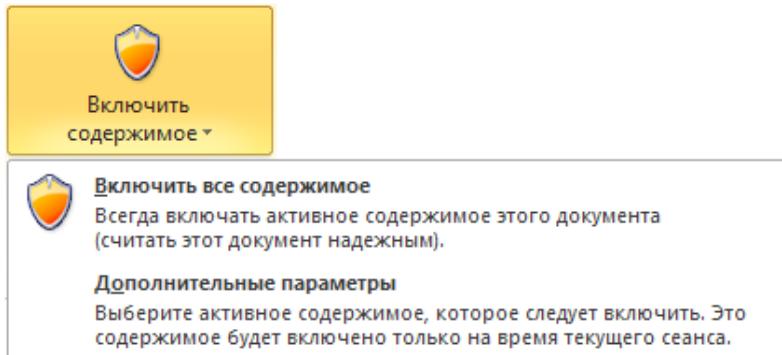
Если база данных не имеет состояния доверенной, программные средства Центра управления безопасностью по умолчанию, не оценивая компоненты базы данных, отключают все выполняемое содержимое в базе данных и запрашивает у пользователя разрешение на его включение, отображая панель сообщений (см. рис. 1.20).

Если пользователь доверяет источнику или берет на себя ответственность за безопасность базы данных, он разрешает использование опасного содержимого, щелкнув на кнопке **Включить содержимое** (Enable Content). При этом источник автоматически попадает в разряд надежных и при следующем открытии панель сообщений не будет выводиться. Для получения дополнительных сведений щелкните на соответствующей подписи на панели сообщений. В открывшемся представлении Backstage в разделе **Сведения** (Info) будет отображено предупреждение системы безопасности **Включить содержимое** (Enable Content), дополненное списком (рис. 1.25). Первое значение списка переводит базу данных в разряд надежных. Второе позволяет установить защиту от неизвестного содержимого или разрешить его выполнение на время текущего сеанса работы. При повторном открытии такой базы данных будет выводиться запрос по поводу отключения опасного содержимого.

При выборе в диалоговом окне **Центр управления безопасностью** (Trust Center) вкладки **Надежное расположение** (Trusted Locations) и варианта **Отключить все надежные расположения** (Disable all Trusted Locations) Access отключит все надежные расположения.

В Access 2007/10 панель сообщений является единственным средством вывода предупреждений, которое появляется по умолчанию уже при открытии базы данных вне доверенного расположения. Это позволяет пользователям иметь дело со значитель-

но меньшим числом предупреждающих сообщений в сравнении с предыдущими версиями Access.



**Рис. 1.25.** Предупреждение системы безопасности при открытии файла базы данных

Для включения панели сообщений откройте вкладку **Файл** (File). В левой части представления Backstage выберите **Параметры** (Options).

На левой панели диалогового окна **Параметры Access** (Access Options) выберите **Центр управления безопасностью** (Trust Center) и щелкните по элементу **Параметры центра управления безопасностью** (Trust Center Settings).

В окне **Центр управления безопасностью** щелкните по элементу **Панель сообщений** (Message Bar). Установите флажок **Показывать панель сообщений во всех приложениях, если активное содержимое такое как элементы ActiveX и макросы заблокированы** (Show the Message Bar in all applications when active content, such as ActiveX controls and macros, has been blocked) и нажмите кнопку **OK**.

После включения и отображения панели сообщений на ней можно включить содержимое.

Другим средством защиты является шифрование баз данных. Это средство объединяет два улучшенных средства прежних версий — кодирование и пароли баз данных. Чтобы зашифровать базу данных, достаточно установить пароль для шифрования базы данных. Стойкий алгоритм шифрования баз данных в формате Access 2010 исключает несанкционированный их просмотр. Все данные становятся нечитаемыми в других программных средствах, и для того чтобы использовать эту базу данных, пользователи должны вводить пароль.

Шифрование с использованием пароля выполняется для базы данных, открытой в монопольном режиме. Щелкните по вкладке **Файл** (File), затем выберите команду **Открыть** (Open), в окне **Открытие файла базы данных** (Open File Database) выделите файл, который нужно открыть, щелкните по стрелке рядом с кнопкой **Открыть** (Open) и выберите **Монопольно** (Open Exclusive).

Для шифрования базы данных с использованием пароля на вкладке **Файл** (File) нажмите кнопку **Сведения** (Info) и выберите **Зашифровать паролем** (Encrypt with Password).

В открывшемся диалоговом окне **Задание пароля базы данных** (Set Database Password) введите пароль и подтверждение и нажмите кнопку **ОК**. При открытии зашифрованной таким образом базы данных потребуется ввод пароля.

Выполнением команды **Расшифровать базу данных** (Decrypt Database), в которую преобразовалась команда **Зашифровать паролем** (Encrypt with Password) пароль базы данных удаляется. При этом также необходимо ввести пароль.

Пароли должны состоять не менее чем из 8 символов. Надежные пароли должны сочетать в себе прописные и строчные буквы, цифры и символы и состоять не менее чем из 14 символов. Важно помнить свой пароль, т. к. восстановить его невозможно.

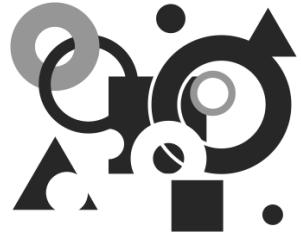
## Контрольные вопросы

1. Изменяется ли набор вкладок ленты при изменении режима обработки объекта в базе данных Access?
2. Для чего предназначена панель быстрого доступа?
3. Какие вкладки ленты выводятся при открытии базы данных?
4. Какая область позволяет отображать все объекты базы?
5. Можно ли управлять набором команд на панели быстрого доступа?
6. Как получить всплывающую подсказку для команды на вкладке ленты?
7. Можно ли изменить состав кнопок на вкладках ленты?
8. Какие объекты составляют базу данных Access?
9. Чем определяется структура реляционной таблицы?
10. Для чего предназначен первичный ключ реляционной таблицы?
11. Какими отношениями характеризуется связь двух таблиц реляционной базы данных?
12. Для чего используется ключ связи?
13. Что предлагает технология интеллектуальной замены имен?
14. Выведется ли сообщение системы безопасности об отключении части функций при ее открытии из надежного расположения?
15. В файле с каким расширением сохраняется файл с базой данных в формате Access 2007?
16. В файле с каким расширением сохраняется исполняемый файл базы данных Access 2007?
17. Можно ли для базы данных Access 2010 в окне документа открывать каждый объект в отдельном окне?
18. Для защиты данных от одновременной корректировки блокировка осуществляется на уровне страницы или записи?
19. С помощью каких интерфейсов обеспечивается доступ к данным Microsoft SQL Server?
20. Какой интерфейс используется в проекте Access для доступа к данным Microsoft SQL Server?

21. С какой целью создаются в Access веб-базы данных?
22. Какой режим используется в веб-базе данных для создания таблиц?
23. С помощью каких программ пользователи получают возможность работать с веб-базой данных из Интернета?
24. Доступна ли область навигации в веб-базе данных?

## Ответы

1. Да.
2. Для более удобного и быстрого выполнения часто используемых команд.
3. Главная (Home), Создание (Create), Внешние данные (External Data) и Работа с базами данных (Database Tools).
4. Область навигации.
5. Да.
6. Переместить на кнопку курсор.
7. Нет.
8. Таблицы, запросы, формы, отчеты, макросы и модули.
9. Составом полей.
10. Для однозначной идентификации каждой записи таблицы.
11. Отношениями записей типа "один-к-одному" (1 : 1) и "один-ко-многим" (1 : M).
12. Для связи двух таблиц с помощью одинакового поля в них.
13. Отслеживать изменение имен таблиц, полей, запросов, элементов управления и др. и выполнять их автозамену в зависимых объектах.
14. Нет.
15. accdb.
16. accde.
17. Да.
18. Блокировка может осуществляться на обоих уровнях.
19. ODBC и OLE DB.
20. OLE DB.
21. Для публикации и совместного использования в среде SharePoint.
22. Режим таблицы.
23. Веб-браузера и Access.
24. Нет.



## ГЛАВА 2

# Проектирование реляционной базы данных

База данных Access является *реляционной базой данных*. Такая база данных состоит из взаимосвязанных реляционных таблиц. На этапе проектирования базы данных для *выбранной предметной области* должна быть определена *логическая структура базы данных*. Проект логической структуры БД устанавливает состав реляционных таблиц, их структуру и логические связи между таблицами. При формировании структуры каждой таблицы определяется совокупность полей (столбцов), для каждого из которых определяется тип, размер данных и другие свойства. Для таблицы должен быть указан уникальный ключ, который может состоять из одного или нескольких полей.

При проектировании базы данных, отвечающей требованиям нормализации, между таблицами определяются логические связи типа 1 : M. Такие связи позволяют осуществлять в Access автоматическое поддержание связной целостности и непротиворечивости данных в базе.

## Этапы проектирования и создания базы данных

Для проектирования базы данных необходимо располагать описанием выбранной предметной области, которое должно охватывать реальные объекты и процессы, определять все необходимые источники информации для обеспечения предполагаемых запросов пользователя и решаемых в приложении задач.

Определение состава и структуры данных, которые должны быть загружены в базу данных, осуществляется на основе анализа предметной области. Структура данных предметной области может отображаться *информационно-логической моделью (ИЛМ)*. Если при построении такой модели обеспечены требования нормализации данных и она, соответственно, представлена в каноническом виде, далее легко определяется проект логической структуры нормализованной базы данных. На основе канонической модели можно создать реляционную базу без дублирования данных.

При разработке модели данных предметной области могут использоваться два подхода. В первом (аналитическом или процессном) сначала формулируются основные задачи, для решения которых строится база, выявляются информационные потребности задач приложения пользователя, и, соответственно, определяются состав и структура информационных объектов модели, а также связи между ними. При втором подходе (интуитивном) сразу устанавливаются типовые информационные объекты предметной области и их взаимосвязи. Наиболее рационально сочетание обоих подходов. Это связано с тем, что на начальном этапе, как правило, нет исчерпывающих сведений обо всех задачах. Использование такой технологии тем более оправдано, что гибкие средства создания реляционной базы данных позволяют на любом этапе разработки внести изменения в базу данных и модифицировать ее структуру без ущерба для введенных ранее данных.

Этапы проектирования и создания базы данных Access иллюстрирует схема, приведенная на рис. 2.1.

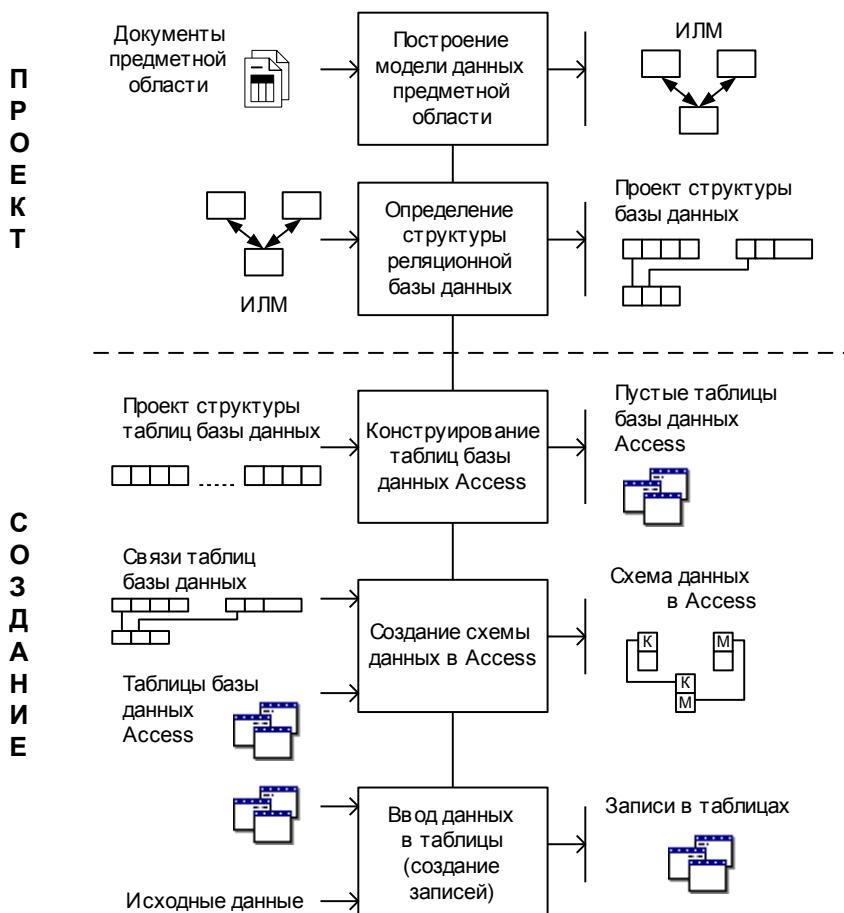


Рис. 2.1. Этапы проектирования и создания базы данных Access

В процессе разработки *канонической модели данных* предметной области для проектирования реляционной базы данных необходимо выделить *информационные объекты (ИО)*, соответствующие требованиям нормализации данных, и определить связи между ИО с типом отношений "один-ко-многим" (1 : M).

При определении проекта логической структуры реляционной базы данных каждый информационный объект канонической модели предметной области адекватно отображается реляционной таблицей, а связям между двумя информационными объектами соответствуют логические связи между парой соответствующих таблиц. Такие связи устанавливаются по уникальному ключу главной в связи таблицы. Во второй таблице, которая является подчиненной, поле связи может быть либо частью ее уникального ключа, либо не входить в состав ключа вовсе.

В процессе создания базы данных на компьютере сначала осуществляется конструирование ее таблиц средствами Access. Для поддержания целостности данных в каждой таблице определяется ключевое поле и правила проверки значений данных в полях. Далее создается схема данных, в которой устанавливаются логические связи таблиц. В схеме данных базы могут быть заданы параметры поддержания связной целостности данных.

*Связная целостность данных* означает, что в базе данных установлены и корректно поддерживаются взаимосвязи между записями разных таблиц при загрузке, добавлении и удалении записей в связанных таблицах, а также при изменении значений ключевых полей. При обеспечении связной целостности в подчиненной таблице не может существовать запись, для которой отсутствует связанная запись в главной таблице.

После формирования в Access схемы данных можно приступать к вводу данных в базу — загрузке с документов предметной области, являющихся источниками данных. В практических приложениях пользователя обычно не используется ввод непосредственно в таблицы, а применяются специально создаваемые экранные формы, играющие роль интерфейса пользователя. Фактически новый документ готовится (заполняется) на компьютере и сохраняется в базе данных.

Проектирование базы данных, основанное на построении нормализованной модели данных предметной области, позволяет легко получить логическую структуру реляционной базы данных Access, в которой автоматически поддерживаются целостность и непротиворечивость данных.

## Построение информационно-логической модели данных

Информационно-логическая модель (ИЛМ) отображает данные предметной области в виде совокупности информационных объектов (ИО) и связей между ними. Эта модель представляет данные, подлежащие хранению в базе данных. Каждый информационный объект в модели данных должен иметь уникальное имя.

## Информационные объекты

*Информационный объект* — это информационное описание некоторой сущности предметной области: реального объекта, процесса, явления или события. Информационный объект является совокупностью логически взаимосвязанных реквизитов, представляющих качественные и количественные характеристики сущности. Примерами сущностей являются: товар, поставщик, заказчик, поставка, отгрузка, сотрудник, отдел, студент, преподаватель, кафедра и т. п.

Информационный объект имеет множество реализаций — экземпляров *объекта*. Например, каждый экземпляр информационного объекта ТОВАР содержит значения реквизитов по товару определенного наименования. Экземпляр объекта должен однозначно определяться среди всего множества экземпляров, т. е. идентифицироваться значением *的独特的 (первичного) ключа* информационного объекта. Уникальность ключа означает, что любое значение ключа не может повторяться в каком-либо другом экземпляре объекта. *Простой ключ* состоит из одного реквизита. *Составной ключ* — из нескольких реквизитов. Таким образом, реквизиты информационного объекта подразделяются на ключевые и описательные, которые являются функционально зависимыми от ключа.

## Функциональные зависимости реквизитов

Информационные объекты могут быть выделены на основе описания предметной области путем определения *функциональных зависимостей между реквизитами* предметной области. Функциональная зависимость реквизитов информационного объекта устанавливает соответствие значений ключевых (определяющих) и неключевых (определяемых) реквизитов.

Необходимость установления функциональных зависимостей связана с требованием баз данных по однозначной определяемости любых данных для их размещения и доступа к ним. Например, для однозначной и полной определяемости реквизита *количество* в поставке товара нужно указать, что поставляется, т. е. товар (его идентификатор), и нужно указать, кем поставляется, т. е. указать поставщика (его идентификатор). Если возможны поставки одного товара поставщиком в разные сроки, то нужно также указать, когда осуществляется поставка, т. е. принять срок поставки, тоже как участвующий в идентификации поставки. Если не будет выявлена полная функциональная зависимость, количество поставки останется не определенным однозначно.

Если в функциональных зависимостях количества поставки не определен срок поставки, то нельзя отличить друг от друга несколько поставок одного и того же товара одним и тем же поставщиком, но в разные сроки. В этом случае имеет место неполная идентификация поставки на этапе проектирования. После реализации проекта это приведет к возможности загрузки в базу данных сведений о количестве только по одной поставке (первой введенной), т. к. для одного значения идентификатора Код товара + Код поставщика, т. е. одного товара и одного поставщика, может быть введено только одно значение количества поставляемого. Наглядная

иллюстрация этой ситуации, когда срок поставки не указан как ключевой (определяющий поставку), представлена в табл. 2.1.

**Таблица 2.1. Пример идентификации поставок товаров**

Реквизит	Код товара	Код поставщика	Срок поставки	Количество поставки	Результат в базе данных
Роль в функциональной зависимости	Ключевой	Ключевой		Зависимый	
	Идентификатор поставки				
Поставка 1	T1	П1	Февраль	10	Введена
Поставка 2	T1	П1	Апрель	20	Не введена

### ЗАМЕЧАНИЕ

При выявлении функциональных зависимостей реквизитов не рассматриваются арифметические зависимости (например, стоимость от количества), поскольку устанавливается только функциональная зависимость, определяющая логические связи описательных и ключевых реквизитов.

При графическом изображении модели данных каждый информационный объект представляется прямоугольником с обозначением его имени и идентификатора — ключа. Пример такого изображения для информационных объектов ТОВАР и ПОСТАВКА показан на рис. 2.2. Здесь код\_т (код товара) — простой ключ объекта ТОВАР, а код\_т + кпост (код поставщика) + срокп (срок поставки) — составной ключ объекта ПОСТАВКА.



**Рис. 2.2. Пример графического изображения информационных объектов с простым и составным ключами**

## Требования нормализации

Реквизиты каждого информационного объекта канонической модели данных должны отвечать требованиям, соответствующим *третьей нормальной форме* реляционной модели данных:

- ❖ информационный объект должен содержать уникальный идентификатор — ключ;
- ❖ все описательные реквизиты должны быть взаимонезависимы, т. е. между ними не должно быть функциональных зависимостей;

- ❖ все реквизиты, входящие в составной ключ, также должны быть взаимонезависимы;
- ❖ каждый описательный реквизит должен *функционально полно* зависеть от ключа, т. е. каждому значению ключа должно соответствовать только одно значение описательного реквизита, а при составном ключе описательные реквизиты должны зависеть целиком от всей совокупности реквизитов, образующих ключ;
- ❖ каждый описательный реквизит должен зависеть от ключа *нетранзитивно*, т. е. не должен зависеть через другой промежуточный реквизит.

### **ЗАМЕЧАНИЕ**

В случае транзитивной зависимости между реквизитами информационного объекта можно выполнить расщепление совокупности реквизитов с образованием двух информационных объектов вместо одного.

Выполнение требований нормализации обеспечивает построение канонической модели данных и создание на ее основе реляционной базы данных без дублирования описательных данных, а также возможность автоматического поддержания связной целостности данных средствами СУБД при обновлении базы данных — добавлении и удалении записей, изменении значений в ключевых полях.

## **Выделение информационных объектов предметной области**

Процесс выделения информационных объектов предметной области, отвечающих требованиям нормализации, может производиться на основе интуитивного или формального (аналитического) подхода. При интуитивном подходе сразу устанавливаются информационные объекты, соответствующие реальным объектам. Однако получаемая при этом информационно-логическая модель, как правило, требует дальнейших преобразований, в частности преобразования много-многозначных связей между объектами. При отсутствии достаточного опыта использования такого подхода возможны существенные ошибки. Последующая проверка выполнения требований нормализации обычно показывает необходимость уточнения структуры информационных объектов.

Теоретические основы аналитического подхода были разработаны и полно изложены известным американским ученым Дж. Мартином в его монографиях по организации баз данных. При аналитическом подходе в ходе исследования предметной области сначала необходимо выявить совокупность данных и различных сведений об объектах и процессах, характеризующих данную область, перечень документов, содержащих эти данные, а также комплекс задач и запросов, которые предполагается реализовать. Таким образом, определению структуры базы данных предшествует выявление информационных потребностей приложений пользователя. Основным источником данных являются справочные, плановые и оперативно-учетные документы.

На основе исследования составляется описание предметной области и документов, данные из которых нужно разместить в базе.

Далее выполняется *информационный анализ* предметной области с целью формализации и моделирования данных. Такая формализация необходима для их организации и обработки данных во внутримашинной сфере. При этом должен быть произведен семантический анализ данных и установлены функциональные зависимости реквизитов. Далее на их основе должны быть выявлены информационные объекты и логические взаимосвязи между ними. В результате данные предметной области будут структурированы.

## Информационный анализ и определение логической структуры информации

Информационный анализ включает:

- ❖ структурирование информации предметной области;
- ❖ формализацию и моделирование данных.

Структура информации внемашинной сферы находит отражение в ее представлении отдельными структурными единицами — *реквизитами*, их группировке в документах-источниках и упорядочении по классификационным признакам.

## Структурирование информации

Рассмотрим структурирование информации применительно к организационно-экономической сфере. Такая информация имеет дискретный характер и поэтому может быть структурирована, т. е. представлена как совокупность отдельных *структурных единиц информации*. Определим важнейшие виды структурных единиц информации:

- ❖ реквизит — простейшая структурная единица информации, неделимая на смысловом уровне, отражающая количественную или качественную характеристику сущности (объекта, процесса и т. п.) предметной области. Можно выделить реквизиты-признаки и реквизиты-основания:
  - ◆ *реквизит-признак* позволяет выделить (идентифицировать) объект из множества однотипных объектов (как правило, символное представление);
  - ◆ *реквизит-основание* содержит количественную характеристику объекта, процесса или другой сущности, определяющую их состояние (как правило, числовое значение).

Например, в плане поставок товаров реквизиты-признаки идентифицируют поставку, а реквизиты-основания определяют количество поставляемого товара, его стоимость;

- ❖ составная единица информации (СЕИ) — логически взаимосвязанная совокупность реквизитов.

Документ является примером составной единицы информации. Семантика и размещение реквизитов в форме документа определяют роль реквизитов в структуре информации, содержащейся в документе.

В процессе информационного семантического (смыслового) анализа нужно выявить функциональную зависимость реквизитов и определить реквизитный состав информационных объектов.

Для минимизации возможных ошибок целесообразно производить семантический анализ по каждой из форм документов в отдельности. Это связано с тем, что форма документа уже отображает структуру данных, т. к. любой документ объединяет логически взаимосвязанные реквизиты.

Функциональную зависимость реквизитов удобно изобразить графически в виде линий со стрелками, идущих от ключевого (определяющего) реквизита к описательному реквизиту (определяемому). Ключевой реквизит обычно отмечается особо. Функциональную зависимость удобно отображать непосредственно в таблице, где представлен состав реквизитов каждого документа. Это показано на примере реквизитов документа "Справочник товаров" (рис. 2.3), где каждый из описательных реквизитов однозначно определяется ключевым реквизитом Код товара.

Наименование реквизита	Имя реквизита	Функциональные зависимости
Код товара	KODT	—
Наименование	NAIM	←
Цена за единицу	CENA	←
Единица измерения	Е	←

**Рис. 2.3.** Функциональная зависимость реквизитов документа "Справочник товаров"

## Выделение информационных объектов

На основе описания предметной области необходимо выявить документы-источники и их реквизиты, подлежащие хранению в базе данных. Затем надо перейти к информационному анализу этих документов для определения функциональных зависимостей и выявления информационных объектов.

Рассмотрим порядок действий, которые могут быть использованы для выделения информационных объектов, отвечающих указанным ранее требованиям нормализации.

### 1. Определяются функциональные зависимости между реквизитами документа.

Для этого анализируется роль реквизитов в структуре информации документа.

Сначала целесообразно выявить реквизит (один или несколько), который играет роль общего идентификатора всей информации документа. Как правило, к таким реквизитам относятся номер документа, идентификатор подразделения предприятия, выпускающего документ, период действия оформления документа и т. п. От такого идентификатора документа будут функционально полно зависимыми некоторые описательные реквизиты в общей части документа (на-

пример, идентификатор документа-основания). Заметим, что в табличной части документа такие реквизиты, как количество товара и стоимость, будут частично функционально зависимыми от общего идентификатора документа.

Все справочные реквизиты реальных объектов (товаров, предприятий, подразделений и т. п.), как в общей, так и в табличной части функционально полно определяются идентификаторами этих объектов (см. рис. 2.3). В результате для каждого определяемого реквизита должны быть выявлены реквизиты (ключевые), которые в совокупности однозначно его определяют (одному значению ключа соответствует одно значение описательного реквизита).

Для графического отображения функциональной зависимости проводится линия связи со стрелкой к зависимому реквизиту от определяющего его реквизита.

2. *В результате просмотра выявленных функциональных зависимостей выбираются зависимые реквизиты и для каждого из них устанавливаются все его ключевые реквизиты, т. е. те (один или несколько), которые в совокупности определяют его однозначно.*

Такое соответствие описательных и ключевых реквизитов удобно представить в таблице, форма которой с примером заполнения представлена далее (табл. 2.2).

**Таблица 2.2. Соответствие описательных и ключевых реквизитов**

Описательные (зависимые) реквизиты	Ключевые реквизиты	Признак ключа	Имя информационного объекта, включающего реквизиты
Количество поставки	Код товара + + Код поставщика + + Срок поставки	Уникальный составной	ПОСТАВКА
Наименование товара	Код товара	Уникальный простой	ТОВАР
Наименование поставщика	Код поставщика	Уникальный простой	ПОКУПАТЕЛЬ

3. *Группируются реквизиты, одинаково зависимые от ключевых реквизитов. Полученные группы зависимых реквизитов вместе с их ключевыми реквизитами образуют реквизитный состав соответствующих информационных объектов. Если в группе несколько ключевых реквизитов, то они являются составным ключом информационного объекта.*

При использовании приведенных правил не требуется отдельно преобразовывать транзитивные зависимости реквизитов. При такой зависимости некоторые реквизиты являются одновременно зависимыми и ключевыми и, соответственно, будут представлены в группе зависимых и ключевых.

После выделения информационных объектов необходимо сформировать их окончательное описание. В таком описании кроме состава реквизитов и указания ключа может быть представлена также семантика информационных объектов — их смысловое определение.

## **ЗАМЕЧАНИЕ**

Обычно при использовании приведенных правил сразу оказываются выделенными объекты, играющие роль связки между объектами, находящимися в отношении "многие-ко-многим" ( $M : N$ ). Соответственно, в модели можно ограничиться рассмотрением только одно-многозначных связей.

Совокупность выделенных информационных объектов после определения связей между объектами позволяет получить информационно-логическую модель, не предусматривающую дальнейших преобразований для создания реляционной базы данных, отвечающей требованиям нормализации.

## **Выделение информационных объектов на примере предметной области "Поставка товаров"**

Рассмотрим выделение информационных объектов на примере предметной области "Поставка товаров".

### **Описание предметной области**

Пусть необходимо построить базу данных, содержащую информацию о планируемых поставках товаров покупателям и фактических отгрузках товаров в соответствии с планом поставок. Такая база данных должна обеспечить подготовку, хранение и просмотр данных по договорам с покупателями и по фактическим отгрузкам товаров, а также по анализу выполнения договорных обязательств на поставку по срокам и объемам.

Информационное обеспечение такого приложения пользователя включает:

- ❖ справочную информацию о поставляемых товарах;
- ❖ справочную информацию о покупателях (заказчиках);
- ❖ справочную информацию о складах предприятия, где хранится товар;
- ❖ данные о плановых поставках товаров;
- ❖ оперативно-учетные данные об отгрузках товаров со складов покупателям.

В результате анализа предметной области выявляются документы — источники для создания базы данных.

Справочная информация содержится в документах "Справочник товаров", "Справочник покупателей", "Справочник складов". На рис. 2.4—2.6 приведены формы этих документов.

СПРАВОЧНИК ТОВАРОВ, ПОСТАВЛЯЕМЫХ ФИРМОЙ

Код товара	Наименование	Единица измерения	Цена	НДС
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____

**Рис. 2.4. Форма документа "Справочник товаров, поставляемых фирмой"**

## СПРАВОЧНИК СКЛАДОВ ФИРМЫ

Фирма /код, наименование/

Код склада	Наименование	Адрес	Отв. лицо
_____	_____	_____	_____
_____	_____	_____	_____

Рис. 2.5. Форма документа "Справочник складов фирмы"

## СПРАВОЧНИК ПОКУПАТЕЛЕЙ ФИРМЫ

Код	ИНН	Наименование	Адрес	Тел.	№ расч. счета	Банк
—	—	_____	_____	—	_____	—
—	—	_____	_____	—	_____	—

Рис. 2.6. Форма документа "Справочник покупателей фирмы"

Информация о планируемых поставках содержится в договорах. Форма договора приведена на рис. 2.7.

ДОГОВОР № \_\_\_\_\_

" \_\_\_\_ " \_\_\_\_\_ 200\_\_ г.

Поставщик

ИНН, код \_\_\_\_\_  
 наименование \_\_\_\_\_  
 адрес \_\_\_\_\_  
 телефон \_\_\_\_\_  
 банк \_\_\_\_\_  
 расчетный счет \_\_\_\_\_

Покупатель

ИНН, код \_\_\_\_\_  
 наименование \_\_\_\_\_  
 адрес \_\_\_\_\_  
 телефон \_\_\_\_\_  
 банк \_\_\_\_\_  
 расчетный счет \_\_\_\_\_

Код товара	Наиме- нование	Единица измерения	Цена	Срок поставки (месяц)	Мин. партия поставки	Кол-во	Сумма

Сумма всего \_\_\_\_\_

Рис. 2.7. Форма договора на поставку товаров

Учетная информация с данными по фактической отгрузке товаров покупателю со склада фирмы в соответствии с договорами содержится в расходных накладных (рис. 2.8).

<b>НАКЛАДНАЯ № _____</b>																											
" ____ " 200__ г.																											
<b><u>Поставщик</u></b>	<b><u>Покупатель</u></b>																										
ИНН, код _____ наименование _____	ИНН, код _____ наименование _____																										
<b><u>Склад-грузоотправитель</u></b>																											
Код _____ наименование _____																											
Договор № ____ от " ____ " 200__ г.																											
<table border="1"> <thead> <tr> <th>Код товара</th> <th>Наименование</th> <th>Единица измерения</th> <th>Цена</th> <th>Ставка НДС</th> <th>Количество</th> <th>Сумма</th> </tr> </thead> <tbody> <tr> <td> </td> </tr> <tr> <td> </td> </tr> </tbody> </table>							Код товара	Наименование	Единица измерения	Цена	Ставка НДС	Количество	Сумма														
Код товара	Наименование	Единица измерения	Цена	Ставка НДС	Количество	Сумма																					
Сумма всего _____																											
Отпустил _____			ФИО материально отв. лица _____																								

**Рис. 2.8.** Форма накладной с данными по фактической отгрузке товаров

Особо отметим, что документы предметной области не только дают возможность выявить структуру данных, но также являются основой для разработки электронных форм ввода/вывода и отчетов для печати документов после их подготовки на компьютере и сохранения в базе данных.

Как отмечалось ранее, информационный анализ для выделения объектов целесообразно производить по каждой из форм документов, содержащих данные, которые должны быть размещены в базе.

### **Выделение объектов справочной информации**

Каждый справочник, как правило, содержит только табличную часть, и в ней имеется реквизит, играющий роль идентификатора строки (код или номер).

Например, в справочнике товаров идентификатором является Код товара, который однозначно определяет описательные реквизиты — Наименование товара,

Единицы измерения и т. д. В справочнике складов идентификатором является Код склада, который однозначно определяет Наименование склада и другие реквизиты. В справочнике покупателей уникальным идентификатором является ИНН, а также Код, который может иметь небольшую длину и присваивается для удобства внутрисистемного использования.

**Анализ справочников товаров и покупателей.** Определим функциональные зависимости между реквизитами документа "Справочник товаров", предварительно составив их перечень, как показано на рис. 2.9. Присвоим реквизитам справочника товаров сокращенные обозначения — имена.

Особенность такого простейшего по структуре данных документа состоит в том, что все реквизиты документа содержатся в таблице. Из анализа документа очевидно, что реквизиты Наименование товара (НАИМ\_ТОВ), Единица измерения (ЕИ), Цена (ЦЕНА), Ставка НДС являются описательными. Каждый из них функционально полно зависит только от уникального идентификатора товара — Кода товара (КОД\_ТОВ). Таким образом, Код\_товара является ключевым.

Отобразим зависимости каждого описательного реквизита товара от кода товара, который однозначно их определяет, линиями со стрелкой от ключевого реквизита к описательному.

Реквизиты справочника товаров	Имя реквизита	Функциональные зависимости
Код товара	КОД_ТОВ	
Наименование товара	НАИМ_ТОВ	←
Единица измерения	ЕИ	←
Цена	ЦЕНА	←
Ставка НДС	СТАВКА_НДС	←

Рис. 2.9. Функциональные зависимости реквизитов справочника товаров

Для документов с такой простейшей структурой данных нет необходимости в дополнительных действиях по выделению информационных объектов, т. к. здесь все реквизиты образуют одну группу реквизитов с общим ключом, т. е. все реквизиты простейшего справочника образуют один информационный объект. Назовем такой объект для данного документа ТОВАР с ключом Код товара (КОД\_ТОВ).

Аналогично легко определить информационный объект по справочнику покупателей. Если в качестве ключа принять Код покупателя (КОД\_ПОК), более короткий по сравнению с ИНН, он однозначно определит реквизиты — ИНН, Наименование, Адрес, Телефон, Банк, Номер расчетного счета. Соответственно выделяется информационный объект ПОКУПАТЕЛЬ с ключом Код покупателя.

### ЗАМЕЧАНИЕ

ИНН можно использовать в качестве альтернативного ключа и создать по нему уникальный индекс в базе данных.

Реквизитный состав информационных объектов справочника товаров и справочника покупателей представлен в табл. 2.3.

**Таблица 2.3. Группировка реквизитов по информационным объектам ТОВАР и ПОКУПАТЕЛЬ**

Реквизиты объекта	Признак ключа	Имя информационного объекта	Семантика объекта
КОД_ТОВ	Простой уникальный	ТОВАР	Сведения о поставляемых товарах
НАИМ_ТОВ ЦЕНА ЕИ СТАВКА_НДС			
КОД_ПОК	Простой уникальный	ПОКУПАТЕЛЬ	Сведения о покупателях фирмы
ИНН НАИМ_ПОК АДРЕС_ПОК ТЕЛ НОМ_РСЧ БАНК			

**Анализ документа "Справочник складов фирмы".** Определим функциональные зависимости между реквизитами документа "Справочник складов фирмы", предварительно составив их перечень (рис. 2.10). Присвоим реквизитам сокращенные обозначения — имена. Все описательные реквизиты склада — Наименование склада (НАИМ\_СК), Адрес (АДРЕС\_СК), отв. лицо (ОТВ\_ЛИЦО) однозначно определены Кодом склада (КОД\_СК), который поэтому является ключевым. Обратим внимание, что в общей части документа указан Код фирмы (КОД\_Ф) — один для всего списка складов. Очевидно, каждому значению кода склада соответствует только одно значение кода фирмы, т. е. можно считать, что имеет место полная функциональная зависимость кода фирмы от кода склада. Наименование фирмы определяется однозначно кодом фирмы.

### **ЗАМЕЧАНИЕ**

Если зависимость между КОД\_Ф и КОД\_СК не выявлена, то все множество реквизитов документа разделится на два не связанных между собой подмножества, а это нелогично для реквизитов одного документа.

Все установленные функциональные зависимости реквизитов документа "Справочник складов фирмы" отражены на рис. 2.10.

Заметим, что реквизит КОД\_Ф одновременно выступает в роли описательного реквизита в одной связи и ключевого — в другой. Таким образом, здесь мы сталки-

ваемся с транзитивной зависимостью. Реквизит `НАИМ_Ф` транзитивно зависит от `КОД_СК` через `КОД_Ф`. Тем не менее специальных действий по расщеплению этой зависимости не потребуется при использовании приведенных выше правил.

Реквизиты справочника складов	Имя реквизита	Функциональные зависимости
Код фирмы	КОД_Ф	
Наименование фирмы	НАИМ_Ф	←
Код склада	КОД_СК	—
Наименование склада	НАИМ_СК	←
Адрес	АДРЕС_СК	←
Отв. лицо	ОТВ_ЛИЦО	←

**Рис. 2.10.** Функциональная зависимость реквизитов справочника складов фирмы

Выберем по функциональным связям реквизиты, зависимые от каких-либо других реквизитов, и укажем для них ключевые реквизиты.

Так, при просмотре списка реквизитов сверху находим первый зависимый реквизит `КОД_Ф`, к которому подходит стрелка, и устанавливаем реквизит (ключевой), от которого идет стрелка — `КОД_СК`. Далее находим второй зависимый (описательный) реквизит `НАИМ_Ф` и устанавливаем его ключевой `КОД_Ф`. Аналогично находим описательный `НАИМ_СК` и устанавливаем его ключевой `КОД_СК` и т. д. Выявленное соответствие описательных и ключевых реквизитов представлено в табл. 2.4.

**Таблица 2.4.** Соответствие описательных и ключевых реквизитов документа "Справочник складов фирмы"

Описательные (зависимые) реквизиты	Ключевые реквизиты	Информационный объект, включающий реквизиты
КОД_Ф	КОД_СК	СКЛАД
НАИМ_Ф	КОД_Ф	ФИРМА
НАИМ_СК	КОД_СК	СКЛАД
АДРЕС_СК	КОД_СК	СКЛАД
ОТВ_ЛИЦО	КОД_СК	СКЛАД

Сгруппируем реквизиты, зависимые от одних и тех же ключевых реквизитов, и объединим их с ключевыми реквизитами в один информационный объект.

Результат группировки реквизитов документа "Справочник складов фирмы" приведен в табл. 2.5.

**Таблица 2.5. Группировка реквизитов по информационным объектам документа "Справочник складов фирмы"**

Реквизиты объекта	Признак ключа	Имя информационного объекта	Семантика объекта
КОД_СК	Простой уникальный	СКЛАД	Сведения о складах фирмы
КОД_Ф НАИМ_СК АДРЕС_СК ОТВ_ЛИЦО			
КОД_Ф	Простой уникальный	ФИРМА	Сведения о фирме
НАИМ_Ф			

Таким образом, на основе анализа документа "Справочник складов фирмы" выделены два информационных объекта — ФИРМА и СКЛАД.

Заметим, что особенность объекта ФИРМА состоит в том, что он имеет единственный экземпляр, поэтому данный объект можно не отображать в базе данных отдельной таблицей.

### **Выделение объектов плановой и учетной информации**

**Анализ документа "Договор на поставку товаров".** Определим функциональные зависимости реквизитов документа "Договор", предварительно составив их перечень (рис. 2.11). Присвоим реквизитам сокращенные обозначения — имена.

Рассмотрим функциональные зависимости между реквизитами общей части документа "Договор". Номер договора присваивается в порядке подготовки нового документа. Этот номер является уникальным среди всех номеров договоров.

Каждый из реквизитов: Дата заключения договора, Идентификатор покупателя (в качестве него примем код, соответствующий ИНН по справочнику покупателей) — имеет единственное значение в договоре. Соответственно, каждый из этих реквизитов однозначно определяется идентификатором документа — Номером договора. Общим идентификатором договора определяется также однозначно реквизит Сумма всего.

Кодом покупателя однозначно определяются описательные реквизиты покупателя — Наименование, ИНН, Адрес, Телефон, Банк, Расчетный счет. В таблице зависимостей эти реквизиты можно не отображать, поскольку информационный объект, образованный этими реквизитами, был уже выделен на основе справочника покупателей.

Описательные реквизиты фирмы, выступающей в данном документе в качестве поставщика, определяются однозначно идентификатором фирмы.

Рассмотрим функциональные зависимости реквизитов табличной части договора. Табличная часть (спецификация договора) содержит реквизиты, имеющие множество значений в соответствующих столбцах, т. к. договор может содержать не-

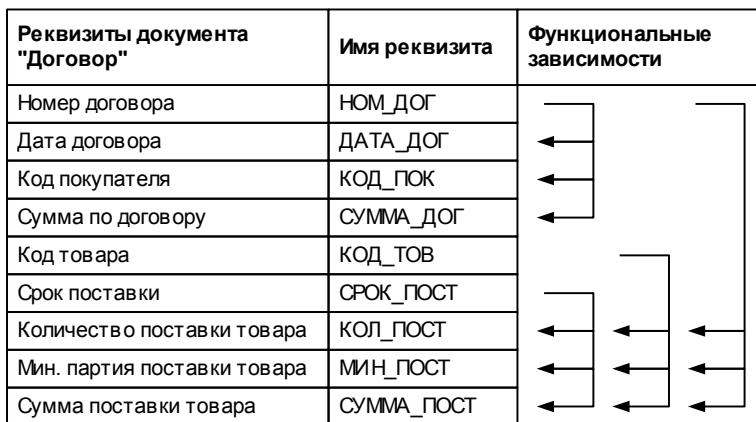
сколько наименований товаров. Среди этих реквизитов особую роль играет Код товара, который фактически является идентификатором каждой строки в документе.

### **ЗАМЕЧАНИЕ**

Поскольку в договоре может быть несколько товаров, номер договора не определяет однозначно товар, указанный в документе, и, соответственно, не может быть установленна функциональная зависимость Кода товара от Номера договора.

Описательные реквизиты товара наименование, Единица измерения, Цена однозначно определены Кодом товара. Эти реквизиты можно не включать в таблицу зависимостей, поскольку их взаимосвязи были установлены ранее, при анализе справочника товаров. Что касается реквизитов-оснований документа, таких как Количество поставки товара, Сумма поставки товара, Мин. партия поставки товара — эти реквизиты внутри документа идентифицируются Кодом товара в соответствующей строке, а полная идентификация среди всех договоров образуется добавлением Номера договора к Коду товара. Все это справедливо лишь в том случае, если для любого товара Срок поставки может быть только один. Причем в этом случае реквизиты Количество поставки товара, Сумма поставки товара, Мин. партия поставки товара и Срок поставки товара функционально полно будут зависеть от составного ключа — Номер договора + Код товара.

Допустим, что в договоре для одного товара возможно несколько сроков поставки, тогда срок поставки должен войти в общий идентификатор для реквизитов Количество поставки товара, Сумма поставки товара и Мин. партия поставки товара. Таким образом, эти реквизиты будут функционально полно зависеть от составного ключа Номер договора + Код товара + Срок поставки (см. рис. 2.11).



**Рис. 2.11. Функциональные зависимости реквизитов документа "Договор"**

**Выделение информационных объектов по документу "Договор".** Проанализировав выявленные функциональные взаимосвязи реквизитов, установим, от каких реквизитов зависит каждый реквизит, к которому подходит стрелка. Таким об-

разом, определим соответствие описательных и ключевых реквизитов. Затем сгруппируем реквизиты, одинаково зависимые от ключевых, и объединим их с ключевыми реквизитами в один информационный объект (ИО). Результат группировки по ИО реквизитов документа "Договор" представлен в табл. 2.6.

**Таблица 2.6. Группировка реквизитов по информационным объектам документа "Договор"**

Реквизиты объекта	Признак ключа	Имя информационного объекта	Семантика объекта
НОМ_ДОГ	Простой уникальный	ДОГОВОР	Общие сведения по договору
ДАТА_ДОГ КОД_ПОК СУММА_ДОГ			
НОМ_ДОГ + КОД_ТОВ + СРОК_ПОСТ	Составной уникальный	ПОСТАВКА_ПЛАН	Сведения о поставках товаров по договору
КОЛ_ПОСТ МИН_ПОСТ СУММА_ПОСТ			

**Анализ документа "Накладная".** Рассмотрим функциональные зависимости реквизитов общей части накладной. Номер накладной присваивается в порядке подготовки нового документа. Этот номер можно считать уникальным только среди всех номеров накладных, выписанных на данном складе (т. е. он не повторяется на данном складе). Для уникальной идентификации накладных по всей фирме надо принять составной идентификатор Номер накладной + Код склада.

Реквизиты Дата выписки накладной (отгрузки товаров) и Номер договора имеют единственное значение в накладной и, соответственно, каждый из них однозначно определяется идентификатором накладной (Номер накладной + Код склада). Общим идентификатором накладной определяется также однозначно реквизит Сумма всего.

Дата заключения договора однозначно определяется Номером договора, что было уже учтено при анализе договора.

Код покупателя, выбранный ранее его идентификатором, имеет единственное значение в накладной, т. е. он однозначно определяется идентификатором накладной. Однако, как было рассмотрено ранее, Код покупателя однозначно определяется в первичном документе "Договор" Номером договора. То есть здесь имеет место транзитивная зависимость идентификатора покупателя от номера договора через идентификатор накладной. Поэтому будем учитывать функциональную зависимость Кода покупателя ТОЛЬКО ОТ Номера договора.

ИИН и Наименование покупателя однозначно определяются Кодом покупателя, что уже было учтено при анализе справочника покупателей.

Описательные реквизиты фирмы (ИНН и Наименование фирмы), выступающей в данном документе в качестве поставщика, определяются однозначно идентификатором фирмы — Кодом фирмы. Как было принято ранее, объект ФИРМА не целесообразно отображать отдельным объектом в базе данных.

Описательные реквизиты Наименование склада и Отв. лицо (ФИО ответственного лица) однозначно определены Кодом склада, что уже было учтено при анализе справочника покупателей.

### **ЗАМЕЧАНИЕ**

Код покупателя, Номер договора, Код склада играют в документе важную роль связи накладной с соответствующими справочными и плановыми данными. К этим данным относятся данные по покупателю (которому отпущен товар), данные по складу (с которого отпущен товар), данные по договору (на основании которого выписывается накладная). Как видно из таблиц функциональных зависимостей, реквизиты Код покупателя, Номер договора с одной стороны являются ключевыми для соответствующих данных, а с другой — описательными для накладной.

Рассмотрим функциональные зависимости реквизитов табличной части накладной. Табличная часть содержит реквизиты, имеющие множество значений в соответствующих столбцах, т. к. накладная может содержать несколько наименований товаров, отпущенных покупателю. Среди этих реквизитов особую роль играет Код товара, который является идентификатором каждой строки в документе.

### **ЗАМЕЧАНИЕ**

Поскольку в накладной может быть несколько товаров, идентификатор накладной не определяет однозначно товар, указанный в документе, и, соответственно, не может быть установлена функциональная зависимость Кода товара от идентификатора накладной.

Описательные реквизиты товара Наименование, Единица измерения, Цена, Ставка НДС однозначно определены Кодом товара, что уже было учтено при анализе справочника покупателей.

Реквизиты-основания накладной Количество отгруженного товара и Сумма за товар определяют количественные характеристики объекта — отгрузка товаров. Эти реквизиты внутри одной накладной идентифицируются Кодом товара в соответствующей строке, а полная идентификация на всем множестве накладных образуется добавлением к Коду товара идентификатора накладной. Таким образом, реквизиты Количество отгруженного товара и Сумма за товар однозначно определяются составным идентификатором Номер накладной + Код склада + Код товара.

На рис. 2.12 наглядно представлены все рассмотренные функциональные зависимости реквизитов накладной.

**Выделение информационных объектов по накладной.** Проанализировав выявленные функциональные взаимосвязи реквизитов, установим, от каких реквизитов зависит каждый реквизит, к которому подходит стрелка. Таким образом, определим соответствие описательных и ключевых реквизитов накладной. Затем сгруппируем реквизиты, одинаково зависящие от ключевых, и объединим их с ключевыми реквизитами в один информационный объект (ИО). Результат группировки по ИО реквизитов накладной представлен в табл. 2.7.



**Рис. 2.12.** Функциональные зависимости реквизитов накладной

**Таблица 2.7.** Группировка реквизитов по информационным объектам документа "Накладная"

Реквизиты информационного объекта	Признак ключа	Имя информационного объекта	Семантика информационного объекта (описание)
НОМ_НАКЛ + КОД_СК	Составной уникальный	НАКЛАДНАЯ	Общие сведения по Накладной
ДАТА_ОТГР НОМ_ДОГ СУММА_НАКЛ			
НОМ_НАКЛ + КОД_СК + КОД_ТОВ	Составной уникальный	ОТГРУЗКА	Данные по отгрузке товара
КОЛ_ОТГР СУММА_ОТГР			

## Связи информационных объектов

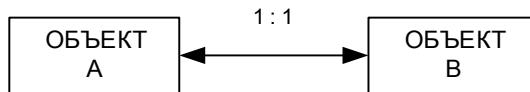
Следующим шагом проектирования после выявления информационных объектов является определение связей между ними. Связь устанавливается между двумя информационными объектами. Наличие связи, как правило, определяется природой реальных объектов, процессов или явлений, отображаемых этими информационными объектами. Связь между объектами существует, если логически взаимосвязаны экземпляры этих информационных объектов. Например, связи имеются между такими парами объектов, как поставщик—товар, склад—товар, кафедра—преподаватель, группа—студент и т. п.

## Тип связи информационных объектов

Связи информационных объектов могут быть разного типа:

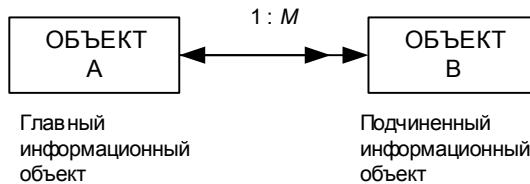
- ❖ одно-однозначные ( $1 : 1$ );
- ❖ одно-многозначные ( $1 : M$ );
- ❖ много-многозначные ( $M : N$ ).

*Одно-однозначные связи* имеют место, когда каждому экземпляру первого объекта (A) соответствует только один экземпляр второго объекта (B), и наоборот, каждому экземпляру второго объекта (B) соответствует только один экземпляр первого объекта (A). Следует заметить, что такие объекты легко могут быть объединены в один, структура которого образуется объединением реквизитов обоих исходных объектов, а ключевым реквизитом может быть выбран любой из альтернативных ключей, т. е. ключей исходных объектов. Графическое изображение одно-однозначной связи приведено на рис. 2.13. Примерами одно-однозначных связей являются пары вида: группа—староста, фирма — расчетный счет в банке и т. п.



**Рис. 2.13.** Графическое изображение одно-однозначных отношений объектов

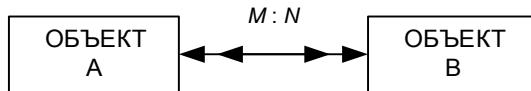
*Одно-многозначные связи* ( $1 : M$ ) — это такие связи, когда каждому экземпляру одного объекта (A) может соответствовать несколько экземпляров другого объекта (B), а каждому экземпляру второго объекта (B) может соответствовать только один экземпляр первого объекта (A). Графическое изображение одно-многозначной связи приведено на рис. 2.14.



**Рис. 2.14.** Графическое изображение одно-многозначных отношений объектов

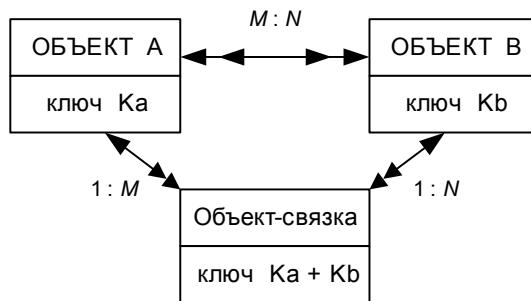
В такой связи объект А является главным, а объект В — подчиненным, т. е. имеет место иерархическая подчиненность объекта В объекту А. Простейшими примерами одно-многозначных связей объектов являются пары объектов: подразделения—сотрудники, кафедра—преподаватель, группа—студент и т. п.

*Много-многозначные связи ( $M : N$ )* — это такие связи, когда каждому экземпляру одного объекта (A) могут соответствовать несколько экземпляров второго объекта (B), и наоборот, каждому экземпляру второго объекта (B) может соответствовать несколько экземпляров первого объекта (A). Графическое изображение связи типа  $M : N$  показано на рис. 2.15.



**Рис. 2.15.** Графическое изображение много-многозначных отношений объектов

Много-многозначные связи не могут непосредственно реализовываться в реляционной базе данных. Поэтому в том случае, когда будут выявлены такие связи, может возникнуть необходимость их преобразования путем введения дополнительного объекта "связка". Исходные объекты будут связаны с этим объектом одно-многозначными связями. Таким образом, объект-связка является подчиненным в одно-многозначных связях по отношению к каждому из исходных объектов (рис. 2.16).



**Рис. 2.16.** Преобразование связи типа  $M : N$  с помощью объекта-связки

Объект-связка должен иметь идентификатор, образованный из идентификаторов исходных объектов  $Ка$  и  $Кb$  (см. рис. 2.16).

При рассмотренном подходе к выделению информационных объектов объект-связка, как правило, выявляется в результате анализа функциональных зависимостей реквизитов, рассмотренного выше. Много-многозначные связи в этом случае не требуют специальной реализации, т. к. осуществляются через объект, выполняющий роль объекта-связки.

Примером много-многозначных связей является пара вида поставщики—товары, если один поставщик поставляет разные наименования товаров, а товар одного наименования может поставляться несколькими поставщиками.

## Определение связей между информационными объектами

Определение связей между информационными объектами и типы отношений, которыми они характеризуются, рассмотрим на примере предметной области "Поставка товаров".

Связи между объектами ПОКУПАТЕЛЬ → ДОГОВОР характеризуются одно-многозначными отношениями ( $1 : M$ ), т. к. с одним покупателем может быть заключено несколько договоров, а один договор всегда заключается с конкретным покупателем.

Поскольку накладные строго привязаны к конкретному договору, а по одному договору может быть оформлено несколько накладных, между объектами ДОГОВОР и НАКЛАДНАЯ имеет место связь типа  $1 : M$ .

Характерным случаем одно-многозначных связей являются связи объектов, образованные из документов с табличной частью (спецификацией). В рассматриваемой предметной области по документу "Договор" был выделен объект ДОГОВОР, соответствующий общей части документа, и объект ПОСТАВКА\_ПЛАН, соответствующий строкам табличной части документа. Очевидна одно-многозначная связь между этими объектами ДОГОВОР → ПОСТАВКА\_ПЛАН, поскольку в одном документе всегда содержится некоторое множество строк, а каждая строка принадлежит только одному документу.

По документу "Накладная" были выделены два объекта, между которыми также имеет место одно-многозначная связь НАКЛАДНАЯ → ОТГРУЗКА.

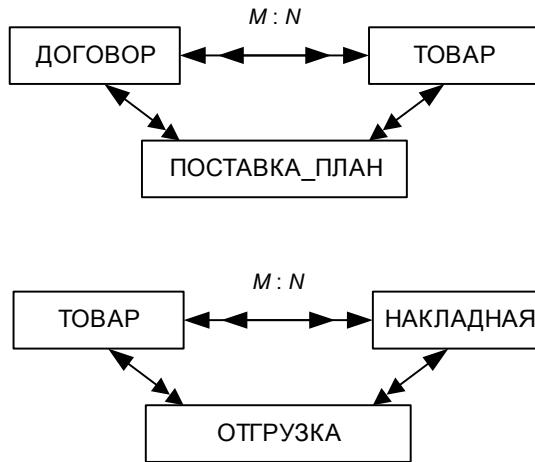
Очевидно наличие связи между объектами ТОВАР → ПОСТАВКА\_ПЛАН. Эту связь также определяют одно-многозначные отношения, поскольку каждый экземпляр поставки (одна из строк спецификации договора) — это данные по одному товару, а товар одного наименования может участвовать в разных плановых поставках товара (одного или разных договоров).

Аналогично устанавливается связь между объектами ТОВАР → ОТГРУЗКА, которые также находятся в одно-многозначных отношениях.

Связь между объектами СКЛАД → НАКЛАДНАЯ может быть установлена как одно-многозначная, поскольку по условиям рассматриваемой предметной области на каждом складе выписывается некоторое множество накладных, но каждая накладная выписывается на конкретном складе.

Следует отметить, что объект ПОСТАВКА\_ПЛАН фактически играет роль объекта-связки в много-многозначных отношениях объектов ДОГОВОР и ТОВАР, а объект ОТГРУЗКА играет роль объекта-связки в много-многозначных отношениях объектов НАКЛАДНАЯ и ТОВАР (рис. 2.17).

В табл. 2.8 перечислены все одно-многозначные связи между объектами и, соответственно, определены главные и подчиненные информационные объекты в этих связях.



**Рис. 2.17.** Примеры много-многозначных отношений информационных объектов

**Таблица 2.8.** Связи информационных объектов

Главный объект	Подчиненный объект	Тип связи
ПОКУПАТЕЛЬ	ДОГОВОР	1 : M
ДОГОВОР	ПОСТАВКА_ПЛАН	1 : M
НАКЛАДНАЯ	ОТГРУЗКА	1 : M
ТОВАР	ПОСТАВКА_ПЛАН	1 : M
ТОВАР	ОТГРУЗКА	1 : M
СКЛАД	НАКЛАДНАЯ	1 : M
ДОГОВОР	НАКЛАДНАЯ	1 : M

## Информационно-логическая модель предметной области

На рис. 2.18 представлена информационно-логическая модель в каноническом виде для рассматриваемой предметной области, построенная в соответствии с выявленными информационными объектами и связями между ними.

Объекты канонической модели размещены по уровням. На нулевом уровне размещаются объекты, не подчиненные никаким другим объектам. Уровень остальных объектов определяется наиболее длинным путем к объекту от нулевого уровня. Такое размещение объектов дает представление об их иерархической подчиненности, делает модель более наглядной и облегчает понимание одно-много-значных отношений между объектами.

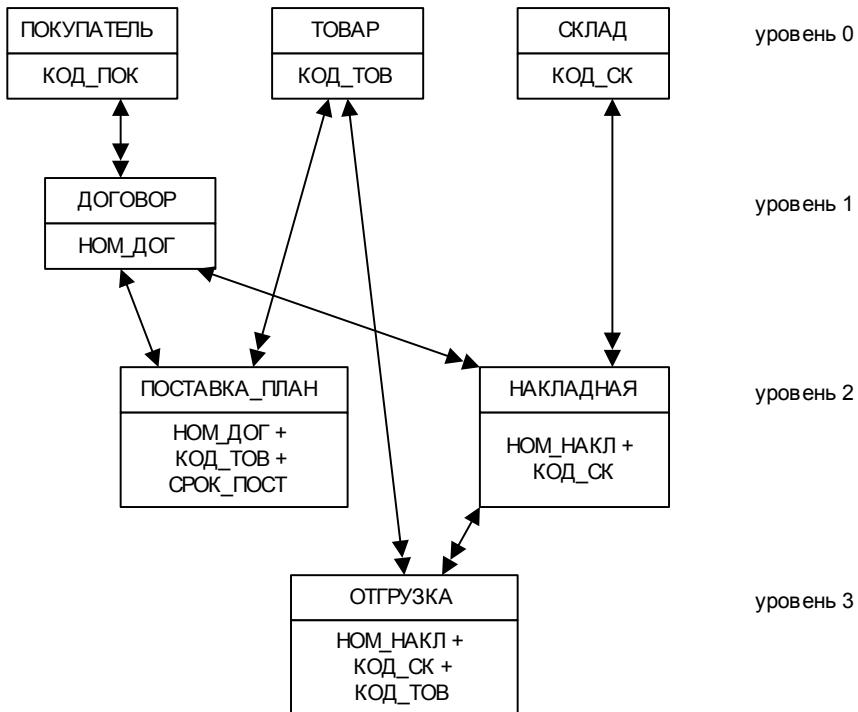


Рис. 2.18. Информационно-логическая модель предметной области "Поставка товаров"

### ЗАМЕЧАНИЕ

Для определения уровня объектов на графике ИЛМ можно, условно удалив объекты нулевого уровня, найти объекты первого уровня. К объектам этого уровня следует отнести объекты, не подчиненные теперь никаким другим объектам. Аналогично определяются объекты каждого следующего уровня. При большом количестве объектов в ИЛМ аналогичные действия выполняются на матрице смежности модели.

## Логическая структура реляционной базы данных

Логическая структура реляционной базы данных Access является адекватным отображением полученной информационно-логической модели предметной области. Для канонической модели не требуется дополнительных преобразований. Каждый информационный объект модели данных отображается соответствующей реляционной таблицей. Структура реляционной таблицы определяется реквизитным составом соответствующего информационного объекта, где каждый столбец (поле) соответствует одному из реквизитов объекта. Ключевые реквизиты объекта обра-

зуют уникальный ключ реляционной таблицы. Для каждого столбца таблицы (поля) задается тип, размер данных и другие свойства. Строки (записи) таблицы соответствуют экземплярам объекта и формируются при загрузке таблицы.

Связи между объектами модели данных реализуются одинаковыми реквизитами — *ключами связи* в соответствующих таблицах. При этом ключом связи типа 1 : M всегда является уникальный ключ главной таблицы. Ключом связи в подчиненной таблице является либо некоторая часть уникального ключа в ней, либо поле, не входящее в состав первичного ключа (например, код фирмы в таблице склад). Ключ связи в подчиненной таблице называется *внешним ключом*.

Все связи в полученной информационно-логической модели предметной области "Поставка товара" характеризуются отношением типа 1 : M. Соответственно, связь между таблицами ПОКУПАТЕЛЬ и ДОГОВОР осуществляется по коду покупателя (КОД\_ПОК), который является уникальным идентификатором главного объекта ПОКУПАТЕЛЬ и неключевым в объекте ДОГОВОР (см. табл. 2.6).

Связь между таблицами ТОВАР и ПОСТАВКА\_ПЛАН осуществляется по уникальному ключу главного объекта ТОВАР — Коду товара, который в подчиненном объекте ПОСТАВКА\_ПЛАН является частью ключа (см. табл. 2.6). Аналогично связь между таблицами ТОВАР и ОТГРУЗКА осуществляется по уникальному ключу главного объекта ТОВАР — Коду товара.

Связь между таблицами ДОГОВОР и НАКЛАДНАЯ осуществляется по уникальному ключу главного объекта ДОГОВОР — Номеру договора (НОМ\_ДОГ), который в подчиненном объекте НАКЛАДНАЯ не входит в состав ключа (см. табл. 2.7).

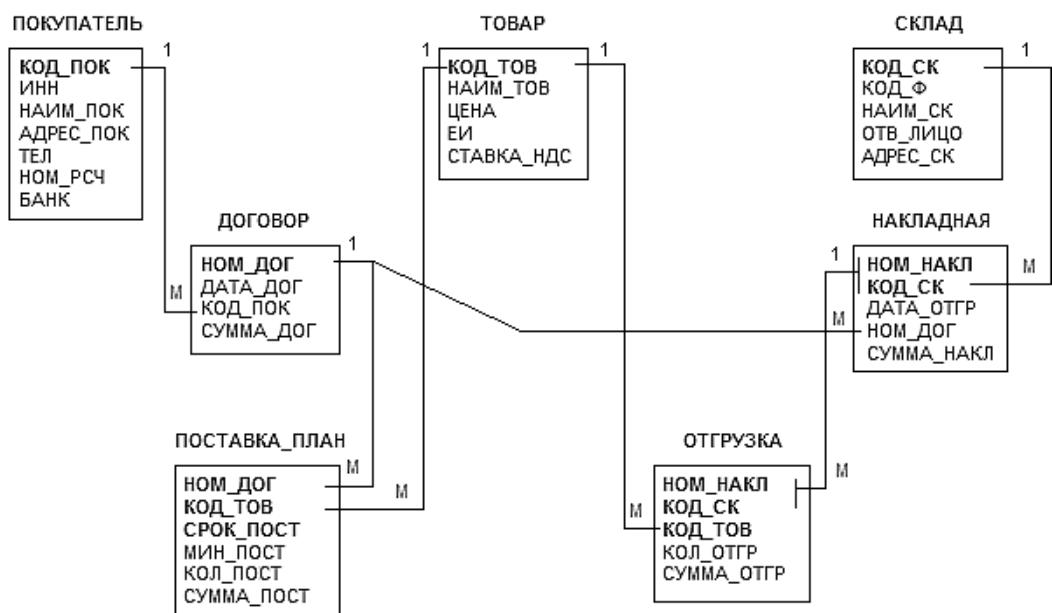


Рис. 2.19. Логическая структура реляционной базы данных предметной области "Поставка товаров"

Связь между таблицами ДОГОВОР и ПОСТАВКА\_ПЛАН осуществляется по уникальному ключу главного объекта ДОГОВОР — Номеру договора (ном\_дог), который в подчиненном объекте ПОСТАВКА\_ПЛАН является частью ключа (см. табл. 2.6). Аналогично связь между таблицами НАКЛАДНАЯ и ОТГРУЗКА осуществляется по уникальному составному ключу главного объекта НАКЛАДНАЯ — ном\_накл + + код\_ск, который в подчиненном объекте ОТГРУЗКА является частью ключа (см. табл. 2.7).

В Access может быть создана *схема данных*, наглядно отображающая логическую структуру базы данных. Определение одно-многозначных связей в этой схеме должно осуществляться в соответствии с построенной моделью данных. Топология проекта схемы данных практически совпадает с топологией информационно-логической модели. Для модели данных предметной области (см. рис. 2.18), построенной в рассмотренном примере, логическая структура базы данных в виде схемы данных Access приведена на рис. 2.19.

На этой схеме прямоугольники отображают таблицы базы данных с полным списком их полей, а связи показывают, по каким полям осуществляется взаимосвязь таблиц. Имена ключевых полей для наглядности выделены и находятся в верхней части полного списка полей каждой таблицы.

## Контрольные вопросы

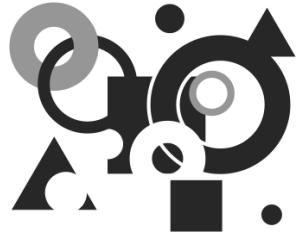
1. Что стоит за понятием сущность предметной области?
2. Что представляет экземпляр информационного объекта ТОВАР?
3. В каком случае описательный реквизит функционально полно зависит от ключа?
4. Как оказывается нормализация данных на свойствах базы?
5. Должен ли информационный объект, отвечающий требованиям нормализации, иметь уникальный ключ?
6. Могут ли описательные реквизиты, входящие в информационный объект, отвечающий требованиям нормализации, быть зависимыми друг от друга?
7. Может ли описательный реквизит, входящий в информационный объект, отвечающий требованиям нормализации, зависеть от ключа транзитивно?
8. Как устраняется транзитивная зависимость между реквизитами?
9. Верно ли утверждение, что документы являются основным источником данных внешинной сферы?
10. Перечислите формальные правила выделения информационных объектов.
11. Может ли реквизит одновременно быть ключевым для одних реквизитов и описательным для других?
12. Назовите виды отношений между двумя информационными объектами?
13. В каких отношениях находятся информационные объекты ДОГОВОР и НАКЛАДНАЯ в рассмотренной предметной области?
14. Сколько договоров соответствует одной накладной?
15. В каких отношениях находятся объекты ТОВАР и ОТГРУЗКА?

16. Какой объект является главным в отношении ДОГОВОР и ПОКУПАТЕЛЬ?
17. Как преобразовать много-многозначные отношения ТОВАР и СКЛАД к одноМногозначным?
18. Какой объект играет роль связки между объектами ДОГОВОР и ТОВАР, находящимися в отношении  $M : N$ ?
19. Какой объект играет роль связки между объектами ТОВАР и НАКЛАДНАЯ, находящимися в отношении  $M : N$ ?
20. Как образуется ключ в объекте "связка"?
21. По какому полю связываются таблицы, находящиеся в одно-многозначных отношениях?
22. Как можно определить уровень, на котором должен быть размещен объект в канонической модели данных?
23. Сколько таблиц будет в базе данных, построенной на основе нормализованной модели данных предметной области?
24. Может ли повторяться значение первичного ключа в нескольких строках реляционной таблицы?

## Ответы

1. Реальный объект, процесс, явление, событие.
2. Значения реквизитов, характеризующие конкретный товар.
3. Если одному значению ключа соответствует только одно значение описательного (зависимого) реквизита.
4. Устраняется дублирование описательных данных, обеспечивается их однократный ввод и корректировка, поддержание целостности средствами СУБД.
5. Да.
6. Нет.
7. Нет.
8. Расщеплением информационного объекта на два.
9. Да.
10. Составить перечень реквизитов документа; определить функциональные зависимости между ними; выбрать все зависимые реквизиты с указанием ключа, от которого они зависят функционально полно; объединить реквизиты, зависимые от одинаковых ключевых реквизитов вместе с ключом в одном объекте.
11. Да.
12. Одно-однозначные, одно-многозначные, много-многозначные.
13. Одно-многозначных.
14. Один.
15. Одно-многозначных.
16. ПОКУПАТЕЛЬ.

17. Через объект "связка" — запас товара на складе с составным ключом `КОД_ТОВ + КОД_СК`.
18. ПОСТАВКА\_ПЛАН.
19. ОТГРУЗКА.
20. Из ключей таблиц, находящихся в отношении  $M : N$ .
21. По уникальному ключу главной таблицы, в подчиненной таблице это поле не может быть ключом, хотя и может входить в составной ключ.
22. По числу связей в наиболее длинном пути от объектов нулевого уровня к данному объекту.
23. Столько, сколько информационных объектов в ИЛМ данных предметной области.
24. Нет.



## ГЛАВА 3

# Создание базы данных

Создание новой нормализованной реляционной базы данных Access осуществляется в соответствии с ее структурой, полученной в результате проектирования. Процесс проектирования реляционной базы данных был рассмотрен в предыдущей главе. Структура реляционной базы данных определяется составом таблиц и их взаимосвязями. Взаимосвязи между двумя таблицами реализуются через ключ связи, входящий в состав полей связываемых таблиц. Напомним, что в нормализованной реляционной базе данных таблицы находятся в отношениях типа "один-ко-многим" или "один-к-одному". Для одно-многозначных отношений в качестве ключа связи, как правило, используется уникальный ключ главной таблицы, в подчиненной таблице это может быть любое из полей, которое называется *внешним ключом*.

Создание реляционной базы данных начинается с формирования структуры таблиц. При этом определяется состав полей, их имена, тип данных каждого поля, размер поля, ключи, индексы таблицы и другие свойства полей. После определения структуры таблиц создается схема данных, в которой устанавливаются связи между таблицами. Access запоминает и использует эти связи при заполнении таблиц и обработке данных.

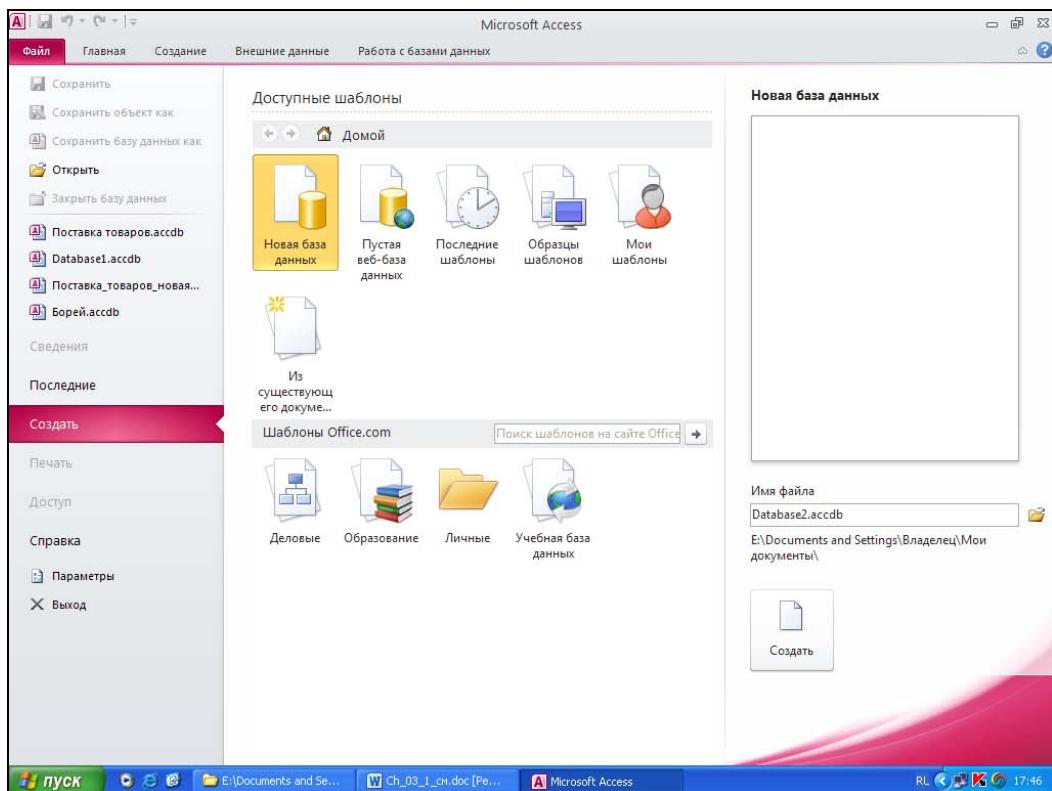
При создании базы данных важно задать параметры, в соответствии с которыми Access будет автоматически поддерживать целостность данных. Для этого при определении структуры таблиц должны быть заданы ключевые поля таблиц, указаны ограничения на допустимые значения данных, а при создании схемы данных на основе нормализованных таблиц должны быть заданы параметры поддержания целостности связей базы данных.

Завершается создание базы данных процедурой загрузки, т. е. заполнением таблиц конкретными данными. Особое значение имеет технология загрузки взаимосвязанных данных. Удобным инструментом загрузки данных во взаимосвязанные таблицы являются формы ввода/вывода, обеспечивающие интерактивный интерфейс для работы с данными базы. Формы позволяют создать экранный аналог документа источника, через который можно вводить данные во взаимосвязанные таблицы. Технология загрузки данных документов с использованием форм рассматривается в главе 5. В настоящей главе рассматривается непосредственный ввод данных в таблицы.

## ВНИМАНИЕ!

Пользователь может начинать работу с базой при любом количестве созданных таблиц еще до создания полной базы, отображающей все объекты модели данных предметной области. База данных может создаваться поэтапно, и в любой момент ее можно дополнить новыми таблицами и вводить связи между таблицами в схему данных, существующие таблицы могут дополняться новыми полями.

На открывшейся после запуска Access странице отображено представление Microsoft Office Backstage — набор команд на вкладке **Файл** (File) — которое предназначено для выполнения команд, применяемых ко всей базе данных. При этом на вкладке **Файл** (File) выбрана команда **Создать** (New). Область этой команды содержит **Доступные шаблоны** (Available Templates), позволяющие создать базу данных с использованием многочисленных шаблонов, размещенных на вашем компьютере или доступных через Интернет (рис. 3.1).



**Рис. 3.1.** Представление Microsoft Office Backstage при открытии Access

Шаблон — это готовая к использованию база данных, содержащая все таблицы, запросы, формы и отчеты, необходимые для выполнения определенной задачи. Эти готовые базы данных позволяют быстро создать приложения, ориентированные на поддержку широкого спектра задач. Непрерывно появляются новые шабло-

ны приложений, которые можно загрузить с веб-узла Microsoft Office Online. Эти стандартные приложения можно использовать без какой-либо модификации и настройки либо, взяв их за основу, приспособить шаблон к вашим нуждам и создать новые поля и таблицы, формы, отчеты и другие объекты базы данных.

Команда **Открыть** (Open) предназначена для открытия любой ранее созданной базы данных. Здесь же доступен список из 4 последних открываемых баз данных. Команда **Последние** (Recent) открывает более длинный список недавно открываемых баз данных. Щелчком на значке кнопки  можно добавить базу данных в список последних, а щелчком на значке  удалить из списка.

## Создание файла базы данных Access

Если для решения ваших задач использовать шаблон не имеет смысла, можно создать базу данных с нуля. Поскольку Access хранит все таблицы базы данных, а также другие объекты в одном файле, прежде чем приступить к созданию таблиц базы данных, необходимо создать файл пустой базы данных.

Для создания файла новой пустой базы данных щелкните в области создания базы данных стартового окна Access на элементе **Новая база данных** (Blank Database) (см. рис.3.1).

Выбор варианта **Новая база данных** (Blank database) или **Пустая веб-база данных** (Blank web database) определяет функции, доступные для работы с базой данных. Базы данных для настольных компьютеров нельзя опубликовать в Интернете, а веб-базы данных не поддерживают некоторые функции баз данных для настольных компьютеров, например итоговые запросы.

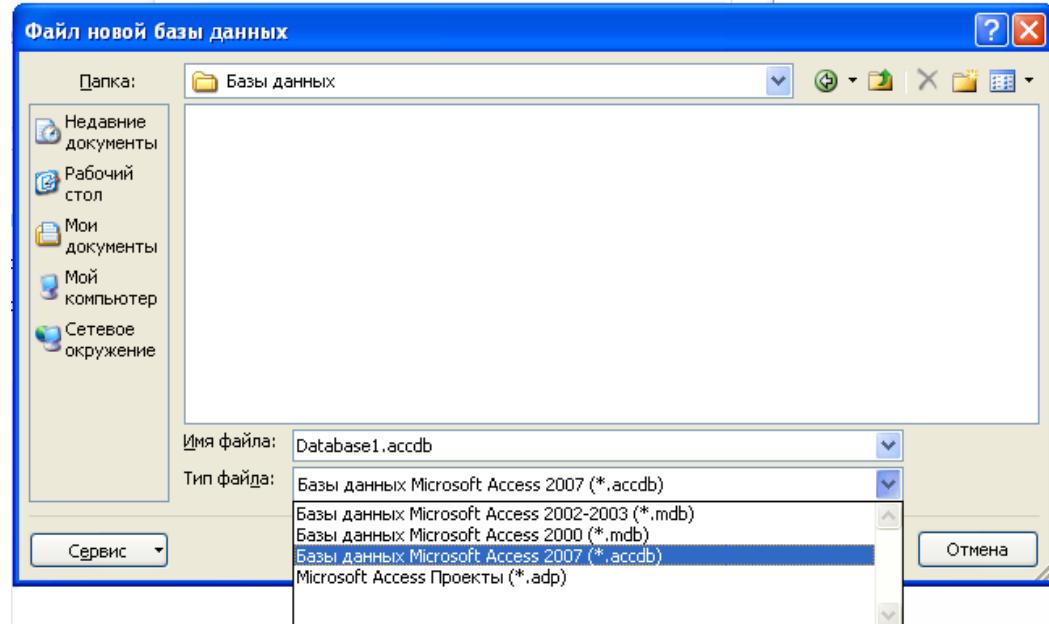
Справа, в области **Новая база данных** (Blank database), введите имя файла в поле **Имя файла** (File Name), например Поставка товаров.accdb.

Задавая имя файла базы данных, следует иметь в виду, что оно не должно содержать символов: \ / : \* ? " < > |, а его предельная длина составляет 215 символов, включая пробелы.

Под именем отображается имя папки, в которой по умолчанию сохранится файл базы данных. Если надо сохранить файл в другой папке, щелкните на значке  **Поиск расположения для размещения базы данных** (Browse for a location to put your database) (справа от имени файла базы данных) и в окне **Файл новой базы данных** (File New Database) (рис. 3.2) откройте нужную папку.

### ЗАМЕЧАНИЕ

Для изменения используемой по умолчанию папки для файлов новых баз данных **Мои документы** (My Documents) на вкладке **Файл** (File) выполните команду **Параметры** (Options) и в появившемся диалоговом окне на вкладке **Общие** (General) в разделе **Создание баз данных** (Creating databases) в поле **Рабочий каталог** (Default database folder), используя кнопку **Обзор** (Browse), выберите путь к папке, в которой предполагается хранить новые базы данных.



**Рис. 3.2.** Окно выбора местоположения, имени и формата файла новой базы данных

В окне **Файл новой базы данных** (File New Database) в поле **Тип файла** (Save as type) выберите формат создаваемой базы данных. По умолчанию формат файла имеет значение **Базы данных Microsoft Office Access 2007 (\*.accdb)** (Microsoft Access 2007 Databases). В Access 2010 сохранился формат базы данных Access 2007. Базы данных в формате Access 2007 сохраняются в файлах с расширением accdb. В предыдущих версиях базы данных сохранялись в файлах с расширением mdb.

### ЗАМЕЧАНИЕ

Для изменения формата выбираемого для новой базы данных по умолчанию на вкладке **Файл** (File) выполните команду **Параметры** (Options) и в появившемся диалоговом окне на вкладке **Общие** (General) в разделе **Создание баз данных** (Creating databases) в поле **Формат файла по умолчанию для пустой базы данных** (Default file format for Blank Database) выберите нужный формат.

Закончив выбор в окне **Файл новой базы данных** (File New Database), щелчком по кнопке **Создать** (Create) завершите процесс создания пустого файла новой базы данных. В результате открывается окно созданной базы данных с пустой таблицей с именем **Таблица1** (Table1) в режиме таблицы (см. рис. 3.3). Курсор находится в первой пустой ячейке столбца **Щелкните для добавления** (Click to Add). Теперь можно приступить к созданию этой таблицы и других объектов новой оригинальной базы данных.

При создании нового файла базы данных может быть выбран формат, предназначенный для работы с базой данных, размещенной на SQL-сервере. Последний фор-

мат называется Microsoft Access Проекты (\*.adp) (см. рис. 3.2). Проект предназначен для разработки объектов, составляющих приложение пользователя. Сама база данных, хотя и может разрабатываться в среде проекта, сохраняется на сервере и имеет соответствующий формат. Проекты сохраняются в файлах с расширением adp.

При создании проекта возможно подключение к существующей на сервере базе данных или создание новой базы данных на сервере.

## Окно Access

Создание новой пустой базы данных в формате Access 2007 приводит к открытию окна Access, в заголовке которого записано **<Имя БД> : база данных (Access 2007)** (рис. 3.3). **<Имя БД>** соответствует заданному в поле **Имя файла** (File Name).

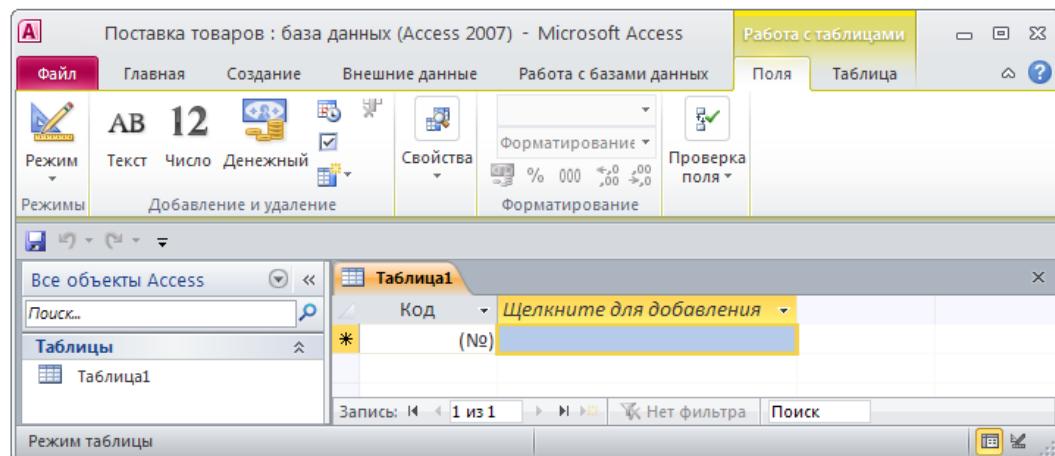


Рис. 3.3. Окно пустой базы данных

Три основных компонента пользовательского интерфейса Access 2010 формируют среду, в которой создаются и используются базы данных:

- ❖ **Лента** — полоса в верхней части окна приложения, содержащая группы команд выбранной вкладки;
- ❖ **Представление Backstage** — набор команд на вкладке **Файл** (File);
- ❖ **Область навигации** — область в левой части окна Access, предназначенная для работы с объектами базы данных. Область навигации появилась в Access 2007 и заменила окно базы данных предыдущих версий.

## Лента

Лента расположена в верхней части окна Access и отображает команды активной вкладки. Лента является основным элементом интерфейса в Access 2010, со-

держащим все инструменты для работы с объектами базы данных. На ленте находятся все команды, которые раньше были размещены в различных меню и панелях инструментов. Вкладки ленты объединяют логически связанные команды, ориентированные на задачу.

Четыре стандартные вкладки — **Главная** (Home), **Создание** (Create), **Внешние данные** (External Data) и **Работа с базами данных** (Database Tools) — пользователь видит на ленте при каждом запуске Access 2010. Они, как правило, отображаются в окне Access и не зависят от выбранного объекта базы данных и режима работы с ним. Команды на каждой вкладке организованы в несколько групп, соответствующих подзадачам вкладки. Некоторые команды могут открывать другие элементы интерфейса, например коллекции или диалоговые окна.

Остальные виды вкладок ленты являются контекстно-зависимыми, соответствуют текущему активному объекту и режиму работы. Набор контекстных вкладок автоматически появляется в интерфейсе наряду со стандартными вкладками по мере необходимости. Названия соответствующих контекстных вкладок подчеркнуты цветом. Контекстные вкладки предоставляют команды для работы с выбранным объектом.

В ряде случаев стандартный набор вкладок при переключении на конкретный вид или режим работы заменяется так называемыми программными вкладками, например при предварительном просмотре отчета.

## Представление Backstage

Представление Backstage (см. рис. 3.1) появилось в Access 2010. Оно пришло на смену традиционному меню **Файл** (File) и теперь открывается щелчком на вкладке **Файл** (File). Представление Backstage содержит команды и сведения, как правило, применимые ко всей базе данных, например **Сжать и восстановить** (Compact & Repair), **Опубликовать в Access Services** (Publish to Access Services), **Зашифровать паролем** (Encrypt with Password), а также команды, которые в более ранних версиях содержались в меню **Файл** (File), например **Печать** (Print), **Сохранить** (Save), **Параметры** (Options). Для возврата к документу достаточно щелчка на отображаемом в правой части представления актуальном окне Access на вкладке **Сведения** (Info).

## Панель быстрого доступа

Панель быстрого доступа — небольшая панель инструментов, на которую можно добавить самые нужные команды. Она может размещаться под лентой или над ней и обеспечивает доступ к командам одним нажатием кнопки. Первоначально панель содержит лишь команды **Сохранить** (Save), **Отменить** (Undo), **Вернуть** (Redo) и **Настройка панели быстрого доступа** (Customize Quick Access Toolbar) . Этую панель легко дополнить любыми командами и даже целыми группами. Для на-

строки панели быстрого доступа можно использовать различные способы. Например, можно воспользоваться соответствующей кнопкой на панели быстрого доступа и выбрать некоторые команды из ее меню. При выполнении **Другие команды** (More Commands) открывается окно **Параметры Access** (Access Options) с отображенной вкладкой **Панель быстрого доступа** (Quick Access Toolbar), на которой можно добавить или удалить любую команду с панели быстрого доступа. Кроме того, можно через контекстное меню любой команды на ленте добавить ее на панель быстрого доступа. В контекстном меню команды на панели быстрого доступа можно выполнить **Удалить с панели быстрого доступа** (Remove from Quick Access Toolbar).

## Область навигации

Появившаяся в Access 2007 область навигации (рис. 3.4) используется для отображения объектов открытой базы данных и выполнения работ с ними. Действия, которые можно выполнить с объектом, представлены как командами ленты, так и командами его контекстного меню.

Команды контекстного меню позволяют открывать объект в нужном режиме, удалять, переименовывать, экспорттировать, просматривать свойства и выполнять некоторые другие действия. Область навигации появилась в Access 2007 и заменила окно базы данных, которое использовалось в предшествующих версиях Access.

Все объекты базы данных в области навигации делятся на категории, которые, в свою очередь, содержат группы.

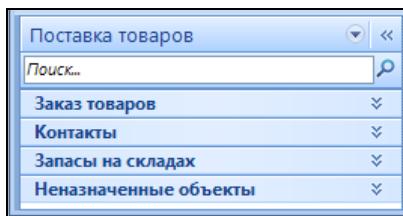


Рис. 3.4. Область навигации со встроенными группами объектов

На приведенном рис. 3.4 показаны группы встроенной категории **Тип объекта** (Object Type), для которой выбран фильтр **Все объекты Access** (All Access Objects). Эта категория группирует компоненты базы данных по их типам — таблицы помещаются в группу **Таблицы** (Tables), формы — в группу **Формы** (Forms) и т. д. Такое представление объектов базы данных в области навигации практически точно соответствует представлению объектов в окне базы данных более ранних версий Access.

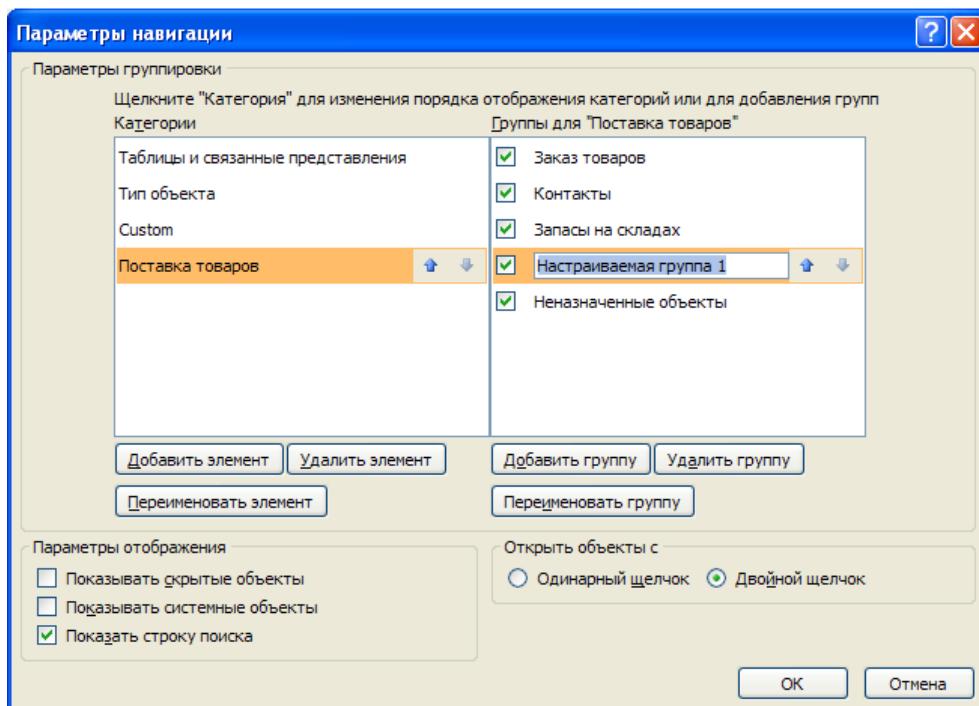
Область навигации можно использовать для упорядочения объектов по типу, дате создания, дате изменения, связанной таблице (на основе зависимостей объектов), а также в пользовательских категориях и группах.

Пользовательские категории и группы объектов базы данных (рис. 3.5) позволяют объединить объекты, например по функциональному назначению. Это облегчает пользователю работу с базой данных и может быть использовано вместо ранее используемых кнопочных форм.



**Рис. 3.5.** Область навигации с пользовательской категорией **Поставка товаров** и группами объектов

Включить в область навигации новую категорию можно в окне **Параметры навигации** (Navigation Options) (рис. 3.6), открывающемся соответствующей командой контекстного меню при щелчке правой кнопкой мыши в пустом пространстве нижней части области навигации. Для добавления категории воспользуйтесь кнопкой **Добавить элемент** (Add Item). Соответствующие кнопки позволят переименовать или удалить категорию. В этом же окне можно добавить, переименовать или удалить группу из категории.



**Рис. 3.6.** Окно **Параметры навигации** с пользовательской категорией **Поставка товаров** и ее группами

После закрытия окна **Параметры навигации** (Navigation Options) для добавления объектов в группу следует в области переходов перейти в созданную категорию, в группе **Неназначенные объекты** (Unassigned Objects) выбрать нужный объект и по контекстному меню включить его в нужную группу. Если группа не была создана к этому моменту, можно создать ее командой этого же контекстного меню.

Элемент области переходов, **Поиск** (Search), позволяет произвести быстрый поиск объектов в больших базах данных. По мере ввода текста группы, которые не содержат объектов, удовлетворяющих условию поиска, сворачиваются.

Чтобы прекратить поиск и восстановить свернутые группы, следует нажать кнопку **Очистить строку поиска** (Clear Search String) , расположенную справа от строки **Поиск** (Search).

Для отображения поля **Поиск** (Search) щелкните правой кнопкой мыши в пустом пространстве области переходов и в контекстном меню выберите пункт **Строка поиска** (Search Bar).

Объекты в каждой группе могут отображаться с разной степенью подробностей. Например, выбор в контекстном меню группы команды **Просмотр | Сведения** (View By | Details) позволяет отобразить для каждого объекта группы дату его создания и изменения.

Область переходов нельзя скрыть за другими окнами, но можно свернуть, воспользовавшись кнопкой **Открыть/закрыть границу области переходов** (Shutter Bar Open/Close Button) , размещенной в ее правом верхнем углу.

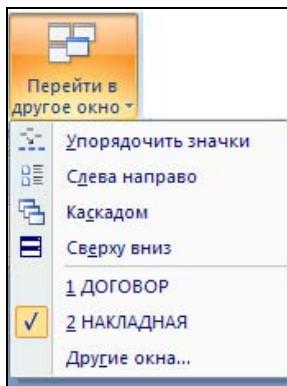
## Область документов

Каждый открываемый в любом режиме объект базы данных отображается в области документов. Предусмотрено два вида представления объекта в области документов: **Перекрывание окон** (Overlapping Windows) и **Вкладки** (Tabbed Documents).

Перекрывающиеся окна объектов могут занимать всю область документов, принимать любые размеры или сворачиваться до размеров заголовков, отображаемых внизу области документов. Для перехода к другому документу достаточно щелчка мыши на нем. Если не все документы видны в окне, можно использовать команду **Перейти в другое окно** (Switch Windows) (рис. 3.7), размещенную в группе **Окно** (Window) на вкладке ленты **Главная** (Home).

Начиная с Access 2007, открытые объекты базы данных могут размещаться на различных вкладках области документов (рис. 3.8). Вкладки используются для переключения между открытыми объектами. При использовании вкладок каждый объект занимает всю область документов и исключается возможность одновременного просмотра ряда объектов. Команда **Перейти в другое окно** (Switch Windows) делается недоступной.

Переход из режима использования вкладок документов к режиму перекрытия окон и обратно осуществляется путем настройки параметров Access.



**Рис. 3.7.** Список команд кнопки **Перейти в другое окно** и открытых документов

Номер договора	Дата заключения	Код покуп	Сумма по договор	Добавить поле
Д111	01.01.2004	П001	1 135 400,00р.	
Д222	29.02.2004	П001	778 024,00р.	
Д333	01.01.2004	П002	75 160,00р.	
Д555	12.01.2003	П002	12 000,00р.	
Д777	14.06.2004	П001	250 000,00р.	
Д888	23.05.2004	П002	30 000,00р.	
Д999	12.09.2004	П002	450 000,00р.	
*			0,00р.	

**Рис. 3.8.** Пример набора вкладок документов. Активна вкладка с таблицей ДОГОВОР

Щелкните на вкладке **Файл** (File), а затем в открывшемся представлении Backstage выполните команду **Параметры** (Options). В левой области диалогового окна **Параметры Access** выберите **Текущая база данных** (Current Database). В разделе **Параметры приложения** (Application Options) в группе **Параметры окна документа** (Document Window Options) выберите **Перекрывание окон** (Overlapping Windows) или **Вкладки** (Tabbed Documents).

Чтобы изменения этих параметров вступили в силу, необходимо закрыть и снова открыть базу данных.

### **ЗАМЕЧАНИЕ**

Параметр вкладки документов относится к конкретной базе данных и должен устанавливаться отдельно для каждой базы данных. В новых базах данных, созданных в формате Access 2007, вкладки документов отображаются по умолчанию. В базах данных, созданных в предыдущих версиях Access, по умолчанию используются перекрывающиеся окна.

## **Строка состояния**

Вдоль нижней границы окна Access размещена строка состояния, отображающая кнопки, с помощью которых удобно переключаться в различные режимы работы с активным объектом.

Если в окне не отображается строка состояния, откройте окно параметров Access. В его левой области выберите **Текущая база данных** (Current Database). В разделе **Параметры приложения** (Application Options) установите флажок **Строка состояния** (Status Bar).

## Создание таблицы базы данных

Как уже говорилось ранее, таблица содержит данные об одном информационном объекте или сущности, например сведения о покупателях или товарах. Каждая запись (строка) в таблице включает данные об одном экземпляре сущности, например о конкретном покупателе. Запись состоит из полей (столбцов), каждое из которых включает такие атрибуты сущности, как наименование, ИНН, адрес и телефон. Для однозначной идентификации записи в таблице определяется уникальный простой или составной ключ. База данных может содержать множество таблиц. Разработка базы данных начинается с создания таблиц.

Создание таблицы БД состоит из двух этапов. На первом этапе определяется ее структура: состав полей, их имена, тип данных каждого поля, размер поля, ключи, индексы таблицы и другие свойства полей. На втором этапе производится создание записей таблицы путем заполнения ее полей данными.

В Access создание таблицы может быть выполнено в одном из двух режимов:

- ❖ в режиме конструктора (Design View), позволяющем максимально полно определить структуру таблицы;
- ❖ в режиме таблицы (Datasheet View), предназначенному, прежде всего, для создания, просмотра, поиска, корректировки ее записей и, кроме того, реализующем функции, обеспечивающие определение структуры таблицы.

Рассмотрим основные параметры полей таблицы, которые могут быть заданы вне зависимости от способа создания таблицы.

### Имена полей и тип данных

Для определения поля таблицы обязательно задаются **Имя поля** (Field Name) и **Тип данных** (Data Type).

**Имя поля** (Field Name). Каждое поле в таблице должно иметь уникальное имя, удовлетворяющее соглашениям об именах объектов в Access. Оно является комбинацией из букв, цифр, пробелов и специальных символов, за исключением точки (.), восклицательного знака (!), надстрочного знака (^) и квадратных скобок ([ ]). Имя не может начинаться с пробела и содержать управляющие символы с кодами ASCII от 0 до 31. Максимальная длина имени 64 символа.

**Тип данных** (Data Type). Тип данных определяется значениями, которые предполагается хранить в поле, и операциями, которые будут выполняться с этими значениями. В Access допускается использование двенадцати типов данных.

Рассмотрим вкратце назначение и допустимые размеры данных всех типов, которые могут назначаться полям таблицы в Access.

- ◆ **Текстовый** (Text) — используется для хранения текста или комбинаций алфавитно-цифровых знаков, не применяемых в расчетах (например, код товара). Максимальная длина поля 255 знаков.
- ◆ **Поле МЕМО** (Memo) — используется для хранения обычного текста или комбинаций алфавитно-цифровых знаков длиной более 255 знаков. Поля с этим типом данных в базах данных формата Access 2007 поддерживают также форматирование текста. Это единственный в Access тип данных, обеспечивающий встроенную поддержку отображения и хранения форматированного текста. Максимальный размер поля 1 Гбайт знаков или 2 Гбайт памяти (2 байта на знак) при программном заполнении полей, и 65 535 знаков при вводе данных вручную в поле и в любой элемент управления, связанный с этим полем.
- ◆ **Числовой** (Number) — служит для хранения числовых значений (целых или дробных), предназначенных для вычислений, исключением являются денежные значения, для которых используется тип данных **Денежный** (Currency). Размер поля 1, 2, 4 и 8 байтов, или 16 байтов (если используется для кода репликации) зависит от типа чисел, вводимых в поле.
- ◆ **Дата/время** (Date/Time) — используется для хранения значений даты и времени в виде 8-байтовых чисел двойной точности с плавающей запятой. Целая часть значения, расположенная слева от десятичной запятой, представляет собой дату. Дробная часть, расположенная справа от десятичной запятой, — это время. Хранение значений даты и времени в числовом формате позволяет выполнять различные вычисления с этими данными.
- ◆ **Денежный** (Currency) — используется для хранения денежных значений в виде 8-байтовых чисел с точностью до четырех знаков после запятой. Этот тип данных применяется для хранения финансовых данных и в тех случаях, когда значения не должны округляться.
- ◆ **Счетчик** (AutoNumber) — используется для уникальных числовых 4-байтовых значений, которые автоматически вводит Access при добавлении записи. Вводимые числа могут последовательно увеличиваться на указанное приращение или выбираться случайно. Обычно используются в первичных ключах.
- ◆ **Логический** (Yes/No) — применяется для хранения логических значений, которые могут содержать одно из двух значений: Да/Нет, Истина/Ложь или Вкл/Выкл. (8 битов = 1 байт). Используется **1** для значений Да и **0** для значений Нет. Размер равен 1 биту.
- ◆ **Поле объекта OLE** (OLE Object) — используется для хранения изображений, документов, диаграмм и других объектов из приложений MS Office и других программ Windows в виде растровых изображений, которые затем отображаются в элементах управления форм или отчетов, связанных с этим полем таблицы. Чтобы в Access просматривать эти изображения, необходимо, чтобы на компьютере, использующем базу данных, был зарегистрирован OLE-сервер (программа, поддерживающая этот тип файлов). Если для данного типа файлов

OLE-сервер не зарегистрирован, отображается значок поврежденного изображения.

- ❖ **Гиперссылка** (Hyperlink) — применяется для хранения ссылок на Web-узлы (URL-адреса), на узлы или файлы интрасети или локальной сети (UNC-адреса — стандартного формата записи пути), а также на узлы или файлы локального компьютера. Кроме того, можно использовать ссылку на объекты Access, хранящиеся в базе данных. Может хранить до 1 Гбайт данных.
- ❖ **Вложение** (Attachment) — используется для вложения в поле записи файлов изображений, электронных таблиц, документов, диаграмм и других файлов поддерживаемых типов точно так же, как в сообщения электронной почты. Вложенные файлы можно просматривать и редактировать в соответствии с заданными для поля параметрами. Эти поля не имеют ограничений, связанных с отсутствием зарегистрированных OLE-серверов. Более рационально используют место для хранения, чем поля с типом данных **Поле объекта OLE** (OLE Object), поскольку не создают растровые изображения исходного файла. Максимальная длина поля для сжатых вложений — 2 Гбайт, для несжатых — примерно 700 Кбайт в зависимости от степени возможного сжатия вложения.
- ❖ **Вычисляемый** (Calculated) — предназначен для создания вычисляемых полей: числовых, текстовых, денежных, дата/время, логических. Значение вычисляемого поля определяется выражением, записанным в поле и использующим другие поля текущей записи, некоторые встроенные функции и константы, связанные арифметическими, логическими или строковыми операторами.
- ❖ **Мастер подстановок** (Lookup Wizard) или **Подстановка и отношения** (Lookup & Relationship) — вызывает мастера подстановок, с помощью которого можно создать поле, позволяющее выбрать значения из списка, построенного на основе значений поля другой таблицы, запроса или фиксированного набора значений. Такое поле отображается как поле со списком. Если список построен на основе поля таблицы или запроса, тип данных и размер создаваемого поля определяется типом данных и размером привязанного столбца; если на основе набора значений — размером текстового поля, содержащего значение. Кроме того, мастер подстановок позволяет определить связь таблиц и включить проверку связной целостности данных.

## Общие свойства поля

Основные свойства задаются для каждого поля и зависят от выбранного типа данных. Если открыть таблицу в режиме конструктора, то весь набор свойств выбранного поля будет представлен в нижней части окна на двух вкладках: **Общие** (General) и **Подстановка** (Lookup). Приведем свойства полей, наиболее важные на первом этапе изучения баз данных.

- ❖ **Размер поля** (Field Size) позволяет для текстового и числового поля уточнить тип данных или размер, задает максимальный размер данных, сохраняемых в поле. Для поля с типом данных **Текстовый** (Text) задается размер от 1 до 255 знаков.

Для поля с типом данных **Числовой** (Number) можно задать:

- ◆ **Байт** (Byte) для целых чисел от 0 до 255, длина поля 1 байт;
- ◆ **Целое** (Integer) для целых чисел от -32 768 до +32 767, занимает 2 байта;
- ◆ **Длинное целое** (Long Integer) для целых чисел от -2 147 483 648 до +2 147 483 647, занимает 4 байта;
- ◆ **Одинарное с плавающей точкой** (Single) для чисел от  $-3,4 \times 10^{38}$  до  $+3,4 \times 10^{38}$  с точностью до 7 знаков, занимает 4 байта;
- ◆ **Двойное с плавающей точкой** (Double) для чисел от  $-1,797 \times 10^{308}$  до  $+1,797 \times 10^{308}$  с числом отображаемых десятичных знаков до 15, занимает 8 байтов;
- ◆ **Действительное** (Decimal) для целых чисел от  $-10^{38}$  до  $+10^{38}$  (при работе с проектами, которые хранятся в файлах типа adp) и от  $-10^{28}$  до  $10^{28}$  (mdb и accdb) с числом отображаемых десятичных знаков до 28, занимает 12 байтов;
- ◆ **Код репликации** (Replication ID). Глобальный уникальный идентификатор (Globally unique identifier, GUID), занимает 16 байтов. Эти длинные генерируемые случайным образом значения обеспечивают малую вероятность их совпадения. Поля такого типа используются Access для создания системных уникальных идентификаторов реплик, наборов реплик, таблиц, записей и других объектов при репликации баз данных. Могут быть использованы в приложениях пользователя для идентификации строк таблицы, например для идентификации товаров.

Для поля с типом данных **Счетчик** (AutoNumber) можно задать:

- ◆ **Длинное целое** (Long Integer) — 4 байта;
- ◆ **Код репликации** (Replication ID) — 16 байтов.

Рекомендуется задавать минимально допустимый размер поля, который понадобится для сохраняемых значений, т. к. сохранение таких полей требует меньше памяти, и обработка данных меньшего размера выполняется быстрее.

Изменения в данных, которые происходят вследствие изменения свойства **Размер поля** (Field Size), нельзя отменить после выполнения сохранения.

- ◆ **Формат поля** (Format) является форматом отображения выбранного типа данных при выводе их на экран или печать в режиме таблицы, в форме или отчете. В Access определены встроенные стандартные форматы отображения для полей с такими типами данных как **Числовой** (Number), **Дата/время** (Date/Time), **Логический** (Yes/No) и **Денежный** (Currency). Ряд этих форматов совпадает с настройкой региональных форматов, определяемых в окне **Язык и региональные стандарты** в Панели управления Windows. Пользователь может создать собственный формат для всех типов данных, кроме **Поле объекта OLE** (OLE Object), с помощью символов форматирования.
- ◆ **Число десятичных знаков** (Decimal Places) задает для числового и денежного типов данных количество знаков после запятой. Можно задать число от 0 до 15. По умолчанию (значение **Авто** (Auto)) это число определяется установкой в свойстве **Формат поля** (Format). Следует иметь в виду, что установка этого свойства не действует, если свойство **Формат поля** (Format) не установлено

или выбрано значение **Основной** (General Namber). Свойство **Число десятичных знаков** (Decimal Places) влияет только на количество десятичных знаков, отображаемых на экране, и не влияет на число сохраняемых десятичных знаков. Для изменения числа сохраняемых знаков нужно изменить свойство **Размер поля** (Field Size).

- ❖ **Подпись** (Caption) поля задает текст, который выводится в таблицах, формах, отчетах.
- ❖ **Описание** (Description) — краткий пользовательский комментарий к полю.
- ❖ **Значение по умолчанию** (Default Value) определяет текст или выражение, значение которого автоматически вводится в поле при создании новой записи. Например, если задана функция `=Now()`, то в поле введется текущая дата и время. При добавлении записи в таблицу можно оставить значение, введенное по умолчанию, или ввести другое. Свойство **Значение по умолчанию** (Default Value) используется только при создании новой записи. Максимальная длина значения свойства составляет 255 знаков. Свойство не определено для полей с типом данных **Счетчик** (AutoNumber) или **Поле объекта OLE** (OLE Object).
- ❖ **Условие на значение** (Validation Rule) позволяет осуществлять контроль ввода, задает ограничения на вводимые значения, при нарушении условий запрещает ввод и выводит текст, заданный свойством **Сообщение об ошибке** (Validation Text).
- ❖ **Сообщение об ошибке** (Validation Text) задает текст сообщения, выводимый на экран при нарушении ограничений, заданных свойством **Условие на значение** (Validation Rule).

### **ВНИМАНИЕ!**

Элементы управления, созданные в формах или отчетах на основе поля таблицы, наследуют установленные для этого поля свойства. Благодаря этому не понадобится определять свойства индивидуально для каждого связанного с полем элемента управления.

### **Свойства вкладки Подстановка**

В окне конструктора таблиц на вкладке **Подстановка** (Lookup) задается свойство **Тип элемента управления** (Display Control). Это свойство определяет, будет ли отображаться поле в таблице и в элементе управления формы в виде **Поле** (Text Box), **Список** (List Box) или **Поле со списком** (Combo Box).

Если для поля выбран тип элемента управления **Список** (List Box) или **Поле со списком** (Combo Box), на вкладке **Подстановка** (Lookup) появляются дополнительные свойства, которые определяют источник данных для строк списка и ряд других характеристик списка. Источник данных определяет, откуда брать значения для столбца подстановок. В качестве источника данных для поля со списком выбирается таблица или запрос, с которым осуществляется постоянная связь, что обеспечивает актуальное состояние списка, для списка строится список конкретных значений, разделенных точкой с запятой. Следует заметить, что не все типы данных позволяют использовать поле со списком.

## Определение первичного ключа

Каждая таблица в реляционной базе данных должна иметь уникальный (первичный) ключ, однозначно определяющий каждую запись в таблице. Это позволяет быстро найти нужную запись. Кроме того, это поле используется для связи данных из разных таблиц в запросах, формах и отчетах. Ключевое поле должно содержать уникальные значения, такие как коды или инвентарные номера, и не может содержать значения **Null**. Если для таблицы определен первичный ключ, то Access предотвращает дублирование ключа и отсутствие значения в его полях. Ключ может быть простым или составным, включающим несколько полей (до 10).

Для задания первичного ключа таблицы, его изменения или удаления необходимо использовать режим конструктора.

Для определения ключа выделяются поля, составляющие ключ, и на ленте **Работа с таблицами | Конструктор** (Table Tools | Design) в группе **Сервис** (Tools) нажимается кнопка **Ключевое поле** (Primary Key) или выполняется команда контекстного меню поля **Ключевое поле** (Primary Key).

Для ключевого поля автоматически строится уникальный индекс с именем **PrimaryKey**, ускоряющий поиск нужной записи по значению ключевого поля. В этом можно убедиться, просмотрев информацию об индексах таблицы. Индекс первичного ключа всегда уникален и не допускает пустых полей в записях. Окно **Индексы** (Indexes) (рис. 3.16) вызывается щелчком на кнопке просмотра и редактирования индексов **Индексы** (Indexes) на ленте **Работа с таблицами | Конструктор** (Table Tools | Design).

### ЗАМЕЧАНИЕ

Индексы предназначены для осуществления быстрого поиска требуемых записей в больших таблицах базы данных по значению первичного или вторичного ключа. Индексы можно представить как некоторые внутренние служебные таблицы, содержащие два столбца: первый содержит значение индексируемого поля, а второй — адреса записей, имеющих это значение в индексируемом поле. В индексной таблице производится упорядочение строк по значениям индексируемого поля, и это позволяет использовать методы быстрого поиска строки с заданным значением индексного поля, вместо последовательного просмотра строк таблицы. По адресу, содержащемуся в найденной строке индексной таблицы, осуществляется прямой доступ к искомой записи данных. Допускается не более 32 индексов на таблицу.

В качестве первичного ключа может быть задано поле с типом данных **Счетчик** (AutoNumber). В этом случае при добавлении каждой новой записи в таблицу в это поле автоматически вводятся уникальные целые, последовательно возрастающие или случайные числа. Указание такого поля является наиболее простым способом создания первичного ключа. Значения этого поля нельзя изменить или удалить. Длина поля 4 байта для длинного целого, для кода репликации — 16 байтов. По умолчанию в поле вводятся последовательные значения. В таблице не может быть более одного поля этого типа. Если первичный ключ не установлен пользователем до сохранения вновь созданной таблицы, Access спросит о необходимости

создания первичного ключа. При утвердительном ответе Access создаст первичный ключ с типом данных **Счетчик** (AutoNumber).

## Создание таблицы в режиме таблицы

Рассмотрим, прежде всего, создание таблицы в режиме таблицы, т. к. этот режим значительно проще для неопытного пользователя, а новая версия Access предоставляет в этом режиме такие возможности разработки структуры таблицы, что они практически мало отличаются от доступных в режиме конструирования.

В процессе создания новой базы данных в ней автоматически создается новая пустая таблица с именем **Таблица1** (Table1), которая открывается в области документов в режиме таблицы (рис. 3.9).

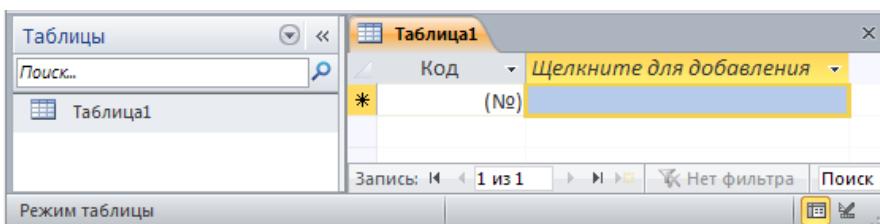


Рис. 3.9. Новая пустая таблица

Одновременно открывается лента **Работа с таблицами | Поля** (Table Tools | Fields), обеспечивающая определение полей таблицы с различными свойствами. На рис. 3.10 представлена эта лента с открытым списком **Другие поля** (More Fields), включающая команды для добавления и удаления полей таблицы с различными типами данных. В конце этого списка имеется раздел **Краткое руководство** (Quick Start), в котором содержатся часто используемые поля, такие как адрес, телефон, категория. Некоторые из них выполняют добавление поля со списком значений. При открытии списка в поле предоставляется возможность изменить его элементы. Таким образом, пользователь может сразу приступить к разработке первой таблицы базы данных в режиме таблицы, дополняя ее полями с помощью разнообразных и удобных средств.

Для создания и открытия следующей таблицы в аналогичном режиме предназначена команда **Таблица** (Table), размещенная на вкладке ленты **Создание** (Create) в группе **Таблицы** (Tables).

В предлагаемой системой таблице (см. рис.3.9) определено ключевое поле с типом данных **Счетчик** (AutoNumber), и в нее в режиме таблицы можно добавлять новые поля, наделенные рядом характеристик. Столбец **Щелкните для добавления** (Click to Add) постоянно отображается в режиме таблицы, за исключением случая, когда в таблице не определен первичный ключ.

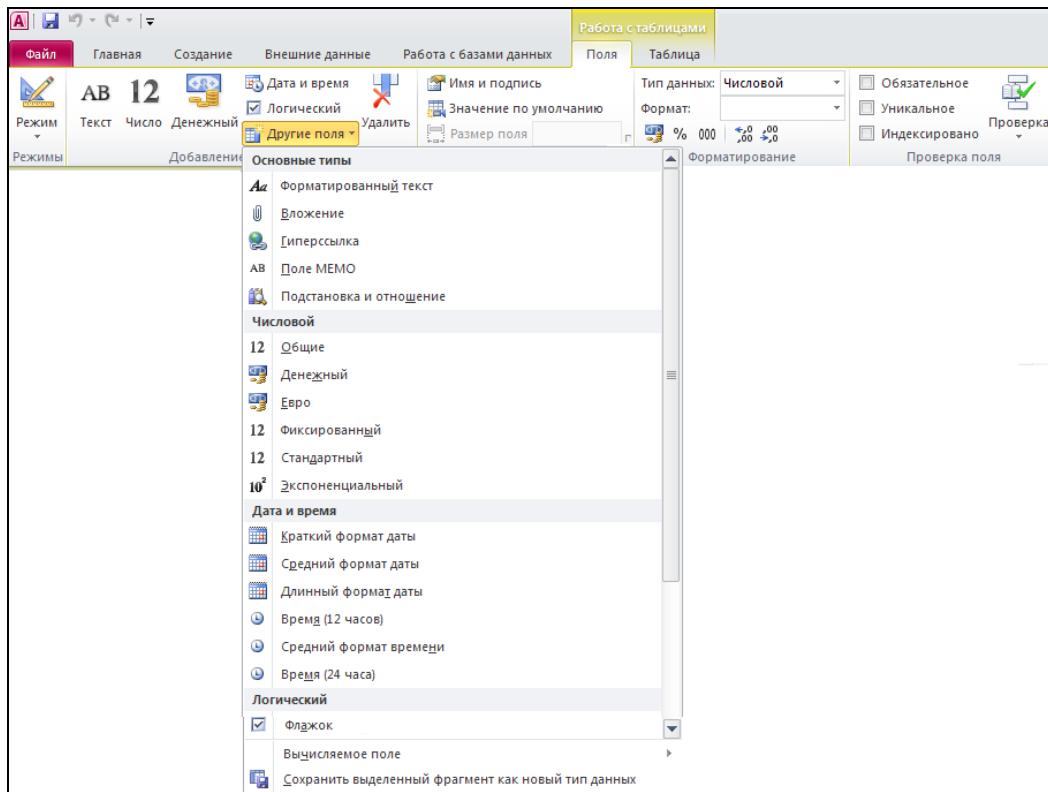


Рис. 3.10. Лента режима Работа с таблицами | Поля

Как уже было сказано ранее, для каждого поля обязательно задается имя, однозначно определяющее это поле в таблице, и тип данных, соответствующий сохраняемым в этом поле данным. Тип данных определяет значения, которые можно сохранить в поле, операции, которые можно выполнить с данными, а также выделяемый объем памяти. С каждым полем также связывается группа параметров, называемых *свойствами*, которые определяют функциональные характеристики и внешний вид этого поля. В Access 2010 в режиме таблицы для поля может быть задано большинство свойств.

## Добавление полей

В режиме таблицы определение полей можно выполнить как выбором необходимых команд, так и простым вводом данных в столбце **Щелкните для добавления** (Click to Add) (см. рис. 3.9). При вводе первого значения Access автоматически определит тип данных (например, дата, число или текст) и добавит новое поле в таблицу. При добавлении полей этим способом Access автоматически назначит им имена: **Поле1** (Field1) для первого, **Поле2** (Field2) для второго и т. д. Чтобы дать полям содержательные имена, измените их. Для этого выполните двойной щелчок

на заголовке поля и введите новое имя или, использовав правую кнопку мыши, в контекстном меню выберите команду **Переименовать столбец** (Rename Column). Имена полей могут содержать до 64 знаков (цифр или букв), включая пробелы.

