Министерство образования
Республики Беларусь

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Экономика и организация машиностроительного производства»

О.А. Лавренова

СЕТЕВЫЕ ТЕХНОЛОГИИ И БАЗЫ ДАННЫХ

Курс лекций

Минск 2009

Министерство образования Республики Беларусь
БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Экономика и организация машиностроительного производства»

О.А. Лавренова

СЕТЕВЫЕ ТЕХНОЛОГИИ И БАЗЫ ДАННЫХ

Курс лекций
для студентов специальности 1-27 01 01
«Экономика и организация производства»

В 2 частях

Часть 1

ОСНОВЫ РАБОТЫ С РЕЛЯЦИОННЫМИ БАЗАМИ ДАННЫХ И СУБД

Минск 2009

УДК [004.65+004.7] (075.8)
ББК 34.97я7
Л 13

Р е ц е н з е н т ы :
А.Л. Ивашутин, А.В. Плясунков

Лавренова, О.А.

Л 13 Сетевые технологии и базы данных: курс лекций для студентов специальности 1-27 01 01 «Экономика и организация производства»: в 2 ч. / О.А. Лавренова. – Минск: БНТУ, 2009. – Ч. 1: Основы работы с реляционными базами данных и СУБД. – 110 с.

ISBN 978-985-525-058-7 (Ч.1).

В издании изложены основные сведения в области теории баз данных, рассмотрены приемы работы с реляционной СУБД Microsoft Access? Включая технологии создания таблиц и формирования схемы базы даны, способы анализа данных при помощи запросов, приемы организации интерфейса пользователя и настройки рабочей среды.

Приведен список учебной литературы, необходимой при изучении дисциплины.

Издание предназначено для студентов дневной и заочной форм обучения специальности 1-27 01 01 «Экономика и организация производства».

УДК [004.65+004.7] (075.8)
ББК 34.97я7

ISBN 978-985-525-058-7 (Ч. 1)
ISBN 978-985-525-059-4

Содержание

Лекция 1. ВВЕДЕНИЕ В ТЕОРИЮ БАЗ ДАННЫХ

1. Понятие и классификация информационных технологий.
2. Основные понятия теории баз данных.
3. Основные структурные единицы реляционных баз данных.
4. Этапы проектирования реляционных баз данных.

1.1. Понятие и классификация информационных технологий

Под **информационной технологией** (ИТ) понимают систему методов и способов сбора, накопления, хранения, поиска и обработки информации на основе вычислительной техники.

К современным ИТ относят компьютерные и телекоммуникационные системы, микроэлектронику.

Любая ИТ использует техническое и программное обеспечение.

Современные **компьютерные ИТ классифицируются** по нескольким признакам:

1. ***По виду информации***:

Вид информации	*Информационная технология*
Текст	Текстовый процессор
Графика	Графический процессор
Данные	Табличные процессоры; СУБД
Знания	Экспертные системы
Объекты реального мира	Системы мультимедиа

Алгоритмические языки программирования могут применяться для обработки любых типов информации.

2. ***По возможностям ИТ:***

a) обеспечивающие (языки программирования, электронные таблицы);

b) функциональные (конкретные приложения пользователя).

3. ***По типу пользовательского интерфейса:***

a) командный интерфейс;
b) WIMP (window-image-menu-pointer);
c) SILK (speech-image-language-knowledge);
d) общественный интерфейс (b+c).

4. ***По степени участия пользователя:***

a) пакетные ИТ (участие человека не требуется);
b) диалоговые ИТ (требуется участие человека).

5. ***По степени взаимодействия технологий:***

a) локальные;
b) сетевые.

6. ***По концепции обработки и хранения данных:***

a) с распределенной информационной базой (данные хранятся на различных компьютерах);

b) ИТ с распределенной обработкой данных (работа с данными осуществляется с разных компьютеров, а управление БД – с одного).

Основные идеи современных ИТ базируются на концепции, согласно которой данные должны быть организованы в базы данных с целью адекватного отображения изменяющегося реального мира и удовлетворения информационных потребностей пользователей. Эти базы данных создаются и функционируют под управлением специальных программных комплексов, называемых ***системами управления базами***

данных (СУБД): Oracle, MS SQL Server, MySQL, Informix, DB2, MS FoxPro, MS Access и др.

Увеличение объема и структурной сложности хранимых данных, расширение круга пользователей информационных систем привели к широкому распространению наиболее удобных и сравнительно простых для понимания реляционных (табличных) СУБД.

1.2. Основные понятия теории баз данных

База данных (БД) – структурированная совокупность логически взаимосвязанных данных конкретной предметной области, организованная на магнитных носителях средствами СУБД.

СУБД – программное средство, предназначенное для создания и обслуживания БД на внешних запоминающих устройствах (ВЗУ).

Любая СУБД поддерживает минимальный набор функций:

Основные функции	***Дополнительные функции***
1. Ввод данных 2. Обновление данных 3. Анализ данных	1. Проверка состояния БД 2. Выдача справочной информации 3. Разграничение прав доступа пользователей к информации

Для выполнения этих функций в состав СУБД входят **язык описания данных**, позволяющий создать структуру описания данных в базе, и **язык манипулирования данными**, позволяющий производить различные операции с данными.

Работу БД обслуживает администрация: выполняет подключение к системе новых пользователей, определяет

нормы и правила доступа к данным, создает копии данных, проверяет и восстанавливает информацию.

В общем случае при работе с БД происходит преобразование данных из внешнего представления во внутреннее в соответствии с логической структурой БД.

Описание общей логической структуры БД называется **схемой БД.** Схема данных отображает информационно-логическую модель предметной области. В схему БД входит полное описание всех типов данных, хранящихся в базе, а также всех типов операций над ними. Схема БД базируется на **модели данных,** которая является ядром любой базы данных.

Модель данных представляет собой множество структур данных, ограничений целостности и операций манипулирования данными. С помощью модели данных могут быть представлены объекты предметной области и взаимосвязи между ними.

Существуют следующие модели данных:

1. **Иерархические** – данные представлены в виде деревьев. Вершины – информационные единицы, дуги – связи. Каждый объект может подчиняться только одному объекту более высокого уровня. Существует единая точка входа (рис. 1.1).

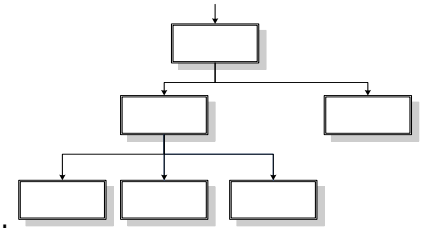

Рис. 1.1. Иерархическая модель данных

2. **Сетевые** – данные связаны системой отношений в виде произвольной сети. Любой объект может быть связан с любым количеством других элементов. Существует несколько точек доступа (рис. 1.2).

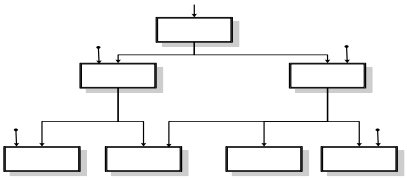

Рис.1.2. Сетевая модель данных

Существенными недостатками как иерархической, так и сетевой моделей являются дублирование данных и неоптимальный доступ к ним.

3. **Реляционные** – данные сгруппированы в двумерные таблицы-отношения, между которыми установлены связи по ключам (рис. 1.3).

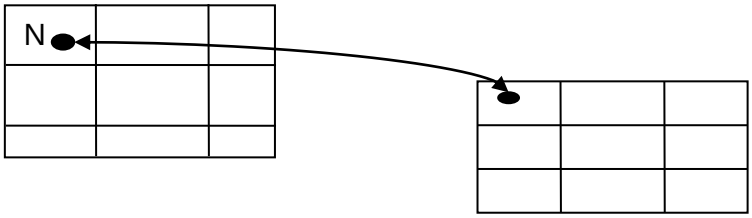

Рис. 1.3. Реляционная модель данных

Впервые термин «реляционная модель данных» появился в статье сотрудника фирмы IBM д-ра Кодда (Codd E.F., A Relational Model of Data for Large Shared Data Banks. CACM 13: 6, June 1970). Кодд (математик по образованию) предложил использовать для обработки данных аппарат теории множеств (объединение, пересечение, разность, декартово произведение). Он показал, что любое представление данных сводится к совокупности двумерных таблиц особого вида, известного в математике как отношение (relation, англ.).

Структуру двумерной таблицы образуют столбцы и строки. Их аналогами в простейшей базе данных являются поля и

записи. Если записей в таблице пока нет, значит, ее структура образована только набором полей.

Реляционной является БД, в которой все данные, доступные пользователю, организованы в виде набора двумерных таблиц, а все операции над данными сводятся к операциям над этими таблицами.

При работе с реляционными моделями используется как математическая терминология, так и терминология, исторически принятая в сфере обработки данных. Основные формальные реляционные термины и соответствующие им неформальные эквиваленты приведены ниже:

Формальный реляционный термин	**Неформальный эквивалент**
Отношение	Таблица
Кортеж	Запись, строка
Атрибут	Поле, столбец

Таблицы-отношения реляционной модели обладают следующими свойствами:

- каждый столбец таблицы соответствует элементу (атрибуту) данных. Повторяющиеся атрибуты отсутствуют;
- в каждой строке таблицы содержится по одному значению в соответствующем столбце;
- все столбцы (поля) таблицы – однородные;
- столбцам присвоены однозначные имена, определяющие атрибуты;
- один или несколько атрибутов являются ключом таблицы, который однозначно идентифицирует запись таблицы;
- в таблице не может быть двух одинаковых строк (записей);

- общее количество строк (записей) не ограничено.

Важным требованием, предъявляемым к таблицам реляционной модели, является **нормализация данных** – минимизация количества повторяющихся данных.

Существует несколько нормальных форм реляционной модели, которые вводят ограничения и позволяют минимизировать дублирование данных, обеспечить целостность данных.

Первично нормализованная таблица содержит простые атрибуты (одному атрибуту соответствует одно значение).

При второй нормальной форме атрибуты являются простыми и каждый описательный (неключевой) атрибут функционально полно зависит от ключа (одному значению ключа соответствует одно значение неключевого атрибута).

При третьей нормальной форме реляционной модели (канонической модели) выполняются следующие **требования нормализации**:

1) каждая таблица должна содержать уникальный ключ (простой или составной);

2) все описательные (неключевые) атрибуты должны быть взаимно независимы;

3) все атрибуты составного ключа должны быть взаимно независимы;

4) каждый описательный атрибут должен функционально полно зависеть от ключа (каждому ключу соответствует одно значение описательного атрибута);

5) при составном ключе каждый описательный атрибут должен функционально полно зависеть от всей совокупности атрибутов ключа (не допускается зависимость описательного атрибута от части ключа);

6) каждый описательный атрибут не может зависеть от ключа транзитивно, т.е. через другой промежуточный атрибут. В случае транзитивной зависимости совокупность атрибутов разделяется с образованием двух таблиц.

1.3. Основные структурные единицы реляционных БД

Основными типами структур данных являются:

1. **Таблица** – множество одинаковых по структуре записей со значениями в полях.

2. **Запись** – совокупность полей, соответствующая логически связанным реквизитам. **Структура записи** определяется составом и последовательностью полей, входящих в нее. Каждая запись однозначно идентифицируется ключом.

3. **Поле** – элементарная единица логической организации данных, которая соответствует неделимой единице информации – реквизиту. ***Реквизит*** – простейшая единица информации, отображающая качественную или количественную характеристику объекта предметной области.

Поля могут быть (рис. 1.4) **ключевыми** и **описательными** (информационными).

Ключевое поле (ключ) – одно или несколько полей, значения которых однозначно определяют запись в таблице.