Access распознает и автоматически определяет следующие типы данных:

- ◆ для текстовых значений — **Текстовый** (Text);
- ◆ для целых числовых значений — **Числовой** (Namber), **Длинное целое** (Long Integer);
- ◆ для числовых значений типа 45,76 или 34,25% или 12% — **Числовой** (Namber), **Двойное с плавающей точкой** (Double). Распознаваемый формат чисел зависит от настройки региональных параметров на вашем компьютере. Так, если в качестве разделителя целой и дробной частей числа выбрана запятая, а введена точка, полю может быть назначен другой тип данных, например **Дата/время** (Date/Time);
- ◆ для гиперссылок — **Гиперссылка** (Hyperlink). Допускается использование любого префикса протокола Интернета, например http://, mailto:;;
- ◆ для даты и времени — **Дата/время** (Date/Time). Распознаваемый формат даты и времени зависит от настройки региональных параметров на вашем компьютере;
- ◆ для денежных значений типа 12,50 р. — **Денежный** (Currency}. Распознаваемый знак валюты зависит от настройки региональных параметров на вашем компьютере.

Если на основании введенных данных Access не может точно определить тип данных, задается тип данных **Текстовый** (Text).

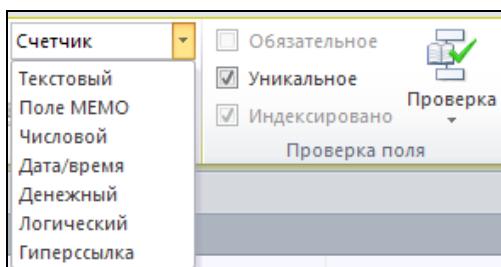
Access, наряду с определением типа данных, может автоматически задавать значение свойства **Формат** (Format) в зависимости от вида вводимых данных. Это свойство определяет вид отображения и печати данных поля. Например, если ввести 01-янв-2010, будет задан тип данных **Дата/время** (Date/Time) и в свойстве **Формат** (Format) значение **Средний формат даты** (Medium Date). К полю с типом данных **Дата/время** (Date/Time) автоматически привязывается календарь для выбора даты.

После определения структуры таблицы таким простым способом ее всегда можно доработать, воспользовавшись командами ленты **Работа с таблицами | Поля** (Table Tools | Fields) в группах **Свойства** (Properties), **Форматирование** (Formatting) и **Проверка поля** (Field Validation) (рис. 3.11). При этом будут доступны почти все используемые Access типы данных (рис. 3.12).

Пусть необходимо создать таблицу ПОКУПАТЕЛЬ, структура которой была определена в главе 2. В новой пустой таблице (см. рис. 3.9) по умолчанию определено поле первичного ключа таблицы с именем **Код** (ID) и типом данных **Счетчик** (AutoNumber). Таблица ПОКУПАТЕЛЬ содержит два уникальных поля код покупателя и ИНН, поэтому нецелесообразно вводить в нее дополнительное ключевое поле. Чтобы изменить имя этого поля, выполните двойной щелчок на заголовке поля и исправьте имя **Код** (ID) на **КОД\_ПОК**. Чтобы изменить тип сохраняемых в поле данных на вкладке ленты **Работа с таблицами | Поля** (Table Tools | Fields) в

группе **Форматирование** (Formatting), откройте список с типами данных и выберите **Текстовый** (Text) (см. рис. 3.11). Для работы с ключевыми полями режим таблицы предоставляет ограниченные возможности. В этом режиме нельзя удалить ключ и нельзя определить ни простой, ни составной ключ. Для выполнения таких операций необходимо использовать режим конструктора. В приведенном примере показано, что в таблице, создаваемой в режиме таблицы, может быть использован только предлагаемый системой ключ с типом данных **Счетчик** (AutoNumber); кроме того, допустимо изменение типа данных для этого поля и как следствие использование в качестве простого ключа любого уникального поля таблицы.

Таким образом, если в таблице не предполагается использование в качестве ключа счетчика, целесообразно процедуру определения ключевых полей перенести в режим конструктора.



**Рис. 3.11.** Группа команд вкладки ленты **Режим таблицы**

Для добавления следующего поля в таблицу ПОКУПАТЕЛЬ в столбце с заголовком **Щелкните для добавления** (Click to Add) (см. рис. 3.9): введите его имя, например ИНН покупателя, предварительно выполнив двойной щелчок на заголовке, в ячейку под именем введите значение например, 0012345678. При вводе этого значения Access автоматически распознает и назначит числовой тип данных для этого поля. Однако, поскольку со значениями поля ИНН не будет производиться никаких арифметических операций и к тому же они могут начинаться с нулей, которые в числовых значениях аннулируются, следует изменить тип данных на **Текстовый** (Text). Для текстового поля автоматически выбирается длина, равная 255, что для полей нашей таблицы многовато. Изменить размер поля можно на ленте **Работа с таблицами | Поля** (Table Tools | Fields) в группе **Свойства** (Properties), введя его в соответствующую позицию. Здесь же можно изменить имя поля, указать его подпись, сформировать с помощью построителя выражений значение по умолчанию.

Имя поля, его подпись, отображаемая в заголовке, а также описание поля, могут быть определены с помощью команды **Имя и подпись** (Name & Caption) на вкладке **Работа с таблицами | Поля** (Table Tools | Fields) в группе **Свойства** (Properties). Заметим, что если свойство **Подпись** (Caption) поля не определено, в заголовке столбца таблицы отображается его имя.

В группе **Свойства** (Properties) для поля можно задать также **Значение по умолчанию** (Default Value), которое будет сохраняться в поле, если пользователь не ввел никакого значения.

Для определения типа данных добавляемого поля совсем не обязательно осуществлять ввод данных. Можно, щелкнув в столбце **Щелкните для добавления** (Click to Add), выбрать тип данных нового поля самостоятельно (рис. 3.12).

Используя команду **Вставить как поля** (Paste as Fields), можно добавить в таблицу новое поле, скопированное из другой таблицы. Откройте таблицу, из которой будет выполняться копирование, выделите нужное поле щелчком мыши на конкретном значении поля при появлении на нем знака плюс . Копируются конкретное значение поля, его имя и все свойства. Командой **Вставить как поля** (Paste as Fields) добавьте новое поле в таблицу.

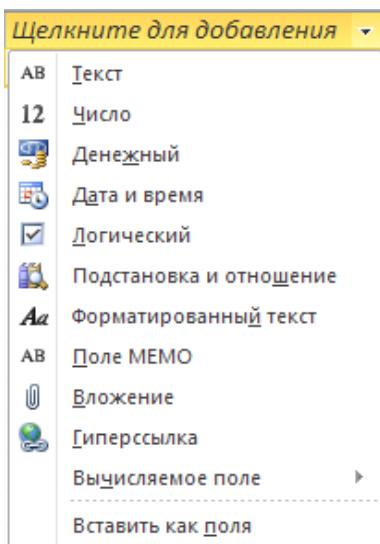


Рис. 3.12. Список типов данных для добавляемого поля

Наиболее простым способом добавления полей в таблицу можно считать использование команд группы **Добавление и удаление** (Add & Delete), в которой отдельными кнопками представлены основные типы данных и в раскрывающемся списке **Другие поля** (More Fields) представлены не только недостающие типы данных, но и форматы отображения поля (см. рис. 3.10).

Определив структуру таблицы в режиме таблицы, можно тут же приступить к заполнению ее данными — формированию записей таблицы. Команды по работе с записями таблицы, а также команды оформления внешнего вида таблицы представлены на ленте **Главная** (Home).

Если таблица была закрыта, открыть ее в режиме таблица можно, выбрав в области навигации и выполнив команду контекстного меню **Открыть** (Open).

### Задание 3.1. Создание таблицы базы данных в режиме таблицы

Завершите разработку структуры таблицы ПОКУПАТЕЛЬ и введите в нее записи, как показано в *Приложении 2*.

## Определение структуры таблицы в режиме конструктора

Для создания таблицы в режиме конструктора на вкладке **Создание** (Create) в группе **Таблицы** (Tables) следует выбрать команду **Конструктор таблиц** (Table Design). В режиме конструктора таблиц открывается окно **Таблица1** (Table1), в котором определяется структура таблицы базы данных (рис. 3.13).

Для перехода из режима таблицы в режим конструктора таблиц щелкните на кнопке **Режим** (View), доступной на вкладках **Работа с таблицами | Поля** (Table Tools | Fields) и **Главная** (Home). Кроме того, можно щелкнуть на кнопке **Конструктор** (Design View) в строке состояния или щелкнуть правой кнопкой мыши на заголовке таблицы и в контекстном меню выбрать команду **Конструктор** (Design View).

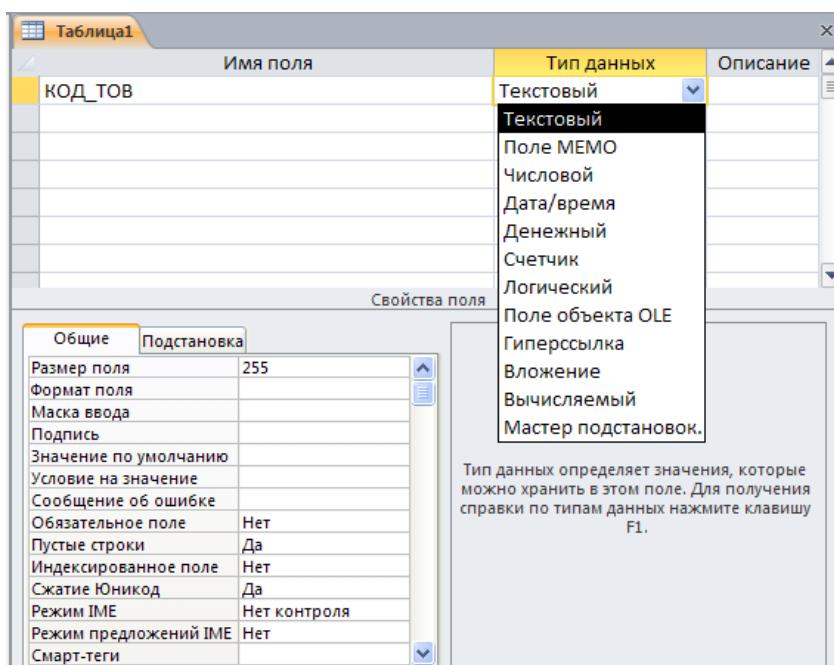


Рис. 3.13. Окно определения структуры таблицы в режиме конструктора

При переходе в режим конструктора таблиц делается активной лента команд **Работа с таблицами | Конструктор** (Table Tools | Design) (рис. 3.14).

### ЗАМЕЧАНИЕ

Назначение кнопки на ленте появляется (всплывает) при наведении курсора мыши на нее.

Для определения поля в окне конструктора таблицы (см. рис. 3.13) задаются **Имя поля** (Field Name), выбирается **Тип данных** (Data Type), вводится **Описание** (Description) — краткий комментарий, в разделе **Свойства поля** (Field Properties)

задаются свойства, представленные на двух вкладках: **Общие** (General) и **Подстановка** (Lookup). К общим относятся такие свойства поля, как максимальный размер, формат, подпись, которая выводится в заголовке столбца таблицы, значение по умолчанию и др. На вкладке **Подстановка** (Lookup) выбирается *тип элемента управления* (Display Control): поле, список фиксированных значений или поле со списком. Свойства поля зависят от выбранного типа данных. Для отображения свойств поля необходимо установить курсор на строке соответствующего поля. Ранее в главе были описаны все типы данных, используемые в Access, а также основные свойства полей.

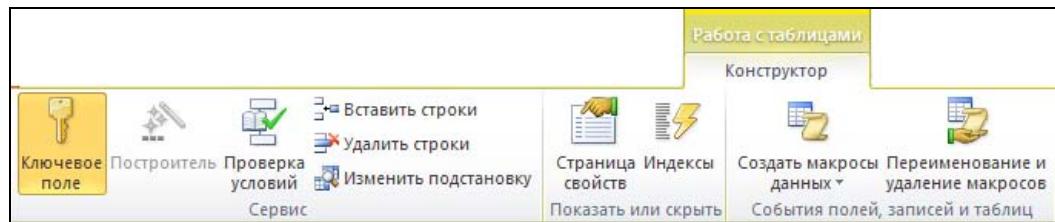


Рис. 3.14. Лента инструментов конструктора таблиц

## Создание таблиц базы данных Поставка товаров

Рассмотрим последовательность действий при создании таблиц на примере спроектированной в главе 2 базы данных *Поставка товаров*.

Напомним, что все объекты базы данных Access создаются и сохраняются в одном файле, который создается щелчком мыши в стартовом окне Access на значке **Новая база данных** (Blank Database). При этом в разделе **Новая база данных** (Blank Database) (см. рис. 3.1) задается имя базы данных, в примерах книги база имеет имя Поставка товаров.accdb, и выбирается папка, в которой нужно сохранить файл базы данных. В результате открывается окно новой пустой базы данных **Поставка товаров: база данных (Access 2007)**, аналогичное приведенному на рис. 3.3.

В соответствии с технологией проектирования реляционной базы данных структура каждой таблицы базы данных *Поставка товаров* должна определяться составом реквизитов соответствующего информационного объекта в ИЛМ.

Основные параметры структуры таблицы представлены в табл. 2.3, 2.5—2.7.

## Создание структуры таблицы

Если ранее в базе данных Поставка товаров была создана таблица ПОКУПАТЕЛЬ (см. разд. "Создание таблицы в режиме таблицы"), то после открытия базы данных в окне Access область документов будет пуста, а таблица ПОКУПАТЕЛЬ отобразится в области навигации. Теперь можно приступить к созданию других таб-

лиц базы данных в любом из режимов, предлагаемых Access. Начнем создание таблицы ТОВАР с определения ее структуры в режиме конструктора таблиц. На вкладке ленты **Создание** (Create) в группе **Таблицы** (Tables) выполним команду **Конструктор таблиц** (Table Design).

В окне конструктора **Таблица1** (Table1) (рис. 3.15) в соответствии с приведенными в табл. 3.1 проектными параметрами структуры определим все поля таблицы ТОВАР.

**Таблица 3.1. Основные параметры структуры таблицы ТОВАР**

Имя поля	Ключевое поле Уникальное	Обязательное поле	Тип данных	Размер	Число десятичных знаков	Подпись поля	Условие на значение	Сообщение об ошибке
КОД_ТОВ	Да	Да	Текстово- ый	5		Код товара		
НАИМ_ТОВ		Нет	Текстово- ый	25		Наиме- нование товара		
ЦЕНА			Денеж- ный	Денеж- ный	2	Цена товара	>=0 And <=35000	Цена должна быть >=0 и <=35000
ЕИ		Нет	Текстово- ый	8		Единица изме- рения		
СТАВКА_НДС		Нет	Числовой	Одинар- ное с пла- вающей точкой	0	Ставка НДС	>=0,05 And <=0,35	Ставка НДС =>5% и <=35%
ФОТО		Нет	Поле объекта OLE			Фото товара		
НАЛИЧИЕ_ТОВ		Нет	Логиче- ский			Наличие товара		
ВЛОЖЕНИЯ		Нет	Вложение	700 Кбайт				Поле вложе- ний
Цена с НДС		Нет	Вычисля- емый	Денеж- ный	2			

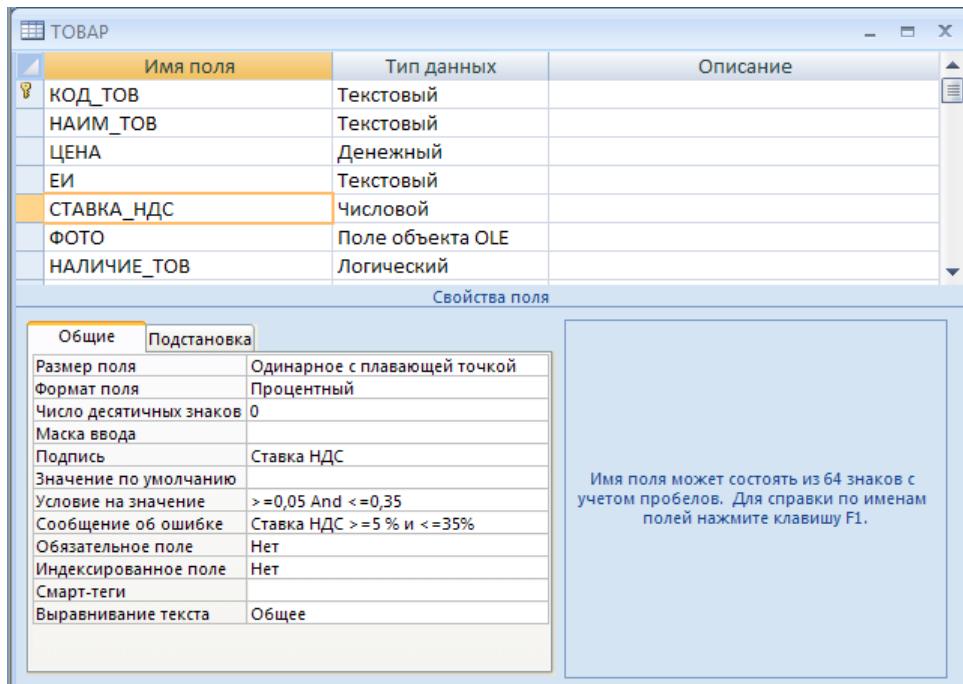


Рис. 3.15. Определение в таблице ТОВАР свойств поля СТАВКА\_НДС

Для каждого поля таблицы ТОВАР определим **Имя поля** (Field Name), **Тип данных** (Data Type) и на вкладке **Общие** (General) зададим свойства полей:

- ❖ для текстовых полей в свойстве **Размер поля** (Field Size) зададим размеры, указанные в соответствующем столбце табл. 3.1;
- ❖ для числового поля СТАВКА\_НДС выберем в свойстве:
  - ◆ **Размер поля** (Field Size) — **Одинарное с плавающей точкой** (Single);
  - ◆ **Формат поля** (Format) — **Процентный** (Percent). При отображении одинарных с плавающей точкой чисел, для которых выбран встроенный формат поля **Процентный** (Percent), их значения умножаются на 100 и к ним добавляется знак процента. При этом ввод ставки НДС должен осуществляться в формате отображения, например, 15%, 7,5%;
  - ◆ **Число десятичных знаков** (Decimal Places) — **0**;
- ❖ для поля ЦЕНА выберем в свойстве:
  - ◆ **Формат поля** (Format) — **Денежный** (Currency). Денежные суммы будут отображаться в соответствии с установленными по умолчанию в Microsoft Windows региональными параметрами в виде 123 456 789,00р.; Денежный тип поля используют для предотвращения округления во время вычислений. В денежных полях обеспечивается 15 знаков слева от десятичной запятой и 4 знака справа. Денежное поле занимает 8 байтов;
  - ◆ **Число десятичных знаков** (Decimal Places) — **2**;

- ❖ свойства **Подпись** (Caption), **Условие на значение** (Validation Rule), **Сообщение об ошибке** (Validation Text) — для каждого из полей зададим, как указано в табл. 3.1:
  - ◆ в поле СТАВКА\_НДС будут сохраняться числовые данные **Одинарное с плавающей точкой** (Single), поэтому в условии на значение должны использоваться соответствующие значения и нельзя вводить их в формате отображения, т. е. в нашем случае в процентах;
- ❖ для полей с числовыми данными ЦЕНА и СТАВКА\_НДС задайте свойство **Значение по умолчанию** (Default Value), например, равное 0. Это позволит правильно выполнять арифметические операции с этими полями;
- ❖ дополните таблицу ТОВАР вычисляемым полем, которое будет содержать цену с НДС. Для этого задайте имя поля, например Цена с НДС, выберите тип данных **Вычисляемый** (Calculated). Откроется построитель выражений, где, выбирая поле, имеющиеся в таблице ТОВАР, и знаки операторов, постройте выражение ЦЕНА+ЦЕНА\*СТАВКА\_НДС. Это же выражение будет записано в соответствующую строку общих свойств поля. Можно записать или откорректировать выражение в этом свойстве, не прибегая к помощи построителя. Выберите для свойства поля **Тип результата** (Result Type) значение **Денежный** (Currency).

### **ЗАМЕЧАНИЕ**

Если в некоторых записях таблицы не будет введено значение цены или ставки НДС, и для значения по умолчанию не было указано никаких числовых данных, результат в вычисляемом поле не отобразится. Это связано с тем, что при отсутствии значения в поле оно получает значение **Null**, с которым не могут производиться вычисления.

Условие на значение, которое заносится в свойство поля, является выражением. Оно может быть сформировано с помощью построителя выражений (рис. 3.16). Построитель вызывается в окне конструктора таблиц (см. рис. 3.15) при нажатии кнопки, расположенной справа от строки **Условие на значение** (Validation Rule), в которую должно быть введено выражение. Построитель можно вызвать и командой **Построитель** (Builder) на панели конструктора в группе **Сервис** (Tools).

### **ЗАМЕЧАНИЕ**

Для ввода операторов "больше равно" и "меньше равно" существуют специальные знаки. Не допускайте пробелов при вводе выражения. Нужные пробелы построитель введет сам.

После ввода выражения в окно построителя и нажатия кнопки **OK** Access выполнит синтаксический анализ выражения и отобразит его в строке **Условие на значение** (Validation Rule).

Теперь определим первичный ключ таблицы. Выделим поле КОД\_ТОВ, щелкнув кнопкой мыши на области маркировки слева от имени поля, и нажмем кнопку **Ключевое поле** (Primary Key) на вкладке ленты **Конструктор** (Design) в группе **Сервис** (Tools) (см. рис. 3.14). Признаком установки ключа является изображение ключа слева от имени поля. Определим свойства ключевого поля в соответствии с табл. 3.1.

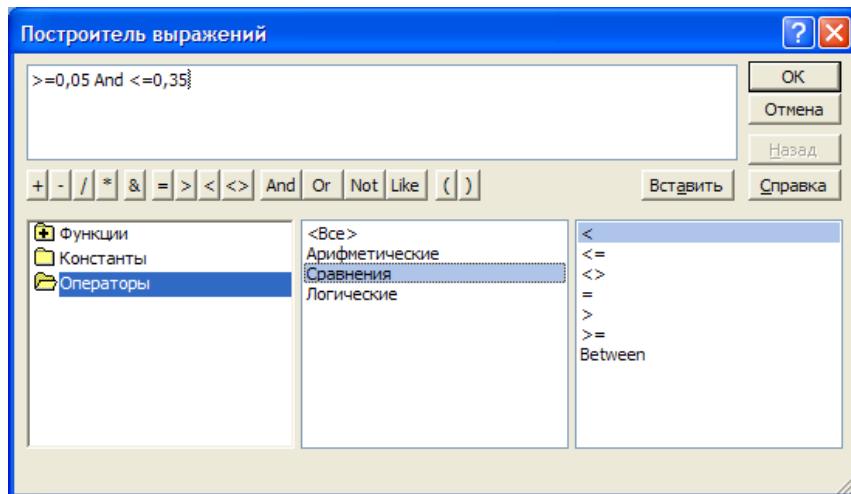


Рис. 3.16. Окно построителя выражений с выведенным списком операторов сравнения

### **ЗАМЕЧАНИЕ**

Для инвентарных номеров и других номеров или кодов, которые часто выступают в качестве ключей и не используются в математических вычислениях, вместо числового целесообразно выбрать текстовый тип данных.

Сохраним созданную структуру таблицы и присвоим имя новой таблице — ТОВАР. Для этого выполним команду **Сохранить** (Save) в контекстном меню таблицы, на **Панели быстрого доступа** (Customize Quick Access Toolbar) или на вкладке **Файл** (File). В окне **Сохранение** (Save As) введем имя таблицы (рис. 3.17).

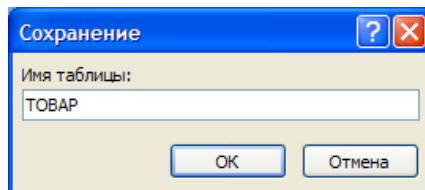
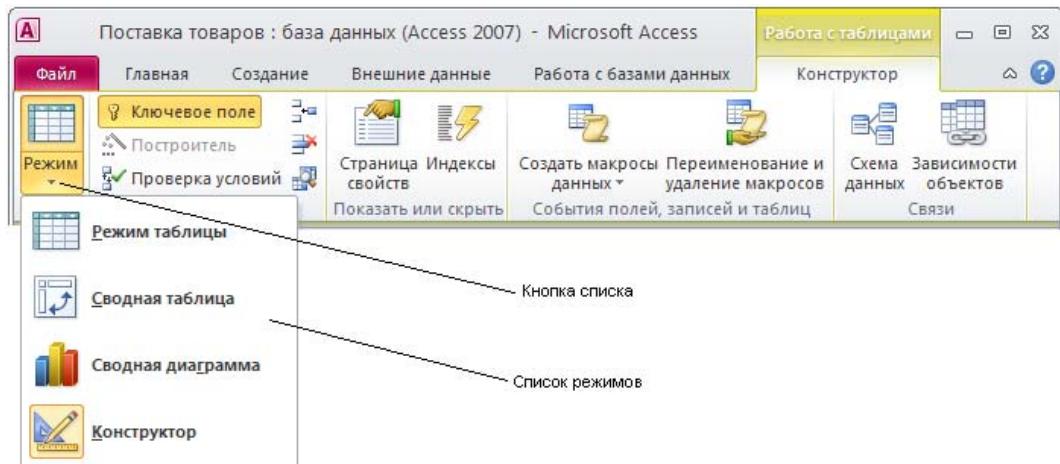


Рис. 3.17. Окно ввода имени сохраняемой таблицы

При сохранении таблицы происходит обновление файла базы данных, в которую помещается созданная таблица.

Таблица ТОВАР появится в списке объектов **Таблицы** (Tables) в области навигации открытой базы данных *Поставка товаров*.

После сохранения структуры таблицы переходите ко второму этапу создания таблицы — созданию записей. Для этого переключитесь в режим таблицы нажатием кнопки **Режим** (View) на ленте конструктора или выбором этого режима при открытии списка этой кнопки (рис. 3.18).



**Рис. 3.18.** Выбор режима представления таблицы

В Access, начиная с версии 2007, появились разнообразные средства для добавления и определения свойств полей таблицы в режиме таблицы. Все таблицы в режиме таблицы, если в них определен ключ, содержат пустой столбец с заголовком **Щелкните для добавления** (Click to Add). Для определения некоторых полей в таблице достаточно ввести данные в первую пустую ячейку под этим заголовком, и Access сам определит тип данных и некоторые свойства поля. Можно также скопировать и вставить данные в пустой столбец. Можно самостоятельно выбрать нужный тип данных для поля и установить многие его свойства. В Access 2010 на ленте **Работа с таблицами | Поля** (Table Tools | Fields) можно определить множество разнообразных типов данных и их свойств.

Находясь в режиме таблицы, когда отображены значения в полях, удобно задавать такие свойства, как **Условие на значение** (Validation Rule) и **Сообщение об ошибке** (Validation Text), а также сопоставить значения полей записи, например [ЦенаПродажи]>[ОптоваяЦена]. Для этого нужно выбрать поле и открыть список кнопки **Проверка** (Validation) в группе **Проверка поля** (Field Validation). Если в списке выбрать **Правило проверки поля** ((Field Validation Rule), откроется **Построитель** (Expression Builder), в котором нужно ввести выражение для проверки вводимых в поле значений. Выбрав **Сообщение проверки поля** (Field Validation Message), можно ввести текст, который будет выводиться при отклонении вводимых значений от допустимых. Другие команды списка позволяют задать выражение для сравнительной проверки значений в полях записи и подготовить сообщение при нарушении условий, заданных выражением. Это выражение сохраняется в свойстве таблицы **Условие на значение** (Validation Rule). Открывается окно свойств таблицы в режиме конструктора кнопкой **Страница свойств** (Property Sheet). Для таблицы может быть введено только одно такое выражение.

Как и в режиме конструктора в режиме таблицы можно удалить столбец. При этом следует помнить, что удаляются все данные столбца, и отменить удаление не-

возможно. Удаление поля первичного ключа в режиме таблицы невозможно. Для этого необходимо использовать режим конструктора.

### Задание 3.2. Создание таблиц базы данных

Создайте структуру таблиц СКЛАД, НАКЛАДНАЯ, ОТГРУЗКА, ДОГОВОР, ПОСТАВКА\_ПЛАН. При создании таблиц используйте проектные параметры их структуры (см. *Приложение I*).

Рассмотрим некоторые особенности определения структуры таблиц ТОВАР, НАКЛАДНАЯ, ОТГРУЗКА, ПОКУПАТЕЛЬ, ДОГОВОР, ПОСТАВКА\_ПЛАН базы данных *Поставка товаров*.

## Определение составного первичного ключа

В таблице НАКЛАДНАЯ в составной первичный (уникальный) ключ входят поля НОМ\_НАКЛ и КОД\_СК. Определение этого ключа возможно только в режиме конструктора таблиц. Выделите оба эти поля, щелкнув кнопкой мыши на области маркировки при нажатой клавише <Ctrl>. Затем нажмите на ленте **Работа с таблицами** | **Конструктор** (Table Tools | Design) в группе **Сервис** (Tools) кнопку **Ключевое поле** (Primary Key).

Аналогично определяются составные ключи в таблицах ОТГРУЗКА и ПОСТАВКА\_ПЛАН.

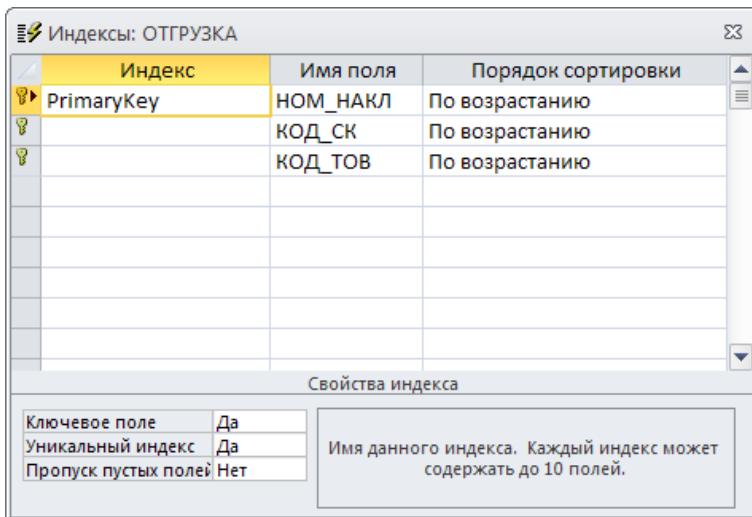


Рис. 3.19. Уникальный индекс по составному первичному ключу таблицы ОТГРУЗКА

Для ключевого поля автоматически создается уникальный индекс. Для его просмотра выполните команду **Индексы** (Indexes) на ленте **Работа с таблицами** |

**Конструктор** (Table Tools | Design) в группе **Показать или скрыть** (Show /Hide). Окно для просмотра и редактирования индексов таблицы ОТГРУЗКА представлено на рис. 3.19.

На рис. 3.19 показано, что индексу составного первичного ключа присвоено имя **PrimaryKey**, в столбце **Имя поля** (Field Name) перечисляются имена полей, составляющих индекс. Индекс первичного ключа всегда уникален и не допускает пустых полей в записях.

## Использование данных типа

### **Поле объекта OLE**

В таблице ТОВАР предусмотрено поле ФОТО, которое должно содержать фотографию товара, хранящуюся в формате графического редактора Paint в файле с расширением bmp. Тип данных такого поля определен как **Поле объекта OLE** (OLE Object). Размещение этого объекта в поле производится на этапе заполнения полей таблицы данными в режиме таблицы или через форму. В режиме таблицы Access отобразит тип объекта "Точечный рисунок". Отображение объекта возможно только в форме или отчете. Объект может быть внедренным или связанным.

#### **ЗАМЕЧАНИЕ**

OLE (Object Linking and Embedding, связывание и внедрение объектов) — это метод передачи информации в виде объектов между приложениями Windows. **Поле объекта OLE** (OLE Object) является средством, позволяющим установить связь с объектами другого приложения или внедрить объект в базу данных. Объектом является документ или его часть, созданная в другом приложении, сохраняющая формат документа-источника и информацию о создавшем приложении. Объектами могут быть простые и форматированные тексты, рисунки, диаграммы, файлы звукозаписи (WAV), музыка в формате MIDI (музыкально-инструментальный цифровой интерфейс), файлы анимации (FLI, MMM), видеоклипы (AVI), электронные таблицы и другие элементы из различных приложений, поддерживающих это средство. СУБД Access, поддерживая OLE, полностью интегрирована с другими приложениями Microsoft Windows.

*Внедренный объект* сохраняется в файле базы данных. Двойным щелчком мыши на ячейке, содержащей внедренный объект, предоставляется возможность редактирования объекта средствами приложения, в котором объект был создан.

*Связанный объект* сохраняется в отдельном файле. Файл объекта можно обновлять независимо от базы данных. Последние изменения будут выведены на экран при следующем открытии формы или отчета. При работе с базой данных также можно просматривать и редактировать объект. Отредактированный связанный объект будет сохраняться в файле объекта, а не в файле базы данных. Связывание объекта удобно при работе с большими объектами, которые нежелательно включать в файл базы данных, а также с объектами, используемыми в различных документах организации, например логотип. Если связанный файл объекта перемещен, необходимо повторно установить с ним связь.

## Использование данных типа **Вложение**

Тип данных **Вложение** (Attachment) можно использовать для хранения нескольких файлов в одном поле, причем в этом поле можно хранить файлы разных типов. Например, в таблице ТОВАР можно в поле с типом данных **Вложение** добавить несколько файлов со сведениями о товаре: сертификат происхождения, сертификат качества, инструкцию по эксплуатации, рекламации, фотографии, схемы и т. д.

Вкладывать файлы можно только в базы данных, созданные в приложении Access 2007/2010 в новом формате файла accdb.

Тип данных **Вложение** (Attachment) предлагается для хранения изображений и документов вместо типа данных **Поле объекта OLE** (OLE Object). Вложениям необходимо значительно меньше места для хранения и не требуется применения программ, называемых OLE-серверами.

В соответствии с правилами нормализации каждое поле в реляционной базе данных должно содержать только один элемент данных. В противном случае поиск данных будет затруднен или даже невозможен. Вложения нарушают правила разработки базы данных, поскольку в поле можно добавлять несколько файлов — элементов данных. Чтобы избежать этих нарушений, Access по мере добавления файлов вложения создает системные таблицы, которые неявно используются для нормализации данных.

## Использование данных типа **Поле МЕМО**

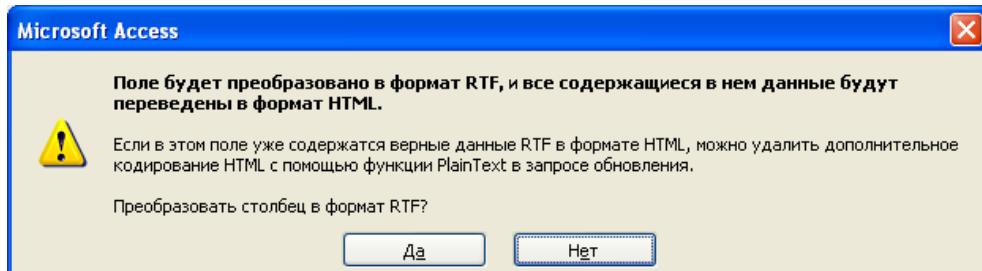
В таблице ПОКУПАТЕЛЬ предусмотрено поле ОПИСАНИЕ, которое будет содержать текстовое данное большой длины — характеристику покупателя. Для такого поля выберите тип данного — **Поле МЕМО** (Memo). Ввод данных в это поле можно выполнить непосредственно в поле таблицы, либо через область ввода, вызываемую нажатием комбинации клавиш <Shift>+<F2>.

Если тексты описаний покупателей подготовлены в некотором текстовом редакторе, например Microsoft Word, и хранятся в отдельных файлах, для этого поля может быть задан тип **Поле объекта OLE** (OLE Object) или **Вложение** (Attachment).

Access 2007/2010 в поле с типом данных МЕМО обеспечивает хранение форматируемого текста. Это единственный тип данных в Access, который имеет встроенную поддержку хранения и отображения форматированного текста. Если в поле нужно хранить форматированный текст, создайте поле МЕМО и затем в режиме конструктора выберите для свойства **Формат текста** (Text Format) значение **Формат RTF** (Rich Text) вместо **Обычный текст** (Plain Text). В режиме таблицы для выбора этого свойства предназначена кнопка **Параметры поля МЕМО** (Memo Settings) на ленте **Работа с таблицами | Поля** (Table Tools | Fields) в группе **Свойства** (Properties). Сообщение системы о преобразовании формата показано на рис. 3.20.

Текст поля МЕМО в формате RTF отображается как форматированный, хранится и интерпретируется как формат HTML, допускает использование общих

средств форматирования приложений Microsoft Office, недоступных в обычном тексте. Access автоматически применяет HTML-форматирование к тексту в формате RTF. Использование HTML обусловлено большей степенью совместимости с полями в формате RTF, хранящимися в списках SharePoint.



**Рис. 3.20.** Сообщение при выборе для свойства **Формат текста** значения **Формат RTF**

К тексту в формате RTF Access допускает применение таких параметров форматирования, как шрифт, размер, цвет, различные виды начертания, выравнивания текста, нумерованные и маркированные списки, уменьшение и увеличение отступа абзаца.

Можно применить форматирование ко всему содержимому поля или его части, редактируя поле в режиме таблицы или в форме. Можно также применить форматирование ко всему полю при просмотре отчета в режиме макета.

Чтобы применить параметры форматирования к тексту RTF, выделите текст, который нужно форматировать, и используйте команды групп **Форматирование текста** (Text Formatting) на ленте **Главная** (Home). Кроме того, после выделения текста мышью появляется мини-панель инструментов, которая содержит кнопки параметров форматирования (рис. 3.21).



**Рис. 3.21.** Мини-панель инструментов форматирования текста RTF в поле МЕМО

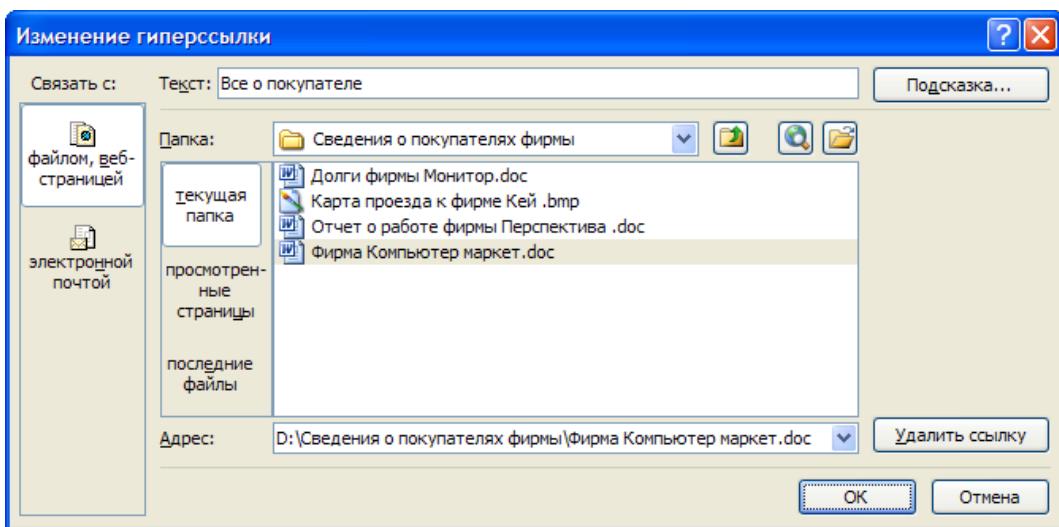
## Использование данных типа **Гиперссылка**

Специальный тип данных **Гиперссылка** (Hyperlink) позволяет хранить гиперссылки в полях таблиц базы данных.

Поля с типом данных **Гиперссылка** (Hyperlink) используются для перехода к документам, созданным в различных приложениях Microsoft Office и сохраняется в файлах на локальных или сетевых дисках, в том числе к другим базам

данных, к страницам веб-серверов в Интернете или интранете. При щелчке мышью на гиперссылке осуществляется переход к документу, который открывается создавшим его приложением. Гиперссылки в полях записываются как URL- или UNC-адреса. Отображается в поле как подчеркнутый текст, заданный при определении гиперссылки.

В таблице ПОКУПАТЕЛЬ предусмотрено поле ЛИЧ\_ДОК, в котором может храниться, например, ссылка на некоторый документ, характеризующий покупателя, или ссылка на его веб-страницу или на его базу данных. Для создания гиперссылки в этом поле откройте таблицу в режиме таблицы. Введите в поле текст, который должен отображаться в поле и в то же время определит содержание открываемого по гиперссылке документа, например **Все о покупателе**. Нажмите правую кнопку мыши и выберите в контекстном меню команду **Гиперссылка | Изменить гиперссылку** (Hyperlink | Edit Hyperlink). В открывшемся окне **Изменение гиперссылки** (Edit Hyperlink) выберите папку и файл, который хранит нужный документ (рис. 3.22).



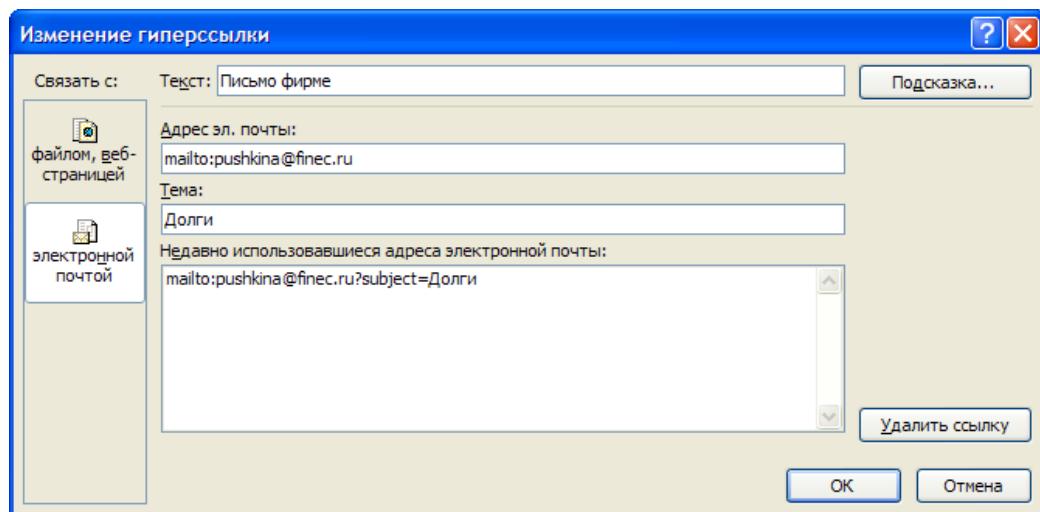
**Рис. 3.22.** Определение файла, который должен открываться по гиперссылке **Все о покупателе**

Чтобы добавить гиперссылку для создания сообщений электронной почты, также введите в поле текст, например **Письмо фирме**, и в диалоговом окне **Изменение гиперссылки** (Edit Hyperlink) заполните поля, как показано на рис. 3.23.

Щелчком на поле с этой гиперссылкой открывается установленная почтовая программа, например Outlook Express или Outlook, и окно нового сообщения с заполненными полями **Кому** и **Тема**.

Для создания гиперссылки на веб-страницу запишите в поле, например, такой текст: **Страница покупателя#http://comp.ru/gorod/firm/mircom.htm#Реквизиты**.

По такой гиперссылке будет выполняться переход на страницу фирмы, хранящуюся на сервере в папке, определяемой путем /gorod/firm/ в файле mircom.htm. Причем будет открыт раздел с именем Реквизиты, а в поле гиперссылки будет записано **Страница покупателя**. Если открыть окно **Изменение гиперссылки** (Edit Hyperlink), то можно убедиться, что введенная в поле информация размещена в соответствующих областях. Страница покупателя — в области **Текст** (Text), а вся остальная часть — в области **Адрес** (Address). Очевидно, что как отображаемый текст гиперссылки, так и адрес может непосредственно вводиться в окне **Изменение гиперссылки** (Edit Hyperlink). Кроме того, адрес может формироваться путем выбора ранее просмотренных страниц.



**Рис. 3.23.** Определение для гиперссылки адреса электронной почты и темы письма

Удалить гиперссылку из поля можно командой контекстного меню поля **Гиперссылка | Удалить гиперссылку** (Hyperlink | Remove Hyperlink) или простым нажатием на выделенной гиперссылке клавиши <Delete>.

## Использование данных типа *Дата/время*

В таблице ДОГОВОР для сохранения даты заключения договора предусмотрено поле ДАТА\_ДОГ. Выберите для этого поля тип данного **Дата/время** (Date/Time). Для отображения данных этого типа может быть выбран один из следующих форматов поля: **Полный** (General Date), **Краткий** (Short Date), **Средний** (Medium Date) или **Длинный формат даты** (Long Date). В режиме конструктора образцы дат в каждом из форматов показаны в строках списка при выборе формата.

В Access 2007/2010 с полем даты автоматически связывается элемент управления Календарь, отображаемый при переходе в поле даты значком справа от него.

Для ввода даты щелкните на этом значке и в открывшемся календаре выберите нужную дату (рис. 3.24).

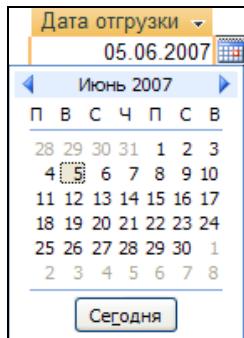


Рис. 3.24. Выбор в календаре значения даты для ввода в поле

## Маска ввода для даты и текста

Для управления вводом даты в таблице ДОГОВОР в свойствах поля ДАТА\_ДОГ может быть определена маска ввода. Маска ввода задает формат, состоящий из постоянных символов (таких как скобки, точки или дефисы) и специальных символов маски, указывающих, в какие позиции, в каком количестве и какого типа данные могут быть введены. Например, маска для ввода даты в кратком формате имеет вид 00.00.0000, в среднем формате — 00->L<LL-0000. Воспользуйтесь значком построителя в конце строки свойства **Маска ввода** (Input Mask) для вызова мастера, который позволит без труда сформировать маску ввода.

### **ВНИМАНИЕ!**

При определении для поля даты маски ввода календарь не отображается.

Для ввода номера телефона в поле ТЕЛ таблицы ПОКУПАТЕЛЬ определите маску (999)000-0099, также воспользовавшись мастером. Эта маска позволит вводить номера телефонов с кодами городов и без них. Число цифр в номере телефона не может быть менее 5 и более 7. Символ 9 означает, что вместо цифр кода можно ввести пробелы, а последние две цифры номера вводить не обязательно.

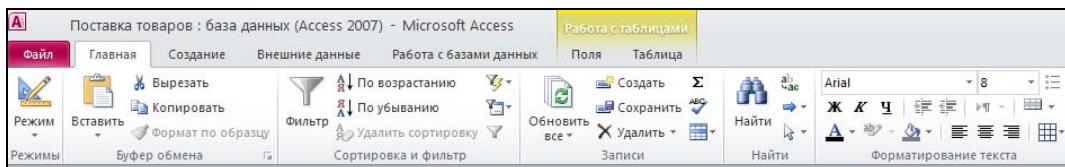
## Непосредственный ввод данных в таблицы

Определив структуру, можно приступить ко второму этапу создания таблицы — вводу данных в нее.

Непосредственный ввод данных в таблицу осуществляется в **Режиме таблицы** (Datasheet). Переход в режим таблицы из режима конструктора осуществляется нажатием кнопки **Режим** (View) на ленте **Главная** (Home) или **Конструктор**

(Design), выполнением команды контекстного меню, вызываемого правой кнопкой мыши на заголовке таблицы, щелчком мыши на соответствующей кнопке в строке состояния. Если таблица закрыта, выберите ее в области переходов и выполните команду контекстного меню **Открыть** (Open) или просто щелкните на таблице дважды.

В этом режиме отображается лента **Главная** (Home) (рис. 3.25), предназначенная для работы с записями таблицы, их сортировки и фильтрации, поиска в таблице конкретных значений и их замены, определения ряда параметров макета. Кроме того, при переходе в режим таблицы делается доступной лента **Работа с таблицами** (Table Tools), предназначенная для изменения параметров структуры таблицы без перехода в режим конструктора. Лента **Работа с таблицами | Поля** (Table Tools | Fields) представлена выше на рис. 3.10.



**Рис. 3.25.** Лента **Главная** при открытой таблице в режиме таблицы

По умолчанию база данных открывается для работы в многопользовательской среде для совместного доступа с возможностью чтения и записи в базу данных для всех пользователей.

В режиме таблицы предоставляется возможность вводить новые записи в таблицу путем заполнения значениями ее полей.

Очевидно, вводимые в поля таблицы значения данных должны соответствовать типам данных, определенным в структуре, а способ их записи — установленным форматам данных. Значения должны удовлетворять накладываемым ограничениям. После ввода значения в ячейку поля при попытке перейти к другой ячейке Access проверяет, является ли введенное данное допустимым для этого поля. Если введено значение, не соответствующее типу данных поля, Access пытается преобразовать его в правильный тип данных. Если значение не является допустимым и преобразование невозможно, например, нельзя преобразовать текст в число, появляется предупреждающее сообщение. Для того чтобы выйти из ячейки, следует ввести правильное значение или отменить внесенные изменения клавишей <Esc>.

Обязательно должны вводиться значения в поля первичного ключа и поля связи, а также в поля, для которых заданы свойства **Обязательное поле** (Required), **Условие на значение** (Validation Rule) или в свойстве **Пустые строки** (Allow Zero Length) выбрано значение **Нет** (No).

При заполнении связанных таблиц необходимо обеспечить правильность связей между записями этих таблиц. Так, при одно-многозначных связях таблиц и вводе записей в подчиненную таблицу необходимо отслеживать наличие в главной таблице записей с соответствующими значениями ключевых полей. При изменении

или удалении ключевых полей в записях главной таблицы нужно изменять или удалять связанные с ними записи в подчиненных таблицах.

## Макет таблицы

Для удобства работы с таблицей можно изменить ее представление на экране. При этом можно менять ширину столбца, высоту строки, шрифт данных таблицы, цвет текста, линий сетки и фона, оформление, которое может быть обычным, приподнятым или утопленным. Можно выводить на экран только те столбцы, которые нужны для текущей работы, можно зафиксировать столбец при просмотре широких таблиц. Эти параметры отображения таблицы на экране называются *макетом* таблицы и сохраняются вместе с ней.

Настройка макета выполняется в режиме таблицы. При этом могут быть использованы команды ленты **Главная** (Home).

Многие операции настройки макета можно выполнить непосредственно в таблице с помощью мыши.

- ❖ *Изменение ширины столбца.* Для изменения ширины столбца курсор мыши устанавливается на линию, разделяющую имена столбцов. При этом он превращается в планочку со стрелками в обе стороны. Далее границу столбца можно перетащить в нужное место.
- ❖ *Изменение высоты строки.* Для изменения высоты строки курсор мыши устанавливается в области маркировки записи, расположенной слева, на границе между записями. Граница строки перетаскивается на требуемое расстояние. При этом изменяется высота всех строк таблицы.
- ❖ *Скрыть/отобразить столбец с экрана.* Убрать столбец можно, перетащив его правую границу влево до исчезновения столбца. Для восстановления отображения скрытого столбца следует установить курсор чуть правее границы столбцов, между которыми размещен скрытый (двунаправленная стрелка при этом имеет разрыв), и перетащить ее вправо.
- ❖ *Изменение порядка расположения столбцов на экране.* Столбец выделяется щелчком кнопки мыши на его имени. Протащив курсор мыши поперек столбцов, можно выделить несколько столбцов. Выделенный столбец перетаскивается в новое место при установке курсора на его имени (области маркировки столбца).
- ❖ *Изменить ширину столбца, высоту строк, скрыть/отобразить столбцы*, не нужные для текущей работы, *закрепить/освободить столбцы* можно с помощью соответствующих команд, доступных в контекстном меню выделенного столбца и в списке команды  **Дополнительно** (More), размещенной на вкладке ленты **Главная** (Home) в группе **Записи** (Records).
- ❖ *Сохранение макета таблицы* выполняется командой **Сохранить** (Save) на вкладке ленты **Главная** (Home) в группе **Записи** (Records), в меню вкладки

**Файл** (File) или при закрытии таблицы после утвердительного ответа на вопрос "Сохранить изменения макета таблицы '<имя таблицы>?'?" (Do you want to save changes to the layout of table <имя таблицы>?).

## Ввод данных в таблицы базы данных

Рассмотрим ввод данных на примерах таблиц базы данных *Поставка товаров*. В режиме таблицы введем несколько записей в таблицу ТОВАР. В области навигации, представляющей все объекты базы данных *Поставка товаров*, установим курсор на таблице ТОВАР и в контекстном меню выберем команду **Открыть** (Open) или просто дважды щелкнем на имени таблицы. Таблица откроется в режиме таблицы (Datasheet View). Заполним записи (строки) открывшейся таблицы (рис. 3.26) в соответствии с названиями полей (столбцов).

Код тов.	Наименование товара	Цена	Единица	Ставка НД	Фото	Наличие		Цена с НДС	Добавить
T001	Монитор 17LG	10 000,00р.	штука	5%	рисунок	<input checked="" type="checkbox"/>	0(1)	10 500,00р.	
T002	FDD 3,5	300,00р.	коробка	6%		<input type="checkbox"/>	0(1)	318,00р.	
T003	HDD Maxtor 20GB	231,00р.	штук	5%		<input checked="" type="checkbox"/>	0(2)	242,55р.	
T004	Корпус MiniTower	916,00р.	штука	10%		<input type="checkbox"/>	0(0)	1 007,60р.	
T005	CD-ROM Panasonic IDE	1 153,00р.	штуки	30%		<input type="checkbox"/>	0(0)	1 498,90р.	
T006	DIMM 64M PC100	300,00р.	штука	15%		<input type="checkbox"/>	0(0)	345,00р.	
T007	Принтер EPSON ST.A4	2 400,00р.	штука	10%	рисунок	<input checked="" type="checkbox"/>	0(0)	2 640,00р.	
T008	СканерAcer	2 338,00р.	штуки	16%		<input type="checkbox"/>	0(0)	2 712,08р.	
T009	Зв. Карта Genius Liv	789,00р.	штука	10%		<input type="checkbox"/>	0(0)	867,90р.	
T010	Модем Genius ext	1 295,00р.	штук	10%		<input type="checkbox"/>	0(0)	1 359,75р.	
*		0,00р.				<input checked="" type="checkbox"/>	0(0)		

Записи: 10 из 10    Назад в строку    Поиск

Кнопки перехода по записям

Маркер строки, в которую вводится новая запись

Маркер записи, в которую вносятся изменения

Рис. 3.26. Непосредственный ввод записей в таблицу ТОВАР

Корректность вводимых данных (соответствие заданному типу поля, размеру и условию на значение, которые определены в свойствах полей в режиме конструктора) проверяется автоматически при их вводе. Отслеживается уникальность значений ключевых полей.

Отменить ввод значения в поля текущей записи можно, нажав клавишу <Esc> или выполнив команду **Отменить** (Undo) на панели быстрого доступа. Переход от одного поля к другому можно выполнить клавишами <Enter>, <Tab> или переводом курсора.

Завершение ввода новых значений записи или редактирования осуществляется при переходе к любой другой записи (при смене текущей записи). После перехода к другой записи можно отменить ввод (редактирование) всей записи, также выполнив команду **Отменить** (Undo) на панели быстрого доступа.

## **ВНИМАНИЕ!**

Команда **Отменить** (Undo) в зависимости от контекста приобретает различный смысл. По этой команде могут быть отменены только изменения, касающиеся одной последней записи.

*Добавление записи* в таблицу начинается с заполнения пустой строки, размещенной в конце таблицы и помеченной звездочкой (\*).

Переход к этой записи можно выполнить также по команде **Создать** (New) на вкладке ленты **Главная** (Home) в группе **Записи** (Records). Корректируемая запись помечается слева символом карандаша.

При создании новой записи должны быть заполнены ключевые поля, поля связи, если установлены связи таблицы с другими таблицами, а также поля, требующие обязательного заполнения в соответствии со свойствами поля.

*Сохранение новой* записи происходит после перехода к другой записи или выполнения команды **Сохранить** (Save) на ленте **Главная** (Home) в группе **Записи** (Records).

Для *удаления записи* в таблице ее нужно выделить и в списке кнопки **Удалить** (Delete) на ленте **Главная** (Home) в группе **Записи** (Records) выбрать соответствующую команду. Для исключения ошибочного удаления в Access предусмотрен запрос на подтверждение удаления. После подтверждения на удаление восстановление удаленной строки невозможно.

*Изменение значений* в полях записи осуществляется непосредственно в ячейках таблицы. Введенное значение проверяется Access при попытке перевода курсора в другое поле. Если значение не является допустимым, появляется предупреждающее сообщение. Для того чтобы выйти из поля, следует ввести правильное значение или отменить внесенные изменения. Отмена изменения значения производится нажатием клавиши <Esc> или выполнением команды **Отменить** (Undo) на панели быстрого доступа. Откорректированная запись сохраняется после перехода к другой записи или принудительного сохранения командой **Сохранить** (Save) на ленте **Главная** (Home) в группе **Записи** (Records).

*Режим ввода записи.* При дополнении таблицы новыми записями может быть использован режим ввода записи, при котором видны только записи, введенные после открытия таблицы. Для реализации этого режима может быть использован фильтр, например, по ключевому полю, в котором отмечается вывод только пустых записей. Откройте таблицу в режиме таблицы, кнопкой списка на заголовке ключевого поля откройте меню и в области перечисления уникальных значений оставьте помеченным только значение **Пустые** (Blanks). В результате в таблице будет отображена только одна строка новой пустой записи, с которой можно начать ввод нужного числа записей. При этом в свойства таблицы будет записан созданный фильтр.

Если необходимо, чтобы при открытии таблицы применялся сохраненный фильтр и устанавливался режим ввода новых записей, выберите в свойстве таблицы **Фильтр при загрузке** (Filter On Load) значение **Да** (Yes). Открывается окно свойств таблицы в режиме конструктора кнопкой **Страница свойств** (Property Sheet).

Чтобы вернуться к отображению без фильтра, необходимо удалить фильтр, например, нажав кнопку **С фильтром** (Filtered) на панели переходов по записям в нижней части окна таблицы или выполнив команду **Удалить фильтр** (Filter) на вкладке **Главная** (Home) в группе **Сортировка и фильтр** (Sort & Filter). При этом фильтр удаляется временно, сохраняя возможность вернуться к исходному представлению. Для повторного применения фильтра используется команда **Применить фильтр** (Toggle Filter) или кнопка **Без фильтра** (Unfiltered).

Для окончательного удаления фильтра необходимо очистить его командой **Очистить все фильтры** (Clear All Filters), размещенной в списке кнопки **Дополнительно** (Advanced) .

- Вывод строки итогов в таблице.* Для подсчета числа товаров в таблице или средней, минимальной или максимальной цены товара выполните на ленте **Главная** (Home) в группе **Записи** (Records) команду **Итоги** (Totals). Стока **Итог** (Total) будет отображена под строкой со звездочкой. В строке **Итог** щелкните в столбце ЦЕНА и нажмите кнопку со стрелкой вниз. В открывшемся списке выберите **Количество значений** (Count). Также просмотрите среднее, минимальное и максимальное значения для цены. Повторным выполнением команды **Итоги** (Totals) скройте строку **Итог** (Total) из таблицы. Если выполнить команду **Итоги** (Totals) еще раз, отобразится строка **Итог** с последним выбранным значением.

## Размещение объекта OLE

Рассмотрим размещение объекта OLE на примере поля ФОТО в таблице ТОВАР. Пусть фотографии хранятся в формате графического редактора Paint в файлах с расширением bmp.

Рассмотрим вариант внедрения объекта в файл базы данных. Установим курсор в соответствующем поле таблицы (см. рис. 3.26). Выполним команду контекстного меню **Вставить объект** (Insert Object). В открывающемся окне (рис. 3.27) отметим переключатель **Создать из файла** (Create from File).

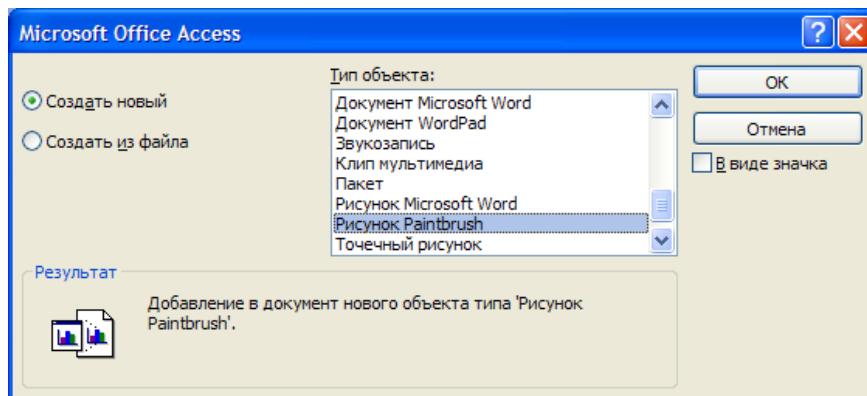


Рис. 3.27. Окно для выбора варианта вставки объекта

В следующем окне (рис. 3.28) нужно ввести имя файла с фотографией. Для поиска файла можно воспользоваться кнопкой **Обзор** (Browse), по которой выведется диалоговое окно, позволяющее просмотреть диски и папки и выбрать необходимый файл.

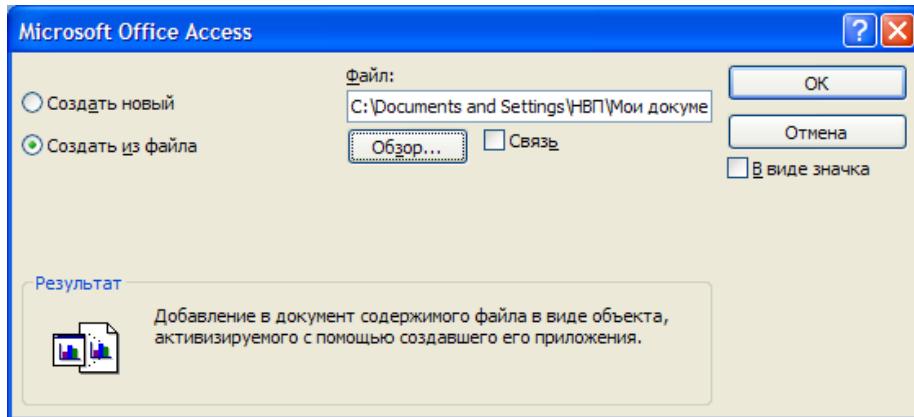


Рис. 3.28. Окно для выбора вставляемого объекта из файла

### **ВНИМАНИЕ!**

Флажок **Связь** (Link) по умолчанию не отмечен и, следовательно, содержимое файла будет введено в поле как внедренный объект. Увидеть содержимое поля можно через форму или отчет. Дальнейшие изменения в файле не будут отражаться на встроенным объекте. Для ввода в поле *связанного объекта* надо установить флажок **Связь** (Link). Это сэкономит место в базе данных и даст возможность отображать все изменения, вносимые в файл другими приложениями.

Результат заполнения таблицы представлен на рис. 3.26, где в соответствующем поле указан вид объекта — Точечный рисунок. Для просмотра внедренного объекта в создавшем его приложении достаточно в соответствующем поле установить курсор и дважды щелкнуть кнопкой мыши.

Для отображения содержимого поля в виде значка, представляющего файл с документом, надо в окне вставки объекта (см. рис. 3.27) установить флажок **В виде значка** (Display As Icon). Значок может быть использован для представления связанного объекта.

### **Размещение вложений**

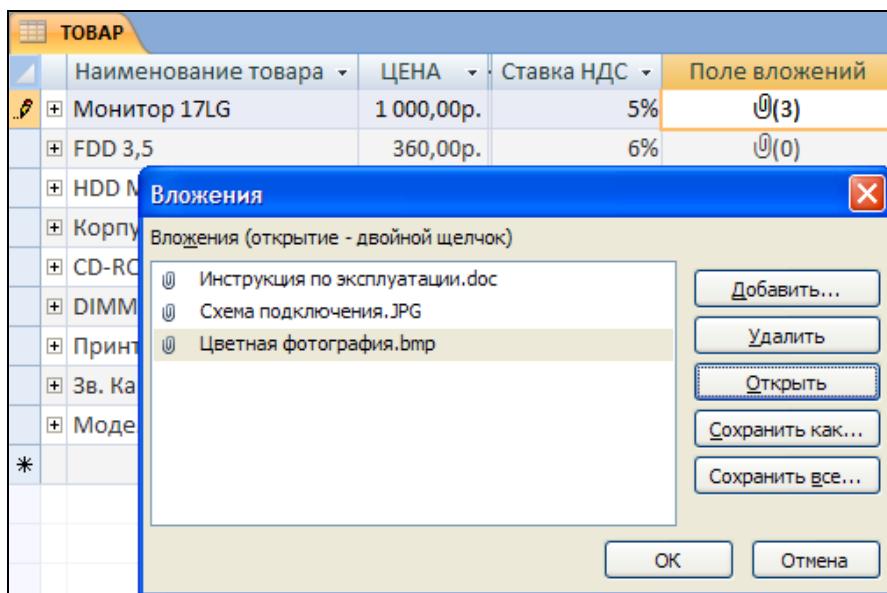
Добавим поле вложения в таблицу базы данных ТОВАР. В Access 2007/2010 имеются два способа добавить поле вложения в таблицу. Такое поле можно добавить в режиме таблицы или в режиме конструктора. В любом режиме для поля достаточно выбрать тип данных **Вложение** (Attachment). В режиме таблицы в строке заголовка поля вложения отображается значок скрепки, и в него нельзя ввести текст. В режиме конструктора можно изменить автоматически сформированное

имя поля **Поле1** (Fields1) и в свойстве поля ввести подпись, например **Поле вложений**, которая заменит значок скрепки в заголовке поля вложения.

### ЗАМЕЧАНИЕ

Преобразовать поле вложения в поле с другим типом данных невозможно. Такое поле можно только удалить.

Для добавления и изменения вложений служит диалоговое окно **Вложения** (Attachments) (рис. 3.29). Это диалоговое окно можно открыть двойным щелчком в поле вложения в таблице. Файлы вложений сохраняются в базе данных.



**Рис. 3.29.** Окно для управления файлами в поле вложений

На рис. 3.29 в окне **Вложения** (Attachments) перечислены добавленные файлы, а в поле вложений указано их число.

Вложения можно просматривать и редактировать, нажав в окне **Вложения** (Attachments) кнопку **Открыть** (Open). Для открытия вложения используются программы, в которых создавались эти файлы, или программы, поддерживающие файлы такого типа. Например, при открытии вложенного в таблицу рисунка BMP запускается программа просмотра изображений и факсов, при открытии документа Word запускается приложение Word и работа с документом происходит в этом приложении. Если приложение Word не установлено на компьютере, появится диалоговое окно с предложением выбрать программу для работы с файлом.

Для рисунка, открытого в программе просмотра, выполните в его контекстном меню команду **Изменить**. Запустится исходная программа, с помощью которой можно выполнить редактирование рисунка.

Отредактируйте открытый файл средствами открывшей его программы. Сохраните внесенные в файл изменения и завершите работу программы. При этом все изменения вложенного файла сохраняются во временной копии файла на жестком диске в папке временных файлов Интернета.

Чтобы сохранить изменения в базе данных, вернитесь в Access и в диалоговом окне **Вложения** (Attachments) нажмите кнопку **OK**. Окно закроется, и появится сообщение с предложением сохранить вложенный файл (рис. 3.30), нажмите кнопку **Да** (Yes).

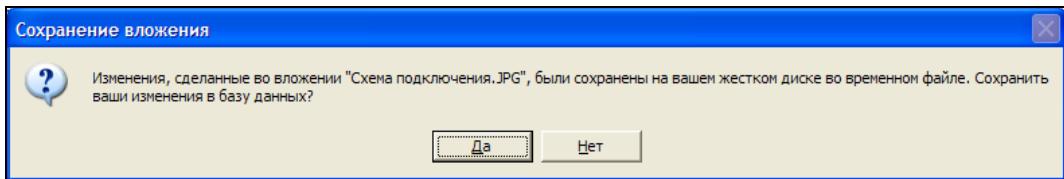


Рис. 3.30. Сообщение при завершении работы с вложениями

Любой из файлов или все файлы, вложенные в поле записи, можно сохранить на жестком диске или в сети. В диалоговом окне **Вложения** (Attachments) нажмите соответственно кнопку **Сохранить как** (Save As) или **Сохранить все** (Save All). В диалоговом окне **Сохранить вложения** (Save Attachment) выберите папку для нового расположения файла или всех файлов вложения и нажмите кнопку **Сохранить** (Save).

При использовании поля вложений в форме или отчете на элементе управления отображаются и могут просматриваться без использования дополнительного программного обеспечения вложения таких форматов графических файлов, как BMP, GIF, JPEG, JPG, TIFF, TIF и ряда других, непосредственно поддерживаемых Access. Для неграфических файлов на элементе управления отображается значок программы, создавшей файл. Для элемента управления в форме также как в таблице доступно окно **Вложения** (Attachments) со всеми его командами управления. В формах и отчетах для перехода по файлам и вызова окна **Вложения** (Attachments) используется прозрачная мини-панель инструментов , вызываемая щелчком мыши в элементе управления.

Как правило, в качестве вложений можно использовать файлы, созданные в любых программах Microsoft Office. Кроме того, вложениями могут служить файлы журнала (LOG), текстовые файлы (TEXT, TXT) и сжатые ZIP-файлы.

## Ввод логически связанных записей

Введем несколько логически взаимосвязанных записей в таблицы ДОГОВОР и ПОСТАВКА\_ПЛАН. Логическая связь этих таблиц обеспечивается полем НОМ\_ДОГ — номер договора, входящий в структуру обеих таблиц.

Объекты ДОГОВОР и ПОСТАВКА\_ПЛАН связаны одно-многозначными отношениями. Связь таких таблиц реализуется в схеме данных с помощью ключа главной таблицы ДОГОВОР. В подчиненной таблице это поле связи является в данном случае частью ее составного ключа. Пока не создана схема данных, в которой устанавливаются связи между таблицами и их свойства, система не может контролировать логическую взаимосвязь вводимых данных. Поэтому для получения целостной базы, в которой каждая запись подчиненной таблицы имеет логически связанную с ней главную запись, пользователю придется самому отслеживать взаимосвязи записей. При вводе подчиненной записи в таблицу ПОСТАВКА\_ПЛАН необходимо проверять наличие в главной таблице ДОГОВОР записи с ключом, значение которого совпадает со значением поля связи (внешнего ключа) в подчиненной записи. То есть при добавлении строки плана поставки для договора Д111 необходимо, чтобы договор с этим номером уже был представлен в таблице ДОГОВОР.

При непосредственном вводе записей во взаимосвязанные таблицы полезно отобразить на экране обе таблицы (рис. 3.31).

ДОГОВОР			
Номер договора	Дата заключения	Код покупателя	
Д111	05.01.2007	П001	
Д222	25.02.2007	П001	
Д333	11.01.2007	П002	
Д555			
Д777			
Д888			
Д999			
*			
Запись: 14 < 12 из 19 > Нет фильтра Поиск			

ПОСТАВКА_ПЛАН				
Номер договора	Код товара	Срок поставки	Количество	
Д111	T001	1	100	
Д111	T001	2	20	
Д111	T001	3	30	
Д111	T002	1	50	
Д111	T002	3	10	
Д111	T003	1	20	
Д222	T001	1	100	
Д222	T001	2	12	
Д222	T003	2	10	
Д222	T004	3	30	
Запись: 14 < 12 из 19 > Нет фильтра Поиск				

Рис. 3.31. Таблицы ДОГОВОР и ПОСТАВКА\_ПЛАН

Для удобного отображения открытых таблиц можно воспользоваться командой меню **Сверху вниз** (Tile Horizontally) или **Слева направо** (Tile Vertically) в списке кнопки **Перейти в другое окно** (Switch Windows) на ленте **Главная** (Home) в группе **Окно** (Window).

### Отображение записей подчиненных таблиц в главной таблице

В Access имеется возможность при просмотре главной таблицы отображать записи подчиненных таблиц, называемых в этом случае *подтаблицами*. Это позволяет

пользователю при добавлении, удалении и изменении записей осуществлять контроль корректности связей в отображаемой цепочке таблиц. Например, если отобразить в таблице ДОГОВОР подчиненную таблицу ПОСТАВКА\_ПЛАН, связь между которыми осуществляется по полю НОМ\_ДОГ (номер договора), то при заполнении таблицы ПОСТАВКА\_ПЛАН невозможно допустить ошибку в поле НОМ\_ДОГ, т. к. оно даже не отображается в подчиненной таблице. Однако это никаким образом не помогает при заполнении других полей связи таблицы ПОСТАВКА\_ПЛАН, таких как КОД\_ТОВ (код товара).

При просмотре таблицы, для которой определена подтаблица, отображается столбец со значками "плюс" в каждой записи (рис. 3.32). Достаточно щелкнуть на значке "плюс" (+) в строке записи, чтобы отобразились записи подчиненной таблицы, связанные с текущей записью. При этом плюс на значке преобразуется в минус (-). Щелчком на минусе подчиненные записи закрываются. Таким образом могут быть открыты подчиненные записи каждой записи главной таблицы. Если открыть все записи главной таблицы, вы увидите все записи подчиненной таблицы (при условии, что для связи обеспечивается целостность), разбитые на подмножества, связанные с конкретными записями главной таблицы.

The screenshot shows a Microsoft Access window with the main table 'ДОГОВОР' at the top. Below it are two subtables, both titled 'ПОСТАВКА\_ПЛАН'. The first subtable has records for 'Д111' (date 05.01.2007, buyer P001) and the second for 'Д222' (date 25.02.2007, buyer P001). Each subtable contains multiple rows of data for different products (T002, T003, T001) with their respective quantities. The interface includes standard Access navigation buttons at the bottom.

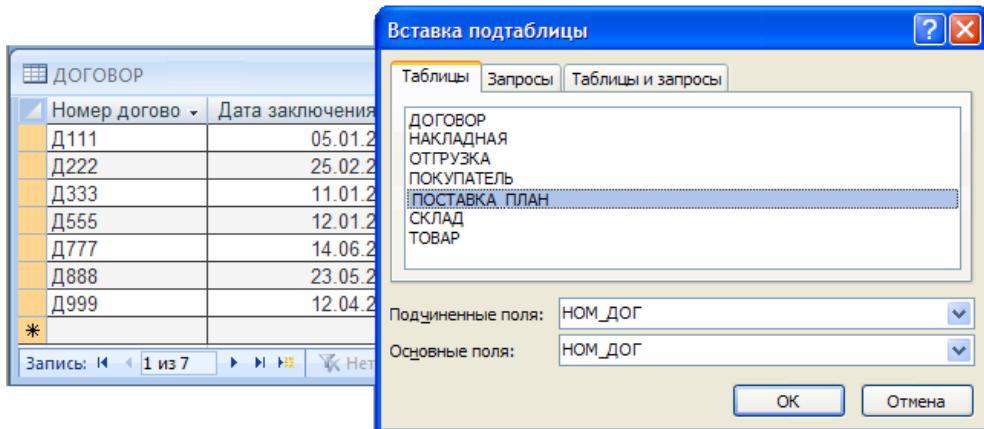
ДОГОВОР				
	Номер договора	Дата заключения	Код покупателя	Сум
	Д111	05.01.2007	П001	
	T002	1	10	50
	T002	3	5	10
	T003	1	4	20
	T001	1	10	100
	T001	2	5	20
	T001	3	5	30
*		0	0	0
	Д222	25.02.2007	П001	
	T003	2	5	10
	T004	3	15	30
	T001	1	10	100
	T001	2	4	12
*		0	0	0
[+]	Д333	11.01.2007	П002	
[+]	Д555	12.01.2007	П002	
[+]	Д777	14.06.2007	П001	

Запись: 5 из 7    Нет фильтра    Поиск

Рис. 3.32. Отображение подчиненных записей подтаблицы в главной таблице

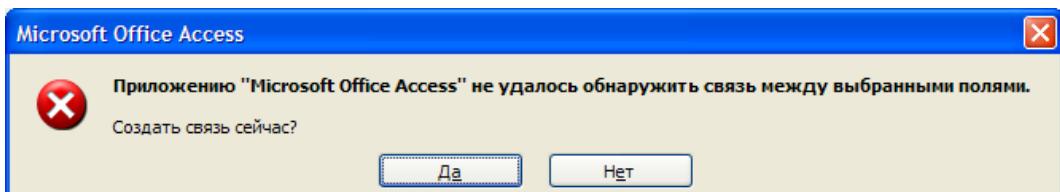
В таблице базы данных Access одновременно можно просматривать данные подтаблиц восьми уровней.

Для определения подтаблицы выполните команду **Подтаблица** (Subdatasheet) в списке кнопки **Дополнительно** (More) на вкладке ленты **Главная** (Home) в группе **Записи** (Records). Отобразится окно **Вставка подтаблицы** (Insert Subdatasheet) (рис. 3.33), в котором следует выбрать подчиненную таблицу или запрос и указать поля связи.



**Рис. 3.33.** Окно для выбора подчиненной таблицы и поля связи с ней

Если в главной и подчиненной таблицах поле связи имеет разные имена, то в окне **Вставка подтаблицы** (Insert Subdatasheet) можно ввести имена полей связи, при составном ключе имена полей разделяются точкой с запятой. Если между таблицами ТОВАР и ПОСТАВКА\_ПЛАН не было установлено связи в схеме данных, Access автоматически добавит эту связь после утвердительного ответа пользователя в окне диалога (рис. 3.34).

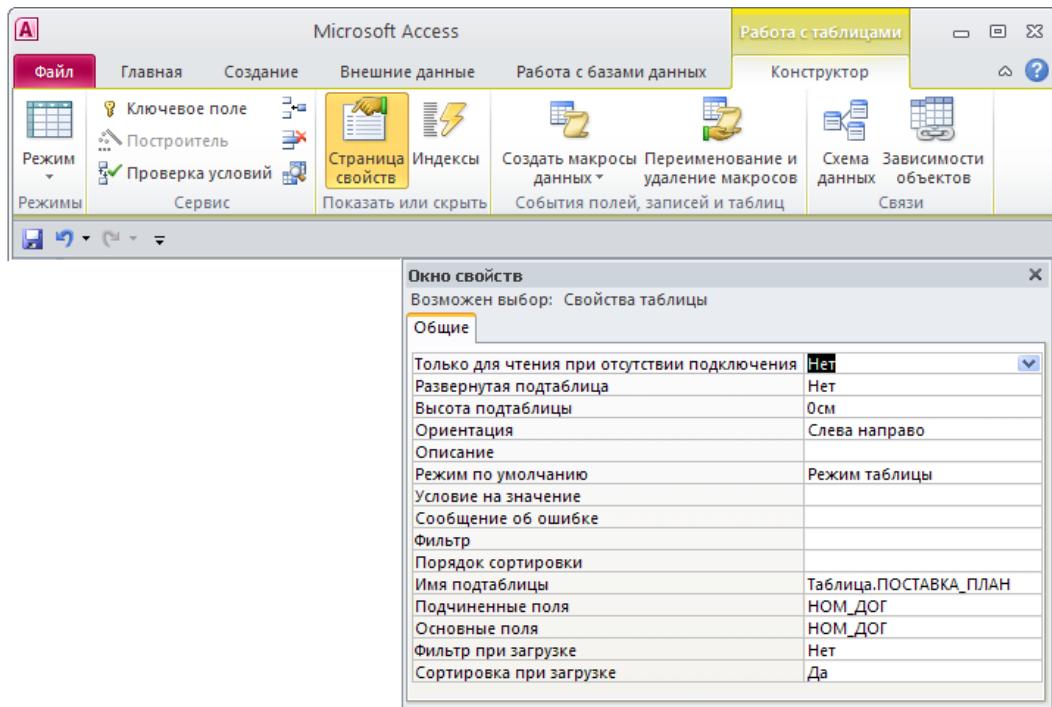


**Рис. 3.34.** Сообщение системы при определении подтаблицы

Открыть или закрыть все подчиненные записи можно, воспользовавшись командой **Подтаблица | Развернуть все** (Subdatasheet | Expand All) или **Свернуть все** (Collapse All) в списке кнопки **Дополнительно** (More) на ленте **Главная** (Home) в группе **Записи** (Records).

После сохранения таблицы имя подтаблицы и имена подчиненного и основного полей записываются в свойства главной таблицы.

Для просмотра свойств главной таблицы, например, ДОГОВОР, откройте ее в режиме конструктора и нажмите кнопку **Страница свойств** (Property Sheet). В строке свойства **Имя подтаблицы** (Subdatasheet Name) указано имя подчиненной таблицы, в свойствах **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields) указано имя ключа связи (простого или составного) (рис. 3.35).



**Рис. 3.35.** Свойства таблицы ДОГОВОР

В режиме конструктора таблиц можно определить другую подтаблицу, выбрав ее имя из списка в строке свойства **Имя подтаблицы** (Subdatasheet Name) и заменив значения в строках, определяющих связь с ней.

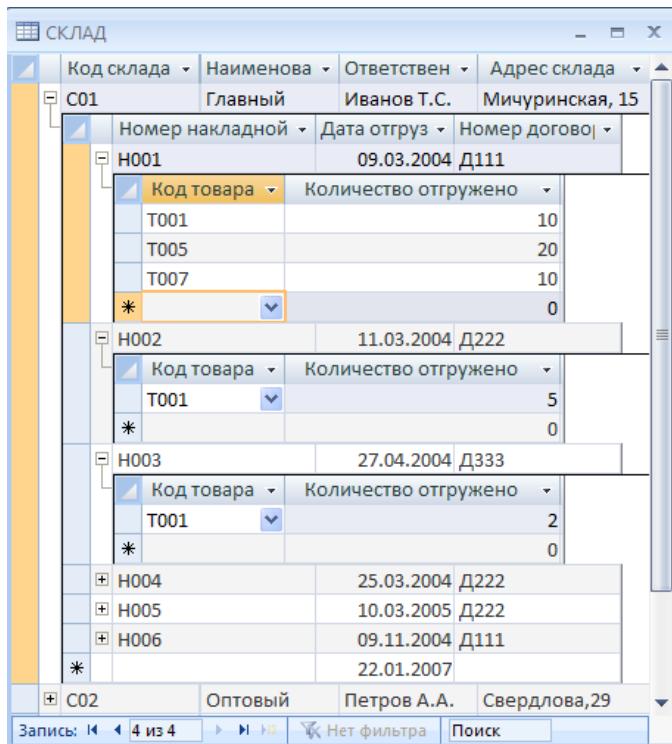
По умолчанию в строке свойства **Имя подтаблицы** (Subdatasheet Name) установлено значение **Авто** ([Auto]), которое означает, что если в схеме данных установлена связь таблиц, Access автоматически выполнит вывод столбца со значками "плюс" для открытия подчиненных записей по этой связи.

Если у таблицы имеются две или более подчиненных таблиц, определенных в схеме, то при щелчке на знаке "плюс" автоматически откроется окно для выбора подчиненной таблицы. После выбора подчиненной таблицы и сохранении главной таблицы эта связь фиксируется в свойствах.

Для того чтобы в таблице не выводился столбец, позволяющий открывать подчиненные записи, в качестве значения свойства **Имя подтаблицы** (Subdatasheet Name) должно быть установлено **Нет** (None). Значение **Нет** (None) можно устано-

вить в режиме таблицы с помощью команды **Подтаблица | Удалить** (Subdatasheet | Remove) в списке кнопки **Дополнительно** (More) на ленте **Главная** (Home) в группе **Записи** (Records).

В подчиненной таблице можно выполнять те же действия, что и в главной: работать с данными и отображать подчиненную более низкого уровня. На рис. 3.36 в таблице СКЛАД открыты подчиненные записи таблицы НАКЛАДНАЯ и в ней подчиненные записи таблицы ОТГРУЗКА.



**Рис. 3.36.** Отображение подтаблиц НАКЛАДНАЯ и ОТГРУЗКА в таблице СКЛАД

Несмотря на то, что в основном через механизм подтаблиц просматриваются связанные записи подчиненных таблиц, система позволяет устанавливать связь и просматривать связанную запись главной таблицы из подчиненной. Например, для подчиненной таблицы ОТГРУЗКА, представленной на предыдущем рисунке, можно установить связь с главной по отношению к ней таблицей ТОВАР, раскрывающей название товара и другие его характеристики (рис. 3.37).

### Задание 3.3. Отображение подтаблиц

Просмотрите подтаблицы всех уровней для таблиц ПОКУПАТЕЛЬ и ТОВАР.

**Рис. 3.37.** Отображение взаимосвязанных записей нескольких таблиц

## Использование поля со списком при вводе записей

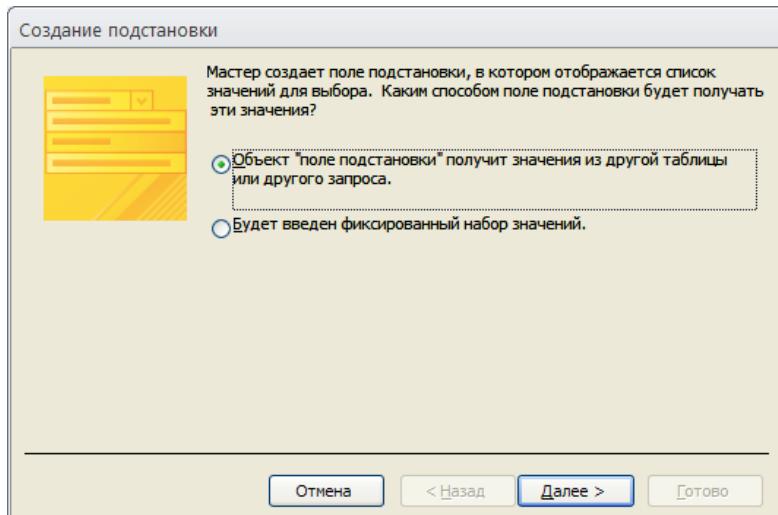
### Создание поля со списком в режиме таблицы

Задачу ввода подчиненных записей только при наличии связанной записи в главной таблице можно решить с помощью поля со списком. Для этого достаточно в подчиненной таблице преобразовать поле связи в поле со списком и в качестве источника строк использовать значения ключевого поля из главной таблицы.

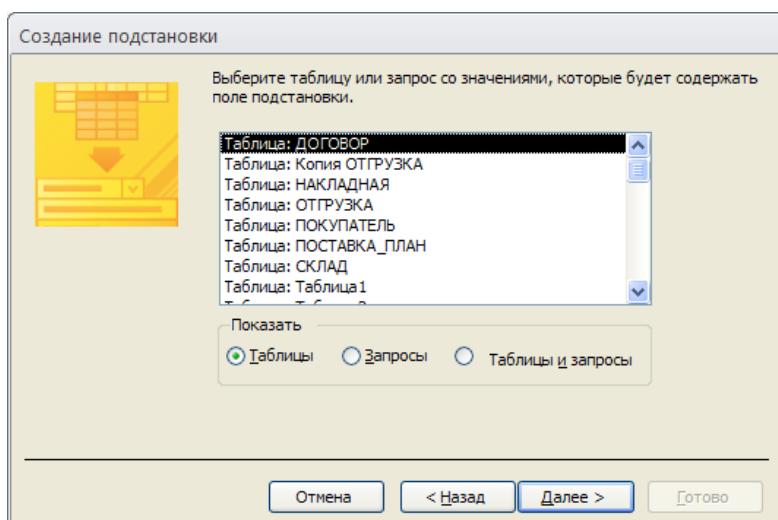
Создайте в таблице ПОСТАВКА\_ПЛАН для поля НОМ\_ДОГ (номер договора) поле со списком, построенным на основе ключа таблицы ДОГОВОР. Выберите таблицу ПОСТАВКА\_ПЛАН в области навигации и откройте ее. Таблица откроется в режиме таблицы и станет доступной вкладка ленты **Работа с таблицами | Поля** (Table Tools | Fields), на которой представлены команды изменения структуры таблицы, доступные в этом режиме (см. рис. 3.10).

Процесс создания поля со списком в режиме таблицы начните с выполнения команды **Подстановка и отношения** (Lookup & Relationship) в списке кнопки **Другие поля** (More Fields) в группу **Добавление и удаление** (Add & Delete) на ленте **Работа с таблицами | Поля** (Table Tools | Fields). При этом автоматически запускается мастер подстановок.

В открывшемся окне мастера **Создание подстановки** (Lookup Wizard) установите тот переключатель, который приведет к построению списка на основе значений из таблицы (рис. 3.38).



**Рис. 3.38.** Первое окно Мастера подстановок



**Рис. 3.39.** Выбор таблицы, из данных которой будет создаваться список

В следующем окне мастера выберите таблицу ДОГОВОР, на значениях которой будет строиться список поля (рис. 3.39).

Далее выберите столбец, из данных которого будет формироваться список (рис. 3.40).

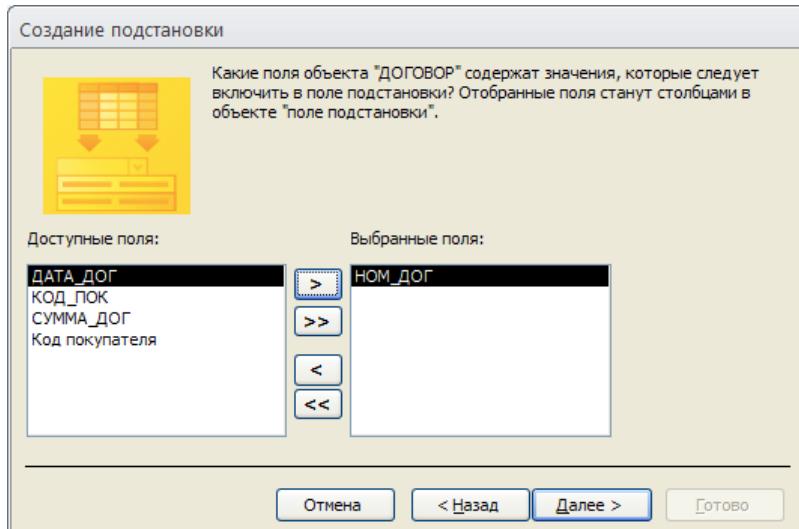


Рис. 3.40. Выбор поля, используемого для формирования списка

В следующих окнах мастера выберите порядок сортировки элементов списка и нужную ширину столбца списка.

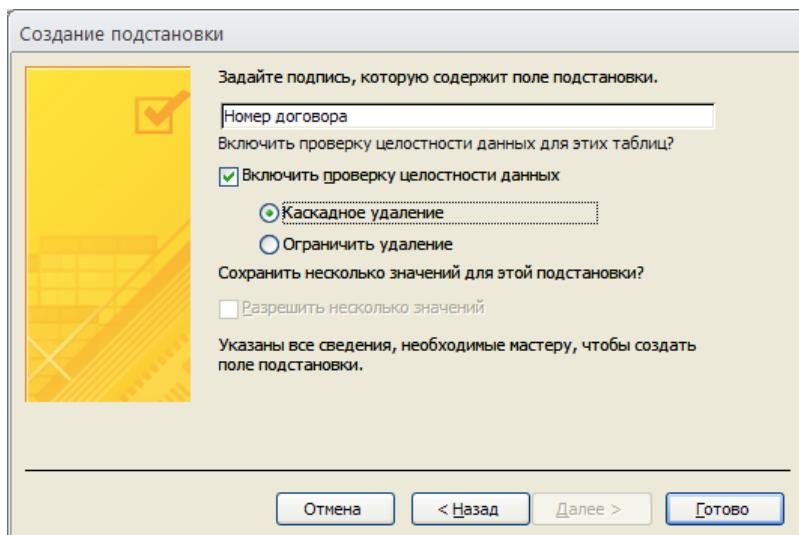


Рис. 3.41. Определение имени поля и проверки связной целостности данных

Далее мастер предлагает задать имя поля со списком и параметры поддержания целостности для связи таблиц (рис. 3.41) (в нашем примере таблиц ПОСТАВКА\_ПЛАН и ДОГОВОР). Связь таблиц сохранится в схеме данных. Для того чтобы увидеть эту связь, нужно открыть схему данных (команда на ленте **Работа с таблицами | Таблица | Связи** (Table Tools | Table | Relationships)) и выполнить команду контекстного меню **Отобразить все** (Show All).

На этом создание мастером нового поля со списком завершается.

Теперь при вводе данных в таблицу ПОСТАВКА\_ПЛАН можно воспользоваться списком, в котором отобразятся все значения поля НОМ\_ДОГ, содержащиеся в таблице ДОГОВОР (рис. 3.42). Выбором нужного значения из списка осуществляется ввод значения в поле номер договора.

The screenshot shows the Microsoft Access Table View for the 'ПОСТАВКА\_ПЛАН' table. The first column is labeled 'Номер договора' (Contract Number). A dropdown arrow is visible next to the cell for the second row, indicating a list of values. The list is displayed below the table, showing values such as 'Д111', 'Д222', 'Д333', etc. The value 'Д222' is currently selected in the dropdown list.

Номер договора	Код товара	Срок поставки
Д111	T001	2
Д111	T001	3
Д111	T002	1
Д111	T002	3
Д111	T003	1
Д222	T001	1
Д222	T001	2
Д222	T003	2
Д111	T004	3
Зап...		
Д222		
Д333		
Д555		
Д777		
Д888		
Д999		

Рис. 3.42. Ввод значения в поле путем выбора из списка

Если между таблицами не было установлено параметра обеспечения целостности связи таблиц, в поле со списком можно вводить значения, не предусмотренные в списке.

В режиме таблицы предоставляется возможность добавить новое поле с построением для него списка и установлением соответствующего отношения между таблицами и нельзя изменить существующее поле на поле со списком.

Создание поля со списком в режиме таблицы целесообразно при первоначальной разработке структуры таблицы в этом режиме. В противном случае такое поле как НОМ\_ДОГ, являющееся полем связи с таблицей ДОГОВОР, уже будет создано в таблице, и после создания нового поля со списком его надо будет удалить. В приведенном примере это можно сделать только в режиме конструктора, т. к. поле НОМ\_ДОГ является частью первичного ключа таблицы. В режиме конструктора поле может быть преобразовано в поле со списком и после разработки структуры таблицы.

Если переключиться в режим конструктора, то можно увидеть параметры списка, подготовленные мастером в свойствах нового поля на вкладке **Подстановка** (Lookup) (см. рис. 3.43).

## Создание поля со списком в режиме конструктора

Выберите таблицу ПОСТАВКА\_ПЛАН в области навигации и, в контекстном меню таблицы, выполните команду **Конструктор** (Design View). Установите курсор в поле НОМ\_ДОГ на столбце **Тип данных** (Data Type), откройте список и выберите строку **Мастер подстановок...** (Lookup Wizard...). Откроется окно мастера **Создание подстановки** (Create Lookup). Работа с мастером, рассмотренная в предыдущем разделе, приведет к преобразованию поля НОМ\_ДОГ в поле со списком.

Параметры списка, подготовленные мастером, отобразятся в свойствах поля НОМ\_ДОГ на вкладке **Подстановка** (Lookup) (рис. 3.43).

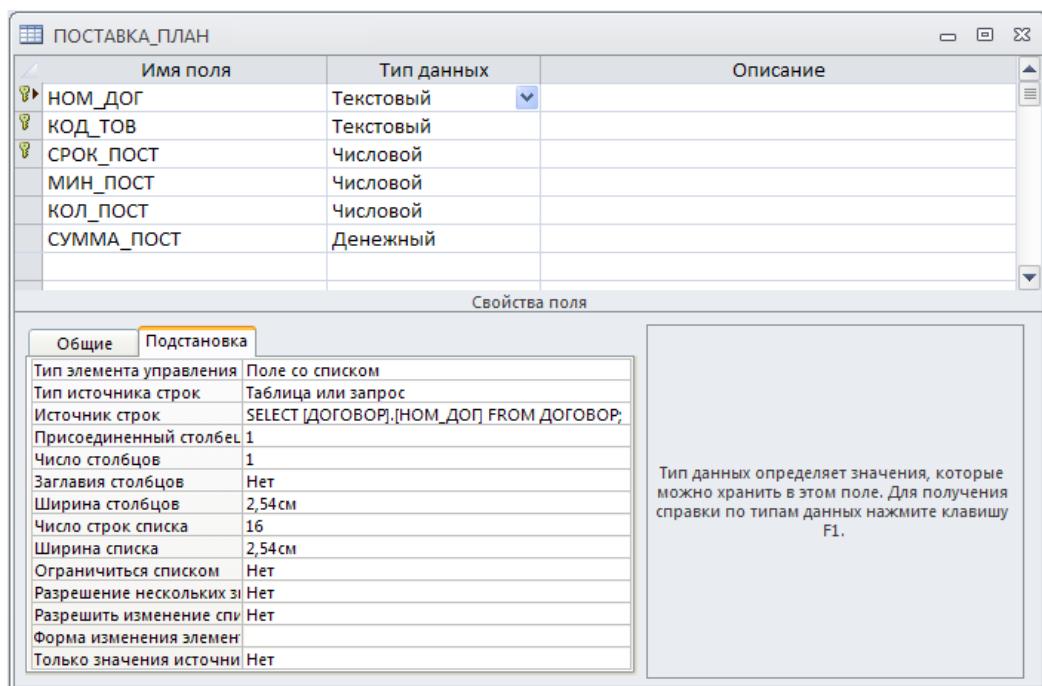


Рис. 3.43. Свойства поля со списком для выбора номера договора

Теперь при вводе данных в таблицу ПОСТАВКА\_ПЛАН можно воспользоваться созданным списком (см. рис. 3.42).

Списком поля ограничивается ввод в поле НОМ\_ДОГ значений, не указанных в списке, только в случае если при работе мастера или в схеме вручную были пропущены параметры обеспечения целостности связи таблиц. Однако, если при определении параметров списка на вкладке **Подстановка** (Lookup) в окне конст-

руктора таблицы (см. рис. 3.43) выбрать в строке **Ограничиться списком** (Limit to List) значение **Да** (Yes), ввод значений, не предусмотренных в списке, сделается в любом случае невозможным. В случае отклонения от значений списка система выдаст сообщение, показанное на рис. 3.44. Таким образом, пользователь вынужден будет придерживаться только значений, представленных в списке, и в подчиненную таблицу будут введены только связанные записи.

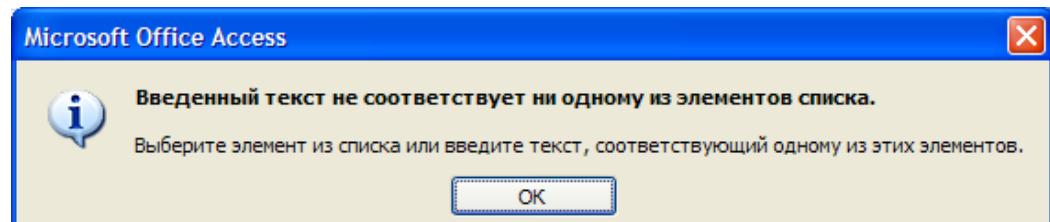


Рис. 3.44. Сообщение о неверно введенном значении

Таким образом, использование списков не только обеспечивает удобный ввод, но позволяет избежать многих ошибок.

#### Задание 3.4. Создание поля со списком для существующего поля

Преобразуйте в таблице ПОСТАВКА\_ПЛАН поле КОД\_ТОВ (код товара) в поле со списком. Для этого откройте таблицу в режиме конструктора, установите курсор в поле КОД\_ТОВ на столбце **Тип данных** (Data Type) и в списке этого столбца выберите строку **Мастер подстановок...** (Lookup Wizard...). В диалоге с мастером выберите главную по отношению к таблице ПОСТАВКА\_ПЛАН таблицу ТОВАР, на основе данных которой создается список, и поля, включаемые в список: КОД\_ТОВ — код товара и НАИМ\_ТОВ — наименование товара. Кроме того, оставьте, как предлагает мастер, помеченный флажок **Скрыть ключевой столбец (рекомендуется)** (Hide key column (recommended)). Результат работы мастера должен получиться как на рис. 3.45.

При выборе параметра **Скрыть ключевой столбец (рекомендуется)** (Hide key column (recommended)) для первого столбца списка устанавливается ширина, равная нулю. При этом параметр **Ограничиться списком** (Limit to List) может принимать только одно значение — **Да** (Yes) (рис. 3.46).

Определение поля со списком с такими параметрами приводит к отображению в поле КОД\_ТОВ вместо значений ключевого поля значений второго поля списка — поля НАИМ\_ТОВ (наименование товара), взятого из главной таблицы ТОВАР. Тем не менее, выбор наименования товара приведет к вводу соответствующего значения ключа в поле КОД\_ТОВ. На рис. 3.47 показано, как в таблице ПОСТАВКА\_ПЛАН отображается поле со списком **Код товара** при нулевой ширине присоединенного столбца.

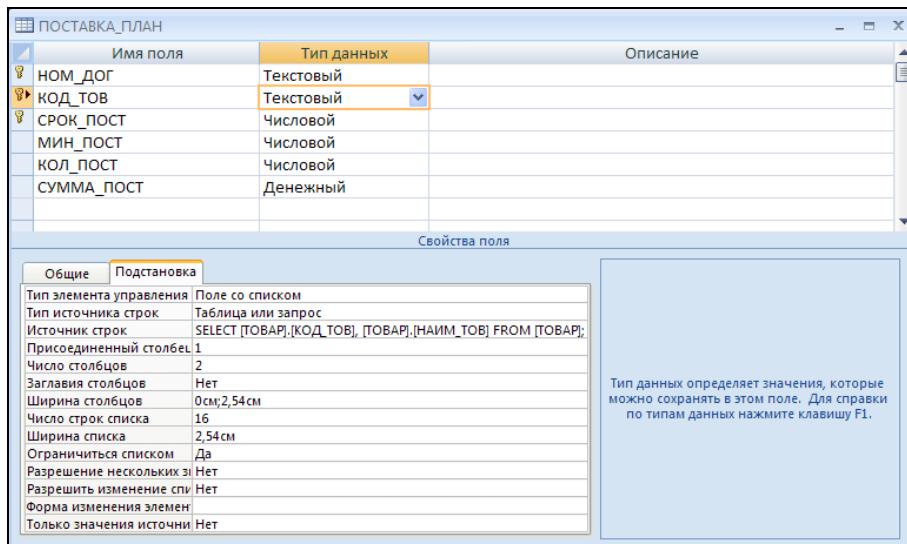


Рис. 3.45. Свойства поля со списком, созданного мастером подстановок, для кода товара

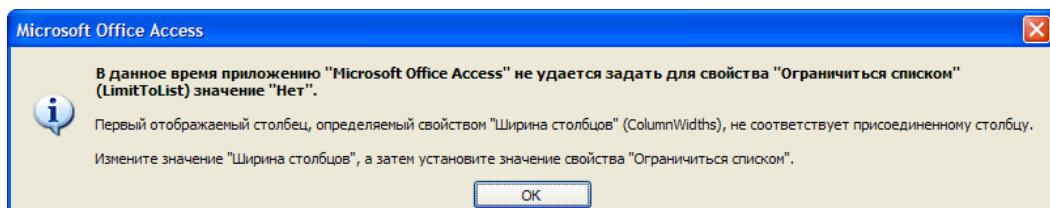


Рис. 3.46. Сообщение при попытке изменить свойство Ограничиться списком

ПОСТАВКА_ПЛАН				
Номер договора	Код товара	Срок поставки	Количество	
Д111	Монитор 17LG		1	100
Д111	Монитор 17LG		2	20
Д111	Монитор 17LG		3	30
Д111	FDD 3,5		1	50
Д111	Монитор 17LG		3	10
Запись: 14 4 из 19 >				
Поиск				

A dropdown menu is open over the 'Код товара' column, showing a list of items starting with 'FDD 3,5'.

Рис. 3.47. Отображение поля со списком Код товара при нулевой ширине присоединенного столбца

Таким образом, использование полей со списком не только упрощает ввод данных, но и помогает избежать ошибок при вводе связанных записей в таблицы как за счет того, что в таблицах вместо ключей могут быть отображены смысловые значения, так и за счет того, что ввод ограничивается набором значений в списке.

### **ОБРАТИТЕ ВНИМАНИЕ!**

При создании полей со списком мастер подстановки автоматически формирует связь между таблицей, в которой создается поле со списком, и таблицей, являющейся источником значений списка. Эту связь можно увидеть в **Схеме данных** (Relationships), которая открывается выполнением команды **Схема данных** (Relationships) в группе **Отношения** (Relationships) на вкладке ленты **Работа с базами данных** (Database Tools).

#### **Задание 3.5. Создание полей со списком**

Создайте поле со списком для отображения в таблице ДОГОВОР наименований покупателей, в таблице ОТГРУЗКА — кодов и наименований товаров, в таблице НАКЛАДНАЯ — кодов и наименований складов.

## **Схема данных в Access**

Реляционная база данных, созданная в соответствии с проектом канонической модели данных предметной области, состоит из нормализованных таблиц, связанных одно-многозначными отношениями. В такой базе данных обеспечивается отсутствие дублирования описательных данных, их однократный ввод, поддержание целостности данных средствами системы. Связи между таблицами позволяют выполнить объединение данных различных таблиц, необходимое для решения большинства задач ввода, просмотра и корректировки данных, получения информации по запросам и вывода отчетов.

Связи между таблицами устанавливаются в соответствии с проектом логической структуры базы данных (см. рис. 2.19) и запоминаются в схеме данных Access. Схема данных является не только средством графического отображения логической структуры базы данных, она активно используется системой в процессе обработки данных. Благодаря связям, установленным в схеме данных, разработчику нет необходимости всякий раз сообщать системе о наличии той или иной связи. Однажды указанные в схеме данных связи используются системой автоматически.

Создание схемы данных позволяет упростить конструирование многотабличных форм, запросов, отчетов, а также обеспечить поддержание целостности взаимосвязанных данных при вводе и корректировке данных в таблицах.

## **Создание схемы данных**

Рассмотрим процесс создания схемы данных соответствующей логической структуре базы данных *Поставка товаров*, разработка которой рассмотрена в главе 2.

Создание схемы данных начинается с выполнения команды **Схема данных** (Relationships) в группе **Отношения** (Relationships) на вкладке ленты **Работа с базами данных** (Database Tools). В результате выполнения этой команды открывается окно схемы данных и диалоговое окно **Добавление таблицы** (Show Table), в котором осуществляется выбор таблиц, включаемых в схему (см. рис. 3.48). Диалоговое окно **Добавление таблицы** откроется автоматически, если в базе данных еще не определена ни одна связь. Если окно не открылось, на ленте **Работа со связями | Конструктор** (Relationship Tools | Design) в группе **Связи** (Relationships) нажмите кнопку **Отобразить таблицу** (Show Table).

## Включение таблиц в схему данных

В окне **Добавление таблицы** (Show Table) (рис. 3.48) отображены все таблицы и запросы, содержащиеся в базе данных. Выберем вкладку **Таблицы** (Tables) и с помощью кнопки **Добавить** (Add) разместим в окне **Схема данных** (Relationships) все ранее созданные таблицы базы данных *Поставка товаров*, отображенные в окне **Добавление таблицы** (Show Table). Затем нажмем кнопку **Закрыть** (Close). В результате в окне **Схема данных** (Relationships) таблицы базы будут представлены окнами со списками своих полей и выделенными жирным шрифтом ключами (см. рис. 3.52).

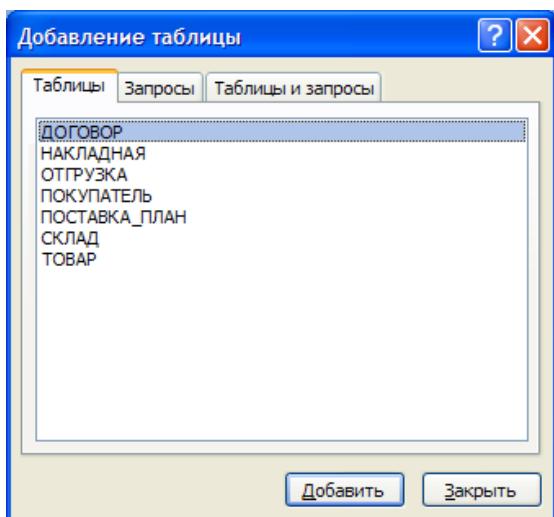


Рис. 3.48. Добавление таблиц в схему данных

Теперь можно приступить к созданию связей между таблицами.

## Создание связей между таблицами схемы данных

При создании связей в схеме данных используется проект логической структуры реляционной базы данных, в котором показаны все одно-многозначные связи таблиц. Реализуются связи с помощью добавления в связанные таблицы общих по-

лей, называемых *ключом связи*. При одно-многозначных отношениях между таблицами ключом связи является ключ главной таблицы (простой или составной). В подчиненной таблице он может быть частью уникального ключа или вовсе не входить в состав ключа таблицы. Одно-многозначные связи являются основными в реляционных базах данных. Одно-однозначные связи используются, как правило, при необходимости распределять большое количество полей, определяемых одним и тем же ключом, по разным таблицам, имеющим разный регламент обслуживания.

### Создание связей по простому ключу

Установим связь между таблицами ПОКУПАТЕЛЬ и ДОГОВОР, которые находятся в отношении "один-ко-многим". Устанавливая связи между парой таблиц, находящихся в отношении типа  $1 : M$ , выделим в главной таблице ПОКУПАТЕЛЬ ключевое поле КОД\_ПОК, по которому устанавливается связь. Далее при нажатой кнопке мыши перетащим его в соответствующее поле подчиненной таблицы ДОГОВОР.

Поскольку поле связи является уникальным ключом в главной таблице связи, а в подчиненной таблице связи не является ключевым, Access выявляет отношение "один-ко-многим" между записями этих таблиц. Значение "один-ко-многим" (One-To-Many) отобразится в окне **Изменение связей** (Edit Relationships) в строке **Тип отношения** (Relationship Type) (рис. 3.49).

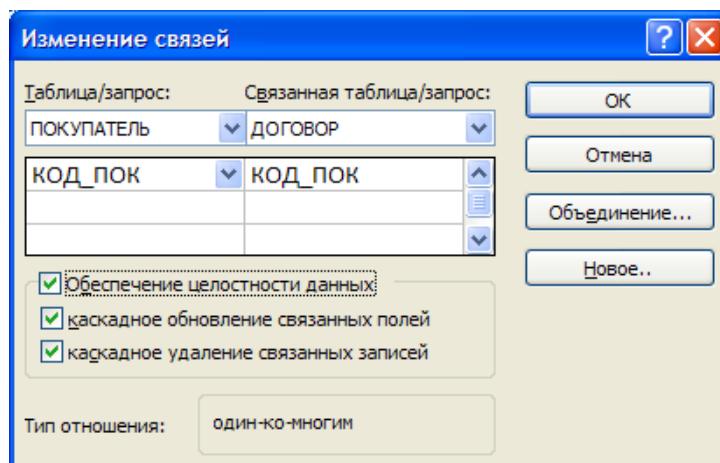


Рис. 3.49. Окно определения параметров связи

### ЗАМЕЧАНИЕ

Если поле связи является уникальным ключом в обеих связываемых таблицах, Access выявляет отношение "один-к-одному". Если для связи таблиц вместо ключевого поля главной таблицы используется некоторый уникальный индекс, система также констатирует отношение таблиц как  $1 : M$  или  $1 : 1$ .

### Задание 3.6. Создание связей по простому ключу

Добавьте в схему данных и установите связь по простому ключу для других пар таблиц базы данных *Поставка товаров*: СКЛАД → НАКЛАДНАЯ (ключ КОД\_СК), ДОГОВОР → ПОСТАВКА\_ПЛАН (ключ НОМ\_ДОГ), ТОВАР → ОТГРУЗКА (ключ КОД\_ТОВ), ТОВАР → ПОСТАВКА\_ПЛАН (ключ КОД\_ТОВ), ДОГОВОР → НАКЛАДНАЯ (ключ НОМ\_ДОГ).

#### **ЗАМЕЧАНИЕ**

Кнопка **Новое** (Create New) в окне **Изменение связей** (Edit Relationships) позволяет перейти к созданию связи между любыми двумя таблицами базы, не выходя в окно схемы данных.

### Определение связей по составному ключу

Определим связи между таблицами НАКЛАДНАЯ → ОТГРУЗКА, которые связаны по составному ключу НОМ\_НАКЛ + КОД\_СК. Для этого в главной таблице НАКЛАДНАЯ выделим оба этих поля, нажав клавишу <Ctrl>, и перетащим их в подчиненную таблицу ОТГРУЗКА.

В окне **Изменение связей** (Edit Relationships) (рис. 3.50) для каждого поля составного ключа главной таблицы НАКЛАДНАЯ, названной **Таблица/запрос** (Table/Query), выберем соответствующее поле подчиненной таблицы ОТГРУЗКА, названной **Связанная таблица/запрос** (Related Table/Query).

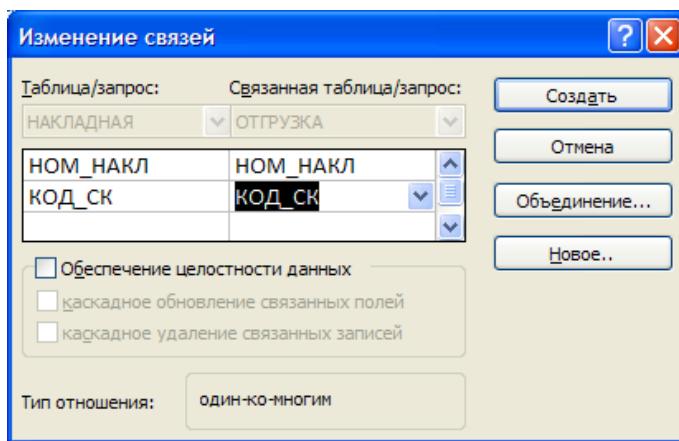


Рис. 3.50. Окно выбора параметров **Изменение связей**

### Обеспечение целостности данных

Если связываемые в схеме данных таблицы находятся в отношении 1 : 1 и 1 : M, для связи можно задать параметр обеспечения связной целостности данных.

Обеспечение связной целостности данных означает, что Access при корректировке базы данных реализует для связанных таблиц контроль соблюдения следующих условий:

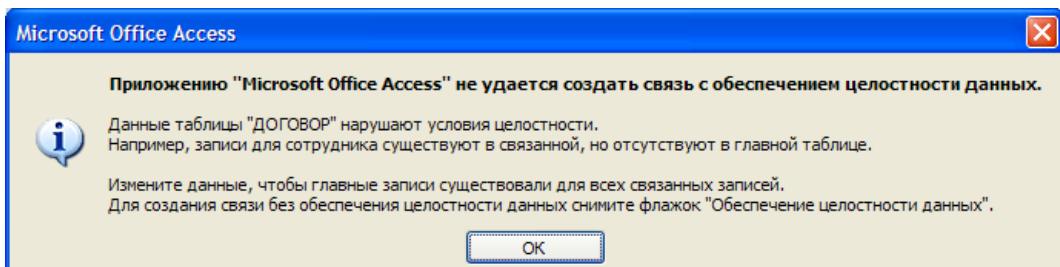
- ❖ в подчиненную таблицу не может быть добавлена запись с несуществующим в записях главной таблицы значением ключа связи;
- ❖ в главной таблице нельзя удалить запись, если не удалены связанные с ней записи в подчиненной таблице;
- ❖ изменение значений ключа связи в записи главной таблицы невозможно, если подчиненная таблица имеет связанные с ней записи, в которых соответственно не меняется ключ связи.

При попытке пользователя нарушить эти условия в операциях добавления и удаления записей или обновления ключевых данных в связанных таблицах Access не допускает выполнения операции и выводит соответствующее сообщение. Access не позволяет установить параметр целостности для связи таблиц, если ранее введенные в таблицы данные не отвечают требованиям целостности.

Установление между двумя таблицами связи и задание для нее параметров целостности данных возможно только при следующих условиях:

- ❖ связываемые поля имеют одинаковый тип данных, причем имена полей могут быть различными;
- ❖ обе таблицы сохраняются в одной базе данных Access;
- ❖ главная таблица связывается с подчиненной по первичному простому или составному ключу или уникальному индексу главной таблицы.

Для каждой из ранее установленных связей базы данных *Поставка товаров* установим в окне **Изменение связей** (Edit Relationships) флагок **Обеспечение целостности данных** (Enforce Referential Integrity). Установление связи с параметром обеспечения целостности данных возможно только в том случае, если таблицы ранее были заполнены корректными данными. В противном случае появится сообщение о невозможности установить связь. Например, если эти условия целостности не соблюдаются для ранее загруженных данных таблиц ПОКУПАТЕЛЬ и ДОГОВОР, то при попытке установить параметр обеспечения целостности выдается сообщение, показанное на рис. 3.51.



**Рис. 3.51.** Сообщение о невозможности установить параметр целостности для связи

## Каскадное обновление и удаление связанных записей

Если для выбранной связи обеспечивается поддержание целостности, можно задать режим **каскадного удаления связанных записей** и режим **каскадного обновления связанных полей**. Такие параметры делают возможным в главной таблице, соответственно, удаление записей и изменение значения в ключевом поле, т. к. при этих параметрах система автоматически выполнит необходимые изменения в подчиненных таблицах, обеспечив сохранение свойств целостности базы данных.

В режиме **каскадного удаления связанных записей** при удалении записи из главной таблицы будут автоматически удаляться все связанные записи в подчиненных таблицах. При удалении записи из главной таблицы выполняется каскадное удаление подчиненных записей на всех уровнях, если этот режим задан на каждом уровне.

В режиме **каскадного обновления связанных полей** при изменении значения ключевого поля в записи главной таблицы Access автоматически изменит значения в соответствующем поле в подчиненных записях.

Установить в окне **Изменение связей** (Edit Relationships) (см. рис. 3.49) флажки **каскадное обновление связанных полей** (Cascade Update Related Fields) и **каскадное удаление связанных записей** (Cascade Delete Related Records) можно только после задания параметра обеспечения целостности данных.

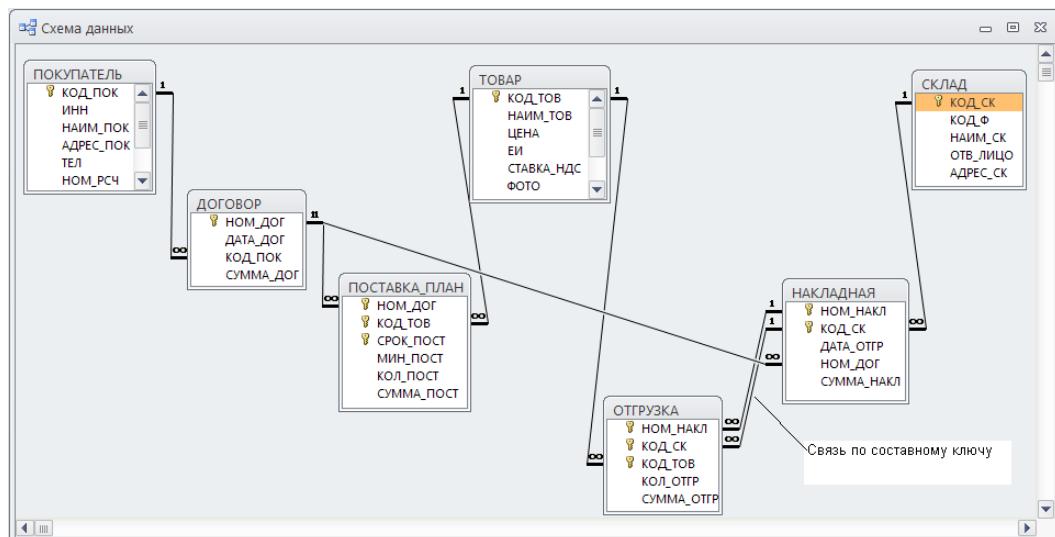


Рис. 3.52. Схема данных БД *Поставка товаров*

После создания связей изображения таблиц могут перемещаться в пределах рабочего пространства окна схемы данных. Перемещения и изменения размеров окон со списками полей таблиц в окне схемы данных осуществляются принятыми в Windows способами. На рис. 3.52 показана схема базы данных *Поставка товаров*, где таблицы размещены в соответствии с размещением информационных объектов

канонической модели данных, полученной при проектировании базы данных в *главе 2* (см. рис. 2.18).

На рис. 3.52 в созданной схеме данных БД *Поставка товаров* все связи отмечены символами **1** и **∞**. Это свидетельствует о том, что одно-многозначные связи установлены правильно (по простому или составному ключу) и для них задан параметр обеспечения целостности данных.

### Задание 3.7. Проверка поддержания целостности в базе данных

Если в схеме данных определена связь таблиц, и для нее установлены параметры обеспечения целостности, при вводе и корректировке данных во взаимосвязанных таблицах пользователь не сможет ввести записи, нарушающие требования связной целостности, рассмотренные в предыдущем разделе. Проверьте, как обеспечивается поддержание целостности при внесении изменений в таблицы ПОКУПАТЕЛЬ — ДОГОВОР, связанные одно-многозначными отношениями.

**Проверка автоматического поддержания целостности при изменении значений ключей связи в таблицах.** Откройте главную таблицу связи ПОКУПАТЕЛЬ в режиме таблицы. Измените значение ключевого поля КОД\_ПОК (код покупателя) в одной из записей. Убедитесь, что во всех записях подчиненной таблицы ДОГОВОР для договоров, заключенных этим покупателем, автоматически также изменится значение поля КОД\_ПОК. Изменение происходит, т. к. был установлен флагок **каскадное обновление связанных полей** (Cascade Update Related Fields) (см. рис. 3.49). Причем это изменение осуществляется мгновенно, как только изменяется запись перестает быть текущей.

**Проверка при добавлении записей в подчиненную таблицу.** Убедитесь, что невозможно включить новую запись в подчиненную таблицу ДОГОВОР со значением ключа связи КОД\_ПОК, не представленным в таблице ПОКУПАТЕЛЬ. Измените значение ключа связи КОД\_ПОК в подчиненной таблице ДОГОВОР на значение, не существующее в записях таблицы ПОКУПАТЕЛЬ, и убедитесь, что такое изменение запрещено, т. к. при поддержании целостности не может существовать запись подчиненной таблицы с ключом связи, которого нет в главной таблице.

**Проверка при удалении записи в главной таблице.** Убедитесь, что вместе с удалением записи в главной таблице ПОКУПАТЕЛЬ удаляются все подчиненные записи в таблице ДОГОВОР, т. к. был установлен флагок **каскадное удаление связанных записей** (Cascade Delete Related Records).

Заметим, если каскадное удаление не разрешено, невозможно удалить запись в главной таблице, если имеются связанные с ней записи в подчиненной.

## Контрольные вопросы

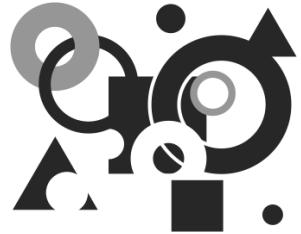
1. В файле какого типа сохраняется база данных Access 2010?
2. Какие объекты представлены в области навигации базы данных?
3. Чем определяется состав полей таблицы?

4. Чем определяется структура таблицы?
5. В каком режиме наиболее полно определяются параметры структуры таблицы?
6. В каком свойстве поля задаются ограничения на его значения?
7. На что влияет свойство **Формат поля** (Format)?
8. Что формирует система на основе заданного первичного ключа?
9. Для чего служат индексы таблицы?
10. Значение какого свойства поля используется при формировании заголовка столбца таблицы?
11. Может ли ключ иметь повторяющиеся значения?
12. Можно ли в строке таблицы не вводить значение ключа?
13. Могут ли в таблице повторяться значения в одном из полей составного ключа?
14. В каком режиме осуществляется ввод данных в таблицу?
15. Назовите кнопки перехода по записям таблицы.
16. В каком режиме визуально выполняется настройка макета таблицы?
17. С помощью какой команды выполняется внедрение объекта в поле с типом данных OLE?
18. Для чего создается схема данных базы Access?
19. В каких отношениях должны находиться таблицы, чтобы для них можно было установить параметры поддержания связной целостности данных?
20. По какому полю должна быть установлена связь между таблицами, чтобы появилась возможность установить параметры поддержания связной целостности данных?
21. Можно ли в подчиненную таблицу ДОГОВОР ввести запись о договоре с покупателем, который не представлен в таблице ПОКУПАТЕЛЬ, если для этих таблиц обеспечивается целостность данных?
22. Можно ли удалить запись о покупателе, если в таблице ДОГОВОР представлены записи о договорах с этим покупателем и не установлен флагок **каскадное удаление связанных записей** (Cascade Delete Related Records)?
23. Что произойдет при изменении значения ключевого поля в главной таблице, если для ее связи с подчиненной установлен флагок **каскадное обновление связанных полей** (Cascade Update Related Fields)?
24. Для чего предназначен значок + (плюс) в левом столбце открытой таблицы?
25. Какая команда позволяет открыть в главной таблице связанные записи нужной подчиненной таблицы?
26. Какое свойство таблицы определяет подтаблицу, из которой выводятся связанные записи при щелчке на значке + (плюс) в левом столбце открытой таблицы?

## Ответы

1. ACCDB.
2. Таблицы, запросы, формы, отчеты, макросы и модули.

3. Реквизитным составом информационного объекта (сущности) информационно-логической модели предметной области.
4. Составом ее полей, для которых заданы имя, тип данных и свойства.
5. В режиме конструктора.
6. **Условие на значение** (Validation Rule).
7. На вид отображаемого данного.
8. Уникальный индекс с именем **PrimaryKey**.
9. Для более быстрого поиска записи (записей) по заданному значению индексного поля (поля).
10. **Подпись** (Caption).
11. Нет.
12. Нет.
13. Да.
14. В режиме таблицы.
15. **Первая запись** (First Record), **Предыдущая запись** (Previous Record), **Следующая запись** (Next Record), **Последняя запись** (Last Record), **Новая (пустая) запись** (New (blank) Record).
16. В режиме таблицы.
17. С помощью команды контекстного меню **Вставить объект** (Insert Object).
18. Схема данных наглядно отображает связи таблиц базы. Установленные в схеме связи автоматически используются при обработке связанных таблиц. В схеме данных можно запросить проверку связной целостности данных, задать способ объединения записей.
19. 1 : M или 1 : 1.
20. По первичному (простому или составному) ключу главной таблицы.
21. Нет.
22. Нет.
23. Обновятся значения связанных полей во всех подчиненных записях.
24. Для открытия записей выбранной подтаблицы.
25. **Подтаблица** (Subdatasheet), размещенная в списке кнопки **Дополнительно** (More) на вкладке ленты **Главная** (Home) в группе **Записи** (Records).
26. **Имя подтаблицы** (Subdatasheet Name).



## ГЛАВА 4

# Запросы

Запросы являются основным инструментом выборки, обновления и обработки данных в таблицах базы данных.

Access в соответствии с концепцией реляционных баз данных для выполнения запросов использует язык структурированных запросов SQL (Structured Query Language). С помощью инструкций языка SQL реализуется любой запрос в Access.

В то же время Access позволяет создавать запросы, не прибегая к записи инструкций языка SQL. Простейшие запросы могут быть созданы с помощью мастера, практически любой запрос можно создать в режиме графического конструктора. При создании запроса этими средствами Access сам автоматически создает эквивалентную инструкцию SQL, которую можно увидеть, переключившись в режим SQL. Конструктор позволяет создавать запросы простым и удобным способом, а просмотр этих запросов в режиме SQL позволяет понять и освоить синтаксис основных инструкций языка SQL, реализованного в Access.

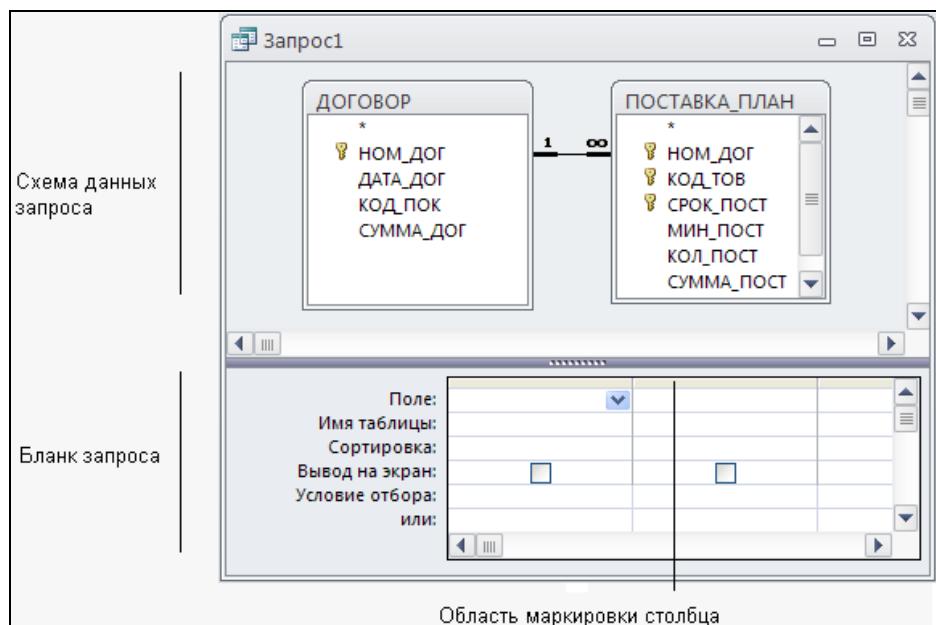
Основным видом запроса является запрос на выборку. Результатом выполнения этого запроса является новая таблица, которая существует до закрытия запроса. Структура такой таблицы определяется выбранными из одной или нескольких взаимосвязанных таблиц полями. Записи формируются путем объединения записей таблиц, на которых построен запрос. Способ объединения записей таблиц указывается при определении их связи в схеме данных или при создании запроса. Условия отбора, сформулированные в запросе, позволяют фильтровать записи, составляющие результат объединения таблиц. Запрос на выборку позволяет сформировать пользовательское представление о данных, не обязательно отвечающее требованиям нормализации.

В Access может быть создано несколько видов запроса:

- ❖ запрос на выборку — выбирает данные из одной таблицы или запроса или нескольких взаимосвязанных таблиц и других запросов. Результатом является таблица, которая существует до закрытия запроса. Формирование записей таблицы производится в соответствии с заданными условиями отбора и при использовании нескольких таблиц путем объединения их записей;
- ❖ запрос на создание таблицы — выбирает данные из взаимосвязанных таблиц и других запросов, но, в отличие от запроса на выборку, результат сохраняет в новой постоянной таблице;

- ❖ запросы на обновление, добавление, удаление — являются запросами действия, в результате выполнения которых изменяются данные в таблицах.

Запрос в режиме конструктора содержит *схему данных*, отображающую используемые таблицы, и *бланк запроса*, в котором конструируется структура таблицы запроса и условия выборки записей (рис. 4.1).



**Рис. 4.1.** Окно конструктора запросов

С помощью запроса можно выполнить следующие виды обработки данных:

- ❖ включить в таблицу запроса выбранные пользователем поля таблицы;
- ❖ произвести вычисления в каждой из полученных записей;
- ❖ выбрать записи, удовлетворяющие условиям отбора;
- ❖ сформировать на основе объединения записей взаимосвязанных таблиц новую виртуальную таблицу;
- ❖ сгруппировать записи, которые имеют одинаковые значения в одном или нескольких полях, одновременно выполнить над другими полями группы статистические функции и в результат включить одну запись для каждой группы;
- ❖ создать новую таблицу базы данных, используя данные из существующих таблиц;
- ❖ произвести обновление полей в выбранном подмножестве записей;
- ❖ удалить выбранное подмножество записей из таблицы базы данных;
- ❖ добавить выбранное подмножество записей в другую таблицу.

Последовательное выполнение ряда запросов позволяет решать достаточно сложные задачи, не прибегая к программированию.

Запросы служат источниками записей для других запросов, форм, отчетов. С помощью запроса можно собрать полные сведения для формирования некоторого документа предметной области из нескольких таблиц, далее использовать его для создания формы — электронного представления этого документа. Если форма или отчет создаются мастером на основе нескольких взаимосвязанных таблиц, то для них в качестве источника записей автоматически формируется запрос. Ненормализованная с повторяющимися данными таблица запроса не хранится в базе данных. Хранится запрос, который может быть использован различными пользователями для выполнения своих задач.

## Однотабличные запросы на выборку

В простейшем случае запрос реализует выбор из одной таблицы нужных полей, записей, соответствующих заданным условиям отбора, и просмотр результатов выполнения запроса.

### Конструирование запросов на выборку с условиями отбора

Рассмотрим процесс конструирования однотабличного запроса на выборку на примере получения информации из таблицы ТОВАР базы данных *Поставка товаров*.

**Задача 1.** Пусть необходимо выбрать ряд характеристик товара по его наименованию.

1. Для создания запроса в окне базы данных выберите вкладку ленты — **Создание** (Create) и в группе **Запросы** (Queries) нажмите кнопку **Конструктор запросов** (Query Design).

Откроется пустое окно запроса на выборку в режиме конструктора — **ЗапросN** (QueryN) и диалоговое окно **Добавление таблицы** (Show Table) (рис. 4.2).

2. В окне **Добавление таблицы** (Show Table) выберите таблицу ТОВАР и нажмите кнопку **Добавить** (Add). Выбранная таблица будет отображена в области схемы данных запроса. Закройте окно **Добавление таблицы** (Show Table), нажав кнопку **Закрыть** (Close).

В результате выполненных действий в окне конструктора запросов (рис. 4.1) в верхней панели появится *схема данных запроса*, которая включает выбранные для данного запроса таблицы. В данном случае одну таблицу ТОВАР. Таблица представлена списком полей. Первая строка в списке полей таблицы, отмеченная звездочкой (\*), обозначает все множество полей таблицы. Нижняя панель является *бланком запроса*, который нужно заполнить.

Кроме того, на ленте появляется и автоматически активизируется новая вкладка **Работа с запросами | Конструктор** (Query Tools | Design) (на рис. 4.3 представлена часть этой вкладки), на которой цветом выделен тип созданного запроса —

**Выборка (Select).** Таким образом, по умолчанию всегда создается запрос на выборку. Команды этой вкладки представляют инструментарий для выполнения необходимых действий при создании запроса. Эта вкладка открывается, когда в режиме конструктора создается новый запрос или редактируется существующий.

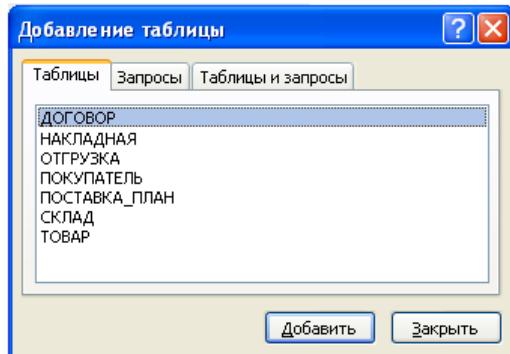


Рис. 4.2. Окно выбора таблиц и запросов для схемы данных запроса

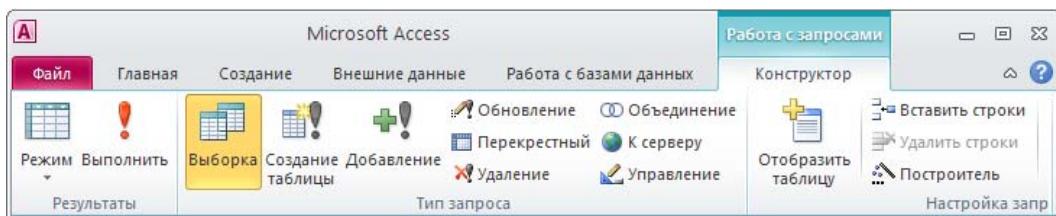
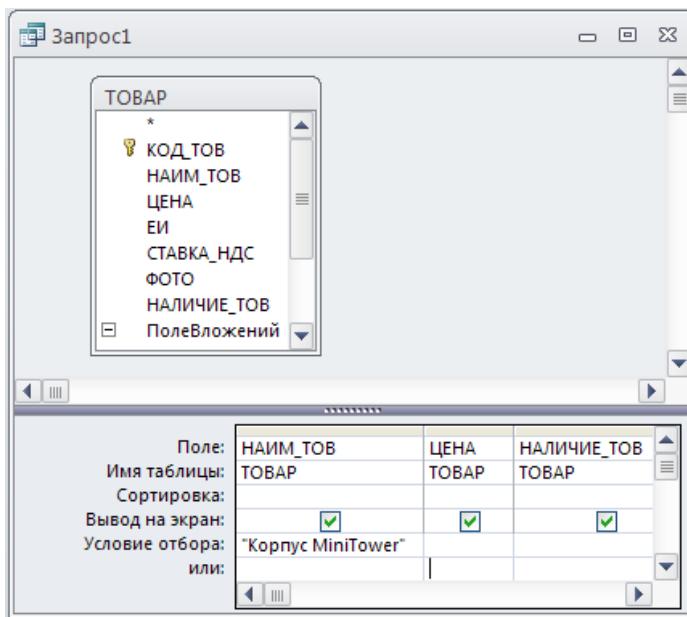


Рис. 4.3. Лента инструментов конструктора запросов

1. Для удаления любой таблицы из схемы данных запроса установите на нее курсор мыши и нажмите клавишу <Delete>. Для добавления — нажмите кнопку **Отобразить таблицу** (Show Table)  в группе **Настройка запроса** (Query Setup) на вкладке **Работа с запросами | Конструктор** (Query Tools | Design) или выполните команду **Добавить таблицу** (Show Table) в контекстном меню, вызываемом на схеме данных запроса.
2. В окне конструктора (рис. 4.4) последовательно перетащите из списка полей таблицы ТОВАР поля НАИМ\_ТОВ, ЦЕНА, НАЛИЧИЕ\_ТОВ в столбцы бланка запроса в строку **Поле** (Field).
3. Для включения нужных полей из таблицы в соответствующие столбцы запроса можно воспользоваться следующими приемами:
  - ◆ в первой строке бланка запроса **Поле** (Field) щелчком мыши вызвать появление кнопки списка и выбрать из списка нужное поле. Список содержит поля таблиц, представленных в схеме данных запроса;
  - ◆ дважды щелкнуть на имени поля таблицы в схеме данных запроса;
  - ◆ для включения всех полей таблицы можно перетащить или дважды щелкнуть на символе \* (звездочка) в списке полей таблицы в схеме данных запроса.

4. Если вы по ошибке перетащили в бланке запроса ненужное поле, удалите его. Для этого переместите курсор в область маркировки столбца сверху, где он примет вид черной стрелки, направленной вниз, и щелкните кнопкой мыши. Столбец выделится. Нажмите клавишу <Delete> или выполните команду **Удалить столбцы** (Delete Columns) в группе **Настройка запроса** (Query Setup).
5. В строке **Вывод на экран** (Show) отметьте поля, иначе они не будут включены в таблицу запроса.
6. Запишите в строке **Условия отбора** (Criteria) наименование товара, как показано в бланке запроса на рис. 4.4. Так как выражение в условии отбора не содержит оператора, то по умолчанию используется оператор =. Используемое в выражении текстовое значение вводится в двойных кавычках, которые добавляются автоматически.



**Рис. 4.4.** Окно конструктора запроса на выборку

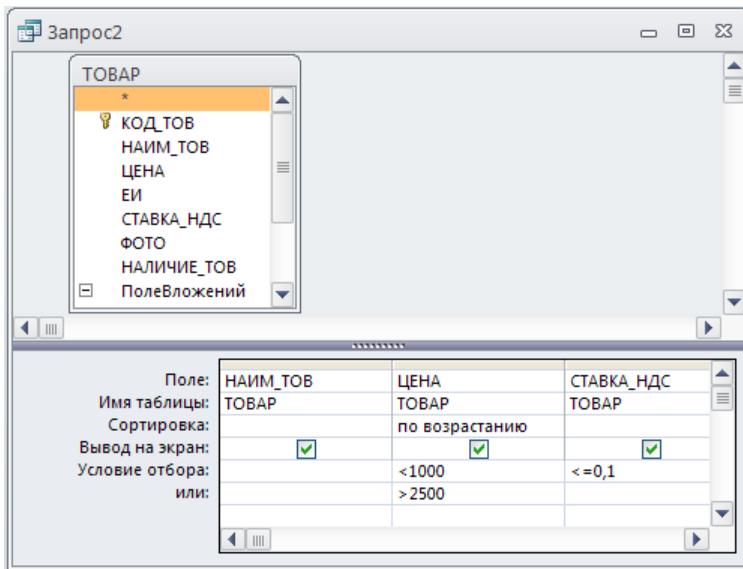
7. Выполните запрос, щелкнув на кнопке **Выполнить** (Run) или на кнопке **Режим** (View) в группе **Результаты** (Results). На экране появится окно запроса в режиме таблицы с записью из таблицы ТОВАР, отвечающей заданным условиям отбора.

### ЗАМЕЧАНИЕ

Окно запроса в режиме таблицы аналогично окну просмотра таблицы базы данных. Через некоторые таблицы запроса может производиться изменение данных базовой таблицы, лежащей в основе запроса. Запрос, просматриваемый в режиме таблицы, в отличие от таблицы базы данных Access 2010, не имеет столбца **Щелкнуть для добавления**

(Click to Add), предназначенного для изменения структуры таблицы. В этом режиме на вкладке ленты **Главная** (Home) доступны те же кнопки, что и при открытии таблицы базы данных.

8. Если при вводе сложного наименования товара вы допустили неточность, товар не будет найден в таблице. Использование *операторов шаблона* — звездочка (\*) и вопросительный знак (?) (стандарт ANSI-89, используемый для запросов по умолчанию) или знак процента (%) и подчеркивания (\_) (ANSI-92, рекомендуемый как стандарт для SQL Server), упрощает поиск нужных строк и позволяет избежать многих ошибок. Введите вместо полного имени товара Корпус\* или Корпус%. Выполните запрос. Если в поле наименования товара одно значение начинается со слова "Корпус", результат выполнения запроса будет таким же, как в предыдущем случае. После выполнения запроса введенное выражение будет дополнено оператором Like "Корпус\*". Этот оператор позволяет использовать символы шаблона при поиске в текстовых полях.
  9. Если необходимо найти несколько товаров, используйте оператор In. Он позволяет выполнить проверку на равенство любому значению из списка, который задается в круглых скобках. Запишите в строке условий отбора In ("Корпус MiniTower"; "HDD Maxtor 20GB"; "FDD 3,5"). В таблице запроса будет выведено три строки. В операторе In не допускается использование символов шаблона.
  10. Сохраните запрос, щелкнув на вкладке **Файл** (File) и выполнив команду **Сохранить** (Save). В окне **Сохранение** (Save As) введите имя запроса **Пример1**. Заметим, что имя запроса не должно совпадать не только с именами имеющихся запросов, но и с именами таблиц в базе данных.
  11. Закройте текущий запрос по команде контекстного меню **Закрыть** (Close) или нажав кнопку окна запроса **Закрыть** (Close).
  12. Выполните сохраненный запрос, выделив запрос в области навигации и выбрав в контекстном меню команду **Открыть** (Open).
  13. Для редактирования запроса выделите его в области навигации и выполните в контекстном меню команду **Конструктор** (Design View).
- Задача 2.** Пусть надо выбрать товары, цена которых не более 1000 руб., и НДС не более 10%, а также выбрать товары, цена которых более 2500 руб. Результат должен содержать наименование товара (НАИМ\_ТОВ), его цену (ЦЕНА) и НДС (СТАВКА\_НДС).
1. Создайте новый запрос в режиме конструктора, добавьте таблицу ТОВАР. В окне конструктора (рис. 4.5) последовательно перетащите из списка полей таблицы ТОВАР в бланк запроса поля НАИМ\_ТОВ, ЦЕНА, СТАВКА\_НДС.
  2. Запишите **Условия отбора** (Criteria), как показано в бланке запроса на рис. 4.5. Между условиями, записанными в одной строке, выполняется логическая операция AND. Между условиями, записанными в разных строках, выполняется логическая операция OR.
  3. Выполните запрос, щелкните на кнопке **Выполнить** (Run) в группе **Результаты** (Results). На экране появится окно запроса в режиме таблицы с записями из таблицы ТОВАР, отвечающими заданным условиям отбора.



**Рис. 4.5.** Окно конструктора запроса на выборку с логическими операциями в условии отбора

4. Сохраните запрос, выполнив соответствующую команду в контекстном меню запроса, которое вызывается при установке курсора на заголовок запроса. Дайте ему имя **Пример2**.

**Задача 3.** Пусть надо выбрать все накладные за заданный период. Результат должен содержать номер накладной (НОМ\_НАК), код склада (КОД\_СК), дату отгрузки (ДАТА\_ОТГР) и общую стоимость отгруженного товара (СУММА\_НАКЛ).

1. Создайте новый запрос в режиме конструктора, добавьте таблицу НАКЛАДНАЯ. В окне конструктора последовательно перетащите из списка полей таблицы НАКЛАДНАЯ в бланк запроса все необходимые поля.
2. Для поля ДАТА\_ОТГР в строке **Условия отбора** (Criteria) запишите Between #11.01.2008# And #31.03.2008#. Оператор Between задает интервал дат (в ANSI-92 вместо знака # используются одинарные кавычки '). Кроме того, этот оператор позволяет задать интервал для числового значения.

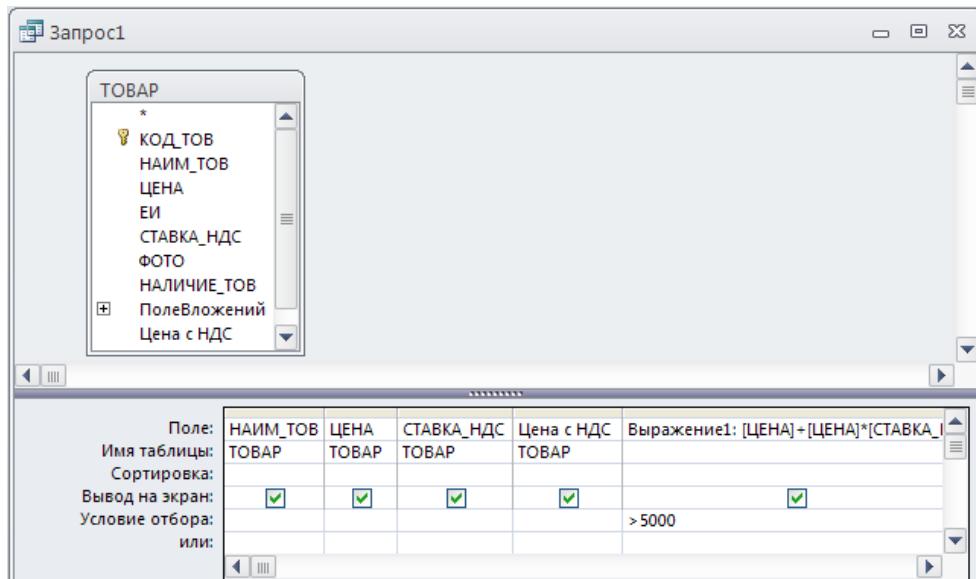
## Вычисляемые поля в запросах

В запросе, как и в таблице, для каждой записи могут производиться вычисления с числовыми, строковыми значениями или значениями дат с использованием данных из одного или нескольких полей. Результат вычисления образует в таблице запроса новое вычисляемое поле. В отличие от вычисляемых полей таблицы, вычисляемые поля запроса в исходных таблицах базы данных новых полей не создают. При каждом выполнении запроса производятся вычисления на основе текущих значений полей.

В выражениях вычисляемых полей помимо имен полей могут использоваться константы и функции. В результате обработки выражения может получаться только одно значение.

**Задача 1.** В таблице ТОВАР имеются поля ЦЕНА и СТАВКА\_НДС, вычислите цену с учетом НДС и сравните ее с полученной в вычисляемом поле таблицы Цена с НДС.

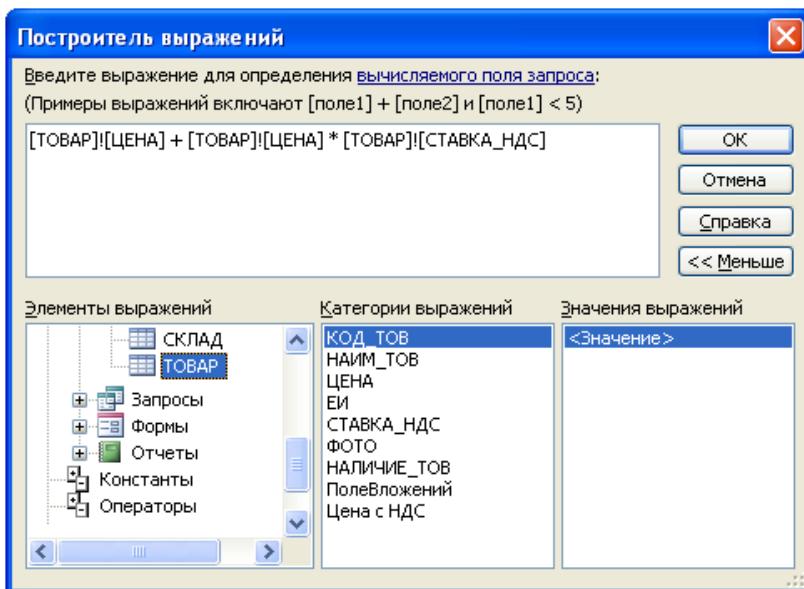
1. Создайте в режиме конструктора запрос на выборку для таблицы ТОВАР. Перетащите в бланк запроса поля НАИМ\_ТОВ, ЦЕНА, СТАВКА\_НДС и Цена с НДС (рис. 4.6).
2. Для подсчета цены с учетом НДС создайте вычисляемое поле, записав в пустой ячейке строки **Поле** (Field) выражение  $[ЦЕНА] + [ЦЕНА] * [СТАВКА_НДС]$ .
3. Для отбора записей со значением выше 5000 в вычисляемом поле в строку **Условие отбора** (Criteria) введите  $> 5000$ .



**Рис. 4.6.** Запрос с вычисляемым полем и условием отбора по его значению

4. После ввода выражения система по умолчанию формирует имя вычисляемого поля Выражение1, которое становится заголовком столбца в таблице с результатами выполнения запроса. Это имя вставится перед выражением  $[ЦЕНА] + [ЦЕНА] * [СТАВКА_НДС]$ . Для каждого нового вычисляемого поля в запросе номер выражения увеличивается на единицу. Имя вычисляемого поля отделяется от выражения двоеточием. Для изменения имени установите курсор мыши в вычисляемом поле бланка запроса и нажмите правую кнопку мыши. В контекстно-зависимом меню выберите **Свойства** (Properties) поля и в строку **Подпись** (Caption) введите новое имя поля — **Цена с НДС1**. Теперь в таблице с результатами выполнения запроса в заголовке вычисляемого столбца отобра-

- зится это имя. Имя поля может быть исправлено также непосредственно в бланке запроса.
- Для отображения результата выполнения запроса щелкните на кнопке **Выполнить** (Run) в группе **Результаты** (Results). Вычисляемое поле таблицы и запроса имеют одинаковые значения.
  - Измените в одной из записей запроса цену товара. Значения в обоих вычисляемых полях будут моментально пересчитаны.
  - Для формирования сложного выражения в вычисляемом поле или условии отбора целесообразно использовать построитель выражений. Постройтель позволяет выбрать необходимые в выражении имена полей из таблиц, запросов, знаки операций, функции. Удалите выражение в вычисляемом поле и используйте построитель для его формирования.
  - Вызовите построитель выражений (Expression Builder), нажав кнопку **Постройтель** (Builder)  в группе **Настройка запроса** (Query Setup) ленты **Конструктор** (Design), или выбрав **Построить** (Build) в контекстно-зависимом меню. Курсор мыши должен быть установлен предварительно в ячейке ввода выражения.



**Рис. 4.7.** Окно построителя выражений при формировании вычисляемого поля

- В левой части окна **Постройтель выражений** (Expression Builder) (рис. 4.7) выберите таблицу ТОВАР, на которой построен запрос. Справа отобразится список ее полей. Последовательно выбирайте нужные поля и операторы, двойным щелчком вставляя в выражение. Выражение сформируется в верхней части окна. Обратите внимание, построитель перед именем поля указал имя таблицы, которой оно принадлежит, и отделил его от имени поля восклицательным знаком.

10. Завершите процесс построения выражения в вычисляемом поле, щелкнув на кнопке **OK**.

11. Сохраните запрос под именем — **Цена с НДС** и закройте его.

12. Выполните сохраненный запрос, выделив его в области навигации и выбрав в контекстном меню команду **Открыть** (Open).

**Задача 2.** В вычисляемых полях и условиях отбора можно использовать *встроенные функции*. В Access определено более 150 функций.

Пусть необходимо выбрать все накладные, по которым производилась отгрузка в заданном месяце. В таблице НАКЛАДНАЯ дата отгрузки хранится в поле ДАТА\_ОТГ с типом данных **Дата/время** (Date/Time).

1. Создайте в режиме конструктора запрос на выборку для таблицы НАКЛАДНАЯ.

Перетащите в бланк запроса поля НОМ\_НАКЛ и КОД\_СК (рис. 4.8).

2. Создайте вычисляемое поле в пустой ячейке строки **Поле** (Field), записав туда одно из выражений:

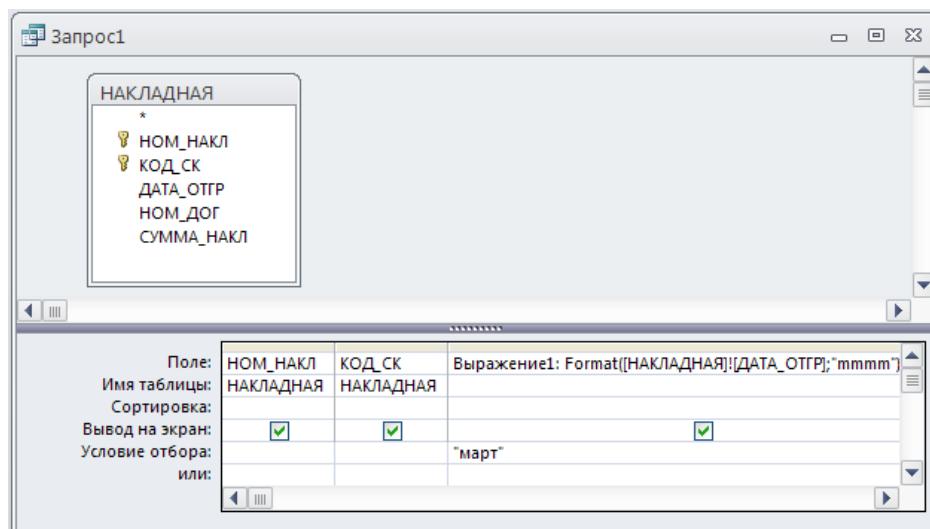
- ◆ Format([НАКЛАДНАЯ]![ДАТА\_ОТГР]; "мммм") — эта функция возвратит полное название месяца

или

- ◆ Format([НАКЛАДНАЯ]![ДАТА\_ОТГР]; "мм") — эта функция возвратит номер месяца.

### ЗАМЕЧАНИЕ

Если необходимо выделить не один элемент из даты, запишите в функции Format несколько символов форматирования, разделяя их знаком \. Например, функция Format([НАКЛАДНАЯ]![ДАТА\_ОТГР]; "мммм\ yyyy") возвратит полное название месяца и четыре цифры года. После знака \ нужно поставить пробел.



**Рис. 4.8.** Запрос с функцией выделения из даты полного названия месяца в вычисляемом поле

3. Для отбора накладных, выписанных в заданном месяце, в вычисляемом поле в строку **Условие отбора** (Criteria) введите название месяца, например март (рис. 4.8), или номер месяца, например 3 в соответствии с параметром в функции Format.
4. Выполните запрос, нажав кнопку **Выполнить** (Run) в группе **Результаты** (Results) на вкладке ленты **Работа с запросами | Конструктор** (Query Tools | Design).
5. Запишите в вычисляемом поле функцию Month(НАКЛАДНАЯ!ДАТА\_ОТГ), и убедитесь, что эта функция возвращает выделенный из даты номер месяца.
6. Для выборки всех строк, относящихся ко второму кварталу, в строку **Условие отбора** (Criteria) введите оператор Between 4 And 6, определяющий, попадает ли значение выражения в указанный интервал.
7. Запишите в вычисляемом поле выражение MonthName(Month(НАКЛАДНАЯ!ДАТА\_ОТГ)) и убедитесь, что функция MonthName преобразует номер месяца в его полное название.

## Параметры в запросах

В предыдущих примерах выражение в условие отбора вводилось в бланке запроса. При этом чтобы задать новое значение в условие отбора, нужно повторно открыть запрос в режиме конструктора и ввести его. При решении практических задач значительно удобнее вводить выражение в условие отбора в процессе выполнения запроса в диалоге с пользователем, не переходя в режим конструктора. Обеспечить такой диалог можно с помощью *параметра запроса*. Имя параметра запроса задается в строке **Условия отбора** (Criteria) в квадратных скобках. При выполнении запроса это имя появится в диалоговом окне **Введите значение параметра** (Enter Parameter Value).

1. Замените в условии отбора рассмотренного запроса (см. рис. 4.8) название месяца март на имя параметра — [Название месяца].
2. Выполните запрос. Открывшееся диалоговое окно (рис. 4.9) позволит ввести значение параметра запроса — **Название месяца**.

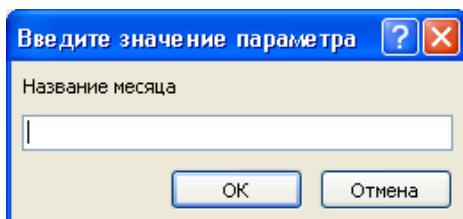


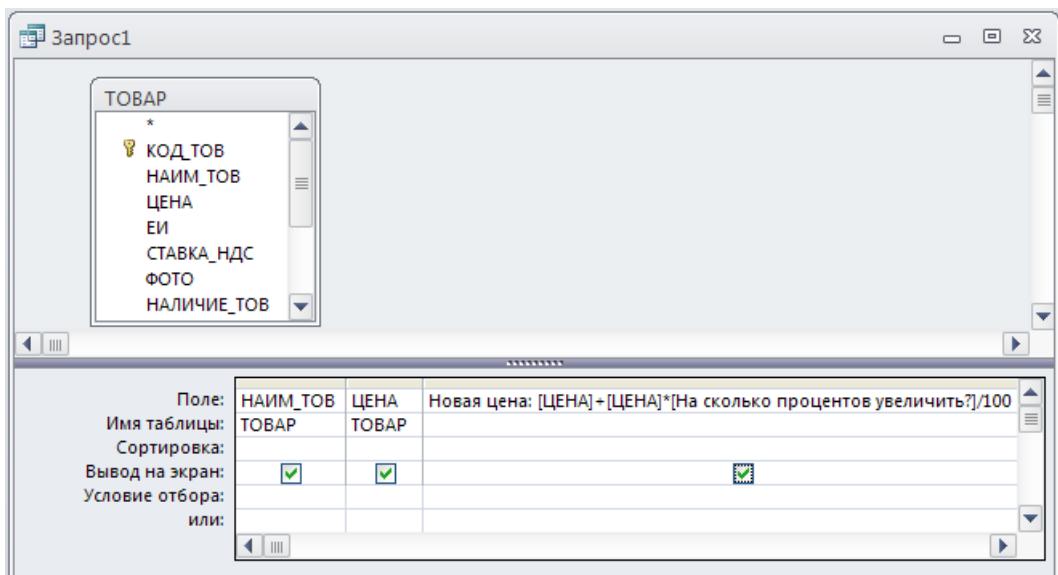
Рис. 4.9. Диалоговое окно ввода значения параметра

3. В запросе может быть определено несколько параметров. Например, для отбора записей по двум месяцам в условии отбора вычисляемого поля запишите два параметра, связанных логической операцией OR — [Название месяца] OR [Еще одно название месяца].

4. Для определения числового интервала используйте параметры в операторе Between [Номер первого месяца периода] and [Номер последнего месяца периода].

Параметры запроса могут быть использованы не только в выражениях условий отбора, а и для ввода значений операндов в вычисляемых полях.

1. Создайте в режиме конструктора запрос на выборку для таблицы ТОВАР. Перетащите в бланк запроса поля НАИМ\_ТОВ и ЦЕНА.
2. Для увеличения цены на заданный процент в вычисляемое поле запишите выражение с параметром запроса — [На сколько процентов увеличить?] (рис. 4.10):  
ЦЕНА+ЦЕНА\* [На сколько процентов увеличить?]/100



**Рис. 4.10.** Использование параметра запроса в выражении вычисляемого поля

### **ВНИМАНИЕ!**

Через этот запрос вы сможете показать увеличенные цены в таблице запроса или использовать их при построении форм, отчетов, но они не будут внесены в поле таблицы ТОВАР.

## **Групповые операции в запросах**

Групповые операции позволяют выделить группы записей с одинаковыми значениями в указанных полях и вычислить итоговые данные для каждой из групп по другим полям, используя одну из статистических функций. Статистические функции применимы, прежде всего, к полям с типом данных Числовой, Денежный, Дата/время.

В Access предусматривается девять статистических функций:

- ◆ Sum — сумма значений некоторого поля для группы;
- ◆ Avg — среднее от всех значений поля в группе;
- ◆ Max, Min — максимальное, минимальное значение поля в группе;
- ◆ Count — число значений поля в группе без учета пустых значений;
- ◆ StDev — среднеквадратичное отклонение от среднего значения поля в группе;
- ◆ Var — дисперсия значений поля в группе;
- ◆ First и Last — значение поля из первой или последней записи в группе.

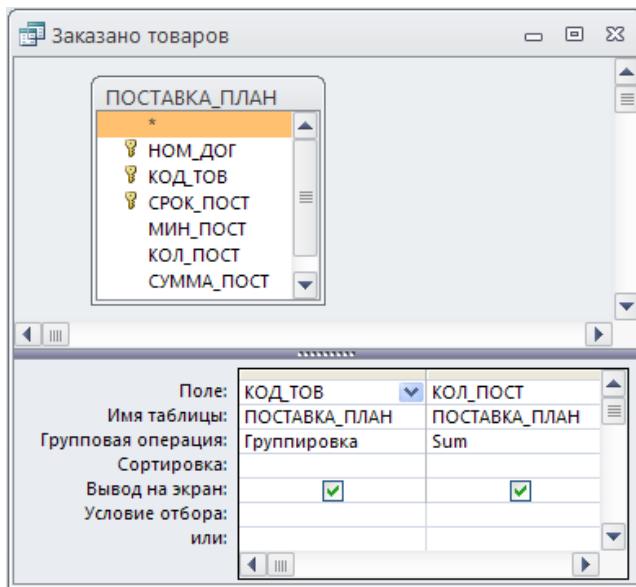
Результат запроса с использованием групповых операций содержит по одной записи для каждой группы. В запрос, прежде всего, включаются поля, по которым производится группировка, и поля, для которых выполняются статистические функции. Кроме этих полей в запрос могут включаться поля, по которым задаются условия отбора.

Рассмотрим конструирование однотабличного запроса с групповой операцией на примере таблицы ПОСТАВКА\_ПЛАН.

### Запрос с функцией Sum

**Задача.** Определите, какое суммарное количество каждого из товаров должно быть поставлено покупателям по договорам. Все данные о запланированном к поставке количестве товара указаны в таблице ПОСТАВКА\_ПЛАН.

1. Создайте в режиме конструктора запрос на выборку из таблицы ПОСТАВКА\_ПЛАН.
2. Из списка таблицы перетащите в бланк запроса поле КОД\_ТОВ — код товара. По этому полю будет производиться группировка записей таблицы.
3. Перетащите в бланк запроса поле КОЛ\_ПОСТ, по которому будет подсчитываться суммарное количество каждого из товаров, заказанных во всех договорах.
4. Выполните команду **Итоги** (Totals) из группы **Показать или скрыть** (Show/Hide). В бланке запроса появится новая строка **Групповая операция** (Total) со значением **Группировка** (Group By) в обоих полях запроса.
5. В столбце КОЛ\_ПОСТ замените слово **Группировка** (Group By) на функцию Sum. Для этого вызовите список и выберите эту функцию. Бланк запроса примет вид, показанный на рис. 4.11.
6. Для отображения результата запроса (рис. 4.12) щелкните на кнопке **Выполнить** (Run) в группе **Результаты** (Results).
7. Замените подпись поля Sum-КОЛ\_ПОСТ на **Заказано товаров**. Для этого перейдите в режим конструктора, в бланке запроса установите курсор мыши на поле КОЛ\_ПОСТ и нажмите правую кнопку. В контекстном меню выберите **Свойства** (Properties). В окне **Свойства поля** (Field Properties) введите в строке **Подпись** (Caption) — Заказано товаров. Для открытия окна свойств может быть выполнена команда **Страница свойств** (Property Sheet) в группе **Показать или скрыть** (Show/Hide).



**Рис. 4.11.** Запрос с группировкой по коду товара и суммированием количества в группе

Заказано товаров	
Код товара	Заказано товара
Монитор 17LG	1227
FDD 3,5	6475
HDD Maxtor 20GB	124
Корпус MiniTower	30
CD-ROM Panasonic IDE	50
DIMM 64M PC100	200
Принтер EPSON ST.A4	10
СканерAcer	4

**Рис. 4.12.** Результат подсчета суммарного количества каждого из товаров

8. Сохраните запрос под именем **Заказано товаров**.
9. Чтобы подсчитать количество товаров, заказанных в каждом месяце, выполните группировку по двум полям: КОД\_ТОВ и СРОК\_ПОСТ, в котором хранится месяц поставки (рис. 4.13).
10. Чтобы подсчитать количество товаров, заказанных в заданном месяце, предыдущий запрос дополните вводом параметра запроса в условие отбора (рис. 4.14).

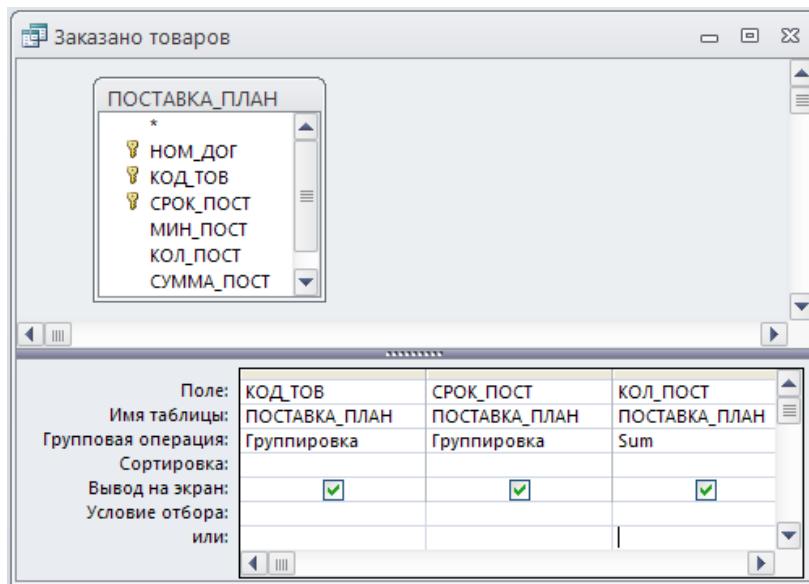


Рис. 4.13. Запрос с группировкой по двум полям

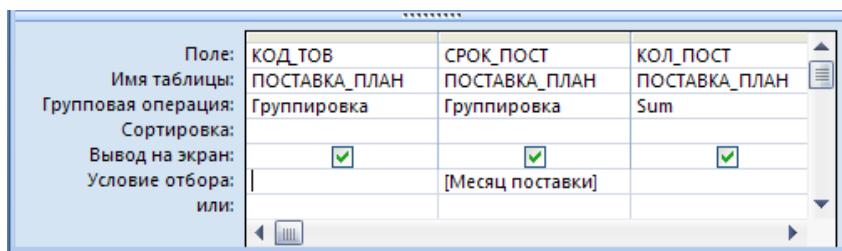
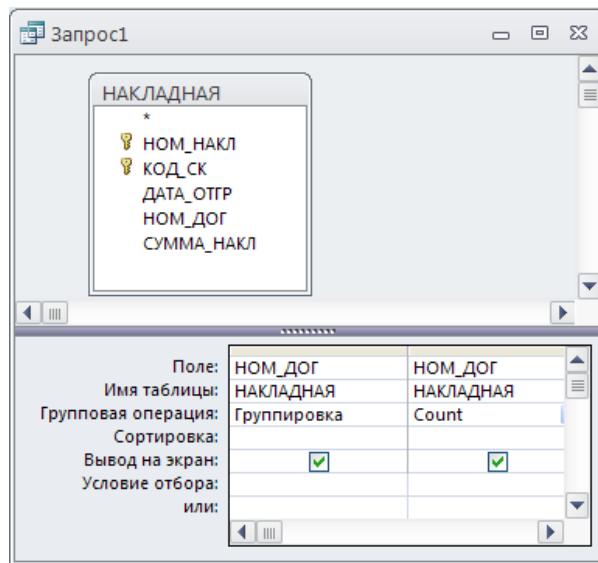


Рис. 4.14. Запрос с группировкой по двум полям и параметром запроса

## Запрос с функцией Count

**Задача.** Определите, сколько раз отгружался товар по каждому из договоров. Факт отгрузки фиксируется в таблице НАКЛАДНАЯ.

1. Создайте запрос на выборку на основе таблицы НАКЛАДНАЯ.
2. Из списка полей таблицы НАКЛАДНАЯ перетащите в бланк запроса поле НОМ\_ДОГ. По этому полю должна производиться группировка.
3. По сути, смысл задачи сводится к подсчету в таблице числа строк с одинаковым номером договора, поэтому неважно по какому полю будет вычисляться функция Count. Перетащите в бланк запроса любое поле, например опять НОМ\_ДОГ.
4. Выполните команду Итоги (Totals) из группы Показать или скрыть (Show/Hide). Замените слово Группировка (Group By) в одном из столбцов с именем НОМ\_ДОГ на функцию Count. Бланк запроса примет вид, показанный на рис. 4.15.



**Рис. 4.15.** Запрос для подсчета числа отгрузок по договорам

- Сохраните запрос под именем **Число отгрузок по договорам**. Выполните запрос. Результат запроса показан на рис. 4.16.

Число отгрузок по договорам	
Номер договора	Count-НОМ_ДОГ
д111	4
д222	5
д333	3

**Рис. 4.16.** Результат подсчета числа отгрузок по договорам

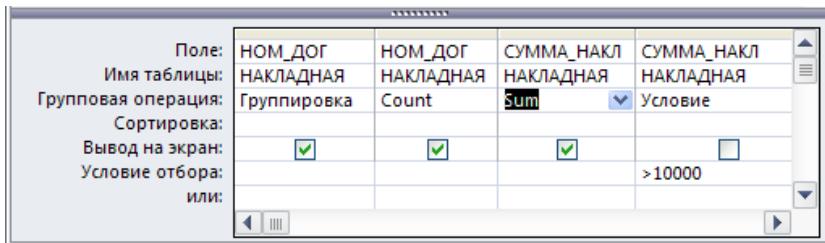
## Условия отбора в запросах с групповыми операциями

В запрос с групповыми операциями помимо полей, по которым производятся группировка и выполнение статистических функций для групп, можно включать поля для задания условий отбора из таблиц записей, включаемых в группы.

**Задача.** Подсчитайте, сколько накладных было выписано по каждому из договоров и какова общая стоимость товаров, отгруженных по этим накладным. В расчете учитывайте только накладные на сумму более 10 000 руб.

- Для подсчета общей стоимости товаров дополните бланк запроса **Число отгрузок по договорам** (см. рис. 4.15, 4.16) полем СУММА\_НАКЛ и замените в нем слово **Группировка** (Group By) на функцию `Sum`.

2. Затем вторично включите поле СУММА\_НАКЛ в бланк запроса и замените слово **Группировка** (Group By) словом **Условие** (Where), выбрав его из списка. В строку **Условие отбора** (Criteria) введите выражение  $>10000$  (рис. 4.17).



**Рис. 4.17.** Запрос с групповой операцией для записей, соответствующих условию отбора

3. Выполните запрос и убедитесь, что число накладных для некоторых договоров уменьшилось, а общая стоимость по договору также считается только с учетом накладных на сумму более 10 000 (рис. 4.18).

Число отгрузок по договорам		
Номер договора	Count-НОМ_ДОГ	Sum-СУММА_НАКЛ
д111	3	820 600,00р.
д222	4	77 740,00р.
д333	1	30 056,00р.

**Рис. 4.18.** Результат выполнения запроса

4. Чтобы отобрать только нужные группы записей, введите условие отбора в поле, по которому производится группировка, или в поле, где записана функция. Например, чтобы отобрать договора с заданными номерами, введите в условие отбора поля НОМ\_ДОГ In ("д111"; "д333"). Чтобы отобрать договора с данной общей стоимостью отгрузки по ним, введите в поле СУММА\_НАКЛ  $>10000$ .

## Отображение строки итогов по столбцу

Строка итогов используется для быстрого расчета и отображения в столбце таблицы или запроса в режиме таблицы таких значений, как итоговая сумма, среднее, минимальное и максимальное, количество значений.

1. Для добавления строки итогов в таблицу запроса откройте запрос **Число отгрузок по договорам** в режиме таблицы. На вкладке ленты **Главная** (Home) в

группе **Записи** выполните команду **Итоги** (Totals). В таблице отобразится строка **Итог** (Total).

- В строке **Итог** (Total) нажмите кнопку со стрелкой вниз в столбце, для которого требуется выполнить расчет, и выберите в списке, например, **Максимальное значение** (Maximum). Результат выбора максимального значения в этом столбце показан на рис. 4.19.

### **ЗАМЕЧАНИЕ**

Кнопка со стрелкой вниз расположена с другой стороны ячейки для того, чтобы избежать нарушений выравнивания текста и чисел.

Номер договора	Count-НОМ_ДОГ	Sum-СУММА_НАКЛ
Д111	3	820 600,00р.
Д222	4	77 740,00р.
Д333	1	30 056,00р.
<b>Итог</b>		820 600,00р.

Запись: 14 < Итоги > 15 Нет фильтра

Нет  
Сумма  
Среднее  
Количество значений  
**Максимальное значение**  
Минимальное значение  
Стандартное отклонение  
Дисперсия

**Рис. 4.19.** Отображение строки итогов с расчетом максимума по столбцу

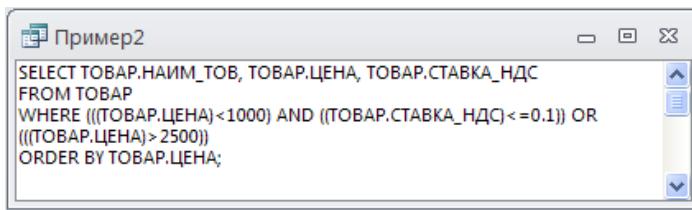
- Для того чтобы скрыть строку итогов, повторно выполните команду **Итоги** (Totals).

## **Просмотр инструкции SQL в запросе**

- Откройте в режиме конструктора рассмотренный ранее запрос (см. рис. 4.5), сохраненный под именем **Пример2**.
- Для перехода в **Режим SQL** (SQL View) выберите из списка кнопки **Режим** (View) соответствующее значение. Можно также воспользоваться кнопкой **Режим SQL** в строке состояния. Access выведет для запроса, созданного в режиме конструктора, эквивалентную инструкцию SQL (рис. 4.20).

Инструкция **SELECT** используется для формирования таблицы запроса, структура которой определяется выбранными из таблиц базы данных полями, а набор записей сформулированными условиями отбора.

Список имен нужных полей следует непосредственно за ключевым словом **SELECT**.



```
SELECT ТОВАР.НАИМ_ТОВ, ТОВАР.ЦЕНА, ТОВАР.СТАВКА_НДС
FROM ТОВАР
WHERE (((ТОВАР.ЦЕНА)<1000) AND ((ТОВАР.СТАВКА_НДС)<=0.1)) OR
(((ТОВАР.ЦЕНА)>2500))
ORDER BY ТОВАР.ЦЕНА;
```

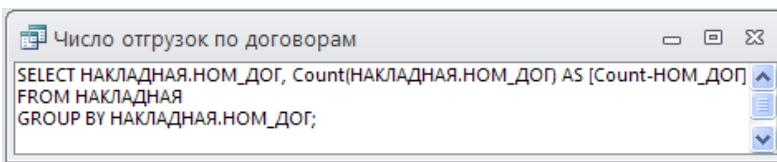
Рис. 4.20. Запрос с логическими операциями в условии отбора в режиме SQL

Предложение `FROM` определяет таблицы или запросы, которые служат источником данных для данного запроса, и способ объединения их записей, если запрос построен на нескольких таблицах.

В предложении `WHERE` задаются условия отбора записей, которые полностью соответствуют сформулированным в задании на создание запроса (см. рис. 4.5).

Предложение `ORDER BY` указывает, по каким полям должна быть выполнена сортировка записей в таблице запроса. При отсутствии сортировки этого предложения нет в инструкции. Данная инструкция указывает на сортировку по возрастанию по полю `ТОВАР.ЦЕНА`.

1. Перейдите в режим конструктора, внесите изменения в бланк запроса, например, добавьте поле КОД ТОВАРА — `КОД_ТОВ`.
2. Вернитесь в режим SQL и посмотрите, как изменилась инструкция SQL.
3. Чтобы убедиться, что запросы можно редактировать в режиме SQL, а не только в режиме конструктора, внесите изменения в инструкцию SQL, удалив из списка полей `КОД_ТОВ`.
4. Затем перейдите в режим конструктора и убедитесь, что из бланка запроса удалено поле `КОД_ТОВ`.
5. Откройте в режиме конструктора запрос **Число отгрузок по договорам**, в котором используется только функция подсчета числа договоров.
6. Перейдите в режим SQL. Инструкция SQL в этом запросе будет записана, как показано на рис. 4.21.



```
SELECT НАКЛАДНАЯ.НОМ_ДОГ, Count(НАКЛАДНАЯ.НОМ_ДОГ) AS [Count-НОМ_ДОГ]
FROM НАКЛАДНАЯ
GROUP BY НАКЛАДНАЯ.НОМ_ДОГ;
```

Рис. 4.21. Запрос с использованием функции `Count` в режиме SQL

В инструкции `SELECT` определен список полей для таблицы запроса. Это поле `НОМ_ДОГ`, по которому производится группировка, и вычисляемое поле, содержащее результат выполнения функции `Count`, примененной к полю `НОМ_ДОГ`. Зарезервированное слово `AS` позволяет определить для поля с результатом вычисления псевдоним `Count-НОМ_ДОГ`, который станет заголовком столбца.

Обратите внимание в инструкции наряду с именами полей, включаемых в результат, допускается использование функций. Возможно включение и более сложных выражений, определяющих содержимое вычисляемых полей.

Предложение `GROUP BY` указывает, что необходимо объединить записи с одинаковыми значениями в указанных полях (`НАКЛАДНАЯ.НОМ_ДОГ`) в одну запись.

## Контрольные вопросы

1. Какая команда позволяет добавить необходимые таблицы в запрос?
2. Как включить в запрос нужные поля таблицы?
3. Можно ли разместить поле в бланке запроса, дважды щелкнув на нем в списке полей таблицы?
4. Можно ли заменить поле в бланке запроса выбором его из раскрывающегося списка?
5. Какие записи составляют таблицу запроса на выборку?
6. Какая логическая операция применяется к условиям отбора, записанным для различных полей в одной строке?
7. Какая логическая операция применяется к условиям отбора, записанным для одного поля в различных строках?
8. Какие операторы сравнения и логические операторы могут быть использованы при записи условия отбора?
9. Как выполнить запрос, не открывая его в режиме конструктора?
10. Можно ли использовать в выражении условия отбора имя поля?
11. Значения полей каких записей используются в выражении вычисляемого поля?
12. Где хранятся значения вычисляемого поля?
13. Где задается имя вычисляемого поля?
14. Как задать параметр запроса?
15. Из каких записей образуется группа при использовании групповой операции?
16. Какие поля выбираются для группировки записей?
17. Можно ли задать условия отбора записей, включаемых в группы?
18. Какое значение должно быть выбрано в строке **Групповые операции** (Total) для поля, по которому задается условие отбора?
19. В каком столбце бланка запроса записывается условие отбора нужных групп?
20. Как просмотреть инструкцию SQL, построенную конструктором или мастером запросов?
21. Какая инструкция SQL соответствует запросу на выборку?
22. Изменяет ли инструкция `SELECT` данные в базе?
23. В каком предложении инструкции SQL указываются таблицы, на которых создается запрос?
24. В каком предложении инструкции SQL задаются условия отбора записей?
25. Можно ли в полях, указанных в списке инструкции SQL, применять функции?

26. В каком предложении инструкции SQL задаются поля, по которым должна быть выполнена группировка записей?
27. Каким знаком разделяются имена полей в списке инструкции SQL?
28. Можно ли в инструкции SQL использовать символ звездочки (\*) для отбора всех полей таблицы?
29. Какой оператор позволяет задать интервал выбираемых значений в условии отбора?
30. Приведите пример использования оператора Between для выбора интервала дат.
31. Имеет ли смысл производить группировку по ключу?
32. Сколько записей сформируется в таблице запроса при группировке по полю КОД\_ТОВ в таблице ПОСТАВКА\_ПЛАН, содержащей все сведения о заказанных товарах?
33. Какая команда позволяет добавить в таблицу запроса строку итогов?

## Ответы

1. Отобразить таблицу (Show Table) на вкладке конструктора запросов или в контекстном меню схемы данных запроса.
2. Разместив их в столбцах бланка запроса.
3. Да.
4. Да.
5. Соответствующие заданным условиям отбора.
6. AND (И).
7. OR (ИЛИ).
8. =, <, >, <>, <=, >=, Between, In , Like , And, Or, Not.
9. Выделив запрос в области переходов базы данных и выполнив команду Открыть (Open) в его контекстном меню.
10. Да, заключив его в квадратные скобки.
11. Значения полей одной записи.
12. Только в записях таблицы запроса.
13. В его свойстве Подпись (Caption) или непосредственно перед выражением вычисляемого поля.
14. Записав имя параметра, заключенное в квадратные скобки, в условиях отбора.
15. Из записей с одинаковыми значениями в полях группировки.
16. Поля, имеющие повторяющиеся значения.
17. Да.
18. Условие (Where).
19. В поле, по которому производится группировка или для которого выполняется функция.
20. Выбрав для запроса в режиме конструктора режим SQL из списка кнопки Режим (View) в группе Результаты (Results).
21. SELECT.

22. Нет.
23. FROM.
24. WHERE.
25. Да.
26. GROUP BY.
27. Запятой.
28. Да.
29. Between.
30. Between #11.01.2008# And #31.03.2008#.
31. Нет.
32. Сколько видов товара было заказано.
33. Команда **Итоги** (Totals) на вкладке ленты **Главная** (Home) в группе **Записи** (Records).

## Многотабличные запросы на выборку данных

- ❖ Многотабличный запрос позволяет сформировать записи результата путем объединения взаимосвязанных записей из таблиц базы данных и выбора из них нужных полей и записей. Многотабличный запрос часто осуществляет объединение данных, которые на этапе проектирования были разделены на множество таблиц, отвечающих требованиям нормализации. В нормализованных таблицах, прежде всего, обеспечивалось отсутствие повторяемости данных в базе, повторяются только значения ключевых полей. В результате выполнения запроса формируется ненормализованная таблица с повторяющимися данными, в которой каждая запись собирает необходимые данные из разных таблиц.
- ❖ Например, при объединении двух нормализованных связанных одно-многозначными отношениями таблиц, для которых обеспечивается связная целостность, результирующая запись образуется на основе записи подчиненной таблицы, в которую добавляются поля из связанной записи главной таблицы. Подобное объединение формирует ненормализованную таблицу, в которой число записей равно числу записей в подчиненной таблице. При этом данные главной таблицы дублируются в различных записях результирующей таблицы.

При конструировании многотабличного запроса важнейшим условием является правильное представление о том, как идет объединение записей таблиц при формировании результата.

В Access имеется возможность задать способ объединения записей двух связанных таблиц как при создании общей схемы базы данных, так и в схеме данных запроса. Для любой пары связанных таблиц может быть выбран один из трех способов объединения записей:

- ❖ способ 1 — объединение только тех записей, в которых связанные поля обеих таблиц совпадают (выбирается по умолчанию);

- ❖ способ 2 — объединение тех записей, в которых связанные поля обеих таблиц совпадают, а также объединение всех записей из первой таблицы, для которых нет связанных во второй, с пустой записью второй таблицы;
- ❖ способ 3 — объединение тех записей, в которых связанные поля обеих таблиц совпадают, а также объединение всех записей из второй таблицы, для которых нет связанных в первой, с пустой записью первой таблицы.

При установлении в схеме данных базы связи между таблицами по умолчанию выбран первый способ объединения. При обработке таблиц используется этот способ объединения, если он специально не переопределен.

### **ЗАМЕЧАНИЕ**

Если между таблицами в запросе не установлена связь, то для Access остается неизвестным, какие записи связаны с какими, и в запросе формируются все комбинации записей таблиц, т. е. объединяются все со всеми. Например, если одна таблица содержит 10 записей, а другая 4, то в таблице запроса будет 40 записей ( $10 \times 4$ ). Такое объединение называется "полное объединение", или "декартово произведение".

## **Объединение записей в многотабличном запросе**

**Задача 1.** Рассмотрим технологию конструирования многотабличного запроса на выборку для решения задачи расчета суммарного количества каждого из товаров, которое должно быть поставлено покупателям по договорам. В таблице запроса выведем, помимо кода товара, его наименование. Для реализации такого запроса необходимы таблицы ТОВАР и ПОСТАВКА\_ПЛАН, находящиеся в отношении 1 :  $M$ . Ранее был создан запрос **Заказано товаров** (см. рис. 4.11, 4.12), решавший аналогичную задачу, но каждый из товаров был представлен только своим кодом из таблицы ПОСТАВКА\_ПЛАН. Добавив таблицу ТОВАР, можно для каждого кода получить его наименование. Прежде чем группировать записи для подсчета суммарного количества каждого из товаров, посмотрим, как образуются эти записи при объединении двух таблиц.

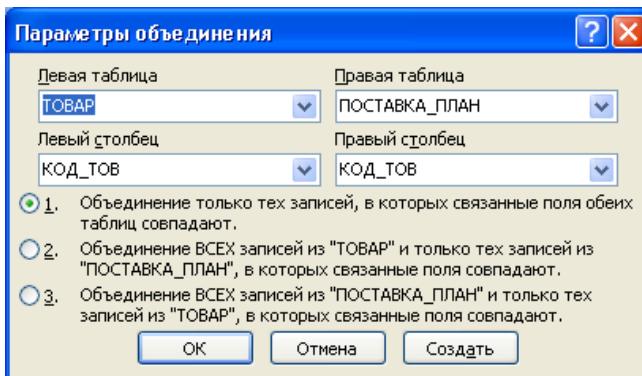
1. Для создания запроса выполните команду **Конструктор запросов** (Query Design), расположенную на ленте **Создание** (Create) в группе **Запросы** (Queries).
2. В окне **Добавление таблицы** (Show Table) (см. рис. 4.2) выберите таблицы ПОСТАВКА\_ПЛАН и ТОВАР и добавьте их в запрос.
3. Закройте окно **Добавление таблицы** (Show Table).
4. В схеме данных запроса будут представлены таблицы ПОСТАВКА\_ПЛАН и ТОВАР, между которыми, в соответствии со схемой данных, автоматически установленна связь 1 :  $M$  по полю КОД\_ТОВ с обеспечением целостности.

### **ЗАМЕЧАНИЕ**

Если в схеме данных не определена связь между таблицами, но таблицы имеют поля с одинаковым именем и одинаковым типом данных, Access может автоматически устано-

вить эту связь. Связи, которые не были установлены Access автоматически, может установить пользователь прямо в схеме данных запроса, перетащив задействованные в связи поля из списка полей одной таблицы в список полей другой.

- Щелчком мыши выделите линию связи таблиц и нажмите правую кнопку. В контекстном меню выберите **Параметры объединения** (Join Properties). В открывшемся окне для связываемых таблиц по умолчанию выбран первый способ объединения (рис. 4.22).



**Рис. 4.22.** Окно выбора способа объединения связываемых таблиц

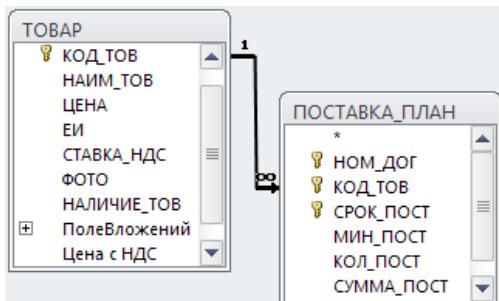
- Перетащите в бланк запроса из таблицы ПОСТАВКА\_ПЛАН поле КОД\_ТОВ и КОЛ\_ПОСТ, а из таблицы ТОВАР — поле НАИМ\_ТОВ.

### ЗАМЕЧАНИЕ

В ряде случаев в бланке запроса наряду с именем поля нужно отображать имя соответствующей таблицы, например, когда поля имеют одинаковые имена в разных таблицах. Для того чтобы в бланке запроса, наряду с именем поля, видеть имя таблицы, выполните команду **Имена таблиц** (Table Names) на ленте конструктора в группе **Показать или скрыть** (Show/Hide).

- Выполните запрос, щелкнув на кнопке **Выполнить** (Run) или **Режим** (View) на ленте конструктора запросов в группе **Результаты** (Results). Число записей в таблице запроса будет равно числу записей в подчиненной таблице ПОСТАВКА\_ПЛАН.
- Если для связи таблиц не было задано параметра обеспечения целостности данных, в таблицу ПОСТАВКА\_ПЛАН можно было бы включить записи с кодами товара, не содержащимися в таблице ТОВАР. Такие записи не были бы включены в таблицу запроса, т. к. первый способ объединения требует наличия связанных записей в таблице ТОВАР. Таким образом, по ряду товаров необходимые сведения были бы потеряны.
- Снова откройте для связи таблиц окно **Параметры объединения** (Join Properties) (см. рис. 4.22) и выберите второй способ объединения записей. Об-

ратите внимание: на линии связи появилась стрелка, направленная от таблицы ТОВАР к таблице ПОСТАВКА\_ПЛАН (рис. 4.23). Она указывает на необходимость включения в таблицу запроса и тех записей из таблицы ТОВАР, для которых нет связанных в ПОСТАВКА\_ПЛАН.



**Рис. 4.23.** Схема данных запроса при объединении таблиц вторым способом

Код товара	Наименование товара	Количество
T001	Монитор 17LG	10
T001	Монитор 17LG	300
T001	Монитор 17LG	100
T002	FDD 3,5	50
T002	FDD 3,5	1000
T002	FDD 3,5	80
T002	FDD 3,5	1000
T002	FDD 3,5	2000
T002	FDD 3,5	2345
T003	HDD Maxtor 20GB	100
T003	HDD Maxtor 20GB	10
T003	HDD Maxtor 20GB	14
T004	Корпус MiniTower	30
T005	CD-ROM Panasonic IDE	40
T005	CD-ROM Panasonic IDE	10
T006	DIMM 64M PC100	200
T007	Принтер EPSON ST.A4	10
T008	СканерAcer	4
	Зв. Карта Genius Liv	
T010	Модем Genius ext	400
*		

**Рис. 4.24.** Результат объединения таблиц ТОВАР и ПОСТАВКА\_ПЛАН вторым способом

10. Выполните запрос. При объединении таблиц вторым способом к записям, полученным первым способом, добавились записи из таблицы ТОВАР, не имеющие связанных записей в таблице ПОСТАВКА\_ПЛАН. Каждая такая запись дополнена полями Количество и Код товара из таблицы ПОСТАВКА\_ПЛАН,

которые содержат значения `Null`. Такая ситуация возникла вследствие того, что в плане отсутствуют заказы на ряд товаров. На рис. 4.24 одна такая запись, для нее поля с количеством и кодом товара остались пустыми.

### **ОБРАТИТЕ ВНИМАНИЕ**

`Null` — это константа, которая означает, что поле не содержит данных. Поле получает значение `Null`, когда неизвестно его содержимое. Такое поле не следует путать с полем, содержащим пустую строку, хотя выглядят они одинаково. Значение `Null` можно ввести в поле или использовать в выражениях и запросах для указания отсутствующих или неизвестных данных. Пустая строка служит для указания того, что строковое значение для этого поля отсутствует. Для ввода пустой строки с клавиатуры следует ввести два символа прямых кавычек без пробела ("").

11. При выборе третьего способа объединения записей в таблице запроса к записям, полученным первым способом, добавились бы записи из таблицы ПОСТАВКА\_ПЛАН, не имеющие связанных записей в таблице ТОВАР. Каждая такая запись дополнена полем НАИМ\_ТОВ, которые содержат значения `Null`. Такая ситуация возникает, если в таблицу ТОВАР не внесены сведения о новых товарах или договор заключен на несуществующий товар. В рассматриваемом примере в таблице ПОСТАВКА\_ПЛАН не может существовать записей с указанием товаров, отсутствующих в таблице ТОВАР, т. к. для связи этих таблиц установлен параметр обеспечения целостности.
12. Рассмотрев, как формируются записи запроса, перейдите к выполнению их группировки. Это позволит решить поставленную задачу. Выполните команду **Итоги (Totals)** из группы **Показать или скрыть (Show/Hide)**. В бланке запроса в строке **Групповая операция (Total)** в столбце КОЛ\_ПОСТ замените слово **Группировка (Group By)** на функцию `Sum`.

Код товара	Наименование товара	Sum-КОЛ_ПОСТ
	Зв. Карта Genius Liv	
T001	Монитор 17LG	1227
T002	FDD 3,5	6475
T003	HDD Maxtor 20GB	124
T004	Корпус MiniTower	30
T005	CD-ROM Panasonic IDE	50
T006	DIMM 64M PC100	200
T007	Принтер EPSON ST.A4	10
T008	СканерAcer	4
T010	Модем Genius ext	400

Запись: 1 10 из 10 11 12 ▶ Нет фильтра Поиск

**Рис 4.25.** Результат группировки при объединении записей таблиц вторым способом

13. Выполните запрос. На рис. 4.25 приведена таблица запроса, полученная путем объединения записей таблиц ПОСТАВКА\_ПЛАН и ТОВАР вторым способом и

последующей их группировки. Таблица содержит не только суммарное количество каждого из заказанных товаров. При выбранном способе объединения записей в результат попадают и те товары, по которым не было сделано заказов.

14. Чтобы отобрать только те товары, по которым не сделано заказов, введите в столбец, где подсчитывается суммарное количество, в строку условие отбора значение `Null`. Выполните запрос, чтобы проверить результат.

## Параметры объединения в инструкциях SQL

1. Откройте в режиме конструктора созданный в предыдущем упражнении запрос, где для исходных таблиц установлен второй способ объединения и для поля `КОЛ_ПОСТ` задано условие отбора `Is Null`.
2. Перейдите в режим SQL, выбрав его из списка кнопки **Режим (View)**. Инструкция SQL для этого запроса будет иметь вид:

```
SELECT ПОСТАВКА_ПЛАН.КОД_ТОВ, ТОВАР.НАИМ_ТОВ,
       Sum(ПОСТАВКА_ПЛАН.КОЛ_ПОСТ) AS [Sum-КОЛ_ПОСТ]
  FROM ТОВАР LEFT JOIN ПОСТАВКА_ПЛАН
    ON ТОВАР.КОД_ТОВ = ПОСТАВКА_ПЛАН.КОД_ТОВ
 GROUP BY ПОСТАВКА_ПЛАН.КОД_ТОВ, ТОВАР.НАИМ_ТОВ
 HAVING ((Sum(ПОСТАВКА_ПЛАН.КОЛ_ПОСТ)) Is Null);
```

Данная инструкция `SELECT` использует для формирования структуры таблицы запроса помимо двух полей из таблиц базы данных, статистическую функцию `Sum`. С помощью предложения `AS` полю, формируемому этой функцией, присваивается альтернативное имя.

### ЗАМЕЧАНИЕ

Имя поля, введенное в его свойство **Подпись** (`Caption`), автоматически не переносится в инструкцию SQL. Чтобы в инструкции `SELECT` в предложении `AS` полю присваивалось альтернативное имя, оно должно быть в бланке запроса введено перед именем поля в следующем виде `Заказано товаров:КОЛ_ПОСТ`.

Предложение `FROM` данной инструкции `SELECT` определяет таблицы, которые служат источником данных для данного запроса, и способ их объединения `LEFT JOIN`. При этом указывается по какому полу связываются таблицы — `КОД_ТОВ`:  
`ON ТОВАР.КОД_ТОВ = ПОСТАВКА_ПЛАН.КОД_ТОВ`

Операция `LEFT JOIN` используется для создания левого внешнего объединения. Левое внешнее объединение включает помимо связанных записей все записи из первой (левой) таблицы, даже если для них нет связанных записей во второй (правой) таблице.

1. Вернитесь в режим конструктора и измените способ объединения таблиц запроса, выбрав вместо второго параметра объединения третий. Предложение `FROM` для такого запроса будет иметь вид:

```
FROM ТОВАР RIGHT JOIN ПОСТАВКА_ПЛАН
  ON ТОВАР.КОД_ТОВ = ПОСТАВКА_ПЛАН.КОД_ТОВ
```

Операция `RIGHT JOIN` используется для создания правого внешнего объединения. Правое внешнее объединение включает помимо связанных записей все записи из второй (правой) таблицы, даже если для них нет связанных записей в первой (левой) таблице.

Если просмотреть запрос, в котором выбран первый способ объединения таблиц, то в предложении `FROM` будет указана соответствующая операция `INNER JOIN`.

Предложение `HAVING` задает условия отбора групп, определяя какие из сгруппированных записей следует включить в таблицу запроса. Предложение `HAVING` аналогично предложению `WHERE`, которым определяется выбор записей.

1. В режиме конструктора дополните запрос полем **ЦЕНА**, в строке **Групповая операция (Total)** замените слово **Группировка (Group By)** словом **Условие (Where)** и введите для него условие отбора `>1000`. В поле с суммарным количеством товара удалите в условии отбора значение `is Null`. Инструкция `SELECT` будет дополнена предложением `WHERE`.
2. В условие отбора поля с суммарным количеством товара введите `>40`, будет выполнен отбор соответствующих сгруппированных записей. Инструкция `SELECT` будет дополнена предложением `HAVING` и примет вид:

```
SELECT ПОСТАВКА_ПЛАН.КОД_ТОВ, ТОВАР.НАИМ_ТОВ,
       Sum(ПОСТАВКА_ПЛАН.КОЛ_ПОСТ) AS [Sum-КОЛ_ПОСТ]
  FROM ТОВАР RIGHT JOIN ПОСТАВКА_ПЛАН
    ON ТОВАР.КОД_ТОВ = ПОСТАВКА_ПЛАН.КОД_ТОВ
 WHERE ((ТОВАР.ЦЕНА)>1000)
 GROUP BY ПОСТАВКА_ПЛАН.КОД_ТОВ, ТОВАР.НАИМ_ТОВ
 HAVING ((Sum(ПОСТАВКА_ПЛАН.КОЛ_ПОСТ))>40);
```

При выполнении такого запроса сначала в соответствии с предложением `WHERE` отбираются формируемые записи, затем они группируются с помощью предложения `GROUP BY`, и после этого в соответствии с предложением `HAVING` производится отбор сгруппированных записей.

## Ссылки на имена полей различных таблиц в условии отбора

В условии отбора в качестве операндов могут использоваться не только конкретные значения для отбора по полям, а и ссылки на имена полей таблиц, на которых основывается запрос.

**Задача.** Пусть необходимо выбрать записи из таблицы **ОТГРУЗКА**, в которых указанная стоимость товара не соответствует произведению количества отгруженного на цену, указанную для этого товара в таблице **ТОВАР**, и рассчитать величину отклонения.

1. Для решения этой задачи создайте запрос, представленный на рис. 4.26.
2. Для отбора записей с неверно указанной суммой отгрузки в строке **Условие отбора (Criteria)** для поля **СУММА\_ОТГР** запишите выражение:  

$$<> [КОЛ_ОТГР] * [ЦЕНА]$$

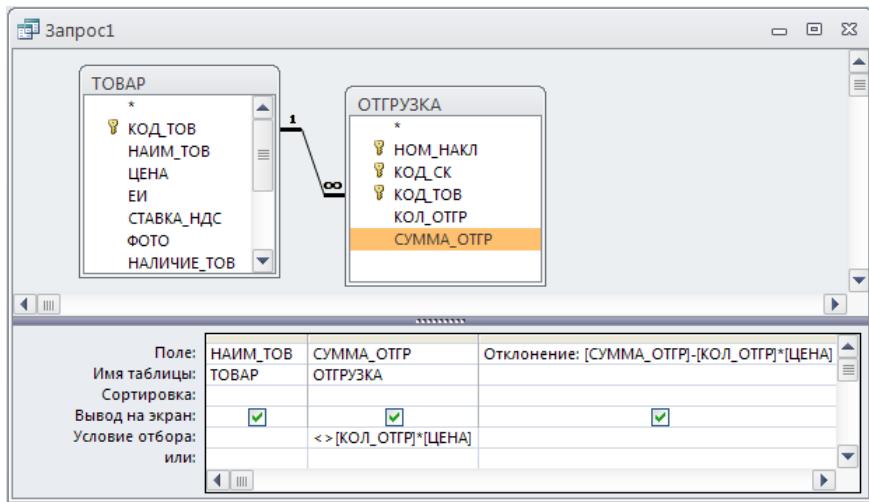


Рис. 4.26. Запрос с использованием в условиях отбора имен полей из разных таблиц

Указание имен таблиц в ссылках на поля является необязательным, потому что поля имеют уникальные имена. В противном случае при ссылке на поле перед его именем необходимо указывать имя таблицы, например `[ТОВАР]![ЦЕНА]`.

1. Для расчета величины, на которую отличается значение суммы в таблице от правильного, создайте вычисляемое поле, записав выражение:  

$$[\text{СУММА\_ОТГР}] - [\text{КОЛ\_ОТГР}] * [\text{ЦЕНА}]$$
2. Замените сформированное по умолчанию имя вычисляемого поля **Выражение1** на **Отклонение**.

Результаты выполнения запроса, в который добавлены еще поля с номером накладной НОМ\_НАКЛ и кодом склада КОД\_СК, приведены на рис. 4.27.

Код накладн.	Код склада	Наименование товара	Сумма отгружено	Отклонение
H001	C02	FDD 3,5	0,00р.	-500,00р.
H001	C01	FDD 3,5	0,00р.	-10,00р.
H001	C03	DIMM 64M PC100	1 800,00р.	-1 200,00р.
H001	C01	СканерAcer	2 338,00р.	-21 042,00р.
H001	C01	Принтер EPSON ST.A4	22 000,00р.	-2 000,00р.
H002	C01	FDD 3,5	0,00р.	-330,00р.
H002	C02	FDD 3,5	0,00р.	-55,00р.
H002	C03	DIMM 64M PC100	360,00р.	60,00р.
H004	C02	Корпус MiniTower	0,00р.	-1 000,00р.
*				

Рис. 4.27. Записи об отгрузке товаров с неверно указанной суммой

## ЗАМЕЧАНИЕ

Если результат выполнения запроса не содержит записей, то это означает, что все суммы подсчитаны правильно, т. е. равны произведению количества на цену товара.

Просмотрите запрос в режиме SQL. Инструкция SELECT для этого запроса имеет вид:

```
SELECT ОТГРУЗКА.НОМ_НАКЛ, ОТГРУЗКА.КОД_СК, ТОВАР.НАИМ_ТОВ,
       ОТГРУЗКА.СУММА_ОТГР, [СУММА_ОТГР] - [КОЛ_ОТГР]*[ЦЕНА] AS Отклонение
  FROM ТОВАР INNER JOIN ОТГРУЗКА ON ТОВАР.КОД_ТОВ = ОТГРУЗКА.КОД_ТОВ
 WHERE (((ОТГРУЗКА.СУММА_ОТГР) <> [КОЛ_ОТГР]*[ЦЕНА]));
```

В инструкции SELECT наряду с полями таблиц, включаемыми в таблицу запроса, представлено вычисляемое поле, которому присвоено имя **Отклонение**.

### Задание 4.1. Создание многотабличного запроса

Подсчитайте суммарное количество каждого из товаров, которое должно быть поставлено заданному покупателю в каждом из месяцев. Результат должен содержать наименование покупателя, месяц поставки, наименование заказанного товара и количество. Используйте таблицы ПОСТАВКА\_ПЛАН, ТОВАР, ПОКУПАТЕЛЬ, а также таблицу ДОГОВОР, через которую осуществляется связь таблицы ПОСТАВКА\_ПЛАН с таблицей ПОКУПАТЕЛЬ.

### Задание 4.2. Создание многотабличного запроса с параметрами

Подсчитайте общее количество каждого из товаров, отгруженных в указанный период. Результат должен содержать наименование товара, количество отгруженного. Используйте таблицы ОТГРУЗКА, ТОВАР и НАКЛАДНАЯ. Основой образования записей этого запроса является таблица ОТГРУЗКА. Код товара в таблице ОТГРУЗКА может быть заменен на наименование, выбранное из связанной записи таблицы ТОВАР. Датой отгрузки товара запись из таблицы ОТГРУЗКА может быть дополнена из связанной записи в таблице НАКЛАДНАЯ.

## Представление данных нарастающим итогом

Данные о плановых поставках товаров заказчику в соответствии с договорами хранятся в таблице ПОСТАВКА\_ПЛАН. Ранее было показано, как с помощью несложного запроса группировки подсчитать, какое суммарное количество каждого из товаров должно быть поставлено заказчикам (выпущено и отгружено предприятием) в каждом отдельном месяце. Однако часто необходимо предоставлять не только помесячный план поставок, но и план поставок товаров нарастающим итогом. В таком плане указывается суммарное количество товара, которое необходимо поставить к концу каждого месяца от начала года, т. е. количество от месяца к месяцу будет нарастать. На рис. 4.28 представлены результаты выполнения запросов,

содержащие данные о плане поставок товаров в каждом месяце и нарастающим итогом. Рассматривается план поставок на три месяца.

**Заказано в каждом месяце**

Код товара	Срок поставки	Sum-КОЛ_ПОСТ
T001	1	415
T001	2	412
T001	3	400
T002	1	1050
T002	2	2425
T002	3	3000
T003	1	114
T003	2	10
T004	3	30
T005	1	50
T006	2	200
T007	2	10
T008	1	4
T010	2	400

Запись: 1 из 14 | Нет фильтра | Поиск

**Наращающий итог**

Код товара	Номер месяца	Sum-КОЛ_ПОСТ
T001	1	415
T001	2	827
T001	3	1227
T002	1	1050
T002	2	3475
T002	3	6475
T003	1	114
T003	2	124
T003	3	124
T004	3	30
T005	1	50
T005	2	50
T005	3	50
T006	2	200
T006	3	200
T007	2	10
T007	3	10
T008	1	4
T008	2	4
T008	3	4
T010	2	400
T010	3	400

Запись: 22 из 22 | Нет фильтра | Поиск

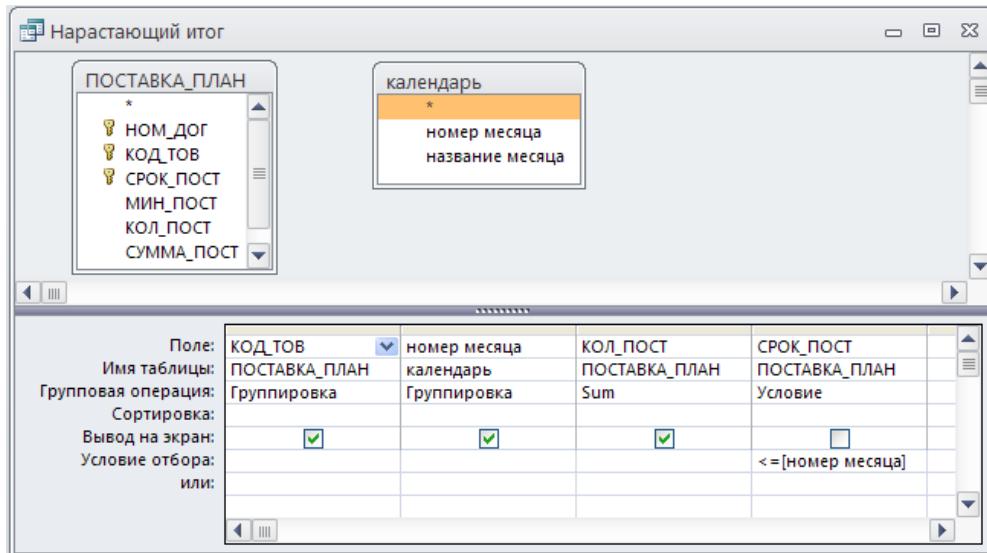
**Рис. 4.28.** Данные о плане поставок товаров в каждом месяце и нарастающим итогом

Создайте запрос, позволяющий получить план поставок товаров нарастающим итогом (рис. 4.29).

1. Прежде чем конструировать запрос создайте дополнительную таблицу КАЛЕНДАРЬ, которая содержит строки с номерами и наименованиями трех месяцев.
2. Включите эту таблицу в запрос наряду с таблицей ПОСТАВКА\_ПЛАН.
3. Между таблицами не устанавливайте никакого способа объединения записей. Это приведет к выполнению операции декартово произведение записей таблиц. При этом каждая строка таблицы ПОСТАВКА\_ПЛАН будет дополнена номером месяца из календаря и включена в результат столько раз, сколько месяцев в календаре, т. е. если в таблице ПОСТАВКА\_ПЛАН 5 строк, декартово произведение даст  $5 \times 3$  строк, т. к. в календаре содержится три строки.
4. Включите в бланк запроса поля, необходимые для выполнения отбора нужных строк, группировки и подсчета суммарных количеств в группах (рис. 4.29).
5. Введите в поле СРОК\_ПОСТ условие отбора  $<=[\text{номер месяца}]$ , чтобы оставить только те строки, в которых срок поставки товара меньше или равен номеру

ру месяца из календаря. Это позволит получить для каждого месяца в календаре группу, включающую строки поставок товара за все предыдущие месяцы.

6. Выполните группировку по коду товара и номеру месяца.
7. Чтобы подсчитать суммарное количество каждого из товаров, которое должно быть выпущено за период до конца каждого месяца, выберите функцию Sum для поля с количеством.



**Рис. 4.29.** Запрос для представления плана поставок изделий нарастающим итогом в режиме конструктора

Чтобы представить работу алгоритма вычисления нарастающего итога в более наглядном виде, сократите число строк в таблицах ПОСТАВКА\_ПЛАН и КАЛЕНДАРЬ. Пусть в таблице ПОСТАВКА\_ПЛАН хранится пять строк, представляющих план поставок изделия на три месяца, а в таблице КАЛЕНДАРЬ — три строки, представляющие три месяца (рис. 4.30).

ПОСТАВКА_ПЛАН				КАЛЕНДАРЬ	
Номер договора	Код товара	Срок поставки	Количество	Номер месяца	Название месяца
Д111	T001	1	100	1	январь
Д111	T001	2	20	2	февраль
Д111	T001	3	30	3	март
Д222	T001	1	50	*	
Д222	T001	2	10	0	
*		0	0		

**Рис. 4.30.** Таблицы с данными о плане поставок товаров и календарь

Декартово произведение этих таблиц представлено на рис. 4.31 и объединяет каждую строку плана с каждым месяцем календаря. Там же показан результат выполнения операции отбора записей, содержащий только 11 строк. В соответствии с заданным условием отбора из декартова произведения остаются только те строки, в которых срок поставки меньше или равен номеру месяца из календаря.

The screenshot shows two Microsoft Access query windows side-by-side. Both windows have the title bar 'Запрос1' and 'Запрос2' respectively.

**Запрос1 (Left Window):**

Код товара	Срок поставки	Количество	Номер месяца
T001	1	100	1
T001	1	100	2
T001	1	100	3
T001	2	20	1
T001	2	20	2
T001	2	20	3
T001	3	30	1
T001	3	30	2
T001	3	30	3
T001	1	50	1
T001	1	50	2
T001	1	50	3
T001	2	10	1
T001	2	10	2
T001	2	10	3

**Запрос2 (Right Window):**

Код товара	Номер месяца	Количество
T001	1	100
T001	2	100
T001	3	100
T001	2	20
T001	3	20
T001	3	30
T001	1	50
T001	2	50
T001	3	50
T001	2	10
T001	3	10

Both windows include a status bar at the bottom with navigation buttons and search fields.

Рис. 4.31. Декартово произведение таблиц ПОСТАВКА\_ПЛАН и КАЛЕНДАРЬ до и после отбора записей в соответствии с условиями

В результате группировки по коду товара и номеру месяца из календаря формируется три группы по числу месяцев в календаре. Таким образом, для товара будет получено столько строк, сколько месяцев включает календарь, а суммарное количество будет равно запланированному к выпуску на конец каждого месяца (рис. 4.32).

The screenshot shows a Microsoft Access query window titled 'Нарастающий итог' (Running Total).

Код товара	Номер месяца	Sum-КОЛ_ПОСТ
T001	1	150
T001	2	180
T001	3	210

The status bar at the bottom includes navigation buttons and a search field.

Рис. 4.32. План поставок товара нарастающим итогом

### ВНИМАНИЕ!

При выполнении запроса сначала производится объединение записей и их отбор в соответствии с заданными условиями и только затем группировка полученных записей.

Просмотрите запрос в режиме SQL. Инструкция SELECT для этого запроса имеет вид:

```
SELECT ПОСТАВКА_ПЛАН.КОД_ТОВ, календарь.[номер месяца],
Sum(ПОСТАВКА_ПЛАН.КОЛ_ПОСТ) AS [Sum-КОЛ_ПОСТ]
```

```
FROM ПОСТАВКА_ПЛАН, календарь
WHERE (((ПОСТАВКА_ПЛАН.СРОК_ПОСТ)<=[номер месяца]))
GROUP BY ПОСТАВКА_ПЛАН.КОД_ТОВ, календарь.[номер месяца];
```

В инструкции `SELECT` в предложении `FROM ПОСТАВКА_ПЛАН, календарь` имена таблиц разделены запятой, что и определяет операцию декартова произведения для объединения записей таблиц. Предложение `WHERE ПОСТАВКА_ПЛАН.СРОК_ПОСТ<=[номер месяца]` определяет отбор нужных записей. И наконец, предложение `GROUP BY` определяет поля группировки. Суммирование количества для группы выполняется соответствующей функцией, указанной в списке полей инструкции.

Тот же результат может быть получен, если в инструкции SQL заменить приведенные в примере предложения `FROM` и `WHERE` одним предложением `FROM ПОСТАВКА_ПЛАН INNER JOIN календарь ON ПОСТАВКА_ПЛАН.СРОК_ПОСТ<=календарь.[номер месяца]`. Однако такая инструкция не может быть представлена в режиме конструктора Access, т. к. в этом режиме в выражении объединения `JOIN` не поддерживается использование знака сравнения, отличного от `=`.

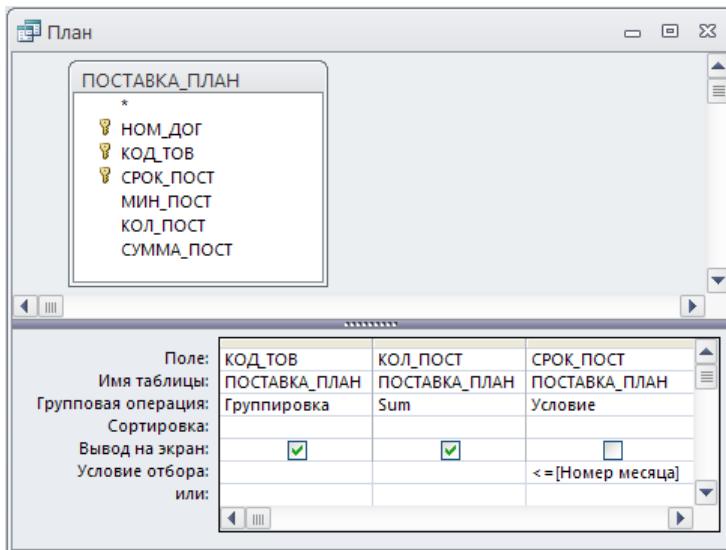
## **Решение задачи, требующей выполнения нескольких запросов**

Чтобы решить сложную задачу, чаще всего необходимо разбить ее на несколько подзадач, алгоритм каждой из которых может быть реализован выполнением одного достаточно простого запроса. Для каждого из запросов определяются входные и выходные данные. В простейшем случае выходные данные предшествующего запроса являются входными для следующего, построенного на нем запроса, и, выполнив только последний запрос в цепочке построенных друг на друге запросов, вы инициируете последовательное выполнение всех запросов цепочки и полное решение задачи. Для представления алгоритмов решения задач целесообразно использовать функционально-технологическую схему, на которой приводится цепочка всех запросов с описанием и входными и выходными данными.

**Задача.** Проанализируйте выполнение общего плана поставок каждого из товаров на конец заданного месяца. При решении этой задачи подсчитайте количество товара, запланированного к поставке, отгруженного покупателям, и получите величину недопоставки товаров на конец заданного месяца.

1. Данные о плановых поставках хранятся в таблице ПОСТАВКА\_ПЛАН. На ее основе создайте запрос для подсчета суммарного количества каждого из товаров, запланированных к поставке на конец заданного месяца (рис. 4.33).
2. Сохраните запрос под именем **План**.

В запросе **План** производится группировка записей таблицы по полю код товара. Для операции используются только записи, в которых срок поставки (месяц) имеет значение меньшее или равное заданному параметром запроса с именем **Номер месяца**.



**Рис. 4.33.** Подсчет суммарного количества товаров, запланированных к поставке

В каждой группе записей о поставках одного товара суммируются количества, запланированные к поставке.

1. Данные о фактически поставленных, отгруженных покупателям товарах хранятся в таблице ОТГРУЗКА. На ее основе создайте запрос для подсчета суммарного количества фактически поставленных товаров. Для отбора поставок, выполненных до конца заданного месяца, в запрос добавьте таблицу НАКЛАДНАЯ, в которой хранится дата отгрузки товаров (рис. 4.34).
2. Сохраните запрос под именем **Факт**.

Таблицы, на которых построен запрос **Факт**, находятся в отношении один-ко-многим. Для их связи по составному ключу установлены параметры обеспечения целостности. В результате объединения этих таблиц первым способом формируется таблица запроса с числом записей, равным числу записей в подчиненной таблице ОТГРУЗКА. Причем каждая запись об отгрузках товара дополняется датой из связанной записи главной таблицы НАКЛАДНАЯ.

1. Для отбора только тех накладных, по которым отгружался товар до конца заданного месяца, из даты отгрузки с помощью функции Month выделяется номер месяца, и для этого вычисляемого поля в условие отбора вводится параметр запроса с именем **Номер месяца**, совпадающим с именем параметра в предыдущем запросе.
2. Для сравнения количества запланированного к поставке и отгруженного создайте новый запрос. Добавьте в него два предыдущих запроса **План** и **Факт**.
3. Поскольку некоторые из запланированных товаров могли не отгружаться и в то же время могла производиться отгрузка товаров, которые не были запланированы, добавьте в запрос таблицу ТОВАР, в которой представлена вся номенклатура товаров фирмы.

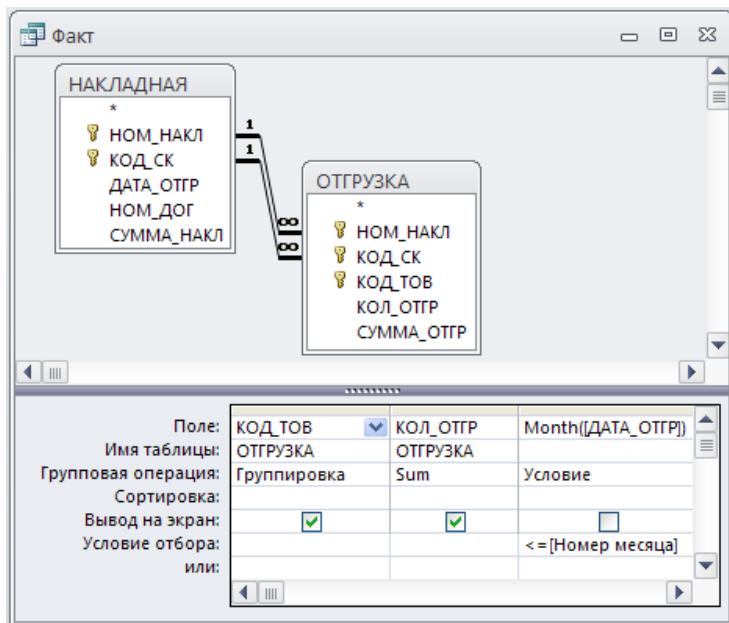


Рис. 4.34. Подсчет суммарного количества отгруженных товаров

Свяжите эту таблицу с каждым из запросов по полю КОД\_ТОВ и укажите на объединение ее записей с записями таблиц запросов План и Факт вторым способом (рис. 4.35).

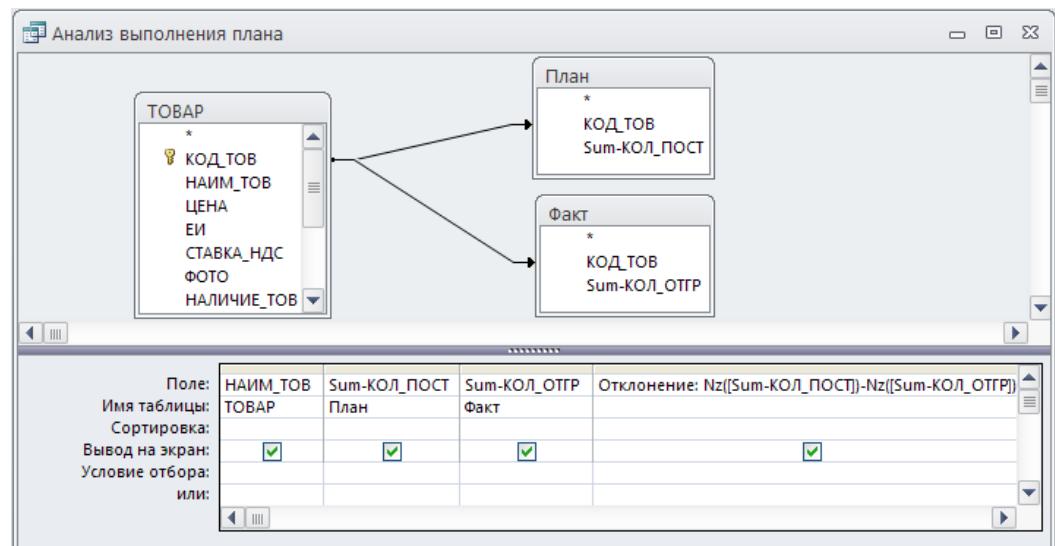


Рис. 4.35. Вычисление разности плановых и фактических поставок товаров

Такое объединение таблиц приведет к формированию результата, в котором будут представлены все записи о товарах:

- ◆ по которым был определен план поставки, даже если эти товары не отгружались вовсе;
  - ◆ которые отгружались, несмотря на то, что не были запланированы к поставке;
  - ◆ которые вообще не планировались к поставке и не отгружались.
4. Обратите внимание, при формировании записи вторым способом объединения в таблицу запроса включаются и те записи из левой таблицы (ТОВАР), для которых нет связанных записей в правой таблице (например, **План**). При этом в формируемой записи поля, выбираемые из правой таблицы, заполняются значением Null.
  5. Учитывая, что во всех пустых полях таблицы запроса определено значение Null, для получения разности между запланированным и фактически поставленным количеством товара создайте вычисляемое поле с выражением  $Nz([Sum-КОЛ_ПОСТ]) - Nz([Sum-КОЛ_ОТГР])$ . Присвойте вычисляемому полю имя **Отклонение**. Функция Nz возвращает существующее в поле значение или для поля со значением Null новое значение, указанное вторым аргументом. Если второй аргумент не указан, по умолчанию для числового поля со значением Null возвращается 0, для символьного пустая строка.
  6. Результатом арифметических операций с полем Null является Null. Убедитесь в этом, исключив из выражения функцию Nz.
  7. Сохраните запрос под именем **Анализ выполнения плана**.
  8. Выполните запрос. В таблице запроса **Анализ выполнения плана** (рис. 4.36) представлен весь список товаров фирмы. Показано, что не все товары были заказаны в договорах, только по ряду заказанных товаров выполнялась отгрузка, а некоторые товары отгружались без предварительного оформления договоров.

Наименование товара	Sum-КОЛ_ПОСТ	Sum-КОЛ_ОТГР	Отклонение
Монитор 17LG	1122	23	1099
FDD 3,5	4130	25	4105
HDD Maxtor 20GB	224	12	212
Корпус MiniTower	30		30
CD-ROM Panasonic IDE	40	40	0
DIMM 64M PC100	200	6	194
Принтер EPSON ST.A4		10	-10
СканерAcer	4	23	-19
3в. Карта Genius Liv		13	-13
Модем Genius ext			0

Запись: 1 из 10    Нет фильтра    Поиск

Рис. 4.36. Результат вычисления разности плановых и фактических поставок товаров

## **ВНИМАНИЕ!**

Выполнение запроса **Анализ выполнения плана** инициирует выполнение запросов **План** и **Факт**, и нет необходимости в их предварительном выполнении.

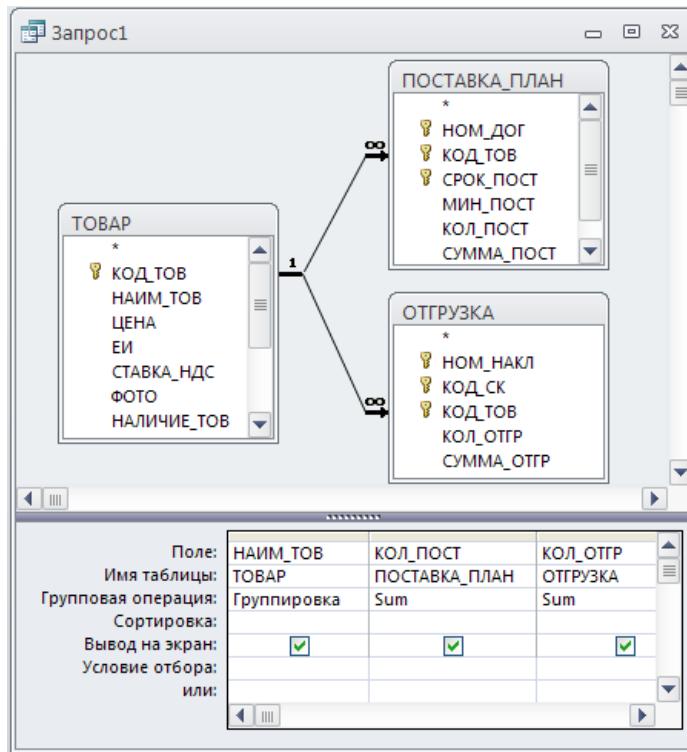
9. Запишите в условие отбора поля `Sum-КОЛ_ПОСТ` значение `Not Null`. Отобразятся только те строки, которые относятся к товарам, на которые были заключены договоры.
10. Для того чтобы явно увидеть, в какой последовательности производится объединение таблиц запроса, просмотрите запрос в режиме SQL. Инструкция `SELECT` для этого запроса имеет вид:

```
SELECT ТОВАР.НАИМ_ТОВ, План.[Sum-КОЛ_ПОСТ], Факт.[Sum-КОЛ_ОТГР],
       NZ([Sum-КОЛ_ПОСТ])-NZ([Sum-КОЛ_ОТГР]) AS Отклонение
  FROM (ТОВАР LEFT JOIN План ON ТОВАР.КОД_ТОВ = План.КОД_ТОВ)
        LEFT JOIN Факт ON ТОВАР.КОД_ТОВ = Факт.КОД_ТОВ;
```

В предложении `FROM` указано, что сначала будет производиться левое внешнее объединение таблиц ТОВАР и План по полю связи код товара (`ТОВАР LEFT JOIN План ON ТОВАР.КОД_ТОВ = План.КОД_ТОВ`). Затем будет производиться левое внешнее объединение таблицы, полученной в результате первого объединения, и таблицы Факт по тому же полю связи.

Объединение таблиц в рассматриваемом запросе дает правильный результат только потому, что в каждой из объединяемых таблиц КОД\_ТОВ имеет уникальные неповторяющиеся значения. В таблице ТОВАР поле КОД\_ТОВ определено каковым изначально. В таблицах запросов **План** и **Факт** поле КОД\_ТОВ также имеет уникальные значения, т. к. они получены в результате группировки по этому полю.

1. Попробуйте подсчитать суммарное количество плановых и фактических поставок товаров одним запросом, представленным на рис. 4.37.
2. Проанализируйте результат выполнения запроса и убедитесь, что он неправильный. Неверный результат образуется вследствие того, что в многотабличных запросах выполнение группировки производится после формирования записей путем объединения записей таблиц и их выборки в соответствии с условиями отбора.
3. Объединение записей таблиц ПОСТАВКА\_ПЛАН и ОТГРУЗКА по полю КОД\_ТОВ, значения которого многократно повторяются в каждой из таблиц, приводит к неоднократному включению в результат как записей из первой таблицы, так и записей из второй таблицы. Например, в результате объединения приведенных на рис. 4.38 таблиц будет получена таблица, в которой количество запланированного к поставке будет повторено дважды и при группировке будет получено удвоенное количество плановой поставки. Если в первой таблице будут неоднократно встречаться записи по одному и тому же товару, в результат объединения будут повторно включаться одни и те же записи об отгрузках. Это приведет при группировке к неверному подсчету количества отгруженного.



**Рис. 4.37.** Запрос, объединяющий записи таблиц с повторяющимися значениями в поле связи

Таблица ПОСТАВКА\_ПЛАН

КОД_ТОВ	КОЛ_ПОСТ
T1	100
T2	100

Таблица ОТГРУЗКА

КОД_ТОВ	КОЛ_ОТГР
T1	10
T1	50
T2	50
T2	60

Результат объединения

КОД_ТОВ	КОЛ_ПОСТ	КОЛ_ОТГР
T1	100	10
T1	100	50
T2	100	50
T2	100	60

**Рис. 4.38.** Пример объединения записей таблиц

**Задание 4.3.** Проанализируйте план поставок товара заданному покупателю.

**Задание 4.4.** Проанализируйте план поставок товара в стоимостном выражении.

**Задание 4.5.** Подсчитайте, на какую сумму отгружен товар покупателям, которые не заключали договоров.

**Задание 4.6.** Подсчитайте, на какую сумму покупатели недополучили товар на текущую дату.

## Контрольные вопросы

1. Какой способ объединения записей двух таблиц по умолчанию устанавливается при их связи?
2. Мешает ли флажок **Обеспечение целостности данных** (Enforce Referential Integrity) установлению параметра объединения?
3. В каком окне, какой кнопкой открывается возможность выбора параметра объединения?
4. Запрос построен на двух таблицах, находящихся в отношениях  $1 : M$ , для которых установлен параметр обеспечения целостности. Первая таблица имеет пять записей, а вторая 10? Причем 10 записей второй таблицы подчинены 3-м записям первой. Сколько записей будет включено в таблицу запроса, если для исходных таблиц выбран первый способ объединения записей и если — второй?
5. Изменится ли параметр объединения таблиц, установленный в запросе, если изменить его в схеме данных базы?
6. Какая команда позволяет отобразить в бланке запроса строку с именами таблиц?
7. Какие поля представлены в списке, открывающемся в строке **Поле** (Field) бланка запроса?
8. Сколько записей будет содержать таблица запроса, если она построена на двух несвязанных таблицах по 5 записей?
9. Какая операция будет применена к записям объединяемых в запросе таблиц, если между ними не установлена связь?
10. Как вводится параметр в запрос?
11. Как указать на использование конкретного поля, если в таблицах, на которых строится запрос, имеются одинаковые имена?
12. Допускается ли группировка записей запроса по нескольким полям?
13. Нужно ли последовательно выполнять запрос1, запрос2, запрос3, если запрос3 построен на запросе2, а запрос2 на запросе1?
14. В каком предложении и какой операцией задается способ объединения записей таблиц в инструкции SQL?
15. После какого ключевого слова в предложении FROM указываются поля связи таблиц?
16. Как в предложении FROM указывается вложенность операций объединения таблиц?
17. Можно ли в инструкции SELECT в качестве значения условия отбора использовать параметр?

18. Каким предложением определяется необходимость в сортировке?
19. Какая операция выполняется первой в многотабличном запросе с группировкой: объединение записей таблиц или группировка?
20. Как устанавливаются параметры объединения таблиц в запросе?

## Ответы

1. Первый, при котором объединяются только те записи, в которых связанные поля обеих таблиц совпадают.
2. Нет.
3. В схеме данных в окне **Изменение связей** (Edit Relationships) кнопкой **Объединение** (Join Type).
4. 10 и 12 соответственно.
5. Нет.
6. Команда **Имена таблиц** (Table Names) в группе **Показать или скрыть** (Show/Hide) на вкладке конструктора запросов.
7. Поля всех таблиц запроса.
8. 25.
9. "Полное объединение" или "декартово произведение".
10. Как текстовая строка, заключенная в квадратные скобки.
11. Перед именем поля поставить имя таблицы и разделить их восклицательным знаком.
12. Да.
13. Нет. Чтобы получить результат, достаточно выполнить запрос3.
14. В предложении `FROM` операциями `INNER JOIN`, `LEFT JOIN` и `RIGHT JOIN`.
15. `ON`.
16. С помощью круглых скобок.
17. Да.
18. `ORDER BY`.
19. Объединение записей таблиц.
20. С помощью команды **Параметры объединения** (Join Properties) в контекстном меню линии связи таблиц.

## Запросы на изменение

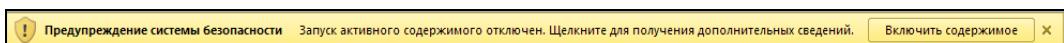
К запросам на изменение относятся запросы на обновление данных в записях таблицы базы, на добавление и удаление записей из таблицы, а также запросы на создание таблицы из записей, сформированных в нем.

Для разработки запросов на изменение в Access используется конструктор. Процесс создания любого запроса на изменение начинается с создания запроса на

выборку, который после добавления в него необходимых таблиц преобразуется в нужный запрос на изменение.

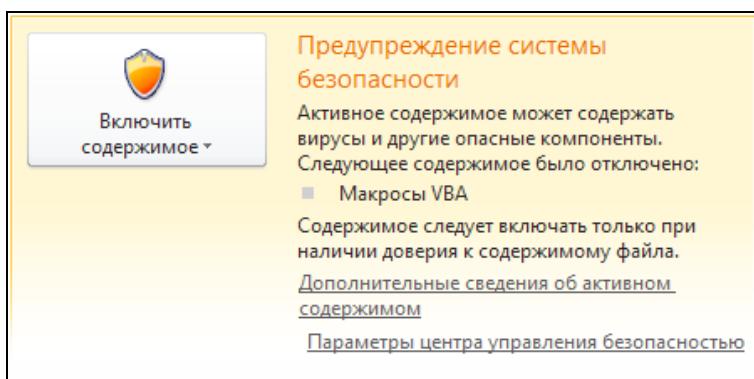
По умолчанию Access 2010 в целях обеспечения безопасности, как правило, блокирует выполнение всех запросов на изменение. Если при выполнении запроса на изменение ничего не происходит, проверьте, не появляется ли в строке состояния Access сообщение: "Данное действие или событие заблокировано в режиме отключения".

Если отображается это сообщение и панель сообщений (Message Bar) (рис. 4.39), для включения заблокированных запросов можно нажать на ней кнопку **Включить содержимое** (Enable content). После этого выполнение запроса будет доступным.



**Рис. 4.39.** Панель сообщений

Если панель сообщений была закрыта и больше не отображается, перейдите на вкладку **Файл** (File) и на открытой странице **Сведения** (Info) в блоке **Предупреждение системы безопасности** (Security Warning) нажмите кнопку **Включить содержимое** (Enable Content) (рис. 4.40).

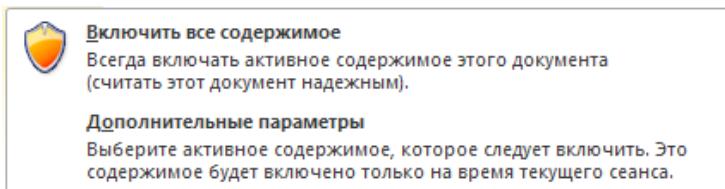


**Рис. 4.40.** Блок включения потенциально опасного содержимого

В открывшемся списке (рис. 4.41) можно **Включить все содержимое** (Enable All Content) открытой базы данных. Это приведет к тому, что при повторных открытиях базы ни панели сообщения, ни сообщений о невозможности выполнения запроса действия выводиться не будет, т. к. база данных будет отнесена к разряду надежных. То же самое происходит, если была нажата кнопка **Включить содержимое** (Enable content) на панели сообщений.

Выбор строки **Дополнительные параметры** (Advanced Options) открывает окно параметров безопасности Microsoft Office, в котором можно включить опасное содержимое только на время сеанса. При следующем открытии базы данных опять

появится панель сообщений и, если не включать содержимое, запросы действия выполняться не будут.



**Рис. 4.41.** Выбор варианта включения потенциально опасного содержимого

Чтобы вернуть возможность управления содержимым, откройте окно **Параметры Access** (Access Options) соответствующей командой на вкладке ленты **Файл** (File), щелкните на строке **Центр управления безопасностью** (Trust Center) и далее по кнопке **Параметры центра управления безопасностью** (Trust Center Settings). В окне центра на странице **Надежные документы** (Trusted Documents) в строке *Сбросить пометку о надежности для всех надежных документов* (Clear all Trusted Documents so that they are no longer trusted) нажмите кнопку **Очистить** (Clear).

### **ВНИМАНИЕ!**

Единое средство вывода предупреждений системы безопасности — панель сообщений — по умолчанию появляется при открытии базы данных Access 2010 вне доверенного расположения. Если точно известно, что можно доверять содержимому базы данных, включите все отключенные потенциально опасные активные компоненты — запросы на изменение, макросы, элементы управления ActiveX, некоторые выражения и программы на VBA — при открытии базы данных, содержащей один или несколько этих компонентов.