Ключевые поля бывают **первичные и внешние**:

1. **Первичные** – одно или несколько полей главной таблицы однозначно идентифицирующих запись (одно поле – простой ключ, иначе – составной). Значения первичного ключа **уникальны**.

2. **Внешние** – одно или несколько полей в таблице-связке, содержащих ссылку на ключевое поле или поля в главной таблице. Значения внешнего ключа **могут повторяться** в нескольких записях таблицы-связки.

При описании логической структуры данных указывают структуру записи, т.е. поля и их порядок, также для каждого поля задаются **имя**, **тип данных** (формат и точность данных) и для ключевых полей – **признак ключа**.

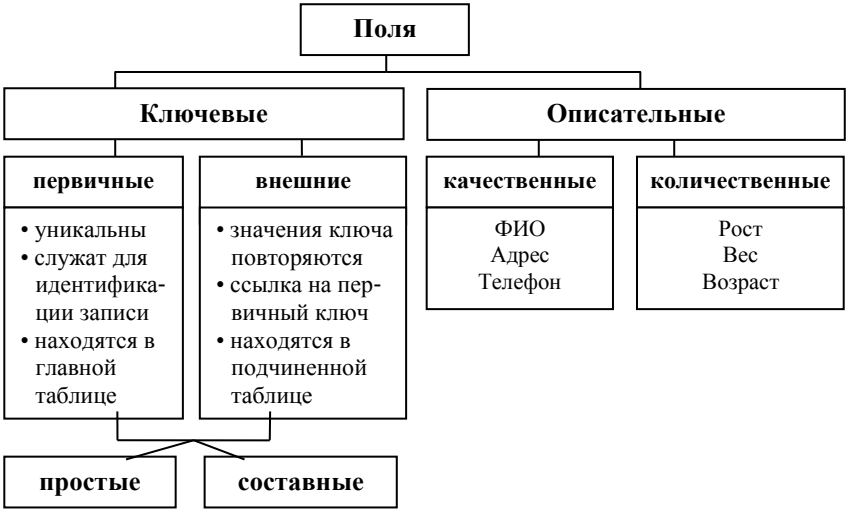

Рис. 1.4. Классификация полей

1.4. Этапы проектирования реляционных БД

Проектирование БД является очень важным этапом, от которого зависят последующие этапы ее реализации средствами СУБД.

Перед разработкой базы данных необходимо располагать описанием выбранной предметной области, иметь всю необходимую информацию для удовлетворения предполагаемых запросов пользователя и определить потребности в обработке данных. Процесс работы над проектом БД представлен на рис. 1.5.

Укрупненно процесс разработки реляционной базы данных состоит из следующих четырех этапов (рис. 1.6):

1 этап. Определение цели создания базы данных.

2 этап. Выделение информационных объектов предметной области.

3 этап. Определение логической структуры БД.

4 этап. Создание и заполнение объектов базы данных средствами конкретной СУБД.

Рис. 1.5. Процесс работы над базой данных

На первом этапе проектирования базы данных необходимо определить:

- цель создания базы данных,
- основные ее функции,
- информацию, которую она должна содержать.

Фактически определяют основные темы таблиц базы данных и информацию, которую будут содержать таблицы.

База данных должна отвечать требованиям конечных пользователей. Поэтому необходимо четко определить темы, которые должна охватывать БД, отчеты, которые она должна выдавать, проанализировать формы, которые в настоящий момент используются для записи данных.

На этом этапе желательно сравнить создаваемую базу данных с готовой аналогичной базой данных.