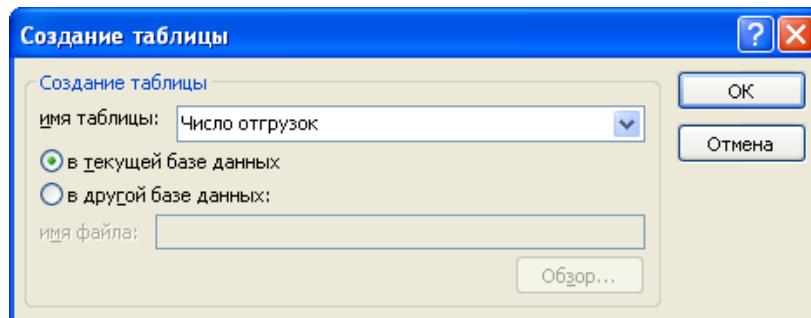
## **Конструирование запроса на создание таблицы**

Запрос на создание таблицы используется для сохранения результата запроса. Этот вид запроса основан на запросе на выборку, но, в отличие от него, сохраняет таблицу с результатами запроса.

Сформируем запрос на создание таблицы на примере ранее полученного запроса на выборку с групповыми вычислениями **Число отгрузок по договорам** (см. рис. 4.15, 4.16).

1. В области навигации выделите названный запрос и с помощью команды контекстного меню откройте его в режиме конструктора.
2. Преобразуйте этот запрос на выборку в запрос на создание таблицы, выполнив команду **Создание таблицы** (Make Table) в группе **Тип запроса** (Query Type) на вкладке конструктора или выбрав команду контекстного меню запроса **Тип запроса | Создание таблицы** (Query Type | Make Table Query).

3. В окне **Создание таблицы** (Make Table) введите имя создаваемой таблицы **Число отгрузок** (рис. 4.42).



**Рис. 4.42.** Определение имени таблицы, создаваемой в запросе

4. Для того чтобы просмотреть, какие записи будут помещены в новую таблицу, щелкните по кнопке **Режим** (View) на ленте конструктора запросов в группе **Результаты** (Results).
5. Выполните запрос, чтобы таблица **Число отгрузок** была сохранена в базе данных. Теперь эту таблицу можно увидеть в списке таблиц области навигации.
6. Перейдите в режим SQL. Эквивалентная запросу на создание таблицы инструкция `SELECT ... INTO` будет записана следующим образом:

```
SELECT НАКЛАДНАЯ.НОМ_ДОГ, Count(НАКЛАДНАЯ.НОМ_ДОГ) AS [Count-НОМ_ДОГ]
INTO [Число отгрузок]
FROM НАКЛАДНАЯ
GROUP BY НАКЛАДНАЯ.НОМ_ДОГ;
```

Инструкция `SELECT ... INTO` аналогично `SELECT` выполняет выборку данных, но, в отличие от нее, для сохранения выбранных данных создает новую таблицу, имя которой указывается в предложении `INTO`. Новая таблица включает перечисленные в `SELECT` поля таблиц, указанных в предложении `FROM`. Допустимо использование предложения `GROUP BY`, определяющего поле группировки. При этом список полей новой таблицы включает поле, вычисляемое с помощью статистической функции `Count`. Если перед предложением `GROUP BY` поместить предложение `WHERE`, группы будут формироваться из записей, отобранных в соответствии с условиями, заданными в предложении `WHERE`.

## Конструирование запроса на обновление

Для обновления данных в полях базовых таблиц может быть использован запрос **Обновление** (Update). Изменения вносятся в группу записей, отбираемых с помощью указанных пользователем условий отбора. Значения для изменений в полях определяются в бланке запроса в строке **Обновление** (Update To).

**Задача.** Рассчитайте стоимость товара в каждой строке таблицы ОТГРУЗКА и сохраните ее в поле СУММА\_ОТГР этой же таблицы.

- Для формирования запроса на обновление сначала создайте запрос **Выборка** (Select) на основе двух таблиц: обновляемой таблицы ОТГРУЗКА и таблицы ТОВАР.
- Преобразуйте запрос на выборку в запрос на обновление, щелкнув на кнопке **Обновление** (Update), размещенной на вкладке ленты **Конструктор** (Design) или выбрав команду **Обновление** (Update) из списка **Тип запроса** (Query Type) в контекстном меню запроса. После выполнения этой команды в бланке запроса появляется строка **Обновление** (Update To) (рис. 4.43).
- Заполните бланк запроса. Перетащите обновляемое поле СУММА\_ОТГР из списка таблицы ОТГРУЗКА в строку **Поле** (Field). В строку **Обновление** (Update To) введите выражение  $[ЦЕНА] * [КОЛ_ОТГР]$ , которое рассчитывает значение для обновления.

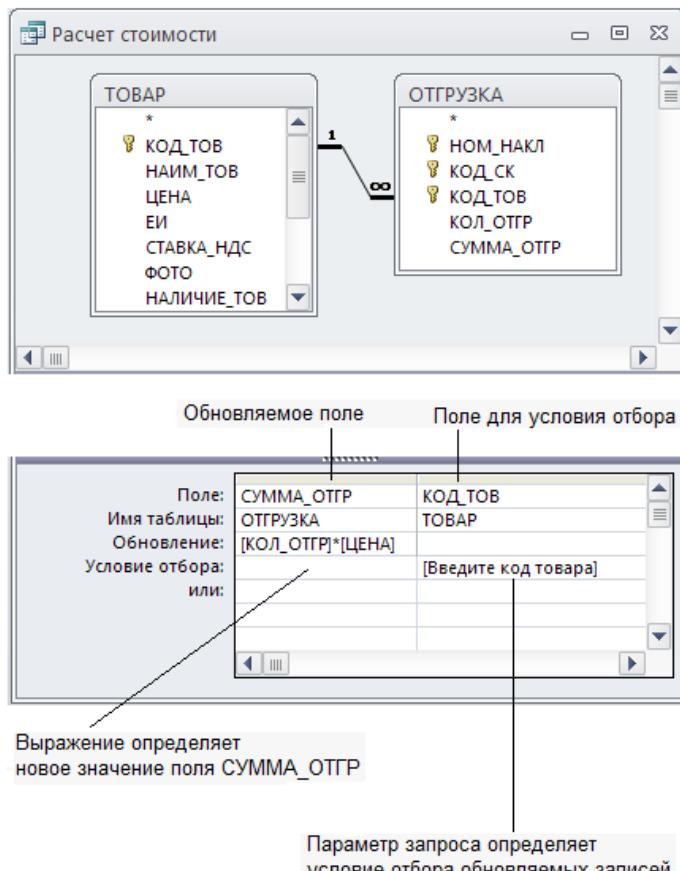


Рис. 4.43. Пример запроса на обновление

4. Просмотрите содержимое обновляемого поля СУММА\_ОТГР перед выполнением запроса, нажав кнопку **Режим** (View) на ленте конструктора запросов в группе **Результаты** (Results).
  5. Для обновления содержимого поля СУММА\_ОТГР выполните запрос, нажав кнопку **Выполнить** (Run) на вкладке ленты **Конструктор** (Design). Открывается диалоговое окно с сообщением о числе обновляемых записей и вопросом о продолжении операции обновления. Подтвердите обновление записей.
  6. Просмотрите содержимое обновляемого поля СУММА\_ОТГР после выполнения запроса. Для этого переключитесь после выполнения запроса в режим таблицы, воспользовавшись кнопкой **Режим таблицы** (Datasheet View) в строке состояния или нажмите кнопку **Режим** (View) на вкладке ленты.
- Таким образом, рассмотренный запрос позволяет автоматизировать расчет стоимости товара, указанного в каждой строке спецификации накладной — записи таблицы ОТГРУЗКА.
1. Если обновлять нужно только некоторые строки таблицы, задайте условия отбора обновляемых записей. Для этого дополните бланк запроса полем, по которому требуется произвести отбор записей. Перетащите поле КОД\_ТОВ в бланк запроса и введите в строку **Условия отбора** (Criteria) параметр [Введите код товара] (см. рис. 4.43).
  2. Выполните запрос. Обновление будет выполнено только для записей с введенным кодом товара.
  3. Сохраните запрос под именем **Расчет стоимости**.

### **ОБРАТИТЕ ВНИМАНИЕ**

При использовании в запросе на обновление таблиц, находящихся в отношении 1 : M, обновлять можно только содержимое столбцов таблицы со стороны "многие".

4. Перейдите в режим SQL. Эквивалентная запросу на обновление инструкция UPDATE будет записана следующим образом:

```
UPDATE ТОВАР INNER JOIN ОТГРУЗКА ON ТОВАР.КОД_ТОВ = ОТГРУЗКА.КОД_ТОВ
SET ОТГРУЗКА.СУММА_ОТГР = [ЦЕНА]*[КОЛ_ОТГР]
WHERE (((ТОВАР.КОД_ТОВ)=[Введите код товара]));
```

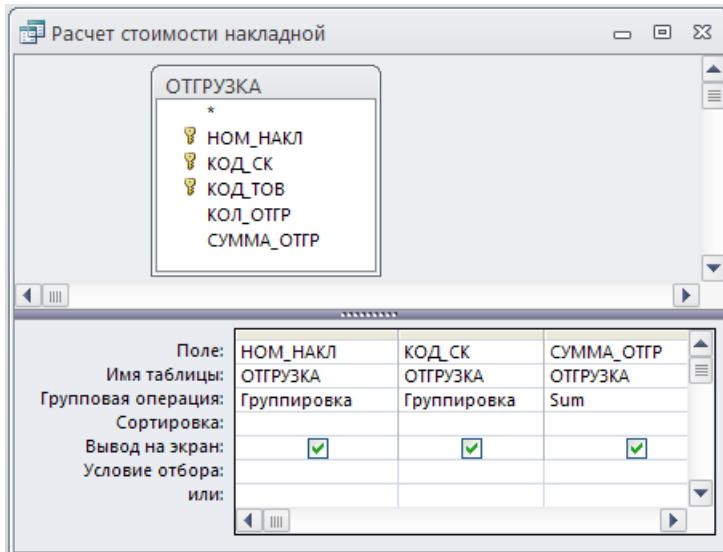
Имена таблиц, используемых в запросе, и способ их объединения задаются непосредственно за именем инструкции UPDATE. Инструкция UPDATE обновляет указанное в предложении SET поле ОТГРУЗКА.СУММА\_ОТГР, присваивая значение, заданное выражением [ЦЕНА]\*[КОЛ\_ОТГР]. Обновление происходит во всех записях, которые удовлетворяют условию отбора, заданному в предложении WHERE.

## **Обновление полей значениями, рассчитанными в запросе с группировкой**

**Задача.** Пусть необходимо рассчитать общую стоимость товара, отгруженного по накладной, и сохранить ее в соответствующем поле таблицы НАКЛАДНАЯ.

Рассмотренный ранее запрос **Расчет стоимости** (см. рис. 4.43) позволяет автоматизировать расчет стоимости товара, указанного в каждой строке спецификации накладной — записи таблицы ОТГРУЗКА. Для этого достаточно убрать в нем параметр условия отбора.

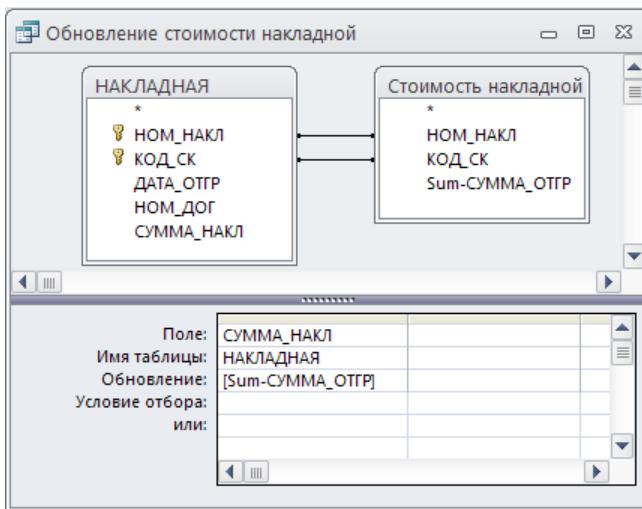
1. Для подсчета общей стоимости товара, отгруженного по каждой накладной, создайте запрос на основе таблицы ОТГРУЗКА.
2. Включите в бланк запроса поля, определяющие составной ключ накладной: номер накладной (НОМ\_НАКЛ) и код склада (КОД\_СК) и поле СУММА\_ОТГР.
3. Выполните группировку по двум полям, определяющим составной ключ накладной. Это приведет к включению в группу всех записей об отгрузках товаров по одной накладной. Для поля СУММА\_ОТГР выберите статистическую функцию `Sum`. Для группы будет подсчитана стоимость всех товаров, отгруженных по накладной.
4. Теперь значения стоимости накладных, рассчитанные с использованием статистической функции `Sum`, нужно сохранить в соответствующем поле таблицы НАКЛАДНАЯ. Однако запрос на обновление непосредственно на запросе с группировкой построить нельзя. Поэтому преобразуйте запрос на выборку в запрос на создание таблицы. Создаваемой в запросе таблице дайте имя **Стоимость накладной**. Далее эту таблицу используйте для обновления поля СУММА\_НАК в накладной.



**Рис. 4.44.** Расчет стоимости всего товара, отгруженного по накладной, и сохранение в новой таблице

5. Если в таблице ОТГРУЗКА не хранится значение стоимости товара, рассчитайте общую стоимость товаров одной накладной в вычисляемом поле по формуле

- [ЦЕНА] \* [КОЛ\_ОТГР] и для этого поля используйте статистическую функцию Sum. Запрос в этом случае создавайте на двух таблицах ОТГРУЗКА и ТОВАР.
6. Сохраните запрос под именем **Расчет стоимости накладной** (рис. 4.44).
  7. Создайте запрос на обновление на основе обновляемой таблицы базы данных НАКЛАДНАЯ и новой таблицы **Стоимость накладной**, содержащей данные для обновления.
  8. В схеме данных запроса установите связь таблиц по полям, однозначно идентифицирующим накладную — НОМ\_НАКЛ и КОД\_СК.
  9. В бланк запроса включите единственное обновляемое в таблице НАКЛАДНАЯ поле СУММА\_НАКЛ, в строке **Обновление** (Update To) введите имя поля Sum-СУММА\_ОТГР из таблицы **Стоимость накладной**, из которого будет выбираться значение для обновления (рис. 4.45).
  10. Сохраните запрос под именем **Обновление стоимости накладной**.



**Рис. 4.45.** Обновление поля с общей стоимостью товара по накладной

11. Для решения задачи вычисления общей стоимости накладной последовательно выполните сначала запрос на создание таблицы **Расчет стоимости накладной**, а затем запрос на **Обновление стоимости накладной**. Если в таблице ОТГРУЗКА не была рассчитана стоимость каждого товара, отгружаемого покупателю, предварительно должен быть выполнен запрос **Расчет стоимости** (см. рис. 4.43) без условий отбора.

## Конструирование запроса на добавление

С помощью запроса **Добавление** (Append) производится добавление записей в таблицу базы данных. Добавляемые записи формируются в создаваемом конструк-

тором запросе на выборку из одной или нескольких таблиц базы данных. При использовании нескольких таблиц их записи объединяются в соответствии с указанным способом объединения.

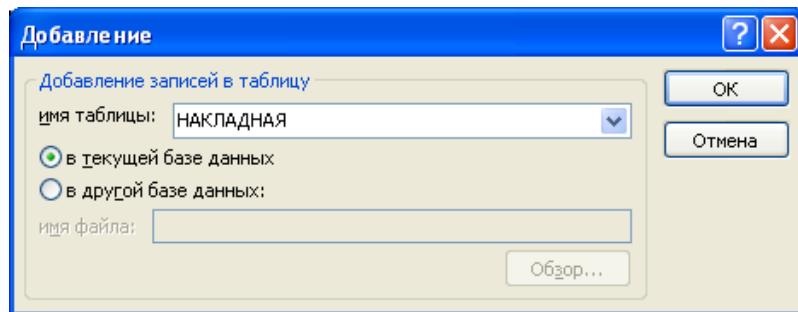
Структура записи таблицы, из которой добавляются записи, может не полностью совпадать со структурой записи дополняемой таблицы. В добавляемых записях может быть меньше полей, если на поля в таблице, куда добавляются записи, не наложено требование по обязательности их заполнения. Допускается несоответствие типов полей, если возможно преобразование типа данных одного поля в тип данных другого поля. Добавляемые записи обязательно должны включать значения ключевых полей. В бланк запроса могут быть включены поля для задания условий отбора добавляемых записей. В качестве значений полей добавляемых записей может быть использовано вычисляемое выражение.

## Добавление данных в связанные таблицы

**Задача.** Пусть на складах фирмы имеются свои базы данных, в которых ведется учет накладных. Данные накладных так же, как и данные в централизованной базе *Поставка товаров*, сохраняются в двух таблицах. Допустим, они имеют имена **Накладные склада** и **Отгрузка склада** и такие же ключевые поля, как и аналогичные таблицы централизованной базы. В конце каждого дня в централизованной базе данных выполняется команда импортирования этих таблиц с данными об отгруженных со склада товарах. Далее с помощью запросов на добавление накладные, оформленные на складе, включаются в централизованную базу данных. Первым должен выполняться запрос на добавление записей в главную таблицу **НАКЛАДНАЯ**. Только после этого могут быть добавлены связанные записи в подчиненную таблицу **ОТГРУЗКА**. Это обусловлено тем, что в схеме данных базы для связи таблиц **НАКЛАДНАЯ** и **ОТГРУЗКА** установлен параметр **Обеспечение целостности данных** (Enforce Referential Integrity), делающий возможным добавление подчиненных записей только в том случае, когда в главной таблице уже имеется связанные записи.

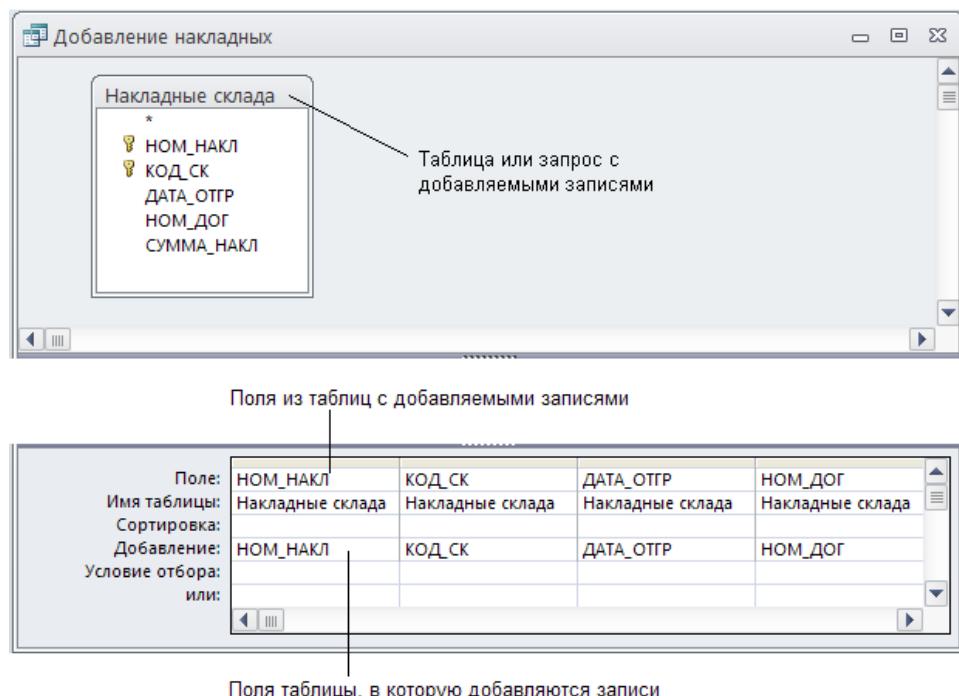
1. Создайте запрос на основе таблицы **Накладные склада**, выполнив команду **Конструктор запросов** (Query Design) на вкладке ленты — **Создание** (Create) в группе **Запросы** (Queries). По умолчанию будет создан запрос на выборку, что подтверждается выделенным на вкладке **Работа с запросами | Конструктор** (Query Tools | Design) типом созданного запроса — **Выборка** (Select).
2. Превратите его в запрос на добавление, выбрав тип запроса **Добавление** (Append). При этом открывается диалоговое окно **Добавление** (Append) (рис. 4.46).
3. В окне **Добавление** (Append) в поле **Имя таблицы** (Table Name) введите или выберите из списка имя таблицы **НАКЛАДНАЯ**, в которую надо добавить записи.
4. Так как таблица — получатель добавляемых записей находится в открытой базе данных, отметьте переключатель **В текущей базе данных** (Current Database). Для таблицы, находящейся в другой базе данных, надо отметить переключатель **В другой базе данных** (Another Database) и ввести имя файла базы дан-

ных. При необходимости надо указать путь к другой базе данных. Помимо базы данных Access, можно указать базу данных Microsoft FoxPro, Paradox, dBASE или ввести строку подключения для базы данных SQL Server.



**Рис. 4.46.** Окно для ввода имени таблицы, в которую добавляются записи

- После преобразования запроса в его бланке появляется строка **Добавление** (Append To) (рис. 4.47).



**Рис. 4.47.** Запрос на добавление записей

- Для формирования добавляемых в таблицу НАКЛАДНАЯ записей обязательно включите в бланк запроса из таблицы Накладные склада поля НОМ\_НАК и

- КОД\_СК, составляющие ключ, и поле связи НОМ\_ДОГ. Включите также поля ДАТА\_ОТГР и СУММА\_НАКЛ.
7. Поскольку в данном примере имена полей обеих таблиц совпадают, Access автоматически сформирует в строке **Добавление** (Append To) имена полей таблицы НАКЛАДНАЯ, в которые будут добавляться значения из соответствующих полей таблицы Накладные склада. Если эти имена не совпадают, откройте в каждой ячейке список полей и выберите нужное имя.
  8. Для предварительного просмотра записей, которые планируется добавить в таблицу, нажмите кнопку **Режим** (View) на вкладке ленты **Конструктор** (Design) или **Режим таблицы** (Datasheet View) в строке состояния. Возврат в режим конструктора запросов производится аналогично.
  9. Для добавления записей выполните запрос, нажав кнопку **Выполнить** (Run) на вкладке ленты. При этом откроется диалоговое окно с сообщением о числе добавляемых записей и вопросом о продолжении операции добавления.

### **ЗАМЕЧАНИЕ**

Если таблица, в которую добавляются записи, содержит ключевое поле, добавляемые записи должны содержать такое же поле. Те записи, добавление которых приведет к появлению совпадающих или пустых значений в ключевом поле, не будут добавлены. Записи не добавляются и в случае, если невозможно преобразование типа данных в добавляемых полях или не выполняются условия на значения.

10. Перейдите в режим SQL. Эквивалентная запросу на добавление инструкция **INSERT INTO** имеет вид:

```
INSERT INTO НАКЛАДНАЯ (НОМ_НАКЛ, КОД_СК, ДАТА_ОТГР, НОМ_ДОГ, СУММА_НАКЛ)
    SELECT [Накладные склада].НОМ_НАКЛ, [Накладные склада].КОД_СК,
           [Накладные склада].ДАТА, [Накладные склада].НОМ_ДОГ,
           [Накладные склада].СУММА_НАКЛ
      FROM [Накладные склада];
```

Инструкция **INSERT INTO** определяет поля таблицы НАКЛАДНАЯ, которые должны заполниться значениями из полей добавляемых записей. Поскольку в запросе в качестве источника добавляемых записей указана таблица Накладные склада, конструктор создал на ее основе запрос на выборку, который определяет поля добавляемых записей. Этот запрос представлен инструкцией **SELECT**. Список имен полей и порядок их перечисления для обеих таблиц должен совпадать.

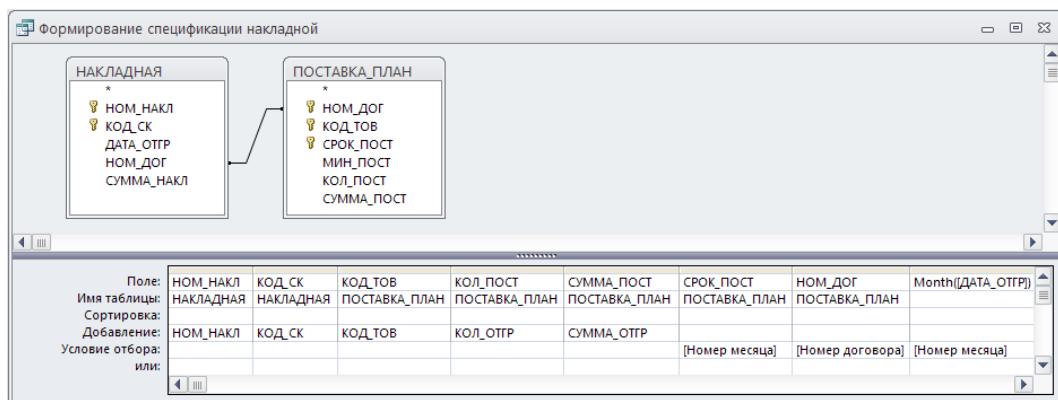
**Задание.** После дополнения таблицы НАКЛАДНАЯ записями из таблицы Накладные склада можно добавлять записи в подчиненную таблицу ОТГРУЗКА из таблицы Отгрузка склада. Создайте запрос на добавление записей в таблицу ОТГРУЗКА из таблицы Отгрузка склада.

## **Добавление данных из нескольких таблиц**

**Задача.** Необходимо создать новую накладную на отгрузку товара покупателю. Пусть товары по накладным отгружаются один раз в месяц в полном соответствии с планом поставок. Это означает, что по каждому договору в месяц выписывается

только одна накладная, а спецификация накладной полностью совпадать со спецификацией договора на заданный месяц. Прежде чем заполнять спецификацию накладной создайте в таблице НАКЛАДНАЯ новую запись, заполнив поля номер накладной, номер склада, номер договора и дата отгрузки. Только теперь можно приступить к добавлению связанных записей в таблицу ОТГРУЗКА, сформировав эти записи на основании записей, выбранных из таблицы ПОСТАВКА\_ПЛАН.

1. Создайте запрос на добавление записей в таблицу ОТГРУЗКА на основе таблиц НАКЛАДНАЯ и ПОСТАВКА\_ПЛАН. Установите между ними связь по полю НОМ\_ДОГ. Таблица НАКЛАДНАЯ включена в запрос для того, чтобы сформировать в добавляемых записях ключевое поле, включающее НОМ\_НАКЛ, КОД\_СК.
2. В запрос включите поля НОМ\_НАКЛ, КОД\_СК, КОД\_ТОВ, КОЛ\_ПОСТ, СУММА\_ПОСТ, которые составят запись, добавляемую в таблицу ОТГРУЗКА, а также СРОК\_ПОСТ, НОМ\_ДОГ и вычисляемое поле Month(ДАТА\_ОТГР) для задания условий отбора добавляемых записей (рис. 4.48).
3. Задайте параметры запроса, которые позволяют ввести номер месяца и номер договора, по которым из таблицы ПОСТАВКА\_ПЛАН будут выбраны все записи, необходимые для добавления в таблицу ОТГРУЗКА. В каждом месяце по договору выписывается только одна накладная, поэтому по номеру договора и номеру месяца из таблицы НАКЛАДНАЯ будет выбрана только одна запись — та, которая была добавлена в таблицу при вводе в базу данных новой накладной. При объединении записей таблиц запроса ключевые поля этой записи будут добавлены ко всем записям, выбранным из таблицы ПОСТАВКА\_ПЛАН. В результате записи обретут составной ключ — НОМ\_НАКЛ, КОД\_СК, КОД\_ТОВ — необходимый для их добавления в таблицу ОТГРУЗКА.
4. Для предварительного просмотра записей, которые будут добавлены в таблицу, нажмите на вкладке ленты кнопку Режим (View). Для добавления записей нажмите кнопку Выполнить (Run).



**Рис. 4.48.** Добавление записей в таблицу ОТГРУЗКА, соответствующих заданному договору и месяцу

5. Перейдите в режим SQL. Эквивалентная запросу на добавление инструкция `INSERT INTO` будет записана следующим образом:

```
INSERT INTO ОТГРУЗКА (НОМ_НАКЛ, КОД_СК, КОД_ТОВ, КОЛ_ОТГР, СУММА_ОТГР)
SELECT НАКЛАДНАЯ.НОМ_НАКЛ, НАКЛАДНАЯ.КОД_СК,
       ПОСТАВКА_ПЛАН.КОД_ТОВ, ПОСТАВКА_ПЛАН.КОЛ_ПОСТ,
       ПОСТАВКА_ПЛАН.СУММА_ПОСТ
  FROM НАКЛАДНАЯ INNER JOIN ПОСТАВКА_ПЛАН ON
        НАКЛАДНАЯ.НОМ_ДОГ = ПОСТАВКА_ПЛАН.НОМ_ДОГ
 WHERE ((ПОСТАВКА_ПЛАН.СРОК_ПОСТ)=[Номер месяца]) AND
       ((ПОСТАВКА_ПЛАН.НОМ_ДОГ)=[Номер договора]) AND
       ((Month([ДАТА_ОТГР]))=[Номер месяца]));
```

Инструкция `INSERT INTO` определяет поля записей, добавляемых в таблицу `ОТГРУЗКА`. Структура и количество добавляемых записей определяется числом записей, возвращаемых инструкцией выбора `SELECT`. Список имен полей и порядок их перечисления для таблицы `ОТГРУЗКА`, куда добавляются записи, и для таблицы, определяемой инструкцией `SELECT`, должны совпадать.

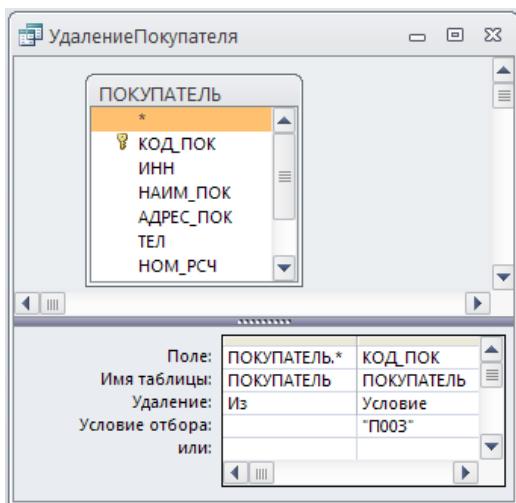
## Конструирование запроса на удаление

Запрос на удаление (Delete Query) позволяет удалить записи из одной таблицы или из нескольких взаимосвязанных таблиц, для связи с которыми установлен параметр обеспечения целостности данных **каскадное удаление связанных записей** (Cascade Delete Related Records). В схему данных запроса включается таблица, из которой должны удаляться записи, и связанные с ней таблицы, для полей которых требуется задать условия отбора удаляемых записей.

**Задача.** Пусть необходимо из справочника удалить запись о покупателе с кодом П003. В схеме данных базы *Поставка товаров* для всех связей установлен параметр обеспечения целостности **каскадное удаление связанных записей** (Cascade Delete Related Records), поэтому удаление записи из таблицы `ПОКУПАТЕЛЬ` повлечет за собой удаление из базы всех подчиненных ему записей на всех уровнях иерархии. Автоматически будут удалены все договоры этого покупателя, а также все накладные, выписанные по этим договорам, т. е. будут удалены связанные записи из таблиц `ДОГОВОР`, `ПОСТАВКА_ПЛАН`, `НАКЛАДНАЯ` и `ОТГРУЗКА`.

1. Создайте запрос на выборку. Добавьте в схему данных запроса таблицу `ПОКУПАТЕЛЬ`, из которой удаляются записи.
2. Преобразуйте запрос на выборку в запрос на удаление, выбрав на вкладке ленты **Конструктор** (Design) тип запроса **Удаление** (Delete) или выполнив команду контекстного меню запроса **Тип запроса | Удаление** (Query Type | Delete).
3. После преобразования запроса в запрос на удаление в его бланке появляется строка **Удаление** (Delete) (рис. 4.49).

4. Далее приступайте к формированию бланка запроса. Переместите символ звездочки (\*) из списка полей таблицы ПОКУПАТЕЛЬ в бланк запроса. В строке **Удаление** (Delete) в столбце этого поля появляется значение **Из** (From).
5. Для того чтобы задать условия отбора удаляемых записей, переместите с помощью мыши в бланк запроса поле КОД\_ПОК из таблицы ПОКУПАТЕЛЬ. В строке **Удаление** (Delete) под именем этого поля появится значение **Условие** (Where). Запишите в строку **Условие отбора** (Criteria) для этого поля значение P003.



**Рис. 4.49.** Запрос на удаление записи в главной таблице

6. Следует иметь в виду, что удаленные записи нельзя восстановить. Поэтому перед удалением записей выполните предварительный просмотр удаляемых записей, нажав кнопку **Режим** (View) на вкладке ленты.
7. Для удаления записей на вкладке ленты нажмите кнопку **Выполнить** (Run). При выполнении запроса будет сообщено об удалении одной записи. Откройте таблицу ПОКУПАТЕЛЬ и подчиненные ей таблицы на всех уровнях и убедитесь, что в них также удалены записи, связанные с этим покупателем.

Результаты работы запроса на удаление зависят от отношений между таблицами и установленных в схеме базы данных параметров обеспечения целостности для их связей.

Если параметры обеспечения целостности для связей таблицы не установлены вообще, то записи удаляются только в указанной в бланке запроса таблице и вне зависимости от ее логических связей.

Если в запросе объединяются две или более таблиц, находящихся в отношении 1 : M, то удалить можно только записи одной из таблиц на стороне "многие".

Если между таблицами, находящимися в отношении 1 : M, установлена связь с поддержанием целостности, но не задан параметр **каскадное удаление связанных записей** (Cascade Delete Related Records), сначала необходимо удалить записи под-

чиненной таблицы и только после этого можно выполнить удаление записей в главной таблице. То есть удаление записи главной таблицы возможно только, если в подчиненной таблице нет связанных записей. Если параметр **каскадное удаление связанных записей** (Cascade Delete Related Records) задан, то для удаления записей главной таблицы и связанных с ними подчиненных записей достаточно указать в запросе удаление записей главной таблицы.

Откройте запрос на удаление, представленный на рис. 4.49, в режиме конструктора и перейдите в режим SQL. Эквивалентная этому запросу инструкция DELETE будет записана следующим образом:

```
DELETE ПОКУПАТЕЛЬ.* , ПОКУПАТЕЛЬ.КОД_ПОК  
FROM ПОКУПАТЕЛЬ  
WHERE ((ПОКУПАТЕЛЬ.КОД_ПОК)="П003");
```

Инструкция DELETE удаляет из таблицы записи, удовлетворяющие условию отбора, заданному в предложении WHERE. Если в предложении FROM объединяется две или более таблиц, то удалить можно только записи одной из таблиц. Для того чтобы указать, из какой именно таблицы должны быть удалены записи, в список полей включается конструкция **имя\_таблицы.\*** (например, ПОСТАВКА\_ПЛАН.\*). Для однотабличного запроса эта конструкция может быть опущена.

## Контрольные вопросы

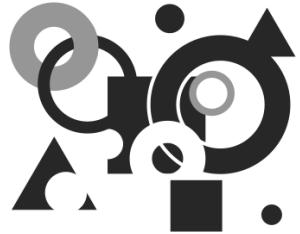
1. Можно ли запросы на изменение создать с помощью мастера?
2. С создания запроса какого типа начинается конструирование запроса на изменение?
3. Как просмотреть в запросе на обновление записи, которые будут изменены в таблице?
4. Определен ли в структуре таблицы, созданной запросом, ключ?
5. Какой командой осуществляется преобразование запроса на выборку в запрос на обновление?
6. В какой строке запроса на обновление указывается новое значение обновляемого поля?
7. Можно ли для подсчета нового значения обновляемого поля использовать выражение?
8. Какая инструкция языка SQL соответствует запросу на создание таблицы?
9. Можно ли в новую таблицу, создаваемую запросом, включать поля нескольких таблиц?
10. Какая инструкция языка SQL соответствует запросу на обновление?
11. Можно ли просмотреть содержимое обновляемых полей до и после обновления, не выходя из режима конструктора запросов?
12. В какой строке запроса на добавление указывается, откуда должны выбираться значения полей добавляемых записей?

13. Где указывается имя таблицы, в которую добавляются записи?
14. В какой строке запроса на добавление указывается, в какие поля должны попадать поля добавляемой записи?
15. Какая инструкция языка SQL соответствует запросу на добавление?
16. Вслед за каким словом в инструкции `INSERT` записывается имя таблицы, в которую добавляются записи?
17. С помощью какого предложения формируются добавляемые записи в инструкции `INSERT`?
18. Могут ли добавляемые записи формироваться на основе нескольких таблиц?
19. Можно ли одним запросом удалить записи из нескольких таблиц?
20. Если в запросе на удаление использованы главная таблица и две подчиненные, находящиеся с главной в отношении  $1 : M$ , из какой таблицы могут удаляться записи?
21. Каким образом в запросе на удаление указывается таблица, из которой удаляются записи?
22. Какая инструкция языка SQL соответствует запросу на удаление?
23. Если в предложении `FROM` инструкции `DELETE` указана одна таблица, нужно ли использовать в списке полей конструкцию `имя_таблицы.*`?
24. Какие поля включаются в список полей запроса на удаление?

## Ответы

1. Нет.
2. Запроса на выборку.
3. Нажав в режиме конструктора запросов кнопку **Режим (View)**.
4. Нет.
5. Командой **Обновление (Update)** из списка контекстного меню запроса **Тип запроса (Query Type)** или на вкладке ленты конструктора запросов.
6. **Обновление (Update To)**.
7. Да.
8. `SELECT ... INTO`.
9. Да.
10. `UPDATE`.
11. Да.
12. В строке **Поле (Field)** указываются имена полей, а в строке **Имя таблицы (Table)** — имена таблиц, из которых выбираются данные для добавления.
13. В диалоговом окне **Добавление (Append)**, открываемом при преобразовании запроса на выборку в запрос на добавление.
14. **Добавление (Append To)**.
15. `INSERT`.

16. `INTO`.
17. С помощью инструкции выбора `SELECT`.
18. Да, как в любом запросе на выборку.
19. Нет. Исключением является автоматическое удаление наряду с записью главной таблицы связанных записей подчиненных таблиц, для которых установлен флажок **каскадное удаление связанных записей** (*Cascade Delete Related Records*).
20. Из любой подчиненной таблицы.
21. Размещением в строке **Поле** (Field) бланка запроса конструкции `имя_таблицы.*` и в строке **Удаление** (Delete) значения **Из** (From).
22. `DELETE`.
23. Нет.
24. Все поля таблицы, из которой удаляются записи, и, если необходимо, поля, для которых формулируются условия отбора.



## ГЛАВА 5

### Формы

Формы являются основой разработки диалоговых приложений пользователя для работы с базой данных. Формы, адекватные формам первичных документов, позволяют выполнять загрузку справочных, плановых и оперативно-учетных данных, в любой момент просматривать и редактировать содержимое ранее введенных в базу данных документов, оформлять новые документы.

Формы обеспечивают удобную работу с данными одной или нескольких взаимосвязанных таблиц, которые выводятся на экран с использованием ее макета, разработанного пользователем. Работая с формой, пользователь может добавлять, удалять и изменять записи таблиц, получать расчетные данные. В процессе работы может осуществляться контроль вводимых данных, могут проверяться ограничения на доступ к данным, выводиться необходимые дополнительные сведения.

Форма состоит из элементов управления, которые отображают поля таблиц, и графические элементы, не связанные с полями таблиц. Графические элементы управления предназначены, прежде всего, для разработки макета формы: полей таблиц и запросов, надписей, внедряемых объектов (рисунков, диаграмм), вычисляемых полей, кнопок, выполняющих печать, открывающих другие объекты или задачи.

Как форма в целом, так и каждый из ее элементов обладает множеством свойств. Посредством их изменения можно настроить внешний вид, размер, местоположение элементов в форме, определить источник данных формы, режим ввода/вывода, привязать к элементу выражение, макрос или программу. Набор свойств доступен в соответствующем окне, где они разбиты на категории, каждая из которых представлена на своей вкладке. Основными вкладками в окне свойств являются:

- ❖ **Макет (Format)** — представляет свойства, ориентированные на определение внешнего вида формы или ее элементов;
- ❖ **Данные (Data)** — представляет свойства для определения источника данных формы или ее элементов, режима использования формы (только ввод, разрешение на изменение, добавление, удаление и т. п.);
- ❖ **События (Event)** — событиями называют определенные действия, возникающие при работе с конкретным объектом или элементом: нажатие кнопки мыши, изменение данных, до обновления, после обновления, открытие или закрытие формы и т. д. Они могут быть инициированы пользователем или системой.

С событием может связываться макрос или процедура обработки события на языке VBA, выполняющая некоторые действия или рассчитывающая значения. Например, в процедуре можно организовать открытие связанной формы, обновление данных таблицы расчетными значениями, печать формы, вывод отчета. Запрограммировав в процедурах вызов различных объектов базы данных, можно автоматизировать выполнение задач приложения.

Для быстрого создания формы предназначены мастера Access. Однако точное формирование макета формы, отвечающего заданным требованиям, дополнение процедурами обработки событий, возникающих в форме, обеспечивается средствами конструирования. Конструктор форм можно использовать как для создания новой формы, так и для редактирования формы, созданной мастером. Кроме того, в Access 2007/2010 включены новые функциональные возможности, позволяющие выполнить доработку формы в режиме макета.

В процессе создания формы выбираются поля таблицы, которые должны быть представлены в форме, осуществляется их размещение в форме, создаются вычисляемые поля, графические элементы — кнопки, выключатели, элементы оформления, поясняющий текст и рисунки. Для настройки различных элементов формы используется типовой набор их свойств.

Формы в Access могут быть представлены в трех режимах.

- ❖ **Режим формы** (Form View) предназначен для ввода, просмотра и корректировки данных таблиц, на которых основана форма.
- ❖ **Режим макета** (Layout View) обеспечивает просмотр данных почти в таком виде, в каком они отображаются в режиме формы, и в то же время позволяет изменять форму. В этом режиме элементы формы становятся выделяемыми, их можно перетаскивать в другие места, редактировать содержимое надписей полей, изменять формат, размер и т. п. Режим макета позволяет удобно настраивать внешний вид формы и может использоваться для внесения большинства структурных изменений. В Access 2010 появилась возможность в режиме макета выполнять действия, ранее доступные только в режиме конструктора. В режиме макета стала доступной лента **Конструктор** (Design). Если некоторую задачу невозможно выполнить в режиме макета, следует переключиться в режим конструктора. В ряде случаев в Access отображается сообщение о том, что для внесения изменений надо переключиться в режим конструктора.
- ❖ **Конструктор** (Design View) предназначен для разработки формы с помощью полного набора инструментов, обеспечивающего более детальную проработку структуры формы, использование всех элементов управления. В этом режиме форму можно разработать с нуля или доработать ее после создания мастером. Просмотр данных при внесении изменений в этом режиме не предусматривается.

## Однотабличные формы

Однотабличная форма предназначена для загрузки, просмотра и корректировки данных одной таблицы. Источником данных такой формы служит единственная

таблица. Она может быть легко создана одним щелчком мыши с помощью команд автоматического создания формы: **Форма** (Form), **Разделенная форма** (Split Form) или **Несколько элементов** (Multiple Items), размещенных на вкладке ленты **Создание** (Create) в группе **Формы** (Forms) (рис. 5.1). Для последующей настройки формы в соответствии с требованиями пользователя ее можно доработать в режиме макета или конструктора. Можно удалить из формы ненужные поля, изменить расположение элементов управления и подобрать их размеры, добавить новые элементы управления, произвести вычисления, задать свойства формы и ее элементов управления.

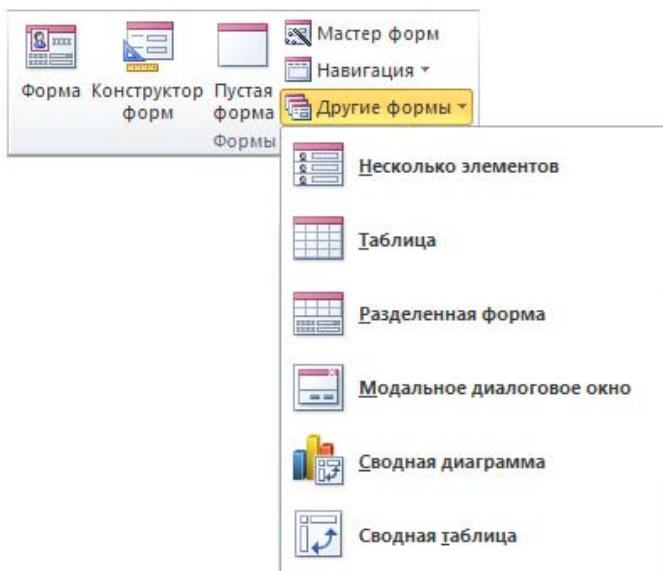


Рис. 5.1. Команды группы **Формы** на вкладке ленты **Создание**

Формы, созданные с помощью перечисленных команд, отличаются, прежде всего, способом отображения записей базового источника данных. В форме, созданной с помощью команды **Форма** (Form), одновременно отображается только одна запись, поэтому ее целесообразно использовать, например, в случае, когда таблица включает поля с данными, требующими большого окна для отображения (поле МЕМО, диаграмма, фотография). Поля отображаемой записи располагаются внутри одного раздела формы, в столбик (макет в столбик) с надписью слева от каждого поля.

В форме, созданной с помощью команды **Несколько элементов** (Multiple Items), отображается сразу несколько записей, и все поля записи размещаются в одной строке, что удобно для ввода данных из простых справочников, имеющих только табличную часть. Этот макет формы аналогичен отображению записей таблицы в режиме таблицы, однако при этом предоставляются многочисленные возможности по настройке отображения и дополнения новыми элементами.

В форме, созданной с помощью команды **Разделенная форма** (Split Form), одновременно отображаются данные в двух представлениях — в одном ее разделе записи отображаются в виде таблицы, в другом выводится единственная выделенная в таблице запись, предназначенная для удобной работы с ее данными.

Общим для этих команд является то, что они автоматически создают для выбранной таблицы форму, не вступая в диалог с пользователем, и сразу выводят на экран форму в режиме макета.

Создать однотабличную форму можно с помощью команды **Мастер форм** (Form Wizard), размещенной на вкладке ленты **Создание** (Create) в группе **Формы** (Forms). В диалоговых окнах мастера пользователь выбирает поля, которые надо включать в форму, способ отображения записей, стиль оформления.

## Создание однотабличной формы

Создайте однотабличную форму СПРАВОЧНИК ТОВАРОВ для ввода, просмотра и корректировки данных таблицы ТОВАР в базе данных *Поставка товаров*. Поскольку в таблице есть поле с фотографией, требующее большой области для отображения, одновременно выводите в форме поля только одной записи таблицы.

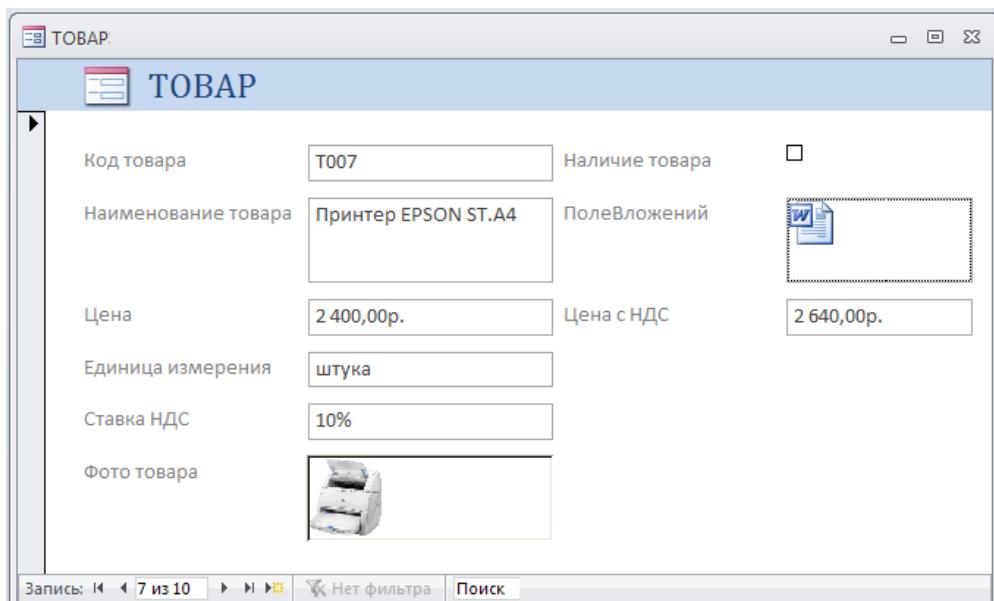


Рис. 5.2. Форма, полученная по команде **Форма**, для работы с таблицей ТОВАР

Чтобы источником записей формы стала таблица ТОВАР, выберите ее в области навигации и выполните команду **Форма** (Form) на вкладке ленты **Создание** (Create). Эта команда обеспечит автоматическое создание формы на основе только одной выбранной таблицы.

## ЗАМЕЧАНИЕ

Если таблица имеет подчиненную таблицу, с которой она находится в отношении  $1 : M$ , то на основе подчиненной таблицы автоматически создается внедренный элемент, отображающий записи подчиненной таблицы, связанные с текущей записью главной. Если необходимо построить однотабличную форму, следует удалить связи. Таблице ТОВАР в схеме данных подчинено две таблицы, поэтому сразу создается однотабличная форма.

Завершается работа команды отображением формы в режиме макета (рис. 5.2). При этом в форме отображаются поля одной записи таблицы, расположенные в макете в столбик. Заголовок формы соответствует имени таблицы источника ТОВАР. Скрытые в таблице поля, например, командой **Скрыть поля** (Hide Fields) из контекстного меню выделенного столбца, не включаются в форму.

## Редактирование формы в режиме макета

В режиме макета можно просматривать данные практически так же, как в режиме формы, и в то же время вносить изменения в форму. Это средство функционирует по принципу WYSIWYG (что видим, то и получаем), и позволяет вносить изменения в форму и тут же видеть результат внесенных изменений, что очень удобно для уточнения местоположения, размера, шрифта элементов управления, изменения текста надписей и выполнения других задач, связанных с внешним видом и удобством формы. При отображении формы в режиме макета появляются вкладки ленты инструментов **Работа с макетами форм | Формат** (Form Layout Tools | Format), **Упорядочить** (Arrange) и **Конструктор** (Design) (рис. 5.3—5.6), которые сохраняются на экране, пока активно окно формы и не выполнено переключение в другой режим.

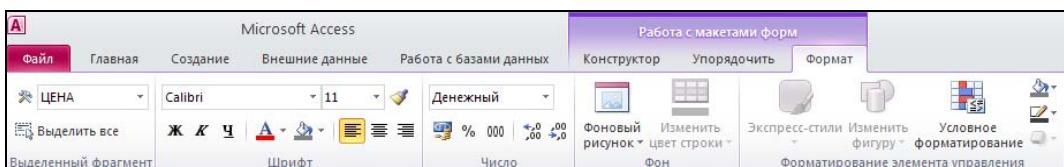


Рис. 5.3. Вкладка ленты Работа с макетами форм | Формат

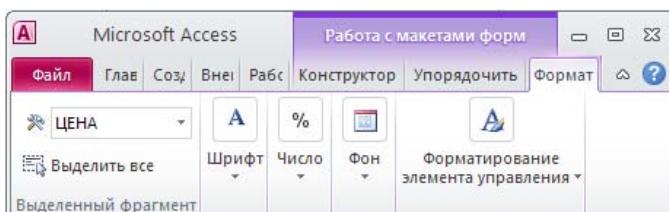
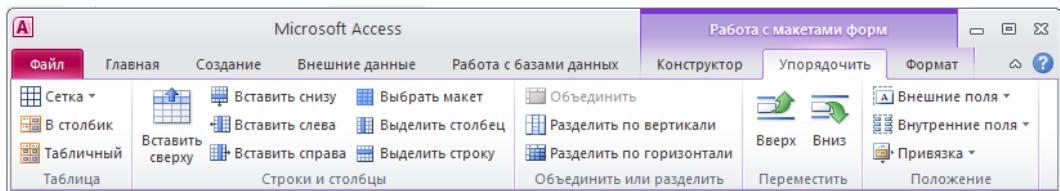
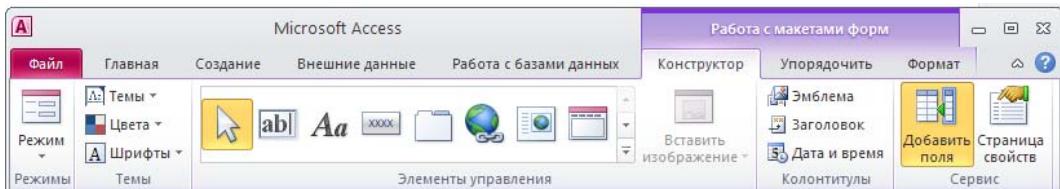


Рис. 5.4. Свернутые группы вкладки ленты Формат

При сворачивании окна Access с уменьшением его размеров сворачиваются и группы ленты. При этом команды группы можно открыть с помощью значка списка (рис. 5.4).



**Рис. 5.5.** Вкладка ленты  
**Работа с макетами форм | Упорядочить**



**Рис. 5.6.** Вкладка ленты  
**Работа с макетами форм | Конструктор**

- Для открытия формы ТОВАР в режиме макета выберите ее в области навигации и в контекстном меню щелкните на строке **Режим макета** (Layout View). Если форма открыта, и необходимо переключиться в режим макета, щелкните на соответствующем значке строки состояния или выполните команду **Режим макета** (Layout View) из списка кнопки **Режим** (View) на вкладке ленты **Конструктор** (Design) или **Главная** (Home).
- Измените заголовок формы на СПРАВОЧНИК ТОВАРОВ. Для перехода к редактированию заголовка щелкните на значке **Заголовок** (Title) в группе **Колонтитулы** (Header/Footer) на вкладке ленты **Конструктор** (Design) или просто сделайте двойной щелчок на заголовке.
- Замените эмблему, которая автоматически вставилась в форму при ее создании. Для этого щелкните на значке **Эмблема** (Logo) и выберите свой рисунок.
- Вставьте в форму дату. Для этого щелкните на кнопке **Дата и время** (Date and Time) на вкладке ленты **Конструктор** (Design) в группе **Колонтитулы** (Header/Footer). В открывшемся окне **Дата и время** выберите формат даты, уберите флајжок **Формат времени** (Include Time) и нажмите кнопку **OK**.
- Access включает множество разнообразных тем, которые можно применить к форме для придания ей нужного стиля. В режиме макета на вкладке ленты **Конструктор** (Design) представлена коллекция тем. Выберите понравившуюся вам тему и щелкните на ней, чтобы применить ее к форме.

## Макеты элементов управления

Макет элементов управления — это объединение элементов управления в группу, для которой произведено выравнивание по вертикали и горизонтали для единогообразного оформления формы. В форме может быть несколько макетов элементов управлению.

Макет можно рассматривать как таблицу, в столбцах которой размещены элементы управления. Естественно в таблице можно менять только ширину всего столбца или всей строки, а не отдельных ее ячеек. В макете как в таблице можно перемещать значение из одной ячейки в другую.

В форме ТОВАР системой создан макет (см. рис. 5.2), в котором поля и их подписи размещены в четырех столбцах. Подписи полей в форме соответствуют значениям, заданным в соответствующих свойствах при определении структуры.

Выделите макет формы, щелкнув на звездочке в левом верхнем углу макета или выполнив команду **Выбрать макет** (Select Layout) в группе **Строки и столбцы** (Rows & Columns) на ленте **Упорядочить** (Arrange). Затем выполните команду **Сетка** (Gridlines) в группе **Таблица** (Table) на ленте **Упорядочить** (Arrange). Выберите в списке команды **Все** (Both), что означает показать все линии таблицы, а также цвет, тип границы и ширину линий такими, чтобы была отчетливо видна таблица, в которую заключены элементы управления формы.

В рамках одного макета, невозможно изменение ширины и высоты отдельных элементов, а местоположение элемента может меняться только в пределах макета. В то же время макет при перетаскивании элементов для изменения их местоположения может легко расширяться путем добавления новых строк и столбцов.

Элемент можно удалить из макета командой контекстного меню **Макет | Удалить макет** (Layout | Remove Layout), и тогда будет обеспечено его свободное перемещение и изменение размеров. Выделив весь макет и выполнив ту же команду, можно полностью разгруппировать элементы управления. Выделив нужную группу элементов, можно создать новый макет в столбик или табличный соответствующими командами на ленте **Упорядочить** (Arrange) в группе **Таблица** (Table). Именно макет позволит вам выровнять все его элементы.

1. Измените неоправданно большую ширину полей формы. Для этого щелкните на любом из них и перетащите его правую границу. Изменится ширина всех полей в столбце формы. Это объясняется тем, что при создании формы с помощью команды **Форма** (Form) автоматически создается макет в столбик.
2. Удалите поле с фотографией из макета формы командой контекстного меню **Макет | Удалить макет** (Layout | Remove Layout), а подпись поля удалите со всем, выделив ее и нажав клавишу <Delete>.
3. Выделите оставшиеся в двух правых столбцах поля с их подписями и переместите под последний элемент первого столбца. Макет имеет способность расширяться и поэтому перемещенные элементы дополнят два первых столбца макета. При этом из макета удалятся два правых столбца, в которых не останется элементов.

4. Выделяя широкие строки полей с кодом товара, ценой и вложениями, уменьшите их высоту.
5. Измените местоположение и размер фотографии в форме, а подпись этого поля удалите. Чтобы убрать рамку вокруг фотографии, выделите поле и на вкладке ленты **Формат** (Format) в группе **Форматирование элемента управления** (Control Formatting) в списке команды **Контур фигуры** (Shape Outline) выберите **Прозрачный** (Transparent).
6. Подпись поля с вложениями измените на *Сертификат качества*.

## Условное форматирование элементов управления

Для изменения внешнего вида элемента управления в форме в зависимости от одного или нескольких условий используйте условное форматирование. Измените в форме ТОВАР цвет денежных значений в поле ЦЕНА на красный, когда они оказываются выше заданной величины, например 2000 руб.

1. Выберите поле ЦЕНА, в котором нужно изменить цвет значения при заданном условии. На вкладке ленты **Формат** (Format) в группе **Форматирование элемента управления** (Control Formatting) выберите команду **Условное форматирование** (Conditional Formatting). Откроется диалоговое окно **Диспетчер правил условного форматирования** Conditional Formatting Rules Manager() (рис. 5.7).
2. Чтобы создать первое правило условного форматирования, щелкните на кнопке **Создать правило** (New Rule).
3. В окне **Новое правило форматирования** (New Formatting Rule) (рис. 5.8) выберите тип правила и сформируйте описания правил, при которых будет производиться заданное форматирование поля.

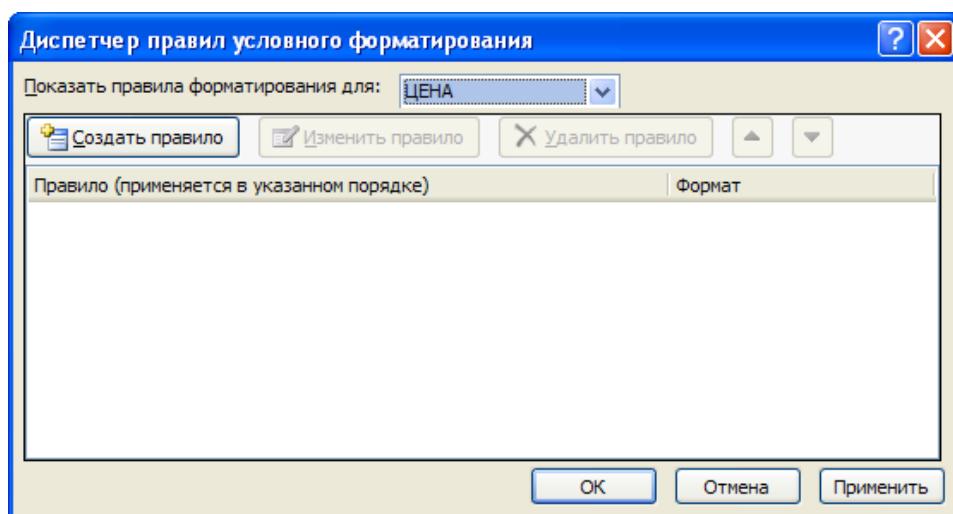


Рис. 5.7. Окно с пустым списком правил форматирования

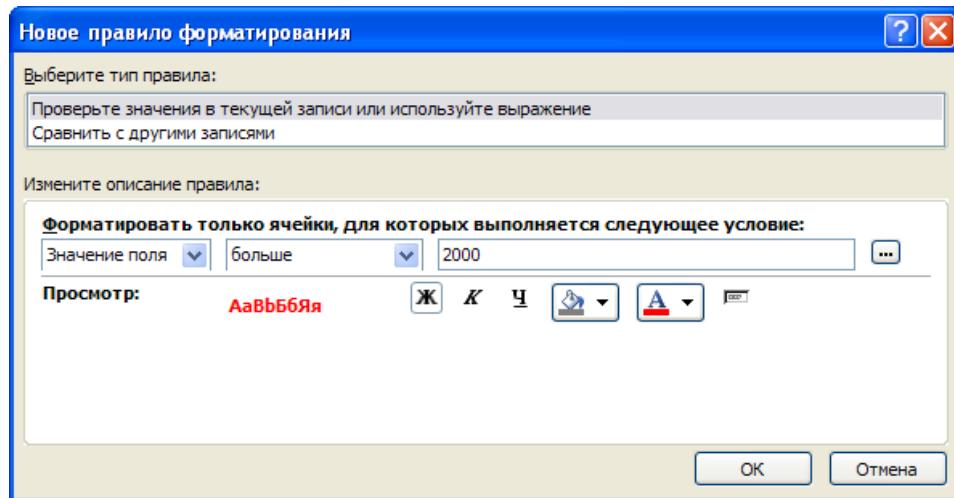


Рис. 5.8. Окно определения условий форматирования и форматов поля

- В первом поле со списком выберите пункт **Значение поля** (Field Value Is), во втором выберите тип сравнения — **больше** (greater than) и введите постоянное значение в третье поле — 2000, не используя знака денежных единиц.
- Выберите начертание шрифта, цвет и другие параметры форматирования, используемые при выполнении условия. Нажмите кнопку **OK**. Выбранное форматирование применяется только в тех случаях, когда значение элемента управления удовлетворяет условию.

Код товара	T007
Наименование товара	Принтер EPSON ST.A4
Цена	2 400,00р.
Единица измерения	штука
Ставка НДС	10%
Наличие товара	<input checked="" type="checkbox"/>
Сертификат качества	
Цена с НДС	2 640,00р.

Рис. 5.9. Форма ТОВАР в режиме макета после доработки

6. Для каждого элемента управления можно указать несколько условий. Чтобы добавить новое условие или удалить существующее, нажмите соответствующие кнопки в окне **Диспетчера правил условного форматирования** (Conditional Formatting Rules Manager).

После выполненных изменений макета форма примет вид, показанный на рис. 5.9. На рисунке выделен один макет в столбик. Выделение макета выполняется щелчком в его левом верхнем углу.

## Свойства формы

Редактирование формы, ее разделов и входящих в нее элементов управления может быть произведено не только графическими средствами, но и путем изменения их свойств.

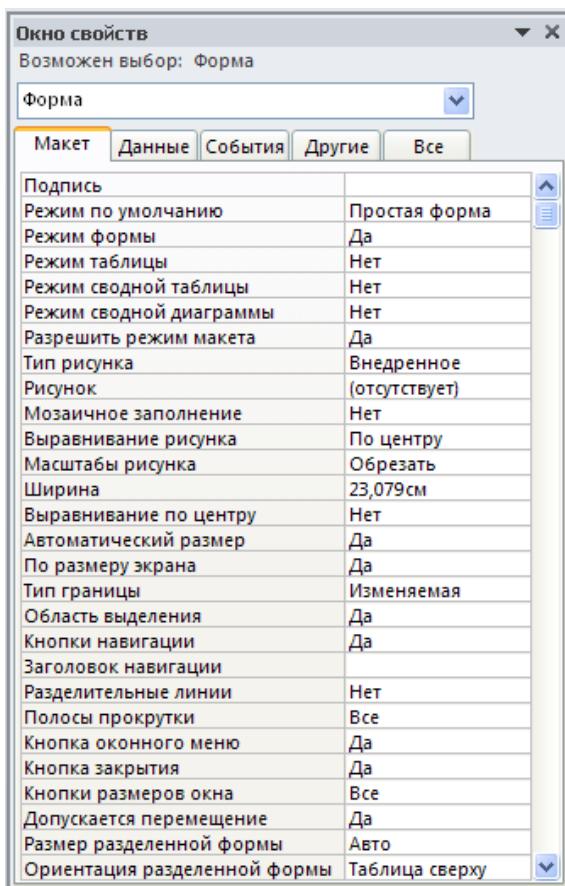


Рис. 5.10. Окно свойств формы

- Для изменения свойств предназначена **Страница свойств** (Property Sheet).  
1. Откройте страницу свойств, выбрав соответствующую кнопку на вкладке ленты **Конструктор** (Design) в группе **Сервис** (Tools). В окне свойств (рис. 5.10) в

- списке поля **Возможен выбор** (Selection Type) отображаются все элементы формы.
2. Выберите в строке списка значение **Форма** (Form). Чтобы убедиться в действенности свойств элементов, в открывшихся свойствах формы на вкладке **Макет** (Format) в строке **Область выделения** (Record Selectors) выберите значение **Нет** (No). Область выделения записи, расположенная слева в области данных, исчезнет из формы.
  3. Верните область выделения в форму.
  4. Перейти к просмотру свойств формы можно, щелкнув на области выделения записи. Следует заметить, что изменение многих свойств формы и ее элементов управления невозможно в режиме макета.

## Добавление полей в форму

В режиме макета возможно удаление и добавление полей в форму. Для удаления поля достаточно выделить его и в контекстном меню выполнить команду **Удалить** (Delete). Для добавления полей в форму щелкните на кнопке **Добавить поля** (Add Existing Fields) на вкладке ленты **Конструктор** (Design) в группе **Сервис** (Tools). Отобразится область **Список полей** (Field List) базовой таблицы или запроса, доступных в форме. Добавить поле в активный макет формы можно двойным щелчком на нем. Для размещения поля в любой макет формы перетащите его туда из области **Список полей** (Field List).

В режиме макета в группе **Элементы управления** (Controls) на вкладке ленты **Конструктор** (Design) представлен набор элементов, для включения которых в форму нет необходимости переходить в режим конструктора. Этот набор содержит элементы для создания поля, надписи, списка, поля со списком, флагка, кнопки и т. п. Набор элементов управления режима макета в сравнении с доступным в режиме конструктора несколько усечен.

## Работа с данными таблицы в режиме формы

Созданная форма ТОВАР предназначена для загрузки, просмотра, корректировки и удаления данных, сохраняемых в одноименной таблице. Основным режимом работы с данными базовой таблицы является режим формы. В режиме макета предусмотрен только просмотр данных, обеспечивающий удобную настройку элементов.

1. Для перехода в режим формы из режима макета щелкните на кнопке **Режим формы** (Form View) в строке состояния или выполните команду **Режим формы** (Form View) контекстного меню формы. Если форма закрыта, выберите ее в области навигации в группе **Формы** (Forms) и в контекстном меню нажмите кнопку **Открыть** (Open).
2. Просмотрите записи таблицы ТОВАР, используя стандартные кнопки панели перехода по записям внизу формы или команды в списке кнопки **Перейти** (Go

- То) на вкладке ленты **Главная** (Home) в группе **Найти** (Find). Измените некоторые значения в полях записи.
3. Для создания новой записи в таблице ТОВАР сделайте текущей новую запись, воспользовавшись кнопкой  на панели перехода по записям или кнопкой **Создать** (New) на вкладке ленты **Главная** (Home) в группе **Записи** (Records), и введите значения в поля формы. Значения, вводимые в поля формы, должны соответствовать типам данных и их свойствам, заданным при определении структуры таблицы.
  4. Для завершения создания (редактирования) записи таблицы достаточно перейти к другой записи или выполнить команду **Сохранить** (Save) на вкладке ленты **Главная** (Home) в группе **Записи** (Records). Запись может быть сохранена только при условии ввода значений ключевых полей таблицы, а также полей, которые были определены в свойствах поля как обязательные.
  5. Чтобы удалить запись, выделите ее, щелкнув на области выделения в левой части области данных формы или выполнив команду **Выбрать | Выделить** (Select | Select) в группе **Найти** (Find), и выполните команду **Удалить | Удалить запись** (Delete | Delete Record) в группе **Записи** (Records).
  6. Закройте форму. Откройте таблицу ТОВАР и убедитесь, что все сделанные изменения внесены в таблицу.

### Задание 5.1. Создание однотабличной формы

Создайте однотабличную форму для работы со справочником покупателей, отредактируйте ее в режиме макета и произведите добавление записей в таблицу. Если в базе данных *Поставка товаров*, выбрав таблицу ПОКУПАТЕЛЬ, для создания формы использовать команду **Форма** (Form), автоматически будет создана форма, содержащая встроенную подчиненную таблицу ДОГОВОР. Источником записей главной формы будет таблица ПОКУПАТЕЛЬ. Такое поведение команды **Форма** (Form) вызвано тем, что таблица ПОКУПАТЕЛЬ имеет единственную подчиненную таблицу ДОГОВОР, с которой она находится в отношениях 1 : M, и эта связь определена в схеме данных. В главной форме будет отображаться запись таблицы ПОКУПАТЕЛЬ, а в подчиненной связанные с ней записи из таблицы ДОГОВОР.

Для создания формы, отображающей сведения только о покупателе, прежде чем воспользоваться командой **Форма** (Form), которая без вмешательства пользователя создаст однотабличную форму в столбец, следует в схеме данных удалить связь таблицы ПОКУПАТЕЛЬ с таблицей ДОГОВОР.

Кроме того, можно воспользоваться услугами мастера форм, который позволяет определить в качестве источника записей формы любое число таблиц, выбрать из них необходимые поля и способ их отображения в форме: в один столбец или выровненный для одновременного отображения одной записи, табличный и ленточный для отображения сразу всех записей.

## Создание формы на основе запроса

Если в базе данных уже имеется запрос, в котором выбраны таблицы, нужные для получения результата, определены поля, которые необходимо включить в результат, параметры, сделаны вычисления, целесообразно создавать форму на основе такого запроса.

В созданном ранее запросе из таблицы НАКЛАДНАЯ выбираются все накладные, по которым производилась отгрузка в заданном месяце. В вычисляемом поле месяца задано имя параметра — [Номер месяца] и запрос сохранен под именем **Накладные месяца** (рис. 5.11).

Поле:	НОМ_НАКЛ	КОД_СК	ДАТА_ОТГР	Месяц: Month([НАКЛАДНАЯ]![ДАТА_ОТГР])
Имя таблицы:	НАКЛАДНАЯ	НАКЛАДНАЯ	НАКЛАДНАЯ	
Сортировка:				
Вывод на экран:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Условие отбора:	[Номер месяца]			
или:				

Рис. 5.11. Запрос для выборки накладных, выписанных в заданном месяце

Выбор запроса с параметром в качестве источника записей позволит создать форму, обеспечивающую диалог с пользователем и вывод только тех записей, которые соответствуют заданным в диалоге условиям отбора.

Для создания такой формы можно использовать любую из команд автоматического создания формы одним щелчком мыши: **Форма (Form)**, **Разделенная форма (Split Form)** или **Несколько элементов (Multiple Items)** (см. рис. 5.1), размещенных в группе **Формы (Forms)** на вкладке ленты **Создание (Create)**.

- Выделив запрос в области навигации, щелкните на кнопке **Несколько элементов (Multiple Items)**. В результате выполнения этой команды откроется диалоговое окно **Введите значение параметра (Enter Parameter Value)**.
- Введите номер месяца, например 3. Форма откроется в режиме макета и отобразит записи, соответствующие введенному значению параметра (рис. 5.12).

	Номер накладной	Код склада	Дата отгрузки	Месяц
▶	H001	C02	11.03.2010	3
	H002	C03	24.03.2010	3
*	H004	C01	25.03.2010	3
			07.03.2010	

Запись: 1 из 3   Нет фильтра   Поиск

**Рис. 5.12.** Форма, полученная по команде **Несколько элементов**

По команде **Несколько элементов** (Multiple Items) в форме автоматически создается табличный макет. В табличных макетах элементы расположены в двух разделах формы: надписи располагаются в заголовке формы, поля записей — по строкам в области данных.

- Сохраните форму, воспользовавшись командой контекстного меню формы, и закройте ее.
- Выберите форму в области навигации, выполните в ее контекстном меню команду **Открыть** (Open) или дважды щелкните на ней. Откроется диалоговое окно для ввода значения параметра. Введите нужное значение. В открывшейся форме будут представлены записи, выбранные в соответствии с заданным условием отбора.
- Для построенной формы открыт режим ввода новых записей. Очевидно это находится в противоречии с заданным условием отбора, поэтому в режиме макета целесообразно открыть **Страницу свойств** (Property Sheet) формы и на вкладке **Данные** (Data) в строке **Разрешить добавление** (Allow Additions) выбрать значение **Нет** (No). После изменения свойства в форме не будет отображаться строка новой записи, а также поменяет соответствующий значок на панели перехода по записям и кнопка **Создать** (New) в группе **Записи** (Records) на вкладке ленты **Главная** (Home). В приведенном примере добавление новой записи вообще невозможно, т. к. в форме не представлено обязательное поле вторичного ключа — номер договора.

## Создание разделенной формы

Разделенная форма позволяет синхронно отображать данные одного источника в двух представлениях — в режиме формы и в режиме таблицы. В любой части разделенной формы можно производить изменение данных — добавление, обновление или удаление, которые тут же будут отображаться в другом представлении. При выделении поля в одной части формы выделяется то же поле в другой части.

Создайте однотабличную разделенную форму для работы с данными таблицы ТОВАР. Выберите таблицу ТОВАР в области навигации и выполните команду **Разделенная форма** (Split Form) на вкладке ленты **Создание** (Create) в группе **Формы** (Forms). Эта команда обеспечит автоматическое создание формы на основе только одной выбранной таблицы. Созданная форма отобразится в режиме макета.

Работа с разделенной формой дает преимущества обоих типов представления данных. Например, можно воспользоваться табличной частью формы, чтобы быстро найти запись о конкретном товаре, а затем просмотреть или изменить ее в другой части формы.

Код тов.	Наименование това	Цена	Единица	Ставка НД	НАЛИЧИ
T001	Монитор 17LG	7 000,00р.	штука	5%	<input checked="" type="checkbox"/>
T002	FDD 3,5	1,00р.	коробка	6%	<input type="checkbox"/>
T003	HDD Maxtor 20GB	1,00р.	штучки	5%	<input checked="" type="checkbox"/>
T004	Корпус MiniTower	10,00р.	штука	10%	<input type="checkbox"/>
T005	CD-ROM Panasonic IDE	3,00р.	штуки	30%	<input type="checkbox"/>
T006	DIMM 64M PC100	300,00р.	штука	15%	<input type="checkbox"/>
T007	Принтер EPSON ST.A4	2 400,00р.	штука	10%	<input checked="" type="checkbox"/>
T008	СканерAcer	2 338,00р.	штуки	16%	<input type="checkbox"/>
T009	Зв. Карта Genius Liv	789,00р.	штука	10%	<input type="checkbox"/>
T010	Модем Genius ext	1 295,00р.	штук	10%	<input type="checkbox"/>
*		0,00р.			<input checked="" type="checkbox"/>

Рис. 5.13. Разделенная форма ТОВАР

Если в вашей базе данных существуют формы, основанные на одной таблице или запросе, совсем не обязательно создавать разделенную форму с тем же источником данных заново, достаточно изменить свойство этой формы **Режим по умолчанию** (Default View). В нашей базе данных ранее была создана форма ТОВАР для просмотра и изменения сведений о товарах. Откройте форму в режиме конструктора. Для этого выберите ее в области навигации, и в контекстном меню выполните

команду **Конструктор** (Design View). На вкладке ленты **Конструктор** (Design) в группе **Сервис** (Tools) выполните команду **Свойства страницы** (Property Sheet). Измените на вкладке **Макет** (Format) значение свойства формы **Режим по умолчанию** (Default View) с **Простая форма** (Single Form) на **Разделенная форма** (Split Form).

Форма ТОВАР примет вид, представленный на рис. 5.13.

Расположение таблицы в верхней или нижней части окна разделенной формы определяется в свойстве формы **Ориентация разделенной формы** (Split Form Orientation). Свойство может принимать значения **Таблица сверху** (Datasheet on Top), **Таблица снизу** (Datasheet on Bottom), **Таблица слева** (Datasheet on Left) и **Таблица справа** (Datasheet on Right). Изменить значение этого свойства можно только в режиме конструктора.

## Вычисления в форме

Вычисления в форме могут осуществляться как в каждой записи формы, так и для группы записей при формировании итоговых величин. Вычисляемые величины отображаются в поле формы, но в отличие от создаваемых в таблице вычисляемых полей не сохраняются в таблице. Для сохранения результатов вычислений в таблице базы данных требуется подготовка макроса или процедуры на VBA (Visual Basic for Applications). Примеры макросов, обеспечивающих обновление поля таблицы значениями, вычисленными в форме, будут рассмотрены в главе 8.

### Вычисления в каждой записи формы

Чтобы произвести вычисления на основе данных каждой записи формы, необходимо создать элемент управления **Вычисляемое поле**, источником данных которого является выражение для расчета. Для создания такого элемента управления откройте форму в режиме макета или конструктора. Оба режима на ленте конструктора имеют в группе **Элементы управления** (Controls) кнопку **Поле** (Text Box). Нажмите ее и разместите вычисляемое поле в нужном месте области данных. Затем введите выражение в элемент, называемый **Свободный** (Unbound). Выражение должно начинаться со знака равенства (=). В качестве операндов выражения чаще всего используются имена полей и константы, а в качестве операторов — знаки арифметических операций.

Пусть необходимо подсчитать и отобразить в форме величину НДС каждого товара в денежном выражении. Откройте созданную ранее простую форму ТОВАР в режиме конструктора. Создайте вычисляемый элемент управления и запишите в него выражение:

```
= [ЦЕНА] * [СТАВКА_НДС]
```

В связанную с полем надпись запишите: Стоимость НДС.

Выражение будет введено в свойство созданного элемента управления **Данные** (Control Source), размещенное на вкладке **Данные** (Data). В режиме макета выра-

жение нужно записать непосредственно в это свойство. Добавить вычисляемое поле в имеющийся в форме макет значительно проще в режиме макета.

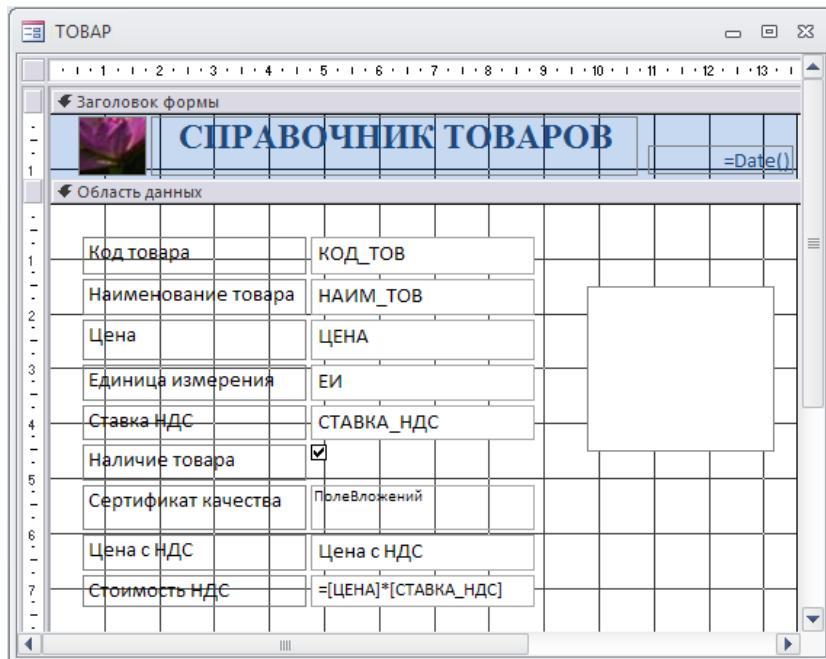


Рис. 5.14. Форма с вычисляемым полем в режиме конструктора

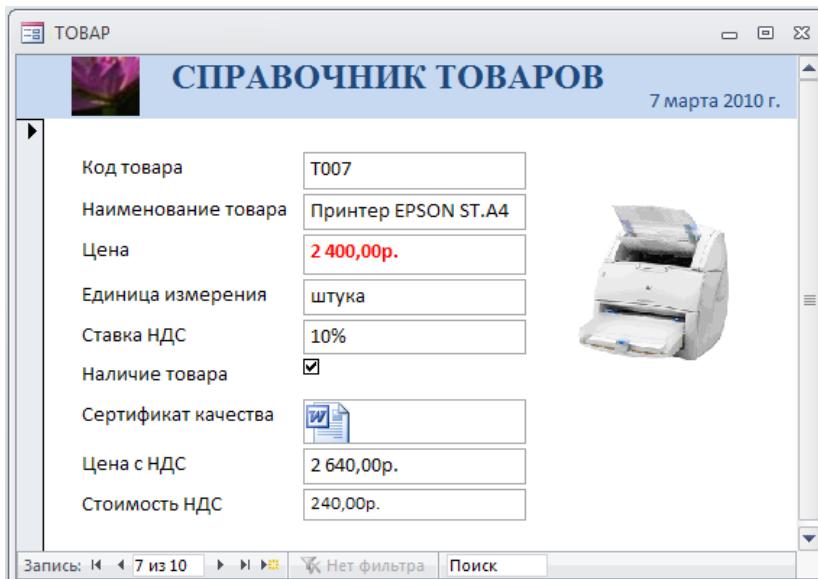


Рис. 5.15. Форма с вычисляемым полем в режиме просмотра

Измените формат поля, выбрав на вкладке **Макет** (Format) в списке свойства **Формат поля** (Format) значение **Денежный** (Currency).

Используйте **Формат по образцу** (Format Painter) для оформления нового элемента в стиле всех других элементов.

Форма в режиме конструктора и режиме просмотра, полученная после создания вычисляемого поля, приведена на рис. 5.14 и 5.15.

Создание вычисляемого поля в форме аналогично созданию такого поля в запросе.

## Вычисление итоговых значений

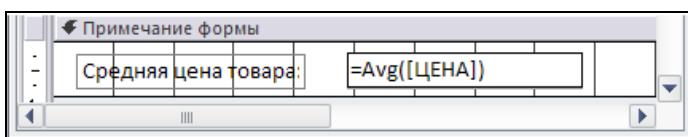
Вычисление итоговых значений в форме выполняется в примечании формы с помощью встроенных статистических функций, записываемых в выражениях в вычисляемых элементах управления.

Откройте форму ТОВАР в режиме конструктора. Чтобы поместить вычисляемое поле в область примечания формы, расширьте ее, установив курсор на границе, так чтобы он принял вид двунаправленной стрелки, и перетащив границу вниз. Затем щелкните на кнопке **Поле** (Text Box) в группе **Элементы управления** (Controls) и в области примечания вычертите вычисляемый элемент управления.

Для расчета среднего значения цены всех товаров запишите в вычисляемый элемент управления выражение:

= Avg ([ЦЕНА]),

а в надпись — **Средняя цена товара:** (рис. 5.16). Измените формат поля на **Денежный** (Currency).



**Рис. 5.16.** Примечание формы в режиме конструктора с полем для вычисления итогов

После перехода в режим просмотра в форме отображается результат расчета (рис. 5.17).



**Рис. 5.17.** Форма в режиме просмотра с расчетным полем для итоговых значений

## Многотабличные формы

Многотабличная форма создается для работы с данными нескольких взаимосвязанных таблиц. Источником данных такой формы является многотабличный запрос. При этом форма также может быть простой, отображающей одну запись в столбик, или ленточной, отображающей все записи в табличном виде с надписями в заголовке формы. Для создания такой формы может быть использована команда **Форма (Form)** или **Несколько элементов (Multiple Items)**. Форма, построенная на многотабличном запросе, может быть названа одиночной.

Многотабличная форма может быть составной: состоять из главной формы и одной или нескольких подчиненных включаемых форм. Подчиненная форма, как правило, строится на основе таблицы, подчиненной таблице-источнику записей главной формы, т. е. находится с ней в отношении  $1 : M$ . Подчиненная форма отображает данные из всех записей подчиненной таблицы, которые связаны с записью, отображаемой в главной форме. Для разработки такой формы можно проделать следующее:

1. На основе главной таблицы создать командой **Форма (Form)** простую форму с макетом в столбик.
2. На основе подчиненной таблицы командой **Несколько элементов (Multiple Items)** создать ленточную форму, это многозаписевая форма.

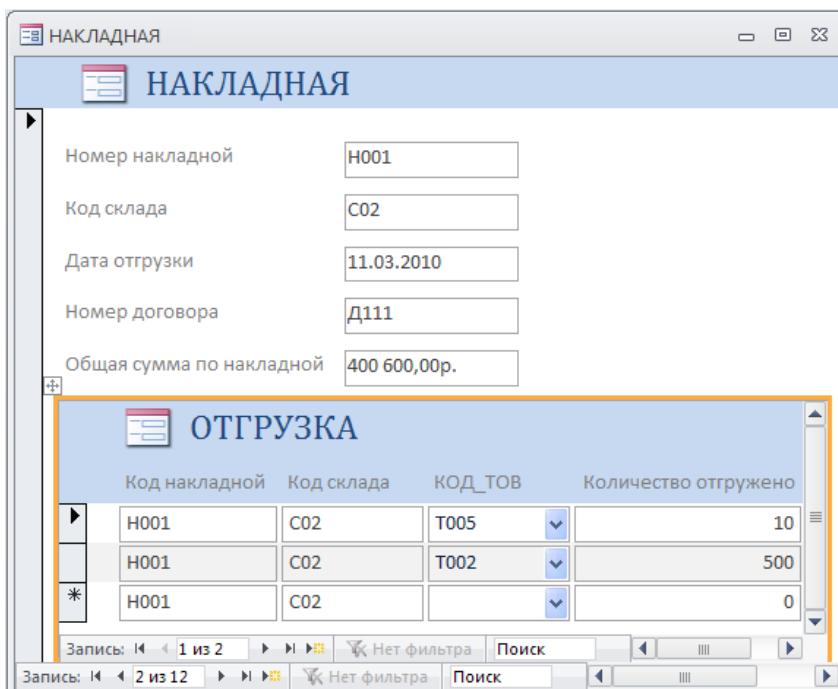


Рис. 5.18. Составная форма в режиме макета

3. Закрыть вторую форму, оставив первую открытой в режиме макета.
4. Перетащить вторую форму из области навигации в первую форму, она отобразится в элементе управления **Подчиненная форма/отчет** (Subform/Subreport).
5. Если не установилась связь между формами, следует в главной форме открыть свойства элемента управления **Подчиненная форма/отчет** (Subform/Subreport) и на вкладке **Данные** (Data) заполнить строки **Основные поля** (Link Master Fields) и **Подчиненные поля** (Link Child Fields). Это можно сделать с помощью построителя, вызываемого в строке первого свойства.

Воспользовавшись приведенным алгоритмом, несложно создать составную форму на основе таблиц НАКЛАДНАЯ и ОТГРУЗКА (рис. 5.18). Связь этих форм осуществляется по составному ключу связи НОМ\_НАКЛ и КОД\_СК. В строке свойств они записываются через точку с запятой.

Очевидно, для разработки полноценной формы для работы с накладными необходимо дополнить главную форму данными из таблиц ДОГОВОР и ПОКУПАТЕЛЬ, а подчиненную форму данными из таблицы ТОВАР.

Решить эту задачу можно, выполнив следующие действия:

1. Создать запросы на выборку для источника записей главной формы на основе таблиц НАКЛАДНАЯ, ДОГОВОР и ПОКУПАТЕЛЬ, подчиненной формы — ОТГРУЗКА и ТОВАР, выбирая необходимые поля.
2. На основе первого запроса создать простую форму командой **Форма** (Form), на основе второго ленточную командой **Несколько элементов** (Multiple Items).
3. Закрыть вторую форму и затем перетащить ее из области навигации в первую.

Код накладной	Код склада	КОД_ТОВ	Наименование товара	Цена	Количество отгружено
H002	C01	T001	Монитор 17LG	7 000,00р.	11
H002	C01	T002	FDD 3,5	500,00р.	330

**Рис. 5.19.** Составная форма для работы с накладными

4. Для установки связи главной и подчиненной форм откройте свойства элемента управления **Подчиненная форма/отчет** (Subform/Subreport) и на вкладке **Данные** (Data) заполните строки **Основные поля** (Link Master Fields) и **Подчиненные поля** (Link Child Fields).

На рис. 5.19 приведена составная форма, в которой источником записей простой главной формы является первый запрос, а источником записей ленточной подчиненной формы второй.

## Создание многотабличной формы с помощью мастера

Основным средством создания многотабличной формы можно считать мастер форм, который, запросив у пользователя сведения о включаемых в форму полях из нескольких взаимосвязанных таблиц и запросов, создает составную или одиночную форму. При этом мастер может сам создать запросы на выборку, используемые в качестве источника записей формы.

Полученная с помощью мастера форма при необходимости может быть отредактирована в режиме макета или конструктора. Режим конструктора позволяет детально просмотреть структуру формы. Настроить любой ее раздел. Некоторые задачи удобнее выполнять в режиме конструктора, а не макета, некоторые могут выполняться только в режиме конструктора.

С помощью мастера создайте форму для работы с данными о покупателях и их договорах. Очевидно, такая форма должна строиться на основе двух таблиц: ПОКУПАТЕЛЬ и ДОГОВОР, находящихся в отношении 1 : M и связанных полем КОД\_ПОК (код покупателя). Для одновременной работы с записью главной и некоторыми связанными записями подчиненной таблицы целесообразно построить многотабличную составную форму, в которой в главную форму будет встроена подчиненная форма.

Для вызова мастера форм выполните на вкладке ленты **Создание** (Create) в группе **Формы** (Forms) команду **Мастер форм** (Form Wizard). Отобразится окно мастера **Создание форм** (Form Wizard), представленное на рис. 5.20.

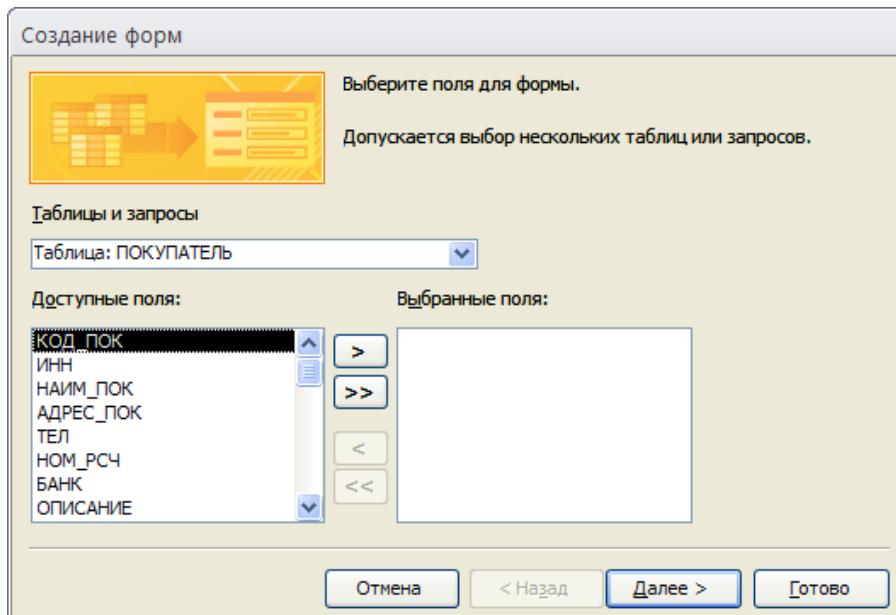
Если предварительно в области переходов не была выбрана таблица ПОКУПАТЕЛЬ, которая представляет сторону "один" в отношении 1 : M таблиц формы, выберите ее в раскрывающемся списке **Таблицы и запросы** (Tables/Queries) диалогового окна мастера. Далее отберите из списка **Доступные поля** (Available Fields), необходимые в форме, и переходите к выбору полей из подчиненной таблицы ДОГОВОР.

### ЗАМЕЧАНИЕ

Порядок выбора таблиц не имеет принципиального значения.

Из таблицы ПОКУПАТЕЛЬ обязательно выберите ключевое поле КОД\_ПОК (код покупателя) и все другие обязательные поля, иначе форма будет непригодна для ввода сведений о новых покупателях. Из таблицы ДОГОВОР обязательно вы-

берите ключевое поле КОД\_ДОГ (код договора), а код покупателя (поле связи таблиц) выбирать не следует. Значения этого поля станут неоднократно повторяться в подчиненной форме, если покупатель заключил несколько договоров, а ввод значения этого вторичного ключа в таблицу ДОГОВОР обеспечивается в форме автоматически.



**Рис. 5.20.** Выбор таблиц и полей для создаваемой формы

### **ВНИМАНИЕ!**

Если выбраны таблицы, для которых не определена связь в схеме данных, появится сообщение о невозможности создать форму. При этом мастер предлагает или изменить состав полей, или выводит схему данных для возможного определения нужных связей. После изменения связей мастер должен запускаться заново.

После выбора полей для обеих таблиц и нажатия кнопки **Далее** (Next) в окне **Создание форм** (Form Wizard) (рис. 5.21) в списке **Выберите тип представления данных** (How do you want to view your data) надо выделить имя таблицы ПОКУПАТЕЛЬ, которая является источником данных главной формы. Поскольку эта таблица была выбрана первой, то она уже выделена.

Таблица ПОКУПАТЕЛЬ, как источник данных основной формы, в данном случае является главной по отношению к другой выбранной для формы таблице ДОГОВОР, поэтому в окне **Создание форм** (Form Wizard) предлагается выбрать один из двух возможных вариантов подключения формы:

- ❖ для включения подчиненной формы в главную надо отметить переключатель **Подчиненные формы** (Form with subform(s));

- ❖ для включения кнопки, вызывающей связанную форму, надо выбрать **Связанные формы** (Linked forms).

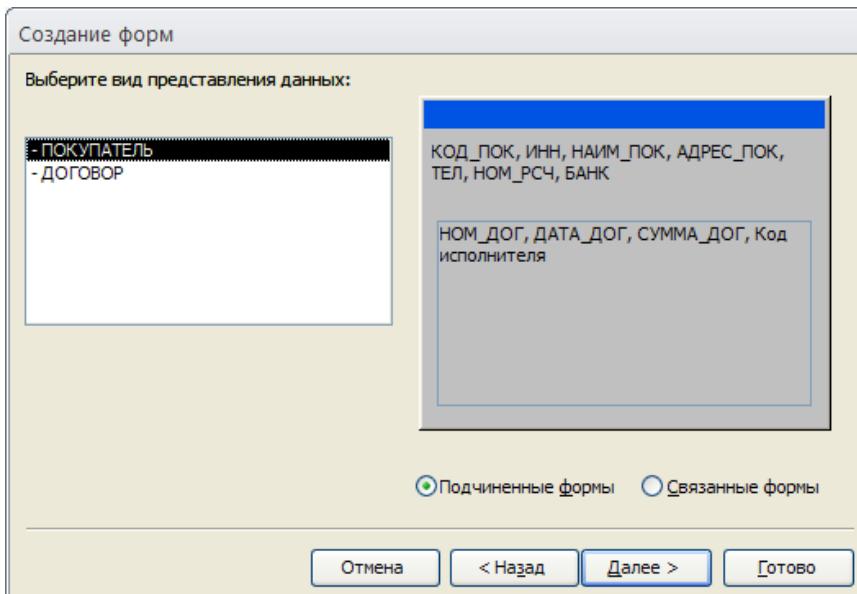


Рис. 5.21. Выбор варианта включения подчиненной формы

Выберем первый вариант с непосредственным включением подчиненной формы. На рис. 5.18 видно, как распределяются поля в основной и подчиненной частях формы при выборе этого варианта.

### ЗАМЕЧАНИЕ

Если таблица-источник основной формы является *подчиненной* по отношению к другой таблице, тоже выбранной для формы, то создаваемая многотабличная форма не будет включать подчиненную форму. При этом форма будет содержать в одной записи поля подчиненной таблицы и отобранные поля из главной таблицы. В окне **Создание форм** (Form Wizard) автоматически будет установлен тип формы **Одиночная форма** (Single Form).

В следующем диалоговом окне предоставляется возможность выбрать вид отображения данных в подчиненной форме: **ленточный** (tabular) или **табличный** (datasheet) (рис. 5.22). При выборе ленточного вида для списка выводимых в подчиненной форме записей названия столбцов (полей) берутся из подписи, заданной в общих свойствах поля при конструировании подчиненной таблицы.

В следующем диалоговом окне мастера (рис. 5.23) предоставляется возможность выбрать стиль оформления, который определяет общий вид формы, отображение надписей и значений полей в форме.

В последнем диалоговом окне **Создание форм** (Form Wizard) можно изменить имена основной и подчиненной формы, если был выбран вариант с непосредствен-

ным включением подчиненной формы (рис. 5.24), или имя связанной формы, если был выбран вариант включения кнопки, вызывающей связанную форму.

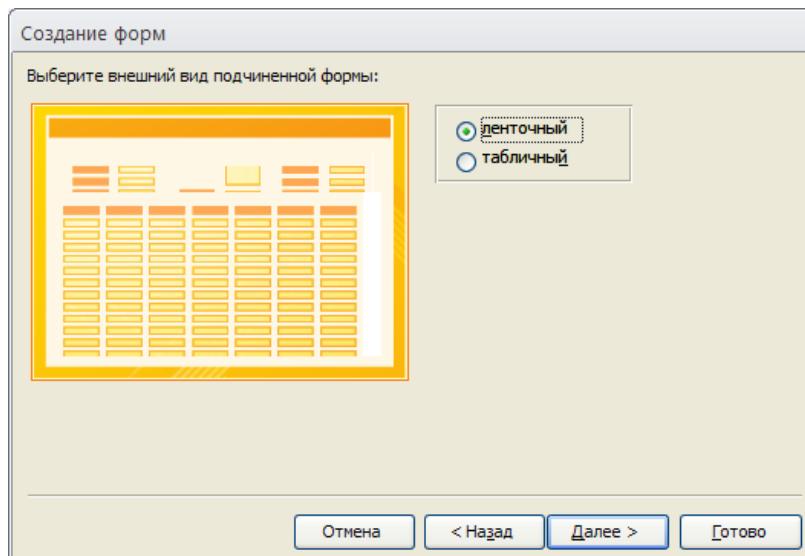


Рис. 5.22. Выбор ленточного вида для подчиненной формы

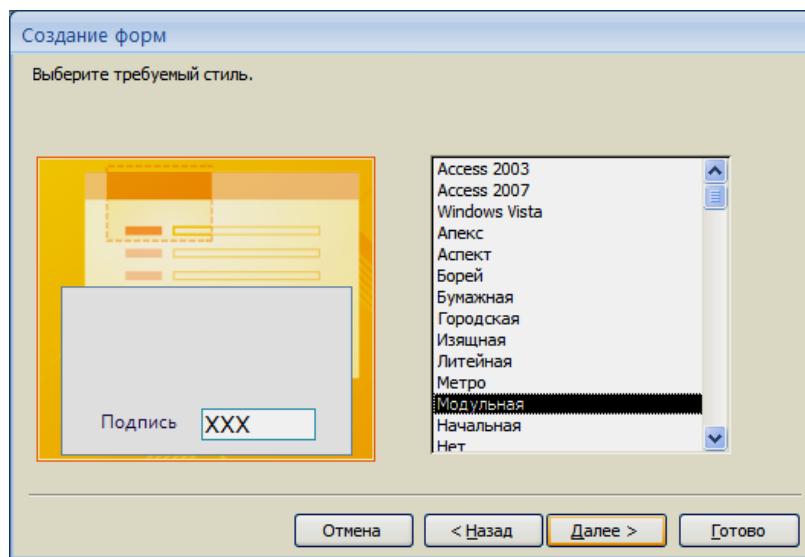


Рис. 5.23. Выбор оформления для создаваемой формы

В том же окне можно выбирать дальнейшие действия — **Открыть форму для просмотра и ввода данных** (Open the form to view or enter information) или **Изменить макет форм** (Modify the form's design).

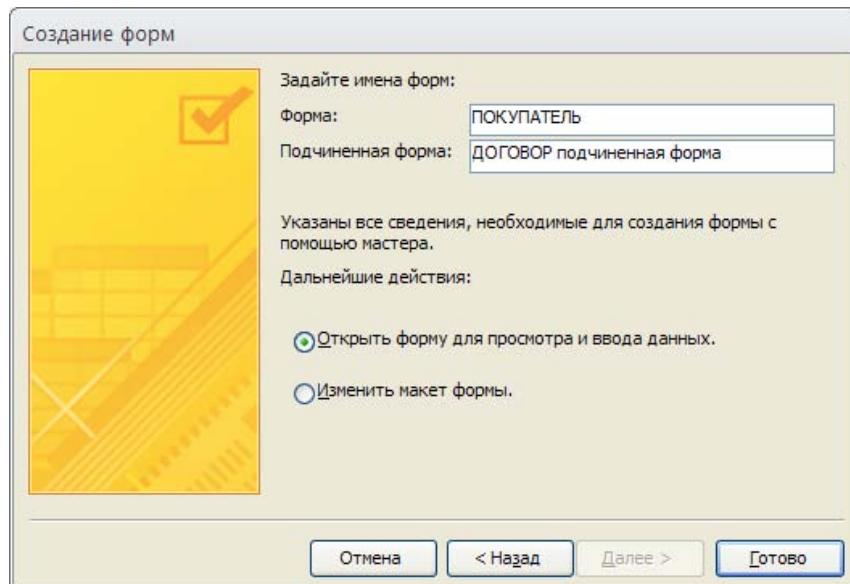


Рис. 5.24. Задание имен форм и выбора режима отображения

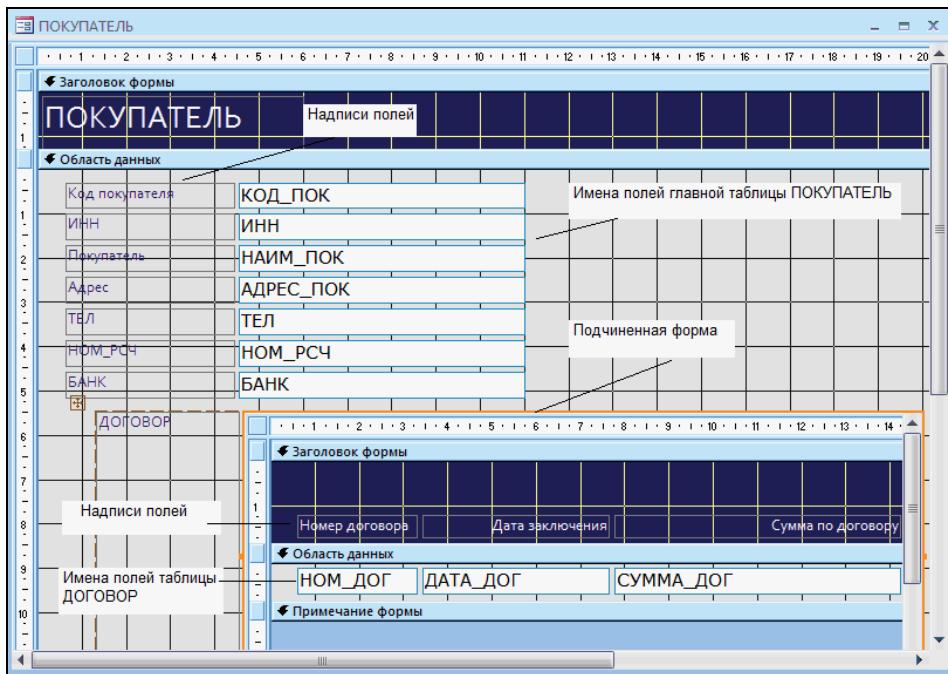
Номер договора	Дата заключения	Сумма по договору
Д222	25.02.2007	778 024,00р.
Д777	14.06.2007	250 000,00р.
Д911	12.12.2007	3 500,00р.
*		0,00р.

Рис. 5.25. Форма, открытая для просмотра, корректировки и ввода данных

Если был выбран вариант **Открыть форму для просмотра и ввода данных** (Open the form to view or enter information), после нажатия кнопки **Готово** (Finish)

мастер завершит создание формы и откроет ее в режиме формы (рис. 5.25), предназначенном для работы с данными таблиц, на которых построена форма. Сохранение обеих форм, созданных мастером, производится автоматически.

При выборе варианта **Изменить макет формы** (Modify the form's design) (см. рис. 5.24) форма после завершения работы мастера выводится в режиме конструктора (рис. 5.26), позволяющем произвести нужную доработку. Форма состоит из главной и подчиненной. В главной — в столбик представлены выбранные поля источника данных главной формы. В подчиненной — в табличном макете представлены выбранные поля источника данных подчиненной формы.



**Рис. 5.26.** Форма в режиме конструктора после завершения работы мастера

В процессе доработки формы, используя технику редактирования формы, можно перемещать поля в главной форме, менять их свойства, в том числе шрифт и размеры, подпись поля, формировать текст в заголовке формы. Аналогичные действия по доработке выполняются для подчиненной формы.

Следует отметить исключительную простоту и универсальность всех действий по изменению размеров, перемещению любого элемента, редактированию подписей и подобных действий. На этапе доработки можно выполнить и более сложные действия по редактированию формы и настроить составную форму в соответствии с любыми требованиями к интерфейсу пользователя. Далее рассмотрены более подробно возможности **Конструктора** (Design) при создании и редактировании форм.

## Создание одиночной многотабличной формы

Одиночную форму, включающую поля из нескольких связанных таблиц, позволяет быстро построить инструмент **Пустая форма** (Blank Form). Выполните команду **Пустая форма** (Blank Form) на вкладке ленты **Создать** (Create) в группе **Формы** (Forms). Откроется пустая форма в режиме макета и отобразится область **Список полей** (Field List). В списке перечислены все таблицы базы данных и предоставляется возможность открыть список полей каждой из них. Чтобы добавить поле в форму, дважды щелкните по нему или перетащите в форму. Для отображения каждого поля Access создает в форме соответствующий элемент управления и привязывает его к полю. Кроме того, для элемента управления создается присоединенная надпись.

Важно понимать, что создание формы начинается с определения источника записей формы. Добавление полей в форму из **Списка полей** (Field List) автоматически решает эту задачу. При добавлении полей в форму автоматически создается инструкция SQL — `SELECT`. Эта инструкция записывается в строку свойств формы **Источник записей** (Record Source). В графическом представлении ее можно просмотреть в построителе запросов, который можно вызвать командой **Изменение источника записи** (Edit Record Source) контекстного меню **Списка полей** (Field List). При последующих добавлениях полей из списка источников автоматически корректируется.

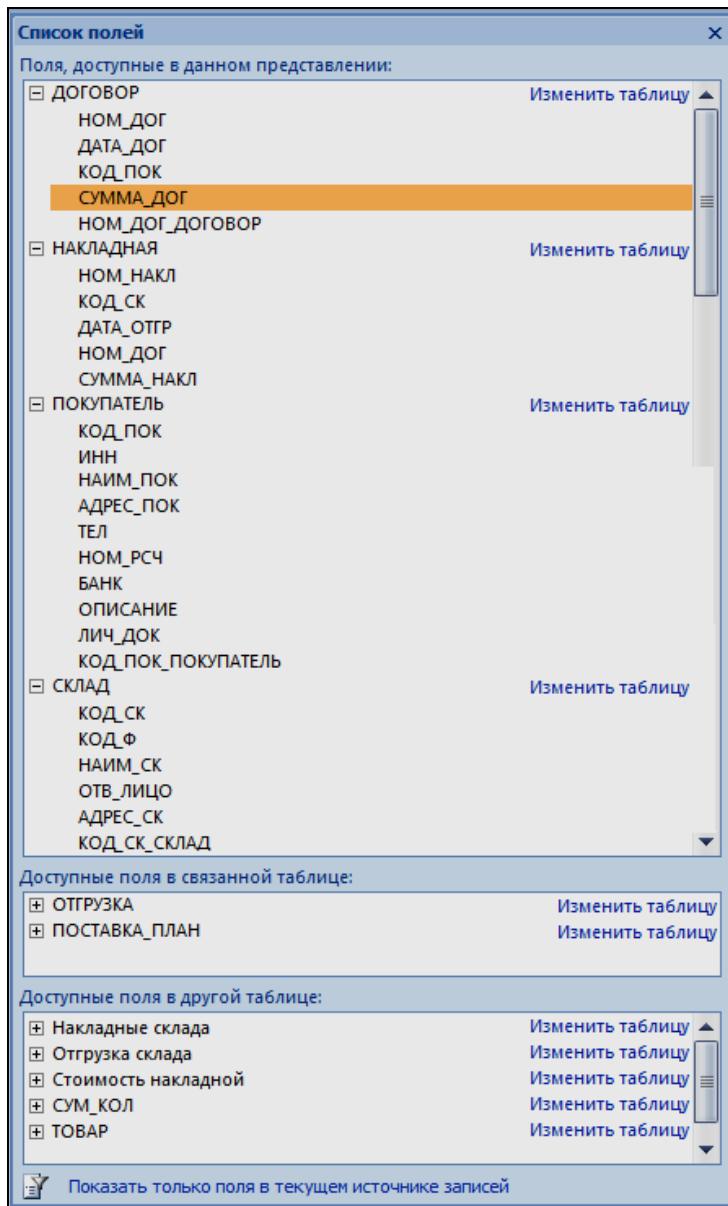
Создание одиночной формы следует начинать с включения полей подчиненной таблицы. Эта таблица определит набор записей, отображаемых через форму, ее запись станет основой для образования записи формы. Далее можно добавлять поля из главных по отношению к ней таблиц.

Создайте форму для просмотра накладных, выписанных для отгрузки товаров. При этом включите в просматриваемую накладную сведения о договоре, на основании которого делается отгрузка, о покупателе и складе, с которого отгружается товар.

Выполните команду **Пустая форма** (Blank Form). В области **Список полей** (Field List) откройте список полей таблицы НАКЛАДНАЯ, щелкнув знак "плюс" (+) рядом с ее именем. Добавьте в форму необходимые поля. Далее последовательно добавляйте поля из таблиц ДОГОВОР, ПОКУПАТЕЛЬ и СКЛАД. После добавления полей из всех таблиц область **Список полей** (Field List) примет вид, показанный на рис. 5.27.

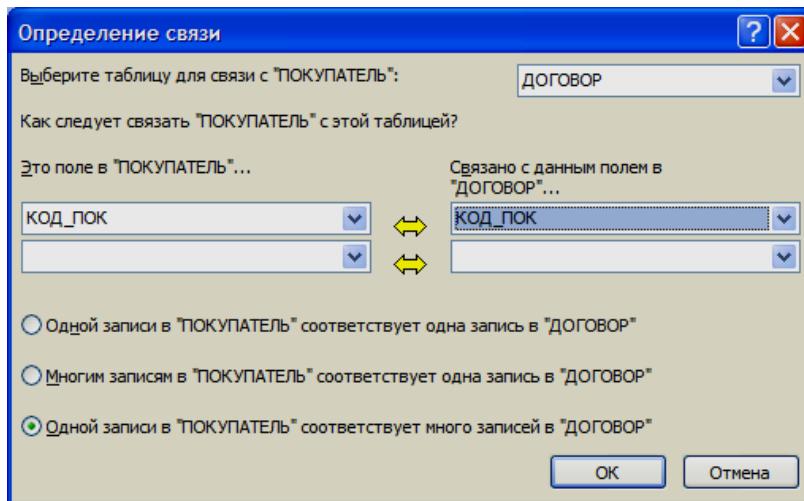
Если при добавлении поля некоторой таблицы обнаруживается, что эта таблица не связана с таблицей, поля которой уже включены в форму, будет выведено окно для определения связи между ними. На рис. 5.28 показано такое окно при добавлении поля из таблицы ПОКУПАТЕЛЬ в случае, когда в форму уже включены поля из договора, а в схеме данных отсутствует связь между таблицами.

На основе выбранных таблиц автоматически создается источник записей формы — запрос на выборку. В этом запросе устанавливаются и созданные при перетаскивании полей в форму связи между таблицами.



**Рис. 5.27.** Область **Список полей**  
при создании одиночной формы Накладная

Просмотреть и изменить запрос, являющийся источником записей формы, можно по команде **Изменение источника записи** (Edit Record Source) контекстного меню области **Список полей** (Field List). Соответствующую инструкцию SQL можно просмотреть в свойстве формы **Источник записей** (Record Source), представленном на вкладке **Данные** (Data).



**Рис. 5.28.** Определение связи между таблицами при создании одиночной формы Накладная

Код покупателя:	Д111
Покупатель:	Перспектива
Номер накладной:	Н001
Код склада:	С01
Наименование склада:	Главный
Ответственное лицо:	Иванов Т.С.
Дата отгрузки:	09.03.2007
Общая сумма по накладной:	56 060,00р.
Номер договора:	Д111
Дата заключения:	05.01.2007
Сумма по договору:	1 135 400,00р.

**Рис. 5.29.** Форма Накладная\_одиночная, созданная с помощью инструмента **Пустая форма**

Пользуясь инструментами группы **Элементы управления** (Controls) на вкладке **Работа с макетами форм | Формат** (Form Layout Tools | Format), добавьте в

форму эмблему компании, заголовок, дату и время. Сохраните форму под именем Накладная\_одиночная.

Форма, построенная с помощью инструмента **Пустая форма** (Blank Form), представлена на рис. 5.29.

## Создание и редактирование формы в режиме конструктора

Средствами конструктора форм можно создать любую форму с нуля. Ранее разработанную любыми средствами форму можно в любой момент отредактировать в режиме конструктора. При создании и редактировании формы может выполняться: определение источника данных формы, добавление новых полей и надписей, включение полей со списком, создание кнопок, добавление подчиненных форм, внедрение объектов из других приложений, например рисунков, диаграмм и т. п.

Наиболее точная и полная настройка структуры и внешнего вида всех разделов и элементов формы производится в режиме конструктора. Ряд элементов управления доступен только в режиме конструктора.

### Создание новой формы конструктором

Конструирование формы начинается выполнением команды **Конструктор форм** (Form Design) на вкладке ленты **Создать** (Create) в группе **Формы** (Forms). В результате выполнения команды открывается окно пустой формы в режиме конструктора с именем формы по умолчанию — **Форма1** (Form1). В пустой форме конструктора представлен только один раздел — **Область данных** (Detail).

Начать конструирование формы следует с определения ее источника данных. Источник данных может быть создан различными способами. Можно открыть свойства формы и на вкладке **Данные** (Data) в строке **Источник записей** (Record Source) открыть список и выбрать таблицу или запрос. Здесь же, воспользовавшись услугами построителя запросов, можно создать запрос в режиме конструктора, а соответствующая инструкция SQL будет записана в строку свойства. Наиболее простым способом определения источника записей является использование области **Список полей** (Field List), в которой отображаются поля всех таблиц базы данных. При перетаскивании нужных полей в форму автоматически строится запрос на выборку — источник записей формы. Для отображения списка полей щелкните на кнопке **Добавить поля** (Add Existing Fields) на вкладке ленты **Конструктор** (Design) в группе **Сервис** (Tools).

Создайте форму для работы с накладными, в которой будет представлена информация об отгруженных товарах.

Очевидно, что такая форма должна строиться на основе главной таблицы НАКЛАДНАЯ с общими данными о накладных, и подчиненной таблицы

ОТГРУЗКА со сведениями об отгруженных по накладным товарах. Соответственно должны быть построены две формы.

Для создания главной формы достаточно перетащить поля из списка таблицы НАКЛАДНАЯ в нужное место области данных пустой формы. Как только вы перетащили первое поле в форму, в области списка полей появляется новый раздел **Поля, доступные в данном представлении** (Fields Available for this View), в который перемещается таблица НАКЛАДНАЯ. Построенный на этой таблице запрос на выборку, называемый представлением, будет автоматически определен в качестве базового источника данных формы. Если необходимо дополнить запись таблицы НАКЛАДНАЯ более подробными сведениями о договоре и/или покупателе, перетащите в область данных недостающие поля соответствующих таблиц. Автоматически откорректируется источник данных формы.

В результате перетаскивания поля в форме появляется элемент управления, в котором отобразится имя поля таблицы. В присоединенной надписи будет использовано значение из свойства поля **Подпись** (Caption), определенное при конструировании таблицы.

## Добавление подчиненной формы

Для добавления подчиненной формы в режиме конструктора воспользуйтесь элементом управления **Подчиненная форма/отчет** (Subform/Subreport) на вкладке ленты **Конструктор** (Design) и поместите его в нужном месте формы, растянув курсором до нужного размера. Чтобы подключился к работе мастер, в группе **Элементы управления** (Controls) должна быть включена кнопка **Использовать мастера** (Use Control Wizards). Мастер помогает включить в качестве подчиненной ранее созданную форму или построить новую на основе таблицы или запроса.

В окне мастера выберите форму ОТГРУЗКА, построенную ранее с помощью команды **Несколько элементов** (Multiple Items) (рис. 5.30).

В следующем окне мастера выберите из предложенных вариантов связи главной и подчиненной форм: НОМ\_НАКЛ (номер накладной), КОД\_СК (код склада) (рис. 5.31). Проверьте правильность установленной связи в свойствах элемента **Подчиненная форма/отчет** (Subform/Subreport) на вкладке **Данные** (Data). Для установления связи источник данных встраиваемой формы должен содержать поля связи, хотя они могут и не отображаться в форме.

В окне мастера подчиненных форм, хотя и предлагается выбор из имеющихся таблиц или запросов, отображаются только формы. При возможности переключиться в режим **Имеющиеся таблицы или запросы** (Use existing Tables and Queries) можно выбрать таблицу или запрос и поля, необходимые в подчиненной форме. Выбор таблицы или запроса приводит к созданию подчиненной формы, отображаемой в режиме таблицы. Макет элементов управления этой подчиненной формы — в столбик — приводит к установлению в свойстве **Режим по умолчанию** (Default View) значения **Одиночная форма** (Single Form).

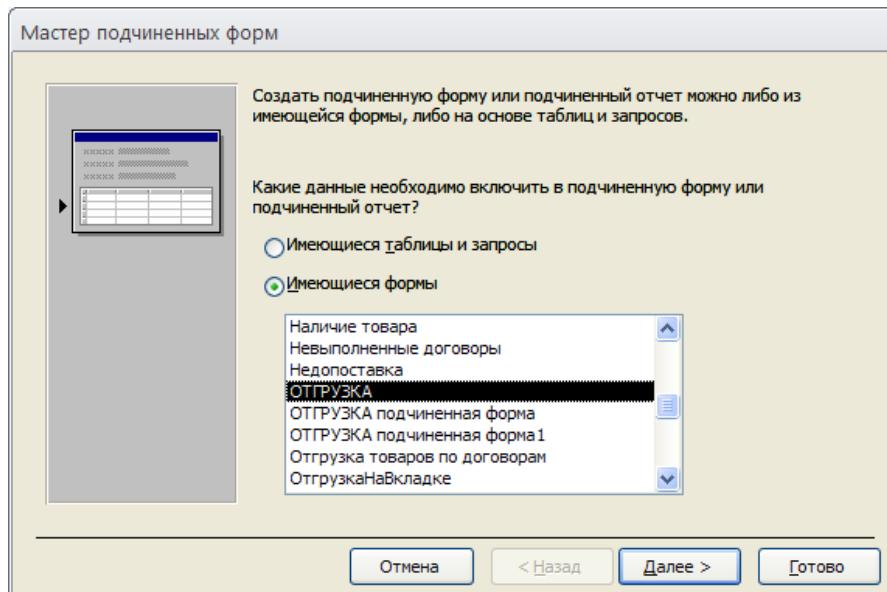


Рис. 5.30. Окно мастера для включения формы в качестве подчиненной

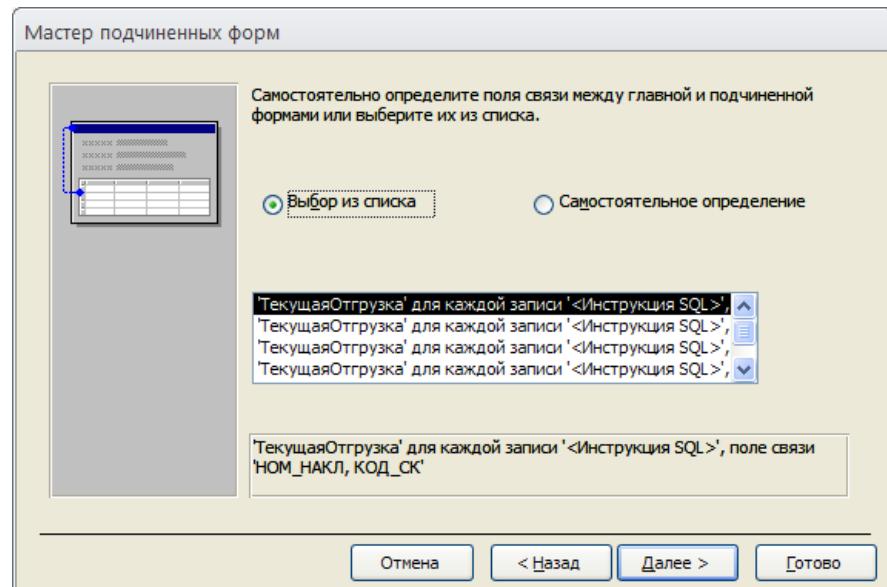


Рис. 5.31. Окно мастера для определения поля связи с подчиненной формой

При изменении значения на **Ленточные формы** (Continuous Form) отобразятся все записи, но поля каждой из них будут представлены в столбик, а подписи полей будут повторяться в каждой записи. Это связано с тем, что макеты в столбик всегда содержатся внутри одного раздела формы. Очевидно, такое размещение полей не

всегда удобно при работе с записями. Для получения многострочного отображения записей в подчиненной форме измените макет в столбик на табличный. Выделите все поля подчиненной формы (макет) и нажмите **Табличный** (Tabular) в группе **Макет элемента управления** (Control Layout) на вкладке ленты **Упорядочить** (Arrange). Поля разместятся в одну строку, а надписи переместятся в заголовок формы. Единственное, что после этого надо сделать, — сократить область данных.

Удалите из подчиненной формы поля, повторяющие значения главной формы: **НОМ\_НАКЛ**, **КОД\_СК**. В Access 2007 после удаления некоторого поля оставшиеся в форме поля автоматически сдвигаются влево, не оставляя пустых мест.

Если необходимо, измените размер элемента **Подчиненная форма/отчет** (Subform/Subreport), получивший имя **Отгрузка**.

Добавьте в форму заголовок и примечание, щелкнув на соответствующей кнопке на вкладке ленты **Упорядочить** (Arrange) в группе **Отображение** (Show/Hide). Перетащите в заголовок формы элемент управления **Надпись** (Label). Введите в элемент нужный заголовок формы. Выделите элемент и измените его шрифт, используя средства вкладки ленты **Конструктор** (Design) или **Главная** (Home) в группе **Шрифт** (Font).

Еще проще выполнить эти действия, используя элемент управления **Заголовок** (Title). Щелчком на этом элементе выполняется вставка в форму заголовка и примечания, а также вставка в заголовок элемента управления **Надпись** (Label) с текстом, соответствующим имени формы.

Если в заголовке нужно отображать номер накладной, добавьте в форму области заголовка и примечания, но не вставляйте элемент **Надпись** (Label), а просто перетащите поле **НОМ\_НАКЛ** в область заголовка и измените его надпись и формат.

Удобным средством настройки внешнего вида является группировка/разгруппировка элементов формы. Выделите, удерживая нажатой клавишу **<Shift>**, все поля главной формы Накладная, не включая их надписи. Нажмите кнопку **Группировать** (Group) в группе **Макет элемента управления** (Control Layout) на вкладке **Упорядочить** (Arrange). Переместите группу вправо, так чтобы были полностью видны надписи полей.

Для улучшения дизайна формы можно, используя свойства формы, на вкладке **Макет** (Format) убрать область выделения записи (слева), полосу прокрутки, кнопки перехода по записям, разделительные линии между областью заголовка, данных и примечаний и т. д.

Форма в режиме конструктора и просмотра, полученная после включения подчиненной формы мастером, приведена на рис. 5.32 и 5.33.

Если не использовать мастера подчиненных форм, то, вставив элемент управления **Подчиненная форма/отчет** (Subform/Subreport), получим в форме **Свободный** (Unbound) элемент подчиненной формы с именем **ВнедренныйN** (ChildN).

Для определения источника данных подчиненной формы следует в окне свойств **Подчиненная форма/отчет** (Subform/Subreport) перейти на вкладку **Данные** (Data). В строке **Объект-источник** (Source Object) выбрать из списка имен форму, которая будет подчиненной. В данном примере — форму **ОТГРУЗКА**. После этого в рамке подчиненной формы отобразится выбранная форма в режиме конструктора.

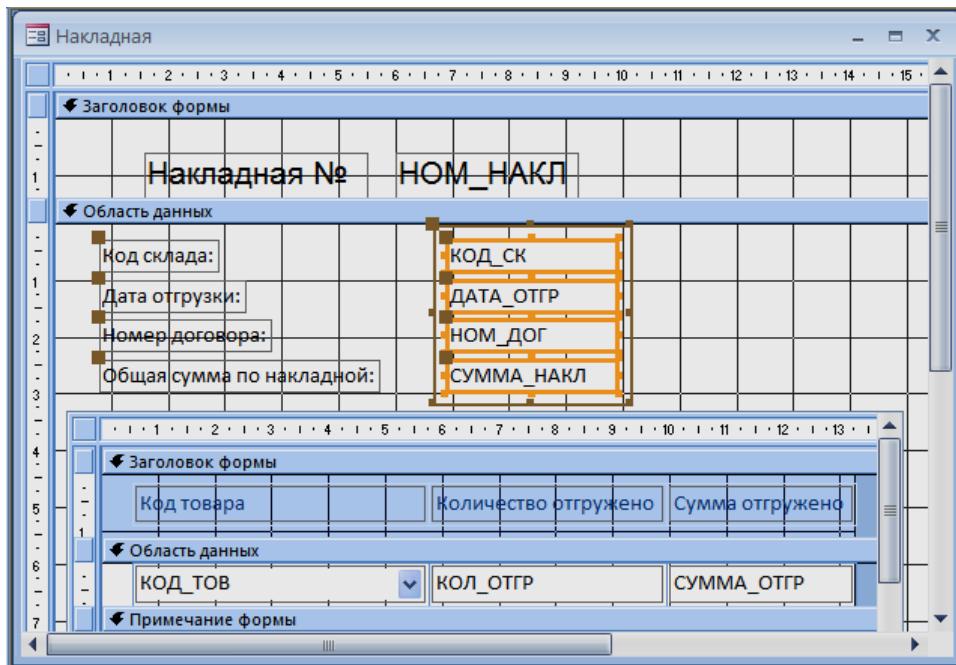


Рис. 5.32. Форма в режиме конструктора  
после включения подчиненной формы мастером

The screenshot shows the Microsoft Access form in 'View' mode. The main form header reads 'Накладная № Н001'. The data entry fields are filled with values: 'Код склада:' (C01), 'Дата отгрузки:' (09.03.2007), 'Номер договора:' (Д111), and 'Общая сумма по накладной:' (56 060,00р.). The subform 'Код Товара' displays the same data as in the designer:

Код товара	Количество отгружено	Сумма отгружено
T005	20	23 060,00р.
T007	10	23 000,00р.
T001	10	10 000,00р.
*	0	0,00р.

At the bottom, there are navigation buttons for records and search fields.

Рис. 5.33. Форма в режиме просмотра

Если ранее в схеме данных была определена связь таблиц, соответствующих формам, связь между основной и подчиненной формой устанавливается автоматически (достаточно щелкнуть на значке в конце строки свойства **Подчиненные поля** (Link Child Fields) или **Основные поля** (Link Master Fields)). Если связи между таблицами не определены, то в окне свойств в строках **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields) следует ввести имена полей, по которым связываются таблицы. В рассмотренном примере связь реализуется полями НОМ\_НАКЛ (номер накладной) и КОД\_СК (код склада) (рис. 5.34). Имена полей в строке свойств разделяются точкой с запятой. Аналогичным образом можно вместо формы вставить таблицу ОТГРУЗКА.

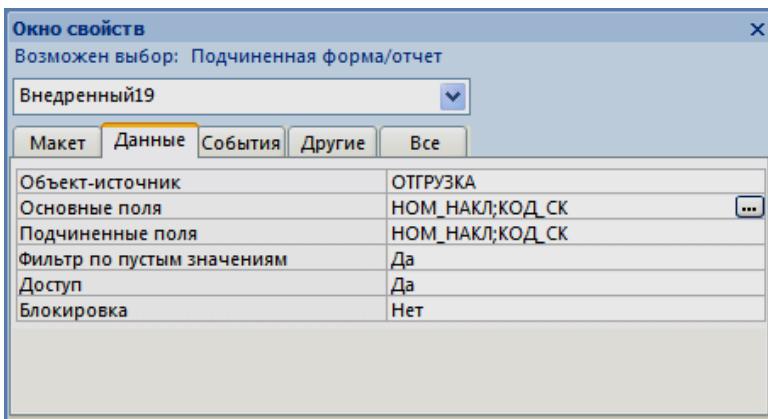


Рис. 5.34. Окно свойств подчиненной формы

Самый простой способ — создание подчиненной формы путем перетаскивания одной формы в другую. Откройте в режиме конструктора форму, которая должна быть главной. Перетащите форму или таблицу, на основе которой нужно создать подчиненную форму, из области переходов в главную форму.

## Вычисление итогового значения в подчиненной форме и вывод его в текущей записи основной формы

Вычисление на основе данных каждой записи формы, а также вычисление итоговых значений в форме с помощью встроенных статистических функций было рассмотрено в предыдущих разделах главы. Теперь рассмотрим, как итоговые значения, вычисленные в подчиненной форме, отобразить или использовать в дальнейших расчетах в основной форме.

Если главная форма и ее подчиненная форма построены на основе таблиц, между которыми установлена связь типа с  $1 : M$ , в основной форме выводится одна текущая запись, а в подчиненной форме — связанные с ней записи.

Пусть необходимо подсчитать, на какую сумму заключил договоры каждый из покупателей, и отобразить ее в форме. Откройте созданную ранее форму

ПОКУПАТЕЛЬ (см. рис. 5.25), включающую одну подчиненную форму ДОГОВОР, в режиме конструктора. Рассчитайте общую сумму по договорам одного покупателя в подчиненной форме. Для этого в примечании подчиненной формы ДОГОВОР создайте вычисляемый элемент управления и запишите в него выражение (рис. 5.35):  
 $=\text{Sum}([\text{СУММА\_ДОГ}])$

**ПОКУПАТЕЛЬ**

**Область данных**

Код покупателя	КОД_ПОК
ИНН	ИНН
Покупатель	НАИМ_ПОК
Адрес	АДРЕС_ПОК
ТЕЛ	ТЕЛ
НОМ_РСЧ	НОМ_РСЧ
БАНК	БАНК

**ДОГОВОР подчиненная форма**

**Заголовок формы**

Номер договора	Дата заключения	Сумма по договору
----------------	-----------------	-------------------

**Область данных**

НОМ_ДОГ	ДАТА_ДОГ	СУММА_ДОГ
---------	----------	-----------

**Примечание формы**

Поле9	$=\text{Sum}([\text{СУММА\_ДОГ}])$
-------	------------------------------------

**Общая стоимость договоров покупателя:**  $=[\text{ДОГОВОР подчиненная форма}.[\text{Form}]![\text{Поле9}]]$

**Рис. 5.35.** Вычисление итогового значения в подчиненной форме  
и отображение его в главной форме

Для отображения результата вычисления в главной форме создайте там вычисляемый элемент управления и запишите в него выражение (см. рис. 5.35):  
 $=[\text{ДОГОВОР подчиненная форма}.[\text{Form}]![\text{Поле9}]]$

Это выражение является ссылкой на элемент управления Поле9 в примечании подчиненной формы, содержащий общую стоимость договоров. Общий формат такой ссылки имеет вид:

$=[\text{Имя подчиненной формы}.[\text{Form}]![\text{Имя поля в подчиненной форме}]]$

В надпись вычисляемого элемента управления введите: Общая стоимость договоров покупателя. В свойстве **Формат поля** (Format) на вкладке **Макет** (Format) измените значение на **Денежный** (Currency).

Чтобы не отображалось примечание подчиненной формы, установите для его свойства **Вывод на экран** (Display When) значение **Нет** (No).

Форма с вычисляемым элементом управления, отображающим в основной форме общую стоимость договоров покупателя, рассчитанную в подчиненной форме, представлена на рис. 5.36.

**Рис. 5.36.** Отображение итогового значения в главной форме, вычисленного в подчиненной форме

## Ограничения доступа к данным через форму

### Защита данных поля от изменений

Для защиты данных поля от изменения используется свойство **Блокировка** (Locked). Блокировка может быть установлена для любого поля формы. Чтобы защитить поле, надо открыть форму в режиме макета или конструктора, установить курсор в его рамке и с помощью контекстно-зависимого меню вызвать свойства поля. В окне свойств на вкладке **Данные** (Data) в строке **Блокировка** (Locked) выбрать **Да** (Yes). По умолчанию для всех полей в свойстве **Блокировка** (Locked) устанавливается значение **Нет** (No). После установки этого свойства поле доступно только для чтения.

### Установка ограничений на корректировку записей через форму

Для того чтобы при работе через форму записи были доступны только для чтения, следует в свойствах формы на вкладке **Данные** (Data) (рис. 5.37) в строках

**Разрешить добавление** (Allow Additions), **Разрешить удаление** (Allow Deletions) и **Разрешить изменение** (Allow Edits) задать значения **Нет** (No). Сделать записи доступными только для чтения можно также, выбрав для свойства **Тип набора записей** (Recordset Type) значение **Статический набор** (Snapshot). Указанные свойства могут устанавливаться независимо друг от друга. Например, при запрете на изменение записей может быть разрешено добавление и удаление записей.

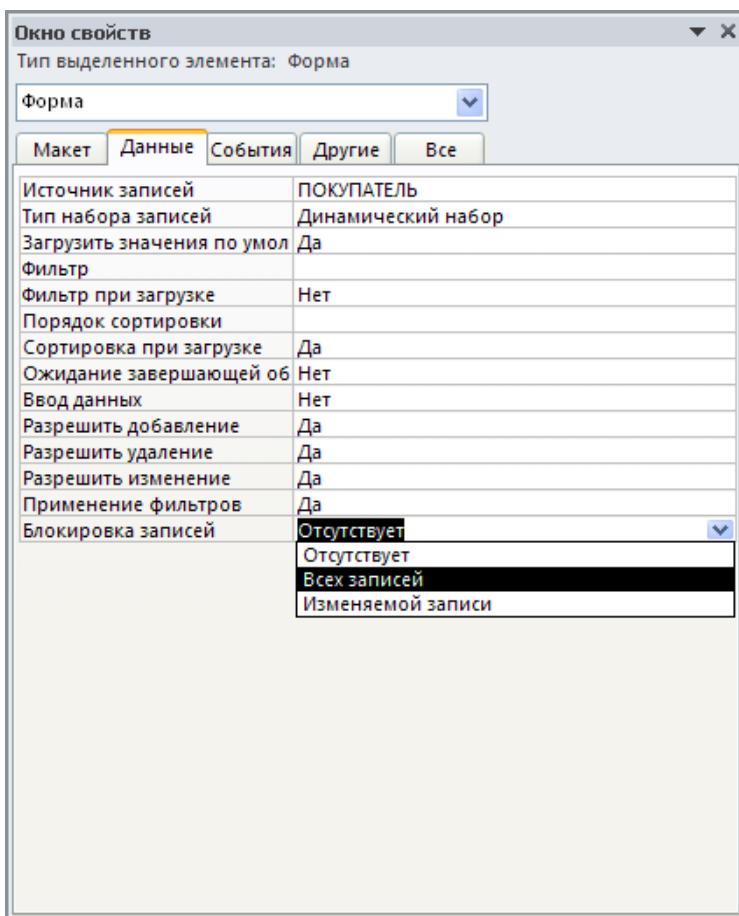


Рис. 5.37. Установка ограничений на корректировку записей

### ЗАМЕЧАНИЕ

Свойство формы **Блокировка записей** (Record Locks) определяет способы блокировки записей при попытке двух пользователей одновременно изменить записи в некотором наборе записей сетевой базы данных. Когда один пользователь изменяет запись, автоматически может блокироваться только изменяемая запись или все записи набора. При этом другие пользователи не могут изменять или только одну запись, или весь набор до завершения работы с ней первого пользователя.

Свойство **Ввод данных** (Data Entry) при значении **Да** (Yes) определяет режим формы, при котором разрешен только ввод новых записей, просмотр существующих записей при этом не доступен. При открытии формы выводится пустая запись, с которой можно начать ввод новых записей. С вновь введенными записями можно выполнять все разрешенные действия: удалять, изменять их. При новом открытии формы снова делается доступным только режим ввода новых записей.

## **Защита данных подчиненной формы от изменений**

Зашитить содержимое всех полей подчиненной формы, сделав его доступным только для чтения, позволяет свойство **Блокировка** (Locked), установленное для подчиненной формы.

Чтобы защитить данные подчиненной формы от изменений надо выделить соответствующий элемент, открыть его свойства. В окне свойств выбрать вкладку **Данные** (Data). В строке **Блокировка** (Locked) выбрать **Да** (Yes). При этом станет невозможным изменение, добавление и удаление данных в полях подчиненной формы. Кроме того, станет невозможным добавление новых записей.

Установка для свойства **Доступ** (Enabled) значения **Нет** (No) приводит не только к запрету на любые изменения, но и невозможности перемещаться по записям, просматривать записи за пределами рамки подчиненной формы.

## **Разработка интерфейса для ввода, просмотра и корректировки документов**

Основным средством создания интерфейса для работы с документами являются формы. С помощью средств разработки форм могут быть созданы электронные аналоги документов, обеспечивающие ввод, просмотр и корректировку документов в базе данных. Формы могут использоваться и для оформления документов на компьютере, например накладных на отгрузку товара, счетов на оплату в соответствии с договорами и т. д. В процессе оформления документа может выполняться и ввод его в базу данных. Кроме того, как правило, предусматривается распечатка подготовленного документа средствами отчетов.

Инструментарий разработки форм предоставляет широкие возможности по созданию графического диалогового интерфейса пользователя для работы с документами, сохраняемыми в базе данных. Такой интерфейс является основой работы с базой данных практического приложения пользователя. После окончательного создания приложения пользователь, как правило, не работает непосредственно с таблицами базы данных. Разработчик приложения часто ограничивает полностью или частично непосредственный доступ пользователя к таблицам.

При конструировании интерфейса пользователя могут использоваться макросы или процедуры VBA для обработки событий, инициируемых пользователем в про-

цессе работы с формой. Для формы и ее элементов управления определен набор типовых событий. Такие связанные с полем события как **Изменение** (On Change), **До обновления** (Before Update), **После обновления** (After Update) или связанные с записью события **Удаление** (Delete), **До обновления** (Before Update), **После обновления** (After Update) часто используются для подключения процедур, обеспечивающих автоматический перерасчет (изменение) показателей в таблицах.

При создании некоторых элементов формы автоматически формируются типовые процедуры обработки событий. Таким элементом является, например, кнопка. С ее событием **Нажатие кнопки** (On Click) связывается выполнение таких категорий действий как:

- ◆ переходы по записям источника формы, обработка записей (добавление, удаление, печать, восстановление);
- ◆ работа с формой (закрытие, открытие других форм, изменение фильтра, обновление данных, печать формы);
- ◆ работа с отчетом (печать, просмотр, отправка, вывод в файл);
- ◆ запуск запроса, макроса, печать таблицы, автонабор номера.

## Этапы разработки интерфейса

Технология создания целостной базы, в которой между таблицами установлены связи и определены параметры поддержания целостности, предполагает упорядочение первоначальной загрузки взаимосвязанных таблиц.

Технология поддержания такой базы данных в актуальном состоянии требует обеспечения процесса ввода оперативных данных и обновления существующих данных.

При этом должен быть разработан удобный интерфейс пользователя, обеспечивающий важнейший аспект технологии работы с базой данных — однократный и корректный ввод взаимосвязанных данных. Использование экранных форм — электронных аналогов первичных документов, являющихся источниками для загрузки справочных, плановых и оперативных учетных данных — позволяет решить эти задачи.

Прежде чем вводить, отображать или корректировать данные таблиц через экранную форму, надо спроектировать и сконструировать ее. Далее рассматриваются основы проектирования форм для построения удобного интерфейса пользователя для работы с документами. Подробно описана технология разработки формы, обеспечивающей первоначальный ввод, просмотр и обновление документов в базе данных.

В процессе разработки технологии загрузки базы данных и проектирования форм целесообразно определить:

- ◆ перечень документов-источников, содержащих необходимые данные для загрузки таблиц базы данных;
- ◆ таблицы — объекты загрузки для каждого документа-источника;

- ❖ содержание и последовательность загрузки; при этом необходимо учитывать, что для обеспечения связной целостности главные таблицы должны быть загружены ранее подчиненных;
- ❖ подсхему данных каждой формы (фрагмент схемы данных), состоящую из таблиц, необходимых для создания электронного документа. При этом для многостабильной (составной) формы выбирается:
  - ◆ таблица, которая будет базовым источником записей главной формы, и таблицы для отображения справочных данных в этой части формы;
  - ◆ таблица, которая будет источником записей подчиненной формы, включаемой в главную форму, и таблицы для отображения справочных данных в подчиненной форме;
- ❖ макет формы, т. е. ее общую структуру, в соответствии со структурой документа-источника и полученной подсхемой данных. При этом распределяется пространство формы для размещения включаемых подчиненных форм;
- ❖ состав и размещение элементов, связанных с полями таблиц, и надписей для каждой из частей составной формы. При этом:
  - ◆ в главную форму обязательно надо вводить ключевые поля таблицы-источника данных (например, идентификатор документа "Договор" — номер договора);
  - ◆ в подчиненной форме предусмотреть только те ключевые поля таблицы — базового источника подчиненной формы, которых нет в таблице-источнике главной формы (например, код товара из спецификации документа "Договор").

После выполнения перечисленных пунктов и получения макета формы можно приступить к разработке форм средствами Access.

## **Определение последовательности загрузки таблиц с документов**

При разработке форм, обеспечивающих загрузку взаимосвязанных таблиц базы данных, следует иметь в виду требования к последовательности загрузки записей в таблицы в соответствии со схемой данных и установленными параметрами поддержания целостности. Эти требования можно сформулировать следующим образом:

- независимо могут создаваться записи таблиц, которые не подчинены каким-либо другим таблицам в схеме данных;
- запись таблицы, подчиненной каким-либо другим таблицам, может создаваться при наличии связанных с ней записей в главных таблицах; записи главной таблицы должны быть загружены ранее (таблицы справочных данных) или должны создаваться вместе с подчиненной записью в одной форме.

В соответствии с этими требованиями можно рекомендовать в практических приложениях предусмотреть сначала ввод в базу данных справочных данных, а затем данных плановых и оперативно-учетных документов. Это связано с тем, что таблицы с плановыми и оперативно-учетными данными в схеме данных являются

подчиненными по отношению к таблицам справочных данных, которые, как правило, находятся на верхнем уровне.

Рассмотрим технологию загрузки на примере базы данных *Поставка товара*. Таблицы базы данных и связи между ними отображены в схеме данных, приведенной в главе 2 на рис. 2.19.

Документы-источники загрузки базы данных *Поставка товара* названы при описании предметной области в главе 2. Определим объекты загрузки базы данных — взаимосвязанные таблицы, подлежащие загрузке с каждого документа предметной области, и последовательность их загрузки.

## Справочная информация

Для документов справочной информации в базе данных *Поставка товаров* следует выделить следующие объекты загрузки:

- ◆ таблица ТОВАР; загрузка этой таблицы производится из документа "Справочник товаров", содержащего сведения о товарах, поставляемых фирмой;
- ◆ таблица СКЛАД; загрузка этой таблицы производится из документа "Справочник складов", содержащего сведения о складах фирмы;
- ◆ таблица ПОКУПАТЕЛЬ; загрузка этой таблицы производится из документа "Справочник покупателей", содержащего сведения о покупателях фирмы.

Таблицы справочной информации ПОКУПАТЕЛЬ, ТОВАР, СКЛАД на схеме данных находятся на верхнем уровне и не подчинены другим таблицам, поэтому их загрузка производится в любой последовательности.

## Плановая информация

Из документа "Договор", содержащего условно постоянную плановую информацию, целесообразно единовременно вводить не только общие сведения о договоре, но и данные о плановых поставках по договору. В соответствии с этим следует выделить единый объект загрузки: таблицы ДОГОВОР — ПОСТАВКА\_ПЛАН. Загрузка записей этих таблиц производится одновременно из документа "Договор", что обеспечит формирование взаимосвязей записей этих таблиц. При этом обеспечивается однократный ввод значений идентификатора договора НОМ\_ДОГ для всех товаров документа.

Загрузка таблицы ДОГОВОР может производиться после загрузки таблицы ПОКУПАТЕЛЬ, т. к. таблица ДОГОВОР в схеме данных подчинена таблице ПОКУПАТЕЛЬ.

Загрузка таблицы ПОСТАВКА\_ПЛАН может производиться только после загрузки таблиц ДОГОВОР и ТОВАР, т. к. таблица ПОСТАВКА\_ПЛАН подчинена этим таблицам.

## Оперативно-учетная информация

Из документа "Накладная", как и в предыдущем случае, целесообразно единовременно вводить общие сведения о накладной и данные об отгрузках товара по

накладной. В соответствии с этим следует выделить единый объект загрузки: таблицы НАКЛАДНАЯ — ОТГРУЗКА. Загрузка записей этих таблиц производится одновременно из документа "Накладная", что обеспечит формирование взаимосвязей записей этих таблиц. При этом осуществляется однократный ввод значений идентификатора накладной — НОМ\_НАКЛ и КОД\_СК для всех отгружаемых по накладной товаров.

Загрузка таблицы НАКЛАДНАЯ может производиться только после загрузки таблиц ДОГОВОР и СКЛАД, т. к. таблица НАКЛАДНАЯ в схеме данных подчинена этим таблицам.

Загрузка таблицы ОТГРУЗКА может производиться только после загрузки таблиц НАКЛАДНАЯ и ТОВАР, т. к. таблица ОТГРУЗКА подчинена этим таблицам.

### **ЗАМЕЧАНИЕ**

Загрузка таблицы СКЛАД может быть осуществлена и после загрузки данных по договорам, поскольку не по каким путям в схеме данных таблицы ДОГОВОР и ПОСТАВКА\_ПЛАН не подчинены таблице СКЛАД.

Таким образом, в результате определена последовательность этапов загрузки базы данных *Поставка товаров*, а также объекты загрузки на отдельных этапах и соответствующие документы-источники данных. Технология загрузки базы данных *Поставка товаров* обобщена в табл. 5.1.

**Таблица 5.1. Технология загрузки базы данных Поставка товаров**

Таблицы БД — объекты загрузки	Документы-источники	Вид информации	Этап загрузки	Примечание
ПОКУПАТЕЛЬ	Справочник покупателей	Справочная	I	Независимые этапы
ТОВАР	Справочник товаров	Справочная	I	
СКЛАД	Справочник складов	Справочная	I или II	
ДОГОВОР-ПОСТАВКА_ПЛАН	Договоры	Плановая	II	
НАКЛАДНАЯ — ОТГРУЗКА	Накладные	Оперативно-учетная	III	

После определения этапов загрузки базы данных можно приступить к определению подсхемы данных для каждого этапа загрузки, к проектированию макета форм и их созданию средствами Access.

## **Проектирование интерфейса для ввода и корректировки документа**

Ввод и корректировка справочных данных может быть осуществлена через простые формы с макетом в столбец или табличный, в которых для проверки значений в

полях заданы ограничения. Для ввода и корректировки данных плановых и оперативно-учетных документов пользователю нужно разработать удобный экранный интерфейс, который позволит минимизировать операции по вводу данных и контролировать их достоверность и корректность. При этом необходимо ограничиваться вводом только идентификаторов и количественных показателей. Справочные данные (наименования, нормативы, цены, тарифные ставки и т. п.) не должны вводиться с этих документов, а должны только отображаться в форме из ранее созданных таблиц справочной информации. Отображение справочных данных позволяет осуществлять визуальный контроль правильности вводимых с плановых или оперативно-учетных документов данных, в которых обычно присутствуют справочные данные.

Разработка интерфейса требует для каждого документа выполнить проектирование формы.

Рассмотрим процесс проектирования формы для ввода, просмотра и корректировки данных о договорах фирмы. Форма служит электронным документом, вид которого должен соответствовать виду бумажного документа. Вид документа "Договор" был приведен в главе 2 на рис. 2.7.

В соответствии с этапами загрузки базы данных *Поставка товаров*, определенными выше (см. табл. 5.1), загрузка данных из документа "Договор" должна производиться в таблицы ДОГОВОР и ПОСТАВКА\_ПЛАН после загрузки таблиц со справочными данными ПОКУПАТЕЛЬ и ТОВАР, что обеспечит установление связей загружаемых записей с соответствующими записями этих таблиц.

При проектировании формы определяется подсхема данных, включающая объект загрузки формы, общая структура формы — проект макета и размещение реквизитов в соответствии со структурой документа "Договор" и подсхемой данных, учитываются особенности назначения и работы с формой.

## Определение подсхемы данных

Выбор подсхемы данных для построения формы аналога документа "Договор", назовем ее ДОГОВОРЫ С ПОКУПАТЕЛЯМИ, определяется следующими соображениями.

- ❖ Загрузка данных по договорам должна производиться в таблицы ДОГОВОР и ПОСТАВКА\_ПЛАН, находящиеся в отношении 1 :  $M$ , следовательно, эти таблицы — объекты загрузки надо включить в подсхему данных формы.
- ❖ В форме должны отображаться справочные данные по покупателям и товарам, указанным в договоре, поэтому в подсхему надо включить также таблицы ПОКУПАТЕЛЬ и ТОВАР, главные по отношению к таблицам ДОГОВОР и ПОСТАВКА\_ПЛАН.
- ❖ Так как форма обеспечивает загрузку двух таблиц, связанных отношением 1 :  $M$ , главная в отношении таблица ДОГОВОР должна быть источником записей основной формы, подчиненная ПОСТАВКА\_ПЛАН — источником записей подчиненной формы. Для отображения справочных данных в основной форме должна использоваться таблица ПОКУПАТЕЛЬ. Для отображения справочных данных в подчиненной форме должна использоваться таблица ТОВАР.

Таким образом, подсхема данных для формы ввода/вывода договоров фирмы должна иметь вид, показанный на рис. 5.38.

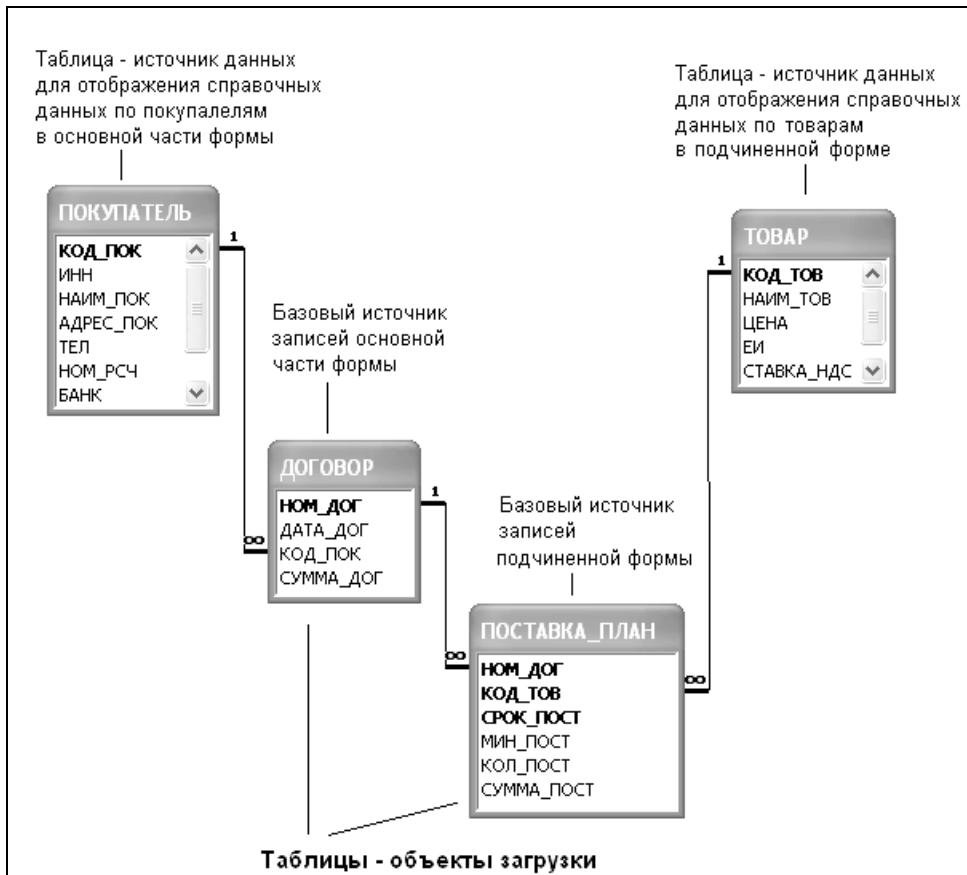


Рис. 5.38. Подсхема данных для формы ввода/вывода договоров фирмы

### **ВНИМАНИЕ!**

В процессе проектирования базы данных (см. главу 2) все реквизиты документа были разбиты на подмножества, составляющие таблицы базы данных. Например, реквизиты документа "Договор" были распределены по таблицам ДОГОВОР, ПОСТАВКА\_ПЛАН, ПОКУПАТЕЛЬ и ТОВАР. Очевидно, для того чтобы форма ДОГОВОР отображала полный документ, ее подсхема данных должна включать все эти таблицы.

### **Разработка макета**

Макет формы разрабатывается в соответствии со структурой документа и полученной подсхемой данных. Макет формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ приведен на рис. 5.39.

		ДОГОВОР № <input type="text" value="НОМ_ДОГ (ДОГОВОР)"/>																													
		от <input type="text" value="ДАТА_ДОГ (ДОГОВОР)"/>																													
<b><u>Покупатель</u></b>																															
код	<input type="text" value="КОД_ПОК (ДОГОВОР)"/>	адрес	<input type="text" value="АДРЕС_ПОК (ПОКУПАТЕЛЬ)"/>																												
наименование	<input type="text" value="НАИМ_ПОК(ПОКУПАТЕЛЬ)"/>	телефон	<input type="text" value="ТЕЛ (ПОКУПАТЕЛЬ)"/>																												
ИНН	<input type="text" value="ИНН(ПОКУПАТЕЛЬ)"/>	расчетный счет	<input type="text" value="НОМ_РСЧ (ПОКУПАТЕЛЬ)"/>																												
		банк	<input type="text" value="БАНК (ПОКУПАТЕЛЬ)"/>																												
<table border="1"> <thead> <tr> <th>Код това-ра</th> <th>Наиме-нова-ние</th> <th>Ед изм</th> <th>Цена</th> <th>Срок поставки (месяц)</th> <th>Мин. пар-тия по-ставки</th> <th>Количе-ство</th> <th>Сумма</th> </tr> </thead> <tbody> <tr> <td><input type="text" value="КОД_ТОВ"/></td> <td><input type="text" value="НАИМ_ТОВ"/></td> <td><input type="text" value="ЕИ"/></td> <td><input type="text" value="ЦЕНА"/></td> <td><input type="text" value="СРОК_ПОСТ"/></td> <td><input type="text" value="МИН_ПОСТ"/></td> <td><input type="text" value="КОЛ_ПОСТ"/></td> <td><input type="text" value="СУММА_ПОСТ"/></td> </tr> <tr> <td colspan="2">ПОСТАВКА_ПЛАН</td> <td colspan="2">ТОВАР</td> <td colspan="4">ПОСТАВКА_ПЛАН</td> </tr> </tbody> </table>								Код това-ра	Наиме-нова-ние	Ед изм	Цена	Срок поставки (месяц)	Мин. пар-тия по-ставки	Количе-ство	Сумма	<input type="text" value="КОД_ТОВ"/>	<input type="text" value="НАИМ_ТОВ"/>	<input type="text" value="ЕИ"/>	<input type="text" value="ЦЕНА"/>	<input type="text" value="СРОК_ПОСТ"/>	<input type="text" value="МИН_ПОСТ"/>	<input type="text" value="КОЛ_ПОСТ"/>	<input type="text" value="СУММА_ПОСТ"/>	ПОСТАВКА_ПЛАН		ТОВАР		ПОСТАВКА_ПЛАН			
Код това-ра	Наиме-нова-ние	Ед изм	Цена	Срок поставки (месяц)	Мин. пар-тия по-ставки	Количе-ство	Сумма																								
<input type="text" value="КОД_ТОВ"/>	<input type="text" value="НАИМ_ТОВ"/>	<input type="text" value="ЕИ"/>	<input type="text" value="ЦЕНА"/>	<input type="text" value="СРОК_ПОСТ"/>	<input type="text" value="МИН_ПОСТ"/>	<input type="text" value="КОЛ_ПОСТ"/>	<input type="text" value="СУММА_ПОСТ"/>																								
ПОСТАВКА_ПЛАН		ТОВАР		ПОСТАВКА_ПЛАН																											
				Сумма всего	<input type="text" value="СУММА_ДОГ ( ДОГОВОР)"/>																										

**Рис. 5.39.** Проект макета формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ

В соответствии с определенными объектами загрузки многотабличная форма ДОГОВОРЫ С ПОКУПАТЕЛЯМИ должна состоять из двух форм основной и включенной в нее подчиненной формы.

Источником записей главной формы будет ДОГОВОР, а таблица ПОКУПАТЕЛЬ будет использована для отображения справочной информации. Через эту часть многотабличной формы выполняется ввод, просмотр и корректировка общих сведений о договоре. Число доступных записей определяется количеством записей в таблице ДОГОВОР.

Источником записей подчиненной формы будет таблица ПОСТАВКА\_ПЛАН, а таблица ТОВАР будет использована для отображения справочной информации. Через эту часть многотабличной формы выполняется ввод, просмотр и корректировка данных о плановых поставках по договорам. Число доступных записей определяется количеством записей в таблице ПОСТАВКА\_ПЛАН.

В макете, как для основной, так и для подчиненной формы, определяется состав и размещение элементов документа, а также их связь с полями таблиц подсхемы данных:

- ❖ в основе обязательно должны размещаться: ключевое поле НОМ\_ДОГ и поле код покупателя КОД\_ПОК, по которому устанавливается связь с таблицей ПОКУПАТЕЛЬ. Эти поля должны быть взяты из таблицы ДОГОВОР — источника записей основной части формы. В противном случае нельзя будет ввести новую запись в таблицу ДОГОВОР;

- ❖ в подчиненной форме должно обеспечиваться формирование ключа таблицы ПОСТАВКА\_ПЛАН — (НОМ\_ДОГ, КОД\_ТОВ, СРОК\_ПОСТ). В противном случае станет невозможным ввод новых записей. Однако можно разместить только те ключевые поля таблицы, которых нет в основной форме, т. е. обязательно нужно разместить только поля КОД\_ТОВ (код товара) и СРОК\_ПОСТ (срок поставки) из таблицы ПОСТАВКА\_ПЛАН, а поле НОМ\_ДОГ (номер договора) можно не размещать.

### **ВНИМАНИЕ!**

Если поле КОД\_ТОВ выбрать из таблицы ТОВАР, то невозможно будет сформировать новую запись в таблице ПОСТАВКА\_ПЛАН.

Многотабличная форма, соответствующая этому макету, обеспечит удобный интерфейс для ввода, просмотра и корректировки данных о договорах, позволит минимизировать операции по вводу данных и контролировать их достоверность и корректность. Вводить нужно только идентификаторы и количественные показатели. Справочные данные (наименования, цена и т. п.) не потребуется вводить из документа, они отобразятся в форме из ранее загруженных таблиц справочной информации. Отображение справочных данных позволяет осуществлять визуальный контроль правильности вводимых из документа данных.

После разработки макета можно приступить к созданию многотабличной формы средствами Access.

## **Создание интерфейса для ввода и корректировки документа**

С помощью мастера форм создайте в соответствии с результатами проектирования форму ДОГОВОРЫ С ПОКУПАТЕЛЯМИ, состоящую из основной формы ДОГОВОР и включенной подчиненной формы ПЛАН ПОСТАВОК. Многотабличная форма после редактирования в конструкторе должна обеспечивать удобный ввод новых записей из документа "Договор" в таблицы ДОГОВОР и ПОСТАВКА\_ПЛАН, просмотр и корректировку существующих договоров. Доступ к отображаемым в форме полям справочных данных о покупателе и товарах должен ограничиваться только чтением, т. к. значения этих полей не должны вводиться и корректироваться при вводе договоров.

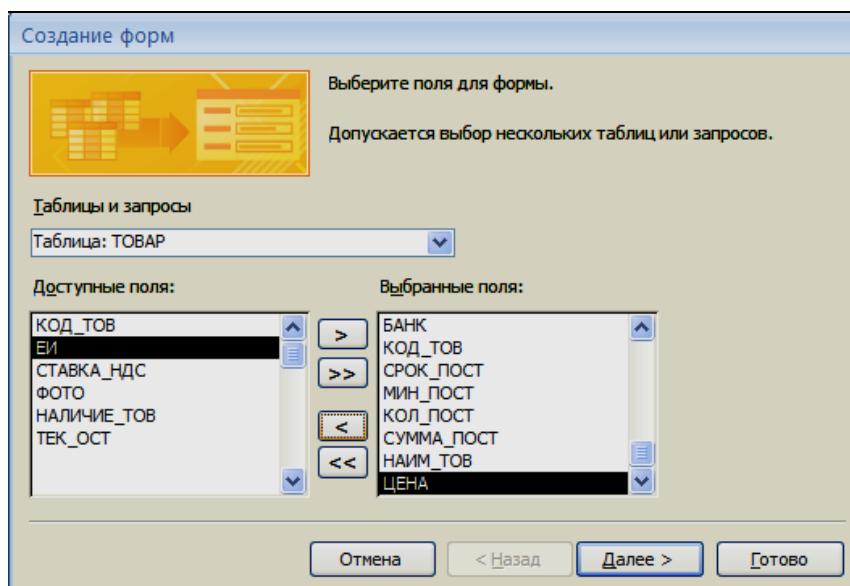
Откройте базу данных *Поставка товаров*. В области переходов выберите таблицу ДОГОВОР, которая будет служить источником записей основной формы. Для вызова мастера форм выполните на вкладке ленты **Создание** (Create) в группе **Формы** (Forms) команду **Мастер форм** (Form Wizard).

В открывшемся окне мастера **Создание форм** (Form Wizard) в списке **Таблицы и запросы** (Tables/Queries) будет отображена выбранная таблица ДОГОВОР и ее поля. Выберите из таблицы ДОГОВОР в списке **Доступные поля** (Available Fields) те поля, которые вошли в спроектированный макет формы, перемещая их в область **Выбранные поля** (Selected Fields).

Последовательно выбирайте из списка **Таблицы и запросы** (Tables/Queries) таблицы ПОКУПАТЕЛЬ, ПОСТАВКА\_ПЛАН, ТОВАР и включайте в область **Выбранные поля** (Selected Fields) нужные поля этих таблиц (рис. 5.40). Нажмите кнопку **Далее** (Next).

Для формирования основной формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ включите:

- ❖ загружаемые поля таблицы ДОГОВОР:
  - ◆ НОМ\_ДОГ — номер договора (уникальный ключ);
  - ◆ ДАТА\_ДОГ — дата заключения договора;
  - ◆ КОД\_ПОК — код покупателя (поле связи);
  - ◆ СУММА\_ДОГ — сумма всего по договору;
- ❖ поля из таблицы ПОКУПАТЕЛЬ, позволяющие отобразить справочную информацию о покупателе, с которым заключен договор:
  - ◆ НАИМ\_ПОК — наименование покупателя;
  - ◆ ИНН;
  - ◆ АДРЕС и другие справочные реквизиты покупателя.
- ❖ Для формирования подчиненной формы включите поля:
- ❖ загружаемые поля таблицы ПОСТАВКА\_ПЛАН:
  - ◆ КОД\_ТОВ — код товара (часть уникального ключа);
  - ◆ СРОК\_ПОСТ — срок поставки (часть уникального ключа);
  - ◆ МИН\_ПОСТ — минимальная партия поставки;
  - ◆ КОЛ\_ПОСТ — плановое количество поставки товара;
  - ◆ СУММ\_ПОСТ — сумма поставки товара;



**Рис. 5.40.** Окно мастера форм при выборе полей из таблиц, составляющих подсхему формы

- ❖ поля из таблицы ТОВАР, позволяющие отобразить справочную информацию о товарах, поставляемых по договору:
- ◆ НАИМ\_ТОВ — наименование товара;
  - ◆ ЕИ — единица измерения;
  - ◆ ЦЕНА — цена товара.

В следующем диалоговом окне мастера (рис. 5.41) уже выделена таблица ДОГОВОР, предлагаемая в качестве базового источника для создания основной формы, и схематично отображен макет формы с перечнем полей в основной и подчиненной форме. Кроме того, мастером отмечен переключатель **Подчиненные формы** (Form with subform(s)) — вариант непосредственного включения подчиненной формы в основную. Переключатель **Связанные формы** (Linked forms) определяет вызов подчиненной формы по кнопке.

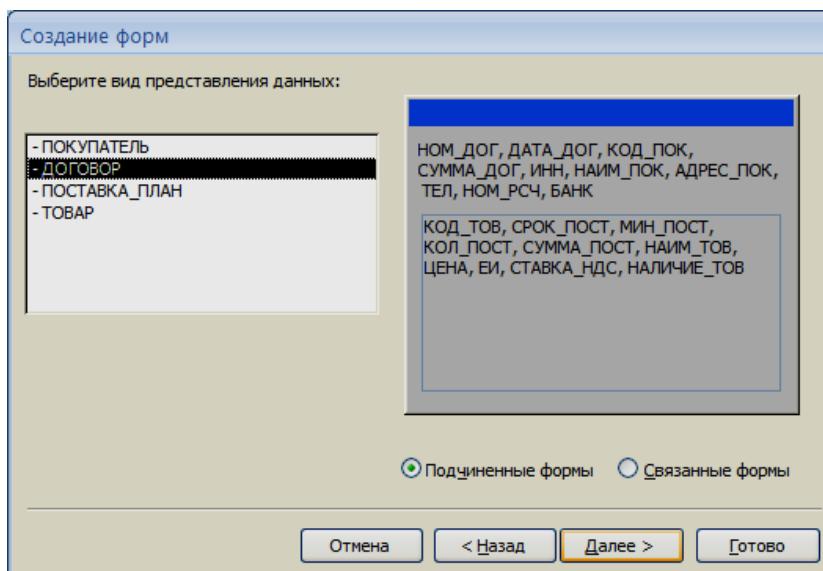


Рис. 5.41. Окно мастера форм при выборе вида представления данных

Для получения многозаписевой подчиненной формы с отображением полей каждой записи в одной строке, а надписей (определенных в свойствах таблиц) в заголовках столбцов выберите в следующем окне мастера вид формы **Ленточный** (Tabular). Далее выберите стиль оформления, например **Стандартная** (Standard).

В последнем окне мастера задайте имя главной формы — ДОГОВОРЫ С ПОКУПАТЕЛЯМИ, а также имя подчиненной формы — ПЛАН ПОСТАВОК. Выберите дальнейшие действия мастера — **Открыть форму для просмотра и ввода данных** (Open the form to view or enter information). Это позволит после завершения работы мастера сразу приступить к работе с договорами.

Для завершения создания формы мастером нажмите кнопку **Готово** (Finish). Сохранение формы производится автоматически.

В форме выводятся данные по договорам, которые были ранее загружены в базу (рис. 5.42). Основная форма содержит сведения из одной текущей записи таблицы ДОГОВОР, в которую добавлены справочные данные из таблицы ПОКУПАТЕЛЬ. В подчиненной форме для каждой текущей записи договора выводятся только записи плановых поставок товаров данного договора, т. е. связанные с отображенной в основной форме.

Код товара	Наименование товара	Срок поставки	Количество	ЦЕНА	Сумма поставки
T002	FDD 3,5	1	1000	360,00р.	0,00р.
T002	FDD 3,5	3	2000	360,00р.	0,00р.
T003	HDD Maxtor 20GB	1	14	1 280,00р.	0,00р.
T006	DIMM 64M PC100	2	200	360,00р.	0,00р.
*					

**Рис. 5.42.** Форма с подчиненной формой, созданная мастером

Каждая запись в подчиненной форме образуется на основе одной записи базового источника — таблицы ПОСТАВКА\_ПЛАН, в которую добавляются справочные данные из таблицы ТОВАР.

## Доработка интерфейса

Чтобы подготовить более удобный интерфейс для работы с документом "Договор", соответствующий макету формы, отредактируйте созданную мастером форму средствами конструктора.

Откройте полученную мастером многотабличную форму ДОГОВОРЫ С ПОКУПАТЕЛЯМИ в режиме конструктора. Для этого выберите ее в области переходов и в контекстном меню выполните команду **Конструктор** (Design View). Если форма была открыта, то для перехода в режим конструктора выберите из списка кнопки **Режим** (View) на вкладке ленты **Главная** (Home) **Конструктор** (Design View).

В основной форме переместите и отредактируйте поля, как показано на рис. 5.46. Уточните текст надписей полей, шрифт и размеры полей и надписей.

Чтобы увеличить рабочее пространство, уберите с экрана область переходов, щелкнув на кнопке **Открыть/закрыть границу области переходов** (Shutter Bar Open/Close Button) в ее правом верхнем углу. Для редактирования внешнего вида элементов формы переключайтесь в режим макета, т. к. в этом режиме можно сразу видеть результаты выполнения многих операций.

Учтите, что все элементы управления основной формы включены в один макет. Для перемещения макета выделите его, щелкнув в левом верхнем углу области макета или выполните команду **Выбрать макет** (Select Layout) на вкладке ленты **Упорядочить** (Arrange) в группе **Строки и столбцы** (Rows & Columns), после этого макет можно перетащить в нужное место. Изменение отдельного элемента требует удаления его из макета. Выделите нужный элемент вместе с надписью и выполните команду **Макет | Удалить макет** (Layout | Remove Layout) в контекстном меню. Для образования нового макета из удаленных элементов управления следует выделить их и выполнить команду **В столбик** (Stacked) или **Табличный** (Tabular).

Для придания элементам одинакового вида удобно копировать форматирование одного элемента и применять его к другим элементам. Это позволяет сделать команда **Формат по образцу** (Format Painter), размещенная на вкладках **Главная** (Home) или **Формат** (Format) в группе **Шрифт** (Font). Если на значке команды **Формат по образцу** щелкнуть дважды, скопированный формат можно применять несколько раз. Закончив выполнение форматирования по образцу, повторно щелкните на значке команды.

Измените порядок отображения полей в подчиненной форме. Начиная с Access 2007 стало удобно изменять размер, перемещать, удалять и добавлять поле. При любом из этих действий все другие поля макета автоматически встают на новые места (сдвигаются или наоборот раздвигаются), сохраняя правильное размещение в строке.

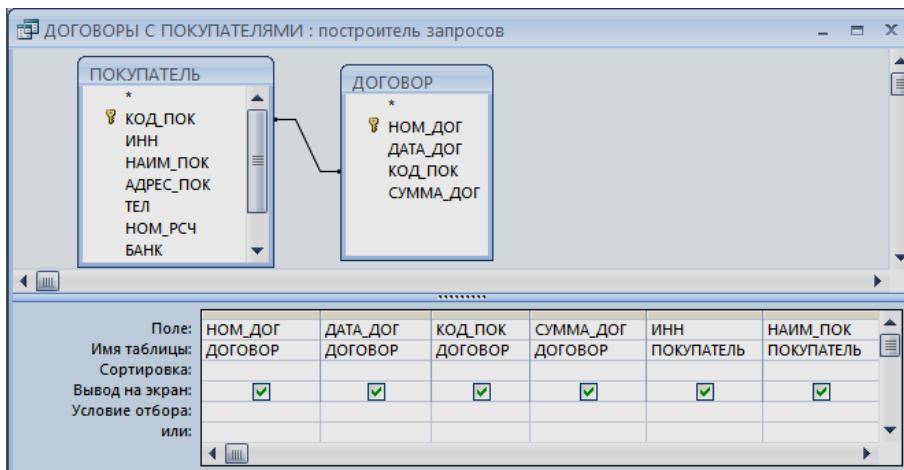
Удалите надпись подчиненной формы. Для этого выделите надпись и сначала удалите ее из макета, выполнив в контекстном меню команду **Макет | Удалить макет** (Layout | Remove Layout), затем в контекстном меню выполните команду **Удалить** (Delete).

Просмотрите источник записей основной формы. Для этого откройте свойства формы, выполнив соответствующую команду в контекстном меню. Открыть свойства формы также можно, выбрав в окне свойств в списке поля **Тип выделенного элемента** (Selection Type) строку **Форма** (Form). На вкладке **Данные** (Data) в строке **Источник записей** (Record Source) представлена инструкция SQL, записанная мастером в качестве источника записей основной формы. Чтобы просмотреть инструкцию SQL в режиме конструктора запросов, нажмите значок построителя в конце строки. Соответствующий запрос в режиме конструктора показан на рис. 5.43.

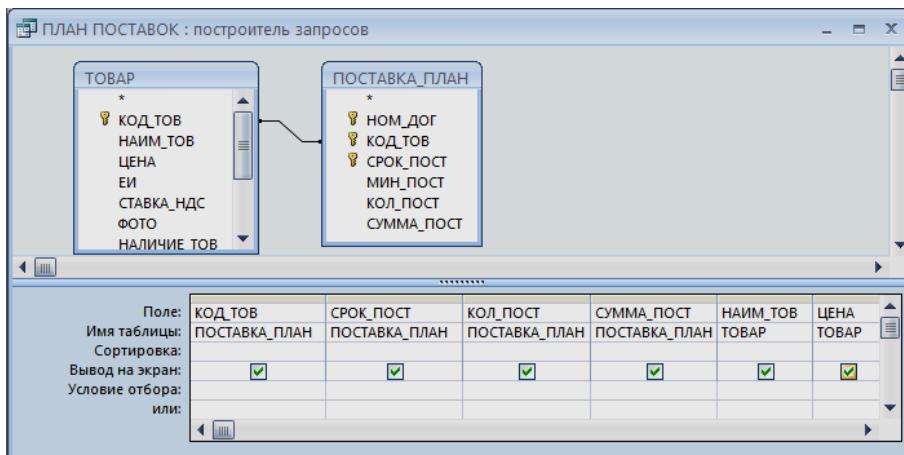
Аналогичным образом просмотрите, какая таблица является источником записей подчиненной формы. Инструкция SQL в режиме конструктора запросов, являющаяся источником записей подчиненной формы, показана на рис. 5.44.

Если в основную или подчиненную форму необходимо добавить поля, представленные в источнике записей, установите курсор на соответствующей форме,

нажмите кнопку **Добавить поля** (Add Existing Fields) на вкладке ленты **Конструктор** (Design) и из открывшегося списка перетащите нужное поле в форму. В список полей включены только те поля таблиц, которые были выбраны в запросе.



**Рис. 5.43.** Запрос — источник записей основной формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ



**Рис. 5.44.** Запрос — источник записей подчиненной формы ПЛАН ПОСТАВОК

Для добавления недостающих полей в источник записей основной или подчиненной формы можно изменить просмотренные запросы (см. рис. 5.43 и 5.44).

Однако проще щелкнуть в области **Список полей** (Field List) на значке **Показать все таблицы** (Show All tables). В разделе **Поля доступные в данном представлении** (Fields available for this view) отобразятся все поля таблиц, включенных в подсхему данных формы (основной или подчиненной). Кроме того, в других разделах отобразятся остальные таблицы базы данных. Перетаскивая поля из любого

раздела области, вы автоматически меняете запрос источника записей формы. Для отображения этого запроса в графическом виде также можно не открывать окно свойств и затем построитель запросов. Достаточно выполнить команду **Изменение источника записи** (Edit Record Source) в контекстном меню области **Список полей** (Field List).

Проверьте связь подчиненной формы с главной. Для этого откройте свойства подчиненной формы, предварительно выделив ее рамку. В свойствах **Подчиненная форма/отчет** (Subform/Subreport) на вкладке **Данные** (Data) в строке **Подчиненные поля** (Link Child Fields) указано имя поля связи НОМ\_ДОГ из подчиненной таблицы ПОСТАВКА\_ПЛАН, в строке **Основные поля** (Link Master Fields) указано имя поля связи, в данном примере тоже НОМ\_ДОГ из главной таблицы ДОГОВОР.

## Создание кнопок

Дополните форму новыми элементами управления — кнопками. Создайте в области заголовка основной формы кнопки для перехода к следующему и предыдущему договору (т. е. для перехода к другой записи источника основной формы таблицы ДОГОВОР).

В режиме конструктора нажмите в группе **Элементы управления** (Controls) кнопку **Использовать мастера** (Use Control Wizards), а затем **Кнопка** (Button). Не отпуская курсора, перенесите кнопку в нужное место и вычертите ее рамку. Запустится мастер создания кнопок (Command Button Wizard) (рис. 5.45).

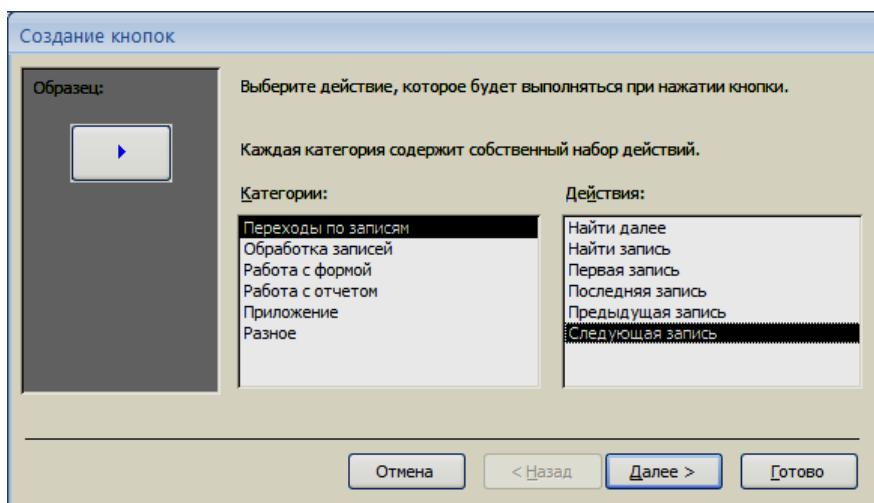


Рис. 5.45. Окно мастера создания кнопок

В окне мастера создания кнопок в группе **Категории** (Categories) выберите строку **Переходы по записям** (Record Navigation), в группе **Действия** (Actions) —

действие, которое должно выполняться при нажатии кнопки: **Предыдущая запись** (Go to Previous Record). В следующем диалоговом окне мастера выберите вид кнопки с текстом на ней **Текст** (Text) или с рисунком **Рисунок** (Picture) и выберите рисунок из списка. После завершения работы мастера кнопка с выбранным текстом или рисунком встраивается в форму. Аналогичные действия произведите для встраивания кнопки перехода к следующей записи, выбрав соответственно в области **Действия** (Actions) — **Следующая запись** (Go to Next Record). Размер и надписи кнопок редактируются, как и другие элементы.

Создайте кнопку для закрытия формы.

Отредактированная форма в режиме просмотра приведена на рис. 5.46.

ДОГОВОР № Д777

от 14.06.2007

**Покупатель** Компьютер маркет

Код покупателя	П001	ИНН	778957651111
Адрес	Москва	БАНК	Мост
Тел.	(812)345-2345	Номер счета	76358509763264536567

Код товара	Наименование товара	Срок поставки	Количество	ЦЕНА	Сумма поставки
T002	FDD 3,5		1	360,00р.	360 000,00р.
T002	FDD 3,5		3	360,00р.	720 000,00р.
T003	HDD Maxtor 20GB		1	1 280,00р.	17 920,00р.
T006	DIMM 64M PC100		200	360,00р.	72 000,00р.
*					

Запись: 1 из 4    Сумма по договору 1 169 920,00р.

Рис. 5.46. Отредактированная форма документа "Договор" в режиме просмотра

## Ограничение доступа к данным таблиц базы

Поля таблиц справочной информации ПОКУПАТЕЛЬ и ТОВАР должны использоваться только для отображения. Поэтому целесообразно защитить их от не-произвольных изменений при работе с формой. В основной форме такая защита нужна для полей таблицы ПОКУПАТЕЛЬ (НАИМ\_ПОК, ИНН, АДРЕС и др.), в подчиненной форме для полей таблицы ТОВАР (НАИМ\_ТОВ, ЦЕНА, ЕИ).

Для защиты поля выделите его и откройте окно свойств. В окне свойств на вкладке **Данные** (Data) в строке **Блокировка** (Locked) выберите **Да** (Yes). После установки этого свойства поле доступно только для чтения.

Если необходимо установить режим, при котором возможно только добавление новых договоров в базу данных, и запрещен просмотр существующих договоров, откройте свойства формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ и на вкладке **Данные** (Data) в строке **Ввод данных** (Data Entry) выберите значение **Да** (Yes).

## Создание полей со списком

Как уже было сказано ранее, при загрузке договоров через разработанную форму обязательно должны вводиться ключевые поля таблиц — объектов загрузки и ключи связи с таблицами справочных данных. Для удобства ввода значений в эти поля целесообразно соответствующие элементы управления создавать как **Поле со списком** (Combo Box). Поле со списком объединяет поле формы, в которое нужно ввести данные, и список отображаемых записей из связанной главной таблицы. В списке можно выбрать нужное значение и ввести его в поле формы (рис. 5.47).

При этом повышается достоверность вводимой информации и снижается вероятность ошибок при вводе. Отображение данных из справочных таблиц при вводе идентификатора обеспечивает ввод в поле связи загружаемой подчиненной записи только тех значений, которые присутствуют в записях главной таблицы, что необходимо для успешного завершения ввода при установленном параметре целостности в схеме данных.

Создайте с помощью мастера поле со списком для ввода в таблицу ДОГОВОР только тех значений кода покупателя КОД\_ПОК (рис. 5.47), которые есть в таблице ПОКУПАТЕЛЬ. Так как сразу после ввода кода покупателя через поле со списком в форме отобразятся все реквизиты покупателя, проверьте соответствие кода и наименования покупателя в документе "Договор".

Код товара	Номер	Количество	ЦЕНА	Сумма поставки
T003	2	10	1 280,00р.	25 900,00р.

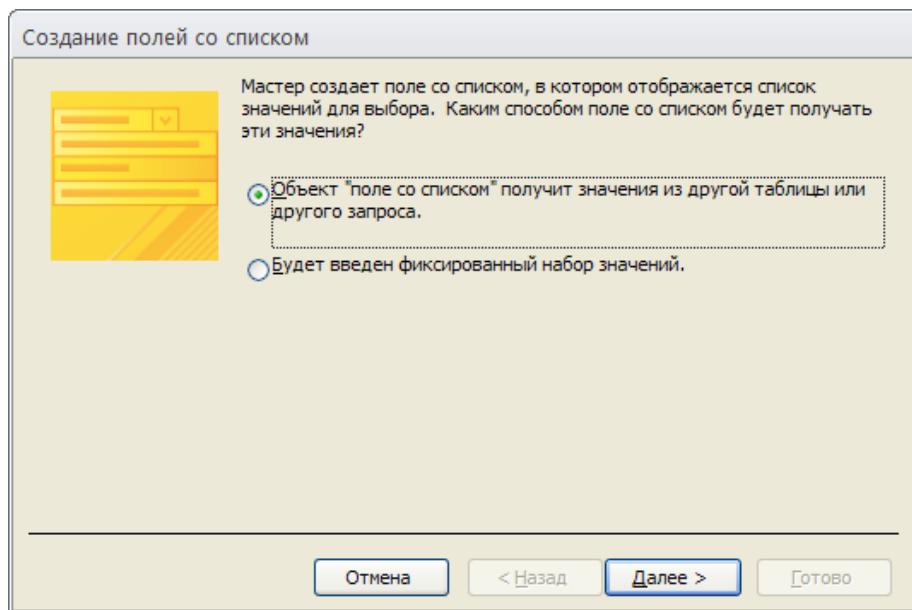
Поле ввода кода покупателя в таблицу ДОГОВОР      Список записей из таблицы ПОКУПАТЕЛЬ

**Рис. 5.47.** Поле со списком записей справочной таблицы ПОКУПАТЕЛЬ для ввода кода в запись таблицы ДОГОВОР

## Создание поля со списком мастером

В открытой в режиме макета или конструктора форме нажмите в группе Элементы управления (Controls) кнопку Использовать мастера (Use Control Wizards), а затем кнопку Поле со списком (Combo Box), переместите курсор в нужное место, нажмите кнопку мыши и, не отпуская ее, вычертите рамку элемента. После отпускания кнопки мыши запустится мастер и откроется диалоговое окно Создание полей со списком (Combo Box Wizard).

В окне мастера определите способ, которым список поля получает свои значения. Для формирования списка из связанной таблицы выберите — **Объект "поле со списком" получит значения из другой таблицы или другого запроса** (I want the combo box to look up the values in a table or query) (рис. 5.48).



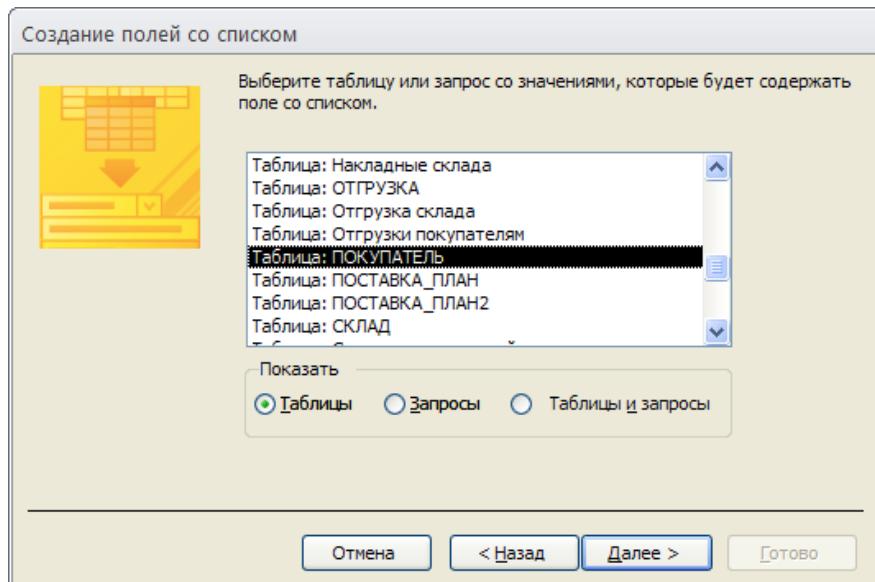
**Рис. 5.48.** Выбор способа формирования списка

В следующем окне выберите таблицу ПОКУПАТЕЛЬ, которая будет поставлять значения в список поля (рис. 5.49).

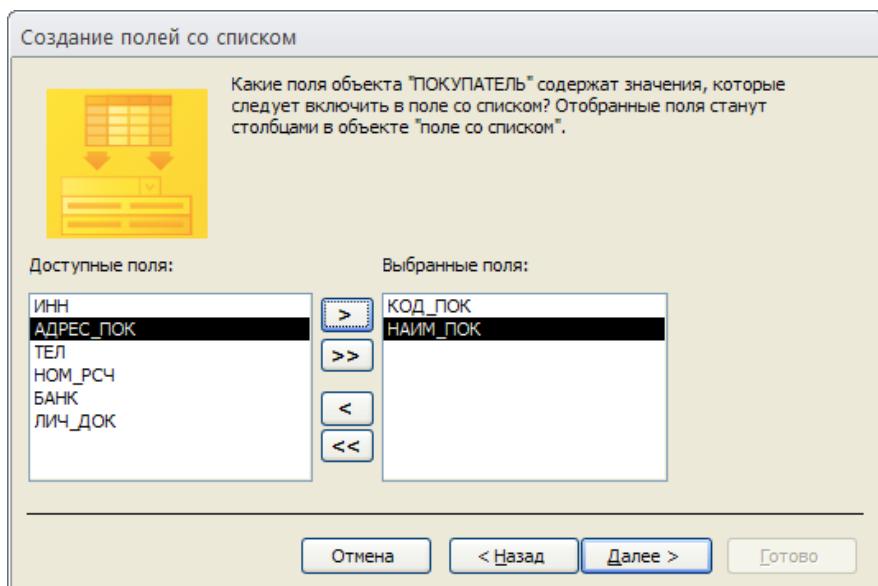
Затем выберите ключевое поле КОД\_ПОК, а также поле НАИМ\_ПОК для отображения кода и наименования в строках списка (рис. 5.50).

В следующем окне мастера, если необходимо, выберите порядок сортировки и далее настройте ширину столбцов списка и вариант с отображением ключевого столбца (рис. 5.51).

Далее выберите поле списка КОД\_ПОК, являющееся ключом связанной таблицы ПОКУПАТЕЛЬ. Из этого поля будет выбираться значение для ввода в поле формы и сохранения его в записи таблицы ДОГОВОР (рис. 5.52).

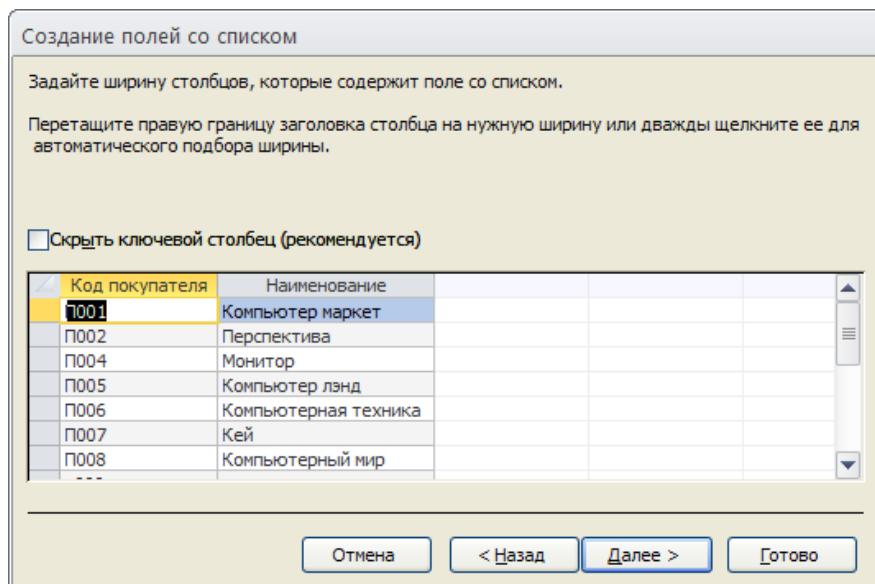


**Рис. 5.49.** Выбор источника данных для формирования списка

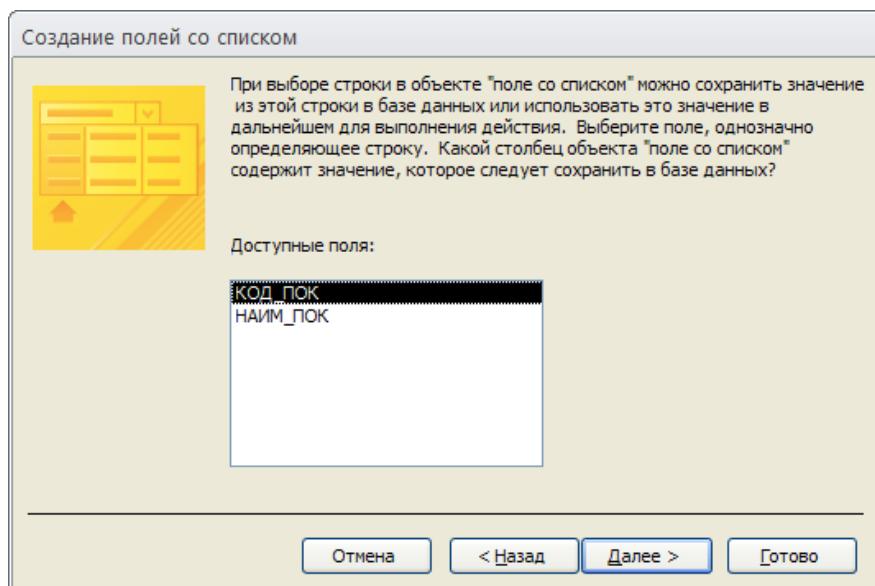


**Рис. 5.50.** Выбор полей для формирования записи списка

В следующем окне (рис. 5.53) отметьте переключатель **Сохранить в поле** (Store that value in this field) и выберите поле формы КОД\_ПОК (поле таблицы ДОГОВОР), в которое будут вводиться значения из списка.



**Рис. 5.51.** Окно для оформления столбцов списка



**Рис. 5.52.** Окно выбора поля источника значений для поля формы

Далее введите надпись поля со списком — **Код покупателя** (рис. 5.54).

Нажмите **Готово** (Finish). В результате будет создано новое поле со списком КОД\_ПОК. Замените поле с кодом покупателя на созданное мастером поле со спи-

ском. Использование поля со списком покупателей в режиме формы иллюстрирует рис. 5.47. Если потребуется настройка ширины столбцов списка, в окне свойств поля со списком в строках **Ширина списка** (List Width) и **Ширина столбцов** (Column Width) задайте подходящие значения.

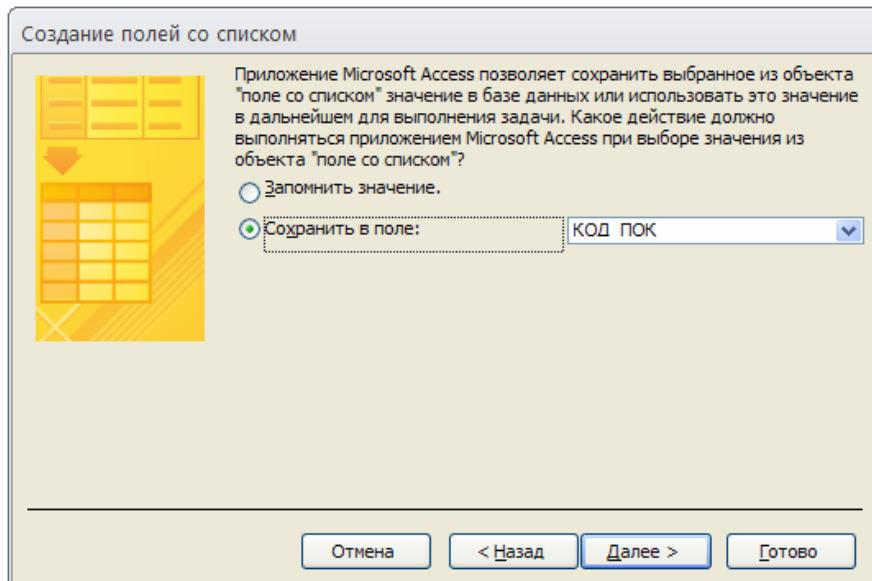


Рис. 5.53. Выбор поля формы, в которое вводится значение из списка

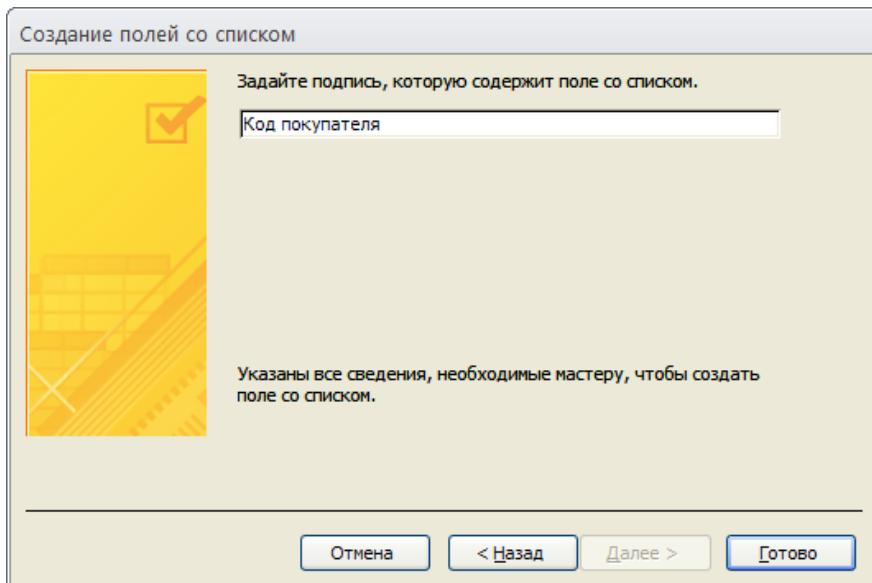


Рис. 5.54. Определение подписи поля со списком

## Создание поля со списком в режиме конструктора

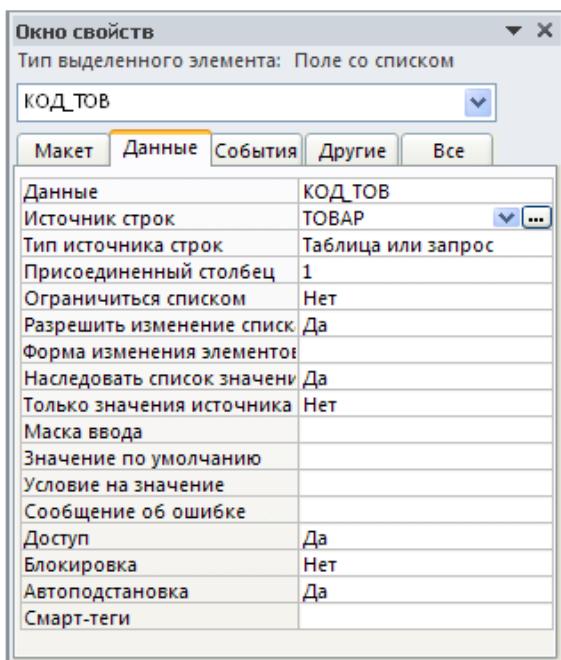
Мастер создает новое поле со списком. В режиме конструктора можно преобразовать существующее поле в поле со списком.

В подчиненной форме ПЛАН ПОСТАВОК преобразуйте поле КОД\_ТОВ (код товара) в поле со списком. Для этого выберите в контекстном меню этого поля команду **Преобразовать элемент в** (Change To) и далее строку — **Поле со списком** (Combo Box).

Затем откройте свойства поля (рис. 5.55). На вкладке **Данные** (Data) в строке **Данные** (Control Source) будет указано поле, которое будет заполняться в форме — КОД\_ТОВ. Выбором в строке **Источник строк** (Row Source) таблицы ТОВАР указывается, что из ее записей надо формировать список. В строке **Тип источника строк** (Row Source Type) автоматически появляется **Таблица или запрос** (Table/Query), а в строке **Присоединенный столбец** (Bound Column) — 1. Таким образом, чтобы сформировать поле со списком, может оказаться достаточным преобразовать обычное поле и выбрать его источник строк.

### **ВНИМАНИЕ!**

Список поля будет включать значения первого указанного в структуре таблицы столбца. В приведенном примере таким полем является поле КОД\_ТОВ (код товара) — ключ таблицы ТОВАР, поэтому список поля будет построен правильно. Если же на первом месте в структуре таблицы окажется другое поле, список будет содержать неверные значения. В таком случае нельзя в качестве источника строк выбирать таблицу. Нужно с помощью построителя запросов или вручную записать в качестве источника строк инструкцию SQL, выбирающую из таблицы нужное поле, в данном случае поле с кодом товара.



**Рис. 5.55.** Определение свойств создаваемого поля со списком без использования мастера

Если в списке необходимо отображать два столбца — код товара и наименование, в конце строки свойства **Источник строк** (Row Source) щелкните на кнопке **Построитель запросов** (Query Builder). Добавьте в бланк запроса таблицу ТОВАР и включите в запрос ее поля КОД\_ТОВ и НАИМ\_ТОВ. Закройте окно построителя выражений, подтвердив сохранение инструкции SQL созданного запроса и обновление свойства. Теперь в строке **Источник строк** (Row Source) будет записана инструкция SQL:

```
SELECT ТОВАР.КОД_ТОВ, ТОВАР.НАИМ_ТОВ FROM ТОВАР;
```

В окне свойств поля на вкладке **Макет** (Format) замените значение свойства поля **Число столбцов** (Column Count) с 1 на 2. Кроме того, установите ширину столбцов и ширину списка, указав, например, **2 см.;5 см.** и **7 см.** соответственно.

Для создания нового поля со списком без помощи мастера надо перетащить в форму элемент управления **Поле со списком** (Combo Box), когда кнопка **Использовать мастера** (Use Control Wizards) не активна.

## Вычисления в документе

Вычисления в форме могут осуществляться как в каждой записи формы, так и для группы записей при формировании итоговых величин. Расчетные величины только отображаются в вычисляемых полях формы. Для сохранения результатов в таблице базы данных может быть использован макрос или процедура на VBA. Примеры процедур, обеспечивающих обновление полей в базе данных при вычислениях в форме, приведены на прилагаемом к книге компакт-диске.

Рассчитайте в форме ДОГОВОРЫ С ПОКУПАТЕЛЯМИ значения стоимости заказанного товара в каждой строке подчиненной формы, а также общую стоимость заказанного в договоре товара. Разместите соответствующие вычисляемые поля, как предусмотрено в макете формы.

## ЗАМЕЧАНИЕ

Если в форме отображается поле СУММА\_ПОСТ из таблицы ПОСТАВКА\_ПЛАН, поместите вычисляемое поле рядом. Сравните введенные в поле СУММА\_ПОСТ данные с расчетными и при необходимости подправьте. Если поле СУММА\_ПОСТ пусто, скопируйте в него рассчитанное значение, и оно сохранится в поле таблицы.

Чтобы произвести вычисления на основе данных одной записи создайте в подчиненной форме вычисляемое поле и запишите туда выражение =КОЛ\_ПОСТ\*ЦЕНА. Для этого откройте подчиненную форму в режиме макета и, нажав на вкладке ленты **Конструктор** (Design) в группе **Элементы управления** (Controls) кнопку **Поле** (Text Box), поместите этот свободный элемент управления, не связанный с источником данных, в конце строки данных. Проследите, чтобы при вставке элемента в конец строки данных рядом с существующим полем появилась желтая вертикальная линия. Это позволит вставить новый столбец непосредственно в макет формы.

Откройте свойства нового поля и в строку **Данные** (Control Source) на одноименной вкладке запишите выражение. При этом можно воспользоваться построи-

телем выражений. Можно переключиться в режим конструктора и ввести выражение непосредственно в элемент управления **Свободный** (Unbound). Выражение должно начинаться со знака равенства. В качестве операндов выражения чаще всего используются имена полей, константы, а в качестве операторов — знаки арифметических операций.

Вычисление итоговых значений для группы записей подчиненной формы может быть выполнено с помощью встроенных статистических функций только в области примечания подчиненной формы. Для подсчета общей стоимости договора создайте в этой области свободный элемент управления и запишите туда выражение, содержащее статистическую функцию подсчета суммарной стоимости товаров =Sum(КОЛ\_ПОСТ\*ЦЕНА). В качестве аргумента статистической функции нельзя использовать имена других вычисляемых элементов управления формы. Ссылаться можно только на имена полей, т. к. они являются источником данных.

Для отображения результата вычисления в основной форме создайте там вычисляемый элемент управления и запишите выражение

= [ПЛАН ПОСТАВОК].Form! [Поле8]

Это выражение является ссылкой на элемент управления Поле8 в подчиненной форме, содержащий общую стоимость товаров. Общий формат такой ссылки приведен выше.

Чтобы элементы управления из области примечаний подчиненной формы не отображались при просмотре формы, установите в свойстве примечания **Вывод на экран** (Visible) значение **Нет** (No).

Перейдите в режим просмотра в форме и убедитесь, что результаты расчетов отображаются правильно.

## Работа с документами

Форма ДОГОВОРЫ С ПОКУПАТЕЛЯМИ предназначена для ввода, просмотра, корректировки и удаления данных о договорах. В основной форме отображаются данные из одной записи таблицы ДОГОВОР, дополненные данными из таблицы ПОКУПАТЕЛЬ. В подчиненной форме выводятся связанные с договором записи из таблицы ПОСТАВКА\_ПЛАН. Эти записи дополнены данными из таблицы ТОВАР.

Чтобы начать работу с формой, выберите ее в области переходов и в контекстном меню выполните команду **Открыть** (Open).

Для просмотра и корректировки данных конкретного договора сделайте нужную запись текущей. Для этого воспользуйтесь созданными в форме кнопками **Предыдущий договор** () и **Следующий договор** () или стандартными кнопками перехода по записям, предусмотренными как для основной, так и для подчиненной формы.

При большом числе записей используйте команду поиска нужной записи. Для этого установите курсор на поле с номером договора и нажмите кнопку **Найти** (Find) в соответствующей группе вкладки **Главная** (Home). В диалоговом окне **Поиск и замена** (Find and Replace) задайте номер нужного договора. После нажа-

тия в этом окне кнопки **Найти далее** (Find Next) в форме отобразятся данные о заданном договоре.

### **ВНИМАНИЕ!**

Грубой ошибкой является попытка перейти к нужной записи путем ввода нового значения в поле идентификации текущей записи, например, нового значения номера договора в форме ДОГОВОР. Такие действия могут привести лишь к изменению значения идентификатора в текущей записи.

Для просмотра договоров в порядке возрастания или убывания его номеров установите курсор на поле с номером договора и нажмите кнопку **Сортировка по возрастанию** (Sort Ascending) или **Сортировка по убыванию** (Sort Descending) в группе **Сортировка и фильтрация** (Sort & Filter) на вкладке ленты **Главная** (Home).

Рассмотрим ввод в базу данных нового документа "Договор". Для загрузки данных о новом договоре через форму ДОГОВОРЫ С ПОКУПАТЕЛЯМИ нажмите на нижней панели кнопок перехода по записям кнопку **Новая запись** (New (blank) record) . Текущей становится пустая запись источника основной формы — таблицы ДОГОВОР и пустая запись источника подчиненной формы — таблицы ПОСТАВКА\_ПЛАН.

В главной форме заполните обязательные поля: номер договора (ключ таблицы ДОГОВОР) и код покупателя (поле связи с таблицей ПОКУПАТЕЛЬ), а также поле даты, загружаемой в таблицу ДОГОВОР. Справочные поля из таблицы ПОКУПАТЕЛЬ автоматически отобразятся после ввода кода покупателя.

С полем типа данных **Дата/время** (Date/Time) связывается календарь. В форме при выборе поля с датой заключения договора справа от него отображается значок календаря. Выбранная в календаре дата вводится в поле ДАТА\_ДОГ. Однако если для поля определена маска ввода, календарь недоступен для использования. Если маска ввода была задана в свойствах таблицы, она автоматически наследуется полем в форме. Если после создания формы удалить маску в поле таблицы, она сохранится в поле формы. Для удаления маски в поле формы откройте свойства поля и очистите соответствующую строку на вкладке **Данные** (Data).

### **ЗАМЕЧАНИЕ**

При наличии календаря сохраняется возможность ручного ввода даты в поле.

Сформированная запись автоматически сохранится в таблице ДОГОВОР при переходе в подчиненную форму.

Поле суммы по договору может быть рассчитано и введено в таблицу ДОГОВОР после ввода данных в подчиненной форме.

### **ЗАМЕЧАНИЕ**

Сохранить новую запись можно, выполнив команду **Сохранить** (Save) в группе **Записи** (Records) на вкладке ленты **Главная** (Home) или просто щелкнув мышью на области выделения записи.

Перейдите к формированию записей в подчиненной форме. В каждой новой записи введите значения обязательных полей — кода товара (ключевое), месяца поставки (ключевое), а также полей — минимальная партия поставки и количество. После ввода кода товара справочные данные — наименование, единица измерения и цена из таблицы ТОВАР — отобразятся автоматически. Сформированная запись сохранится в таблице ПОСТАВКА\_ПЛАН при переходе к другой записи.

Если в строке подчиненной формы создано вычисляемое поле для расчета стоимости поставки по каждому товару, значение стоимости будет автоматически вычислено после заполнения полей ЦЕНА и КОЛИЧЕСТВО.

Если в примечании подчиненной формы создано вычисляемое поле для расчета общей стоимости поставки и на него сделана ссылка в вычисляемом поле основной части формы, после сохранения записи в подчиненной форме в главной форме отобразится общая сумма по договору.

### **ЗАМЕЧАНИЕ**

Для расчета стоимостей с сохранением результата в соответствующих полях таблиц ПОСТАВКА\_ПЛАН и ДОГОВОР должны быть подготовлены процедуры обработки событий на VBA, рассмотренные в прилагаемом компакт-диске.

Рассмотренные ранее способы перехода по записям формы не обеспечивают пользователя достаточно удобным интерфейсом для выборки нужных документов. При работе с формами — электронными аналогами документов — пользователю часто необходимо иметь средства, обеспечивающие быстрый поиск нужного документа по заданным условиям отбора. Например, могут задаваться условия отбора для выборки документа по его номеру или по заданному временному диапазону дат. В общем случае целесообразно предусмотреть выборку по заданному идентификатору объекта (товара, покупателя, склада, договора и т. п.) или по диапазону количественных показателей.

## **Выборка документа по его идентификатору**

Рассмотрим организацию выборки нужного договора по заданному номеру. Предусмотрим возможность ввода номера договора в диалоге с пользователем.

Для этого в источнике записей главной формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ — запросе на выборку из записей двух взаимосвязанных таблиц — в поле номера договора НОМ\_ДОГ определим параметр запроса [Номер договора] (рис. 5.56). Представленное на рисунке окно построителя запросов открывается при щелчке на соответствующем значке в строке свойства формы **Источник записей** (Record Source).

Теперь при открытии формы будет выводиться диалоговое окно (рис. 5.57) для ввода значения параметра запроса (Номер договора).

После ввода нужного номера договора в форме будут доступны данные соответствующего договора. Однако для перехода к просмотру другого договора потребуется заново открыть форму.

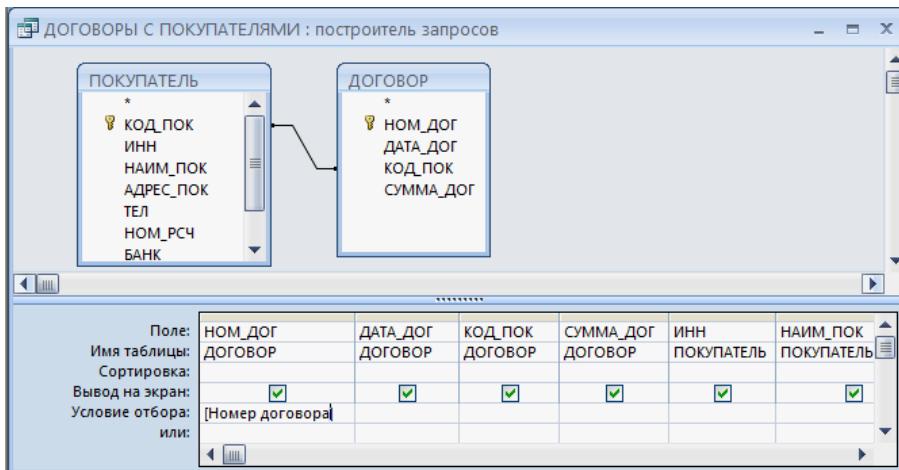


Рис. 5.56. Источник записей главной формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ

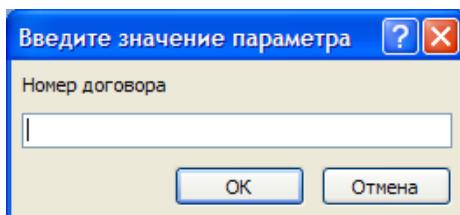
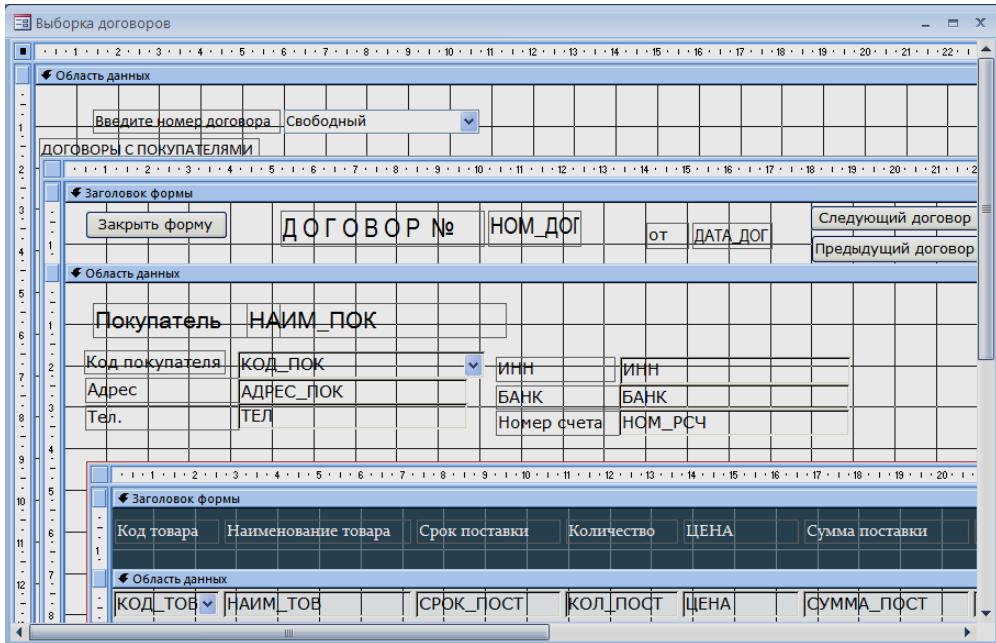


Рис. 5.57. Диалоговое окно ввода значения параметра запроса

Для удобства пользователя желательно создание такого интерфейса, который обеспечит возможность выбора документа при постоянно открытой форме. Для этих целей может быть подготовлена форма интерфейса, в которую встраивается форма документа. Форма интерфейса должна включать элемент управления, который обеспечивает ввод значения параметра запроса.

Создадим пустую форму (без источника записей), воспользовавшись командой **Конструктор форм** (Form Design) в группе **Формы** (Forms) на вкладке ленты **Создание** (Create), и присвоим ей имя Выборка договоров. Создадим в форме, открытой в режиме конструктора, с помощью мастера элемента управления — поле со списком. Выберем в качестве источника формирования значений списка таблицу ДОГОВОР и ее поле — Номер договора. Полю со списком в его свойствах дадим имя, совпадающее с названием параметра запроса — Номер договора. В надписи поля запишем обращение к пользователю "Введите номер договора" (рис. 5.58). Перетащим из окна базы данных форму ДОГОВОРЫ С ПОКУПАТЕЛЯМИ в область данных формы Выборка договоров. Полученная составная форма в режиме конструктора приведена на рис. 5.58.

Удалите надпись встроенной формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ, в свойствах макета формы Выборка договоров **Область выделения** (Record Selectors), **Кнопки перехода** (Navigation Buttons), **Полосы прокрутки** (Scroll Bars) выберите значение **Нет** (No).



**Рис. 5.58.** Форма для выборки договора по задаваемому в диалоге номеру в режиме конструктора

Код товара	Наименование товара	Срок поставки	Количество	ЦЕНА	Сумма поставки
T003	HDD Maxtor 20GB		2	1 280,00р.	25 900,00р.
T004	Корпус MiniTower		3	916,00р.	27 480,00р.
T001	Монитор 17LG		1	1 000,00р.	0,00р.
T001	Монитор 17LG		2	1 000,00р.	0,00р.
*					

Сумма по договору 778 024,00р.

**Рис. 5.59.** Интерфейс, обеспечивающий выборку нужного договора в поле со списком

В режиме формы при выборе из списка номера нужного договора он отобразится во встроенной форме ДОГОВОРЫ С ПОКУПАТЕЛЯМИ (рис. 5.59).

## Выборка документа по диапазону дат

Для формы с именем ДОГОВОРЫ С ПОКУПАТЕЛЯМИ предусмотрим возможность ввода пользователем диапазона дат заключения договоров и выборки соответствующих договоров. Для этого подготовим интерфейс, обеспечивающий выборку документов по диапазону дат, по аналогии с подготовкой интерфейса, рассмотренной в предыдущем примере.

В запросе — источнике записей главной формы (см. рис. 5.56) удалим ранее заданный параметр запроса [Номер договора] и зададим в поле даты заключения договора ДАТА\_ДОГ два параметра запроса. Параметры определим в выражении Between [ДатаНачальная] And [ДатаКонечная], записанном в условии отбора.

**Рис. 5.60.** Интерфейс, обеспечивающий выборку документов по диапазону дат

Как в предыдущем примере, создадим форму без источника записей, в которой разместим два свободных поля, не связанных с каким-либо источником данных. В свойствах одному полю дадим имя параметра датаНачальная, второму —

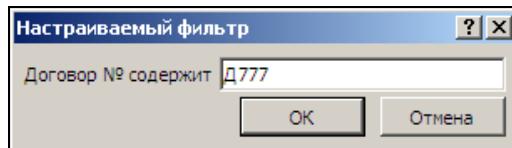
датаКонечная. Надписи полей изменим на **Дата начала периода** и **Дата окончания периода** соответственно. Перетащим из окна базы данных форму ДОГОВОРЫ С ПОКУПАТЕЛЯМИ в область данных формы.

Для связи календаря с полями ввода дат диапазона выберите для каждого поля в свойстве **Формат поля** (Format) на вкладке **Макет** (Format) нужный формат, например **Краткий формат даты** (Short Date).

В режиме формы при вводе дат начала и конца диапазона во встроенной форме отобразятся выбранные договоры (рис. 5.60).

## Выборка документов с помощью фильтрации

Используйте фильтры для выборки различных сведений из договоров. Установите курсор на поле с номером договора. Щелкните на кнопке **Фильтр** (Filter). В открывшемся окне откройте список **Текстовые фильтры** (Text Filters). Для поля доступны эти фильтры, потому что поле имеет текстовый тип данных. В списке щелкните **Содержит** (Contains), откроется настраиваемый фильтр (рис. 5.61). Введите нужный номер договора и нажмите **OK**. В форме отобразится выбранный договор. Открыть список **Текстовые фильтры** (Text Filters) можно также вызвав контекстное меню.



**Рис. 5.61.** Диалоговое окно задания условия отбора для фильтра

Чтобы снять фильтр с формы, используйте команду контекстного меню поля **Снять фильтр с НОМ\_ДОГ** (Clear filter from НОМ\_ДОГ). Если необходимо, задайте новые условия отбора по полю с номером договора.

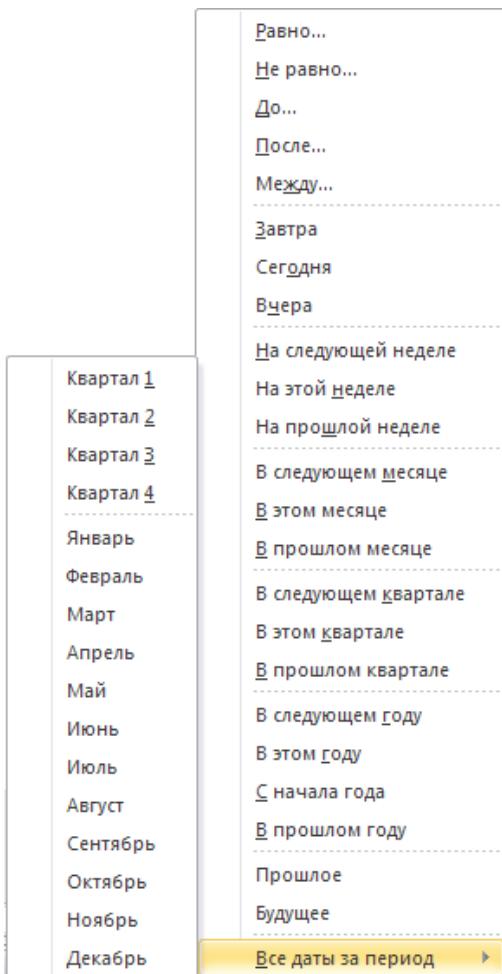
Чтобы отобрать договоры за заданный период, в контекстном меню поля с датой заключения договора откройте **Фильтры дат** (Date Filters). Список фильтров для дат позволяет осуществить выборку по самым разным критериям (рис. 5.62).

### ВНИМАНИЕ!

Итоговые данные по договору подсчитываются только для выбранных в результате фильтрации данных.

Для отбора в договорах сведений только о заданном товаре используйте текстовый фильтр по полю с наименованием товара. Для отбора заданного товара, заказанного в заданном месяце, не снимая фильтра с наименования товара, выполните фильтрацию по полю СРОК\_ПОСТ (срок поставки). Числовой фильтр по полю СРОК\_ПОСТ позволит отобрать данные в любом заданном интервале. При

использовании фильтров легко проверить запланированные поставки заданного товара в договорах и, если необходимо, откорректировать или дополнить их новыми строками.



**Рис. 5.62.** Фильтры для выборки записей по полю с типом данных **Дата/время**

Таким образом, использование фильтра позволит просматривать данные в различном представлении.

#### Задание 5.2. Создание интерфейса для работы с документом "Накладная"

Создайте многотабличную форму для ввода и корректировки в базе данных накладных на отгрузку товаров.

Подсхема для такой формы приведена на рис. 5.63. В результате загрузки в базу данных по накладным должны создаваться только записи таблиц НАКЛАДНАЯ и ОТГРУЗКА.

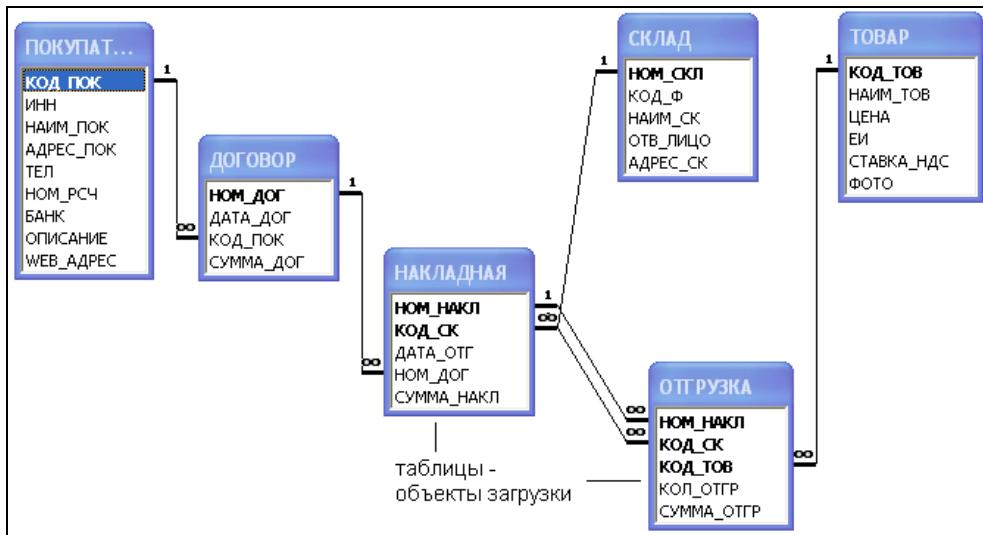


Рис. 5.63. Подсхема для составной формы по работе с накладными

- Осуществите проектирование интерфейса для загрузки данных документа НАКЛАДНАЯ:
  - определите общую структуру составной формы в соответствии с подсхемой данных для формы, приведенной на рис. 5.63;
  - разместите реквизиты в проекте формы так, чтобы обеспечить удобный ввод данных с документа, а также отображение справочной информации по договору, покупателю и товару.
- Выполните конструирование экранной формы, через которую будет осуществляться ввод, добавление и изменение записей таблиц НАКЛАДНАЯ и ОТГРУЗКА.
- Загрузите данные с документа "Накладная" через построенную форму.

## Контрольные вопросы

- Может ли форма, созданная мастером, содержать данные из нескольких таблиц?
- Как в форме установить защиту от изменения данных поля?
- В каком разделе свойств формы определяются основные параметры ее вида?
- Значения какой таблицы содержит поле со списком Код покупателя на рис. 5.47?
- В запись какой таблицы вводится значение, выбранное в поле со списком Код покупателя на рис. 5.47?
- Может ли поле со списком включать значения нескольких полей одной записи?
- Можно ли с помощью одного поля со списком сразу выбрать несколько значений и ввести их в несколько полей?

8. В каком случае при создании поля со списком не подключается мастер?
9. Откуда может получать значения поле со списком?
10. Можно ли для поля со списком пользоваться командами поиска и сортировки?
11. Где можно просмотреть информацию об источнике данных для поля со списком?
12. Где хранится имя поля, в которое должно вводиться значение, выбранное в списке?
13. Какое свойство приводит к открытию формы в режиме добавления записей, при котором невозможен просмотр ранее введенных записей?
14. Какой внешний вид подчиненной формы нужно выбрать, чтобы в ней выводились надписи полей, определенные в свойствах таблиц?
15. Позволяет ли мастер выбирать нужный стиль оформления формы?
16. Какой элемент управления позволяет создать вычисляемое поле в форме?
17. С какого знака начинается выражение, записываемое в вычисляемое поле?
18. Как вызвать построитель для формирования выражения в вычисляемом поле?
19. В каком разделе формы размещается вычисляемое поле для расчета итогового значения для подчиненной формы?
20. Можно ли при расчете итогового значения использовать в аргументе функции `Sum` имя другого вычисляемого элемента управления?
21. Как отобразить итоговое значение, рассчитанное в подчиненной форме, в главной форме?
22. Какой формат имеет ссылка на вычисляемое поле в подчиненной форме?
23. Сохраняется ли значение вычисляемого поля в таблице?
24. Что нужно сделать, если не работает связь основной и подчиненной форм?
25. Должны ли поля, по которым устанавливается связь между основной и подчиненной формами, иметь одинаковые имена?
26. Может ли связь между основной и подчиненной формами устанавливаться по нескольким полям?
27. Можно ли путем перетаскивания таблицы из окна базы данных в форму создать в ней подчиненную форму?
28. Может ли мастер построить форму на основе несвязанных таблиц?
29. Какие записи подчиненной таблицы отображаются в подчиненной форме?
30. Что указывает мастер в качестве источника записей в форме, содержащей поля нескольких взаимосвязанных таблиц?
31. Как просмотреть запрос — источник записей многотабличной формы в режиме конструктора?
32. Можно ли, изменив инструкцию SQL или запрос в режиме конструктора, повлиять на состав доступных в форме полей?
33. Может ли мастер построить форму на основе ранее созданного запроса?
34. Можно ли при создании новой записи в форме СПРАВОЧНИК ТОВАРОВ не вводить значение кода товара?

35. Какие значения содержит поле со списком Код товара, определенное в форме НАКЛАДНАЯ?
36. В запись какой таблицы вводится значение, выбранное в поле со списком Код товара в форме НАКЛАДНАЯ?
37. В каком случае к созданию кнопки не подключается мастер?
38. Можно ли изменить вид кнопки после завершения работы мастера?
39. Какие таблицы можно дополнить данными при работе в форме ДОГОВОРЫ С ПОКУПАТЕЛЯМИ?
40. Какие поля входят в список доступных в форме полей?
41. В каком элементе управления размещается подчиненная форма?
42. В каких свойствах указываются поля связи основной и подчиненной формы?
43. Какая команда позволяет создавать форму на основе выбранной таблицы?
44. Какая вкладка ленты активируется при открытии формы?
45. Откуда выбирается текст при автоматическом формировании надписи поля в форме?
46. Какие вкладки ленты используются при конструировании формы?
47. Сохраняет ли в форме поле такие свойства, как **Поле со списком** (List Box), **Список** (Combo Box), заданные при его определении в таблице?
48. Для чего предназначена группа **Элементы управления** (Controls) конструктора форм?
49. Как в режиме конструктора просмотреть доступные в форме поля?
50. Где хранится ссылка на источник записей формы?
51. Где размещена кнопка, по которой создается элемент управления **Надпись** (Label)?
52. По какой команде просматриваются свойства элемента управления?
53. Какая кнопка позволяет переключаться из режима конструктора в режим формы и обратно?
54. Можно ли в форме с помощью кнопок перехода по записям сделать текущей новую запись?
55. В какой последовательности загружаются таблицы базы данных?
56. Какой объект нужно использовать для первоначальной загрузки и корректировки взаимосвязанных таблиц базы данных?
57. В каких отношениях, как правило, находятся таблицы — источники основной и подчиненной формы?
58. Какие поля записеобразующей таблицы должны быть обязательно включены в подчиненную форму, чтобы через нее можно было вводить новые записи?
59. Можно ли через многотабличную форму осуществить ввод данных сразу в несколько таблиц?
60. Каким требованиям должен отвечать интерфейс пользователя для работы с документами, сохраняемыми в базе данных?

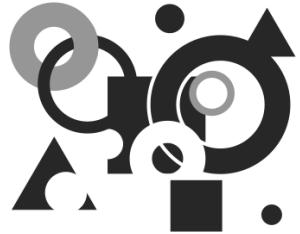
61. Назовите основные этапы проектирования формы для работы с документами, сохраняемыми в базе?
62. Какими средствами целесообразно создавать интерфейс для подготовки и ввода данных документа?
63. Можно ли через форму Накладная ввести запись об отгрузке товара, отсутствующего в таблице ТОВАР?

## Ответы

1. Да.
2. В свойствах поля на вкладке **Данные** (Data) в строке **Блокировка** (Locked) выбрать значение **Да** (Yes).
3. На вкладке **Макет** (Format).
4. Значения из поля Код покупателя таблицы ПОКУПАТЕЛЬ.
5. В поле Код покупателя текущей записи таблицы ДОГОВОР.
6. Да.
7. Нет.
8. Мастер не установлен на компьютере или на панели элементов не нажата кнопка **Использовать мастера** (Use Control Wizards).
9. Из таблицы, запроса, фиксированного набора значений.
10. Да.
11. В свойствах поля со списком, на вкладке **Данные** (Data) в строке **Источник строк** (Row Source).
12. В свойствах поля со списком, на вкладке **Данные** (Data) в строке **Данные** (Control Source).
13. Свойство формы **Ввод данных** (Data Entry) на вкладке **Данные** (Data).
14. **Ленточный** (Tabular).
15. Да.
16. Элемент управления **Поле** (Text Box), создаваемый по соответствующей кнопке на панели элементов.
17. Со знаком равенства (=).
18. Открыть свойства поля и на вкладке **Данные** (Data) в конце строки **Данные** (Control Source) вызвать построитель выражений.
19. В области примечания формы.
20. Нет.
21. Создать новое вычисляемое поле в главной форме и разместить в нем ссылку на вычисляемое поле в подчиненной форме.  
22. = [Имя подчиненной формы] . [Form]!  
[Имя вычисляемого поля в подчиненной форме].
23. Нет.
24. Установить правильные значения в свойствах элемента управления **Подчиненная форма/отчет** (Subform/ Subreport) **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields).

25. Нет.
26. Да.
27. Да, если установлен мастер подчиненных форм.
28. Нет.
29. Связанные с записью главной таблицы, отображаемой в основной форме.
30. Инструкцию SQL.
31. Вызвав построитель запросов кнопкой в конце строки **Источник записей** (Record Source) в окне свойств формы.
32. Да.
33. Да.
34. Нет. Код товара является ключом в таблице ТОВАР.
35. Значения из поля Код товара таблицы ТОВАР.
36. В поле Код товара текущей записи таблицы ОТГРУЗКА.
37. Если предварительно на вкладке ленты **Конструктор** (Design) в группе **Элементы управления** (Controls) не нажата кнопка **Использовать мастера** (Use Control Wizards).
38. Да.
39. ДОГОВОР и ПОСТАВКА\_ПЛАН.
40. Все поля таблицы или запроса, выбранного в качестве источника записей формы.
41. **Подчиненная форма/отчет** (Subform/ Subreport).
42. В свойствах элемента управления **Подчиненная форма/отчет** (Subform/ Subreport): **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields).
43. **Форма** (Form), **Несколько элементов** (Multiple Items), **Разделенная форма** (Split Form).
44. **Главная** (Home).
45. Из соответствующего свойства поля, заданного при определении структуры таблицы.
46. **Работа с макетами форм | Конструктор** (Form Layout Tools | Design), **Работа с макетами форм | Упорядочить** (Form Layout Tools | Arrange), **Работа с макетами форм | Формат** (Form Layout Tools | Format) и **Главная** (Home).
47. Да.
48. Для создания в форме полей и других элементов управления.
49. По кнопке **Добавить поля** (Add Existing Fields) на вкладке ленты **Работа с макетами форм | Конструктор** (Form Layout Tools | Design) в группе **Сервис** (Tools).
50. В свойствах формы на вкладке **Данные** (Data) в строке **Источник записей** (Record Source).
51. На вкладке ленты **Работа с макетами форм | Конструктор** (Form Layout Tools | Design) в группе **Элементы управления** (Controls).
52. **Страница свойств** (Property Sheet).

53. Кнопка **Режим** (View) на вкладке ленты инструментов формы или в строке состояния.
54. Да.
55. Сначала главные таблицы, затем подчиненные.
56. Многотабличную форму.
57. Один-ко-многим.
58. Все поля, составляющие ключ и не представленные в основной форме.
59. Да.
60. Интерфейс должен быть построен на основе форм, в которых наряду с основными данными документа отображаются справочные данные.
61. Проектирование подсхемы данных формы с указанием роли таблиц и макета формы с указанием связи элементов формы с полями таблиц.
62. Мастером форм с последующей доработкой в режиме макета и/или конструктора.
63. Нет.



## ГЛАВА 6

# Сводные таблицы и диаграммы. Анализ данных

Результаты решения различных задач часто представляются в двумерных не-нормализованных таблицах с многократно повторяющимися данными.

Даже нормализованные таблицы содержат повторяющиеся данные, такие как ключи связи. Например, в рассматриваемом примере базы данных подчиненная таблица ОТГРУЗКА содержит ключи связи с таблицами ТОВАР и НАКЛАДНАЯ, которые, естественно, многократно повторяются. Объединение таблицы ОТГРУЗКА с таблицами ТОВАР и НАКЛАДНАЯ дает таблицу с более полными сведениями, необходимыми при решении задач, и с большим числом повторяющихся описательных данных.

Просматривать и анализировать такие таблицы очень сложно. Используя многомерные интерактивные сводные таблицы, можно простыми средствами представлять данные, сгруппированные по различным измерениям, с различной степенью подробностей и вычислением итогов. Например, на рис. 1.3 была представлена сводная таблица для анализа отгруженного количества по любому из товаров, по различным покупателям и договорам по всем или некоторым месяцам, кварталам, годам.

В Access любая таблица, запрос или форма могут быть представлены в режиме сводной таблицы или диаграммы. Сводные таблицы и диаграммы позволяют анализировать данные большого объема и различной сложности в интерактивном режиме. Глубокий и всесторонний анализ данных может выполняться удобными средствами, практически несколькими щелчками мыши.

Благодаря интерактивности сводных таблиц и диаграмм можно изменять представление данных для просмотра дополнительных подробностей или вычисления различных итогов, таких как суммарное количество или среднее значение.

Источником данных для сводной таблицы и сводной диаграммы может быть не только таблица или запрос, но и базовый источник данных формы, в проекте Access — представление, хранящая процедура или пользовательская функция. Сводная таблица только использует данные источника. Изменять данные источника через сводную таблицу невозможно.

## Режим сводной таблицы

В Access таблица или запрос могут быть представлены не только в режиме конструктора и режиме таблицы, но и в режиме сводной таблицы и режиме сводной диаграммы. Для представления и управления данными в сводной таблице пользователь имеет возможность создавать и быстро модифицировать ее макет, выбирая и перетаскивая поля из раскрывающегося списка полей источника в рабочую область. Макет сводной таблицы позволяет отображать различные подмножества данных, рассчитывать и сравнивать итоговые значения для выбранных элементов данных. Режим сводной диаграммы предназначен для графического анализа данных таблицы.

В режимах сводной таблицы и сводной диаграммы могут открываться также источники данных форм. В базах данных SQL Server в режимах сводной таблицы и сводной диаграммы могут открываться представления, сохраненные процедуры и функции.

### Разработка сводной таблицы для таблицы базы данных

Рассмотрим таблицу ОТГРУЗКА в режиме сводной таблицы. Чтобы открыть таблицу ОТГРУЗКА в этом режиме, выберите ее в области переходов, в контекстном меню щелкните на кнопке **Открыть** (Open) и затем перейдите из режима таблицы в режим сводной таблицы, выбрав его в списке кнопки **Режим** (View).

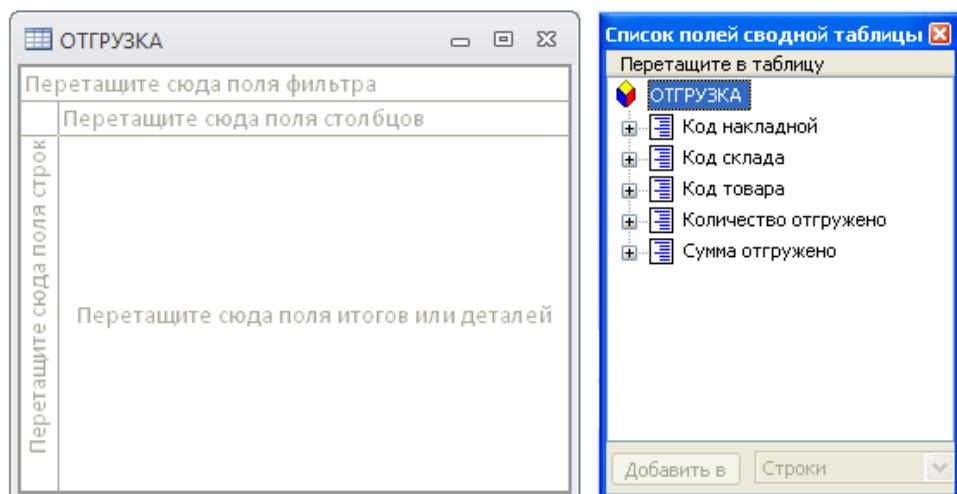


Рис. 6.1. Пустой макет сводной таблицы ОТГРУЗКА  
и список полей источника записей

При первом открытии таблицы в режиме сводной таблицы чаще всего в ее рабочую область не включено ни одного поля таблицы. Если список полей не открылся автоматически, нажмите кнопку **Список полей** (Field List) на вкладке ленты **Работа со сводными таблицами | Конструктор** (Pivot Table Tools | Design) в группе **Показать или скрыть** (Show/Hide). На рис. 6.1 показаны таблица ОТГРУЗКА, открытая в режиме сводной таблицы, и список доступных полей.