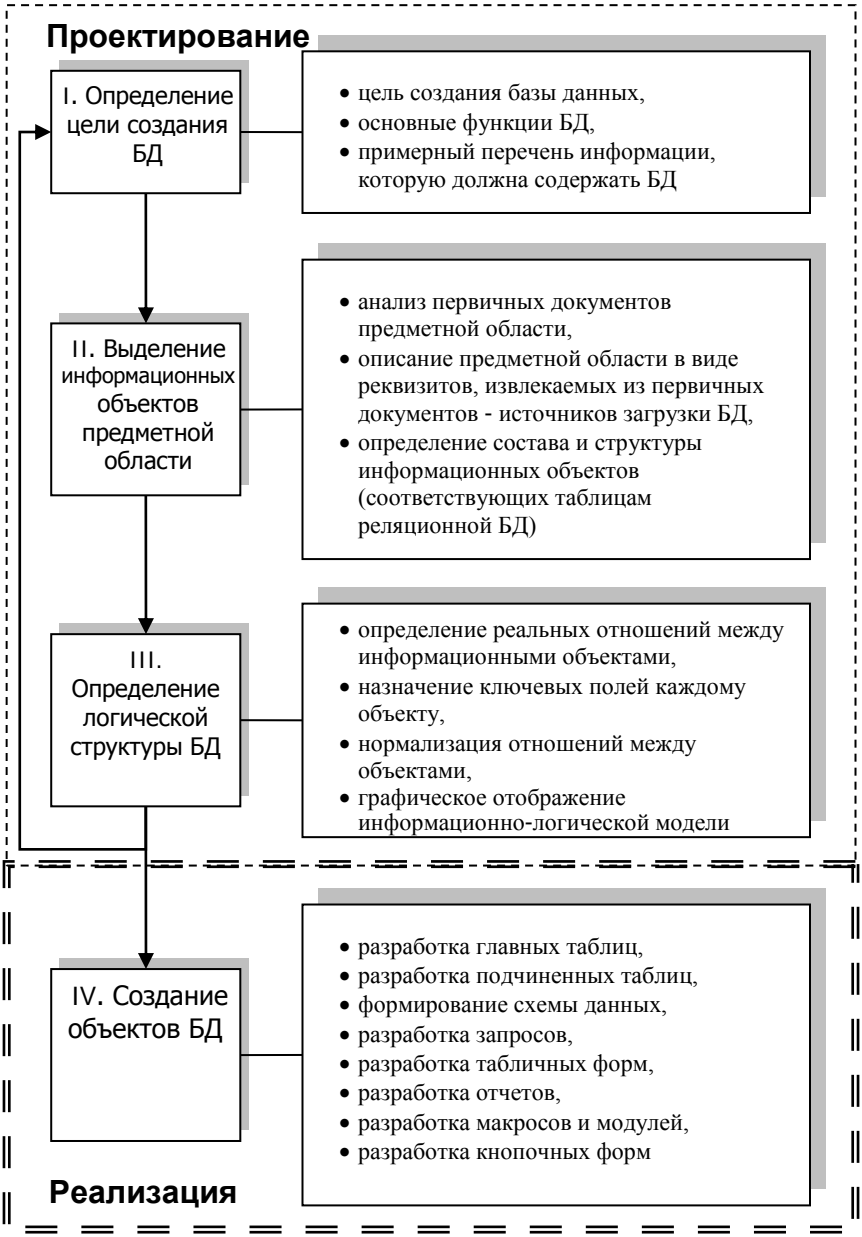

Рис. 1.6. Этапы разработки реляционной БД

На втором этапе выделяют информационные объекты предметной области.

Данный этап включает:

a) анализ предметной области, при котором определяется состав и структура данных;

b) описание предметной области в виде реквизитов, извлекаемых из первичных документов – источников загрузки БД.

В общем случае, чтобы выделить из перечня информации информационные объекты, необходимо выполнить следующие действия:

- установить функциональные зависимости между реквизитами;
- разделить все реквизиты на ключевые и описательные, установить между ними соответствие;
- образовать информационный объект, сгруппировав описательные реквизиты, одинаково зависимые от ключевого;
- определить структурные связи.

На рис. 1.7 представлен пример выделения информационных объектов для предметной области «Хозяйственная деятельность» (на основе анализа первичного документа «Договор»).

Информационными объектами предметной области «Хозяйственная деятельность» являются:

1. **Договор** – **номер договора**, дата, сумма, ***код заказчика***, ***код изделия***, количество.
2. **Заказчик** – **код заказчика**, наименование заказчика, адрес, банк.
3. **Изделие** – **код изделия**, наименование изделия, единица измерения, минимальная партия, цена.

Документ	Реквизиты	Функциональные зависимости
Договор	**Номер**	
	Дата	
	Сумма	
	Количество	
	Код заказчика	
	Наименование заказчика	
	Адрес заказчика	
	Банк	
	Минимальная партия	
	Цена	
	Наименование изделия	
	Единица измерения	
	Код изделия	

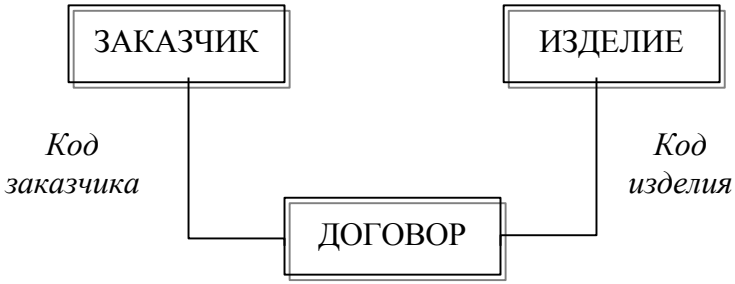

Рис. 1.7. Выделение информационных объектов

В соответствии с требованиями реляционной модели каждому информационному объекту будет соответствовать **таблица базы данных**. Проектирование таблиц является одним из наиболее сложных этапов, так как результаты, которые должна выдавать база данных (отчеты, выходные формы и др.) не всегда дают полное представление о структуре таблицы.

При проектировании таблиц не используется СУБД, структура разрабатывается на бумаге. При этом рекомендуется соблюдать следующие **требования к содержанию таблиц реляционной БД**:

1. **Информация в таблице не должна дублироваться. Не должно быть повторений и между таблицами.**

Когда определенная информация хранится только в одной таблице, то и изменять ее придется только в одном месте. Это делает работу более эффективной, а также исключает возможность несовпадения информации в разных таблицах. Например, в одной таблице должны содержаться адреса и телефоны клиентов.

2. **Каждая таблица должна содержать информацию только на одну тему.**

Сведения на каждую тему обрабатываются намного легче, если они содержатся в независимых друг от друга таблицах. Например, адреса и заказы клиентов хранятся в разных таблицах, чтобы при удалении заказа информация о клиенте осталась в базе данных.

3. **Каждая таблица должна содержать первичный ключ**.

Для того чтобы связать данные из разных таблиц, например, данные о клиенте и его заказы, каждая таблица должна содержать поле или набор полей, которые будут задавать индивидуальное значение каждой записи в таблице. Такое поле или набор полей называют **первичным ключом**.

4. **В таблице должна присутствовать вся необходимая информация.**

Информацию в таблицы БД рекомендуется **закладывать с избытком**. Каждая таблица содержит информацию на отдельную тему, а каждое поле в таблице содержит отдельные сведения по теме таблицы. Например, в таблице с данными о клиенте могут содержаться поля с названием компании, адресом, городом, страной и номером телефона.

5. **Каждое поле должно быть связано с темой таблицы.**

6. **Не рекомендуется включать в таблицу данные, которые являются результатом вычисления.**

7. **Информацию следует разбивать на наименьшие логические единицы** (например, поля «Фамилия», «Имя» и «Отчество», а не общее поле «ФИО»).

На третьем этапе определяют логическую структуру БД. Фактически определяют тип связей между объектами (таблицами), учитывая, что реальное отношение определяется отношением между экземплярами двух типов информационных объектов.

Отношения между объектами могут быть следующих типов:

1. **Одно-однозначные (1 : 1)** – имеет место, когда каждой записи одной таблицы соответствует одна запись другой. Такие таблицы могут быть легко объединены в одну, и первичным ключом может стать любой ключ из таблицы. Объекты равноправные.