## Размещение полей в макете сводной таблицы

Для создания макета следует переместить поля в рабочую область сводной таблицы.

The screenshot shows two windows side-by-side. On the left is a PivotTable with the title 'OTGRUZKA'. The table has 'Код товара' (Code) in the rows and 'Количество отгружено' (Quantity Shipped) in the columns. The data looks like this:

	Количество отгружено
T001	10 5 2 2 2
T002	5 10 10
T003	8 3 1
T005	20 4 6 10
T006	5 1
T007	10
T008	1 12 10
T009	3 10

To the right is the 'Field List' window titled 'Список полей сводной таблицы' (List of fields in the PivotTable). It lists the available fields: 'OTGRUZKA' (with 'Код накладной' (Bill of Lading), 'Код склада' (Warehouse Code), 'Код товара' (Product Code), 'Количество отгружено' (Quantity Shipped), and 'Сумма отгружено' (Total Shipped Amount)), and a summary field 'Сумма отгружено' (Total Shipped Amount). A dropdown menu 'Добавить в' (Add to) is open, showing options: 'Строки' (Rows), 'Столбцы' (Columns), 'Фильтр' (Filter), 'Данные' (Data), and 'Детали' (Details). The 'Данные' (Data) option is selected.

Рис. 6.2. Сводная таблица с кодами товаров в заголовках строк и перечнем всех строк с количествами в области данных

Рассмотрим возможности сводных таблиц, последовательно создавая различные макеты для вывода детальных данных и подсчета итогов в различных разрезах данных.

Пусть необходимо подсчитать общее количество каждого из отгруженных товаров. Перетащим из списка полей сводной таблицы поле Код товара в область строк. Уникальные значения кодов товара составят заголовки строк сводной таблицы. Для размещения в сводной таблице строк со значениями количества отгруженного товара следует переместить соответствующее поле в область деталей. В области деталей размещаются поля, по которым будут подводиться итоги. Как правило, это числовые данные. После таких изменений макет примет вид, представленный на рис. 6.2.

В этом макете в первой колонке только один раз отображен код товара, а значения количества отгруженного сохранены в том же виде, что и в исходной таблице. Таблица ОТГРУЗКА в режиме таблицы представлена на рис. 6.3.

Код наклад	Код склада	КОД_ТОВ	Количество отгружено	Сумма отгруже
H001	C01	T001	10	10 000,00р.
H002	C01	T001	5	5 000,00р.
H002	C03	T001	2	2 000,00р.
H003	C01	T001	2	2 000,00р.
H003	C03	T001	2	2 000,00р.
H004	C01	T001	2	2 000,00р.
H002	C02	T002	5	1 800,00р.
H003	C02	T002	10	3 600,00р.
H004	C02	T002	10	3 600,00р.
H002	C02	T003	8	10 240,00р.
H002	C03	T003	3	3 840,00р.
H004	C02	T003	1	1 280,00р.
H001	C01	T005	20	23 060,00р.
H001	C02	T005	4	4 612,00р.
H002	C02	T005	6	6 918,00р.
H003	C02	T005	10	11 530,00р.
H001	C03	T006	5	1 800,00р.
H002	C03	T006	1	360,00р.
H001	C01	T007	10	0,00р.
H001	C01	T008	1	2 338,00р.
H003	C01	T008	12	28 056,00р.
H004	C01	T008	10	23 380,00р.
H002	C03	T009	3	2 367,00р.
H003	C02	T009	10	7 890,00р.
*			0	0,00р.

Рис. 6.3. Данные записей таблицы ОТГРУЗКА

Добавлять поля в области сводной таблицы можно также с помощью кнопки **Добавить в** (Add to), размещенной в нижней части окна списка полей (см. рис. 6.2). Выделите добавляемое поле. Выберите одно из значений из раскрывающегося списка справа от кнопки **Добавить в** (Add to):

- ❖ для добавления поля в область строк — **Строки** (Row Area);
- ❖ для добавления поля в область столбцов — **Столбцы** (Column Area);
- ❖ для добавления поля в область фильтра — **Фильтр** (Filter Area);
- ❖ для добавления поля в область деталей — **Детали** (Detail Data);
- ❖ для включения только сводных значений поля — **Данные** (Data Area).

Нажмите кнопку **Добавить в** (Add to). Если кнопка **Добавить в** (Add to) становится недоступной при выборе какой-либо области, то выделенное поле может быть неподходящим для использования в ней.

Удалить поле из любой области сводной таблицы можно, выделив его и выполнив команду **Удалить** (Remove) в контекстном меню или просто нажав клавишу <Delete>. Выделение добавленного в сводную таблицу поля выполняется щелчком мыши на его заголовке.

Поля, уже размещенные в сводной таблице, в списке полей выделяются полужирным шрифтом.

## Вычисление итоговых значений

Для подсчета итоговых значений щелкните на заголовке поля **Количество отгружено**, выделив таким образом это поле.

Код товара	Сумма "Количество отгружено"
T001	23
T002	25
T003	12
T005	40
T006	6
T007	10
T008	23
T009	13

Список полей сводной таблицы

- Итоги
  - Сумма "Количество отгружено"
- Код накладной
- Код склада
- Код товара
- Количество отгружено
- Сумма отгружено

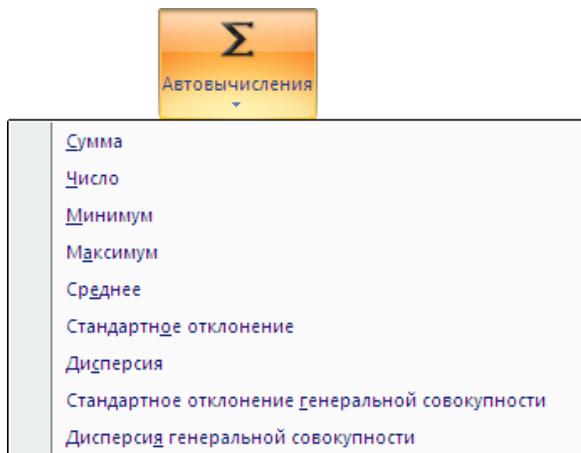
Рис. 6.4. Выполнение группировки по одному полю и вывод только итогов в сводной таблице

На вкладке ленты **Работа со сводными таблицами | Конструктор** (Pivot Table Tools | Design) в группе **Сервис** (Tools) нажмите кнопку **Автовычисления** (Auto-

Calc) и в открывшемся списке выберите статистическую функцию **Сумма** (Sum). Выполнение команды приводит к отображению как промежуточных, так и общих итогов. Для отображения только итогов нажмите на вкладке ленты в группе **Показать или скрыть** (Show/Hide) кнопку **без подробностей** (Hide Details) . Сводная таблица с выполненной группировкой по одному полю Код товара и выводом итоговых суммарных количеств для каждого из товаров представлена на рис. 6.4.

Список полей сводной таблицы после выполнения автovычисления дополняется группой **Итоги** (Totals), в которой представляются все внесенные в сводную таблицу итоговые поля. В примере группа содержит поле Сумма "Количество отгружено" (см. рис. 6.4). Для удаления итогового поля достаточно щелкнуть на нем и выполнить в контекстном меню команду **Удалить** (Delete). Итоговое поле удаляется и из списка полей и из сводной таблицы.

Список функций, которые могут использоваться при выборе команды **Автovычисления** (AutoCalc), представлен на рис. 6.5.



**Рис. 6.5. Список функций кнопки Автovычисления**

### ЗАМЕЧАНИЕ

Если первоначально для размещения поля выбрать не область деталей, а область данных, в сводную таблицу будут добавлены только итоговые значения, и нельзя будет просмотреть детали.

Суммирование количества различных товаров не имеет смысла, поэтому скроем значение общего итога. Для этого сначала выделим столбец **Код товара**, щелкнув на его заголовке. Затем щелкнем на вкладке ленты в группе **Сервис** (Tools) по доступной теперь кнопке **Итоги** (Subtotal) .

Для отображения данных только о нужных товарах щелкните на стрелке, расположенной в правой части поля Код товара, снимите флажок **Все** (All) и установите флажок у нужных товаров. Для отображения подробностей по одному из товаров щелкните по знаку "плюс" (+) в области значения с кодом товара. Значок "минус" (-) скроет эти подробности.

Пусть далее необходимо подсчитать общее количество каждого товара, отгруженного с каждого склада и со всех складов в целом. Изменим макет сводной таблицы, перетащив в область столбцов поле Код склада. Выполнив эту единственную операцию, получим сводную таблицу с суммарными итоговыми значениями количества для каждого из товаров по каждому из складов и по всем складам в целом. Значение общего итога по всем товарам по каждому из складов не имеет смысла, поэтому целесообразно скрыть его в данной сводной таблице. Здесь также можно отображать или скрывать подробности произведенных вычислений. По сути, в таком макете сводной таблицы будут отображены результаты группировки по двум полям: Код товара и Код склада (рис. 6.6).

Код склада ▾		C01	C02	C03	Общие итоги
Код товара ▾		Сумма "Количество"	Сумма "Количество"	Сумма "Количество"	Сумма "Количество"
T001	+	19			4
T002	+		25		
T003	+		9	3	12
T005	+	20	20		40
T006	+			6	6
T007	+	10			10
T008	+	23			23
T009	+		10	3	13

Рис. 6.6. Выполнение группировки по двум полям в сводной таблице

Чтобы в сводной таблице скрыть не задействованные области (на рис. 6.6 область фильтра) щелкните на кнопке **Зоны** (Drop Zones), размещенной на вкладке ленты **Работа со сводными таблицами | Конструктор** (Pivot Table Tools | Design) в группе **Показать или скрыть** (Show/Hide).

## Разработка сводной таблицы для запроса

Рассмотрим создание сводной таблицы для запроса **Отгрузка товаров**, представленного на рис. 6.7. В этом запросе записи таблицы ОТГРУЗКА дополняются данными из главных по отношению к ней таблиц ТОВАР и НАКЛАДНАЯ, а также из таблиц ДОГОВОР и ПОКУПАТЕЛЬ. В результате образуется виртуальная таблица запроса с полными сведениями об отгрузках.

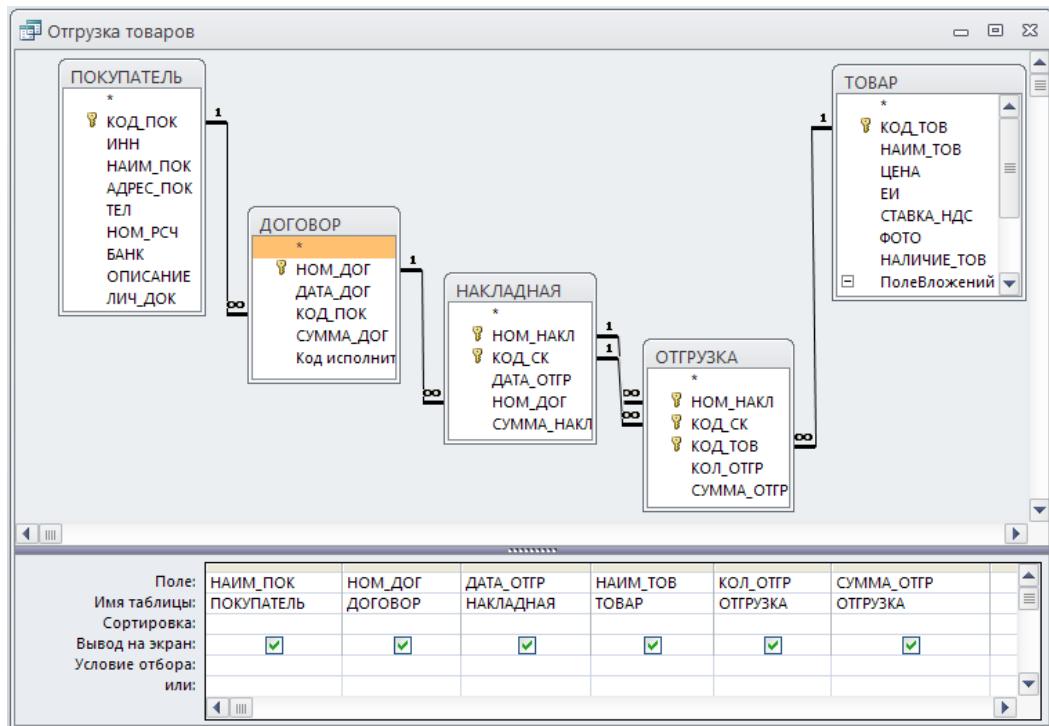


Рис. 6.7. Запрос на выборку для включения подробных сведений об отгрузках

Сводная таблица для запроса создается также как для таблицы базы данных. После перехода в режим сводной таблицы так же открываются макет и список полей, доступных для использования. Перетащим в макет поле Наименование товара для образования строк, поле Номер договора для образования столбцов.

Дата отгрузки по месяцам

		Номер договора			
		Д111	Д222	Д333	Общие итоги
Наименование товара	Сумма "Количество"				
	CD-ROM Panasonic IDE	24	16		40
DIMM 64М PC100	5		1	6	
FDD 3,5		15	10	25	
HDD Maxtor 20GB				12	
Зв. Карта Genius Live				13	
Монитор 17LG	12			23	
Принтер EPSON ST.A4	10			10	
СканерAcer	1	10	12	23	

Значение: 15  
Итог: Сумма "Количество отгружено"  
Компонент строки: FDD 3,5  
Компонент столбца: Д222  
Фильтр: Дата отгрузки по месяцам = Все

Рис. 6.8. Сводная таблица, источником данных которой является запрос Отгрузка товаров

В область фильтра перетащим поле Дата отгрузки по месяцам.

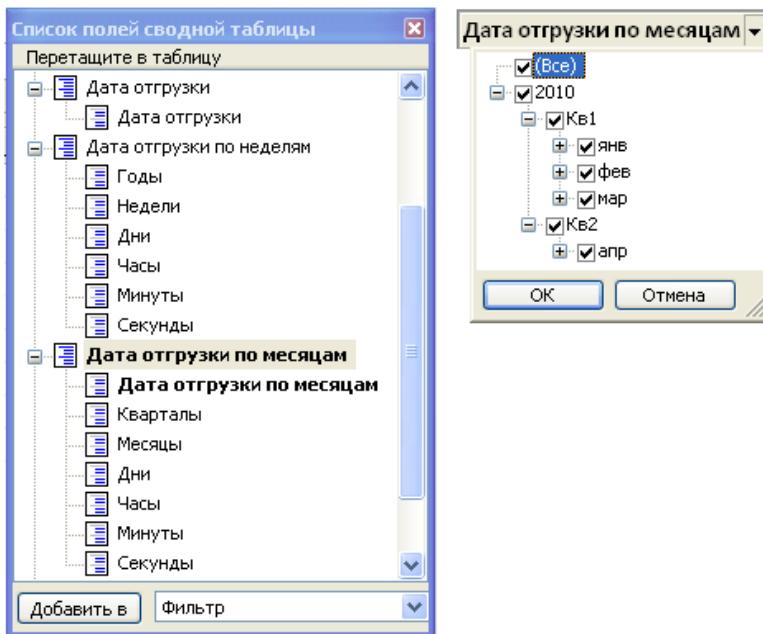
Поле Количество отгружено перетащим в область деталей для подсчета промежуточных и общих итогов для выбранных в строках, столбцах и фильтре значений.

Для вывода в сводной таблице итогов выполним команду **Автовычисления** (AutoCalc) и функцию **Сумма** (Sum). В созданной сводной таблице (рис. 6.8) можно проанализировать количество каждого их товаров, отгруженных по каждому из договоров и по всем договорам в целом в разрезе дат отгрузки. Можно произвести анализ отгрузок по заданному товару, по заданному договору, за заданный период.

В больших сводных таблицах удобно пользоваться подсказкой, высвечиваемой при подводе курсора к значению (см. рис. 6.8).

## Работа с датами в сводных таблицах

Поле Дата отгрузки имеет тип **Дата/время** (Data/Time) и поэтому в списке полей сводной таблицы представлено тремя строками: **Дата отгрузки**, **Дата отгрузки по неделям** (By Week) и **Дата отгрузки по месяцам** (By Month) (рис. 6.9).



**Рис. 6.9.** Отображение поля с датой в сводной таблице

В списке полей сводной таблицы можно выбрать любое измерение даты. Такое представление даты позволяет провести детальный анализ в любом временном измерении и не требует от пользователя никаких действий по преобразованию даты в тот или иной формат.

В сводной таблице **Отгрузка товаров** в качестве фильтра использовано поле Дата отгрузки по месяцам, при этом для анализа данных не сложно выбрать любой период времени. На рис. 6.9 для анализа выбраны все данные за 2010 год.

## Использование нескольких полей в областях сводной таблицы

В режиме сводной таблицы допускается использование нескольких полей в строках, столбцах, фильтре, области итогов и деталей. При этом поля, ближайшие к области итогов и деталей, называют *внутренними полями*. Остальные поля называют *внешними*. Пользователь имеет возможность менять местами внутренние и внешние поля. Перетащим поле с наименованием покупателя в область столбцов. Теперь в сводной таблице рассчитаны, помимо общих итогов по договорам, итоги по покупателям, которые могут иметь несколько договоров (рис. 6.10).

Наименование покупателя ▾ Номер договора ▾						
	Компьютер маркет		Перспектива		Общие итоги	
	Д111	Д222	Итоги	Д333	Итоги	+/-
Наименование товара ▾	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"	Сумма "Колич"
CD-ROM Panasonic IDE	+/-	4	16	20		20
DIMM 64M PC100	+/-	5		5	1	6
FDD 3,5	+/-		15	15		15
HDD Maxtor 20GB	+/-		8	8	3	11
Зв. Карта Genius Liv	+/-		10			13
Монитор 17LG	+/-	2	7			13
СканерAcer	+/-		10			22

Значение: 8  
Итог: Сумма "Количество отгружено"  
Компонент строки: HDD Maxtor 20GB  
Компонент столбца: Компьютер маркет - Итоги  
Фильтр: Дата отгрузки по месяцам = 2007

Рис. 6.10. Сводная таблица с двумя полями в области столбцов

Для вычисления стоимости отгруженного по договорам, покупателям и фирме в целом добавьте поле Сумма отгружено в область деталей сводной таблицы, поставив его после поля Количество отгруженного. Для вычисления итогов выполните команду **Автовычисления** (AutoCalc).

## Добавление полей в источник записей сводной таблицы

Чтобы добавить новые поля в источник записей сводной таблицы, откройте запрос в режиме конструктора и перетащите их из схемы данных в бланк запроса.

Если необходимо, добавьте новые таблицы в схему запроса. Открыв после этого запрос в режиме сводной таблицы, убедитесь, что ранее созданная сводная таблица сохранила прежний вид, а в списке полей отображены новые поля.

## Форматирование элементов сводной таблицы

Используя свойства элементов сводной таблицы (рис. 6.11), можно изменить ее формат, переименовать поля, выбрать шрифт, формат текста, числовых значений итоговых полей.

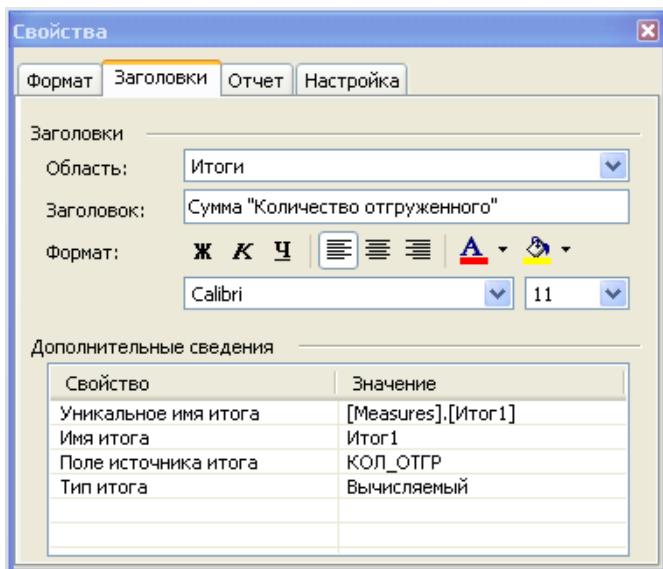


Рис. 6.11. Свойства элемента сводной таблицы

## Вычисляемые итоги и поля в сводной таблице

Пусть необходимо в сводной таблице показать стоимость отгруженного товара в рублях и в евро.

Удалим из сводной таблицы запроса **Отгрузка товаров** (см. рис. 6.10) поле Количество отгружено, размещенное в области деталей. Для этого выделим поле, щелкнув на его заголовке, и в контекстном меню выполним команду **Удалить** (Remove). Если детали скрыты, поле Количество отгружено не отображается в сводной таблице. Выполните команду **с подробностями** (Show Details), чтобы отобразить поле.

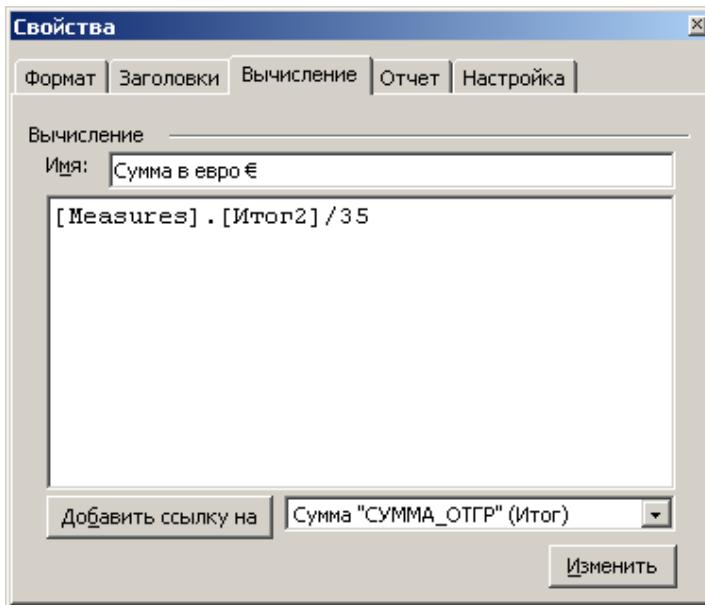
Для удаления итоговых значений количеств отгруженного выберем в списке полей соответствующую строку и в контекстном меню выполним команду **Удалить** (Delete).

Переместим из списка полей в область деталей сводной таблицы поле Сумма отгружено. Выполним для этого поля команду **Автовычисления** (AutoCalc).

Для расчета стоимости отгруженных товаров в евро добавим вычисляемое поле. На вкладке ленты **Работа со сводными таблицами | Конструктор** (Pivot Table Tools | Design) в группе **Сервис** (Tools) откроем список **Формулы** (Formulas) и выполним команду **Создание вычисляемого поля сведений...** (Create Calculated Detail Field...). В открывшемся окне свойств (рис. 6.12) на вкладке **Вычисление** (Calculation) в поле **Имя** (Name) заменим выводимое по умолчанию значение **Вычисляемое поле** (Calculated) на **Сумма в евро €**. Для формирования выражения [Сумма отгружено]/35 используем кнопку **Добавить ссылку на** (Insert Reference To). Щелкнем на кнопке **Изменить** (Change), и результат отобразится в списке полей сводной таблицы и в самой сводной таблице.

Дополним сводную таблицу рассчитанными по вычисляемому полю итоговыми значениями, выполнив команду **Автовычисления** (AutoCalc).

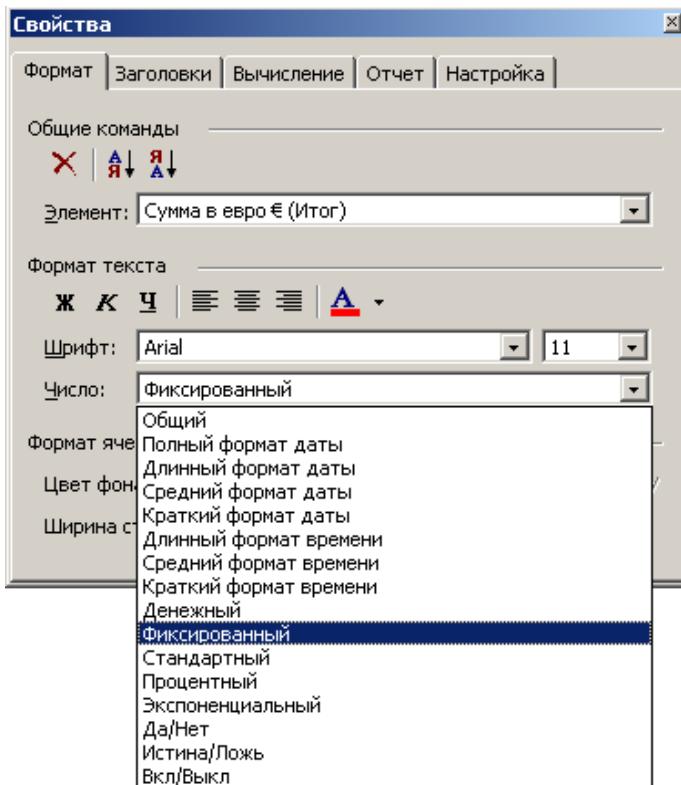
Можно, не добавляя вычисляемого поля, дополнить сводную таблицу вычисляемыми итоговыми значениями. Откроем список кнопки **Формулы** (Formulas) и выполним команду **Создание вычисляемого итога...** (Create Calculated Total...). В открывшемся окне свойств на вкладке **Вычисление** (Calculation) сформируем выражение на основе ранее вычисленных итогов для поля **Сумма отгружено** (рис. 6.12).



**Рис. 6.12.** Окно создания вычисляемого поля итогов

Изменим формат отображения вычисляемого итога. Для этого в свойствах поля на вкладке **Формат** (Format) выберем значение **Фиксированный** (Fixed) (рис. 6.13).

Изменить выражение вычисляемого поля или параметры можно в любое время, открыв окно свойств соответствующей командой в его контекстном меню.



**Рис. 6.13.** Окно определения формата вычисляемого итога

Предположим, что в таблице ОТГРУЗКА не заполнено или отсутствует поле Сумма отгружено. Вычислим в сводной таблице стоимость отгруженного товара и итоговые значения стоимости.

Удалим из области деталей и данных рассматриваемой сводной таблицы все поля. Откроем список **Формулы** (Formulas) и выполним команду **Создание вычисляемого поля сведений...** (Create Calculated Detail Field...). В открывшемся окне свойств на вкладке **Вычисление** (Calculation) дадим вычисляемому полю **Стоимость товара**.

Используя кнопку **Добавить ссылку на** (Insert Reference To), сформируем выражение `[ЦЕНА] * [КОЛ_ОТГР]`. Щелкнем на кнопке **Изменить** (Change). В сводной таблице отобразится созданное вычисляемое поле, а список полей дополнится новой строкой. Выберем формат **Денежный** (Currency). Команда **Автовычисления** (AutoCalc) позволит получить промежуточные и общие итоговые значения стоимости отгруженных товаров. Чтобы команда стала доступной, выделите поле **Стоимость товаров**.

Для вывода данных о стоимости товаров, отгруженных в заданном месяце, пометим в списке фильтра **Дата отгрузки по месяцам** (By Month), например, март. Для отображения данных только по покупателям, без разнесения сумм по договорам, щелкнем на значке "минус" (-) у названий покупателей. При этом подсчет итоговых значений по-прежнему будет вестись с учетом всех договоров. Если в сводной таблице не отобразились итоги по покупателям, щелкните на заголовке строки **Наименование товара** и затем на вкладке ленты кнопку **Итоги** (Subtotal).

Сводная таблица после выполнения всех перечисленных действий со скрытыми подробностями представлена на рис. 6.14.

Отгрузка товаров			
Дата отгрузки по месяцам			
мар			
Наименование покупателя	Номер договора	Общие итоги	
Компьютер маркет	Перспектива	+/-	
Сумма "Стоймость товара"	Сумма "Стоймость товара"	Сумма "Стоймость товара"	
CD-ROM Panasonic IDE	4 612,00р.		4 612,00р.
DIMM 64M PC100		360,00р.	360,00р.
HDD Maxtor 20GB		3 840,00р.	3 840,00р.
Зв. Карта Genius Liv		2 367,00р.	2 367,00р.
Монитор 17LG	7 000,00р.	2 000,00р.	9 000,00р.
СканерAcer	23 380,00р.		23 380,00р.
Общие итоги	34 992,00р.	8 567,00р.	43 559,00р.

Рис. 6.14. Сводная таблица с расчетом итоговых сумм по каждому товару и покупателю за заданный месяц

Отгрузка товаров			
Дата отгрузки по месяцам			
мар			
Наименование покупателя	Номер договора	Общие итоги	
Компьютер маркет	Перспектива	+/-	
Сумма "Стоймость товара"	Сумма "Стоймость товара"	Сумма "Стоймость товара"	
CD-ROM Panasonic IDE	100,00%		100,00%
DIMM 64M PC100		100,00%	100,00%
HDD Maxtor 20GB		100,00%	100,00%
Зв. Карта Genius Liv		100,00%	100,00%
Монитор 17LG	77,78%	22,22%	100,00%
СканерAcer	100,00%		100,00%
Общие итоги	80,33%	19,67%	100,00%

Рис. 6.15. Сводная таблица с итоговыми суммами в процентах от общих итогов строки

Пусть необходимо подсчитать, какой процент составляет стоимость товара, отгруженного покупателю, от общей стоимости отгруженного товара. Для решения этой задачи достаточно в сводной таблице **Отгрузка товаров** (см. рис. 6.14) щелкнуть на заголовке столбца **Сумма "Стоймость товара"**, а затем на вкладке ленты **Работа со сводными таблицами | Конструктор** (Pivot Table Tools | Design) в группе **Сервис** (Tools) открыть список кнопки **Форматы** (Show As) и выполнить команду **Процент итогов по строке** (Percent of Row Total). Сводная таблица с итоговыми значениями в процентах от общих итогов по строке представлена на рис. 6.15.

## Режим сводной диаграммы

Если вы построили сводную таблицу, то переход к диаграмме потребует от вас только выбора соответствующего режима.

Пусть запрос **Отгрузка товаров** в режиме сводной таблицы содержит в области строк поле **Наименование товара**, в области столбцов поле **Наименование покупателя и Номер договора**, в области фильтра **Дата отгрузки по месяцам**, в области данных **Стоймость товара** (см. рис. 6.15). Выберите в списках элементов полей строк, столбцов и фильтра значение **Все** (All). Для перехода из режима сводной таблицы в режим сводной диаграммы выберите его в списке кнопки **Режим** (View).

В рассматриваемом примере наименования товаров размещаются вдоль горизонтальной оси. Покупатели и договоры, по которым отгружался товар, представляются на диаграмме рядами. Каждый ряд имеет свой цвет. Категорию составляют элементы полей **Наименование покупателя - Номер договора** (ряды), отобранные для одного товара. По вертикальной оси указывается, на какую сумму отгружен товар покупателю по договору.

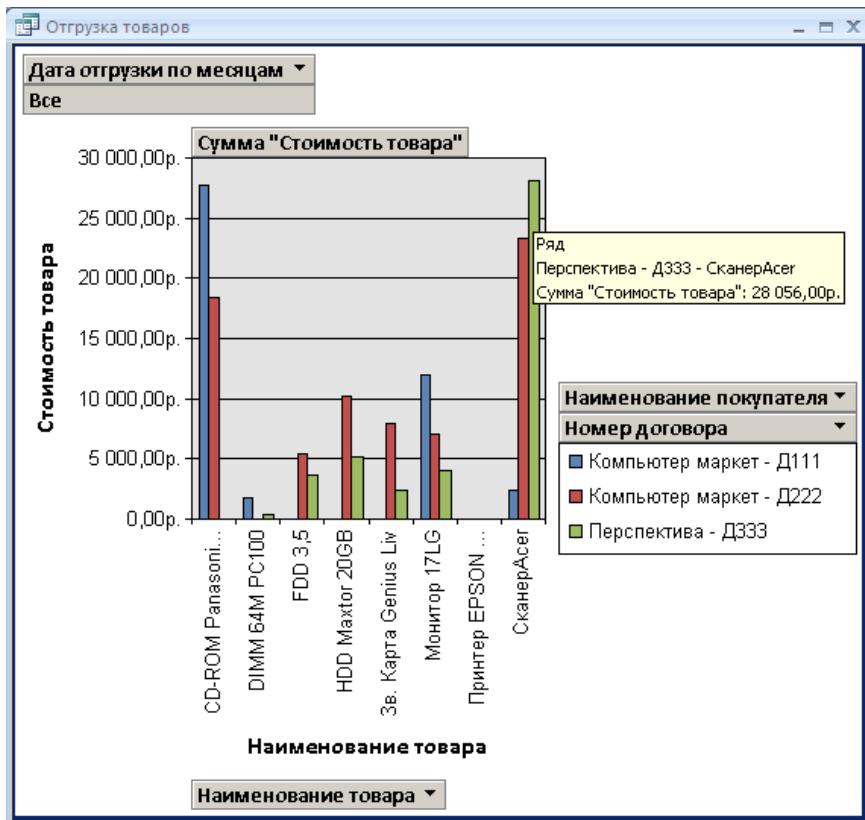
Щелчком мыши на кнопке **Легенда** (Legend), размещенной на вкладке ленты **Работа со сводными диаграммами | Конструктор** (Pivot Chart Tools | Design) в группе **Показать или скрыть** (Show/Hide), отобразим на диаграмме название каждого из рядов. В рассматриваемом примере в качестве названий использованы **Наименование покупателя - Номер договора**.

Для изменения названия оси щелкнем кнопкой мыши на ней и на кнопке **Страница свойств** (Property Sheet). В открывшемся окне выберем вкладку **Формат** (Format) и введем заголовок для оси значений — **Стоймость товара**, для оси категорий — **Наименование товара**. Выберем другие параметры форматирования.

Просмотрим масштаб оси значений. Выделим ось значений и откроем окно свойств. Перейдем на вкладку **Масштаб** (Scale). В разделе **Диапазон** (Range) можно изменить значение максимума, установив новое значение в соответствии с самым большим значением оси, выбрать другой шаг отображения основных единиц.

Аналогично можно изменить масштаб оси категорий: задав максимальное число выводимых наименований товара, шаг подписей.

Сводная диаграмма **Отгрузка товаров** после внесенных изменений примет вид, показанный на рис. 6.16.



**Рис. 6.16.** Запрос **Отгрузка товаров** в режиме сводной диаграммы

Подсказка содержит значения столбца (наименование покупателя — номер договора), строки (наименование товара) и суммарное значение стоимости товара для выбранного ряда.

Сводную диаграмму можно представлять в различном виде. Например, если необходимо, чтобы на диаграмме была видна суммарная стоимость отгруженного товара, в окне сводной диаграммы щелкните на кнопке вкладки ленты **Изменить тип диаграммы** (Change Chart Type) и выберите тип **Гистограмма** (Column) с накоплением, которая отражает вклад каждой категории в общую сумму (рис. 6.17). Если кнопка не отображена на вкладке, предварительно выделите область диаграммы, щелкнув в любом месте окна сводной диаграммы, не занятом каким-либо элементом.

Диаграмма примет вид, представленный на рис. 6.18. На этой диаграмме все ряды категорий размещены в одном столбце, что позволяет видеть общую стоимость отгруженного товара. Масштаб оси значений при этом увеличился до величины, равной 60 000.

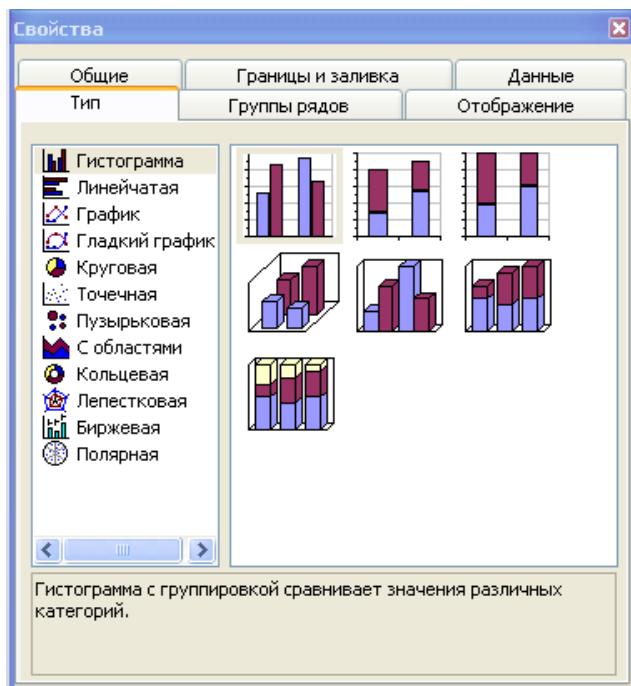


Рис. 6.17. Окно выбора типа диаграммы

Как и в сводной таблице, в диаграмме можно выбирать из списков полей строк, столбцов и фильтра значения, для которых нужно отобразить итоги на диаграмме. Макет сводной диаграммы также легко настраивается в интерактивном режиме. Сводная диаграмма и сводная таблица изменяются синхронно, т. е. все изменения, сделанные в сводной таблице, автоматически отображаются в сводной диаграмме и наоборот.

Поменяем местами поля в области столбцов и поле в области строк. Диаграмма примет вид, представленный на рис. 6.19. На этой диаграмме рядами категорий являются все виды отгруженных по договору товаров. Размещение рядов категории в одном столбце позволяет видеть общую стоимость товаров, отгруженных по договору. Диаграмма дополнена заголовком. Откройте свойства области диаграммы и на вкладке **Общие** (General) в разделе **Добавление** (Add) щелкните на значке **Добавить заголовок** (Add Title).

По умолчанию в режиме сводной диаграммы отображаются итоговые значения по данным из источника. Для отображения на диаграмме значений отдельных записей вместо итоговых значений по группам записей нужно в свойствах диаграммы на вкладке **Данные** (Data Details) установить флажок **Отображать детали** (Plot detail records). При этом на вкладке **Общие** (General) в поле со списком **Элемент** (Select) должен быть выбран элемент **Область диаграммы** (Cart Workspace). Установка и снятие флажка **Отображать детали** (Plot detail records) приводит к очистке элементов форматирования и частичной или полной очистке макета диаграммы и соответственно сводной таблицы.

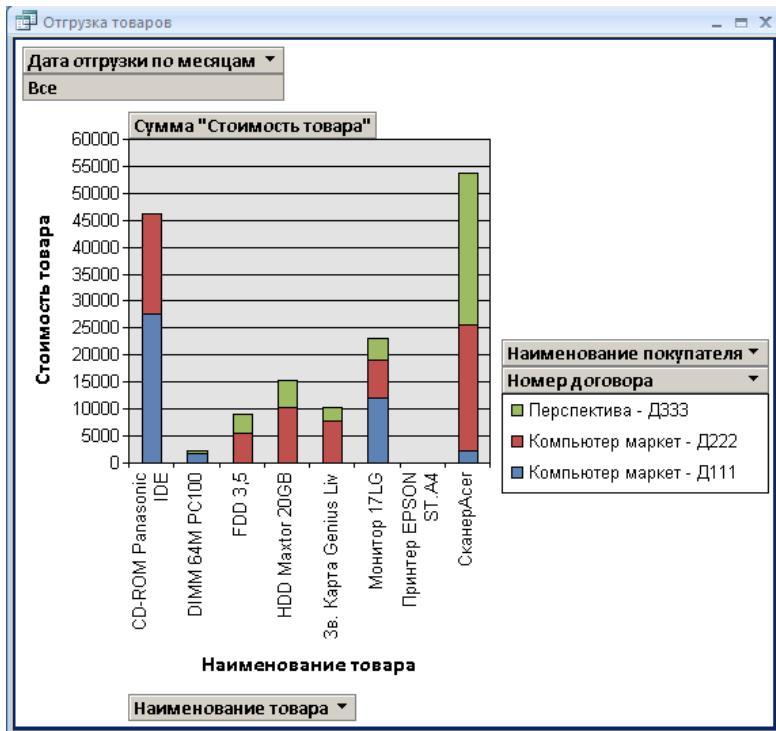


Рис. 6.18. Сводная диаграмма типа Гистограмма с накоплением сумм по товарам

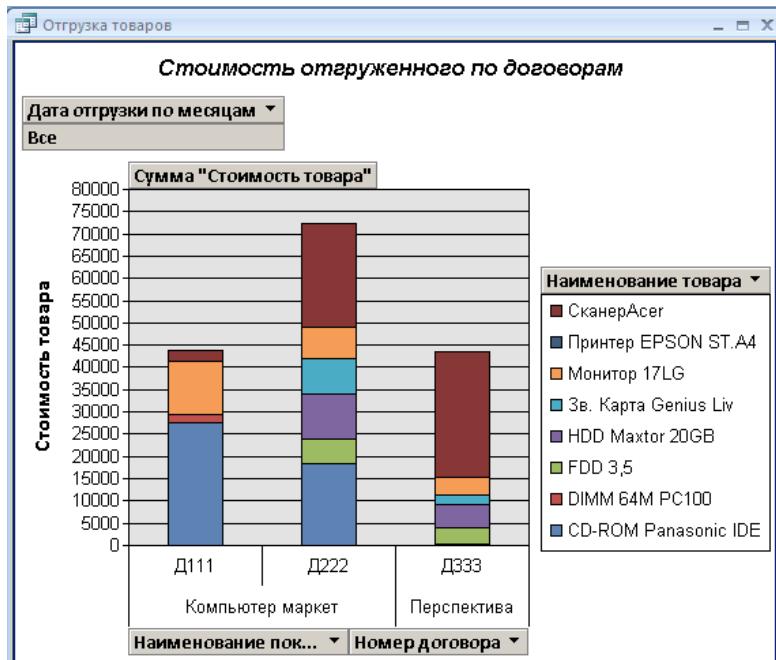


Рис. 6.19. Сводная диаграмма типа Гистограмма с накоплением сумм по договорам

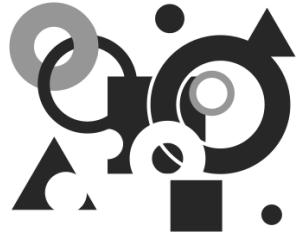
Сводная таблица и сводная диаграмма могут распечатываться как отчет с заголовком, датой выдачи и номерами страниц. Перед распечаткой они могут быть предварительно просмотрены на экране. Средства форматирования сводных таблиц и диаграмм позволяют придать отчету вид, соответствующий требованиям пользователя.

## Контрольные вопросы

1. Можно ли, используя сводную таблицу, изменять данные в ее источнике — объекте базы данных?
2. Какие объекты базы данных Access могут быть представлены в режиме сводной таблицы и диаграммы?
3. Выполнение какой команды позволяет перейти к созданию сводной таблицы?
4. Назовите области сводной таблицы, в которые могут быть размещены поля объекта?
5. Как открыть список полей, доступных для использования в сводной таблице?
6. Какая команда позволяет включить в сводную таблицу итоговые строки для поля деталей?
7. Возможно ли использование нескольких полей источника в одной области сводной таблицы?
8. Отразятся ли изменения в данных источника на ранее созданной сводной таблице или диаграмме?
9. Как поле с типом данных **Дата/время** (Date/Time) представляется в списке полей сводной таблицы?

## Ответы

1. Нет.
2. Таблицы, запросы, формы.
3. **Режим | Сводная таблица** (View | Pivot Table View) (объект должен быть открыт в режиме просмотра или конструктора).
4. Область строк, столбцов, фильтра, деталей или итогов.
5. Кнопкой **Список полей** (Field List) в группе **Показать или скрыть** (Show/Hide) на вкладке ленты **Работа со сводными таблицами | Конструктор** (Pivot Table Tools | Design).
6. **Автовычисления** (AutoCalc) для выделенной области деталей и **Итоги** (Subtotal) для выделенного столбца или строки.
7. Да.
8. Да.
9. Тремя строками, которые позволяют работать с временными интервалами, заданными датами, неделями или месяцами.



## ГЛАВА 7

# Отчеты

Средства Access по разработке отчетов предназначены для конструирования макета отчета, в соответствии с которым осуществляется вывод данных из определенного источника записей в виде выходного печатного документа. Эти средства позволяют создавать отчет любой сложности, обеспечивающий вывод взаимосвязанных данных из многих таблиц, их группировку, вычисления итоговых значений. При этом могут быть выполнены самые высокие требования к оформлению документа.

Перед началом конструирования пользователь должен спроектировать макет отчета. При этом определяются состав и содержание разделов отчета, размещение в нем значений, выводимых из полей таблиц (запросов) базы данных, и вычисляемых реквизитов, определяются поля, по которым нужно группировать данные. Для каждого уровня группировки определяются заголовки и примечания, вычисляемые итоговые значения. Кроме того, оформляются заголовки и подписи реквизитов отчета. Определяется также порядок вывода данных в отчете.

Отчет может создаваться с помощью мастера или в режиме конструктора отчетов. Во многих случаях удобно использовать мастера отчетов. Созданный мастером отчет можно доработать в режиме конструктора.

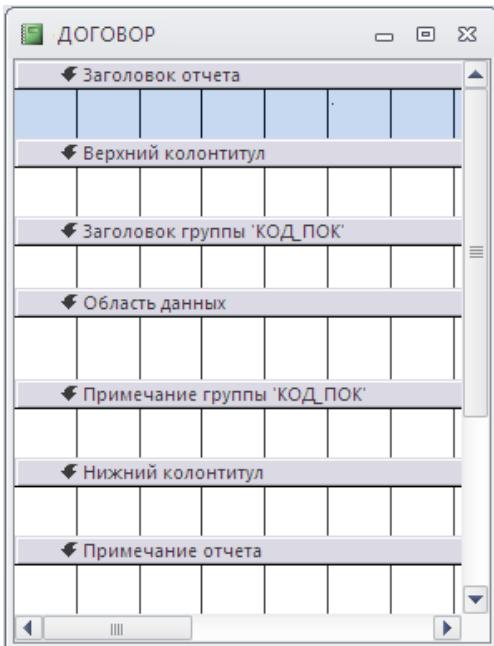
При необходимости вывода в отчете результатов решения задачи в качестве основы для отчета может быть использован многотабличный запрос, представляющий эти результаты. На запрос могут быть возложены наиболее сложные виды выборки и предварительной обработки данных. Разнообразные возможности конструктора отчетов позволяют легко структурировать и оформить полученные в запросе данные.

Новые средства Access 2010 позволяют создать профессионально оформленные отчеты не только с помощью мастера или конструктора, но и в режиме макета. При этом простыми средствами перетаскивания в отчет нужных полей из таблиц базы данных строится запрос — источник записей отчета, а использование свойств WYSIWYG позволяет сразу видеть, как именно будут выглядеть содержащиеся в нем данные на странице, и усовершенствовать макет.

## Основы конструирования отчетов

Чтобы правильно создавать отчеты, необходимо понимать назначение каждого его раздела. Например, от выбора раздела, в который будет помещен вычисляемый элемент управления, зависит способ вычисления результата.

Создание и изменение макета отчета осуществляется в расчете на структуру отчета. Пустые разделы отчета, открытого в режиме конструктора, показаны на рис. 7.1.



**Рис. 7.1.** Разделы отчета в режиме конструктора

Назначение каждого из разделов:

- ❖ **Заголовок отчета** (Report Header) обычно включает эмблему компании, название отчета, дату. Заголовок отображается перед верхним колонтитулом только один раз в начале отчета;
- ❖ **Верхний колонтитул** (Page Header) отображается вверху каждой страницы и используется в случае, когда нужно, чтобы название отчета и другая общая информация повторялись на каждой странице;
- ❖ **Заголовок группы** (<имя поля> Header) используется при группировке записей отчета для вывода названия группы и однократного отображения полей, по которым производится группировка. Отображается перед каждой новой группой записей. Например, если отчет сгруппирован по покупателям, в заголовке группы можно указать название покупателя, а также адрес, телефон и другие реквизиты. Допускается до 10 уровней группировки выводимых записей;
- ❖ **Область данных** (Detail) отображает записи из источника данных, составляющие основное содержание отчета;

- ❖ **Примечание группы** (<имя поля> Footer) используется для отображения итогов и другой сводной информации по группе в конце каждой группы записей. Если поместить в примечание группы вычисляемый элемент управления, использующий статистическую функцию `Sum`, сумма будет рассчитываться для текущей группы;
- ❖ **Нижний колонтитул** (Page Footer) применяется для нумерации страниц и отображения другой информации внизу каждой страницы;
- ❖ **Примечание отчета** (Report Footer) служит для отображения итогов и другой сводной информации по всему отчету один раз в конце отчета. Если в примечании отчета поместить вычисляемый элемент управления, использующий статистическую функцию `Sum`, сумма рассчитывается для всего отчета.

### **ЗАМЕЧАНИЕ**

В конструкторе **Примечание отчета** (Report Footer) находится под нижним колонтитулом. Однако при печати и в режиме предварительного просмотра оно помещается над нижним колонтитулом последней страницы.

Наличие или удаление заголовка и примечания отчета может быть выполнено командой контекстного меню отчета **Заголовок/примечание отчета** (Report Header/Footer). Кроме того, заголовок и примечание отчета вставляются при добавлении в отчет элементов управления **Заголовок** (Title), **Эмблема** (Logo) или **Дата и время** (Date & Time). Для включения или удаления колонтитулов можно воспользоваться командой контекстного меню **Колонтитулы страницы** (Page Header/Footer).

Заголовок и примечание группы добавляются в отчет при выполнении команды **Группировка** (Group & Sort) с последующим добавлением группы и выбором поля группировки и команды **Итоги** (Totals) на вкладке ленты **Инструменты конструктора отчетов | Конструктор** (Report Design Tools | Design) или **Работа с макетами отчетов | Формат** (Report Layout Tools | Format).

При создании отчета его разделы нужно заполнить элементами в соответствии с разработанным пользователем макетом отчета. В заголовок помещается текст из шапки макета отчета. В верхний и нижний колонтитулы обычно помещают надписи с поясняющим текстом, в том числе заголовки столбцов отчета, номера страниц. При определении содержания этих разделов следует исходить из требований к оформлению отдельных страниц отчета.

Поля таблиц базы данных или запросов с неповторяющимися значениями размещаются в области данных, которой можно придать вид табличной части отчета. Поля с повторяющимися значениями, по которым производится группировка записей, целесообразно размещать в заголовке группы. Здесь же отображаются данные, которые позволяют идентифицировать группу.

Элементами разделов отчета, кроме полей таблиц или запросов, на которых строится отчет, являются также тексты подписей, внедряемые объекты, линии, прямоугольники и т. п.

Для каждого элемента, а также раздела и отчета в целом могут быть уточнены свойства. Технология размещения элементов и определения их свойств практически такая же, как и при разработке форм.

Инструменты разработки отчетов представлены на вкладках лент, которые связаны с режимами изменения и просмотра отчетов, и появляются при переходе из одного режима в другой.

В Access существуют два представления, в которых можно вносить изменения в отчет: режим макета и режим конструктора. Режим макета является наиболее удобным для внесения изменений в отчет, поскольку пользователь сразу видит данные отчета. В этом режиме предусмотрено большинство инструментов, необходимых для его настройки. В нем можно изменить ширину столбцов, поменять их местами, добавить или изменить уровни группировки и итоги. Можно также разместить в макете отчета новые поля, а также задать свойства отчета и элементов управления.

В режиме конструктора отображаются разделы отчета и предусмотрены дополнительные инструменты и возможности разработки. Переходите в режим конструктора, если не удается выполнить изменения в режиме макета. В определенных случаях в Access отображается сообщение о том, что для внесения изменений следует переключиться в режим конструктора.

Просматривать отчет можно в режимах **Представление отчета** (Report View), **Предварительный просмотр** (Print Preview) или **Макет** (Layout View). В режиме **Представление отчета** (Report View) можно отфильтровать данные для отображения только заданных строк, найти нужные данные, скопировать текст отчета или его часть в буфер обмена. Режим предварительного просмотра предназначен для просмотра отчета перед печатью. В этом режиме можно увеличивать масштаб для просмотра деталей или уменьшать его для проверки размещения данных на странице, изменить параметры страницы. Режим макета позволяет, просматривая данные отчета, изменять его макет.

## Однотабличные отчеты

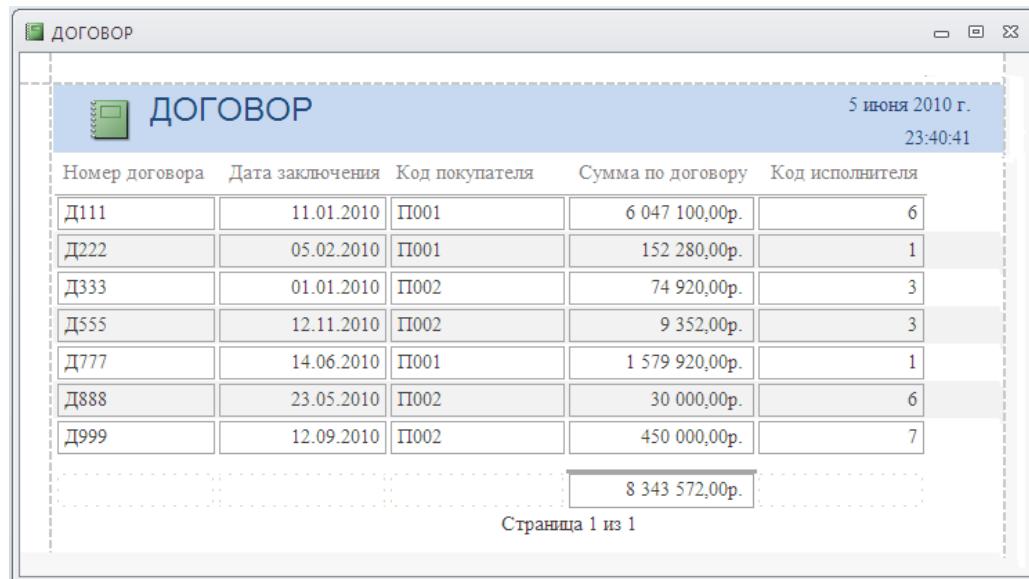
Наиболее простым способом создания отчета на основе таблицы или запроса является использование средства **Отчет** (Report). С помощью этого средства отчет формируется без диалога с пользователем и выводит все поля выбранного источника. Рассмотрим создание однотабличного отчета о договорах покупателей фирмы.

Пусть в результате проектирования макета отчета **Договоры покупателей фирмы** определены следующие требования к отчету:

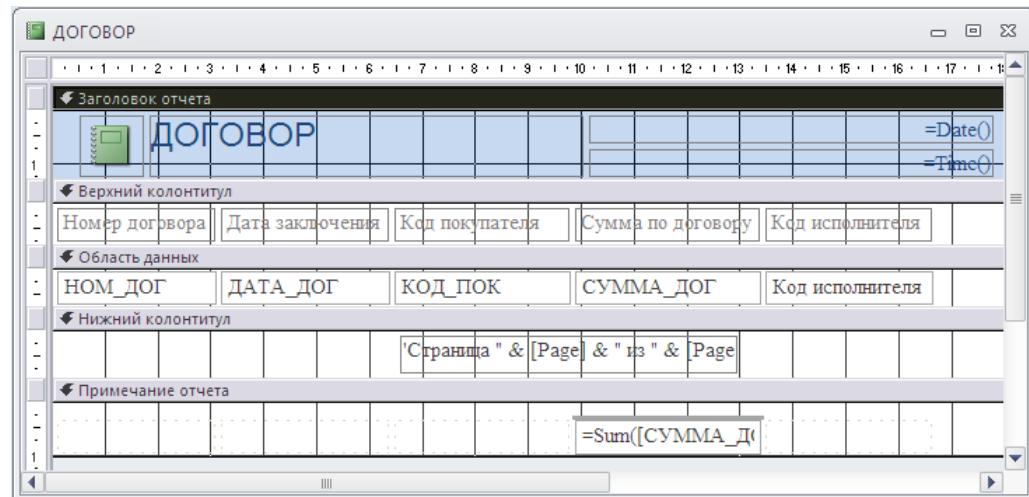
- ❖ в отчете должны последовательно выводиться со своими заголовками списки договоров для каждого покупателя;
- ❖ список договоров каждого покупателя должен начинаться с новой страницы;
- ❖ записи о договорах должны выводиться в порядке возрастания их номеров;
- ❖ по каждому покупателю и фирме в целом должны выводиться итоговые суммы.

Выберем в области переходов таблицу ДОГОВОР, данные из которой будут источником записей отчета. Чтобы сразу перейти к созданию нового отчета, на вкладке ленты **Создание** (Create) в группе **Отчеты** (Reports) выполним команду **Отчет** (Report).

Access создаст отчет и отобразит его в режиме макета (рис. 7.2). В отчете будут представлены все записи таблицы ДОГОВОР. Размещение полей таблицы — источника записей отчета — в разделах отчета представлено на рис. 7.3.



**Рис. 7.2.** Отчет, созданный соответствующей командой, в режиме макета



**Рис. 7.3.** Разделы однотабличного отчета в режиме конструктора

В области данных отчета размещены все поля таблицы. Это обеспечивает вывод в отчете всех записей в табличном виде. Размещение заголовков столбцов в верхнем колонтитуле обеспечивает вывод их на каждой странице отчета. Для добавления номера страницы в нижнем колонтитуле в вычисляемом элементе сформировано выражение = "Страница " & [Page] & " из " & [Pages].

В заголовке отчета размещены: эмблема, название отчета, текущая дата и время. Для добавления в отчет текущей даты использована встроенная функция =Date() из категории **Дата/время** (Date/Time). В свойствах этого элемента на вкладке **Макет** (Format) в строке **Формат поля** (Format) установлен формат отображения даты — **Длинный формат даты** (Long Date). Для добавления в отчет текущего времени использована встроенная функция =Time().

В примечании отчета в вычисляемом элементе управления записана статистическая функция =Sum([СУММА\_ДОГ]), рассчитывающая стоимость всех договоров, представленных в отчете.

## Доработка отчета в режиме макета

В режиме макета (см. рис. 7.2) легко привести созданный отчет в соответствие с заданными требованиями. Ориентируясь на фактические данные отчета, можно отрегулировать ширину столбцов, высоту строк, изменить их порядок, добавить уровни группировки и итоговые значения. Можно добавить в отчет новые поля, а также изменить свойства отчета и входящих в него элементов управления.

Измените название отчета на ДОГОВОРЫ ФИРМЫ. Для этого выполните двойной щелчок на нем и введите новое название. Для замены эмблемы удалите старую и, щелкнув на значке **Эмблема** (Logo) на вкладке ленты **Формат** (Format), выберите нужный рисунок. Эмблема будет добавлена в то же место заголовка отчета. При желании ее можно перетащить в другое место и изменить размер.

Элементы управления отчета "Договор" организованы в табличный макет. Табличные макеты элементов всегда охватывают несколько разделов отчета; в данном случае надписи расположены в верхнем колонтитуле выше области данных, в которой расположены поля данных, а в примечании отчета расположен вычисляемый элемент, также включенный в этот макет. Выделите макет отчета и измените размер шрифта, выбрав его на вкладке **Главная** (Home) в группе **Шрифт** (Font). Все элементы макета изменят шрифт на выбранный.

Для изменения размеров элементов управления макета можно выделить весь макет, и тогда все его элементы будут менять размер и местоположение. Можно выделить отдельный столбец макета и изменять ширину только этого столбца и перемещать внутри макета, перетаскивая его на нужное место. При перетаскивании вертикальная полоска указывает, где оно будет помещено, если отпустить кнопку мыши. Аналогично можно изменить высоту строки.

При необходимости изменять положение и размер отдельных элементов макета их нужно удалить из макета и создать новый дополнительный макет. Выделите элемент с общей суммой по договорам и выполните в контекстном меню команду

**Удалить** макет (Remove Layout). Теперь стало возможным изменение ширины только этого элемента.

Для выделения раздела отчета щелкните слева от него, и он будет обрамлен жирной линией. Воспользовавшись кнопкой **Цвет заливки/фона** (Fill/Back Color) на вкладке ленты **Главная** (Home), выберите из коллекции нужный цвет.

Используйте условное форматирование для изменения внешнего вида значения в поле **Сумма по договору** в зависимости от одного или нескольких условий. Измените цвет денежных значений на красный, когда они оказываются выше заданной величины, например 450 000 руб. Для этого щелкните в столбце **Сумма по договору** и выполните команду **Условное форматирование** (Conditional Formatting) на вкладке ленты **Формат** (Format) в группе **Форматирование элемента управления** (Control Formatting). В окне **Диспетчер правил условного форматирования** (Conditional Formatting Rules Manager) в поле **Показать правила форматирования для** (Show formatting rules for) будет указано имя поля СУММА\_ДОГ, для которого нужно создать правила. Щелкните на кнопке **Создать правило** (New Rule). В окне **Новое правило форматирования** (New Formatting Rule) измените описание правила, оставив в первом поле **Значение поля** (Field Value Is), выбрав во втором **больше** (greater than) и в третьем введя значение 400000. В области **Просмотр** (Preview) выберите формат, используемый при выполнении условия для значений поля жирный шрифт и красный цвет.

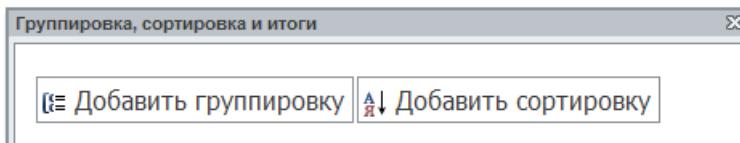
Для изменения внешнего вида отчета в целом используйте богатый выбор коллекции **Темы** (Themes) на вкладке ленты **Конструктор** (Design).

Для изменения параметров страницы используйте команды на соответствующей вкладке ленты. Пунктирной линией в отчете отмечена граница полей страницы. Отображение этой линии регулируется кнопкой **Показать поля** (Show Margins) на вкладке ленты **Параметры страницы** (Page Setup). Для выбора размеров полей страницы может быть использована коллекция, отображаемая при нажатии кнопки **Поля** (Margins). На этой же вкладке можно выбрать размер бумаги, ориентацию страницы и ряд других параметров.

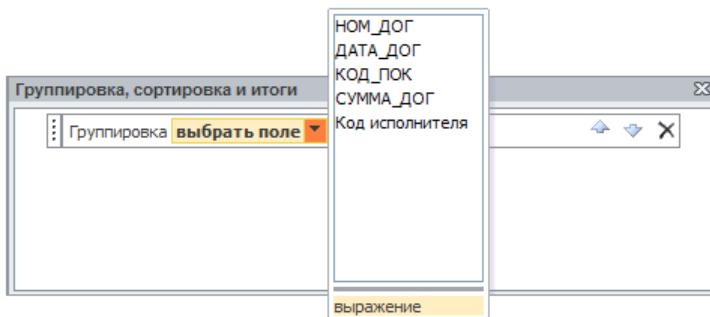
## Группировка и сортировка данных отчета

Для выполнения группировки и сортировки данных, отображаемых в отчете, предназначена область **Группировка, сортировка и итоги** (Group, Sort, and Total). Если область не отображена, выполните команду **Группировка** (Group & Sort) на вкладке ленты **Конструктор** (Design). Первоначально в отчете, созданном с помощью средства **Отчет** (Report), нет ни группировки, ни сортировки и область содержит лишь кнопки **Добавить группировку** (Add a group) и **Добавить сортировку** (Add a sort) (рис. 7.4).

Выполните группировку данных отчета по полю КОД\_ПОК (код покупателя). Щелкните на кнопке **Добавить группировку** (Add a group). В области **Группировка, сортировка и итоги** (Group, Sort, and Total) появится новая строка и будет отображен список доступных полей (рис. 7.5).



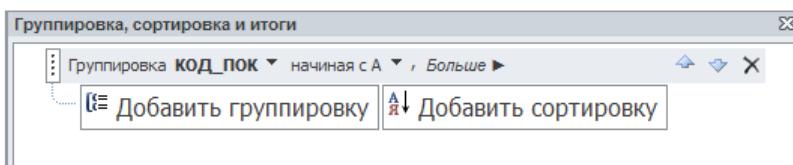
**Рис. 7.4.** Область управления группировкой и сортировкой в отчете



**Рис. 7.5.** Выбор поля группировки в отчете

Щелкните для группировки в поле КОД\_ПОК, вид отчета немедленно изменится. Поле группировки КОД\_ПОК (код покупателя) переместится на первое место и в отчет добавится уровень группировки. В отчете можно определить до 10 уровней группировки и сортировки.

В области **Группировка, сортировка и итоги** (Group, Sort, and Total) появится строка с указанием на группировку первого уровня по полю КОД\_ПОК. Для каждого уровня группировки существует ряд параметров. Для отображения всех параметров щелкните **Больше** (More) на уровне, который нужно изменить (рис. 7.6).



**Рис. 7.6.** Стока группировки первого уровня по полю КОД\_ПОК

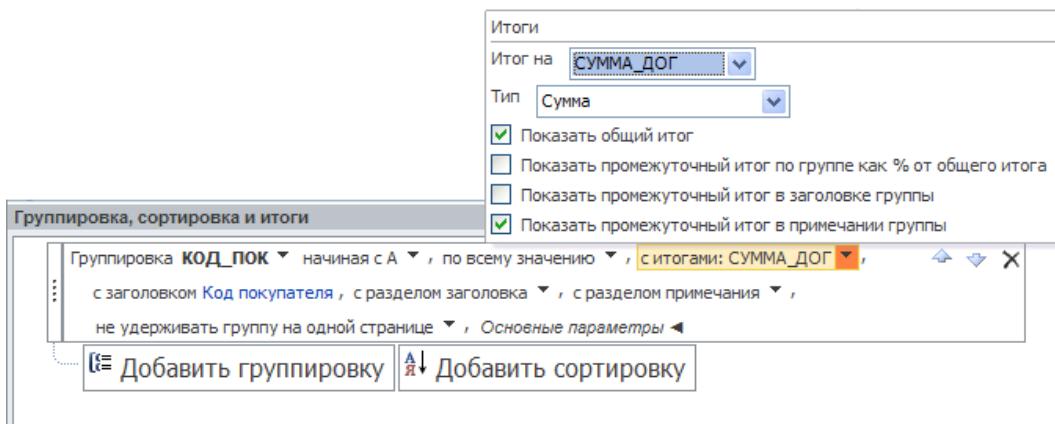
Кнопки, расположенные в строке группировки справа, позволяют повысить (стрелка вверх), понизить (стрелка вниз) или удалить уровень группировки.

Для добавления в отчет итоговых сумм по каждому покупателю и общей суммы нажмите кнопку **Больше** (More) и откройте список значений параметра **с итогами: СУММА\_ДОГ** (with SUMMA\_DOG totaled). В поле списка **Итог на** (Total On) выберите поле СУММА\_ДОГ, по которому нужно рассчитать итоговые значения. Отметьте флажок **Показать общий итог** (Show Grand Total) и определите ме-

сто отображения итоговых значений по каждому покупателю выбором флашка **Показать в примечании группы** (Show in group footer) (см. рис. 7.7).

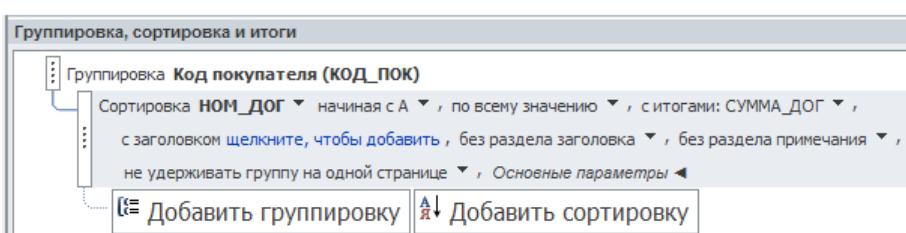
Можно подвести итоги для нескольких полей группы. Для этого следует выбирать каждое поле в раскрывающемся списке **Итог на** (Total On) и нужные параметры. Для закрытия всплывающего окна **Итоги** (Totals) щелкните по любому месту за его пределами.

По умолчанию интервал группировки выбирается **по всему значению** (by entire value) (рис. 7.7). Этот параметр определяет способ группировки записей. Например, можно выполнить группировку по первой букве текстового поля, так что будут объединены в одну группу элементы, начинающиеся с буквы А, затем — с Б и т. д. Для поля даты можно задать группировку по дню, неделе, месяцу, кварталу или по специальному интервалу.



**Рис. 7.7.** Страна группировки  
с открытым списком параметра **с итогами**: СУММА\_ДОГ

Параметр **с заголовоком** (with title) позволяет изменить заголовок поля группировки. Чтобы изменить заголовок, щелкните выделенный синим цветом текст за словами **с заголовоком** (with title). В окне **Область ввода** (Zoom) введите новый заголовок и нажмите кнопку **OK**.



**Рис. 7.8.** Страна сортировки по полю НОМ\_ДОГ

**ДОГОВОРЫ ФИРМЫ**

6 июня 2010 г.  
14:10:33

Код покупателя Номер договора Дата заключения Сумма по договору Код исполнителя

П001	Д111	11.01.2010	6 047 100,00р.	6
	Д222	05.02.2010	152 280,00р.	1
	Д777	14.06.2010	1 579 920,00р.	1

7 779 300,00р.

П002	Д333	01.01.2010	74 920,00р.	3
	Д555	12.11.2010	9 352,00р.	3
	Д888	23.05.2010	30 000,00р.	6
	Д999	12.09.2010	450 000,00р.	7

564 272,00р.

8 343 572,00р.

Страница 1 из 1

**Рис. 7.9.** Отчет с группировкой по полю Код покупателя и итогами для каждого покупателя

**ДОГОВОРЫ ФИРМЫ**

6 июня 2010 г.  
14:25:25

Код покупателя Номер договора Дата заключения Сумма по договору Код исполнителя

П001	7 779 300,00р.			
П002	564 272,00р.			
	8 343 572,00р.			

Страница 1 из 1

**Рис. 7.10.** Отчет только с итоговыми данными по каждому покупателю и фирме в целом

Для установления в группе сортировки по номеру договора щелкните на кнопке **Добавить сортировку** (Add a sort) и в списке доступных полей на строке

НОМ\_ДОГ. Строки отчета немедленно отсортируются, а в области **Группировка, сортировка и итоги** (Group, Sort, and Total) появится новая строка с указанием на сортировку по полю НОМ\_ДОГ (рис. 7.8).

В качестве значения параметра **Порядок сортировки** по умолчанию выбирается **начиняя с А** (With A on top). Для изменения значения выберите его из раскрывающегося списка.

Отчет в режиме макета после внесения перечисленных изменений представлен на рис. 7.9.

Если в отчете необходимо отобразить только итоговые данные (рис. 7.10), выполните команду **Без подробностей** (Hide Details) на вкладке ленты **Конструктор** (Design) в группе **Группировка и итоги** (Grouping & Totals). Повторное выполнение команды **Без подробностей** (Hide Details) восстанавливает строки отчета, поскольку записи только скрываются, а элементы управления в скрытом разделе не удаляются.

## Просмотр и печать отчета

Просмотр отчета возможен в одном из трех режимов:

- ❖ режим **Представление отчета** (Report View) предназначен для работы с отображаемым на экране отчетом и обеспечивает отбор записей по заданным условиям, поиск данных по образцу, копирование части данных в буфер обмена;
- ❖ режим **Предварительный просмотр** (Print Preview) показывает, в каком виде отчет будет выведен на печать, и обеспечивает необходимую настройку параметров страницы;
- ❖ **Режим макета** (Layout View), как было показано в предыдущем разделе, обеспечивает просмотр отчета при одновременной настройке макета отчета.

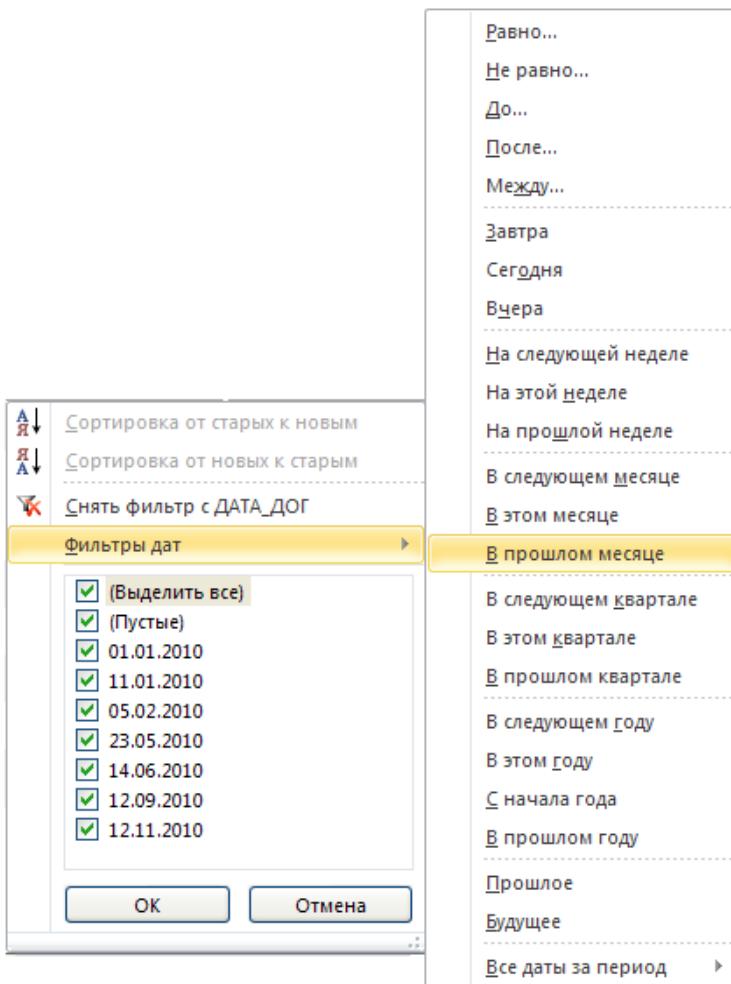
### Представление отчета

Для перехода в режим **Представление отчета** (Report View) выберите его из списка кнопки **Режим** (View) на вкладке ленты **Главная** (Home). Если отчет закрыт, выберите его в области переходов и выполните двойной щелчок или в контекстном меню выполните команду **Открыть** (Open). В режиме **Представление отчета** (Report View) строки данных не разбиты на страницы и отображаются в сплошной ленте. Для этого режима не существует специальной вкладки на ленте, и доступные команды представлены на вкладке **Главная** (Home).

В этом режиме, так же как в режиме макета, для отбора нужных записей в отчете можно использовать фильтр. Фильтр позволяет изменить отображаемые в представлении данные отчета, не изменения макет. После применения фильтра в представление включаются только те записи, которые удовлетворяют заданным условиям отбора. Остальные записи будут скрыты до тех пор, пока фильтр не будет удален.

Например, чтобы отобрать все договоры, заключенные в заданном периоде, можно применить фильтр. Установите курсор в столбце с датой заключения до-

говора. Выполните команду **Фильтр** (Filter) на вкладке ленты **Главная** (Home) в группе **Сортировка и фильтр** (Sort & Filter). В открывшемся окне откройте список **Фильтры дат** (Date Filters) и щелкните на строке **В прошлом месяце** (Last Month) (рис. 7.11). В отчете отобразятся только те договоры, которые были заключены в указанном периоде, и только для этих договоров будут рассчитаны итоговые значения.



**Рис. 7.11.** Фильтр для столбца с датами

Щелкните в списке **Фильтры дат** (Date Filters) на строке **После...** (After...) и откроется окно **Настраиваемый фильтр** (Custom Filter), в котором можно задать дату для отбора договоров, заключенных после ее наступления (рис. 7.12).

Для задания в условии отбора диапазона дат (рис. 7.13) используйте команду **Между...** (Between...) из списка кнопки **Фильтры дат** (Date Filters) (см. рис. 7.11).

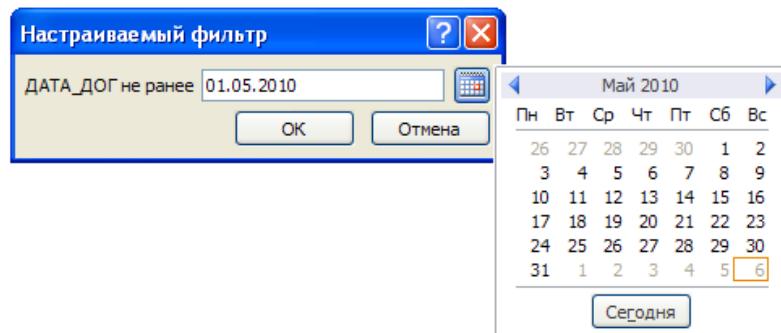


Рис. 7.12. Настраиваемый фильтр для столбца с датами



Рис. 7.13. Выбор диапазона дат для условия отбора строк отчета

Чтобы вернуться к отображению всех строк отчета, удалите фильтр, выполнив команду **Фильтр** (Toggle Filter) с подсказкой **Удалить фильтр** (Remove Filter) на вкладке **Главная** (Home) в группе **Сортировка и фильтр** (Sort & Filter). После этого подсказка команды получает имя **Применить фильтр** (Toggle Filter). Выполнение этой команды позволит снова задействовать фильтр. Для окончательного удаления фильтра выполните команду **Очистить все фильтры** (Clear All Filters) из списка кнопки **Дополнительно** (Advanced)).

Для изменения, сохранения и загрузки ранее сохраненных фильтров используйте команду **Расширенный фильтр** (Advanced) из списка кнопки **Дополнительно** (Advanced)).

В зависимости от типа данных в выбранном столбце доступны различные фильтры. Установите курсор в столбец с суммой договора и в окне фильтра вместо **Фильтры дат** (Date Filters) отобразятся **Числовые фильтры** (Number Filters). Для столбца с текстом отобразятся **Текстовые фильтры** (Text Filters).

Таким образом, использование фильтров позволяет, не создавая новых более специфичных отчетов, выполнять анализ данных, просматривать наборы записей, сформированные при различных условиях отбора.

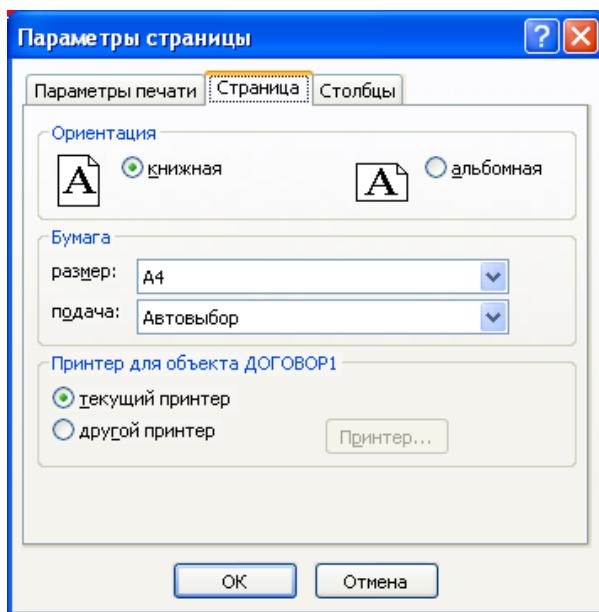
## Предварительный просмотр

Для предварительного просмотра страницы отчета и внесения в нее изменений перед печатью предназначен соответствующий режим. Предварительный просмотр позволяет убедиться, что отчет будет напечатан правильно. Переход из режима макета в режим предварительного просмотра отчета осуществляется с помощью

кнопки **Режим** (View) на вкладке ленты **Главная** (Home). При этом все вкладки ленты автоматически заменяются на вкладку **Предварительный просмотр** (Print Preview).

Для просмотра ранее созданного отчета выберите в области переходов нужный отчет и в контекстном меню выполните команду **Предварительный просмотр** (Print Preview). Отчет отобразится на экране таким, каким он будет напечатан. Для перехода от одной страницы отчета к другой можно использовать стандартное поле номера страницы в левом нижнем углу окна отчета.

Прежде чем переходить к печати отчета, необходимо уточнить параметры его страниц. Команда **Параметры страницы** (Page Setup), доступная в режиме предварительного просмотра отчета в группе **Разметка страницы** (Page Layout), открывает диалоговое окно **Параметры страницы** (Page Setup) (рис. 7.14), в котором можно выбрать принтер, задать формат и ориентацию (книжная, альбомная) бумаги, размер полей (вкладка **Параметры печати** (Print Options)), а также число, размер и макет столбцов. Эти же параметры представлены в группе **Размер страницы** (Page Size) отдельными кнопками. Кнопки **Поля** (Margins) и **Размер** (Size) открывают наглядные коллекции возможных вариантов полей страницы и размера бумаги.



**Рис. 7.14.** Выбор параметров страницы для печати отчета

### ЗАМЕЧАНИЕ

Окно **Параметры страницы** (Page Setup) открывается, только если на компьютере установлен локальный или сетевой принтер. Причем установленный принтер может быть и недоступен. В противном случае выводится сообщение о необходимости установить его.

В примере выбрана бумага формата А4, книжная ориентация страниц отчета и использование принтера по умолчанию. Если для печати отчета нужно выбрать

другой доступный принтер, отметьте соответствующий переключатель. В открывшемся окне выберите принтер из доступных принтеров в раскрывающемся списке **Имя** (Name) (рис. 7.15) и установите необходимые свойства выбранного принтера.

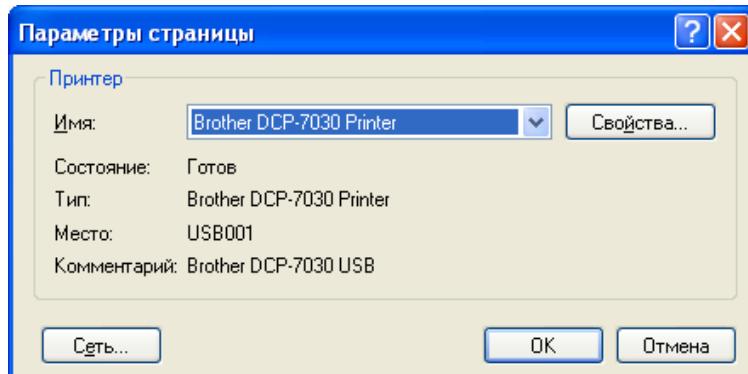


Рис. 7.15. Выбор принтера для печати отчета

Access сохраняет заданные параметры страницы вместе с отчетом.

Размещение разделов отчета на страницах регулируется их свойствами. Для вывода отчета по каждой группе с новой страницы в свойствах заголовка группы на вкладке **Макет** (Format) в строке **Конец страницы** (Force New Page) выбирается значение **До раздела** (Before Section). Сведения, которые необходимо отображать на каждой странице, размещаются в верхнем и нижнем колонтитулах. Для вывода заголовка отчета на отдельной странице в его свойстве **Конец страницы** (Force New Page) выбирается значение **После раздела** (After Section).

Если в рассматриваемом примере требуется печатать отчет для каждого покупателя на отдельной странице, для заголовка группы КОД\_ПОК установите в свойстве макета **Конец страницы** (Force New Page) значение **До раздела** (Before Section).

## Печать отчета

Для вывода отчета на печать нужно в режиме предварительного просмотра нажать кнопку **Печать** (Print). В диалоговом окне печати (рис. 7.16) можно выбрать принтер, на который будет отправлен отчет, настроить его свойства, дополнительно проверить и при необходимости настроить параметры страницы, выбрать для печати отдельные страницы отчета, распечатать заданное число копий, вывести отчет в файл, который должен распечатываться в другое время. Установив нужные для печати параметры, отчет можно отправить в очередь печати выбранного принтера.

Отчет печатается с использованием настроек, заданных в диалоговом окне **Печать** (Print).

Чтобы напечатать отчет, его не обязательно открывать. Если нет необходимости проверить отчет, подправить его макет или убедиться, что будут напечатаны

нужные данные, выберите отчет, который нужно напечатать, в области переходов. Нажмите кнопку **Файл** (File), а затем выберите пункт **Печать** (Print). Будет предоставлена возможность быстро напечатать отчет на используемом по умолчанию принтере без внесения каких-либо изменений или отобразить диалоговое окно **Печать** (Print) для выбора принтера и параметров печати или перейти к предварительному просмотру отчета и внесению изменений.

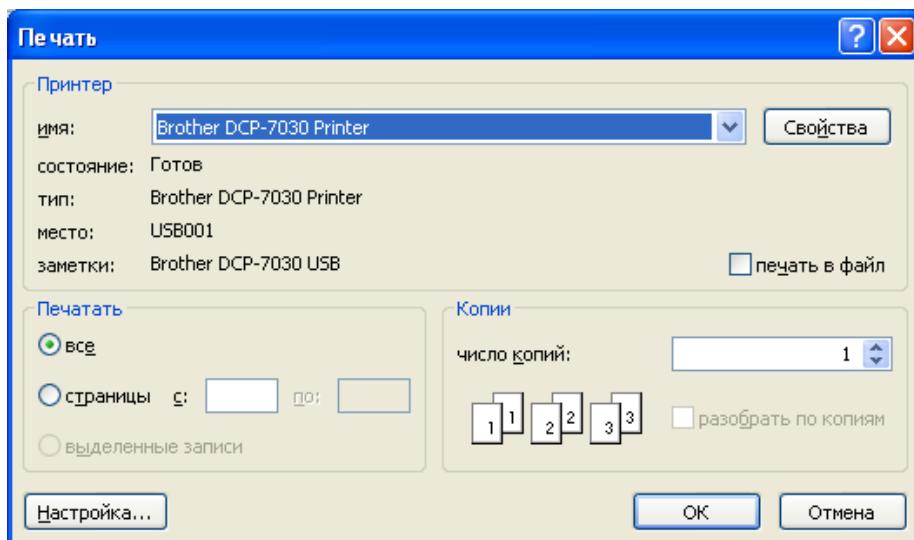


Рис. 7.16. Диалоговое окно печать отчета

## Изменение источника записей отчета

В режиме макета Access обеспечивает не только форматирование, но и изменение источника записей отчета простыми средствами добавления полей из списка всех таблиц базы данных.

Пусть необходимо в отчете о договорах покупателей вывести подробные сведения о покупателе.

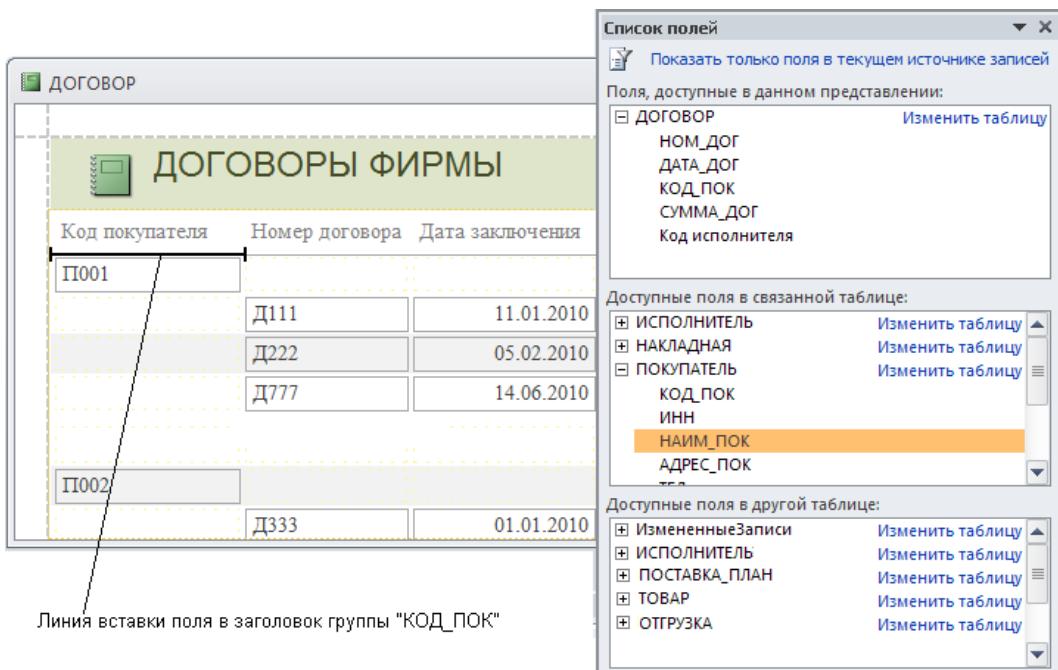
Откройте отчет Договоры фирмы в режиме макета. На вкладке ленты **Работа с макетами отчетов | Конструктор** (Report Layout Tools | Design) в группе **Сервис** (Tools) выберите команду **Добавить поля** (Add Existing Fields). В области **Список полей** (Field List) отображены поля, доступные в данном отчете. Щелкните в верхней части области на строке **Показать все таблицы** (Show all tables). В разделе **Доступные поля в связанной таблице** (Fields available in related tables) найдите таблицу **ПОКУПАТЕЛЬ**, в которой содержатся все сведения о покупателе.

Чтобы отобразить список полей таблицы, щелкните знак "плюс" (+) рядом с ее именем.

Для добавления нужного поля к отчету достаточно перетащить его из области **Список полей** (Field List) в нужный раздел отчета. Добавьте поле НАИМ\_ПОК в заголовок группы КОД\_ПОК. Чтобы поле попало в заголовок группы КОД\_ПОК, поместите его над значением поля с кодом покупателя (рис.7.17) или справа от него.

### **ЗАМЕЧАНИЕ**

Двойной щелчок в поле приводит к его размещению в конце области данных.



**Рис. 7.17.** Добавление поля в отчет в режиме макета

После перемещения поля в отчет Access создает соответствующий элемент управления, связанный с полем. Кроме того, создается присоединенная подпись. Оба элемента управления размещаются в заголовке группы. В предыдущей версии присоединенная подпись размещалась в верхнем колонтитуле, как и подпись поля с кодом покупателя.

В списке полей таблица ПОКУПАТЕЛЬ переместится из раздела **Доступные поля в связанной таблице** (Fields available in related tables) в раздел **Поля, доступные в данном представлении** (Fields available for this view). При этом автоматически изменяется источник записей отчета. В соответствующий запрос на выборку — представление — добавляются таблица ПОКУПАТЕЛЬ и перемещенное поле. Таким образом, поля текущего источника записей будут дополнены полем НАИМ\_ПОК.

Из области полей можно продолжить перемещение нужных полей в отчет. При этом источник записей отчета будет автоматически подправляться.

Просмотреть источник записей можно и в свойствах отчета. Откройте свойства отчета, щелкнув на кнопке **Страница свойств** (Property Sheet) на вкладке ленты **Конструктор** (Design). В окне свойств на вкладке **Данные** (Data) в строке **Источник записей** (Record Source) будет записана следующая инструкция SQL:

```
SELECT ДОГОВОР.* , ПОКУПАТЕЛЬ.НАИМ_ПОК
FROM ПОКУПАТЕЛЬ INNER JOIN ДОГОВОР
ON ПОКУПАТЕЛЬ.КОД_ПОК = ДОГОВОР.КОД_ПОК;
```

Отчет в режиме предварительного просмотра после добавления в него поля из таблицы ПОКУПАТЕЛЬ и незначительных изменений принял вид, показанный на рис. 7.18.

The screenshot shows a Microsoft Access report preview window titled 'ДОГОВОР'. The main title of the report is 'ДОГОВОРЫ ФИРМЫ'. The report header includes the date '6 июня 2010 г.' and time '19:34:12'. The data is presented in a table with the following columns: Покупатель (Buyer), Номер договора (Contract Number), Дата заключения (Date Signed), Сумма по договору (Contract Amount), and Код исполнителя (Executor Code). The data rows are:

Покупатель	Номер договора	Дата заключения	Сумма по договору	Код исполнителя
П001 Компьютер маркет	Д111	11.01.2010	6 047 100,00р.	6
	Д222	05.02.2010	152 280,00р.	1
	Д777	14.06.2010	1 579 920,00р.	1
			<b>7 779 300,00р.</b>	

The report footer shows the page number 'Страница: 1' and a note 'Нет фильтра' (No filter).

Рис. 7.18. Первая страница отчета в режиме предварительного просмотра

## Многотабличные отчеты

В предыдущем разделе отчет, автоматически созданный на одной таблице ДОГОВОР, был с помощью простого добавления полей из связанной таблицы пре-вращен в отчет на основе запроса, объединяющего поля двух таблиц. Таким образом, было показано, как, не создавая заранее запрос, можно строить отчет на основе данных нескольких связанных таблиц непосредственно в режиме макета.

В рассмотренном выше примере в отчет добавлялись поля из таблицы ПОКУПАТЕЛЬ главной по отношению к таблице ДОГОВОР — источнику записей отчета. Причем в отчете предварительно была выполнена группировка по коду по-

купателя, поэтому задача сводилась к добавлению полей в существующий заголовок группы.

В большинстве случаев создание отчета на основе нескольких таблиц, как путем перетаскивания полей таблиц в отчет, так и на основе ранее созданного запроса, требует значительной работы по настройке отчета. Некоторые операции по настройке такого отчета могут быть выполнены только в режиме конструктора.

Мастер отчетов позволяет в режиме диалога с пользователем создать многотабличный отчет путем выбора необходимых таблиц и полей, определения полей группировки, итоговых значений для записей. Создание отчета мастером является простой процедурой, а полученный отчет без больших усилий может быть приведен к желаемому виду.

Многотабличные отчеты, так же как формы, могут состоять из главного отчета и включаемого в него подчиненного отчета. Для каждого из этих отчетов в качестве источника данных выбирается своя таблица или запрос, построенный на нескольких таблицах. Чтобы обеспечить соответствие записей, выводящихся в подчиненном отчете, записям в главном отчете, устанавливается связь подчиненного отчета с главным. Однако мастер отчетов в отличие от мастера форм при выборе всех необходимых таблиц, например, для разработки и печати документа "Договор", не создает такого составного отчета. Мастер отчетов решает эту задачу с помощью группировки.

Решить эту задачу можно воспользовавшись мастером сначала для создания главного отчета, затем для создания подчиненного отчета. При создании мастером главного отчета выбрать макет в столбец, при создании подчиненного макет — табличный. На последнем этапе, открыв главный отчет в режиме макета или конструктора, перетащить в него подчиненный и, открыв свойства данных подчиненного отчета, установить его связь с главным.

Настройка отображения полученного отчета выполняется такими же средствами, как и настройка форм, достаточно просто, хотя и требует понимания назначения каждого из разделов отчета. Например, чтобы убрать заголовок/примечание подчиненного отчета, достаточно выполнить соответствующую команду контекстного меню. Чтобы отобразить названия столбцов в подчиненном отчете, достаточно в свойствах его макета для свойства **Отображать колониттулы страниц** (Show Page Header and Page Footer) выбрать Да (Yes). Чтобы отчет содержал на одной странице сведения из одной записи главной формы и связанных с ней записей подчиненной формы, т. е. отображал один документ, например, выводил данные об одном договоре, достаточно установить для макета области данных главного отчета свойство **Конец страницы** (Force New Page) со значением **До раздела** (Before Section). Свойство **Конец страницы** (Force New Page) со значением **После раздела** (Section) для заголовка главного отчета позволит отображать и печатать его на отдельной странице.

Рассмотрим создание многотабличного отчета сложной структуры с помощью мастера, который создаст на основе выбранных пользователем таблиц запрос — источник записей отчета, разместит в отчете поля, создаст необходимые группировки, итоги и т. д. При этом отчет за счет группировки, выполненной в нем, может

приобретать такой же вид, как отчет, состоящий из главной и подчиненной табличной частей, т. е. группировкой можно заменить создание подчиненного отчета.

Для придания отчету окончательного вида рассмотрим возможности инструментария Access, предоставляемого в режиме макета и конструктора.

## Разработка отчета с помощью мастера

Пусть необходимо подготовить отчет, позволяющий распечатать документ "Договор" на основе данных, ранее сохраненных с помощью формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ (см. главу 5).

Предположим, что макет отчета должен иметь вид, представленный на рис. 7.19. В соответствии с проектом макета в отчете предполагается выводить данные по каждому договору, включая сведения о покупателе, заключившем договор. Эти данные содержатся в таблицах ДОГОВОР и ПОКУПАТЕЛЬ. В табличной части по каждому договору необходимо вывести построчно данные о заказанных товарах, которые содержатся в таблицах ПОСТАВКА\_ПЛАН и ТОВАР.

<b>Договор №</b> _____		<b>от</b> _____					
Покупатель _____		Адрес _____	Телефон _____				
ИНН _____							
Банк _____							
Расчетный счет _____							
<b>Наименование товара</b>	<b>Срок поставки</b>	<b>Мин. партия</b>	<b>Количество</b>	<b>Цена</b>	<b>Единица измерения</b>	<b>Ставка НДС</b>	<b>Стоимость</b>
_____	_____	_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____	_____	_____
<b>Общая сумма</b> _____							

Рис. 7.19. Проект макета отчета для вывода данных по договору фирмы

## Выбор таблиц для отчета

Для создания отчета мастером выберем в области переходов таблицу ДОГОВОР, необходимую для формирования источника записей отчета. Первая выбранная таблица определит имя и заголовок отчета по умолчанию, а также вид представления данных в отчете.

Выполним команду **Мастер отчетов** (Report Wizard) в группе **Отчеты** (Reports) на вкладке ленты **Создание** (Create). В первом открывшемся окне мастера **Создание отчетов** (Report Wizard) (рис. 7.20) отобразится таблица ДОГОВОР и список ее полей. Здесь необходимо выбрать поля этой таблицы, а также все другие таблицы и поля, включаемые в отчет.

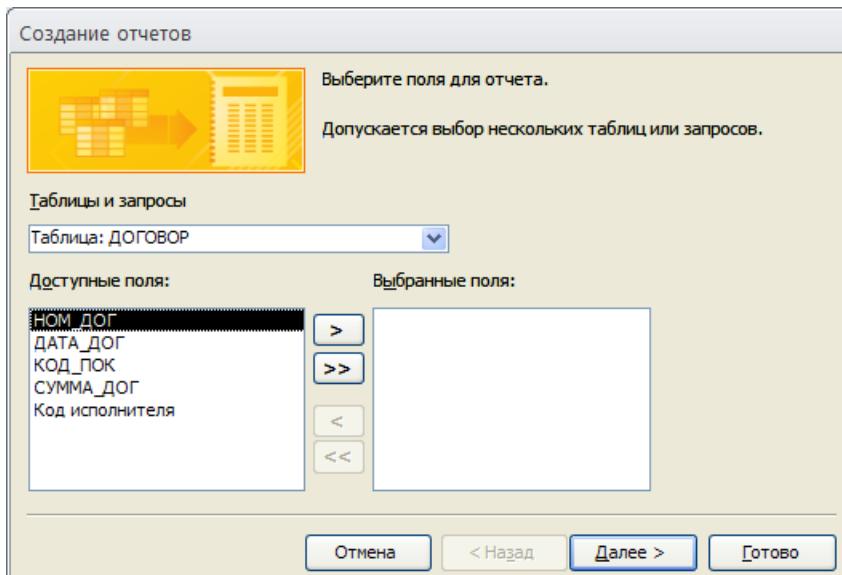


Рис. 7.20. Первое окно мастера создания отчетов

- В примере в соответствии с проектом отчета (см. рис. 7.20) выберем таблицы:
- ❖ ДОГОВОР, из нее поля:
    - ◆ номер договора — НОМ\_ДОГ;
    - ◆ дата заключения договора — ДАТА\_ДОГ;
  - ❖ ПОКУПАТЕЛЬ, из нее поля:
    - ◆ идентификационный номер налогоплательщика — ИНН;
    - ◆ наименование покупателя — НАИМ\_ПОК;
    - ◆ адрес покупателя — АДРЕС\_ПОК;
    - ◆ телефон — ТЕЛ;
    - ◆ банк, который обслуживает счет покупателя — БАНК;
    - ◆ номер расчетного счета покупателя — НОМ\_РСЧ.
  - ❖ ПОСТАВКА\_ПЛАН, из нее поля:
    - ◆ срок поставки — СРОК\_ПОСТ;
    - ◆ минимальная партия поставки — МИН\_ПОСТ;
    - ◆ количество товара — КОЛ\_ПОСТ;
  - ❖ ТОВАР, из нее поля:
    - ◆ наименование товара — НАИМ\_ТОВ;

- ◆ цена — ЦЕНА;
- ◆ единица измерения — ЕИ;
- ◆ ставка НДС — СТАВКА\_НДС.

При выборе взаимосвязанных таблиц мастер отчетов в соответствии со схемой базы данных *Поставка товаров* (см. рис. 3.52) автоматически установит связи между ними и построит запрос, объединяющий записи этих таблиц и включающий в таблицу запроса указанные поля. Этот запрос будет использован в качестве источника данных отчета, а соответствующая инструкция SQL будет записана мастером в свойство отчета **Источник записей** (Record Source). Объединение записей таблиц будет производиться способом, указанным параметрами объединения для каждой из связей в схеме данных. Подсхема данных, включающая таблицы, на которых будет строиться отчет, приведена на рис. 7.21.

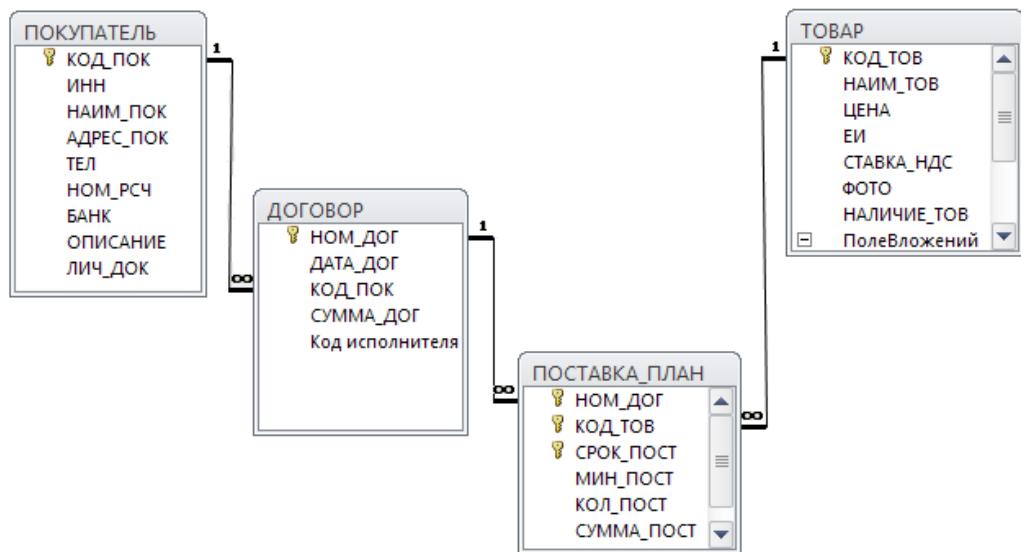


Рис. 7.21. Подсхема данных отчета ДОГОВОРЫ ФИРМЫ

В окне мастера (рис. 7.22) выберем вид представления данных в отчете. Чтобы получить отчет, заданный в рассматриваемом примере, выберем таблицу ДОГОВОР в качестве основы главной части отчета и увидим, что мастером на основе полей этой таблицы будет осуществлена группировка. Причем к полям группировки будут отнесены и поля таблицы ПОКУПАТЕЛЬ, поскольку эта таблица является главной по отношению к таблице ДОГОВОР, что позволяет дополнить записи подчиненной таблицы ДОГОВОР сведениями о покупателе, заключившем договор.

Поля таблицы ПОСТАВКА\_ПЛАН составят основу записей для построения табличной части отчета. Причем записи о товарах, перечисленных в этой таблице, будут дополнены реквизитами из таблицы ТОВАР.

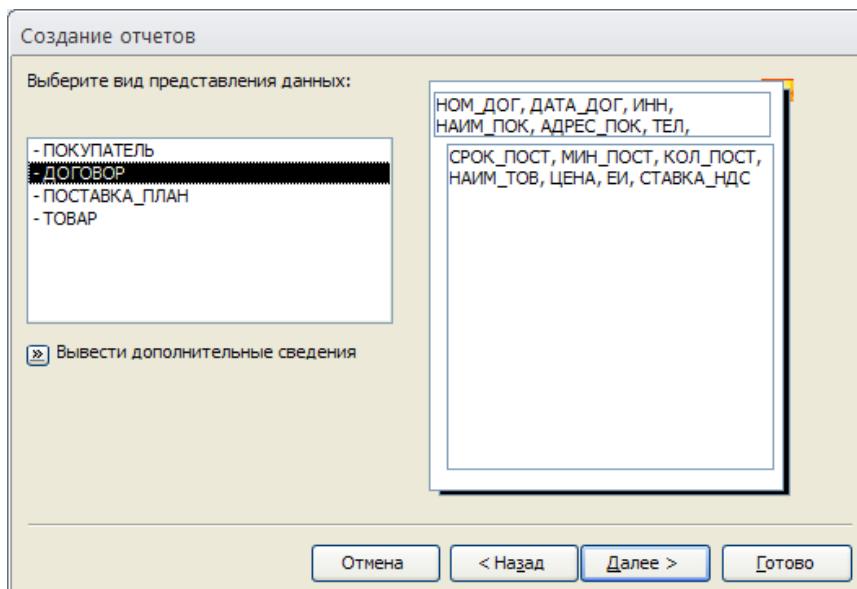


Рис. 7.22. Окно мастера отчетов при выборе вида представления данных

Если, выбирая вид представления данных, остановиться на таблице ПОКУПАТЕЛЬ, мастер создаст два уровня группировки: на первом будут собраны данные о покупателе, на втором — о его договорах (рис. 7.23).

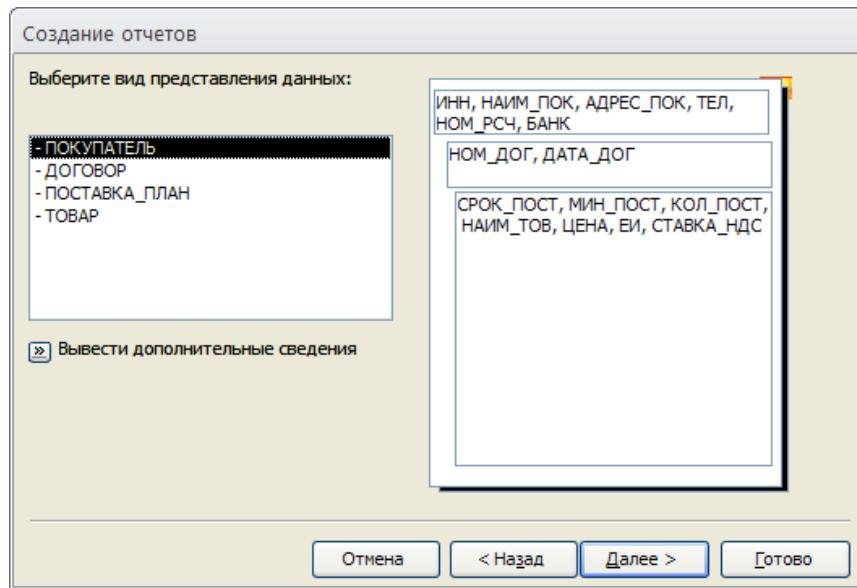
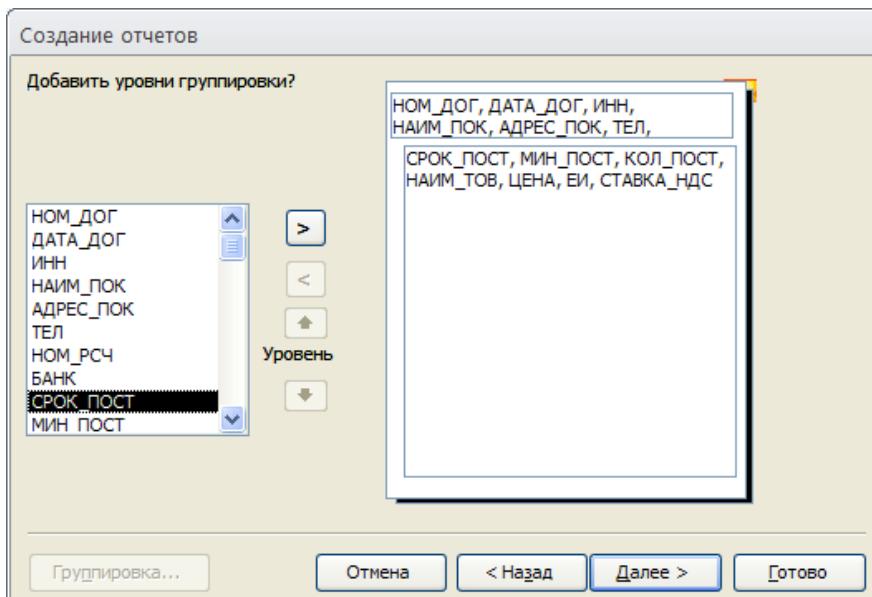


Рис. 7.23. Окно мастера отчетов при выборе представления с двумя уровнями группировки

В следующем окне мастера (рис. 7.24) можно добавить уровни группировки. Если в предыдущем окне группировка производилась в соответствии с уровнями иерархии таблиц в схеме данных, что позволило отразить их подчиненность в отчете, то в этом окне добавление группировки может быть выполнено по любому из полей таблицы на любом уровне иерархии. Очевидно, что группировка должна иметь смысл. Например, в наборе строк из таблицы ПОСТАВКА\_ПЛАН можно выполнить группировку по сроку поставки. В отчете, создаваемом мастером, можно определить до четырех уровней группировки.



**Рис. 7.24.** Окно мастера отчетов при добавлении новых уровней группировки

Чтобы добавить уровни группировки, дважды щелкните по добавляемому в отчет имени поля из списка. Для удаления уровня группировки дважды щелкните по нему на экране страницы в правой части диалогового окна. Для добавления и удаления уровней группировки можно также воспользоваться кнопками со стрелками. Таким образом добавляется очередной уровень группировки, который отображается вложенным в предыдущий уровень группировки. Пример с включением уровня группировки по полю СРОК\_ПОСТ (срок поставки) показан на рис. 7.25.

Для построения отчета остановимся на варианте группировки, показанном на рис. 7.25. Далее в диалоге с мастером выберем сортировку по наименованию товара, причем подведение итогов выполнять не будем, т. к. в данном примере это не имеет какого-либо заметного смысла. Выберем вид макета **структура** (Stepped), альбомную ориентацию страниц отчета, попросим мастера настроить ширину полей для размещения на одной странице и выберем требуемый стиль оформления отчета.

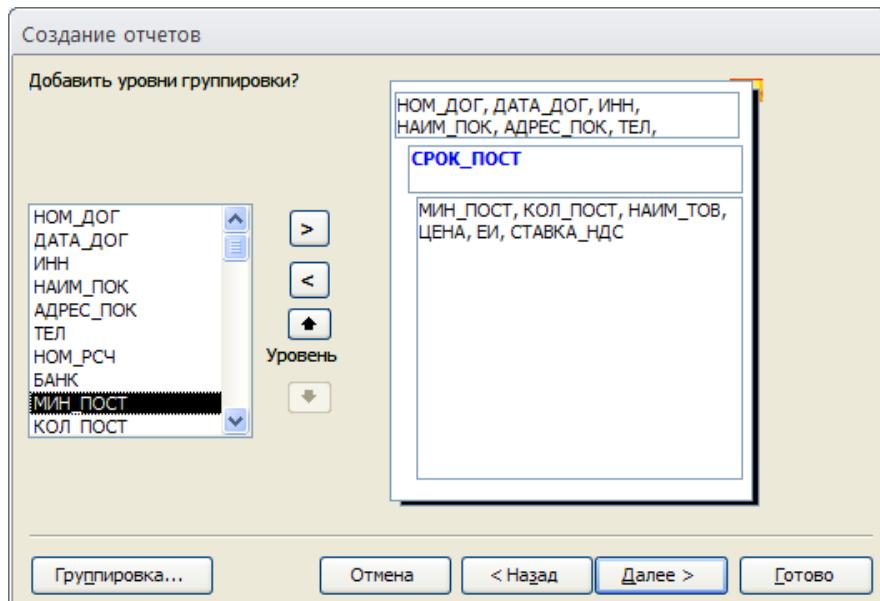


Рис. 7.25. Окно мастера отчетов при добавлении уровня группировки по сроку поставки

Дадим отчету имя Договоры фирмы и в качестве дальнейших действий выберем **Изменить макет отчета** (Modify the report's design). Результат работы мастера представлен на рис. 7.26.

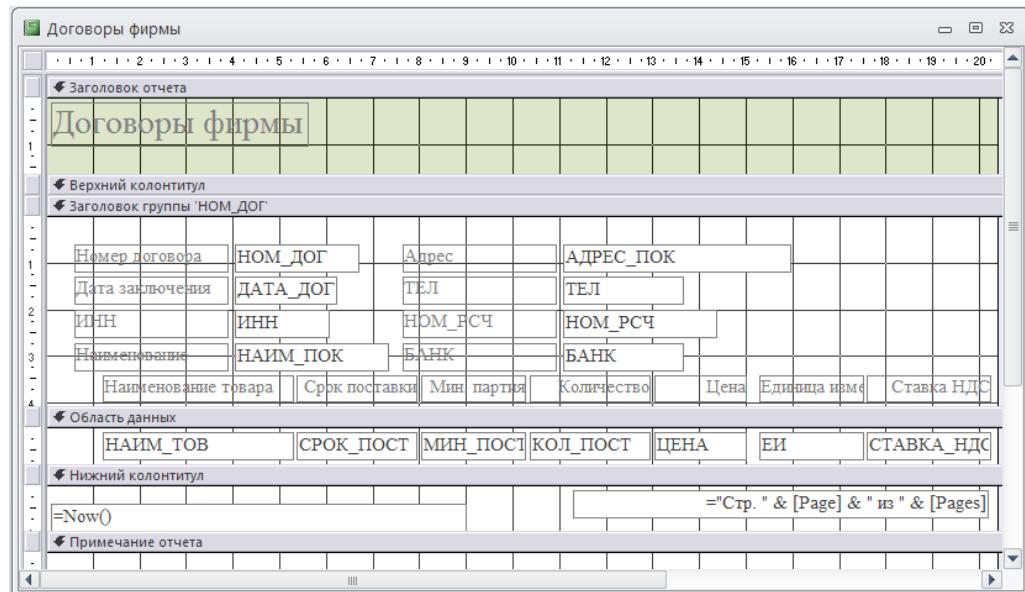


Рис. 7.26. Отчет на основе нескольких взаимосвязанных таблиц в режиме конструктора

## Источник записей отчета

Мастер на основе выбранных таблиц создает запрос. Просмотрим его, открыв свойства отчета. Для этого в режиме конструктора в любом месте окна отчета откроем контекстное меню и выполним команду **Свойства отчета** (Report Properties). В окне свойств отчета на вкладке **Данные** (Data) в строке **Источник записей** (Record Source) содержится инструкция SQL:

```
SELECT [ДОГОВОР].[НОМ_ДОГ], [ДОГОВОР].[ДАТА_ДОГ], [ПОКУПАТЕЛЬ].[ИНН],
       [ПОКУПАТЕЛЬ].[НАИМ_ПОК], [ПОКУПАТЕЛЬ].[АДРЕС_ПОК],
       [ПОКУПАТЕЛЬ].[ТЕЛ], [ПОКУПАТЕЛЬ].[НОМ_РСЧ], [ПОКУПАТЕЛЬ].[БАНК],
       [ПОСТАВКА_ПЛАН].[СРОК_ПОСТ], [ПОСТАВКА_ПЛАН].[МИН_ПОСТ],
       [ПОСТАВКА_ПЛАН].[КОЛ_ПОСТ], [ТОВАР].[НАИМ_ТОВ], [ТОВАР].[ЦЕНА],
       [ТОВАР].[ЕИ], [ТОВАР].[СТАВКА_НДС]
FROM ТОВАР INNER JOIN ((ПОКУПАТЕЛЬ INNER JOIN ДОГОВОР
ON ПОКУПАТЕЛЬ.КОД_ПОК=ДОГОВОР.КОД_ПОК) INNER JOIN
ПОСТАВКА_ПЛАН ON ДОГОВОР.НОМ_ДОГ=ПОСТАВКА_ПЛАН.НОМ_ДОГ)
ON ТОВАР.КОД_ТОВ=ПОСТАВКА_ПЛАН.КОД_ТОВ;
```

Инструкция SELECT определяет список полей запроса на выборку, а ее предложение FROM — последовательность и способ объединения таблиц.

Значком построителя, размещенным справа в строке **Источник записей** (Record Source), можно открыть инструкцию SQL в режиме конструктора запросов. Запрос, являющийся источником записей отчета, можно изменить в любом из этих режимов: добавить или удалить поля, таблицы.

### ЗАМЕЧАНИЕ

Если отчет создается в режиме конструктора или с помощью средства **Пустой отчет** (Blank Report), который отличается тем, что открывает пустой отчет в режиме макета, прежде всего, необходимо определить источник записей. Задается источник записей в свойствах отчета, как в рассмотренном примере, путем записи инструкции SQL или с помощью построителя запросов. Кроме того, запрос может быть создан путем перетаскивания полей из области **Список полей** (Field List), доступной как в режиме конструктора, так и в режиме макета.

## Доработка отчета в режиме конструктора

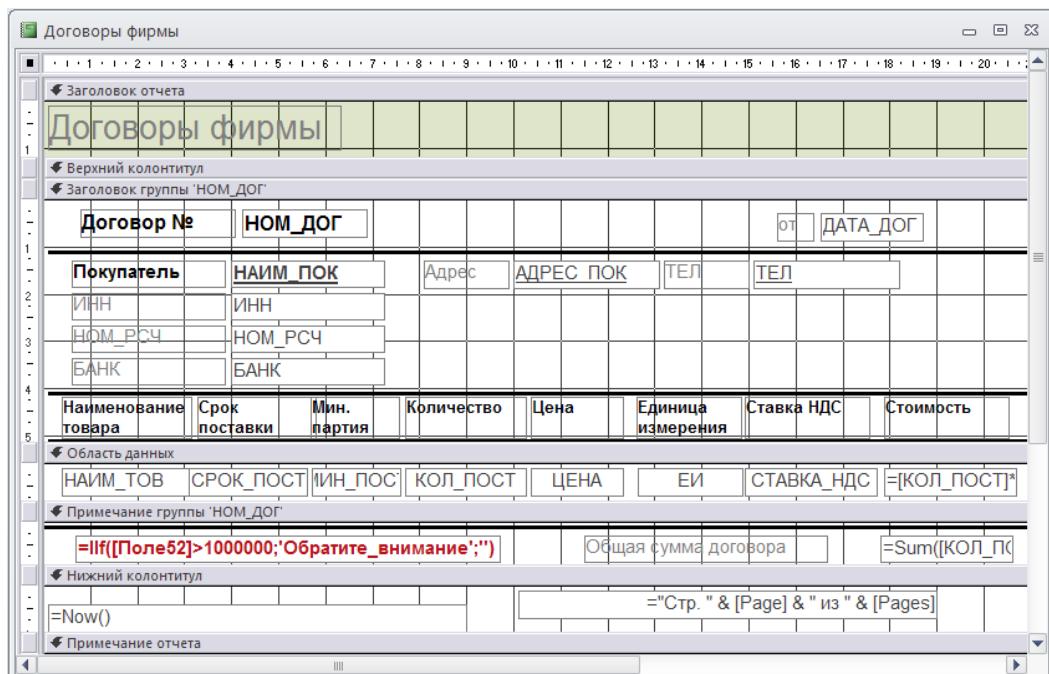
Для того чтобы отчет соответствовал проекту макета, его надо доработать в режиме конструктора.

После создания отчета мастером надписи полей группировки и их значения размещены в заголовке группы (см. рис. 7.26). Здесь же размещены надписи столбцов табличной части отчета. Это позволяет выводить названные элементы один раз в начале каждого договора.