2. **Одно-многозначные (1 : ∞)** – каждой записи одной таблицы может соответствовать несколько записей другой. Имеют место иерархические групповые отношения. Один объект определяется как главный, а другой – как подчиненный.

3. **Много-многозначные (∞ : ∞)** – каждой записи одной таблицы может соответствовать несколько записей другой и

наоборот. Это сетевые групповые отношения. Непосредственно не могут поддерживаться в реляционных СУБД. Обычно реализуются через третий объект, с которым исходные объекты связаны одно-многозначными отношениями, который вводится дополнительно и называется **объектом-связкой**.

Чаще всего реальные отношения между информационными объектами являются отношениями «многие-ко-многим», которые непосредственно не поддерживаются реляционными СУБД. Поэтому реальные отношения «многие-ко-многим» трансформируются в отношения «один-ко-многим» путем ввода объекта-связки.

На этапе определения связей между объектами применяют следующие ***практические приемы нормализации*** отношений:

1. Необходимо проверить, что между каждым полем и записью в таблице существует связь 1:1.
2. Если между таблицей и некоторыми данными имеется связь **1:**∞, следует поместить эти данные в другую таблицу, которая содержит первичный ключ первой таблицы в качестве внешнего ключа.
3. Связь ∞**:**∞ между двумя таблицами разбивают на две связи **1:**∞ и создают третью таблицу, которая содержит первичные ключи обеих таблиц в качестве внешних.

После проектирования таблиц, полей и связей необходимо еще раз проанализировать структуру базы данных и выявить возможные недочеты. Желательно сделать это на этапе формирования информационно-логической схемы БД.

На четвертом этапе средствами конкретной СУБД осуществляют создание и заполнение объектов базы данных.

Первоначально необходимо сформировать макеты всех таблиц, установить связи между ними (создать схему БД).

Необходимо исключить из таблиц все возможные повторения данных. Если структуры таблиц отвечают поставленным требованиям, то можно вводить все данные.

Для проверки работоспособности базы данных необходимо создать несколько таблиц, определить связи между ними и ввести несколько записей в каждую таблицу, затем проанализировать, насколько база данных отвечает поставленным требованиям. Рекомендуется также создать черновые выходные формы и отчеты и проверить, выдают ли они требуемую информацию.

После этого можно создавать любые запросы, формы, отчеты, макросы и модули.

Лекция 2. РАБОТА С СУБД MICROSOFT ACCESS

1. **Основные объекты СУБД MS Access.**
2. **Способы создания БД.**
3. **Разработка макетов таблиц.**
4. **Типы данных MS Access.**
5. **Свойства полей.**
6. **Порядок формирования схемы БД.**

2.1. Основные объекты СУБД MS Access

Реляционная СУБД Microsoft Access входит в состав интегрированного пакета для офиса Microsoft Office и является одним из распространенных настольных приложений для работы с базами данных. Важным достоинством MS Access (особенно для начинающего пользователя) является стандартный графический интерфейс. Кроме того, MS Access предоставляет пользователю широкий набор средств для обработки и представления данных, обеспечивает интеграцию с другими приложениями MS Office и может использоваться для разработки уникальных решений в малых и средних БД.

В окне БД, открытой в СУБД MS Access, на панели **Объекты** отображается список доступных в СУБД объектов, а в правой части окна – список конкретных объектов открытой БД (например, на рис. 2.1 представлен список таблиц учебной БД «Борей»).

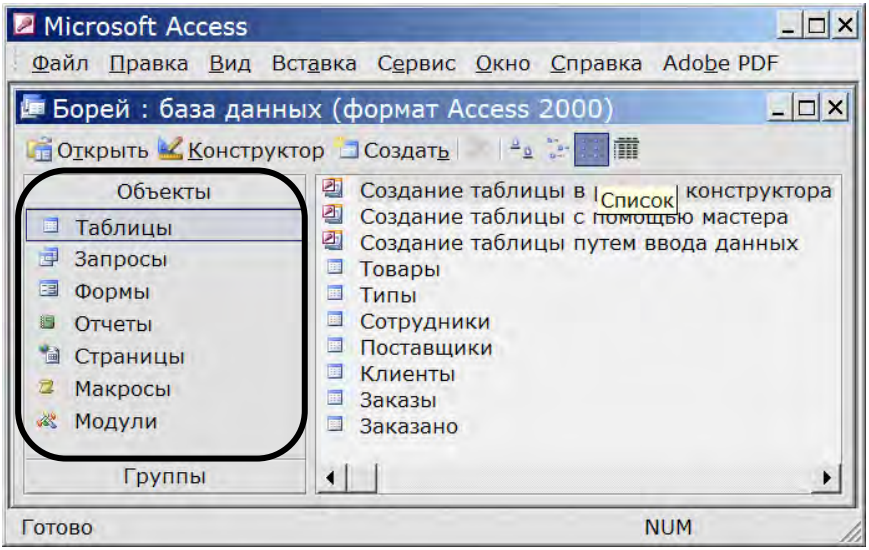

Рис. 2.1. Представление объектов БД на вкладке Таблицы

В MS Access база данных представляет собой один файл с расширением .mdb, в котором содержатся следующие объекты:

Таблицы - объекты СУБД MS Access, предназначенные для структурированного хранения данных.

Запросы - объекты СУБД MS Access, предназначенные для анализа данных из таблиц.

Формы - объекты СУБД MS Access, предназначенные для наглядного отображения, ввода и редактирования данных из таблиц (табличные формы), а также для управления объектами базы данных (кнопочные формы).

Отчеты - объекты СУБД MS Access, предназначенные для подготовки и вывода на печать итоговых документов в удобном для пользователя виде.

Страницы доступа к данным – web-страницы (самостоятельные html-документы), предназначенные для работы с данными через сеть (глобальную или локальную).

Макросы - объекты СУБД MS Access, представляющие собой набор инструкций (макрокоманд) и предназначенные для автоматизации часто повторяющихся действий пользователя.