Используя средства конструктора отчетов, разместим и отформатируем надписи и поля в разделах отчета, как показано на рис. 7.27.

Чтобы каждый договор отображался с новой страницы отчета, установите для заголовка группы значение свойства макета **Конец страницы** (Force New Page) равным **До раздела** (Before Section). Если договор занимает несколько страниц для

отображения реквизитов договора и заголовков столбцов на каждой странице установите для заголовка группы значение Да (Yes) в свойстве макета **Повторение раздела** (Repeat Section). При этом на каждой странице одного договора будут повторяться как надписи столбцов табличной части отчета, так и все общие реквизиты договора.



**Рис. 7.27.** Отчет о договорах фирмы, доработанный в режиме конструктора

Если необходимо в отчете сформировать титульную страницу, установите в свойствах заголовка отчета на вкладке **Макет** (Format) в строке **Конец страницы** (Force New Page) значение **После раздела** (After Section). Увеличив размер области заголовка отчета до размера страницы, можно придать титульной странице необходимый вид.

## Вычисляемые поля в отчете

Рассчитаем стоимость каждого из товаров, заказанных в договоре. Для этого нужно дополнить перечень полей области данных вычисляемым полем и записать в него выражение  $=[\text{ЦЕНА}] * [\text{КОЛ\_ПОСТ}]$ . Вычисляемое поле создадим, щелкнув на кнопке **Поле** (Text Box) на вкладке ленты **Конструктор** (Design) в группе **Элементы управления** (Controls) и перетащив его в область данных. В созданном таким образом элементе **Свободный** (Unbound) запишем нужное выражение. Переместим надпись этого поля в заголовок группы, заменив ее значение на **Стоимость**.

Это выражение будет внесено в свойства элемента. Откроем свойства элемента на вкладке **Данные** (Data) и увидим выражение в строке **Данные** (Control Source). На вкладке **Макет** (Format) в строке **Число десятичных знаков** (Decimal Places) выберем точность значения — 2, в строке **Формат поля** (Format) — денежный формат отображения значения поля.

Рассчитаем стоимость всех товаров, заказанных в договоре. Для создания вычисляемого поля с итоговым значением по группе записей его надо разместить в разделе примечания группы. Чтобы отобразить примечание группы, откроем область **Группировка, сортировка и итоги** (Group, Sort, and Total), щелкнув на кнопке **Группировка** (Group & Sort) на вкладке ленты **Конструктор** (Design). В строке **Группировка НОМ\_ДОГ** (Group on НОМ\_ДОГ) откроем дополнительные параметры группировки, щелкнув на значке **Больше** (More), и выберем значение параметра **с разделом примечания** (with a footer section).

В примечании группы создадим вычисляемый элемент **Свободный** (Unbound), в который может быть записано выражение для выполнения групповой операции (суммирования, расчета среднего арифметического, подсчета числа записей в группе и др.). Запишем в него выражение `=Sum([КОЛ_ПОСТ]*[ЦЕНА])`. Также как для вычисляемого поля **Стоимость**, установим точность значения и формат отображения. Подпись изменим на **Общая сумма договора**.

Пусть необходимо особо пометить договоры, общая сумма которых составляет более миллиона рублей. Создадим в примечании группы вычисляемое поле и запишем в него выражение `=If([Поле42]>1000000,'Обратите_внимание','')`, где Поле42 — имя поля со значением общей суммы, созданного выше. Для этого вычисляемого поля выберем новые значения для шрифта, размера шрифта, насыщенности и цвета текста. Надпись поля удалим. Для договора на сумму более миллиона рублей в отчете будет выводиться фраза "Обратите внимание", оформленная в соответствии со сделанным выбором. Для остальных договоров никакие слова не выводятся, т. к. третий operand функции `If` задает в качестве значения пустую строку.

Страница отчета с данными по одному договору в режиме предварительного просмотра приведена на рис. 7.28. В этом примере заголовок отчета Договоры фирмы и все добавленные в него сведения, рисунки, графики выводятся на отдельном титульном листе, один раз на первой странице отчета.

Для нумерации строк в текстовой части отчета создайте в области данных вычисляемое поле, в котором запишите выражение `=1`. Затем для этого поля установите на вкладке **Данные** (Data) в свойстве **Сумма с накоплением** (Running Sum) значение **Для группы** (Over Group). Не следует использовать это свойство для вычисляемых полей примечания.

Если необходимо подсчитать общую сумму по всем договорам, создайте вычисляемое поле с таким же выражением, как при подсчете общей суммы по одному договору: `=Sum([КОЛ_ПОСТ]*[ЦЕНА])`, но в области примечания или заголовка отчета. Например, создайте такое поле с надписью **Договоров заключено на сумму** на титульном листе отчета.

The screenshot shows a Microsoft Access report preview window. The title bar says 'Договоры фирмы'. The main header of the report is 'Договор № Д111' with a date 'от 11.01.2010'. Below this, there's a section for 'Покупатель' (Buyer) with fields: ИНН '778957651111', НОМ\_РСЧ '76358509763264536', and БАНК 'Мост'. The main body of the report is a table with columns: Наименование товара (Product Name), Срок поставки (Delivery Date), Мин. партия (Min. Order Qty), Количество (Quantity), Цена (Price), Единица измерения (Unit of Measurement), Ставка НДС (VAT Rate), and Стоимость (Total Cost). The table contains 8 rows of data. At the bottom of the report, there's a red note 'Обратите внимание' (Please note) and a summary row: 'Общая сумма договора' (Total Contract Amount) '5 714 000,00р.'.

Наименование товара	Срок поставки	Мин. партия	Количество	Цена	Единица измерения	Ставка НДС	Стоимость
CD-ROM Pana	1	0	710	800,00р.	штуки	30%	568 000,00р.
FDD 3,5	1	10	50	500,00р.	коробка	6%	25 000,00р.
FDD 3,5	3	5	100	500,00р.	коробка	6%	50 000,00р.
HDD Maxtor 12	1	4	100	1 120,00р.	штучки	5%	112 000,00р.
Монитор 17LG	2	5	100	7 000,00р.	штука	5%	700 000,00р.
Монитор 17LG	3	5	300	7 000,00р.	штука	5%	2 100 000,00р.
Монитор 17LG	1	10	305	7 000,00р.	штука	5%	2 135 000,00р.
Принтер EPS	2	0	10	2 400,00р.	штука	10%	24 000,00р.

**Рис. 7.28.** Страница отчета с данными договора в режиме предварительного просмотра

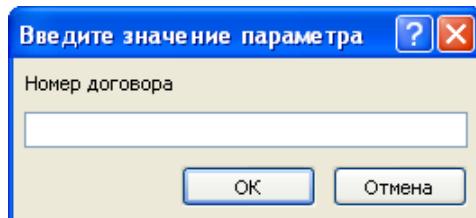
По вычисляемому полю данных, такому как **Стоимость**, нельзя автоматически задать вывод итогов ни по группе, ни в отчете в целом. При использовании вычисляемых полей в примечании группы и отчета, так же как при автоматическом определении итоговых полей, можно просмотреть отчет без подробностей, т. е. с отображением только указанных вычисляемых полей.

## Определение параметров в отчете

Пусть необходимо в отчете получать данные только о заданном договоре. Эта задача может быть решена с помощью ввода параметра в запрос, на котором основывается отчет.

Откроем отчет Договоры фирмы в режиме конструктора. В свойствах отчета на вкладке **Данные** (Data) в конце строки **Источник записей** (Record Source) щелкнем на кнопке построителя. В окне построителя запросов перейдем к полю с номером договора — НОМ\_ДОГ. В строке **Условие отбора** (Criteria) определим параметр, записав его имя в квадратных скобках, например [Номер договора].

Теперь при просмотре отчета будет выводиться диалоговое окно ввода значения параметра (рис. 7.29).



**Рис. 7.29.** Диалоговое окно ввода значения параметра отчета

После ввода значения параметра отчет будет содержать данные только одного договора.

Чтобы введенные значения параметров отображались в просматриваемом или напечатанном отчете, выполните следующие действия. Откройте отчет в режиме конструктора. Вставьте в отчет вычисляемый элемент управления. Для этого нажмите на вкладке **Конструктор** (Design) кнопку элемента управления **Поле** (Text Box) и вычертите элемент в месте, куда нужно его поместить. Запишите в это новое поле выражение, содержащее только имя параметра. Параметр должен записываться точно так же, как он записан в условии отбора запроса, например, `= [Договоры на сумму более]`.

Для просмотра условий отбора откройте его в режиме предварительного просмотра. Значения параметров отобразятся в точности так, как они введены в приглашении.

## Анализ данных отчета средствами фильтрации

В отчете с параметром вы получили сведения лишь об одном договоре и не можете всесторонне в интерактивном режиме проанализировать данные о договорах фирмы. Удалите параметр из запроса — источника записей отчета. Откройте отчет в режиме **Представление отчета** (Report View). Пользуйтесь для переключения режимов строкой состояния.

Используйте фильтры для выборки и анализа различных сведений из договоров.

Установите курсор на поле с номером договора. Щелкните на кнопке **Фильтр** (Filter). В открывшемся окне откройте список **Текстовые фильтры** (Text Filters). Для поля доступны эти фильтры, потому что поле имеет текстовый тип данных. В списке щелкните по **Содержит** (Contains), откроется настраиваемый фильтр (рис. 7.30), в котором укажите нужный номер договора. Для открытия списка **Текстовые фильтры** (Text Filters) можно также воспользоваться контекстным меню поля. В отчете отобразится выбранный договор.

Чтобы снять фильтр с отчета, используйте соответствующую команду контекстного меню поля. Если необходимо, задайте новые условия отбора по полю с номером договора.

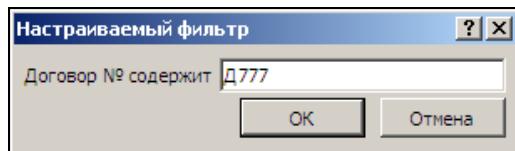


Рис. 7.30. Диалоговое окно задания условия отбора для фильтра

Чтобы отобрать договоры за заданный период, в контекстном меню поля с датой заключения договора откройте **Фильтр дат** (Date Filters).

### **ВНИМАНИЕ!**

Итоговые данные по договору подсчитываются только для выбранных в результате фильтрации данных.

Для отбора договоров, в которых заказан заданный товар, используйте текстовый фильтр по полю с наименованием товара. Для отбора заданного товара, заказанного в заданном месяце, не снимая фильтра с наименования товара, выполните фильтрацию по полю СРОК\_ПОСТ (срок поставки). Числовой фильтр позволит отобрать данные в любом заданном интервале.

Таким образом, использование фильтра позволит без изменения отчета получать данные в различном представлении. Самые сложные данные можно представить в удобном для понимания виде, создавая в интерактивном режиме отчеты для определенных целей.

#### **Задание 7.1. Создание отчета**

Создайте отчет для вывода договоров, сгруппированных по заказчикам — покупателям. В предметной области указано, что один покупатель может заключать с фирмой несколько договоров.

## **Составные отчеты**

Пусть необходимо получить отчет об отгрузках товаров за весь учетный период. Отчет должен содержать все данные о товаре и список всех отгрузок товара, сгруппированных по месяцам (рис. 7.31).

Для создания отчета, соответствующего заданию, потребуется использовать данные из следующих взаимосвязанных таблиц:

❖ ТОВАР, поля:

- ◆ код товара — КОД\_ТОВ;
- ◆ наименование товара — НАИМ\_ТОВ;
- ◆ цена — ЦЕНА;
- ◆ единица измерения — ЕИ;

- ◆ ставка НДС — СТАВКА\_НДС;
- ◆ фотография товара — ФОТО;
- ◆ ОТГРУЗКА, поля:
  - ◆ код склада — КОД\_СК;
  - ◆ код товара — КОД\_ТОВ;
  - ◆ количество отгруженного товара — КОЛ\_ОТГР;
- ◆ НАКЛАДНАЯ, поле:
  - ◆ дата отгрузки — ДАТА\_ОТГР.

**ОТГРУЗКА ТОВАРА**

Код товара	<input type="text"/>	<input type="text"/>		
Наименование товара	<input type="text"/>			
Цена	<input type="text"/>			
Единица измерения	<input type="text"/>			
Ставка НДС	<input type="text"/>			
	Код склада	Код товара	Количество	Дата
Месяц	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Число отгрузок:	<input type="text"/>	Количество отгруженного:		
	<input type="text"/>			
<b>ИТОГО ОТГРУЖЕНО</b>	<input type="text"/>			

Рис. 7.31. Проект макета для создания отчета об отгрузке товаров

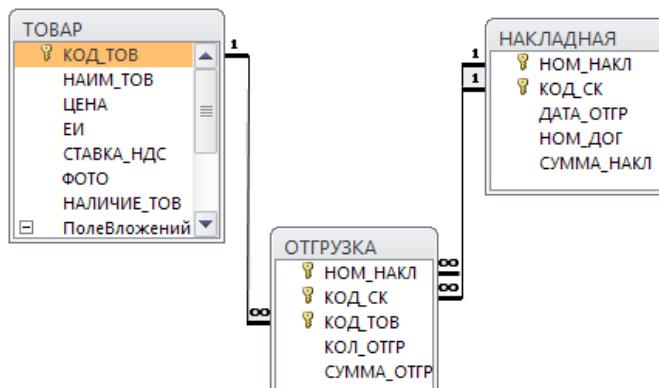


Рис. 7.32. Подсхема данных отчета об отгрузке товаров

Связи перечисленных таблиц определены в подсхеме данных, представленной на рис. 7.32.

Очевидно, что главную часть отчета должны составить данные из таблицы ТОВАР, табличную часть отчета — данные из подчиненной таблицы ОТГРУЗКА. Причем записи таблицы ОТГРУЗКА должны быть дополнены датой, которая хранится в главной по отношению к ней таблице НАКЛАДНАЯ.

В соответствии с этим создадим составной отчет, содержащий главный отчет и включенный в него подчиненный отчет.

## Создание главного отчета

Главная часть составного отчета должна содержать все сведения о товаре, которые хранятся в таблице базы данных ТОВАР, и может быть создана как обычный однотабличный отчет о товарах фирмы. Для автоматического создания отчета, включающего все поля одной таблицы, достаточно выбрать в области переходов таблицу ТОВАР и выполнить команду **Отчет (Report)** на вкладке ленты **Создание (Create)**. Таблица ТОВАР будет указана в качестве источника данных этого отчета.

Созданный однотабличный отчет открывается в режиме макета. В качестве заголовка отчета использовано имя таблицы. Он включает все поля таблицы. Присоединенные элементы управления для каждого поля отчета связаны с подписями и образуют один табличный макет, размещенный в области данных.

В режиме макета пунктирными линиями показаны границы текста на странице, что позволяет правильно выбрать местоположение и размер элементов управления.

Можно создать отчет и с помощью команды **Пустой отчет (Blank Report)**, которая открывает пустой макет и **Список полей (Field List)**. Перетащив нужные поля таблицы в макет отчета, можно получить такой же отчет, как и по команде **Отчет (Report)**. Область **Список полей (Field List)** может быть открыта командой **Добавить поля (Add Existing Fields)** на вкладке ленты **Работа с макетами отчета | Конструктор (Report Layout Tools | Design)** в группе **Сервис (Tools)**.

Поскольку в главной части отчета должны отображаться данные только одной записи, преобразуем его макет в столбик. Для этого выделим табличный макет, щелкнув в его левом верхнем углу, и выполним команду **В столбик (Stacked)** на вкладке ленты **Работа с макетами отчета | Упорядочить (Report Layout Tools | Arrange)** в группе **Таблица (Table)**.

Для того чтобы сделать отчет более компактным, переместим поле с фотографией товара в верхнюю правую часть отчета, освободив место для подчиненного отчета. Перемещать отдельный элемент управления за пределы макета, а также удалять из отчета, можно только после его удаления из макета. Для удаления из макета поля с фотографией и его надписи выделим их и выполним соответствующую команду контекстного меню. После исключения элементов управления из макета, не снимая выделения, переместим их в новое место.

Удалим надпись поля с фотографией, выделив ее и нажав клавишу <Delete>.

Размещение элементов в макете значительно упрощает их упорядочение в отчете, но требует выполнения операций по объединению элементов управления в группу и удалению из нее.

При создании отчета в его примечании автоматически сформировалось вычисляемое поле с расчетом суммарного значения для единственного числового поля в таблице ЦЕНА. Учитывая бессмысленность этого вычисления, удалим его.

Переключимся в режим конструктора, чтобы хорошо видеть структуру разделов отчета. Для переключения щелкнем на соответствующем значке в строке состояния. Отчет с данными о товарах в режиме конструктора представлен на рис. 7.33.

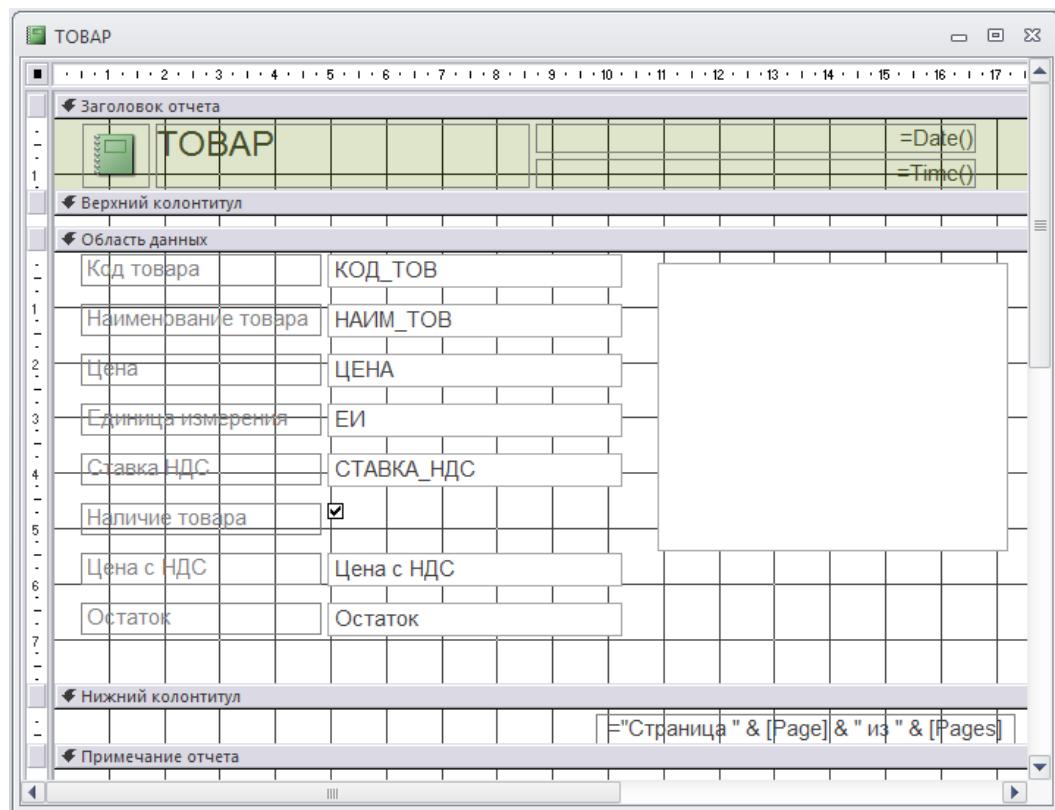


Рис. 7.33. Отчет о товарах фирмы в режиме конструктора

## Создание подчиненного отчета

Табличная часть отчета должна содержать все сведения об отгрузках, которые хранятся в таблицах ОТГРУЗКА и НАКЛАДНАЯ. Создадим на основе этих таблиц отчет, который будет включен в главную часть отчета в качестве подчиненного.

Создадим по таблице ОТГРУЗКА однотабличный отчет с помощью команды **Отчет** (Report), который откроется в режиме макета. Из области **Список полей** (Field List) добавим в отчет нужные поля из таблицы НАКЛАДНАЯ. Источник записей отчета автоматически дополнится таблицей НАКЛАДНАЯ и ее полями.

Табличный макет этого отчета вполне подходит для реализации нужного представления. Подберем ширину элементов управления таким образом, чтобы все столбцы отчета поместились на странице, по умолчанию ориентированной на размер бумаги А4 и книжную ориентацию. Для того чтобы получить возможность уменьшить ширину столбцов, сохранив полное отображение содержимого, разместим подписи в заголовках столбцов в двух строках, увеличив их высоту.

Для задания интервала между элементами управления и линиями сетки макета выделите макет отчета, щелкнув в его левом верхнем углу, и выберите нужный интервал из списка команды **Внутренние поля** (Control Padding), размещенной на вкладке ленты **Упорядочить** (Arrange).

## Сортировка и группировка записей отчета

Для группировки данных отчета по дате отгрузки выполним команду **Группировка** (Group & Sort) на вкладке ленты **Конструктор** (Design). В открывшейся области **Группировка, сортировка и итоги** (Group, Sort, and Total) щелкнем на кнопке **Добавить группировку** (Add a group). Выберем для предлагаемого уровня группировки поле **ДАТА\_ОТГР**. Откроем список, щелкнув на параметре **по кварталам** (by quarter), и выберем **по месяцам** (by month) (рис. 7.34). Для закрытия списка щелкните по любому месту за его пределами.

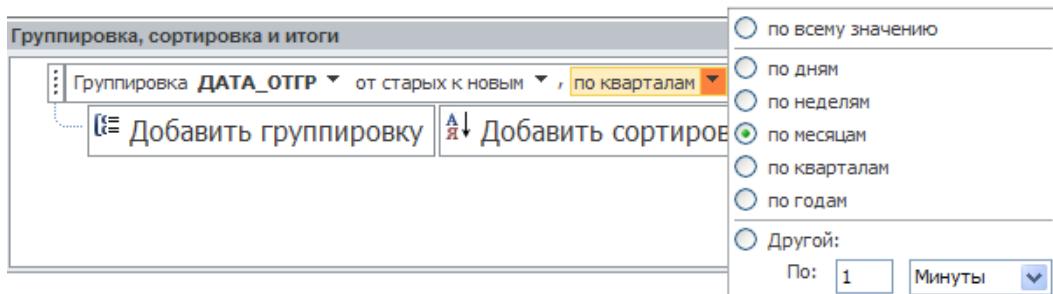


Рис. 7.34. Выбор для поля с датой группировки по месяцам

Для добавления в отчет итоговых сумм по каждому месяцу и общей суммы щелкните на кнопке **Больше** (More), в области параметров откройте список значений параметра **с итогами**: СУММА\_ОТГР (with SUMMA\_OTGR totaled). В поле списка **Итог на** (Total On) выберите поле СУММА\_ОТГР, по которому нужно рассчитать итоговые значения. Отметьте флажок **Показать общий итог** (Show Grand Total) и определите место отображения итоговых значений по каждому месяцу выбором флажка **Показать в примечании группы** (Show in group footer) (рис. 7.35).

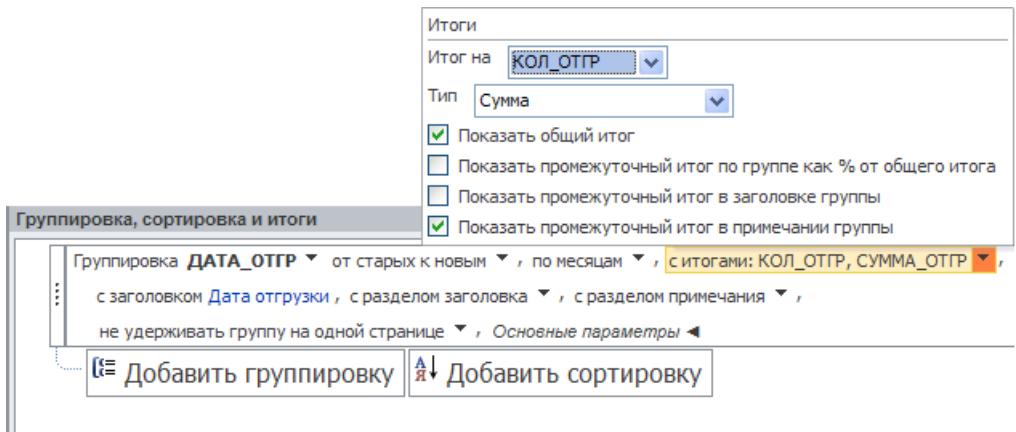


Рис. 7.35. Группировка с итогами: по двум полям КОЛ\_ОТГР и СУММА\_ОТГР

ОТГРУЗКА							10 июня 2010 г.	18:26:20				
							Дата отгрузки	Код накладной	Код склада	Код товара	Количество	Сумма отгружено
<b>Январь 2010</b>												
18.01.2010	H001	C01	T008	10	2 338,00р.							
18.01.2010	H001	C01	T001	50	350 000,00р.							
18.01.2010	H001	C01	T007	10	22 000,00р.							
18.01.2010	H001	C01	T005	2	8 000,00р.							
18.01.2010	H001	C01	T002	10	5 000,00р.							
21.01.2010	H004	C02	T001	10	70 000,00р.							
21.01.2010	H004	C02	T002	10	5 000,00р.							
21.01.2010	H004	C02	T003	1	1 120,00р.							
21.01.2010	H004	C02	T004	100	70 000,00р.							
				203	533 458,00р.							
<b>Февраль 2010</b>												
04.02.2010	H001	C03	T006	10	7 000,00р.							
11.02.2010	H002	C02	T002	55	27 500,00р.							
11.02.2010	H002	C02	T003	8	20 160,00р.							

Рис. 7.36. Подчиненный отчет ОТГРУЗКА в режиме макета

Поскольку нам нужно подсчитать итоги и по количеству отгруженного, в раскрывающемся списке **Итог на** (Total On) выберем поле КОЛ\_ОТГР и отметим те же флажки.

## ЗАМЕЧАНИЕ

В данном отчете расчет итоговых значений для поля с количеством отгруженного не имеет смысла, т. к. происходит сложение количеств разных товаров. Однако после того как отчет будет вставлен в главный отчет, из общего набора строк будут отбираться только строки, связанные с конкретным товаром, и, следовательно, будет суммироватьсья количество для одного товара.

В качестве значения параметра **Порядок сортировки** по умолчанию выбирается устраивающее нас значение **от старых к новым** (from oldest to newest).

Отчет в режиме макета после внесения перечисленных изменений представлен на рис. 7.36.

Закроем отчет, сохранив его под данным по умолчанию именем ОТГРУЗКА.

## Включение подчиненного отчета

Если отчет ТОВАР закрыт, откройте его в режиме конструктора. В области переходов выделите имя отчета ОТГРУЗКА и перетащите в его область данных главного отчета ТОВАР. Отчет ОТГРУЗКА встраивается в качестве подчиненного в главный отчет, где он доступен для редактирования.

Access автоматически устанавливает связь между главным и подчиненным отчетом по полю КОД\_ТОВ (код товара), т. к. она определена в схеме данных. При этом не имеет значения, включено ли поле связи в подчиненный отчет. Проверить имена полей связи можно в свойствах элемента **Подчиненная форма/отчет** (Subform/Subreport) на вкладке **Данные** (Data) в строках **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields) (рис. 7.37). Не путайте свойства подчиненного отчета и элемента **Подчиненная форма/отчет** (Subform/Subreport), содержащего подчиненный отчет.

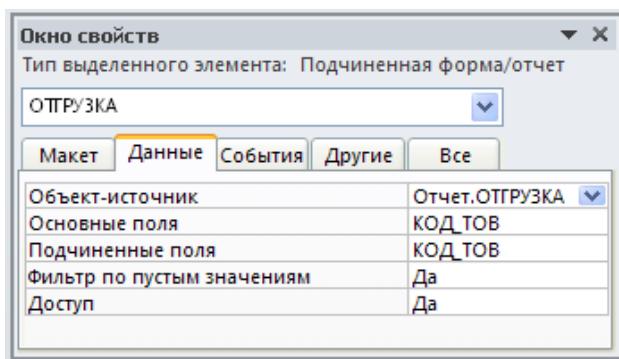


Рис. 7.37. Свойства элемента  
**Подчиненная форма/отчет**

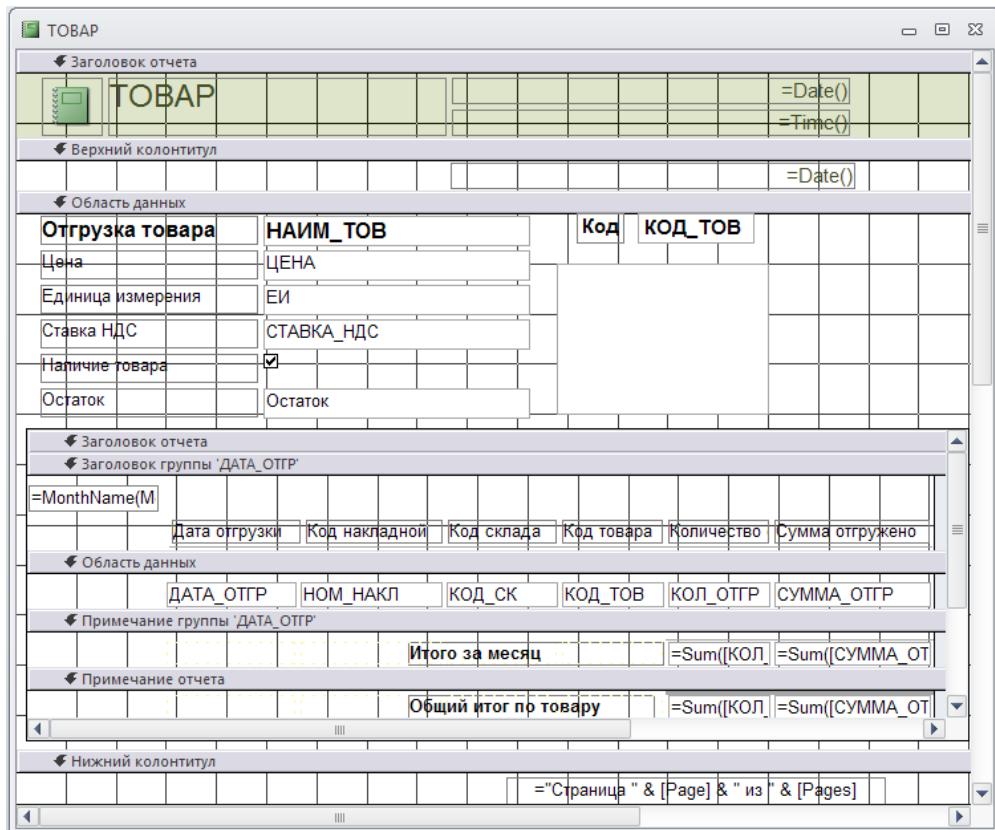
Установленная связь отчетов ТОВАР и ОТГРУЗКА позволяет для каждой записи в отчете ТОВАР отображать только связанные записи в подчиненном отчете ОТГРУЗКА.

Для включения подчиненного отчета можно также воспользоваться услугами мастера подчиненных отчетов. Инициировать работу мастера можно на вкладке ленты **Конструктор** (Design) в коллекции кнопки Элементы управления кнопкой

**Подчиненная форма/отчет** (Subform/Subreport). При этом должна быть выключена кнопка **Использовать мастера** (Use Control Wizards). Мастер создает в области данных основного отчета элемент **Свободный** (Unbound) с надписью **ВнедренныйN** (ChildN) и предлагает выбрать существующий подчиненный отчет или создать его на основе таблиц и запросов. Далее определяются поля связи основного и подчиненного отчетов, имя подчиненного отчета и на этом работа мастера завершается. Свободный элемент превращается в элемент, называемый **Подчиненная форма/отчет** (Subform/Subreport), и в его свойствах на вкладке **Данные** (Data) в строке **Объект-источник** (Source Object) указывается имя встроенного отчета, а в строках **Подчиненные поля** (Link Child Fields), **Основные поля** (Link Master Fields) указываются поля связи основного и подчиненного отчетов.

## Доработка составного отчета

В режиме конструктора и макета в составном отчете доступны для редактирования как основной, так и подчиненный отчеты.



**Рис. 7.38.** Отчет ТОВАР в режиме конструктора со встроенным подчиненным отчетом ОТГРУЗКА

Над рамкой подчиненного отчета размещен элемент с его именем (Отгрузка). Удалим этот элемент.

Верхний колонтитул подчиненного отчета ОТГРУЗКА, в котором размещены подписи полей, не отображается при просмотре составного отчета. Поэтому для отображения заголовков столбцов табличной части отчета перенесем названия столбцов из верхнего колонтитула в заголовок группы ДАТА\_ОТГР. Удалим содержимое заголовка подчиненного отчета.

После этого в подчиненном отчете удалим оба колонтитула, вызвав контекстное меню в любом месте внутри подчиненного отчета и выполнив команду **Колонтитулы страницы** (Page Header/Footer).

Если размер подчиненного отчета не соответствует установленной ширине страницы или не позволяет отобразить часть данных, измените его, щелкнув по элементу управления подчиненного отчета и перетащив маркеры размера в нужные позиции.

Используя инструменты на вкладках лент **Конструктор** (Design), **Формат** (Format), **Упорядочить** (Arrange) в режиме макета, а также обращаясь к свойствам элементов отчета, откорректируем подписи, изменим формат полей и подписей, проведем недостающие линии.

Отчет ТОВАР в режиме конструктора после внедрения подчиненного отчета ОТГРУЗКА и доработки показан на рис. 7.38.

## Добавление текущей даты и номера страницы

При автоматическом создании отчета с помощью инструмента **Отчет** (Report) или с помощью мастера отчетов в отчет всегда включается текущая дата и номер страницы. Эти элементы управления можно перенести в нужный раздел отчета или удалить вовсе.

Для добавления в отчет текущей даты используется встроенная функция `Date()` из категории **Дата/время** (Date/Time), которая, как любое другое вычисляемое выражение, может быть создана с помощью кнопки **Поле** (Text Box) и записи в свободный элемент функции `=Date()`. В свойствах этого элемента на вкладке **Макет** (Format) в строке **Формат поля** (Format) можно выбрать формат отображения даты. Подпись этого поля можно удалить. В примере дата скопирована в область верхнего колонтитула главного отчета и будет выводиться на каждой странице отчета.

Для добавления номера страницы в области нижнего колонтитула мастер в свободном элементе сформировал выражение `="Страница " & [Page] & " из " & [Pages]`. Это выражение может быть записано в свободный элемент отчета с помощью построителя выражений, который вызывается в окне свойств на вкладке **Данные** (Data) в строке **Данные** (Control Source). В левой колонке нижней части окна построителя выберем папку **Общие выражения** (Common Expressions). В средней колонке — тип элемента выражения **Страница N из M** (Page N of M). В правой колонке отобразится выражение, которое и следует вставить в вычисляемый элемент отчета.

Существуют и другие способы формирования поля даты и номера страницы. Добавить в раздел отчета поле текущей даты и времени можно, выполнив в режиме макета на вкладке ленты **Конструктор** (Design) команду **Дата и время** (Date & Time). Установка в диалоговом окне **Дата и время** (Date and Time) флажков **Формат даты** (Include Date) и/или **Формат времени** (Include Time) позволяет вставить текущую дату и/или текущее время и выбрать нужный вариант форматов (рис. 7.39).

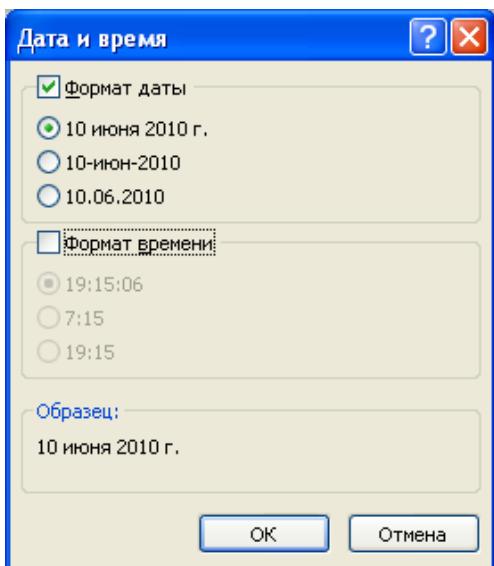


Рис. 7.39. Окно выбора формата даты и/или времени

В результате в отчете будет создано поле, в свойствах которого на вкладке **Данные** (Data) в строке **Данные** (Control Source) будет записано выражение `=Date()`.

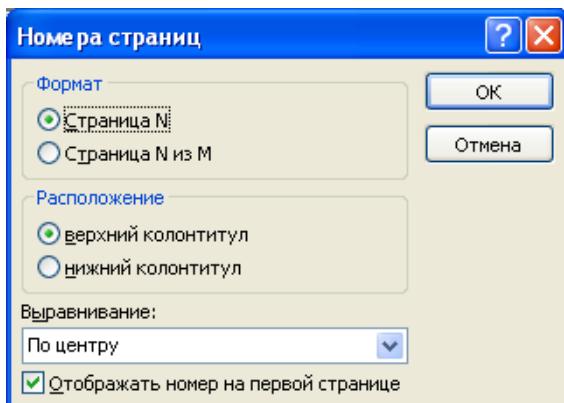


Рис. 7.40. Окно выбора параметров для номеров страниц

Добавить в отчет поле нумерации страниц можно, выполнив команду **Номера страниц** (Insert Page Number). В диалоговом окне **Номера страниц** (Page Numbers) выбираются параметры, определяющие формат, расположение и выравнивание

номеров страниц (рис. 7.40). Для печати номера страницы на первой странице устанавливается флагок **Отображать номер на первой странице** (Show Number on First Page).

## Просмотр отчета

Перейдем от конструирования к просмотру отчета, содержащего подчиненный отчет, щелкнув на кнопке **Предварительный просмотр** (Report View) в строке состояния. Страница составного отчета об отгрузке товаров в режиме предварительного просмотра представлена на рис. 7.41. В отчете записи об отгрузке товара сгруппированы по месяцам, причем учтено, что сведения об отгрузках могут сохраняться не один год.

ТОВАР

10 июня 2010 г.

Отгрузка товара Монитор 17LG Ко T001

Цена 7 000,00р.

Единица измерения штука

Ставка НДС 5%

Наличие товара

Остаток 420



Январь 2010					
Дата отгрузки	Код накладной	Код склада	Код товара	Количество	Сумма отгружено
18.01.2010	H001	C01	T001	50	350 000,00р.
21.01.2010	H004	C02	T001	10	70 000,00р.
Итого за месяц				60	420 000,00р.

Февраль 2010					
Дата отгрузки	Код накладной	Код склада	Код товара	Количество	Сумма отгружено
11.02.2010	H002	C02	T001	30	210 000,00р.
11.02.2010	H002	C01	T001	40	280 000,00р.
Итого за месяц				70	490 000,00р.

Март 2010					
Дата отгрузки	Код накладной	Код склада	Код товара	Количество	Сумма отгружено
24.03.2010	H002	C03	T001	30	210 000,00р.
25.03.2010	H004	C01	T001	2	14 000,00р.
Итого за месяц				32	224 000,00р.
Общий итог по товару				162	1 134 000,00р.

Страница 2 из 18

Страница: 14 2 ► ▶ ⟲ ⟳ 🔍 Нет фильтра ⟲ ⟳ ⟲ ⟳ ⟲ ⟳ ⟳ ⟳ ⟳

Рис. 7.41. Отчет об отгрузке товаров в режиме предварительного просмотра

Чтобы сведения об отгрузках каждого товара отображались с новой страницы, установите для области данных главного отчета в свойстве **Конец страницы** (Force New Page) значение **До раздела** (Before Section).

Вывод отчета на печать можно осуществить нажатием кнопки **Печать** (Print) на вкладке ленты **Предварительный просмотр** (Report View).

В составном отчете реализуется режим, при котором запись главного отчета выводится и в том случае, когда для нее нет связанных записей в подчиненном отчете. В рассматриваемом примере отчет будет включать и те товары, для которых не было отгрузок.

Если аналогичный рассмотренному отчет построить, исходя из предварительно созданного запроса на основе трех таблиц, можно исключить вывод записей о товарах, по которым не было отгрузки, или, наоборот, включить в отчет только их. Для этого достаточно в запросе определить нужный способ объединения записей таблиц ТОВАР и ОТГРУЗКА. Создание аналогичных запросов рассмотрено в *главе 4*.

## Вывод значений нарастающим итогом

В отчетах можно вывести значения некоторого поля записи или итогового поля группировки нарастающим итогом. В рассматриваемом примере можно накапливать количество каждого отгруженного товара от месяца к месяцу. Так, если в первый месяц количество отгруженного равно 12, во второй — 4, в третьей — 7, используя свойство поля **Сумма с накоплением** (Running Sum) со значением **Для всего** (Over All), можно получить значения суммы с накоплением: для первого месяца 12, для второго 16, для третьего 23. Это свойство размещено на вкладке **Данные** (Data).

### Задание 7.2. Создание отчета о заказанных товарах

Создайте отчет. В отчет включите наименование товара, сроки его поставки, количество заказанного и стоимость нарастающим итогом (от месяца к месяцу), а также сведения о покупателях, заказавших товар. Разработайте отчет на основе запроса на выборку. В запросе используйте таблицу ПОСТАВКА\_ПЛАН, в которой содержатся сведения обо всех заказах товара и указан срок поставки, таблицу ТОВАР, а также таблицы ДОГОВОР и ПОКУПАТЕЛЬ. В отчете выполните группировку по наименованию товара и сроку поставки. Для накапливания суммарных значений количества и стоимости используйте свойство **Сумма с накоплением** (Running Sum). Выполните фильтрацию данных отчета по покупателю, товару, месяцу поставки.

## Контрольные вопросы

1. Из каких разделов состоит отчет?
2. Назовите основные вкладки ленты, используемые при конструировании отчета?

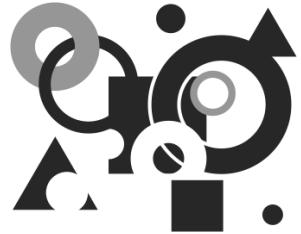
3. Как просмотреть список доступных в конструкторе отчетов полей и вставить нужное поле в отчет?
4. Какое значение размещается в подписи поля при его перетаскивании в отчет?
5. Возможно ли выполнить в отчете группировку записей, не отображая заголовка и (или) примечания группы?
6. В каком порядке сортируются группы по умолчанию?
7. Чем отличается группировка в запросе от группировки, выполненной в отчете?
8. Где целесообразно размещать значение поля, по которому производится группировка?
9. Где должно размещаться вычисляемое поле с итоговым значением, рассчитанным для группы?
10. Какая функция позволяет включить в отчет дату?
11. Какой элемент, по какой кнопке создается для размещения в нем вычисляемого поля?
12. Какая вкладка ленты позволяет выбрать размер страницы отчета, ее поля?
13. Можно ли с помощью перетаскивания включить в отчет подчиненный отчет?
14. В каком режиме должен быть открыт отчет, чтобы в него можно было включить подчиненный?
15. Нужно ли устанавливать связь между главным и подчиненным отчетами, если связь соответствующих таблиц определена в схеме данных?
16. В свойствах какого элемента сохраняются поля связи отчетов?
17. Какое условие должно быть выполнено, чтобы при включении подчиненного отчета с помощью кнопки **Подчиненная форма/отчет** (Subform/Subreport) заработал мастер?
18. В какой строке свойств подчиненного отчета указывается источник данных?
19. В какой строке свойств отчета указывается источник данных?
20. В каком месте отчета нужно щелкнуть мышью, чтобы сразу открыть его свойства?
21. Какой элемент отчета должен быть выделен, чтобы просмотреть свойства подчиненного отчета?
22. На основе каких объектов строит отчет мастер?
23. Возможен ли выбор полей из различных таблиц и запросов при построении отчета мастером?
24. Сколько уровней группировки может определить мастер в отчете?
25. В каком случае мастер не выведет кнопку **Итоги** (Summary Options), которая позволяет указать, какие именно итоговые значения нужно вывести в отчете?
26. Какие функции позволяет использовать мастер для подведения итогов?
27. Что будет использовано в качестве источника записей при построении отчета мастером на основе нескольких взаимосвязанных таблиц?
28. Как просмотреть созданную мастером инструкцию SQL в конструкторе запросов?
29. Чем определяется состав полей, доступных в многотабличном отчете, построенном мастером?

30. Каким свойством надо воспользоваться для того, чтобы каждая группа печаталась с новой страницы?
31. Какое свойство поля записи или итогового поля нужно использовать, чтобы выводить его значения нарастающим итогом?
32. За счет чего повторяющиеся значения в таблице запроса могут быть отображены в отчете только один раз?
33. Будет ли при просмотре отчета выводиться диалоговое окно ввода параметра, если отчет создан на запросе с параметром?
34. В какой строке бланка запроса указывается наименование параметра?
35. Как просмотреть список полей, доступных в отчете?
36. Какие поля содержит этот список, если отчет построен на запросе?
37. Какое значение получит вычисляемое поле, если в него записана встроенная функция управления `if([СУММА_ПОСТ]>100000; 'Особое внимание'; '')`, а в текущей записи поле СУММА ПОСТАВКИ имеет значение "70000"?

## Ответы

1. Заголовка отчета, верхнего колонтитула, заголовка группы, области данных, а также примечания группы, нижнего колонтитула и примечания отчета.
2. **Инструменты конструктора отчетов** (Report Design Tools), **Работа с макетами отчетов | Конструктор** (Report Layout Tools | Design), **Формат** (Format), **Упорядочить** (Arrange), **Параметры страницы** (Page Setup).
3. В режиме макета или конструктора отчетов нажать на панели инструментов кнопку **Добавить поля** (Add Existing Fields). Перетащить нужное поле в раздел отчета.
4. Подпись берется из свойства поля, заданного в таблице.
5. Нет. Только выбор одного из этих разделов приводит к выделению групп с одинаковыми значениями в выбранном поле.
6. По возрастанию.
7. В отчете записи группы не объединяются в одну запись. Один раз выводятся только поля, по которым осуществляется группировка.
8. В заголовке группы.
9. В примечании группы.
10. Встроенная функция `Date()`.
11. Свободный элемент создается по кнопке **Поле** (Text Box).
12. **Параметры страницы** (Page Setup).
13. Да.
14. В режиме конструктора.
15. Нет.
16. В свойствах элемента **Подчиненная форма/отчет** (Subform/Subreport) на вкладке **Данные** (Data).

17. Должна быть включена кнопка **Использовать мастера** (Use Control Wizards).
18. **Объект-источник** (Source Object).
19. **Источник записей** (Source Records).
20. На пересечении линеек.
21. **Подчиненная форма/отчет** (Subform/Subreport).
22. Таблиц и запросов.
23. Да.
24. 4.
25. Если в отчете не предусмотрена группировка или не включено ни одно числовое поле.
26. Sum, Avg, Min, Max.
27. Инструкция SQL, запрос.
28. Щелчком мыши на кнопке **Построитель** (Builder) в строке **Источник записей** (Record Source) свойств отчета.
29. Перечисленными в инструкции SQL полями.
30. Свойством заголовка группы **Конец страницы** (Force New Page) на вкладке **Макет** (Format).
31. Свойство **Сумма с накоплением** (Running Sum).
32. За счет группировки.
33. Да.
34. В строке **Условие отбора** (Criteria).
35. Открыв отчет в режиме макета или конструктора и выполнив на вкладке **Конструктор** (Design) команду **Добавить поля** (Add Existing Fields).
36. Все поля запроса.
37. Пустая строка.



## ГЛАВА 8

# Разработка приложений с использованием макросов

Как показано в предыдущих главах книги, база данных Access это не только набор таблиц, хранящих взаимосвязанные данные, но и набор средств для работы с данными, таких как запросы, формы, отчеты. При разработке приложения с помощью Access эти объекты составляют его основу, а формы и отчеты называются объектами приложения. При этом формы составляют основу интерфейса, обеспечивая интерактивный ввод, просмотр и изменение данных. Может быть создана форма для управления приложением, представляющая все его подзадачи и обеспечивающая пользователя простым доступом ко всем его функциям. Отчеты обеспечивают обобщенное представление результатов обработки данных и их вывод на экран или печать.

Для автоматизации выполнения задач, связи различных объектов, создания, редактирования и автоматизации более сложной логики приложений необходимо использовать средства программирования. Объектно-ориентированный язык программирования VBA (Visual Basic for Applications), являясь общим средством программирования для всего семейства Microsoft Office, позволяет создать в среде Access целостные графические диалоговые приложения пользователя с большими возможностями по управлению и контролю за их выполнением, решить и автоматизировать выполнение самых сложных задач.

Простейшим языком программирования, позволяющим добавлять функциональные возможности и автоматизировать выполнение задач приложения, является язык макросов. Использование макросов для автоматизации управления реакцией приложения на действия пользователя в формах или отчетах позволяет создавать полноценные интерактивные приложения без написания кода на VBA.

Макросы обеспечивают выполнение части команд, доступных в VBA, и для большинства пользователей создание макроса оказывается проще, нежели написание кода VBA. Новый конструктор макросов в Access 2010 с ясным и понятным интерфейсом, поддерживающий функции IntelliSense, позволяет не только избежать написания кода VBA, но и упростить процесс добавления функциональных возможностей в приложение базы данных.

В веб-приложениях Access, базирующихся на базах данных, опубликованных в SharePoint, для программирования необходимо использовать только макросы, так как код VBA несовместим со средствами веб-публикации.

Новое средство Access 2010 — макросы данных обеспечивают добавление логики к данным в исходных таблицах. Для связи макросов данных с действиями по добавлению, обновлению, удалению записей в таблице достаточно в открытой таблице на вкладке **Работа с таблицами | Таблица** (Table Tools | Table) щелкнуть по нужной кнопке (см. рис. 8.22). Сосредоточение с помощью макросов данных бизнес-логики в таблицах позволяет распространить автоматизацию за пределы клиентского приложения Access на веб-базы данных SharePoint и другие приложения, обновляющие таблицы Access.

*Макрос* — программа, состоящая из последовательности макрокоманд (макрос от слова "макрокоманда"). *Макрокоманда* — это инструкция, ориентированная на выполнение определенного действия над объектами Access и их элементами.

Например, макрокомандой можно открыть форму, отчет, напечатать отчет, запустить на выполнение запрос, применить фильтр, присвоить значение, создать свое меню, организовать выполнение различных ветвей алгоритма в зависимости от условий. Макрокоманда **ЗапускКомандыМеню** (RunMenuItemCommand) позволяет выполнить любые встроенные команды Access, которые выводятся на вкладках ленты или в контекстных меню и соответствуют режиму выполнения макрокоманды. Имеющийся в Access набор макрокоманд (более 50) реализует практически любые действия, которые необходимы для решения задачи.

Макрос может быть наряду с другими объектами представлен как отдельный объект (изолированный макрос), который отображается в области навигации в группе **Макросы** (Macros). Кроме того, макрос, связанный с любым событием в форме, отчете или элементе управления, может быть внедрен в форму или отчет (внедренный макрос). При этом он не отображается как объект в группе **Макросы** (Macros), а становится компонентом формы или отчета.

Макросы могут запускаться на выполнение прямо из области навигации. Возможно решение задач с помощью ряда взаимосвязанных макросов, первый из которых будет запускаться на выполнение из области навигации. Пользователь запускает главный макрос на выполнение и далее все управление выполнением задачи осуществляется изнутри макроса. Макрос сам открывает нужные объекты, выбирает и обрабатывает данные, вызывает другие макросы, следуя алгоритму, приводящему к решению задачи. При необходимости из макроса может быть иницирован диалог с пользователем. Для перехода по различным ветвям макроса используется блок управления **Если** (If).

Изолированный макрос может выполняться в ответ на многочисленные виды событий, возникающих в формах, отчетах и их элементах управления. Внедренный макрос всегда связывается с событием и сохраняется в форме или отчете. События наступают, прежде всего, при выполнении определенных действий пользователя с объектами. Примерами событий являются: изменение данных в поле, открытие или закрытие формы или отчета, нажатие кнопки в форме и просто передача фокуса от

одного поля к другому. Связь макросов с событиями позволяет автоматизировать приложения, используя макросы для открытия форм, печати отчетов, выполнения последовательности запросов, для выполнения действий, зависящих от значений некоторого поля в базе данных, для вывода пользовательских сообщений или отключения предупреждающих сообщений во время выполнения запросов действия и многое другое. Сохранение внедренных макросов вместе с формами и отчетами упрощает управление объектами приложения.

Программы на языке макросов реализуют алгоритмы решения отдельных задач приложения. Механизм связывания макросов с событиями в объектах позволяет объединить разрозненные задачи приложения в единый комплекс, управляемый пользователем. Пользователь, выполняя различные действия, прежде всего в формах, инициирует выполнение макросов, автоматизирующих решение связанных с действиями пользователя задач.

## Конструирование макроса

Создание макросов осуществляется в диалоговом режиме и сводится к записи в окне конструктора макроса последовательности макрокоманд, для которых задаются аргументы. Каждому макросу присваивается имя. При выполнении макроса макрокоманды выполняются последовательно в порядке их расположения. При этом используются объекты или данные, указанные в аргументах макрокоманд. Для изменения порядка выполнения макрокоманд может быть использован логический блок управления **Если** (If).

Выполнение макросов инициируется простой операцией и может сводиться к его открытию, как это делается и для других объектов базы данных. Помимо этого, Access предоставляет возможность автоматически инициировать выполнение макроса при наступлении некоторого события. Для связи макроса с событием достаточно в окне свойств объекта или его элемента управления внести в строку этого события имя макроса или создать внедренный макрос. События, с которыми можно связать макрос, определяются в свойствах форм и отчетов и их элементов управления.

Создание изолированного макроса, являющегося отдельным объектом базы данных, начинается с выполнения команды **Макрос** (Macro) на вкладке ленты **Создание** (Create) в группе **Макросы и код** (Macros & Code). В результате выполнения команды открывается окно макроса и каталог макрокоманд (см. рис. 8.1).

Для создания внедренного макроса выберите в области навигации форму или отчет и выполните в его контекстном меню команду **Конструктор** (Design) или **Режим макета** (Layout View). Далее откройте окно свойств и выберите элемент управления, раздел или форму (отчет) целиком, на вкладке **События** (Event) выберите событие, в которое нужно встроить макрос, затем нажмите кнопку построителя  . В диалоговом окне **Построитель** (Choose Builder) выделите пункт **Макросы** (Macro Builder) и затем нажмите кнопку **OK**. В результате откроется окно макроса, точно такое же, как при создании изолированного макроса.

Внедренные макросы отличаются от изолированных макросов тем, что они хранятся в формах и отчетах. Они не отображаются в виде объектов в группе **Макросы** (Macros) в области навигации. Это упрощает управление базой данных, поскольку не нужно следить за тем, какие макросы относятся к какой форме или отчету. Внедренные макросы сохраняются в составе формы или отчета и при их копировании, импорте или экспорте.

## Формирование макрокоманд в окне макроса

При создании нового макроса в его окне отображается поле **Добавить новую макрокоманду** (Add New Action) с раскрывающимся списком (рис. 8.1). В списке представлен весь доступный набор макрокоманд. Этот набор можно изменять, нажимая кнопку **Показать все действия** (Show All Action) на ленте конструктора макросов в группе **Показать или скрыть** (Show/Hide). Если кнопка не нажата, то в набор не включаются так называемые небезопасные макрокоманды. К ним относятся макрокоманды, изменяющие базу данных или получающие доступ к ресурсам вне базы. Это позволяет пользователю при необходимости исключить использование небезопасных макрокоманд в своем приложении и таким образом предоставить возможность открывать базу данных с полным набором функций, даже если она не получила статус доверенной. Доверенной база данных становится, например, в случае, если она получена из источника, включенного в список надежных.

Для постоянного отображения в окне конструктора макросов списка всех доступных макрокоманд нужно нажать кнопку **Каталог макрокоманд** (Action Catalog). В окне каталога для простоты поиска макрокоманды разбиты на функциональные группы и обеспечена очень удобная возможность поиска макрокоманды по имени (см. рис. 8.1). Кроме того, в каталоге имеется раздел **В этой базе данных** (In this Database), где представлены не только все макросы из области навигации, но и макросы, внедренные в формы и отчеты. Содержимое этих макросов также может быть скопировано в другие конструируемые макросы.

Для ввода макрокоманды в поле можно нажать кнопку раскрытия списка макрокоманд и выбрать нужную. Можно ввести имя макрокоманды с клавиатуры, при этом система помогает сформировать его. Можно в каталоге выбрать макрокоманду и перетащить ее в окно макроса. Место размещения новой макрокоманды отмечается оранжевой полосой.

Окно конструирования внедренного макроса отличается от окна конструирования изолированного макроса только записью в заголовке окна. Для изолированного макроса там записано имя макроса, для внедренного — имя формы (отчета): имя элемента: свойство события, например, ТОВАР: ЦЕНА: После обновления.

После ввода макрокоманды в макросе отображается блок, содержащий имя макрокоманды и строки ее аргументов (рис. 8.2). Значения аргументов задаются путем выбора их из списка, открывающегося в строке аргумента, с помощью построителя или вручную. Для выбранного аргумента или макрокоманды выводится всплывающая подсказка.

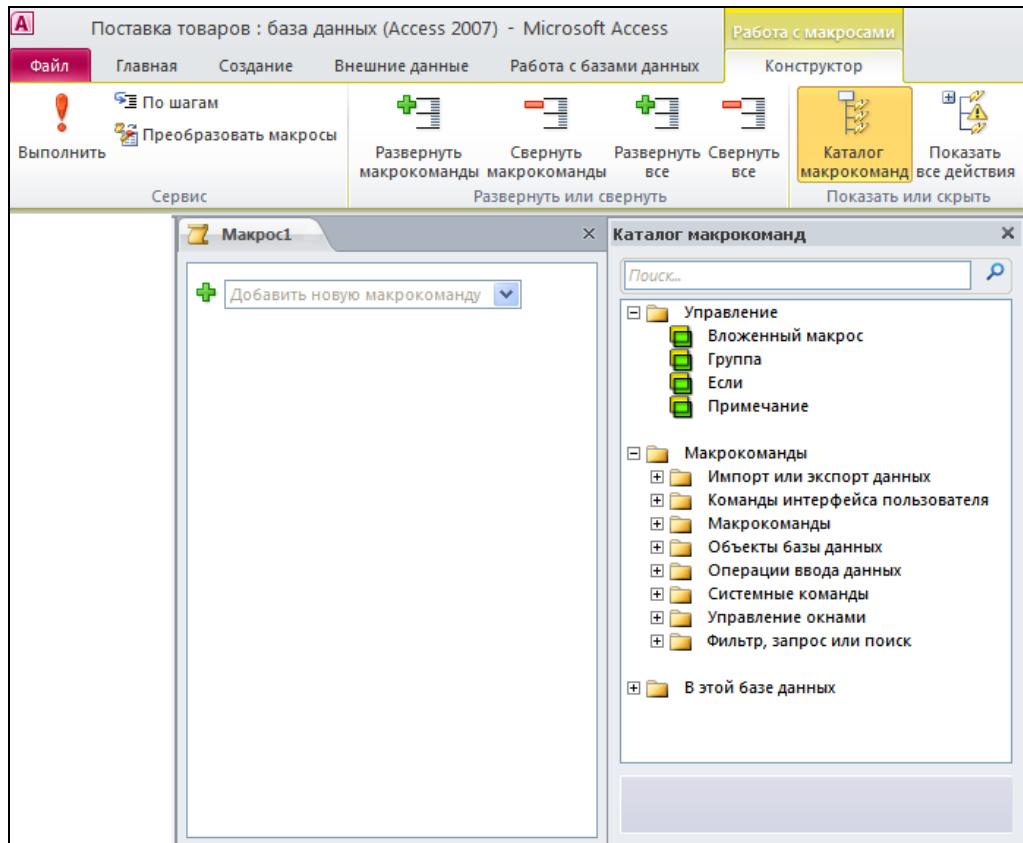


Рис. 8.1. Окно конструирования макроса с Каталогом макрокоманд

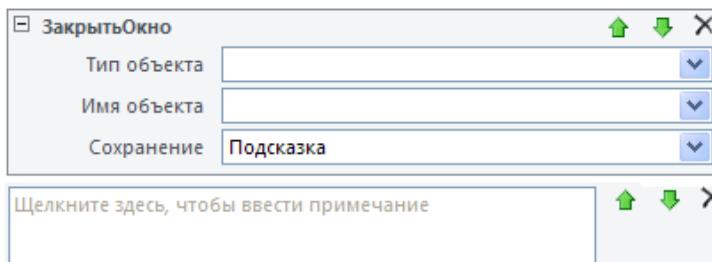


Рис. 8.2. Блок конструирования макрокоманды и ввода комментария

Знак минус слева от имени макрокоманды позволяет скрыть ее аргументы. Зеленые стрелки в правой части блока позволяют перемещать макрокоманду выше или ниже других макрокоманд. Здесь же имеется значок удаления макрокоманды.

Для ввода комментария в макрос используется блок **Примечание** (Comment), размещенный в окне каталога макрокоманд в разделе **Управление** (Program Flow).

При перетаскивании этого блока в макрос создается пустой блок, в который и вводится нужный комментарий. После завершения ввода комментарий отображается строкой зеленого цвета, заключенной в знаки /\* и \*/.

Каждая новая макрокоманда макроса добавляется в нужное место макроса и проще всего это сделать перетаскиванием ее из каталога макрокоманд. Порядок размещения макрокоманд в бланке определяет последовательность их выполнения.

После ввода всех макрокоманд в макрос его надо сохранить, воспользовавшись командой **Сохранить** (Save).

В режиме конструктора для выполнения макроса предназначена команда **Выполнить** (Run). Если макрос уже открыт, то для его выполнения надо в области навигации выбрать макрос и в контекстном меню выбрать **Выполнить** (Run). Кроме того, для выполнения макроса можно использовать кнопку **Выполнить макрос** (Run Macro) на вкладке ленты **Работа с базами данных** (Database Tools) в группе **Макрос** (Macros). Таким образом, реализованная макросом задача может решаться по инициативе пользователя.

Выполнение внедренного макроса происходит автоматически при наступлении события, с которым он связан.

Изолированный макрос так же как и внедренный может быть связан с любым событием формы (отчета) или их элементов. Для этого нужно, вместо того чтобы вызывать построитель для создания макроса, выбрать ранее созданный изолированный макрос из списка в строке свойства события.

Для просмотра и редактирования существующего изолированного макроса надо выбрать его в области навигации и в контекстном меню нажать кнопку **Конструктор** (DesignView). Для редактирования внедренного макроса нужно также как при его создании воспользоваться кнопкой построителя в строке свойства события.

## Формирование макроса с помощью мыши

Создание некоторых макрокоманд в макросе может быть выполнено путем переноса с помощью мыши объекта базы данных из области навигации в окно макроса. Макрокоманда по умолчанию создается со значениями аргументов, соответствующими выбранному объекту. Например, при перетаскивании таблицы создается макрокоманда **ОткрытьТаблицу** (OpenTable), открывающая таблицу в режиме таблицы со значением **Изменение** (Edit) для аргумента **Режим данных** (Data Mode).

Возможно перемещение любых объектов, представленных в области навигации базы данных. Соответственно перемещаемому объекту могут быть созданы макрокоманды **ОткрытьТаблицу** (OpenTable), **ОткрытьФорму** (OpenForm), **ОткрытьЗапрос** (OpenQuery), **ОткрытьОтчет** (OpenReport), **ОткрытьМодуль** (OpenModule) или **ЗапускМакроса** (RunMacro).

# Использование в макросах ссылок на объекты

В программах на языке макросов, как и на языке VBA, при обработке данных в базе необходимо уметь правильно ссылаться на эти данные. В Access определен ряд объектов, через которые предоставляется возможность получить доступ к данным базы. К таким объектам относятся формы и отчеты. Ссылки на формы, отчеты, их элементы управления и свойства формируются по определенным правилам. Построитель выражений позволяет сформировать такие ссылки простым выбором объекта, элемента управления и свойства в списках. Ко времени выполнения в макросах выражения со ссылкой на объект этот объект должен быть открыт.

## Ссылки на объекты и их элементы управления

Ссылаться на объекты можно по имени, но нужно учесть, что в Access объекты объединяются в семейства. Формы объединены в семейство **Формы** (Forms), отчеты — в семейство **Отчеты** (Reports). Поэтому ссылка на объект включает имя семейства и через восклицательный знак имя объекта. Если имя включает пробелы или специальные символы, его надо брать в квадратные скобки. Например, для ссылки на форму надо записать `Forms! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ]`.

В ссылке на элемент управления вслед за именем объекта через восклицательный знак записывается имя элемента управления, заключенное в квадратные скобки.

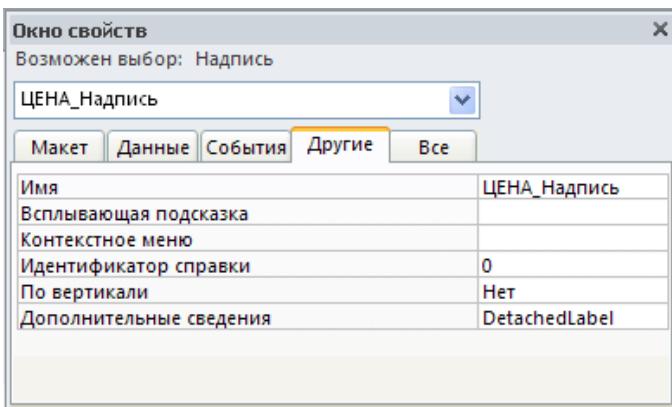


Рис. 8.3. Окно свойств элемента управления **Надпись**

Например, ссылку на поле в форме надо записать  
`Forms! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ]! [СУММА_ДОГ].`

Ссылку на элемент управления **Надпись** (Caption) с именем **ЦЕНА\_Надпись** надо записать  
`Forms! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ]! [ЦЕНА_Надпись]`

Обратите внимание, что имя этого элемента управления не совпадает с подписью, выводимой в форме. Имя записывается в свойстве элемента управления на вкладке **Другие** (Other) в строке **Имя** (Name) (рис. 8.3). В форме **ДОГОВОРЫ С ПОКУПАТЕЛЯМИ** элемент управления — **Надпись** (Caption) с именем **ЦЕНА\_Надпись** — отображается как **Цена**. Это значение записано на вкладке **Макет** (Format) в свойстве **Подпись** (Caption). Проверяйте имя элемента управления в окне свойств, оно может по умолчанию принимать и другие значения, например, **Надпись15**.

## Ссылки на свойство объекта

В ссылке на свойство объекта вслед за именем объекта через точку записывается имя свойства. Перечень свойств формы или отчета можно посмотреть, открыв объект в режиме конструктора или макета и вызвав окно его свойств. Многие свойства названы несколькими словами с пробелами между ними. Истинное имя свойства таких пробелов не имеет, поэтому в ссылках оно записывается без них. Например, имя свойства **Область выделения** (Record Selectors) должно записываться — **ОбластьВыделения** (RecordSelectors), имя свойства **Полосы прокрутки** (Scroll Bars) — **ПолосыПрокрутки** (ScrollBars).

Ссылку на это свойство формы надо записать

`Forms! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ].ScrollBars`

В русифицированной версии Access в строках окна свойств указаны русские имена свойств. В ссылках следует использовать английский вариант имени свойства.

## Ссылки на свойство элемента управления

Для записи ссылки на свойство элемента управления нужно дополнить ссылку на элемент управления через точку именем свойства. Различные типы элементов управления имеют разные свойства.

Например, ссылка на свойство **ВыводНаЭкран** (английский вариант имени **Visible**) элемента управления с именем **ЦЕНА\_Надпись**, которое соответствует строке **Вывод на экран** (Visible) на вкладке **Макет** (Format) (рис. 8.4), надо записать:

`Forms! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ]! [ЦЕНА_Надпись].Visible`

В общем виде правило записи ссылок может быть сформулировано следующим образом: оператор **!** указывает, что следующий за ним элемент является элементом, определяемым пользователем. Например, оператор **!** ставится перед ссылкой на открытую форму, на открытый отчет или элемент управления в открытой форме или отчете. Оператор **.** (точка) обычно указывает, что следующий за ним элемент определен в Access. Например, оператор **.** (точка) ставится перед ссылкой на свойства форм, отчетов и элементов управления.

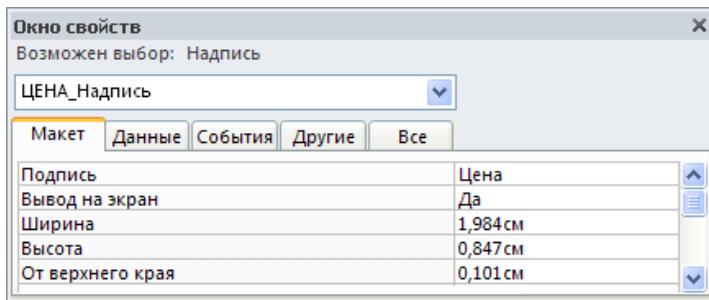


Рис. 8.4. Отображение свойства Visible в окне свойств

## Ссылка на подчиненную форму или отчет

Подчиненная форма (или отчет) рассматривается в Access как элемент управления формы (отчета). Форма после внедрения содержится в элементе управления **Подчиненная форма** (Subform), отчет — в элементе управления **Подчиненный отчет** (Subreport). Поэтому ссылка на подчиненную форму (или отчет) записывается как ссылка на элемент управления формы (отчета).

Элемент управления **Подчиненная форма/отчет** (Subform/Subreport) имеет специальное свойство **Форма** (Form) или, соответственно, **Отчет** (Report). Это свойство позволяет ссылаться на элементы управления подчиненных объектов и их свойства. Полная ссылка на свойство элемента управления в подчиненной форме имеет в общем виде следующую структуру:

```
Forms! [Имя_формы] ! [Элемент_Подчиненная_форма].Form! [Элемент_подчиненной_формы].Имя_свойства
```

Заметим, что при ссылках на элемент управления в подчиненной форме или подчиненном отчете не обязательно указывать свойство Form или Report.

Далее приведена ссылка на элемент управления — поле СУММА\_ПОСТ в подчиненной форме ПЛАН ПОСТАВОК, встроенной в форму ДОГОВОРЫ С ПОКУПАТЕЛЯМИ:

```
Forms! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ] ! [план поставок] ! [СУММА_ПОСТ]
```

*/\* Расчет новой суммы поставки текущего изделия в подчиненной форме \*/*

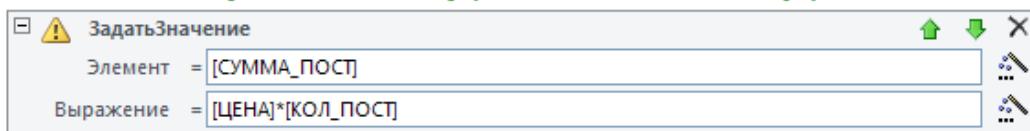


Рис. 8.5. Ссылки на имена элементов управления текущей формы ПЛАН ПОСТАВОК

Следует заметить, что не всегда нужно в ссылке использовать полное имя элемента управления. Так, для расчета стоимости товара в текущей записи подчинен-

ной формы ПЛАН ПОСТАВОК и его сохранения в поле СУММА\_ПОСТ достаточно записать в аргументах макрокоманды **ЗадатьЗначение** (SetValue): в строке **Элемент** (Item) имя [СУММА\_ПОСТ], а в строке **Выражение** (Expression) — [КОЛ\_ПОСТ] \* [ЦЕНА] (рис. 8.5).

Если макрос вызывается при наступлении некоторого события, например, при обновлении пользователем полей КОЛ\_ПОСТ или ЦЕНА в форме ПЛАН ПОСТАВОК, то при передаче управления в макрос текущим объектом остается эта форма. Поэтому в макросе для ссылок достаточно использовать только имена полей.

## **Создание ссылок построителем выражений**

Ссылки на объекты, элементы управления и свойства удобно создавать с помощью построителя выражений.

Чтобы вызвать построитель выражений, надо выбрать в окне макроса строку аргумента макрокоманды, в которую требуется ввести выражение, и нажать появившуюся кнопку построителя .

В открывшемся окне **Построитель выражений** (Expression Builder) в поле записи выражений надо сформировать выражение. Для этого в левом поле в нижней половине окна построителя раскройте двойным щелчком кнопки мыши папку, содержащую объекты. Затем выберите папку необходимого объекта. В среднем поле выберите нужный элемент управления объекта, а в правом — **Значение** (Value), если формируется ссылка на элемент управления, или нужное свойство и двойным щелчком вставьте ссылку в выражение.

Закончив создание выражения, нажмите кнопку **OK**.

Access скопирует созданное выражение в ту строку, из которой был вызван построитель выражений.

Если строка аргумента, из которой вызывается построитель выражений, уже содержит выражение, то оно автоматически копируется в поле выражений, где может быть отредактировано.

Заметим, если нужный объект или свойство не появляется в нижней части окна построителя выражений, это означает, что их нельзя использовать в том контексте, в котором был вызван построитель выражений.

## **Вложенные макросы**

Access 2010 предоставляет возможность создавать в макросе вложенные макросы. Вложенный макрос имеет имя и может содержать любые макрокоманды. В предыдущих версиях также предоставлялась возможность объединения макросов в группы. Работать с группой часто оказывается удобнее, чем с несколькими отдельными макросами, поскольку группа макросов в области навигации отображается как один объект. Целесообразно объединять несколько макросов в одном, если они связаны с решением одной задачи или используются при работе с одним объектом.

Вызывается вложенный макрос с помощью макрокоманды **ЗапускМакроса** (RunMacro) или в ответ на событие. Для ссылки на вложенный макрос используется следующий синтаксис:

`ИмяГруппыМакросов.ИмяВложенногоМакроса`

Для включения в макрос вложенного макроса нужно выбрать соответствующую макрокоманду из раскрывающегося списка в поле **Добавить новую макрокоманду** (Add New Action) или перетащить **Вложенный макрос** (Submacro) в нужное место из **Управление** (Program Flow) каталога макрокоманд. В макросе отобразится блок (рис. 8.6), в котором по умолчанию вложенному макросу присвоено имя Sub1, предоставлется возможность добавления макрокоманд и вставлен признак конца вложенного макроса.

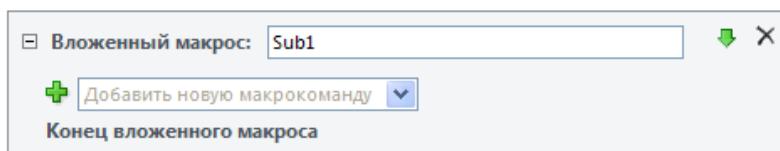


Рис. 8.6. Блок пустого вложенного макроса

Рассмотрим как с помощью вложенных макросов, которые запускаются при инициировании пользователем событий в одной форме, можно отфильтровать записи в другой форме.

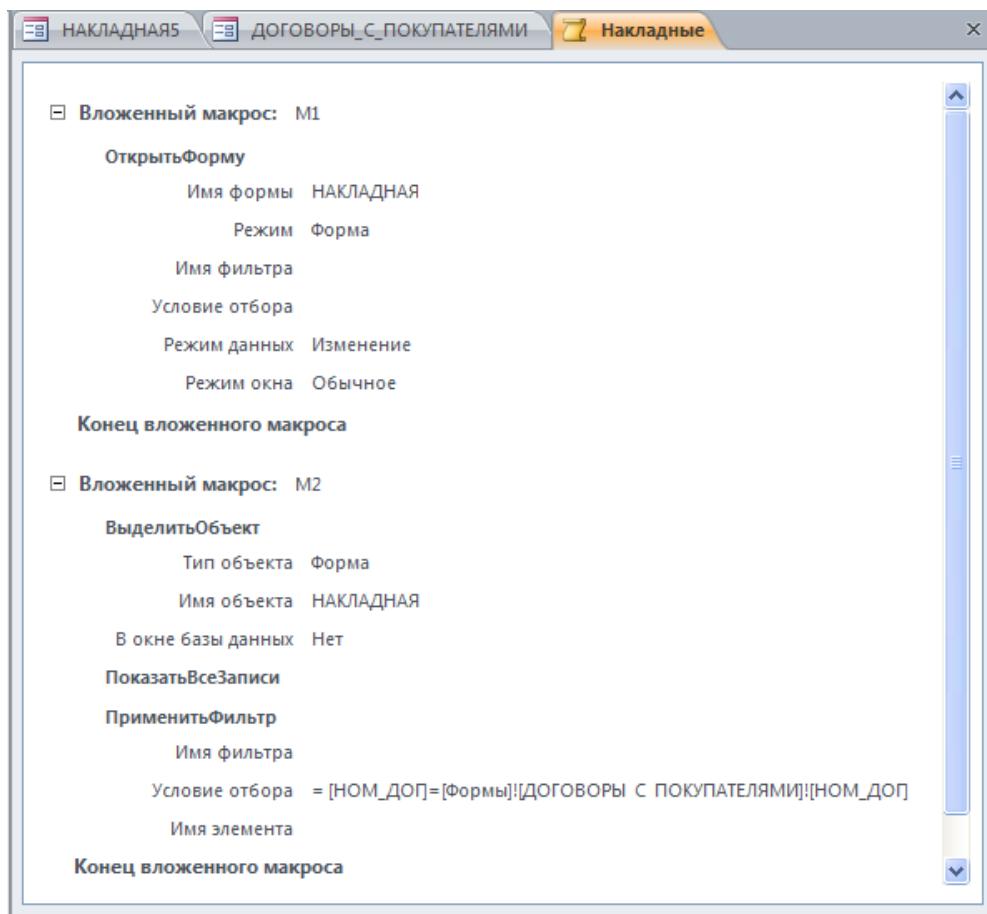
**Задача.** Пусть при просмотре данных о договорах необходимо иметь возможность отображать информацию о накладных, по которым отгружался товар в соответствии с просматриваемым договором.

Для решения задачи используем формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ и НАКЛАДНАЯ, созданные ранее (см. главу 5). Обе формы имеют поле с номером договора НОМ\_ДОГ. При работе с формой ДОГОВОРЫ С ПОКУПАТЕЛЯМИ должна открываться форма НАКЛАДНАЯ, в которой представлены только соответствующие просматриваемому договору накладные.

Создайте изолированный макрос с именем **Накладные**, включающий вложенные макросы **M1** и **M2**. Пусть макрос **M1** при открытии формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ открывает форму НАКЛАДНАЯ. Макрос **M2** фильтрует записи с накладными по значению поля НОМ\_ДОГ, взятыму из текущей записи формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ. Макрос **M2** должен выполняться, когда пользователь, работая в форме с договорами, инициирует событие **Двойное нажатие кнопки** (On Dbl Click) или **Вход** (Enter) для поля с номером договора. Создание группы позволяет объединить макросы, предназначенные для решения одной задачи, и упростить сопровождение приложения.

1. Для создания изолированного макроса выполните команду **Макрос** (Macro) на вкладке ленты **Создание** (Create) в группе **Макросы и код** (Macros & Code).
2. В открывшемся окне макроса с помощью макрокоманды управления **Вложенный макрос** (Submacro) создайте вложенные макросы с именами **M1** и **M2** (рис. 8.7).

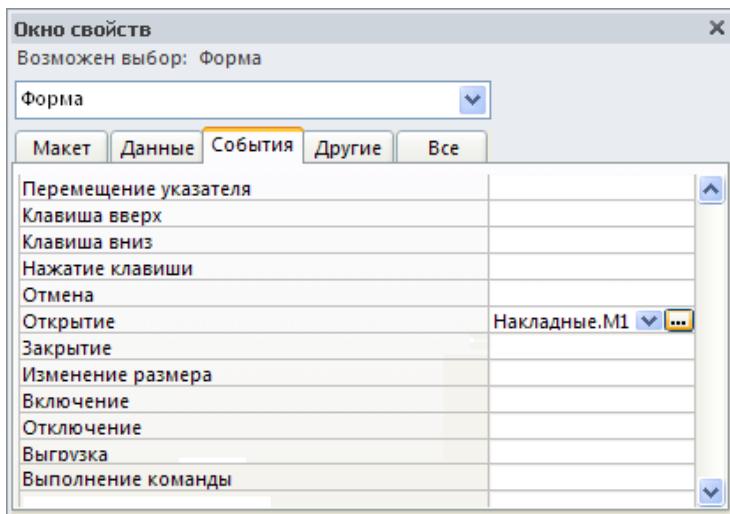
3. Макрос **M1** дополните макрокомандой **ОткрытьФорму** (OpenForm) НАКЛАДНАЯ.
4. Макрос **M2** дополните:
  - ◆ макрокомандой **ВыделитьОбъект** (SelectObject) НАКЛАДНАЯ, которая сделает объект текущим и позволит выполнять для него макрокоманды;
  - ◆ макрокомандой **ПоказатьВсеЗаписи** (ShowAllRecords), которая отменит фильтр, примененный ранее к форме;
  - ◆ макрокомандой **ПрименитьФильтр** (ApplyFilter), которая отфильтрует записи формы НАКЛАДНАЯ в соответствии с условием отбора.
5. Сохраните группу макросов под именем **Накладные**. Это имя будет выводиться в списке макросов в области навигации.



**Рис. 8.7.** Макрос, организующий синхронный просмотр данных в двух формах

6. Для запуска макроса **M1** установите связь события **Открытие** (On Open) формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ с макросом. Откройте форму в режиме макета или конструктора, щелкните на вкладке ленты конструктора в группе

**Сервис** (Tools) по кнопке **Страница свойств** (Property Sheet). В окне свойств формы на вкладке **События** (Event) в качестве значения свойства **Открытие** (On Open) выберите из списка имя макроса **Накладные.М1** (рис. 8.8).



**Рис. 8.8.** Связь вложенного макроса со свойством формы  
ДОГОВОРЫ С ПОКУПАТЕЛЯМИ

7. Для запуска макроса **М2** в свойствах поля **НОМ\_ДОГ** формы **ДОГОВОРЫ С ПОКУПАТЕЛЯМИ** в качестве значения события **Двойное нажатие кнопки** (On Dbl Click) выберите **Накладные.М2**.  
Событие **Двойное нажатие кнопки** (On Dbl Click) для элемента управления **НОМ\_ДОГ** возникает, когда пользователь дважды нажимает и быстро отпускает левую кнопку мыши в тот момент, когда курсор мыши установлен на поле **НОМ\_ДОГ** или присоединенной к нему надписи.
8. Чтобы проверить работу макросов, откройте форму **ДОГОВОРЫ С ПОКУПАТЕЛЯМИ**. На экране будет отображен результат работы макроса **М1** — открытая форма **НАКЛАДНАЯ**. Причем в форме будут доступны все записи.
9. Выполните двойной щелчок по полю с номером договора в форме **ДОГОВОРЫ С ПОКУПАТЕЛЯМИ**. Результат выполнения макроса **М2** отобразится в форме **НАКЛАДНАЯ**. К ней применен фильтр, в котором в качестве условий отбора записано **[НОМ\_ДОГ] =Формы! [ДОГОВОРЫ С ПОКУПАТЕЛЯМИ] ! [НОМ\_ДОГ]**, и поэтому доступны только записи со значением номера договора из формы **ДОГОВОРЫ С ПОКУПАТЕЛЯМИ**.

### **ЗАМЕЧАНИЕ**

Для ссылки на элемент управления в текущем объекте достаточно указать только имя элемента, для ссылки на элемент управления других открытых объектов должен исполь-

зоваться полный синтаксис. Пользуйтесь построителем для формирования сложной ссылки на элемент. Построитель вызывается щелчком на кнопке в правой части строки аргумента.

10. Чтобы макрос **M2** выполнялся, когда пользователь, работая в форме ДОГОВОРЫ С ПОКУПАТЕЛЯМИ, входит в поле НОМ\_ДОГ:

- ◆ удалите связь макроса **M2** с событием **Двойное нажатие кнопки** (On Dbl Click) для элемента управления НОМ\_ДОГ;
- ◆ установите его связь с событием **Вход** (Enter) этого же элемента управления.

Событие **Вход** (Enter) наступает при перемещении курсора в поле НОМ\_ДОГ, при переходе к другой записи, если текущим является поле НОМ\_ДОГ. Кроме того, при открытии формы наряду с событием **Открытие** (On Open) для текущего элемента управления формы наступает событие **Вход** (Enter). При открытии формы текущим элементом управления является поле, которое стоит первым в списке, определяющем последовательность перехода по элементам управления области данных. Если поле НОМ\_ДОГ размещено в области данных формы и стоит на первом месте в списке, то оно будет текущим и для него наступит событие **Вход** (Enter). В результате при открытии формы выполняются оба макроса.

11. Просмотрите последовательность перехода по элементам управления области данных в форме ДОГОВОРЫ С ПОКУПАТЕЛЯМИ, выполнив команду **Переходы** (Tab Order) на вкладке **Инструменты конструктора форм | Конструктор** (Form Design Tools | Design) в группе **Сервис** (Tools).

12. Если поле НОМ\_ДОГ в списке последовательности переходов стоит не на первом месте, переместите его.

### **ВНИМАНИЕ!**

Одно действие пользователя может приводить к возникновению нескольких событий. При этом важно знать последовательность наступления этих событий, т. к. это определяет и последовательность выполнения макросов.

13. Если в форме ДОГОВОРЫ С ПОКУПАТЕЛЯМИ поле НОМ\_ДОГ размещено в области данных и стоит первым в списке последовательности переходов, то при открытии формы и при переходе от записи к записи синхронно отображаются соответствующие записи формы НАКЛАДНАЯ. Курсор должен оставаться в поле НОМ\_ДОГ. Убедитесь в этом.

## **Внедренный макрос**

Внедренный макрос всегда связывается с событием и сохраняется в форме или отчете. Рассмотрим создание внедренного макроса на примере. В базе данных **Поставка товаров** для просмотра, корректировки и ввода данных по договорам создана многотабличная форма ДОГОВОРЫ С ПОКУПАТЕЛЯМИ. Она состоит из

главной формы, построенной на таблицах ДОГОВОР и ПОКУПАТЕЛЬ, и подчиненной формы, построенной на таблицах ПОСТАВКА\_ПЛАН и ТОВАР и представляющей спецификацию договора.

**Задача.** Пусть необходимо при изменении значения поля КОЛ\_ПОСТ (количество поставки), а также при добавлении или удалении строк спецификации договора автоматизировать расчет суммы поставки (поле СУММА\_ПОСТ) в каждой строке и общей суммы по договору (поле СУММА\_ДОГ) с сохранением результатов в таблицах ПОСТАВКА\_ПЛАН и ДОГОВОР.

1. Для расчета и обновления полей СУММА\_ПОСТ и СУММА\_ДОГ в таблицах ПОСТАВКА\_ПЛАН и ДОГОВОР при изменении в форме значения поля КОЛ\_ПОСТ (оно размещается в подчиненной форме) создайте следующий внедренный макрос для обработки события **После обновления** (AfterUpdate) для этого поля (рис. 8.9).

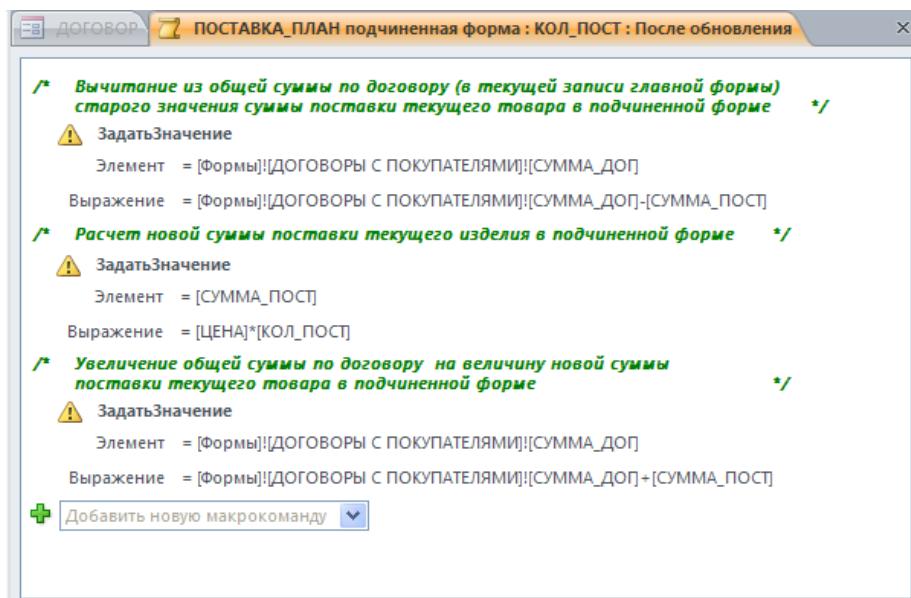
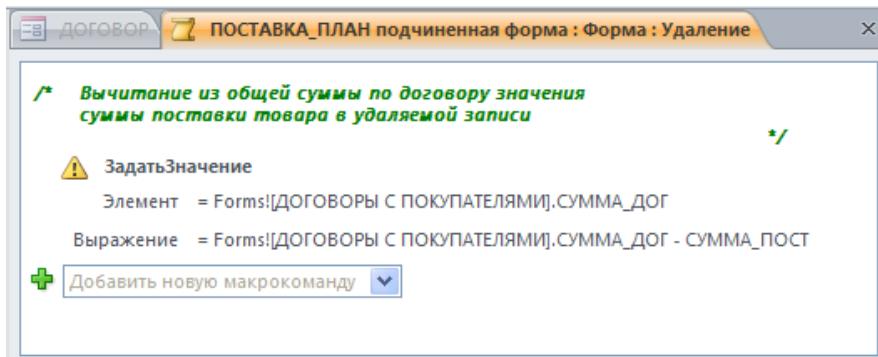


Рис. 8.9. Окно конструирования внедренного макроса

2. Убедитесь, что этот макрос правильно производит расчет также при добавлении новой строки в подчиненную форму.
3. Для расчета и обновления данных в таблице ДОГОВОР при удалении строки из подчиненной формы должен быть создан внедренный макрос обработки события **Удаление** (Delete) для подчиненной формы.
4. Для перехода к созданию этого макроса щелкните мышью на подчиненной форме и в контекстном меню откройте ее свойства. На вкладке **События** (Events) в строке **Удаление** (Delete) щелкните на значке **Построитель** (Choose Builder), выберите пункт **Макросы** (Macro Builder) и затем нажмите кнопку

**OK.** В открывшееся окно макроса введите макрокоманду **ЗадатьЗначение** (SetValue) (рис. 8.10).



**Рис. 8.10.** Внедренный макрос обработки события Удаление

Убедитесь в том, что после сохранения внедренного макроса его имя отобразится в окне **Каталог макрокоманд** (Action Catalog) в разделе **В этой базе данных** (In this Database) под именем формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ.

### **ВНИМАНИЕ!**

Чтобы отобразить полный список макрокоманд, выполнение которых возможно только в базе данных со статусом доверенной, щелкните на кнопке **Показать все действия** (Show All Actions). Когда доступен полный список макрокоманд, кнопка **Показать все действия** (Show All Actions) выделена. Небезопасные макрокоманды помечены в списке каталога макрокоманд и в теле макроса значком .

## **Управление последовательностью выполнения макрокоманд**

В макросе макрокоманды выполняются в порядке их расположения. Однако для реализации алгоритма в программах часто необходимо нарушить последовательность их выполнения в зависимости от некоторых условий. Условия позволяют определить порядок передачи управления между макрокомандами в макросе и обеспечивают выполнение определенных ветвей алгоритма. Например, в макросе проверяется значение поля в форме на соответствие заданным условиям, и для одних значений может потребоваться вывести сообщение, а для других значений произвести вывод отчета.

Для изменения порядка выполнения макрокоманд может быть использован управляющий блок **Если условное выражение то [макрокоманды\_то] иначе [макрокоманды\_иначе] Конец блока "Если"** (If...Then...Else...End If). Блок обеспеч-

чиает выполнение одной или другой группы макрокоманд в зависимости от значения условного выражения.

**Условное выражение** является логическим выражением, которое возвращает значение **Истина** (True) или **Ложь** (False). Если **условное выражение** имеет значение Истина, то выполняются [макрокоманды\_то], в противном случае — [макрокоманды\_иначе].

Для включения в макрос блока **Если** (If) нужно выбрать его из раскрывающегося списка в поле **Добавить новую макрокоманду** (Add New Action) или перетащить в нужное место из каталога макрокоманд. Блок для выполнения набора макрокоманд, если условное выражение истинно, отобразится в макросе в виде, представленном на рис. 8.11. Если условное выражение ложно, то набор макрокоманд не выполнится и управление будет сразу передано на макрокоманду, следующую за блоком **Если** (If).

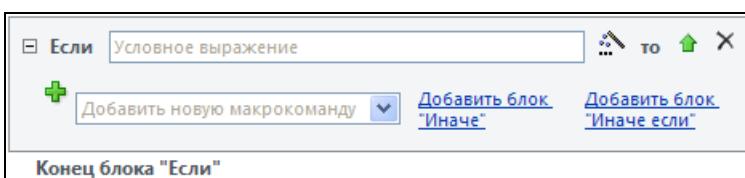


Рис. 8.11. Блок **Если ... то ... Конец блока "Если"**

После выполнения **Добавить блок "Иначе"** (Add Else) (см. рис. 8.11) получим блок **Если ... то ... иначе ... Конец блока "Если"**, который позволит выполнить набор макрокоманд **то**, если условное выражение истинно, и набор макрокоманд **иначе** в противном случае (рис. 8.12).

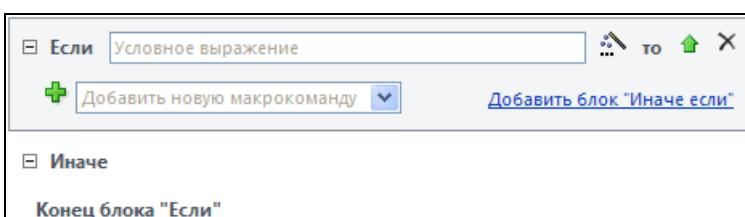


Рис. 8.12. Блок **Если ... то ... иначе ... Конец блока "Если"**

На рисунках видно, что можно **Добавить блок "Иначе если"** (Add Else If), и тогда в набор макрокоманд **иначе** будет включен новый блок **Если**, который позволит организовать проверку нового условного выражения и выполнение нового набора макрокоманд. Структура блока в этом случае может быть записана в следующем виде: **Если условное выражение то [макрокоманды\_то] иначе Если условное выражение то [макрокоманды\_то] Конец блока "Если"**. Таким образом, условие позволяет пропускать или выполнить связанный с первым условием набор макрокоманд и перейти к проверке следующего условия. Добавляя новые логиче-

ские блоки, можно реализовать самые сложные алгоритмы, связанные с условными переходами к различной логике.

Допускается ввод в поле **Условное выражение** (Conditional expression) значений **Нет** (No) (**Да** (Yes)) или **Ложь** (False) (**Истина** (True)). Этот прием может использоваться при отладке макросов, например для того, чтобы пропустить выполнение соответствующей группы макрокоманд.

**Задача.** Пусть необходимо по договору проверить текущее выполнение плановых поставок конкретного товара. В форме ДОГОВОР, включающей подчиненную форму с перечнем заказанных товаров, проверим для одного из заказанных товаров наличие товара на складе. В случае отсутствия товара выявим фактически произведенные отгрузки по заказанному товару, отобразив отчет об отгрузках этого товара по накладным, выписанным для просматриваемого договора.

В источник записей подчиненной формы ПОСТАВКА\_ПЛАН, построенный на таблицах ПОСТАВКА\_ПЛАН и ТОВАР, включите поле НАЛИЧИЕ\_ТОВ, не отображая его в форме. Подготовьте отчет об отгрузках товара на таблице ОТГРУЗКА, которую дополните датой и номером договора из таблицы НАКЛАДНАЯ, а для поля КОЛИЧЕСТВО\_ОТГРУЖЕННОГО предусмотрите итоговую величину.

1. Для реализации этой задачи создайте изолированный макрос, представленный на рис. 8.13. Свяжите его выполнение с событием **Получение фокуса** для поля КОД\_ТОВ (код товара) в ПОСТАВКА\_ПЛАН.
2. В макросе в первом блоке **Если ... то ... иначе ... Конец блока "Если"** записано условное выражение для проверки значения в поле НАЛИЧИЕ\_ТОВ, которое имеет тип данных **Логический** (Yes/No). В зависимости от результата проверки условия будут выполнены различные ветви программы.
  - а) если логическое выражение возвращает значение **Истина** (True), выполняются макрокоманды **то: ОкноСообщения** (MessageBox) и **ОстановитьМакрос** (StopMacro). Первая выводит сообщение о том, что товар, на который был переведен фокус, есть на складе. Последняя завершает выполнение макроса;
  - б) при возврате значения **Ложь** (False) выполняется набор макрокоманд **иначе: ОкноСообщения** (MessageBox) и **ОткрытьОтчет** (OpenReport). Первая макрокоманда выводит сообщение об отсутствии товара на складе, вторая открывает отчет об отгрузках по открытому договору того товара, на который был переведен фокус.
3. После этого выполняется второй блок **Если ... то ... Конец блока "Если"**, в котором проверяется ответ пользователя на выведенное в диалоговом окне предложение закрыть отчет (рис. 8.14). Вывод диалогового окна обеспечивается функцией **MsgBox('Закрыть отчет?';1)**. При утвердительном ответе отчет закрывается. На этом макрос завершает свою работу.
4. В приведенном примере переместите второй блок **Если** на одну позицию выше, щелкнув на соответствующей зеленой стрелке в правой части блока. Он переместится в набор макрокоманд **иначе**. При этом можно удалить из набора **то** макрокоманду **ОстановитьМакрос** (StopMacro).

```

Если [Формы]![ДОГОВОР]![ПОСТАВКА_ПЛАН подчиненная форма]![НАЛИЧИЕ_ТОВ]=Да то

    ОкноСообщения
        Сообщение Товар на складе есть
        Сигнал Да
        Тип Предупреждающее!
        Заголовок
        ОстановитьМакрос

Иначе

    ОкноСообщения
        Сообщение Товара на складе нет!
        Сигнал Да
        Тип Информационное
        Заголовок Товара нет!

    ОткрытьОтчет
        Имя отчета ОТГРУЗКА_ТОВАРА
        Режим Просмотр
        Имя фильтра
        Условие отбора = [НОМ_ДОП]=[Формы]![ДОГОВОР]![НОМ_ДОП] And
                           [КОД_ТОВ]=[Формы]![ДОГОВОР]![ПОСТАВКА_ПЛАН подчиненная форма]![КОД_ТОВ]
        Режим окна Обычное

    Конец блока "Если"

Если MsgBox('Закрыть отчет?';1)=1 то

    ЗакрытьОкно
        Тип объекта Отчет
        Имя объекта ОТГРУЗКА_ТОВАРА
        Сохранение Подсказка

    Конец блока "Если"

```

**Добавить новую макрокоманду**

Рис. 8.13. Пример макроса с блоками Если

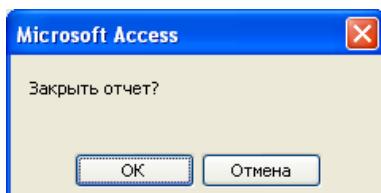


Рис. 8.14. Диалоговое окно функции MsgBox('Закрыть отчет?';1)

- Убедитесь, что и теперь макрос при перемещении фокуса на новый код товара выполняется правильно.

6. Чтобы сделать структуру программы более наглядной, щелкните на команде **Свернуть макрокоманды** (Collapse Actions). При этом макрос свернет макрокоманды, отобразив их имена и аргументы в строках, а блоки оставит не свернутыми.
7. Убедитесь в том, что после сохранения этого изолированного макроса его имя отобразится в списке макросов области навигации, а также в разделе **В этой базе данных** (In this Database) в окне каталога макрокоманд.

## Примеры условных выражений

Рассмотрим примеры условных выражений в макросах.

- ❖ Проверка значений полей в текущей записи формы:
  - ◆ значения в поле ЦЕНА  
ЦЕНА > = 2000;
  - ◆ нахождение значения в поле ДАТА\_ДОГ (дата заключения договора) в интервале не раньше 2-фев-07 и не позже 2-мар-07:  
[ДАТА\_ДОГ] Between #2-фев-07# And #2-мар-07#
  - ◆ значения в поле СУММА на равенство пустому (Null) значению:  
sNull([СУММА]) или [СУММА] Is Null.
- ❖ Проверка одновременного выполнения двух условий на равенство заданным значениям в полях НАИМ\_ТОВ и КОЛ\_ПОСТ текущей записи:  
[НАИМ\_ТОВ] = "Монитор" And [КОЛ\_ПОСТ] > 15

### ЗАМЕЧАНИЕ

В условиях могут использоваться функции, например DCount, DMax, MsgBox.

- ❖ Подсчет числа выписанных накладных и сравнение его с заданным:  
DCount (Формы! [НАКЛАДНАЯ] ! [НОМ\_НАКЛ]; 'НАКЛАДНАЯ') > 50  
или  
DCount ('\*' ; 'НАКЛАДНАЯ') > 50
- ❖ Выбор максимальной цены товара и сравнение ее с заданным значением:  
DMax (Формы! [ТОВАР] ! [ЦЕНА]; 'ТОВАР') > 500
- ❖ Функция MsgBox может использоваться для вывода на экран диалогового окна, содержащего нужное сообщение, и кнопок **OK** и **Отмена** (Cancel), нажатие которых формирует одно или другое значение функции. Сравнение значения, сформированного функцией MsgBox, с заданным значением позволяет выбрать пользователю вариант действий:

MsgBox ("Выдать справку?"; 1) = 1

Первый параметр функции задает выводимое на экран сообщение, единица на месте второго параметра определяет отображение в диалоговом окне кнопок **OK**

и **Отмена** (Cancel). Если пользователь нажимает кнопку **OK**, функция возвращает значение 1. Если нажимает кнопку **Отмена** (Cancel), функция возвращает значение 2.

### **ЗАМЕЧАНИЕ**

В условиях могут использоваться значения элементов управления из объектов, которые не являются текущими в данный момент.

- ❖ Сравнение значения поля КОД\_ПОК в текущей записи формы ПОКУПАТЕЛЬ со значением аналогичного поля в открытой форме ДОГОВОР:

```
Forms! [ДОГОВОР] ! [КОД_ПОК] = [КОД_ПОК]
```

## **Организация выполнения макросов**

При запуске макроса выполнение начинается с первой макрокоманды и следует по алгоритму, реализуемому макросом. В процессе выполнения проверяются условия и в зависимости от результата выполняются те или иные макрокоманды, вызываются другие макросы. При вызове другого макроса ему передается управление. Вызванный макрос может выполняться несколько раз. После выполнения вызванного макроса управление возвращается к вызывающему макросу и продолжается выполнение его макрокоманд.

При этом, следуя алгоритму, макрос выполняется по одному из заранее определенных путей из множества возможных. Таким образом, макрос сам выбирает этот путь в зависимости от выполнения условий.

Имеется возможность организовать выполнение макросов, используя механизм расширенной обработки событий. Access распознает определенные события, к которым может привязываться запуск макроса или процедуры обработки событий. Событиями, например, являются открытие отчета, изменение данных в поле формы или перевод фокуса на другую запись или поле в форме, щелчок мышью на элементе управления. Для формы определены события формы, раздела формы, записи и элемента управления. Для отчета определены только события отчета и раздела отчета.

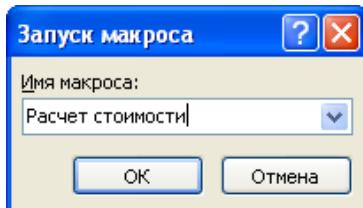
Порядок выполнения макросов зависит от порядка возникновения событий, и в значительной степени определяется действиями пользователя в формах. Таким образом, управление программой в основном осуществляется пользователем, который выполняет действия, а программа реагирует на них.

### **Запуск макроса**

Для запуска изолированного макроса, открытого в режиме конструктора, надо нажать кнопку **Выполнить** (Run) на вкладке ленты.

Для запуска изолированного макроса из области навигации надо установить курсор на имя макроса и дважды щелкнуть на нем или нажать кнопку **Выполнить** (Run) в его контекстном меню. Можно, не выбирая макроса в области навигации, на вкладке **Работа с базами данных** (Database Tools) в группе **Макрос** (Macro) щелкнуть на **Выполнить макрос** (Run Macro).

Откроется диалоговое окно **Запуск макроса** (Run Macro), в котором в поле со списком **Имя макроса** (Macro Name) можно выбрать нужный макрос (рис. 8.15).



**Рис. 8.15.** Окно выбора запускаемого макроса

В диалоговом окне **Запуск макроса** (Run Macro) в поле со списком **Имя макроса** (Macro Name) отображаются не только имена макросов, представленных в области навигации, но имена вложенных макросов. В поле **Имя макроса** (Macro Name) имя вложенного макроса записывается как

*ИмяГруппыМакросов. ИмяВложенногоМакроса.*

Пользователь имеет возможность создать макрос, автоматически запускающийся при каждом открытии базы данных. Этот макрос должен иметь имя **AutoExec**. В процессе открытия базы данных Access осуществляет поиск макроса с этим именем и, если такой макрос существует, автоматически запускает его.

В макрос **AutoExec** целесообразно поместить макрокоманды, которые подготовят нужную рабочую среду для пользователя, откроют формы и, если необходимо, другие объекты базы данных, разместят их на экране в удобном виде.

Этот макрос создается как любой другой макрос и сохраняется под именем **AutoExec**. При следующем открытии базы данных Access автоматически запустит его.

Если требуется открыть базу данных, не выполняя при этом макрос **AutoExec**, надо открывать базу данных при нажатой клавише <Shift>. Клавиша <Shift> позволяет отключать макрос **AutoExec**, если в диалоговом окне **Параметры Access** (Access Options) на вкладке **Текущая база данных** (Current Database) в разделе **Параметры приложений** (Application Options) установлен флажок **Специальные клавиши Access** (Use Access Special Keys).

## Выполнение макроса с наступлением события

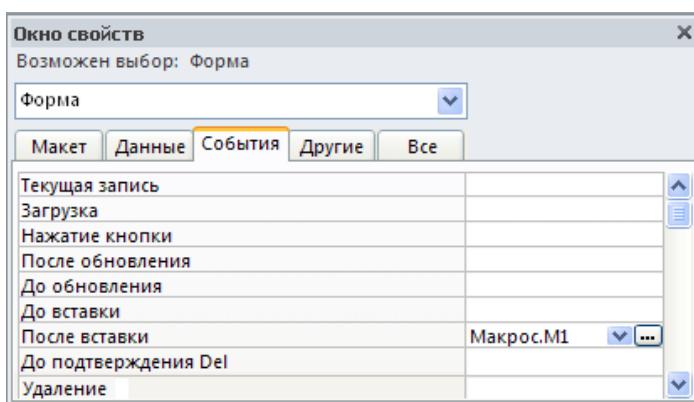
Наряду с рассмотренными способами запуска макросов в Access предусмотрена возможность автоматического запуска макроса при возникновении некоторого события в форме, отчете или их элементах управления. Для этого необходимо в свой-

ствах в строку соответствующего события ввести имя макроса или организовать создание внедренного макроса.

Связь события с программой позволяет организовать взаимодействие различных компонентов приложения. Причем поскольку события чаще всего возникают в результате действий пользователя, управление приложением осуществляется по его инициативе.

Пусть необходима обработка данных при добавлении записей через форму. Если эта обработка реализуется некоторым макросом, то для автоматического запуска этого макроса можно использовать события **До вставки** (Before Insert) и **После вставки** (After Insert), которые являются событиями формы.

Событие **До вставки** (Before Insert) наступает при вводе пользователем первого символа в новую запись, но до фактического добавления записи в базу данных. Событие **После вставки** (After Insert) наступает после добавления новой записи в базу данных. Пример вызова вложенного макроса, размещенного в области навигации, для обработки события **После вставки** (After Insert), приведен на рис. 8.16.



**Рис. 8.16.** Связь вложенного макроса с событием формы

Пусть необходима обработка данных при обновлении значения в поле записи через форму. Для автоматического запуска макроса, выполняющего такую обработку, можно использовать события **До обновления** (Before Update) и **После обновления** (After Update), которые определены как для формы, так и для ее элементов управления.

Событие **До обновления** (Before Update) наступает, когда пользователь приступает к обновлению данных в элементе управления или записи.

Событие **После обновления** (After Update) наступает после обновления данных в элементе управления или записи. Данное событие возникает при потере фокуса элементом управления, т. е. при переходе к другому элементу, или после сохранения записи. Сохранение записи выполняется при переходе к другой записи или выполнения пользователем команды **Записи | Сохранить** (Records | Save) на вкладке ленты **Главная** (Home). Событие возникает для новых и существующих записей.

Пусть в подчиненной форме ПОСТАВКА\_ПЛАН формы ДОГОВОРЫ С ПОКУПАТЕЛЯМИ необходимо при обновлении значений в полях КОЛ\_ПОСТ или ЦЕНА вызывать макрос, например, **Расчет стоимости**, в котором содержится макрокоманда **ЗадатьЗначение** (SetValue), выполняющая расчет значения поля СУММА\_ПОСТ = ЦЕНА\*КОЛ\_ПОСТ (см. рис. 8.5). Для вызова такого макроса в свойствах полей КОЛ\_ПОСТ и ЦЕНА на вкладке **События** (Event) в строке **После обновления** (After Update) выберите имя макроса (рис. 8.17).

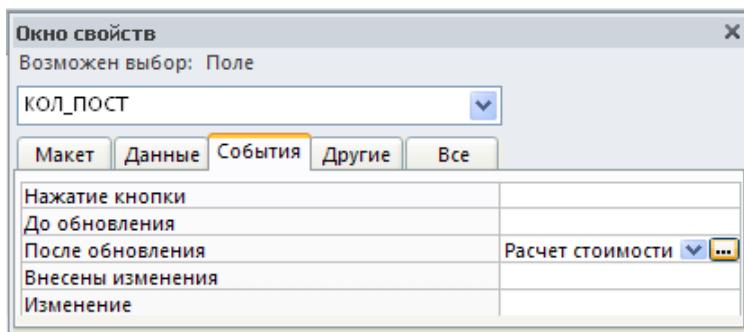


Рис. 8.17. Связь макроса с событием элемента управления

В результате выполнения макроса результат вычисления выражения сохранится в таблице ПОСТАВКА\_ПЛАН в поле СУММА\_ПОСТ.

В приведенном примере макрос хранится как отдельный объект, представленный в области навигации. При большом числе объектов становится трудно управлять макросами, связь которых с конкретными формами или отчетами не очевидна. Значительно удобнее использовать внедренные макросы, сохраняемые в формах или отчетах.

Для внедрения макроса в подчиненную форму ПОСТАВКА\_ПЛАН откройте ее в режиме конструктора или макета. Откройте окно свойств для поля КОЛ\_ПОСТ и в пустой строке события **После обновления** (After Update) нажмите кнопку построителя. В окне **Построитель** (Choose Builder) выберите **Макросы** (Macro Builder) (рис. 8.18).

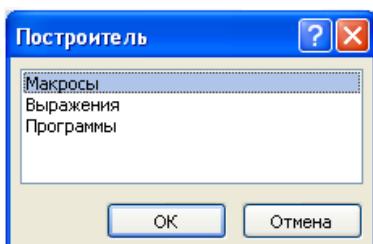
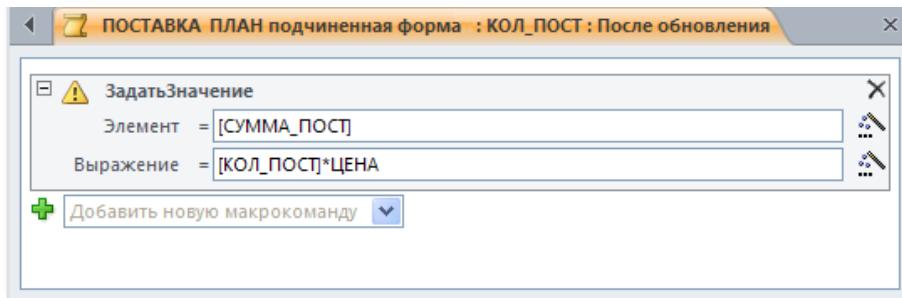


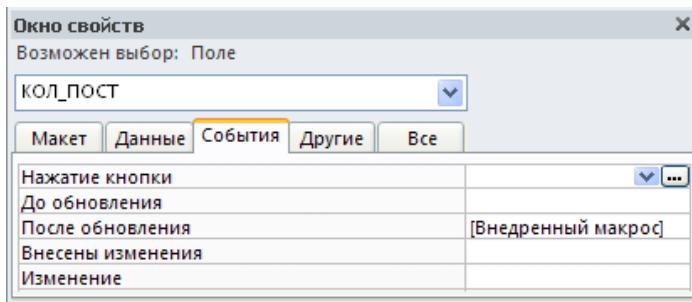
Рис. 8.18. Начало процесса по созданию внедренного макроса

Откроется окно конструктора макросов. В этом окне, как и в окне изолированного макроса, выберите макрокоманду для вычисления значения суммы (см. рис. 8.1).



**Рис. 8.19.** Внедренный макрос, выполняющийся при наступлении в поле КОЛ\_ПОСТ события **После обновления**

В окне свойств поля КОЛ\_ПОСТ в строке события **После обновления** (After Update) будет записано **[Внедренный макрос]** ([Embedded Macro]) (рис. 8.20).



**Рис. 8.20.** Связь внедренного макроса с событием

Для одной и той же формы или отчета могут одновременно использоваться и изолированные, и внедренные макросы.

## Порядок выполнения макросов, вызываемых событиями

Для правильного определения порядка выполнения макросов, которые запускаются при возникновении событий, необходимо понимание того, когда и в какой последовательности возникают события. Каждое отдельное действие, например, переход на другой элемент управления, может явиться причиной ряда событий, возникающих в определенной последовательности. Если созданы макросы, которые должны выполняться в определенном порядке, пользователь должен быть уверен, что события, вызывающие эти макросы, возникают в том же порядке.

Поясним последовательность возникновения событий при закрытии формы и при переходе с одного элемента управления на другой.

Когда пользователь закрывает форму, для текущего элемента управления возникают события **Выход** (Exit) и **Потеря фокуса** (Lost Focus), а также события

формы, такие как **Выгрузка** (Unload), **Отключение** (Deactivate) и **Закрытие** (Close). Порядок возникновения событий будет следующим:

**Выход** ⇒ **Потеря фокуса** ⇒ **Выгрузка** ⇒ **Отключение** ⇒ **Закрытие**

Когда пользователь переводит курсор (фокус) на другой элемент управления, для элемента, с которого переводится курсор, сначала возникает событие **Выход** (Exit), а затем **Потеря фокуса** (Lost Focus); для элемента, на который переводится курсор, сначала возникает событие **Вход** (Enter), а затем **Получение фокуса** (Got Focus).

## Вызов макроса из другого макроса

Если некоторый набор макрокоманд будет использоваться в нескольких макросах, то вместо того, чтобы включать эти команды в каждый макрос, пользователь имеет возможность создать макрос, содержащий эти макрокоманды, и вызывать его из других макросов.

Вызов макроса выполняется макрокомандой **ЗапускМакроса** (RunMacro). Макрокоманда вводится в той позиции макроса, из которой должен вызываться другой макрос.

Аргументы макрокоманды **ЗапускМакроса** (RunMacro):

- ❖ **Имя макроса** (Macro Name). Если запускается вложенный макрос, имя задается с указанием имени группы:  
*ИмяГруппыМакросов.ИмяВложенногоМакроса*
- ❖ **Число повторов** (Repeat Count) вызова макроса. Если этот аргумент и аргумент **Условие повтора** (Repeat Expression) не заданы, вызванный макрос выполняется один раз.
- ❖ **Условие повтора** (Repeat Expression) — логическое выражение. При значении **Ложь** (False) повторное выполнение вызванного макроса не производится и возобновляется выполнение вызывающего макроса. Если заданы значения обоих аргументов, **Число повторов** (Repeat Count) и **Условие повтора** (Repeat Expression), вызываемый макрос будет выполняться до тех пор, пока выражение, заданное аргументом **Условие повтора** (Repeat Expression), не получит значение **Ложь** (False), либо пока не исчерпается число указанных повторов.

Макрокоманда **ЗапускМакроса** (RunMacro) с помощью аргументов **Число повторов** (Repeat Count) и **Условие повтора** (Repeat Expression) позволяет реализовать циклы в программах на языке макросов.

При запуске исходного макроса его макрокоманды выполняются поочередно до тех пор, пока не наступит очередь макрокоманды **ЗапускМакроса** (RunMacro). В этот момент происходит вызов второго макроса. После завершения выполнения макрокоманд вызванного макроса выполнение вызывающего макроса возобновляется со следующей макрокоманды.

Допускается выполнение макрокоманды **ЗапускМакроса** (RunMacro) из вызванного макроса. Это означает, что сначала вызывается макрос А, из которого в

свою очередь вызывается макрос В и т. д. В каждом случае по завершении вызванного макроса выполняются следующие макрокоманды вызывающего макроса.

## Создание кнопки запуска макроса в форме

Можно предусмотреть запуск макроса пользователем из формы путем нажатия кнопки. Для связи кнопки с макросом в свойствах кнопки на вкладке **События** (Event) в строке **Нажатие кнопки** (On Click) определяется связь с макросом.

## Создание кнопки запуска макроса с помощью мыши

Для создания кнопки запуска макроса с помощью мыши надо в области навигации выбрать макрос, который необходимо запустить при нажатии кнопки, и перетащить его в форму, открытую в режиме конструктора. В форму будет помещена кнопка, по которой будет выполняться данный макрос. Кнопка будет иметь подпись, соответствующую имени макроса.

Если макрос содержит вложенные макросы, при использовании этого метода организуется запуск по кнопке только первого вложенного макроса. Для запуска другого вложенного макроса необходимо в окне свойств кнопки на вкладке **События** (Event) в строке **Нажатие кнопки** (On Click) выбрать вложенный макрос (рис. 8.21).

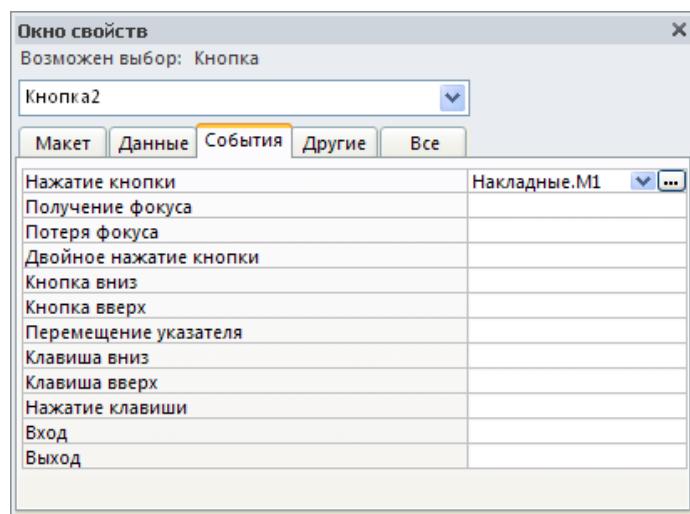


Рис. 8.21. Окно свойств кнопки, вызывающей выполнение макроса

## Создание кнопки запуска макроса мастером

Кнопка для запуска макроса может быть создана в форме с помощью мастера. Для этого надо нажать на вкладке ленты конструктора форм в группе **Элементы управления** (Controls) кнопку **Кнопка** (Command Button). После вычерчивания

кнопки в форме открывается диалоговое окно мастера **Создание кнопок** (Command Button Wizard), в котором надо выбрать категорию **Разное** (Miscellaneous) и действие **Выполнить макрос** (Run Macro). Далее следует выбрать макрос, который будет запускаться при нажатии кнопки. Мастер позволяет выбрать рисунок или задать текст подписи на кнопке, ввести имя кнопки запуска макроса.

Мастер создает для кнопки внедренный макрос, что отражается в свойстве **Нажатие кнопки** (On Click) записью **[Внедренный макрос]** ([Embedded Macro]). Созданный мастером внедренный макрос содержит одну макрокоманду запуска изолированного макроса, выбранного в ходе работы мастера.

Очевидно, что для кнопки также как для любого другого элемента формы, можно создать любой макрос.

## Макросы данных

Макросы данных являются новшеством в данной версии. Они предназначены для добавления логики к данным в исходных таблицах. Макросы данных связываются с событиями в базовой таблице и выполняются только при изменении, вставке или удалении записи. Такие макросы позволяют проверять данные или выполнять вычисления. Привязывая логику к данным и сосредотачивая ее в исходных таблицах, макросы данных обеспечивают доступ к ней из объектов приложения, а при использовании базы в сети и из различных приложений. Так, выполнение операций с записями в формах, построенных на базовых таблицах, инициирует выполнение макросов данных.

Макросы данных, по сути, позволяют реализовать в базе данных Access 2010 функции, которые в серверных базах данных реализуются с помощью особого вида хранимых процедур — триггеров. Триггер также связывается с определенной операцией в базовой таблице и каждый раз при выполнении этой операции вызывается.

С помощью макросов данных, как и триггеров, могут выполниться действия по обеспечению реляционной целостности данных, не поддерживаемые стандартными средствами. Макрос данных также не имеет ни параметров, ни возвращаемого значения.

При программировании в приложении получить доступ к данным базы можно только через формы и отчеты. В макросах данных, в отличие от макросов, привязанных к событиям в объектах приложения, для доступа к данным используются ссылки на поля таблиц. Обращение к данным через поля объектов приложения не допустимо.

Следует подчеркнуть, что реализация в макросах данных циклов макрокомандой **ДляКаждойЗаписи** (ForEachRecord) обеспечивает простоту решения многих задач бизнес-логики.

Если ваш компьютер настроен для отправки и приема сообщений электронной почты, в макросе данных можно использовать макрокоманду **ОтправитьПочту**

(SendEmail). В противном случае системой будет выдано сообщение о том, что не удалось отправить сообщение электронной почты.

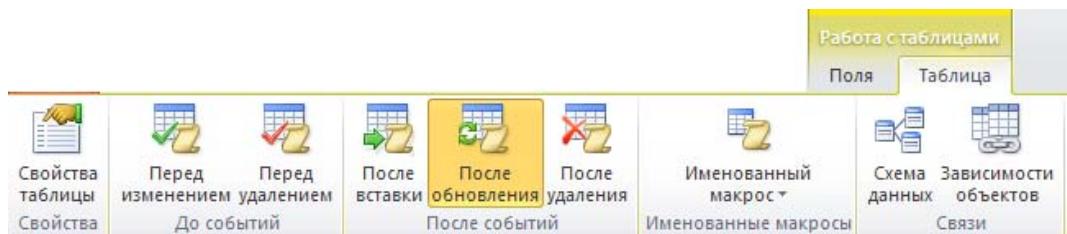


Рис. 8.22. Команды работы с макросами данных на вкладке Таблица

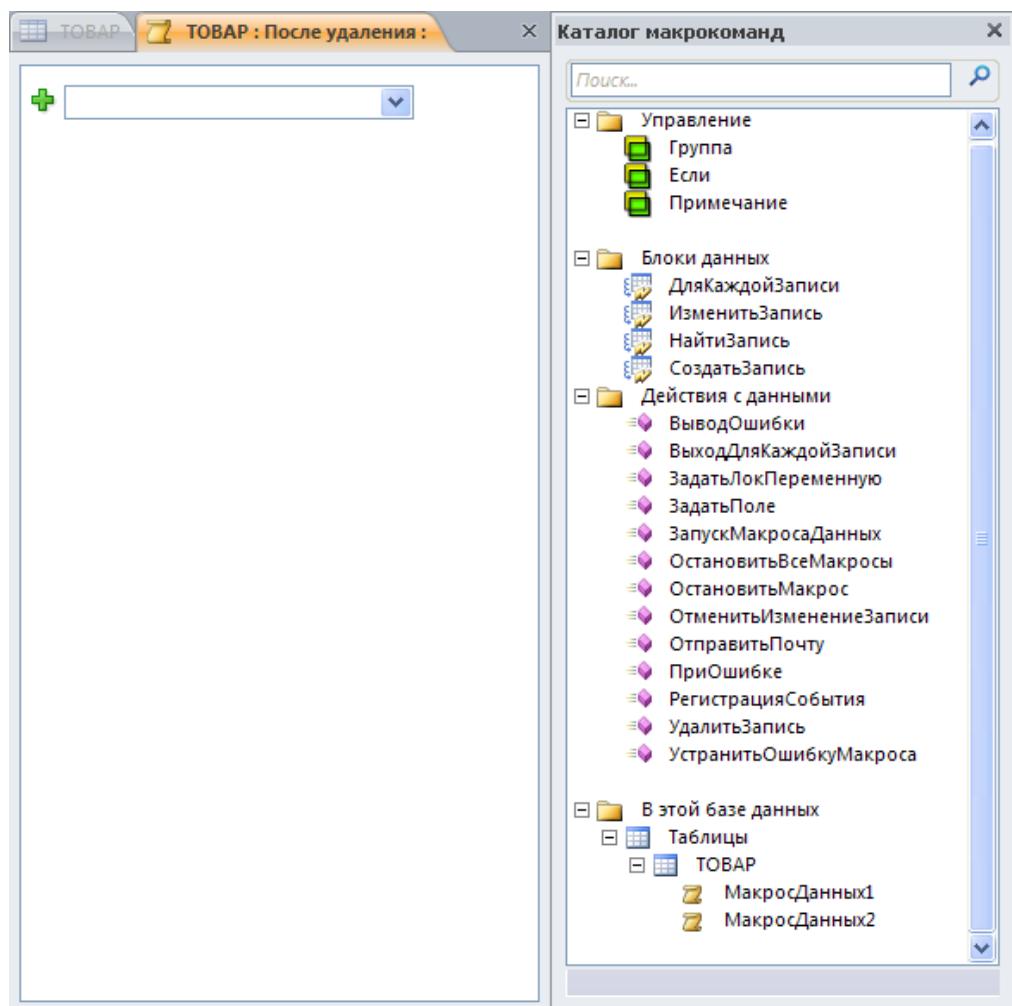


Рис. 8.23. Окно макроса данных с каталогом макрокоманд

Чтобы начать создание макроса данных, необходимо открыть таблицу в режиме таблицы. При этом на вкладке ленты **Работа с таблицами | Таблица** (Table Tools | Table) отображаются кнопки создания макросов данных, которые будут выполняться **Перед изменением** (Before Change), **После обновления** (After Update) и т. д. (рис. 8.22).

Для каждой таблицы может быть создано по одному макросу данных на каждое из представленных на ленте действий. Щелчком мыши на кнопке открывается окно создания макроса данных. На рис. 8.23 показано окно создания макроса данных таблицы ТОВАР, который будет выполняться **После удаления** (After Delete).

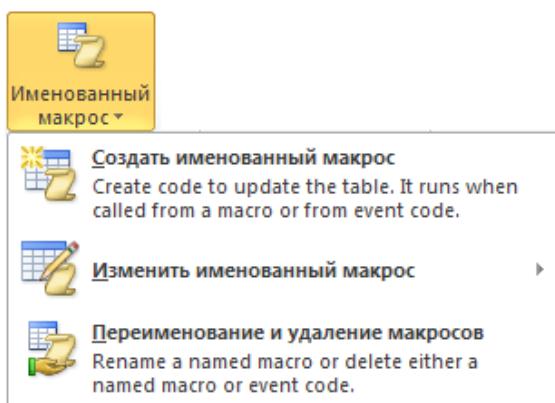
Список макрокоманд для макроса данных отличается от предлагаемого для макросов базы данных. В разделе каталога макрокоманд **В этой базе данных** (In this Database) отображаются макросы данных, созданные для таблиц.

Макросы данных внедряются в таблицу. Если для некоторого действия в таблице создан макрос данных, то соответствующая кнопка на вкладке ленты отображается в оранжевом цвете.

## Именованные макросы

Кроме макросов данных, связанных с отображенными на вкладке действиями, могут быть созданы именованные макросы данных. Эти макросы данных могут выполняться только при вызове из других макросов данных. В именованном макросе могут определяться параметры. Значения параметрам присваиваются в вызывающем макросе.

Щелчком мыши на списке кнопки **Именованный макрос** (Named Macro) открывается окно, показанное на рис. 8.24.



**Рис. 8.24.** Список команд именованного макроса

Первая команда списка позволяет **Создать именованный макрос** (Create Named Macro), вторая — **Изменить именованный макрос** (Edit Named Macro), третья — **Переименование и удаление макросов** (Rename/Delete Macro) — просмотреть все таблицы, содержащие макросы.

Первый именованный макрос по умолчанию имеет имя МакросДанных1 (DataMacro1). При обращении к нему дополняется именем таблицы, в которой был создан, и записывается, например так: ТОВАР.МакросДанных1.

Запуск именованного макроса выполняется макрокомандой **ЗапускМакроса-Данных** (RunDataMacro), доступной как из макроса данных, так и из любых других макросов приложения. Например, работая в форме, можно не выполнять вычисления через обращения к полям формы, а вызвать именованный макрос данных, который выполнит вычисления непосредственно в таблицах. Изменения в таблицах тут же отобразятся и в форме.

Командой **Переименование и удаление макросов** (Rename/Delete Macro) открывается окно диспетчера макросов данных, представленное на рис. 8.25. В нем перечислены все таблицы базы, содержащие макросы данных. Именованные макросы также отображаются в окне диспетчера макросов данных.

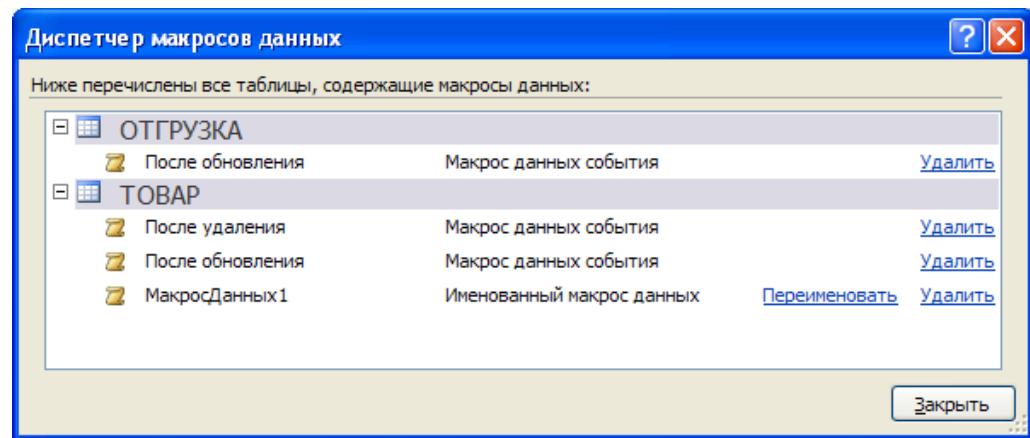


Рис. 8.25. Список таблиц открытой базы, содержащих макросы данных

## Использование макросов для решения задач

При разработке практических приложений часто необходимо при изменениях в одних документах решать задачи, позволяющие актуализировать текущие данные в других. Рассмотрим примеры таких задач.

**Задача.** Рассмотрим создание макроса данных для таблицы ТОВАР, в котором после обновления записи проверяется изменение цены товара и перерасчет стоимости отгруженного товара в таблице ОТГРУЗКА.

1. Откройте таблицу ТОВАР и на вкладке ленты **Работа с таблицами | Таблица** (Table Tools | Table) щелкните по кнопке **После обновления** (AfterUpdate). В открывшееся окно макроса данных введите макрокоманды, выбирая их в каталоге и заполняя аргументы, как показано на рис. 8.26.

```

    ТОВАР : После обновления :

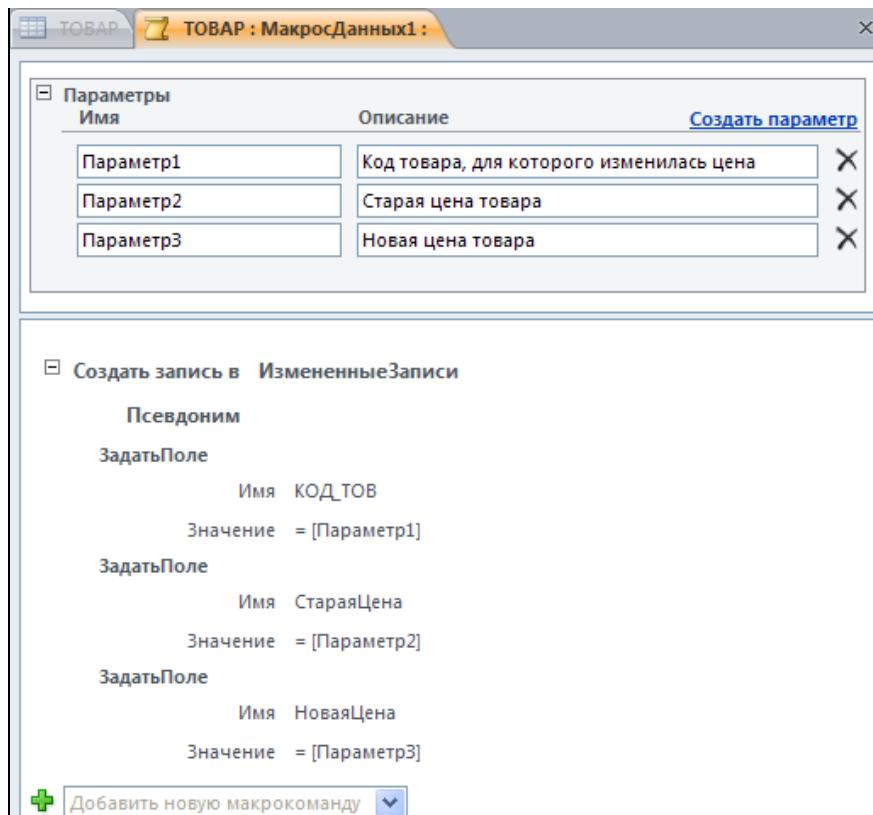
    ◻ Если Updated("ЦЕНА")=Истина то
        ◻ Для каждой записи в ОТГРУЗКА
            Условие отбора = [КОД_ТОВ]=[ТОВАР].[КОД_ТОВ]
            Псевдоним ИзменениеСтоимости
        ◻ ИзменитьЗапись
            Псевдоним ИзменениеСтоимости
            ЗадатьПоле
                Имя СУММА_ОТГР
                Значение = [ТОВАР].[ЦЕНА]*[КОЛ_ОТГР]
            Конец изменения записи
    Конец блока "Если"

```

**Рис. 8.26.** Макрос данных для обновления поля в записях таблицы ОТГРУЗКА

2. В блоке **Если** (If) в логическом выражении условия с помощью функции **Updated ("имя поля")** проверяется изменялось ли значение в поле ЦЕНА. При истинном значении выражения макрокомандой **Для каждой записи в** (ForEachRecord) организуется цикл, в котором для каждой записи таблицы ОТГРУЗКА, содержащие значение кода товара из обновленной записи таблицы ТОВАР, макрокомандой **ИзменитьЗапись** (EditRecord) рассчитается новое значение в поле СУММА\_ОТГР. Обратите внимание: макрокоманда **ИзменитьЗапись** (EditRecord) должна быть вложена в макрокоманду **Для каждой записи в** (ForEachRecord), иначе не будет найден псевдоним текущей записи цикла. Макрокоманда **ЗадатьПоле** (SetField) в свою очередь должна быть вложена в **ИзменитьЗапись** ((EditRecord)).
3. Далее создайте именованный макрос, в котором сведения об обновлениях цены в таблице ТОВАР будут сохраняться в некоторой таблице, например **ИзмененныеЗаписи**.
4. Создайте таблицу ИзмененныеЗаписи, содержащую поля код товара, старая цена, новая цена и ключ с типом данных **Счетчик** (AutoNumber). Использование такого ключа позволит включать в таблицу неоднократные изменения записи с одним и тем же кодом товара.
5. При открытой таблице ТОВАР создайте именованный макрос, выбрав соответствующую команду на вкладке **Работа с таблицами | Таблица** (Table Tools | Table). В макросе для получения из вызывающего макроса значений кода, старой и новой цены товара создайте соответствующие параметры (рис. 8.27). Далее выберите макрокоманду **Создать запись** (CreateRecord) с аргументом Из-

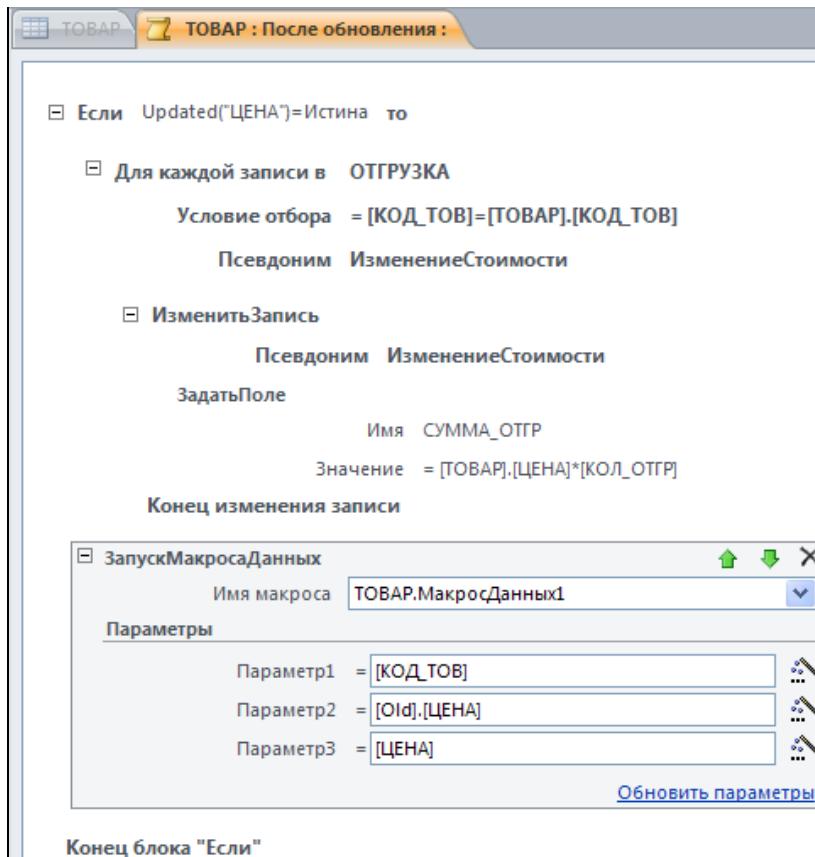
мененные Записи, отображаемую в макросе как **Создать запись в** (Create a Record In), и три макрокоманды **Задать Поле** (SetField) для присвоения значений полям таблицы **Измененные Записи**.



**Рис. 8.27.** Именованный макрос данных с параметрами

6. Закройте макрос. По умолчанию макрос получит имя ТОVAR.МакросДанных1.
7. Организуйте вызов именованного макроса при изменении цены в таблице ТОVAR. Для этого при открытой таблице ТОVAR щелкните **ПослеОбновления** (AfterUpdate) и дополните ранее созданный для нее макрос данных макрокомандой **ЗапускМакросаДанных** (RunDataMacro) (рис. 8.28).
8. Для вывода в макросе строки параметров щелкните **Обновить параметры** (Update Parameters). Присвойте параметрам значения из обновленной записи таблицы ТОVAR, а для получения значения старой цены используйте функцию `[Old].[ЦЕНА]`. Закройте макрос, сохранив внесенные изменения.
9. Измените в таблице цену одного из товаров и убедитесь, что в таблице ОТГРУЗКА во всех записях с кодом товара обновленная сумма отгрузки пересчиталась, а в таблицу **Измененные Записи** внесена новая запись. Обновления в таблицы вносятся вне зависимости от того открыты они или

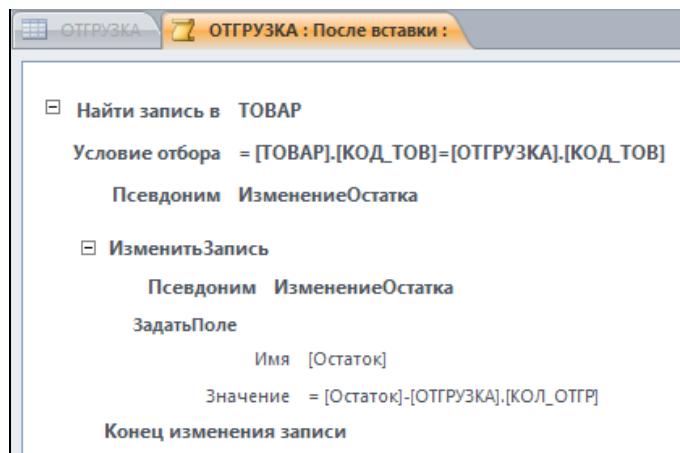
закрыты. Если таблица открыта, может понадобиться обновить ее отображение, например, закрыв и повторно открыв.



**Рис. 8.28.** Вызов именованного макроса данных и передача ему параметров

**Задача.** Пусть необходимо при отгрузке товара по накладной уменьшать его остаток на складе. Предположим, в нашей базе данных склад и товар связаны отношением 1 : M. Если товар хранится только на одном складе, остаток товара можно разместить в поле таблицы ТОВАР. Дополните таблицу ТОВАР полем Остаток.

1. Откройте таблицу ОТГРУЗКА в режиме таблицы и щелчком мыши на кнопке **ПоследВставки** (AfterInsert) создайте макрос данных, который будет выполнятьсь при добавлении новых записей в нее. В макросе используйте макрокоманду **Найти запись** (LookupRecord) с аргументом ТОВАР, отображаемую в макросе как **Найти запись в** (Look Up A Record In), и для найденной записи макрокоманду **ИзменитьЗапись** (EditRecord). В рамках последней с помощью макрокоманды **ЗадатьПоле** (SetField) присвойте полю Остаток новое значение, уменьшенное на количество отгруженного товара (рис. 8.29).



**Рис. 8.29.** Макрос данных, выполняющийся после вставки записи в таблицу ОТГРУЗКА

- Убедитесь, что при добавлении новой записи в таблицу ОТГРУЗКА через форму также будет выполняться макрос данных таблицы. Откройте форму Накладная, построенную на таблице НАКЛАДНАЯ и ОТГРУЗКА. В последней хранятся данные из спецификации накладной. Дополните одну из накладных строкой спецификации, указав какой товар и в каком количестве отгружается по накладной. Откройте таблицу ТОВАР и в поле Остаток увидите обновленное значение.

**Задача.** Пусть необходимо при добавлении новой записи об отгрузке товара рассчитать, какое количество товара всего отгружено по договору. Такой расчет позволит поддерживать в базе данные о выполнении плана отгрузки в актуальном состоянии. Ввод данных по отгрузке может осуществляться непосредственно в таблицу ОТГРУЗКА или через форму.

Количество товара, которое должно быть отгружено покупателю в соответствии с договором в конкретном месяце, хранится в записях таблицы ПОСТАВКА\_ПЛАН. Для сохранения общего количества товара, отгруженного по договору, можно предусмотреть в этой таблице дополнительное поле ВыполнениеПлана.

Макрос для решения этой задачи, так же как и при расчете остатка товара на складе, должен выполняться при наступлении действия **ПослеВставки** (AfterInsert). Поиск в таблице ПОСТАВКА\_ПЛАН, требующий обновления записи, может быть выполнен как в макросе **ПослеВставки** (AfterInsert), так и вынесен в именованный макрос. В последнем случае вызов именованного макроса и подготовка необходимых для него параметров выполняются в макросе **ПослеВставки** (AfterInsert).

На рис. 8.30 представлен именованный макрос таблицы ОТГРУЗКА Макрос-Данных1, в котором определены параметры, необходимые для поиска записи и обновления значения в поле ВыполнениеПлана таблицы ПОСТАВКА\_ПЛАН. Напомним, что поля НОМЕР ДОГОВОРА, МЕСЯЦ ОТГРУЗКИ и КОД ТОВАРА

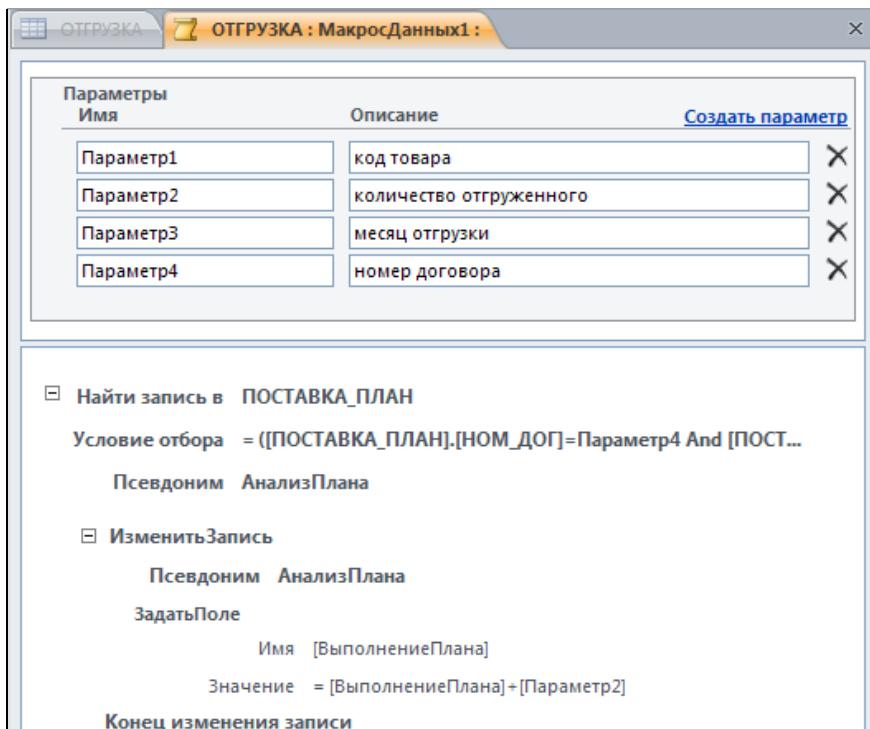
составляют ключ этой таблицы. Макрокоманда **Найти запись** (LookupRecord) в таблице ПОСТАВКА\_ПЛАН возвращает запись, соответствующую заданному условию отбора:

```
([ПОСТАВКА_ПЛАН].[НОМ_ДОГ]=Параметр4 And  
[ПОСТАВКА_ПЛАН].[СРОК_ПОСТ]=[Параметр3] And  
[ПОСТАВКА_ПЛАН].[КОД_ТОВ]=[Параметр1])
```

Макрокоманда **ИзменитьЗапись** (EditRecord), входящая в блок макрокоманды **Найти запись в** (Look Up A Record In), ссылаясь на псевдоним найденной записи, выполняет макрокоманду **ЗадатьПоле** (SetField), которая прибавляет указанное в записи об отгрузке количество к имеющемуся в поле ВыполнениеПлана.

### ЗАМЕЧАНИЕ

Чтобы выражение могло быть вычислено при первой отгрузке товара, полю ВыполнениеПлана по умолчанию должно присваиваться нулевое значение.

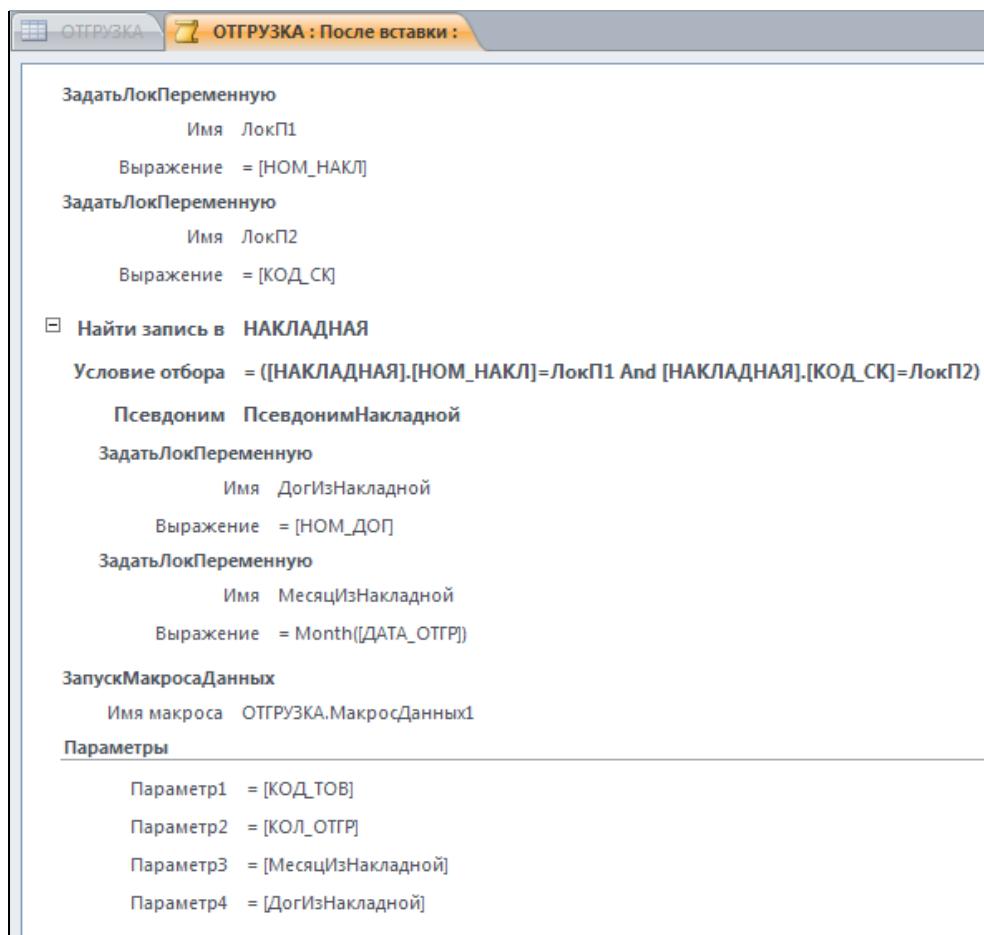


**Рис. 8.30.** Именованный макрос поиска и обновления записи в таблице ПОСТАВКА\_ПЛАН

При открытой таблице ОТГРУЗКА в режиме таблицы щелчком мыши на кнопке **ПослеВставки** (After Insert) откройте существующий макрос данных (см. рис. 8.29). Дополните макрос макрокомандами, которые позволят получить значения необходимых в именованном макросе параметров, и макрокомандой вызова именованного макроса (рис. 8.31). Код товара и количество отгруженного выбираются из встав-

ленной новой записи таблицы ОТГРУЗКА, а два недостающих НОМЕР ДОГОВОРА и МЕСЯЦ ОТГРУЗКИ надо выбрать из таблицы НАКЛАДНАЯ.

Чтобы найти запись в накладной, нужно задать значения его ключевых полей НОМЕР НАКЛАДНОЙ и КОД СКЛАДА. Значения этих полей имеются в добавленной в таблицу ОТГРУЗКА записи. Однако использование в условии отбора логического выражения с оператором `And` приводит к необходимости определения локальных переменных. Сформируйте логическое выражение, как показано на рисунке, и заключите его в круглые скобки. Аргумент **ПсевдонимНакладной** макрокоманды **Найти запись** ((LookupRecord) накладной можно опустить, т. к. он в рассматриваемой программе не используется.



**Рис. 8.31.** Вызов именованного макроса с параметрами

Чтобы запомнить значения полей из найденной записи, определите в рамках макрокоманды **Найти запись** (LookupRecord) НАКЛАДНОЙ локальные переменные. Для выделения из даты отгрузки номера месяца, необходимого при поиске

записи в таблице ПОСТАВКА\_ПЛАН, используйте в выражении, присваиваемом переменной, функцию Month.

После того как значения всех параметров подготовлены и доступны, вызовите именованный макрос ОТГРУЗКА.МакросДанных1. Для вывода строк с параметрами и присвоения им значений щелкайте по надписи **Обновить параметры** (Update Parameters).

### ЗАМЕЧАНИЕ

Если в логическом выражении условия отбора использованы логические операторы And, Or или другие, то вместо имен полей нужно использовать локальные переменные и заключать такое выражение в круглые скобки.

## Регистрация событий

Возникающие при выполнении макросов ошибки заносятся системой в журнал регистрации — системную таблицу с именем **USysApplicationLog**. При этом в строке состояния отображается надпись *Новые ошибки приложения*. Щелкните на ней, чтобы просмотреть эти ошибки. Обнаруженные системой ошибки отмечены в столбце **Category** значением **Execution**.

Используя макрокоманду **РегистрацияСобытия** (LogEvent), пользователь может включать в журнал свои замечания. Они отмечаются в столбце категории значением **User**.

Если в параметрах навигации установить значок *Показать системные объекты*, то можно увидеть журнал регистрации в категории таблиц наряду с пользовательскими. Щелкните правой кнопкой мыши в нижней части области навигации в ее свободном пространстве и в списке найдите команду **Параметры навигации** (Navigation Options).

При отсутствии ошибок, обнаруживаемых системой, просмотреть пользовательские сообщения в журнале регистрации можно, открыв соответствующую таблицу в области навигации.

### Задание 8.1

Измените рассмотренный выше макрос данных **ПослеВставки** (AfterUpdate) для таблицы ОТГРУЗКА так, чтобы в поле ВыполнениеПлана отображалось недопоставленное количество товара.

### Задание 8.2

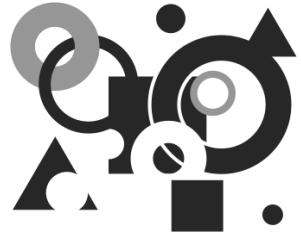
Создайте макрос данных, который будет выполняться при изменении цены в таблице ТОВАР и изменять стоимость в таблице ПОСТАВКА\_ПЛАН для договоров, заключенных после заданной даты, для которых еще не было отгрузок.

## Контрольные вопросы

1. Чем определяется последовательность выполнения макрокоманд?
2. Какая макрокоманда позволяет изменить текущую запись в объекте?
3. Должен ли объект, в котором меняется текущая запись, быть текущим?
4. Сдается ли объект текущим после выполнения макрокоманды, изменяющей текущую запись в нем?
5. Какая макрокоманда позволяет изменить значение поля записи?
6. Какой объект будет текущим в макросе, если он не установлен явно?
7. Можно ли в логическом выражении условия использовать функцию?
8. Как записывается ссылка на элемент управления в форме?
9. В ссылке на свойство каким знаком отделяется имя свойства от имени элемента управления?
10. Как организовать последовательное выполнение нескольких запросов действия?
11. Какие аргументы имеет макрокоманда **ОткрытьЗапрос** (OpenQuery)?
12. Какое значение должен иметь аргумент **Режим** (View), чтобы макрокоманда **ОткрытьЗапрос** (OpenQuery) приводила к выполнению запроса?
13. Какая макрокоманда позволяет из макроса вывести на экран сообщение?
14. Какая команда позволяет начать выполнение макроса из области навигации?
15. Как просмотреть все внедренные в формы макросы?
16. Как просмотреть, какие именованные макросы имеются в базе данных?
17. Как вставить содержимое одного макроса в другой?
18. Как организовать выполнение макроса при открытии формы?
19. Какая макрокоманда позволяет создать в макросе группу макросов, каждый из которых доступен для выполнения?
20. Как записывается ссылка на макрос, который включен в группу?
21. Какая макрокоманда позволяет выполнить фильтрацию записей?
22. К какому объекту применяется команда фильтрации записей?
23. Какая макрокоманда позволяет снять действие фильтра?
24. Почему в макрокоманде **ЗадатьЗначение** (SetValue) в выражении [Формы] ! [Отгрузки покупателям] ! [СКОЛ\_ОТГР] + [КОЛ\_ОТГР] при записи ссылки на поле СКОЛ\_ОТГР использован полный синтаксис, а для ссылки на поле КОЛ\_ОТГР в форме ОТГРУЗКА\_ПФ нет?
25. Может ли одно действие пользователя приводить к возникновению нескольких событий?
26. Отображается ли внедренный макрос среди объектов области навигации?
27. С какими объектами работает макрос данных?
28. При каких действиях в таблице выполняются макросы данных?
29. Можно ли для таблицы создать несколько макросов данных **ПослеВставки** (AfterUpdate)?
30. Можно ли именованный макрос данных вызвать из внедренного макроса формы?
31. Какая макрокоманда позволяет организовать цикл в макросе данных?

## Ответы

1. Порядком размещения макрокоманд в программе и условиями, заданными в блоке **Если** (If).
2. **НаЗапись** (GoToRecord).
3. Нет.
4. Нет.
5. **ЗадатьЗначение** (SetValue).
6. Объект, из которого вызывается макрос.
7. Да.
8. `Forms! [Имя_формы] ! [Имя_элемента_управления].`
9. Точкой.
10. Записать в макросе соответствующее число макрокоманд **ОткрытьЗапрос** (OpenQuery).
11. **Имя запроса** (Query Name), **Режим** (View) и **Режим данных** (Data Mode).
12. **Таблица** (Datasheet).
13. **ОкноСообщения** (MsgBox).
14. **Выполнить** (Run) в контекстном меню или на вкладке **Работа с базами данных** (Database Tools).
15. В каталоге макрокоманд в блоке **В этой базе данных/Формы** (In this Database/Forms).
16. В каталоге макрокоманд в блоке **В этой базе данных/Таблицы** ((In this Database/Tables).
17. Открыть принимающий макрос и, выбрав вставляемый в каталоге макрокоманд в блоке **В этой базе данных** (In this Database), выполнить двойной щелчок.
18. Записать в строку свойства события формы **Открытие** (On Open) имя макроса или создать внедренный макрос.
19. **Вложенный макрос** (Submacro).
20. `ИмяГруппыМакросов.имяВложенногоМакроса.`
21. **ПрименитьФильтр** (ApplyFilter).
22. К текущему объекту.
23. **ПоказатьВсеЗаписи** (ShowAllRecords).
24. Форма ОТГРУЗКА\_ПФ является текущей.
25. Да.
26. Нет.
27. С таблицами.
28. При указанных на вкладке **Работа с таблицами | Таблица** (Table Tools | Table).
29. Нет.
30. Да.
31. **ДляКаждойЗаписи** (ForEachRecord).



## ГЛАВА 9

# Разработка интерфейса приложения

## Управление приложением на основе форм

Для управления приложением предметной области могут разрабатываться интерфейсы, основанные на формах. При этом формы по своей структуре, как правило, сложнее, чем формы для представления отдельного электронного документа. Такой интерфейс может включать в качестве основного компонента какой-либо электронный документ предметной области, при работе с которым одновременно решается ряд смежных задач. При этом могут открываться другие документы, заполняться новые, выполняться процедуры проверки допустимости вводимых значений и обновления расчетных величин, хранимых в базе. Примерами таких задач могут быть проверка при выписке накладной для отгрузки товара степени выполнения договорных обязательств, проверка допустимости отгрузки с учетом наличия товара на складе и многое другое.

Разнообразные средства разработки форм позволяют создавать удобный интерфейс управления приложением, предоставляющий пользователю иметь под рукой все необходимое для выполнения его функций. Такие формы должны включать элементы управления, позволяющие открывать разные документы, производить необходимые запросы, анализ текущего состояния тех или иных объектов предметной области. Таким образом, обеспечивая основу для принятия тех или иных управленических решений. Например, при решении вопроса о возможности отгрузки товара проверять запасы на складе и оценивать допустимость отгрузки с точки зрения требований к нормативным запасам. При этом одновременно можно осуществить анализ платежей по выставленным счетам. Таким образом могут быть сожмечены в пределах одного интерфейса проверки, неываемые автоматически структурой базы данных. К последним, например, относится отслеживание возможности формирования накладной на отгрузку по существующим в базе данных договорам. Обычно такие интерфейсы предлагаются в рамках проблемно ориентированных прикладных систем.

Рассмотрим некоторые инструменты, которые могут быть использованы при разработке интерфейса управления приложением с помощью форм.

## Диалоговое окно входа в приложение

Создадим пользовательское диалоговое окно, которое будет отображаться при открытии приложения. В окне обеспечим ввод имени исполнителя, приступающего к работе, вход в основную форму управления приложением и сохранение кода исполнителя с помощью встроенного макроса. Сохраненный код позволит при открытии других объектов выбирать из них сведения, касающиеся только этого исполнителя. Вид такого диалогового окна приведен на рис. 9.1.

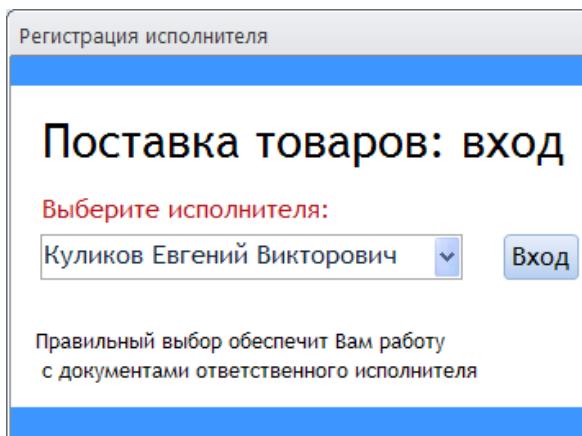


Рис. 9.1. Пользовательское диалоговое окно входа в приложение

Поскольку диалоговое окно, которое нужно разработать, требует ввода исполнителя работ, будем исходить из того, что в рассматриваемой предметной области для выполнения всех работ по ведению договоров решено назначать ответственного исполнителя. Это решение потребует дополнения базы данных таблицей ИСПОЛНИТЕЛЬ и внесения нового поля в существующую таблицу ДОГОВОР для связи ее с исполнителем. Реляционные базы данных позволяют выполнить эти изменения в ее структуре в любой момент без какого-либо ущерба для ранее разработанных объектов приложения. Для автоматической идентификации исполнителя определим ключевое поле Код исполнителя с типом данных Счетчик (AutoNumber) (рис. 9.2). Таблицу ДОГОВОР дополним полем связи Код исполнителя с типом данных Числовой (Number) и размером поля Длинное целое (Long Integer).

1. Для получения пользовательского диалогового окна нужно создать модальную всплывающую форму. Сначала создайте пустую форму, воспользовавшись соответствующей командой на ленте **Создание** (Create) в группе **Формы** (Forms). Пустая форма не имеет источника записей и одновременно с ее открытием в режиме макета открывается список полей, отображающих все таблицы базы данных. Они доступны для использования в этой форме и позволяют создать ее источник записей. В нашем случае список полей можно закрыть.

The screenshot shows the Microsoft Access 'ИСПОЛНИТЕЛЬ' (Executor) table structure and its properties. The main grid displays 14 fields: Код исполнителя (Counter), Фамилия (Text), Имя (Text), Отчество (Text), Должность (Text), Адрес электронной почты (Text), Рабочий телефон (Text), Домашний телефон (Text), Мобильный телефон (Text), Адрес (Text), Веб-страница (Hyperlink), Характеристика (Memo), and Вложения (Attachment). The 'Код исполнителя' field is highlighted. Below the grid, the 'Свойства поля' (Field Properties) dialog is open, showing the 'Общие' (General) tab selected. It lists properties like Размер поля (Field Size) set to Длинное целое (Long Integer), and Индексированное поле (Indexed) set to Да (Yes). A note in the right pane states: 'Имя поля может содержать не более 64 знаков (включая пробелы). Для получения справки по именам полей нажмите клавишу F1.' (The field name can contain no more than 64 characters (including spaces). To get help on field names, press the F1 key.)

Имя поля	Тип данных	Описание
Код исполнителя	Счетчик	
Фамилия	Текстовый	
Имя	Текстовый	
Отчество	Текстовый	
Должность	Текстовый	
Адрес электронной почты	Текстовый	
Рабочий телефон	Текстовый	
Домашний телефон	Текстовый	
Мобильный телефон	Текстовый	
Адрес	Текстовый	
Веб-страница	Гиперссылка	
Характеристика	Поле МЕМО	
Вложения	Вложение	

Рис. 9.2. Структура таблицы ИСПОЛНИТЕЛЬ

- Поскольку разрабатываемая форма предназначена только для запоминания идентификатора пользователя, приступающего к работе с приложением, создайте в ней поле со списком. Для этого выберите на ленте **Работа с макетами форм | Конструктор** (Form Layout Tools | Design) элемент управления **Поле со списком** и поместите его в области данных — единственном доступном разделе пустой формы. С помощью мастера определите в качестве источника строк поля со списком таблицу ИСПОЛНИТЕЛЬ и выберите поля: Код исполнителя, Фамилия, Имя, Отчество и Должность.
- Для созданного поля со списком мастер построит запрос на выборку заданных полей из таблицы ИСПОЛНИТЕЛЬ, который можно просмотреть в строке свойства **Источник строк** (Row Source) на вкладке **Данные** (Data), как в виде инструкции SELECT, так и в режиме конструктора запросов. Поскольку в поле со списком желательно отображать фамилию, имя и отчество исполнителя в едином поле, объедините эти поля в вычисляемом поле. Для этого в режиме

конструктора создайте вычисляемое поле, записав выражение [Фамилия] & " " & [Имя] & " " & [Отчество]. Поля с фамилией, именем и отчеством удалите. Закройте конструктор запросов, сохранив сделанные в нем изменения.

4. В окне свойств поля со списком на вкладке **Макет** (Format) измените **Число столбцов** (Column Count) на 3 (Код исполнителя, вычисляемое поле и Должность). В свойстве **Ширина столбцов** (Column Widths) уберите значения для несуществующих столбов и увеличьте ширину второго столбца списка, чтобы в нем размещались значения вычисляемого поля.
5. Настройте ширину самого поля со списком. Измените надпись поля со списком на Выберите исполнителя.
6. Для того чтобы в поле со списком по умолчанию выводилось значение первого исполнителя в таблице, в строку свойства **Значение по умолчанию** (Default Value) на вкладке **Данные** (Data) введите выражение =DFirst ("[Код исполнителя]" ; "[ИСПОЛНИТЕЛЬ]").
7. Сохраните форму под именем **Регистрация исполнителя**. Перейдите в режим формы и убедитесь, что в поле со списком отображаются фамилия, имя и отчество в одном поле, а должность в другом.
8. Для перехода к форме управления приложением и сохранения идентификатора исполнителя, выбранного в поле со списком, создайте кнопку и с событием **Нажатие кнопки** (On Click) свяжите встроенный макрос. Для этого воспользуйтесь мастером построения кнопки и выберите действие, которое будет выполняться при нажатии кнопки **Открыть форму** (Open Form) в категории **Работа с формой** (Form Operations). Будет построен внедренный макрос, связанный с нажатием кнопки, который включит макрокоманду **Открыть-Форму** (Open Form) с именем формы **Управление приложением**.
9. Дополните макрос макрокомандами, как показано на рис. 9.3.

Определение в макросе временной переменной, сохраняющей значение идентификатора текущего исполнителя, позволит использовать ее в других макросах приложения для выбора документов, назначенных для исполнения данному сотруднику.

1. Используйте свойства формы для приведения ее к виду, показанному на рис. 9.1.
2. Для того чтобы форма стала пользовательским диалоговым окном задайте на вкладке **Другие** (Other) свойствам **Модальное окно** (Modal) и **Всплывающее окно** (Pop Up) значение **Да** (Yes), а на вкладке **Макет** (Format) свойству **Тип границы** (Border Style) — **Окно диалога** (Dialog).

При открытии формы с этими свойствами делается недоступной работа со всеми объектами базы данных и отключается доступ ко всем инструментам на лентах, пока не будет закрыта форма. Окно такой формы не является вкладкой, как другие окна. Таким образом в нашем примере удастся потребовать от пользователя выбора исполнителя, необходимого для продолжения работы в вызываемом окне управления приложением.

```

/* Проверяет введено ли значение в поле со списком */
```

Если NotIsNull([ПолеСоСписком0]) то

/\* Эта переменная будет доступна в других макросах приложения \*/

ЗадатьВремПеременную

Имя ИД\_исполнителя

Выражение = [ПолеСоСписком0]

```

/* Закрывается активное окно формы */
```

ЗакрытьОкно

Тип объекта

Имя объекта

Сохранение Подсказка

ОткрытьФорму

Имя формы Управление приложением

Режим Форма

Имя фильтра

Условие отбора

Режим данных

Режим окна Обычное

ОстановитьМакрос

Конец блока "Если"

ОкноСообщения

Сообщение Выберите исполнителя

Сигнал Да

Тип Отсутствует

Заголовок

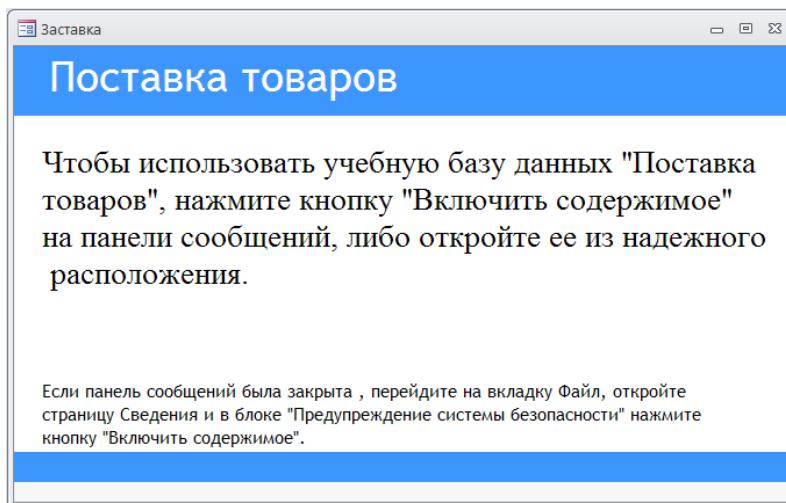
**Рис. 9.3.** Внедренный макрос, обеспечивающий вход в форму **Управление приложением**

### ЗАМЕЧАНИЕ

В модальной форме нельзя переключаться из режима формы в другие режимы, однако из области навигации можно открыть форму в режиме конструктора или макета. Форма в режиме конструктора или режиме макета не является модальной.

1. Для того чтобы форма отображалась сразу после открытия базы данных создайте автоматически запускающийся при этом макрос с именем **AutoExec** и включите в него макрокоманду **ОткрытьФорму** (OpenForm) с именем формы **Регистрация исполнителя**.
2. Если база данных не отнесена к разряду надежных, то при ее открытии будет выведена панель сообщений, предлагающая включить потенциально опасное

содержимое (см. рис. 4.39—4.41). Чтобы обратить внимание пользователя на ненадежность базы, включите в макрос макрокоманду **Если** (If), в условном выражении которой выполните проверку текущей базы данных на надежность Not [CurrentProject].[IsTrusted].



**Рис. 9.4.** Окно, предлагающее включить базу данных в список надежных

```

Если Not [CurrentProject].[IsTrusted] то
    ОткрытьФорму
        Имя формы Заставка
        Режим Форма
        Имя фильтра
        Условие отбора
        Режим данных
        Режим окна Обычное

Иначе
    ОткрытьФорму
        Имя формы Регистрация исполнителя
        Режим Форма
        Имя фильтра
        Условие отбора
        Режим данных
        Режим окна Обычное

Конец блока "Если"

```

**Рис. 9.5.** Макрос AutoExec с проверкой надежности базы данных

В случае ненадежности базы данных выведите окно формы **Заставка** (рис. 9.4) с сообщением о необходимости включить содержимое и только при ее надежности обеспечьте выполнение макрокоманды **ОткрытьФорму** (OpenForm) (рис. 9.5).

Нажатием кнопки **Включить содержимое** (Enable Content) вы присваиваете базе данных статус надежной. При этом повторяется проверка надежности базы данных и открывается форма **Управление приложением**. При следующем открытии базы данных из того же места панель сообщений выводиться не будет.

### **ЗАМЕЧАНИЕ**

Если при открытии базы данных нужно только автоматически открыть некоторую форму и нет необходимости в выполнении каких-либо дополнительных действий, можно вместо макроса **Autoexec** просто указать имя формы в параметрах Access. Для этого откройте вкладку **Файл** (File) и выберите пункт **Параметры** (Options). Далее выберите пункт **Текущая база данных** (Current Database) и в группе **Параметры приложений** (Application Options) в списке поля **Форма просмотра** (Display Form) выберите имя формы, которая должна отображаться при открытии базы данных.

## **Главная форма управления приложением**

Создадим главную форму управления приложением, в которой будут представлены:

- ❖ данные о невыполненных договорных обязательствах, за которые отвечает приступивший к работе исполнитель;
- ❖ данные о запасе товаров на складе;
- ❖ ссылки на наиболее часто используемые документы и отчеты;
- ❖ диаграмма стоимости товаров, не отгруженных по договорам.

Кроме того, предоставим возможность выбирать другого исполнителя и соответственно отображать связанные с ним данные.

Такая форма управления приложением в окончательном виде приведена на рис. 9.6.

1. Создайте пустую форму. Дополните открывшуюся в режиме макета форму заголовком, выполнив соответствующую команду на ленте **Конструктор** (Design) в группе **Колонтитулы** (Header/Footer).
2. Замените значение в надписи заголовка на Поставка товаров.
3. Скопируйте поле со списком из созданной ранее формы **Регистрация исполнителя**, открыв ее в режиме конструктора. В разрабатываемой форме перейдите в режим конструктора, увеличьте ширину заголовка формы и вставьте скопированное поле со списком в заголовок. Чтобы в поле со списком отображался исполнитель, выбранный в окне регистрации, в свойствах поля на вкладке **Данные** (Data) измените **Значение по умолчанию** (Default Value) на `= [TempVars] ! [ИД_исполнителя]`. Использованная здесь временная переменная была определена в макросе кнопки **Вход** из формы **Регистрация исполнителя**.
4. Дополните заголовок формы кнопками **Новый договор покупателя** и **Отгрузки по невыполненным договорам**. Свяжите с событием **Нажатие кнопки**

(On Click) вызов внедренного макроса с макрокомандой **ОткрытьФорму** (OpenForm). Для открытия формы в режиме ввода нового договора покупателя установите в макрокоманде **Режимданных** (DataView) — **Добавление**. Для формы с отгрузками по невыполненным договорам установите **Режим данных** (Data Mode) — **Только чтение** (Read Only).

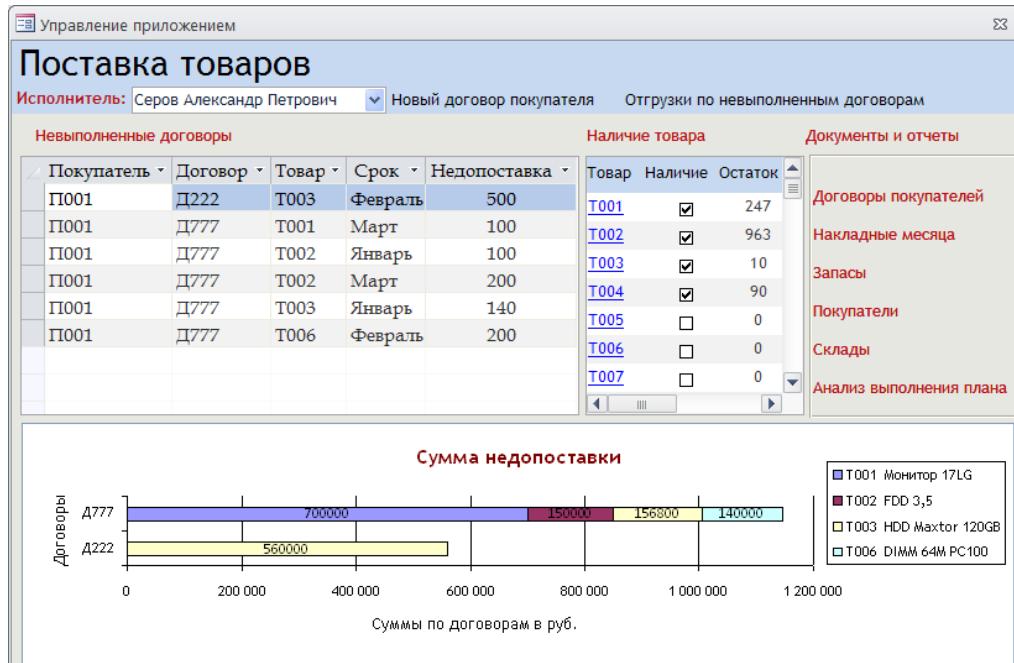


Рис. 9.6. Форма управления приложением

Для вывода в форме управления приложением данных о невыполненных договорах используйте подчиненную форму, построенную на запросе.

1. Создайте запрос, в котором из таблиц ПОСТАВКА\_ПЛАН, ДОГОВОР и ТОВАР выбираются необходимые поля и рассчитывается недопоставка. Запрос представлен на рис. 9.7. Условие отбора в поле **Недопоставка** позволит выбрать только те строки спецификации договора, в которых не выполнен план поставок. В запросе кроме отображенных на рисунке полей используйте еще два поля:
  - ◆ в первом для поля со сроком поставки — СРОК\_ПОСТ запишите условие отбора `<=Month(Date())`, что позволит выбрать только те записи, в которых срок поставки просрочен или истечет в текущем месяце;
  - ◆ во втором для вывода срока поставки в виде наименования месяца запишите вычисляемое поле выражение1: `MonthName([СРОК_ПОСТ])`.
2. Создайте форму с помощью мастера форм. В диалоге с мастером определите в качестве источника записей запрос Недопоставка и выберите необходимые поля (см. форму **Невыполненные договоры** на рис. 9.6). Выберите внешний вид

формы **табличный** (datasheet). Задайте имя формы Невыполненные договоры. Созданную форму закройте.

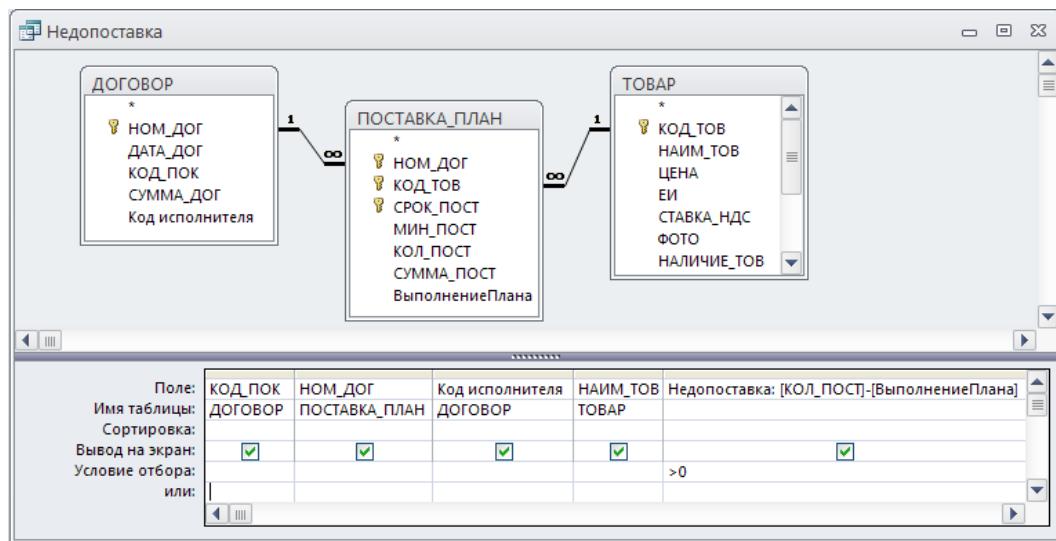


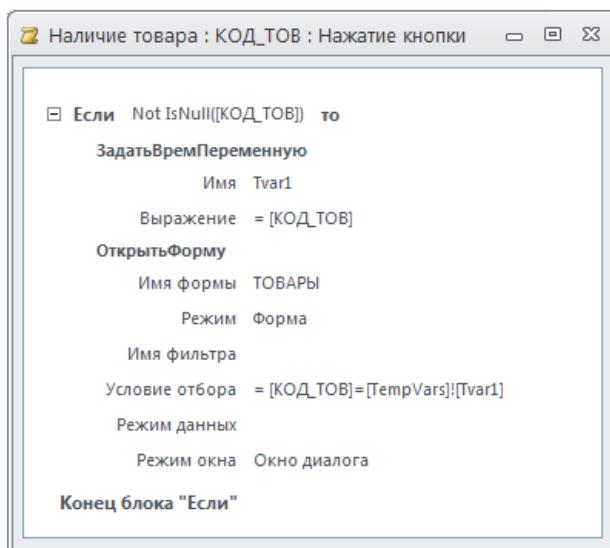
Рис. 9.7. Запрос – источник записей подчиненной формы Невыполненные договоры

- Перетащите созданную форму из области навигации в область данных пустой формы, открытой в режиме макета или конструктора.
- В свойствах вставленной подчиненной формы **Невыполненные договоры** на вкладке **Данные** введите Поле Со Списком0 в свойство **Основные поля** (Link Master Fields) и Код исполнителя в свойство **Подчиненные поля** (Link Child Fields). Проверьте, действительно ли поле со списком имеет указанное имя и включен ли в список полей, доступных в данном представлении Код исполнителя. Этот список можно открыть кнопкой **Добавить поля** (Add Existing Field) на ленте **Инструменты конструктора форм** (Form Design Tools), если форма открыта в режиме конструктора, или на ленте **Работа с макетами форм | Сервис** (Form Layout Tools | Tools), если форма открыта в режиме макета. После определения этих свойств в подчиненной форме будут выводиться только записи, связанные с выбранным в поле со списком исполнителем.
- Создайте надпись Невыполненные договоры и настройте элементы подчиненной формы как показано на рис. 9.6.

Для вывода в форме управления приложением данных о запасе товаров на складе используйте подчиненную форму, построенную на таблице ТОВАР.

- Создайте форму с помощью мастера форм. В диалоге с мастером определите в качестве источника записей таблицу ТОВАР. Выберите доступные поля: код товара, наличие товара, остаток и наименование товара. Выберите внешний вид формы **ленточный** (Continuous Forms). Задайте имя формы Наличие товара. Созданную форму закройте.

2. Перетащите созданную форму из области навигации в область данных пустой формы, открытой в режиме макета или конструктора.
3. В созданной таким образом подчиненной форме измените заголовок, оставив в нем только надписи полей.
4. Оформите поля с кодом и наименованием товара как гиперссылки. Для этого в свойстве макета **Отображать как гиперссылку** (Display As Hyperlink) выберите значение **Только на экране** (Screen Only). Теперь поля на экране будут выглядеть как гиперссылки. Однако по щелчку на такой "гиперссылке" никаких действий выполняться не будет.
5. Чтобы по щелчку на таком поле выполнялись нужные действия, создайте в окне свойств поля для события **Нажатие кнопки** (On Click) внедренный макрос (рис. 9.8). В макросе предусмотрите открытие формы ТОВАРЫ с вкладками **Сведения о товаре** и **Журнал отгрузок товара**. При этом в вызываемой форме должны отображаться только записи, связанные с выбранным в вызывающей форме товаром.



**Рис. 9.8. Макрос, внедренный в форму **Наличие товара****

6. Во внедренном макросе для ссылки на поле КОД\_ТОВ в вызывающей форме **Наличие товара** используйте временную переменную. Это позволит правильно сформировать условие отбора.
7. Аналогичным образом оформите поле Наименование товара. Поскольку для этого поля целесообразно предусмотреть выполнение такого же макроса, как и для поля Код товара, скопируйте внедренный макрос **КОД\_ТОВ : Нажатие кнопки**:
  - ◆ в окне свойств поля НАИМ\_ТОВ на вкладке **События** в строке **Нажатие кнопки** щелкните на значке построителя и выберите **Макросы**;
  - ◆ в открывшемся окне конструктора макросов откройте в каталоге макро-команд раздел **В этой базе данных** (In this Database); откройте список форм

и список формы **Наличие товара**, выберите макрос **КОД\_ТОВ.OnClick** и выполните команду его контекстного меню **Добавить копию макроса** (*Add Copy of Macro*);

8. Создайте надпись Наличие товара и настройте элементы подчиненной формы как показано на рис. 9.6.

Для вывода в форме управления приложением ссылок на наиболее часто используемые документы и отчеты используйте набор кнопок, заключенных в прямоугольник.

1. Создайте кнопки, используя соответствующий элемент управления на ленте конструктора. В диалоге с мастером выберите одно из действий, которое будет выполняться при нажатии кнопки **Открыть форму** (Open Form) или **Открыть отчет** (Open Report). Эти действия содержатся в категориях **Работа с формой** (Form Operations) и **Работа с отчетом** (Report Operations).
2. Выберите форму или отчет, для открытия которого предназначена кнопка. Введите текст, который нужно разместить на кнопке, соответствующий названию документа или отчета. Мастер для открытия выбранного объекта создаст внешний макрос, который будет связан с событием **Нажатие кнопки** (On Click).
3. Чтобы кнопка приняла вид простой надписи, выберите в свойствах макета **Тип границы** (Border Style) — **Отсутствует** (Transparent), а **Тип фона** (Back Style) — **Прозрачный** (Transparent).
4. Создайте надпись Документы и отчеты, используя соответствующий элемент управления на ленте конструктора.

Для вывода в форме управления приложением диаграммы стоимости товаров, не отгруженных по договорам, используйте рассмотренный выше запрос **Недоставка** (см. рис. 9.7). В режиме SQL этот запрос имеет вид:

```
SELECT ДОГОВОР.КОД_ПОК, ПОСТАВКА_ПЛАН.НОМ_ДОГ, ДОГОВОР.[Код исполнителя],  
ТОВАР.НАИМ_ТОВ, [КОЛ_ПОСТ]-[ВыполнениеПлана] AS Недоставка,  
ПОСТАВКА_ПЛАН.КОД_ТОВ AS Товар, ([КОЛ_ПОСТ]-[ВыполнениеПлана])*[ЦЕНА] AS Стоимость,  
MonthName([СРОК_ПОСТ]) AS Выражение1, [ПОСТАВКА_ПЛАН]![КОД_ТОВ] & " "  
& [НАИМ_ТОВ] AS Объединение  
  
FROM ДОГОВОР INNER JOIN (ТОВАР INNER JOIN ПОСТАВКА_ПЛАН ON ТОВАР.КОД_ТОВ =  
ПОСТАВКА_ПЛАН.КОД_ТОВ) ON ДОГОВОР.НОМ_ДОГ = ПОСТАВКА_ПЛАН.НОМ_ДОГ  
WHERE ((([КОЛ_ПОСТ]-[ВыполнениеПлана])>0) AND  
([ПОСТАВКА_ПЛАН.СРОК_ПОСТ]<=Month(Date())));
```

1. Для создания диаграммы откройте форму в режиме конструктора и с помощью элемента управления **Диаграмма** (Chart) на ленте конструктора нарисуйте область диаграммы.
2. В диалоге с мастером выберите запрос **Недоставка** и поля, которые необходимы для построения диаграммы.
3. Выберите тип диаграммы **Линейчатая** (Bar Chart) и тип отображения данных на ней.
4. Для изменения диаграммы при переходе между значениями поля со списком выберите поля, связывающие его с диаграммой: ПолеСоСписком0 (проверьте имя поля со списком) и Код исполнителя.

5. В области диаграммы отобразится макет. Для просмотра диаграммы перейдите в режим формы.
6. Для изменения диаграммы в режиме конструктора выделите ее и выполните команду контекстного меню **Объект Диаграмма | Изменить** (Chart Object | Edit). Двойной щелчок мыши в области диаграммы также позволяет перейти в режим ее изменения. Чтобы отражался вклад каждой категории в общей сумме измените тип диаграммы на **Линейчатая с накоплением** (Stacked Bar). Для этого выделите **Область построения диаграммы** и выполните команду **Тип диаграммы** (Chart Type).
7. Выполнив команду контекстного меню **Параметры диаграммы** (Chart Options), настройте вид диаграммы: определите заголовки осей, подписи данных. Для возврата в форму щелкните в любой ее части за пределами диаграммы.
8. Если при создании диаграммы не была установлена ее связь с полем со списком, выделите диаграмму и откройте ее свойства. На вкладке **Данные** (Data) введите имена полей связи. В строку **Основные поля** (Link Master Fields) введите ПолеСоСписком0, в строку **Подчиненные поля** (Link Child Fields) — Код исполнителя.
9. В строке **Источник строк** (Row Source) диаграммы просмотрите перекрестный запрос. В режиме SQL инструкция TRANSFORM этого источника строк будет записана в виде:

```
TRANSFORM Sum(Недопоставка.[Стоимость]) AS Сумма_Стоимость
  SELECT Недопоставка.[НОМ_ДОГ]
    FROM Недопоставка
   GROUP BY Недопоставка.[НОМ_ДОГ]
PIVOT Недопоставка.Объединение;
```

TRANSFORM указывает, по какому полю и с помощью какой статистической функции рассчитываются значения рядов данных диаграммы (столбцов) Sum(Недопоставка.[Стоимость]) AS Сумма\_Стоимость.

Встроенная инструкция SELECT указывает поле Недопоставка.[НОМ\_ДОГ], значения которого используются на оси категорий. Предложение GROUP BY задает группировку строк запроса по этому полю.

Предложение PIVOT определяет поле, значения которого будут использованы в легенде диаграммы, определяющей названия рядов данных. Кроме того, по этому полю производится группировка среди записей, сгруппированных по значениям строк. Заметим, что в предложении PIVOT вместо поля может использоваться выражение или набор фиксированных значений.

## Использование форм с вкладками при разработке интерфейса

При разработке интерфейса приложения для представления функционально связанных объектов в одной форме может быть использован набор вкладок. Раз-

мещение на страницах вкладок различных подчиненных форм/отчетов позволяет удобно организовать работу специалиста с необходимыми при решении конкретных задач документами. При этом они всегда доступны и нет необходимости отвлекаться на их поиск.

Создадим форму, на вкладках которой представим данные, необходимые для оценки ситуации по отгрузке каждого заказанного в договорах товара:

- ❖ данные о запланированных по договорам отгрузках;
- ❖ данные о фактических отгрузках товара;
- ❖ данные по анализу итогов отгрузки товара.

Такая форма с именем **Отгрузка товаров по договорам** в окончательном виде приведена на рис. 9.9. На вкладках формы могут размещаться как формы, так и отчеты, в приведенном примере они связаны с основной формой первой вкладки, т. е. отображают данные об отгрузке товара, выбранного на первой вкладке.

1. Создайте пустую форму. Дополните открывшуюся в режиме макета форму заголовком, выполнив соответствующую команду на ленте **Конструктор** (Design) в группе **Колонтитулы** (Header/Footer).
2. Замените значение в надписи заголовка на ВСЕ ОБ ОТГРУЗКЕ ТОВАРА.

**Отгрузка товаров по договорам**

**ВСЕ ОБ ОТГРУЗКЕ ТОВАРА Монитор 17LG** 19 апреля 2010 г.

Товар	План отгрузки	Отгружено	Итого по товару										
Код товара	T001												
Наименование товара	Монитор 17LG												
Цена	7 000,00р.												
Ставка НДС	5%												
Наличие товара	<input checked="" type="checkbox"/>												
Остаток	500												
На сегодня отгружено по договорам													
<table border="1"> <thead> <tr> <th>Доставка</th> <th>План отгрузки</th> </tr> </thead> <tbody> <tr> <td>Д111</td> <td>50</td> </tr> <tr> <td>Д222</td> <td>40</td> </tr> <tr> <td>Д333</td> <td>30</td> </tr> </tbody> </table>				Доставка	План отгрузки	Д111	50	Д222	40	Д333	30		
Доставка	План отгрузки												
Д111	50												
Д222	40												
Д333	30												
<table border="1"> <thead> <tr> <th>Доставка</th> <th>Отгружено</th> </tr> </thead> <tbody> <tr> <td>Д111</td> <td>700</td> </tr> <tr> <td>Д222</td> <td>120</td> </tr> <tr> <td>Д333</td> <td>50</td> </tr> <tr> <td>Д777</td> <td>400</td> </tr> </tbody> </table>				Доставка	Отгружено	Д111	700	Д222	120	Д333	50	Д777	400
Доставка	Отгружено												
Д111	700												
Д222	120												
Д333	50												
Д777	400												

Запись: 1 из 10 | Нет фильтра | Поиск

**Рис. 9.9.** Форма с вкладками для получения сведений об отгрузках товара

3. На ленте **Конструктор** (Design) щелкните на элементе **Вкладка** (Tab Control) и вычертите в области данных формы область вкладок. В форме будут созданы две вкладки с произвольными подписями типа Вкладка3 и Вкладка4.
4. Одновременно с созданием пустой формы открывается список полей, в котором отображена запись: **Нет полей, доступных для добавления в текущее представление** (No fields available to be added to the current view). При этом возможно открытие списка всех таблиц базы данных. Щелкните на ссылке **Показать все таблицы** (Show all tables) и переместите в область первой вкладки поля таблицы ТОВАР, приведенные на вкладке формы **Товар**. Если список полей не отобразился после создания пустой формы, выполните команду **Добавить поля** (Add Existing Field) на ленте **Конструктор** (Design) в группе **Сервис** (Tools).
5. После перемещения полей на вкладку таблица ТОВАР будет отображена в разделе списка полей: **Поля, доступные в данном представлении** (Fields available for this view), а для формы в качестве источника записей создан запрос на выборку — представление, записанное в соответствующей строке свойств формы инструкцией SQL на вкладке **Данные** (Data).
6. Измените отображаемую подпись вкладки на Товар. Для этого откройте свойства вкладки и в строке **Подпись** (Caption) на вкладке свойств **Макет** (Format) введите Товар.
7. Постройте диаграммы для отображения на вкладке сведений о плановых и фактических отгрузках товара. Для создания источника строк диаграмм используйте запросы, представленные на рис. 9.10 и 9.11.

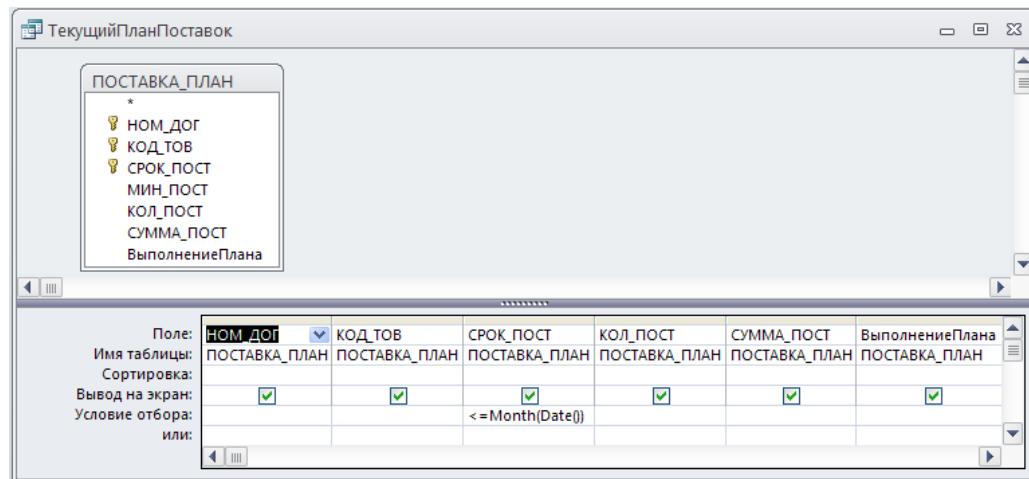


Рис. 9.10. Запрос для получения сведений о запланированных поставках товара

- В этом запросе отбираются записи со сроком поставки товаров до текущего месяца.
1. Для создания диаграммы перейдите в режим конструктора и на ленте выберите элемент управления **Диаграмма** (Chart). Вычертите область размещения диаграммы.

грамм на вкладке — ее рамку. Предварительно увеличите область, занимаемую вкладкой.

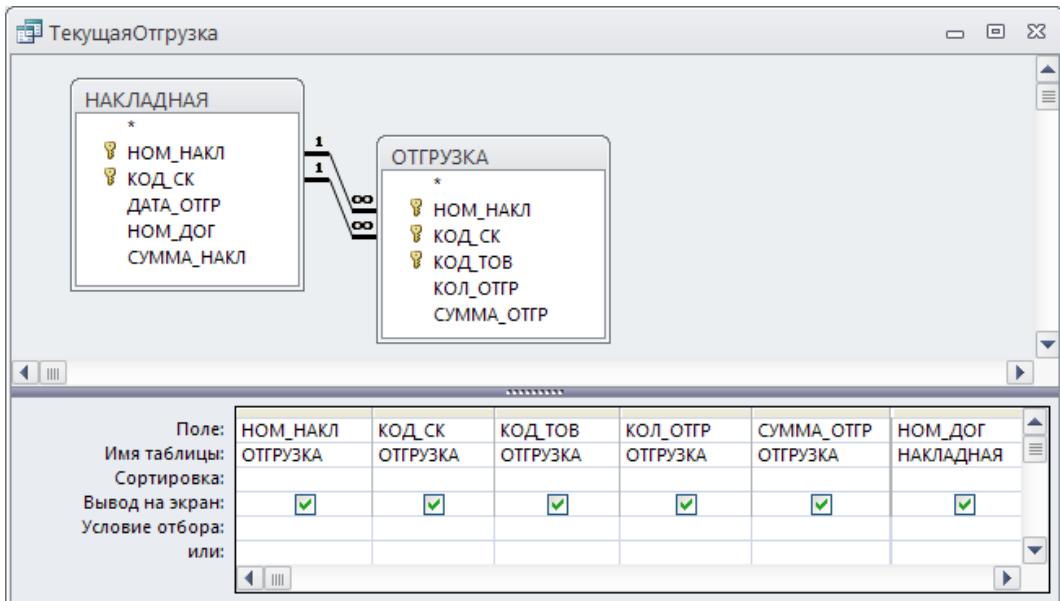


Рис. 9.11. Запрос для получения сведений о реальных отгрузках товара на текущую дату

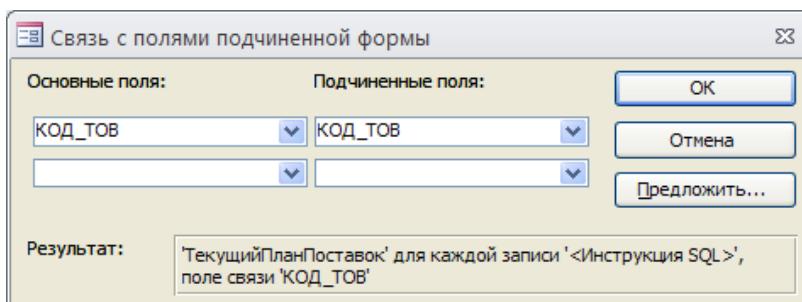
2. В диалоговом окне запущившегося мастера создания диаграмм выберите нужный запрос, например **ТекущийПланПоставок**.
3. В следующем окне мастера выберите поля с данными, которые необходимо отобразить в диаграмме: НОМ\_ДОГ — номер договора, КОД\_ТОВ — код товара, КОД\_ПОСТ — количество, которое необходимо поставить.
4. В следующем окне выберите тип диаграммы: **Гистограмма (Column Chart)**.
5. Оставьте предложенный мастером тип отображения данных на диаграмме, т. к. он соответствует заданному в форме на вкладке **Товар**. Номера договоров отобразятся на оси, код товара в рядах, суммарное количество товара, поставленного по каждому из договоров, в области данных.
6. В следующем окне для изменения диаграммы при переходе между записями выберите поле, связывающее документ с диаграммой — КОД\_ТОВ.
7. Задайте название диаграммы: На сегодня должно быть поставлено по договорам.
8. Для правильного отображения всех элементов диаграммы измените размеры рамки диаграммы.
9. Для изменения диаграммы в целом или ее отдельных элементов, находясь в режиме конструктора формы, выполните двойной щелчок на ней. Доступный теперь инструментарий используйте для настройки диаграммы. Для возврата в конструктор форм щелкните в любом месте формы за пределами диаграммы.

10. Аналогичным образом постройте диаграмму на основе запроса Текущая отгрузка, выбрав поля с данными, которые необходимо отобразить в диаграмме: НОМ\_ДОГ — номер договора, КОЛ\_ОТГР — количество, которое было отгружено, КОД\_ТОВ — код товара. Если тип отображения данных на диаграмме не соответствует желаемому, переместите поля с диаграммы в список и затем переместите из списка поле НОМ\_ДОГ на **Оси** (Axis), поле КОД\_ТОВ в **Ряды** (Data Series) и поле КОЛ\_ОТГР в **Данные** (Data).

11. Для отображения в заголовке основной формы наименования выбранного товара создайте там поле, в которое запишите выражение = [НАИМ\_ТОВ].

На следующей вкладке организуйте вывод подробных данных о запланированных по договорам отгрузках выбранного товара. Для этого:

1. Создайте на основе запроса **ТекущийПланПоставок** ленточную форму, выбрав этот запрос и выполнив на ленте **Создание** (Create) в списке **Другие формы** (More Forms) команду **Несколько элементов** (Multiple Items).
2. Вставьте в заголовок формы дату, переместите в него поле код товара, поскольку на вкладке отображаться будет только один вид товара.
3. В примечании формы создайте поле, в которое введите выражение для подсчета общего количества отгруженного товара =Sum( [КОЛ\_ПОСТ] ).
4. Перетащите созданную форму на вторую вкладку формы, открытой в режиме макета или конструктора.
5. Если при этом не установилась связь с основной записью на первой вкладке, откройте свойства вставленной формы и на вкладке **Данные** (Data) щелкните на значке построителя в строке **Основные поля** (Link Master Fields). В открывшемся окне (рис. 9.12) в качестве поля связи должно быть выбрано поле КОД\_ТОВ. Очевидно, что можно КОД\_ТОВ ввести в строки свойств **Основные поля** (Link Master Fields) и **Подчиненные поля** (Link Child Fields) вручную, не обращаясь к помощи построителя.



**Рис. 9.12.** Определение связи форм на первой и второй вкладках

6. Измените отображаемую подпись вкладки на План отгрузки. Для этого откройте свойства вкладки и в строке **Подпись** (Caption) на вкладке свойств **Макет** (Format) введите План отгрузки.

Форма с подробными данными о запланированных по договорам отгрузках выбранного товара, размещенная на второй вкладке, представлена на рис. 9.13.

На	Товар	Номер договора	Срок поставки	Количество	Сумма	Выполнено
20.04.2010	T001	Д111	1	305	1 000 000,00р.	191
		Д111	2	100	2 000 000,00р.	100
		Д111	3	300	3 000 000,00р.	0
		Д222	1	100	658 700,00р.	100
		Д222	2	12	79 044,00р.	12
		Д777	1	10	0,00р.	10
		Д777	2	300	0,00р.	300
		Д777	3	100	0,00р.	0
		Д222	4	10	0,00р.	0
		Д333	3	50	0,00р.	0
				Итого запланировано:	1287	
Запись: 1 из 10   Нет фильтра   Поиск						

**Рис. 9.13.** Форма с данными о запланированных поставках товара на вкладке **План отгрузки**

### Задание 9.1

Добавьте новую вкладку и организуйте на ней вывод подробных данных о фактических отгрузках выбранного товара по договорам. Для этого:

1. Для добавления новой вкладки в форму щелкните на имеющейся вкладке и выполните команду контекстного меню **Вставить вкладку** (Insert Page).
2. Также как в предыдущем случае создайте на основе запроса **ТекущаяОтгрузка** ленточную форму, доработайте форму, вставьте ее на новую вкладку, установите необходимую связь этой формы с записью первой вкладки и измените подпись вкладки на **Отгружено**.
3. Если возникла необходимость изменения последовательности размещения вкладок в форме, используйте команду ее контекстного меню **Последовательность вкладок** (Page Order). В открывшемся окне **Порядок страниц** (Page Order) любая из вкладок может быть перемещена в нужное место.

### Задание 9.2

Добавьте еще одну вкладку **Итого по товару**. Организуйте на ней вывод итоговых данных о плановых, фактических отгрузках выбранного товара и отклонении от плана. Для получения результата используйте запросы, позволяющие подсчитывать общее количество каждого из товаров, запланированного на текущий месяц, фактически отгруженного на текущую дату и разность между ними (аналогичные запросы для решения этой задачи **План, Факт и Анализ выполнения плана** приведены в главе 4, не используйте в них параметров). На основе последнего запроса создайте отчет, который так же просто как форма размещается на странице вкладки.

### Панель навигации формы с вкладками

Заголовок и Панель навигации (кнопки перехода по записям) основной формы, размещенной на первой вкладке, сохраняется на всех других вкладках формы. Это позволяет переходить к другой записи основной формы и просматривать связанные данные на любой другой вкладке, например со сведениями по отгрузкам любого товара, не возвращаясь для выбора к первой вкладке.

Пользуйтесь фильтрами для перехода к нужной записи.

1. На вкладке **Товары** установите курсор на поле с кодом товара и в контекстном меню выполните команду **Текстовые фильтры | Равно** (Text Filters | Equals).
2. В диалоговом окне **Настраиваемый фильтр** (Custom Filter) введите нужный код товара (рис. 9.14). В форме отобразится только запись с введенным кодом товара.

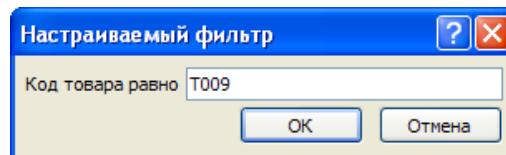


Рис. 9.14. Диалоговое окно ввода значения для фильтра

3. Уберите фильтр, щелкнув мышью на кнопке панели навигации **С фильтром** (Filtered). Для повторного использования последнего фильтра щелкните на кнопке **Без фильтра** (Unfiltered).
4. Чтобы совсем удалить фильтр, выполните команду контекстного меню **Снять фильтр с Код товара** (Clear filter from Код товара). При этом кнопка фильтра тускнеет и получает имя **Нет фильтра** (No Filter).

Текстовые фильтры позволяют отобрать записи не только по равенству значений в поле, а и по значительно более сложным критериям, таким как **Начинается с...** (Begins With...), **Заканчивается на...** (Ends With...) и т. д.

Для числовых полей используйте **Числовые фильтры** (Number Filters), позволяющие отобрать записи со значениями, заданными условием фильтрации.

Для перехода к нужной записи можно воспользоваться инструментами поиска.

На панели навигации щелкните в поле **Поиск** (Search). Курсор может быть установлен в любом поле. Начинайте ввод нужного наименования товара. По мере ввода значения будет отображаться запись с наименованием товара, начинающимся с введенных символов. Если нет товара с наименованием, начинающимся с введенного символа или символов, текущей остается старая запись. Можно вводить и код товара. Поиск осуществляется по всем полям записей, в том числе и по числовым. Если ввести одну или несколько букв или цифр, повторным нажатием на клавишу <Enter> можно просмотреть все значения, начинающиеся с них.

#### Задание 9.3

С помощью фильтра отберите только те товары, которые имеются в наличии. Установите курсор на поле **Наличие товара** и в контекстном меню выполните команду **Выделено** (Is Selected) или **Не выделено** (Is Not Selected), в зависимости от того, свидетельствует значение о наличии товара или нет. Будут отобраны все записи, в которых значение в поле **Наличие товара** равно или не равно значению в текущей записи.

#### Задание 9.4

Создайте форму на основе таблицы ИСПОЛНИТЕЛЬ и в нее вставьте вкладки со связанными формами и отчетами о деятельности этого исполнителя. В заголовке формы выведите должность и фамилию исполнителя.

## Формы навигации

Форма навигации предназначена для создания на ней горизонтальных, вертикальных или тех и других вкладок одновременно. На этих вкладках могут размещаться формы и отчеты. Для размещения формы или отчета на странице вкладки достаточно выбрать его в области переходов и перетащить в ряд кнопок навигации, первоначально вставляя новую вкладку перед словом **Создать** (Add New). Можно также ввести имя формы или отчета вместо слова **Создать** (Add New) на кнопке навигации. Впоследствии можно изменить имя на кнопке навигации и при этом на странице сохраняется ранее размещенный объект. Даже если на кнопке навигации имя меняется на совпадающее с именем некоторой формы или отчета, на странице вкладки сохраняется прежний объект.

Для удаления формы на странице вкладки нужно выделить кнопку навигации и выполнить команду контекстного меню **Удалить** (Delete) или просто нажать клавишу <Delete>.

Размещенная на одной странице вкладки подчиненная форма/отчет не связывается с объектами, размещенными на страницах других вкладок.

Создайте форму навигации с горизонтальным расположением вкладок в два уровня. Для этого выполните соответствующую команду, расположенную на ленте **Создание** (Create) в группе **Формы** (Forms) в списке **Навигация** (Navigation).

Кнопки навигации первого уровня используйте для открытия группы вкладок второго уровня. Дайте этим кнопкам имена, которые будут определять суть содержимого группы, например Справочники, Плановые документы, Оперативные документы, Отчеты (рис. 9.15). Имена не должны совпадать с именами имеющихся в базе форм и отчетов.

**Форма навигации**

<b>Справочники</b>	<b>Плановые документы</b>	<b>Оперативные документы</b>	<b>Отчеты</b>		
Отгрузки каждого товара по месяцам	Все отгрузки по месяцам	Стоймость договоров покупателей			
2 апреля 2010 г.					
<b>ОТГРУЗКА ТОВАРА</b>		<b>Монитор 17LG</b>	<b>код Т001</b>		
Цена	7 000,00р.				
Единица измерения	штука				
Ставка НДС	5%				
Наличие товара	<input checked="" type="checkbox"/>				
	Дата отгрузки	Код накладной	Код склада	Количество отгружено	Сумма отгружено
<b>Январь 2010</b>					
	10.01.2010	H005	C01	40	280 000,00р.
	21.01.2010	H004	C02	10	70 000,00р.
	18.01.2010	H001	C01	50	350 000,00р.
	<b>Итого за месяц</b>			100	700 000,00р.
<b>Февраль 2010</b>					
	11.02.2010	H002	C02	30	210 000,00р.
	11.02.2010	H002	C01	40	280 000,00р.
	<b>Итого за месяц</b>			70	490 000,00р.
<b>Март 2010</b>					
	25.03.2010	H004	C01	2	14 000,00р.
	24.03.2010	H002	C03	30	210 000,00р.
	<b>Итого за месяц</b>			32	224 000,00р.
	<b>Общая сумма</b>			1 414 000,00р.	

**Рис. 9.15.** Форма навигации с горизонтальными вкладками в два уровня

Разместите на страницах вкладок второго уровня формы справочников базы данных, формы для работы с плановыми документами, такими как договоры, оперативно-учетными документами, такими как накладные, отчеты, позволяющие проанализировать текущую ситуацию. Для размещения нужных форм и отчетов на страницах вкладок достаточно перетаскивать их в нужные места в ряду кнопок навигации или вводить их имена на кнопках навигации.

Формы, размещаемые на страницах вкладок, в свою очередь могут включать вкладки, обеспечивая таким образом дополнительные уровни иерархии.

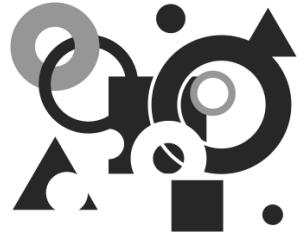
Очевидно, формы навигации должны настраиваться на работу специалиста с определенным кругом обязанностей и в организации для каждой группы сотрудников, имеющих схожие функциональные обязанности, могут быть разработаны соответствующие формы навигации.

## Контрольные вопросы

1. К каким последствиям приводят значения свойств формы **Модальное окно** (Modal) и **Всплывающее окно** (Pop Up)?
2. Можно ли в поле со списком в качестве источника строк использовать вычисляемое поле?
3. Может ли в поле со списком вводиться значение по умолчанию?
4. В какой момент выполняется макрос с именем **AutoExec**?
5. К каким последствиям приводит нажатие кнопки **Включить содержимое** (Enable Content) на панели сообщений?
6. Какая инструкция языка SQL записывается в строке **Источник строк** (Row Source) диаграммы?
7. Возможно ли установление связи между записями различных вкладок?
8. Сохраняются ли на второй и последующих вкладках кнопки перехода по записям формы с первой вкладки?
9. Какие объекты могут размещаться на вкладках формы навигации?
10. Можно ли поместить форму на вкладку формы навигации, заменив на кнопке навигации слово **Создать** (Add New) именем формы?
11. Допускает ли форма навигации создание нескольких уровней вкладок?

## Ответы

1. При открытии формы с этими свойствами делается недоступной работа со всеми объектами базы данных и отключается доступ ко всем инструментам на лентах.
2. Да.
3. Да.
4. В момент открытия базы данных.
5. Разрешает выполнение в базе данных опасного содержимого, такого как запросы на изменение.
6. Инструкция TRANSFORM.
7. Да.
8. Да.
9. Формы и отчеты.
10. Да.
11. Да.



## ПРИЛОЖЕНИЕ 1

# Структура таблиц базы данных *Поставка товаров*

## Таблицы справочных данных

*Таблица П1.1. Описание свойств полей таблицы ТОВАР*

Имя поля	Признак первично-го ключа	Обяза-тельное поле	Тип данных	Размер	Формат	Число дес. знаков	Подпись поля
КОД_ТОВ	Ключ простой	Да	Текстовый	5			Код товара
НАИМ_ТОВ		Нет	Текстовый	25			Наимено-вание товара
ЦЕНА		Нет	Денежный		Денежный	2	Цена
ЕИ		Нет	Текстовый	8			Единица измерения
СТАВКА_НДС		Нет	Числовой	Одинарное с плаваю-щей точкой	Процентный	0	Ставка НДС
ФОТО		Нет	Поле объекта OLE	До 1 Гбайт			Фото товара
НАЛИЧИЕ_ТОВ		Нет	Логиче-ский	1 бит	Да/Нет		Наличие товара на складе
ВЛОЖЕНИЯ		Нет	Вложение	700 Кб			Поле вложений

Требования к заполнению таблицы:

1. Для поля СТАВКА\_НДС нужно предусмотреть:
  - ◆ Условие на значение:  $>=0,05$  And  $<=0,35$ ;
  - ◆ Сообщение об ошибке: "Ставка НДС должна быть  $>=5\%$  и  $<=35\%$ ".

2. Для поля ЦЕНА нужно предусмотреть:
  - ◆ **Условие на значение:**  $>=0$  And  $<=35000$ ;
  - ◆ **Сообщение об ошибке:** "Цена должна быть  $>=0$  и  $<=35000$ ".
3. Для данной таблицы и остальных: если для поля указан признак первичного ключа, это означает следующее:
  - ◆ когда ключ "простой", поле надо выделить и присвоить признак ключа (нажатием кнопки **Ключевое поле**), при этом в свойстве **Индексированное поле** автоматически устанавливается значение **Да (Совпадения не допускаются)**;
  - ◆ когда ключ "составной", надо выделить все поля, образующие первичный ключ, и только после этого присвоить признак ключа. При этом в свойстве **Индексированное поле** для каждого поля, входящего в ключ, нужно оставить значение по умолчанию **Нет**.
4. Неуказанные в таблице параметры в конструкторе таблиц должны сохранить значение по умолчанию.

**Таблица П1.2. Описание свойств полей таблицы СКЛАД**

Имя поля	Признак первичного ключа	Обязательное поле	Тип данных	Размер	Подпись поля
КОД_СКЛ	Ключ простой	Да	Текстовый	5	Номер склада
КОД_Ф		Нет	Текстовый	5	Код фирмы
НАИМ_СК		Нет	Текстовый	20	Наименование склада
ОТВ_ЛИЦО		Нет	Текстовый	20	Ответственное лицо
АДРЕС_СК		Нет	Текстовый	20	Адрес склада

**Таблица П1.3. Описание свойств полей таблицы ПОКУПАТЕЛЬ**

Имя поля	Признак первичного ключа	Обязательное поле	Тип данных	Размер	Подпись поля
КОД_ПОК	Ключ простой	Да	Текстовый	5	Код покупателя
ИНН		Нет	Текстовый	12	
НАИМ_ПОК		Нет	Текстовый	20	Наименование
АДРЕС_ПОК		Нет	Текстовый	20	Адрес
ТЕЛ		Нет	Текстовый	10	Телефон
НОМ_РСЧ		Нет	Текстовый	20	Номер расчетного счета
БАНК		Нет	Текстовый	10	
ОПИСАНИЕ		Нет	Поле МЕМО		
WEB_АДРЕС		Нет	Гиперссылка		

Требования к заполнению таблицы:

1. Для поля ТЕЛ следует задать маску ввода: \ (999\ ) 000\ -0099.
2. Для поля ИНН нужно выполнить следующие операции:
  - ◆ указать маску ввода: 000000000000;
  - ◆ в свойстве **Индексированное поле** выбрать значение **Да (Совпадения не допускаются)**.

## Таблицы плановых данных

*Таблица П1.4. Описание свойств полей таблицы ДОГОВОР*

Имя поля	Признак первично-го ключа	Обяза-тельное поле	Тип данных	Размер	Формат	Число дес. знаков	Подпись поля
НОМ_ДОГ	Ключ простой	Да	Текстовый	5			Номер договора
ДАТА_ДОГ		Нет	Дата/время	5	Краткий формат даты		Дата
КОД_ПОК		Да	Текстовый	5			Код покупателя
СУММА_ДОГ		Нет	Денежный		Денежный	Авто	Сумма по договору

Маска ввода поля ДАТА\_ДОГ:

- ◆ в кратком формате: 00.00.0000;
- ◆ в среднем формате: 00->L<LL-0000.

*Таблица П1.5. Описание свойств полей таблицы ПОСТАВКА\_ПЛАН*

Имя поля	Признак первично-го ключа	Обяза-тельное поле	Тип данных	Размер	Формат	Число дес. знаков	Подпись поля
НОМ_ДОГ	Ключ составной	Да	Текстовый	5			Номер договора
КОД_ТОВ		Да	Текстовый	5			Код товара
СРОК_ПОСТ		Да	Числовой	Байт	Фиксиро-ванный		Срок поставки (№ месяца)
МИН_ПОСТ		Нет	Числовой	Целое		Авто	Минимальная партия поставки

**Таблица П1.5 (окончание)**

Имя поля	Признак первично-го ключа	Обяза-тельное поле	Тип данных	Размер	Формат	Число дес. знаков	Подпись поля
КОЛ_ПОСТ		Нет	Числовой	Длинное целое		Авто	Количество поставки
СУММА_ПОСТ		Нет	Денежный		Денежный	Авто	Сумма поставки

## Таблицы оперативно-учетных данных

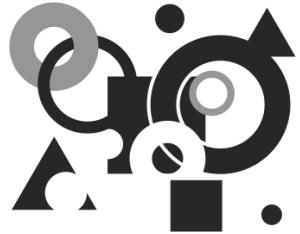
**Таблица П1.6. Описание свойств полей таблицы НАКЛАДНАЯ**

Имя поля	Признак первично-го ключа	Обяза-тельное поле	Тип данных	Размер	Формат	Число дес. знаков	Подпись поля
НОМ_НАКЛ	Ключ составной	Да	Текстовый	5			Номер накладной
КОД_СК		Да	Текстовый	5			Код склада
ДАТА_ОТГ		Нет	Дата/время		Краткий формат даты		Дата отгрузки
НОМ_ДОГ		Да	Текстовый	5			Номер договора
СУММА_НАКЛ		Нет	Денежный		Денежный	Авто	Сумма по накладной

Необходимо предусмотреть для поля ДАТА\_ОТГ ввод значения по умолчанию текущей даты: Date() .

**Таблица П1.7. Описание свойств полей таблицы ОТГРУЗКА**

Имя поля	Признак первичного ключа	Обяза-тельное поле	Тип данных	Размер	Формат	Число дес. знаков	Подпись поля
НОМ_НАКЛ	Ключ составной	Да	Текстовый	5			Номер накладной
КОД_СК		Да	Текстовый	5			Код склада
КОД_ТОВ		Да	Текстовый	5			Код товара
КОЛ_ОТГР		Нет	Числовой	Длинное целое		Авто	Количество
СУММА_ОТГР		Нет	Денежный		Денежный	Авто	Сумма по товару



## ПРИЛОЖЕНИЕ 2

# Пример заполненных документов для загрузки в базу данных *Поставка товаров*

## Справочная информация

Справочник товаров					
Код товара	Наименование товара	Цена	Единица изм	Ставка НДС	
T001	Монитор 17LG	6 587,00р.	штука	5%	
T002	FDD 3,5	363,00р.	коробка	20%	
T003	HDD Maxtor20GB	2 590,00р.	штука	10%	
T004	Корпус MiniTower	1 916,00р.	штука	20%	
T005	CD-ROM Panasonic IDE	1 153,00р.	штука	30%	
T006	DIMM64MPC100	360,00р.	штука	15%	
T007	Принтер HP LaserJet1220	5 432,00р.	штука	10%	
T008	СканерAcer	2 338,00р.	штука	15%	
T009	Зв. Карта Genius Liv	789,00р.	штука	5%	
T010	Модем Genius ext	1 295,00р.	штука	5%	
T011	Плоттер	3 000,00р.	штука	10%	
T012	Миникомпьютер	39 000,00р.	штука	15%	

Справочник складов				
Номер склада	Код фирмы	Наименование	Отв. лицо	Адрес скл.
01	100	Главный	Иванов Т.С.	Мичуринская,
02	100	Оптовый	Петров А.А.	Свердлова,29
03	100	Торговый	Смирнов О.Н.	Речной, 38

Справочник покупателей						
Код покупателя	ИНН	Наименование	Адрес	Телефон	Номе расч. счета	БАНК
П001	778957651111	Компьютер маркет	Витебская, 12	(812)345-2345	76358509763264536567	Мост
П002	789889798798	Перспектива	Ладожский, 38	(995)345-6789	29384789823942938489	Кредит
П003	770239413213	Инфоцентр	Кодратьевский, 54	(234)234-45	76358509763264536765	Мост
П004	456575675677	Монитор	Средний.4	( )123-4567	5856878976897467854	Надежда
П005	771243567822	Компьютер лэнд	Кавалергардский, 9	( )123-56	59796908790879842589	Ивест
П006	587879879123	Компьютерная техника	Малый,19	( )123-4567	76358509763264536354	Мост

## Плановая информация

ДОГОВОР N		Д11	От	05.01.2007
ПОКУПАТЕЛЬ		Перспектива		
Код покупателя		П002		
Код товара	Наименование товара	Месяц поставки	Мин. партия поставки	Количество поставки
T001	Монитор 17LG	1	10	100
T001	Монитор 17LG	2	5	20
T001	Монитор 17LG	3	5	30
T002	FDD 3,5	1	10	50
T002	FDD 3,5	3	5	10

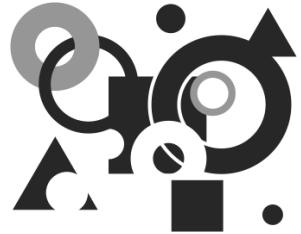
Вводятся только те реквизиты, которые обведены.

ДОГОВОР N		Д22	От	25.02.2007
ПОКУПАТЕЛЬ		Компьютер маркет		
Код покупателя		П001		
Код товара	Наименование товара	Месяц поставки	Мин. партия поставки	Количество поставки
T001	Монитор 17LG	1	10	100
T001	Монитор 17LG	2	4	12
T003	HDD Maxtor20GB	2	5	10
T004	Корпус MiniTower	3	15	30

ДОГОВОР N		Д33	От	11.01.2007
ПОКУПАТЕЛЬ		Перспектива		
Код покупателя		П002		
Код товара	Наименование товара	Месяц поставки	Мин. партия поставки	Количество поставки
T005	CD-ROM Panasonic IDE	1	10	40

# Оперативно-учетная информация

Накладная N	<b>0001</b>	от	<b>09.03.2007</b>
Номер склада	<b>01</b>		
Номер договора	<b>Д111</b>		
Код товара	Наименование товара	Количество	
T001	Монитор 17LG	10	
T002	FDD 3,5	40	
Накладная N	<b>0002</b>	от	<b>11.03.2007</b>
Номер склада	<b>02</b>		
Номер договора	<b>Д111</b>		
Код товара	Наименование товара	Количество	
T001	Монитор 17LG	10	
T002	FDD 3,5	230	
Накладная N	<b>0003</b>	от	<b>04.02.2007</b>
Номер склада	<b>03</b>		
Номер договора	<b>Д111</b>		
Код товара	Наименование товара	Количество	
T002	FDD 3,5	12	
Накладная N	<b>0004</b>	от	<b>25.03.2007</b>
Номер склада	<b>01</b>		
Номер договора	<b>Д222</b>		
Код товара	Наименование товара	Количество	
T004	Корпус MiniTower	4	
T001	Монитор 17LG	5	
Накладная N	<b>0005</b>	от	<b>13.05.2007</b>
Номер склада	<b>02</b>		
Номер договора	<b>Д222</b>		
Код товара	Наименование товара	Количество	
T001	Монитор 17LG	10	
T003	HDD Maxtor20GB	8	



## ПРИЛОЖЕНИЕ 3

### Описание компакт-диска

В компакт-диск включены:

- ❖ Дополнительные материалы:
  - ◆ Программирование на языке Visual Basic для приложений.pdf
  - ◆ Работа Access с данными на SQL-сервере.pdf
  - ◆ Разработка проекта Access - приложения MS SQL Server.pdf
- ❖ файлы с базами данных:
  - ◆ Поставка товаров.accdb
  - ◆ Учебный процесс.accdb

В файле Программирование на языке Visual Basic для приложений.pdf изложены возможности встроенного в Access объектно-ориентированного языка программирования Visual Basic for Applications. На примерах показано использование языка VBA в среде Access для автоматизации выполнения задач, организации связи между различными объектами приложения пользователя, реализованного в базе данных *Поставка товаров*. Подробно рассмотрена разработка процедур обработки событий. Представлены процедуры для проведения расчетов в связанных записях, поиска данных, доступа к базам данных SQL Server, процедуры создания формы и таблицы. Все процедуры вы сможете найти в соответствующей базе данных на компакт-диске.

В файле Работа Access с данными на SQL-сервере.pdf рассмотрена технология работы в сети с данными базы коллективного использования, размещенной на Microsoft SQL Server через интерфейс ODBC. Рассмотрено определение внешних источников данных ODBC и различные способы использования данных: запросы на языке SQL к базам данных SQL Server; связь с таблицами баз данных SQL Server; импорт объектов из базы данных SQL-сервера в Access и экспорт объектов Access в базу данных сервера.

Для выполнения примеров по импорту объектов базы данных *Поставка товаров* и доступа к данным сервера через ODBC необходимо иметь доступ к Microsoft SQL Server. Возможна установка на вашем локальном компьютере свободно распространяемого, доступного в Интернете выпуска Microsoft SQL Server Express, который может устанавливаться не только на компьютере с серверными операционными системами Windows, но и с локальными ОС, такими как Windows XP Professional Edition. Этот сервер позволит выполнить приведенные примеры по досту-

пу к данным сервера через ODBC на локальном компьютере, не подключаясь к базе данных SQL Server на удаленном сервере.

В файле Разработка проекта Access - приложения MS SQL Server.pdf рассмотрена разработка клиентского проекта Access, подключаемого к серверу баз данных через OLE DB. Проект Access, реализуя архитектуру "клиент-сервер", функционирует в среде Access и работает только с данными базы на SQL-сервере. Воспользовавшись приведенным в файле описанием, вы сможете создать проект и получить доступ к существующей на сервере базе данных или создать на сервере новую базу и ее таблицы, схемы данных. Импортировав на сервер объекты базы данных *Поставка товаров*, можно выполнить примеры по разработке в проекте Access логики приложения на стороне сервера: представлений, хранимых процедур и пользовательских функций.

В базе данных *Поставка товаров* представлена соответствующая предметная область и ее задачи. Все возможности Access 2010, рассмотренные в самоучителе, реализуются в объектах предлагаемой базы данных и служат их иллюстрацией.

База данных включает таблицы, структура которых соответствует объектам предметной области. В них введены данные контрольного примера. Созданная схема данных представляет все связи таблиц и задает проверку связной целостности.

Примеры запросов демонстрируют использование основных элементов выборки, способы объединения записей таблиц, изменения данных. Показано использование запросов для решения задач предметной области.

Созданные формы документов предметной области дают представление о возможностях конструирования, просмотра, ввода и корректировки данных документов в интерактивном режиме.

Отчеты представляют возможности Access по разработке выходных документов, их просмотру и печати. Кроме того, отчеты позволяют простыми средствами фильтрации анализировать данные отчета в различных разрезах.

Примеры макросов демонстрируют новые возможности Access 2010. Макросы данных связываются с событиями в базовой таблице, выполняются только при изменении, вставке или удалении записи и позволяют проверять данные или выполнять вычисления. Привязывая логику к данным и сосредотачивая ее в исходных таблицах, макросы данных обеспечивают доступ к ней из различных объектов приложения.

База содержит примеры интерфейса приложения на основе форм. Здесь же даны примеры форм навигации, которые позволяют простыми средствами представить функционал приложения.

Воспользуйтесь базой данных для выполнения заданий, приведенных в книге.

База данных из сферы учебного процесса содержит примеры объектов всех типов и может быть полезна для широкого круга читателей, далеких от особенностей предметной области основного примера. На предметной области, хорошо знакомой практически всем читателям, можно познакомиться с возможностью формирования документов, сопровождающих учебный процесс, например экзаменационной ведомости.

# Предметный указатель

## A

AutoExec 393

## D

Drag and Drop 45

## O

ODBC 28

OLE 24, 114

Object 96, 97, 99, 124

OLE DB 28

## S

SharePoint Server 3, 29

SQL 28, 149

инструкция TRANSFORM 400

SQL Server 27

SQL Server Express 28

Structured Query Language 6

## U

URL-адрес сервера 30

## W

Web-базы данных 3, 29

Windows NT:

Server 27

Workstation 27

---

## A

Автозамена имен объектов 22

Автоотчет 335

## Б

База данных:

загрузка 246

зашифровать паролем 53

логическая структура 55, 57

пример 79

многопользовательская 26

монопольный режим 52

параметры автозамены имен 22

преобразование в формат

    SQL Server 29

реляционная 10, 55

    структура 85

создание 56, 85

технология загрузки 248

типовая 18

формат ACCDE 18

формат ACCDR 18

формат LACCDB 17

этапы проектирования 56

Бланк запроса 172

Блокировка:

записей 244

на уровне записи 27

на уровне страниц 27

## В

Вкладки документов 35, 43

Вкладки ленты 211

    Работа с макетами форм |

        Упорядочить 211, 216

    Работа с макетами форм |

        Формат 211

Включить содержимое 395

Вложенный макрос 358

Внедренный макрос 350, 351, 352, 362,

    372, 392, 398

Внешний ключ 11  
Временная переменная 392, 398  
Всплывающее окно 392  
Вычисления:  
    в форме 222

## Г

Главная таблица 11  
Группировка 309  
Групповые операции 160

## Д

Диалоговое окно 34, 36  
Доступные шаблоны 18, 86

## З

Запись 7, 11  
    добавление 123  
    изменение значений 123  
    удаление 123  
Запросы 14, 149  
    вычисляемое поле 155  
    групповые операции 161  
        условия отбора 164  
    конструирование 161  
    многотабличные 170, 171  
    на выборку 149  
    на добавление 196  
    на обновление 150, 192  
    на создание таблицы 191  
    на удаление 201  
    однотабличные 161  
    параметры 159  
    режим SQL 178, 181, 186, 194,  
        199, 201, 203  
    режим таблицы 153, 194  
    условия отбора 164, 176  
Защита баз данных 48  
    надежное расположение 49  
    небезопасные компоненты 49  
    Панель сообщений 51  
    Центр управления безопасностью 49

## И

Изолированный макрос 350  
Именованные макросы данных 378

Инструкции SQL 166, 178, 181, 186  
DELETE 203  
INSERT INTO 199, 201  
UPDATE 194  
Информационно-логическая модель 55, 57  
    пример 78  
    разработка 56  
Информационный объект 57, 58  
    выделение 60, 62  
        пример 64  
    главный 75  
    ключ 58  
    объект-связка 76  
    подчиненный 75  
    связи 74  
    экземпляр 58  
Итоги 310

## К

Каскадное обновление связанных  
    записей 145  
Каскадное удаление связанных  
    записей 145  
Каталог макрокоманд 352  
Ключ:  
    внешний 85  
    первичный 100  
    связи 11, 80  
Кнопка:  
    Открыть 122  
    Свойства 131  
Коллекция 34, 36  
Команда:  
    Подтаблица 130, 148  
Конструктор 21  
    запросов:  
        панель инструментов 152  
        форм 256  
Контекстное меню 38  
Корпоративная сеть 25

## Л

Лента 34, 35, 89  
Локальная сеть 26  
    Microsoft Windows Server 27  
    NetWare Novell 27  
    одноранговая 26  
    с файловым сервером 27

**M**

Макет таблицы 121  
 Макет элементов управления 213  
 Макрокоманда 350  
     ЗадатьЗначение 388  
     ЗапускМакроса 374  
     НаЗапись 388  
     ОткрытьЗапрос 387, 388  
     Сообщение 388  
 Макросы 350  
     AutoExec 370  
     блок Если 364  
     блок-схема выполнения 365  
     внедренный макрос 350  
     выполнение 369  
     данных 376  
     журнал регистрации ошибок 386  
     запуск 370  
         нажатием кнопки 375  
         через события 369, 371  
     запуск:  
         автоматический 370  
     изолированный макрос 350  
     создание 351  
     условные выражения 368  
 Маска ввода 119  
     мастер 119  
 Мастер:  
     диаграмм 20  
     по анализу таблиц 21  
     подстановок 134, 137  
     подстановок 20  
     сводных таблиц 21  
     форм 210, 227, 253  
 Мастера Access 19  
 Мини-панель инструментов 35, 45  
 Модальное окно 392  
 Модель данных 6  
 Модули 15

**H**

Набор вкладок 400  
 Надежность базы данных 395  
 Нормализация данных 10, 57  
     требования 59

**O**

Область документов 93  
 Область навигации 34, 40, 91  
     новая категория 92

**Объект OLE:**

Внедренный 114  
     пример 124  
     связанный 114

**Объекты Access 13**

Оглавление справочной системы 47

Окно:  
     конструктора запросов 151  
     макросов 352

**Окно Access 89**

Отчеты 15, 303  
     вычисляемое поле 329  
     группировка 309, 337  
     добавление даты 341  
     источник записей 320, 324, 328  
     итоги 310  
         без подробностей 313  
         мастер отчетов 323  
     однотабличные 306  
     параметры 331  
     печать 317

Представление отчета 316

подчиненные 336

просмотр 313, 315

разделы 304

режимы 306

Предварительный просмотр 306, 315,  
     331, 343

Представление отчета 306, 313

режим конструктора 306, 329

режим макета 306

свойства 347

сортировка 312

составные 333

главный отчет 335

подчиненный отчет 336

сумма с накоплением 344

фильтрация 274, 313, 332

**П**

Панель быстрого доступа 34, 90

Панель инструментов:

конструктор запросов 152

таблица в режиме таблицы 120

быстрого доступа 38

Панель сообщений 35, 44, 51, 393

Панель элементов 21

Параметры Access 49, 52

- автозамена имен 22  
Блокировка на уровне записи 27  
Настройка 39  
Параметры окна документа 43  
Центр управления безопасностью 49
- Параметры:  
запроса 159  
клиента 32, 33  
навигации 92
- Первичный ключ 11, 110  
составной 113
- Подчиненная таблица 11
- Поле 11  
Memo 115  
гиперссылка 116  
Дата/время 118  
защита от изменений 243  
имя 95  
Мастер подстановок 134, 137  
общие свойства 109  
подпись 99  
размер 99  
формат 98  
общие свойства 97, 107  
формат 98  
число десятичных знаков 98
- объекта OLE:  
вставка 124, 147  
пример 114  
определение 95, 106  
свойства 107  
со списком 99, 261  
сообщение об ошибке 99  
тип данных 95, 138  
условие на значение 99
- Поле со списком 133
- Построитель выражений 157, 358  
условие на значение 110
- Предметная область 55  
документы 64, 66  
пример описания 64
- Представление Backstage 34, 35, 86, 90
- Предупреждение системы  
безопасности 44
- Проект Access 28
- P**
- Рабочая станция 26  
Расширенная обработка событий 369  
Режим ввода записи 123
- Режим макета 211  
Реквизиты:  
зависимые 59, 82  
ключевые 58  
описательные 58  
транзитивная зависимость 60, 63  
функциональная зависимость:  
пример 67, 68, 70, 72, 73, 74
- Реляционная модель 6
- Реляционная таблица 10
- C**
- Сводная диаграмма 15, 297  
Сводная таблица 15, 284  
вычисление итоговых значений 287  
вычисляемые поля 293  
работа с датами 291  
форматирование элементов 293
- Свойства поля 137
- Связи информационных объектов 74  
много-многозначные 76  
примеры 77  
одно-многозначные 75  
примеры 77, 80  
одно-однозначные 75  
примеры 77
- Связная целостность 13
- Связь и внедрение объектов (OLE) 24, 114
- Связь таблиц 11, 140
- Сервер баз данных SQL 27
- Система управления базой данных (СУБД) 6
- Службы Access 3, 29
- Смарт-теги 45
- Создание диаграммы 399, 402
- Сохранить и опубликовать 30
- Список 99
- Справочная система Access 46 ,47
- Ссылка:  
на подчиненную форму/отчет 357  
на свойство 356  
на элемент управления 355  
создание построителем 358
- Строка состояния 35, 43
- Структура записи 7
- Структура реляционной таблицы 11
- Схема данных 13, 140  
определение связей 141, 142  
по составному ключу 143  
пример 81, 145  
создание 141

**T**

Таблицы БД 14:  
 ввод данных 119  
 логически связанных 128  
 пример 122  
 индексы 100  
 макет 121  
 режим конструктора 106  
 режим таблицы 111  
 создание 95  
 пример 108  
 структура 95, 106  
 пример 108  
 структура 85

Тип данных:  
 Мастер подстановок 137

**У**

Уникальный ключ 11  
 определение 100  
 простой 59  
 составной 59

**Ф**

Файл базы данных 17  
 имя 87  
 рабочий каталог 87  
 создание 87  
 Файловый сервер 26  
 Фильтр 123  
 Форма:  
 вычисления 222  
 навигации 407  
 одиночная многотабличная 233

Формат даты 118

Формы 14  
 Автоформат 212

вкладки 401  
 выборка документа 270  
 вычисление итогов 224

Добавить поле 258  
 добавление полей 217  
 заголовок 212  
 загрузка БД 247  
 защита данных 243

источник записей 257  
 кнопки управления 375  
 конструирование 236, 256  
 конструирование поля со списком 266  
 макет 247  
 мастер 253

поле со списком 262  
 мастер форм 227  
 многотабличные 225  
 подсхема данных 250  
 создание 253  
 на основе запроса 219  
 Несколько элементов 209, 219  
 однотабличные 208  
 панель навигации 406  
 параметры 219  
 подготовка к созданию 246  
 подчиненная 237, 239

вычисления 241  
 подчиненные 225, 255  
 защита данных 245  
 связь 241  
 поле со списком 261  
 Пустая форма 233  
 работа с документом 268  
 Разделенная форма 209, 220, 222  
 режим формы 217  
 режимы 208  
 свойства 216, 220  
 создание кнопок 259  
 Условное форматирование 214  
 элемент управления 217  
 Поле со списком 267  
 свойства 207  
 эмблема 212

**Ц**

Целостность данных 57, 143  
 задание параметров 144  
 проверка 146  
 условия установления 144

**Э**

Элемент управления:  
 Вставить диаграмму 20  
 свойства 375