Модули - объекты СУБД MS Access, представляющие собой законченные программы на языке VBA и предназначенные для решения прикладных задач пользователя.

С любыми объектами СУБД MS Access можно работать как минимум в двух режимах:

- в **режиме Конструктора**,
- в **режиме объекта** (режиме таблицы, формы и т.д.).

Переключение между режимами осуществляется с помощью командного **меню Вид** или **кнопки Вид** на панели инструментов.

В режиме **Конструктора** создается **макет объекта** (таблицы, запроса, формы и т.д.), а в **режиме объекта** (таблицы, формы и т.д.) – отображается соответствующий результат обработки данных (записи таблицы, запроса и т.д.).

2.2. Создание новой БД

Существует несколько способов создания новой БД в СУБД MS Access из меню **Файл ⇨ Создать…**:

1. Использовать один **из шаблонов**, которые поставляются вместе с MS Access.
2. Использовать **Мастер создания БД**.

3. Создать **пустую БД**. При создании пустой БД в самом начале работы указывают местоположение БД (**диск, папка**) и **имя файла**. По умолчанию имя файла новой БД – db1.mdb (не рекомендуется использовать имя по умолчанию).

2.3. Разработка макетов таблиц

Создание объектов БД начинается с создания таблиц. На вкладке **Таблицы** необходимо нажать кнопку **Создать** (см. рис. 2.1), после чего в диалоговом окне выбрать один из режимов создания (рис. 2.2).

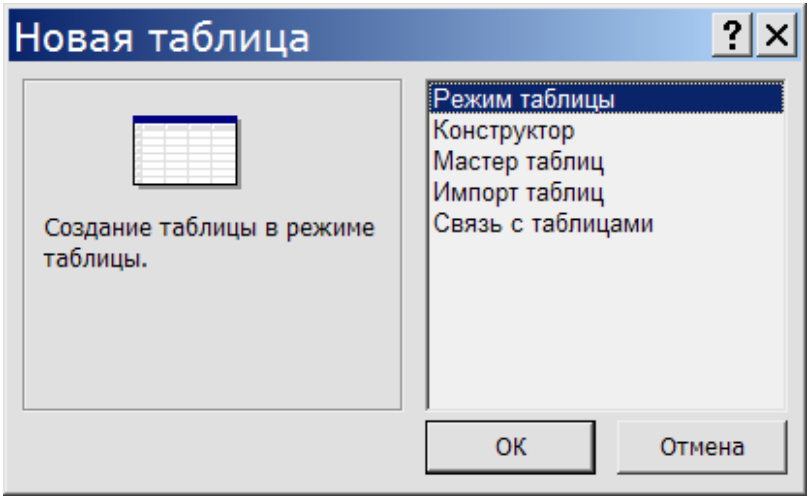

Рис. 2.2. Режимы создания таблиц БД

1. **Режим таблицы** – создает новую таблицу из 21 записи, структуру каждой из которых составляет 10 полей (Поле1, Поле2 и т.д.). При сохранении таблицы автоматически определяется тип данных в каждом поле в зависимости от введенной информации.
2. **Конструктор** – в этом режиме необходимо самостоятельно спроектировать макет таблицы: задать имена

полей, типы данных, свойства полей, указать ключевое поле. Затем сохранить таблицу и заполнить ее (в режиме Таблица).

3. **Мастер таблиц** – позволяет создать макет таблицы в пошаговом режиме, выбирая названия полей из списка.

4. **Импорт таблиц** – позволяет добавить в базу ранее созданные таблицы, причем как макет, так и данные.

5. **Связь с таблицами** – позволяет создать специальные типы таблиц, используемые для работы с данными из других приложений.

Последовательность действий по созданию таблицы в режиме Конструктора:

При выборе режима Конструктора появляется окно, содержащее три столбца: **Имя поля**, **Тип данных**, **Описание**, а также **вкладку Свойства** (рис. 2.3).

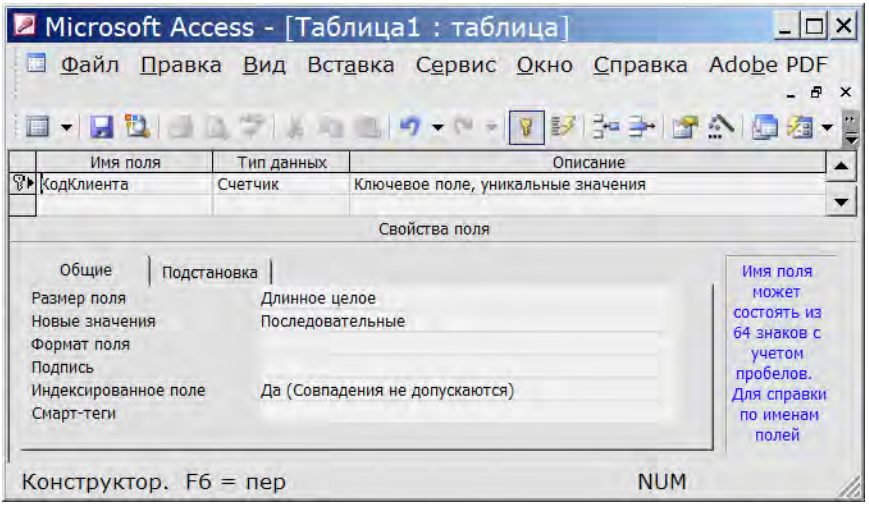

Рис. 2.3. Окно Конструктора таблицы

Обязательно необходимо заполнить **Имя поля**, **Тип данных**, поля вкладки **Свойства** и **желательно** – **Описание** (содержимое Описания будет отображаться в строке состояния при вводе данных в таблицу).

Следует учитывать правила присвоения имен (полей и таблиц):

1. Длина имени поля − до 64 символов.
2. Не допускаются символы: [ ] ! . (точка).
3. Пробелы могут быть, но не должны быть в начале имени.
4. Не могут включаться управляющие символы (с кодами ASCII 00 – 31).

При перемещении курсора в столбец **Тип данных** появляется список, из которого нужно выбрать необходимый тип в соответствии с хранимой информацией.

Кроме вышеперечисленной информации необходимо задать **свойства поля** (набор характеристик, определяющих способ ввода, отображения, обработки и сохранения данных).

В каждой таблице обязательно задать **поле первичного ключа** (для связи с другими таблицами) в меню **Правка⇨Ключевое поле**. Если ключевое поле не задано, то автоматически при сохранении таблицы в качестве ключевого будет выбрано поле с типом данных **счетчик** (если таковое присутствует в таблице) либо создано новое ключевое поле (имя поля – **Код**, тип данных – **Счетчик**).

При сохранении созданной таблицы указывается **имя таблицы** (должно соответствовать содержимому таблицы).

2.4. Типы данных MS Access

При проектировании макета таблицы для каждого поля необходимо выбрать подходящий тип данных.

Тип данных определяет набор допустимых значений в поле и операции над ними.

При выборе типа данных необходимо учитывать, что данные с меньшим размером занимают меньше дискового пространства, оперативной памяти и быстрее обрабатываются. Но слишком малые поля могут привести **к искажению или потере данных**.

В СУБД Microsoft Access определены следующие типы данных полей (табл. 2.1).

Таблица 2.1

Краткая характеристика типов данных MS Access

Тип данных	Описание, допустимые значения в поле	Размер	Индексирование	Ключ
1	2	3	4	5
Текстовый	Текст, числа, не требующие вычислений, и их комбинации. Адреса, номера телефонов, почтовые индексы	До 255 символов, по умолчанию 50	**+**	**–/+** первичный внешний
MEMO	Длинный текст или числа. Описания, характеристики, аннотации	64 Кбайт (65535 символов)	**–**	**–**
Числовой	Числа, используемые для математических вычислений (кроме денежных операций). Типы чисел и их точность зависят от свойства **Размер поля**	1,2,4,8,12 байт 16 байт (код репликации)	**+/–**	**+** первичный внешний
Денежный	Значения валют, используемые для денежных операций. Предотвращает округление при вычислениях. Допускает 15 знаков целой части числа и 4 знака в дробной части. Вид валюты зависит от выбранной национальной настройки (Панель Управления)	8 байт	**+/–**	**–**

Продолжение табл. 2.1

1	2	3	4	5
Дата/время	Значения дат и времени (от 100 до 9999 года). Хранятся как **десятичные числа** (дата– целая часть числа, время – дробная, доля суток от полуночи). Вычисления выполняются в единицах измерения времени	8 байт	**+/–**	**–**
Счетчик	**Уникальные** последовательные или случайные числа в полях **первичного ключа**, которые вводятся в поле автоматически при создании новой записи. Не могут быть изменены пользователем. В таблице может быть **только одно поле** данного типа	4 байта 16 байт (код репликации)	**+**	**+** первичный
Логический	Одно из двух значений: Истина/Ложь, Да/Нет, Включено/Выключено (On/Off). Обычно имеет вид флажков, применяется для полей Оплачено, Заказано, Доставлено	1 бит	**+/–**	**–**
Поле объекта OLE	Объекты, созданные в других приложениях, поддерживающих технологию OLE (рисунки, аудиозаписи, видеоклипы и др.). Объекты могут быть связанными или внедренными	До 1 Гбайта (ограничен объемом диска)	**–**	**–**

Окончание табл. 2.1

1	2	3	4	5
Гиперссылка	Адрес гиперссылки - путь к объекту, документу или Web-странице. Адреса образуются из текстовых значений, либо из комбинации текстовых и числовых значений (сохраняемых в текстовом формате)	До 2048 символов	–	–
	Адрес гиперссылки может состоять максимум **из 4 частей** (каждая часть не более 2048 символов) и записывается в следующем формате: [экранный_Текст]#адрес#[доп_Адрес]#[всплывающая_Подсказка] Адрес гиперссылки может представлять **адрес** URL (в Интернет, например, http://www.someone.homepage/def.html) или **сетевой маршрут UNC** (к файлу в локальной сети, например, \\MyWorkstation\Samples\Борей.mdb.). Адрес гиперссылки может содержать некоторую специальную адресную информацию (например, объект базы данных, закладку Microsoft Word или диапазон ячеек Microsoft Excel, на которые указывает адрес)			
***Мастер подстановок**	**Не является самостоятельным типом данных.** Позволяет создать **поле со списком** (столбец подстановки) для выбора значений из фиксированного набора значений, введенных пользователем, или из таблицы БД (для внешних ключей)	Такой же, как и у поля источника подстановки	+	+ внешний
	После применения Мастера подстановок для поля устанавливается **тип данных источника подстановки**. При этом на вкладке Подстановка отображаются все свойства сформированного поля со списком. В режиме таблицы в поле Мастера подстановок находится **раскрывающийся список**, позволяющий выбирать требуемые значения			

2.5. Свойства полей

Каждое поле в таблице БД имеет **свойства – набор характеристик, определяющих параметры отображения, обработки и сохранения данных.**

Свойства поля управляют поведением значений в поле.

Набор свойств поля зависит от выбранного типа данных (табл. 2.2).

Таблица 2.2

Краткая характеристика основных свойств полей

Свойство/ Тип данных	**Значение, описание**
1	2
Размер поля	Определяет количество байт для хранения данных
Текстовый	1 – 255 символов, по умолчанию – 50
Числовой	**Байт** (1 байт, целые числа от 0 до 255) **Целое** (2 байта, целые числа от - 32768 до 32767) **Длинное целое** (4 байта, целые числа от - 2 147 483 648 до 2 147 483 647) **Одинарное с плавающей точкой** (4 байта, действительные числа с точностью до 7 значащих цифр, от - $3{,}410^{38}$ до $3{,}410^{38}$) **Двойное с плавающей точкой** (8 байт, действительные числа с точностью до 15 значащих цифр, от - $1{,}79710^{308}$ до $1{,}79710^{308}$) **Код репликации** (16 байт, только при создании реплик БД для глобальных уникальных идентификаторов) **Действительное** (12 байт, действительные числа с заданной точностью, от -10^{28} до 10^{28})
Формат поля	Определяет способ отображения данных в поле, не влияет на хранение
Текстовый, MEMO	Для задания способа отображения данных в поле используются специальные форматы, состоящие из двух секций (разделитель – ;): ***текст; пустые значения*** Для создания формата используются коды: @ − обязательный символ (@@@-@@-@@ -№ тел.) & − не обязательный символ > − преобразует буквы в прописные < − преобразует буквы в строчные

Продолжение табл. 2.2

1	2
Числовой, Денежный, Счетчик	Основной Денежный Евро Фиксированный С разделителями разрядов Процентный Экспоненциальный Специальный формат, создаваемый пользователем, состоит из 4 секций (разделитель ;): ***положительные; отрицательные; ноль; пустые значения*** **Например,** # ##0,0” $”; -# ##0,0” $”; 0 – пользовательский формат, отображающий положительные и отрицательные значения с разделителями групп разрядов, с одним десятичным знаком и обозначением валюты ($), а ноль – в виде 0
Дата/время	Полный формат даты (дата + время) Длинный формат даты (из Панели управления) Средний формат даты Краткий формат даты (из Панели управления). Не рекомендуется Длинный формат времени Средний формат времени Краткий формат времени
Логический	Да/Нет (по умолчанию) Истина/Ложь Вкл/Выкл
Число десятичных знаков	
Числовой, Денежный, Счетчик	0 − 15 (по умолчанию − Авто)
Подпись Все типы	Используется как заголовок поля в режиме Таблицы, в элементах управления форм и отчетов. Чаще всего совпадает с именем поля

Продолжение табл. 2.2

1	2
Маска ввода	Определяет формат ввода данных в поле (в отличие от формата поля)
Текстовый, Дата/Время	Мастер создания масок ввода позволяет использовать стандартные маски ввода или создать пользовательскую, используя специальные коды: 0 – обязательная цифра 9 – необязательная цифра или пробел # − необязательная цифра, пробел, знаки плюс, минус L – обязательная буква ? – необязательная буква A – обязательная буква или цифра a – необязательная буква или цифра & − обязательный произвольный символ или пробел C – необязательный произвольный символ или пробел > − преобразует все символы справа к верхнему регистру < − преобразует все символы справа к нижнему регистру ! – указывает, что маску заполняют справа налево (символы слева являются необязательными) \ − следующий символ воспринимать буквально «литерал» – литерал в кавычках воспринимается буквально **Например,** >LLLL – маска ввода для текстовых значений из 4 прописных букв (999)000-000 – маска ввода для телефонного номера
Значение по умолчанию	Значение, которое отображается в поле при создании каждой новой записи
Текстовый, MEMO, Дата/время, Гиперссылка	По умолчанию используется значение Null
Числовой, Денежный	По умолчанию используется значение 0
Логический	По умолчанию используется значение **Ложь**

Продолжение табл. 2.2

1	2
Условие на значение Все, **кроме** Счетчик, Поле объекта OLE, код репликации	Выражение, которое должно быть истинным при вводе или редактировании данных (в таблице, форме или запросе). Если вводимые данные не соответствуют заданному условию, то выводится Сообщение об ошибке. Для задания условия обычно используют Построитель выражений. Условие обычно состоит из операторов сравнения и операндов. **Операторы сравнения:** > >= < <= = <> Between … And … − проверка попадания в заданный диапазон (Between 0 And 1). Несколько сравнений могут быть связаны при помощи логических операторов AND, OR (>=50 AND <=100)
Сообщение об ошибке Все, кроме Счетчик, Поле объекта OLE, код репликации	Текст, который выводится в диалоговом окне, если вводимые данные не соответствуют заданному Условию на значение
Обязательное поле Все, **кроме** Счетчик	Если нельзя оставить поле незаполненным, устанавливают значение ДА Не допускает значения поля Null
Пустые строки Текстовый, MEMO	Позволяет ввести в поле пустую строку – ""(две пары кавычек без пробела), что будет означать, что для поля не существует значения. Позволяет различать поля с пустыми значениями Null (незаполненное или неизвестное значение) от полей, не имеющих значений (пустая строка)

Окончание табл. 2.2

1	2
Индексированное поле Текстовый, Числовой, Дата/время, Денежный, Счетчик, Логический	Индексирование поля ускоряет доступ к хранящимся в нем данным. **Индекс** – специальная таблица, содержащая в упорядоченном виде возможные значения поля и соответствующее местоположение записи с данным значением. Выполняет роль оглавления или указателя. Индексирование применяется к полям, для которых часто выполняются операции поиска, сортировки, группировки. Индекс обеспечивает быстрый доступ к данным в поле. Индексы могут быть созданы по одному или нескольким полям (отображаются в меню **Вид/Индекс**). **Первичные ключи индексируются автоматически.**
Сжатие Юникод Текстовый, MEMO	Позволяет сжимать до 1 байта все символы, первый байт которых в кодировке Unicode равен 0. При сохранении происходит сжатие, при выборке – восстановление. По умолчанию установлено значение − Да.

2.6. Порядок формирования схемы БД

Схема БД представляет собой графическое отображение информационно-логической модели предметной области. Установленные в схеме БД связи между таблицами обеспечивают корректную обработку данных при эксплуатации БД. На рис. 2.4 приведена схема учебной БД «Борей» (вызов из меню **Справка/Примеры баз данных…**).

Формирование схемы обычно выполняют после завершения работы над всеми таблицами БД, соблюдая следующий порядок:

1. Закрыть все таблицы БД (создать связи при открытых таблицах нельзя).
2. Переключиться в окно БД.
3. Выполнить команду **Сервис⇨Схема данных**.

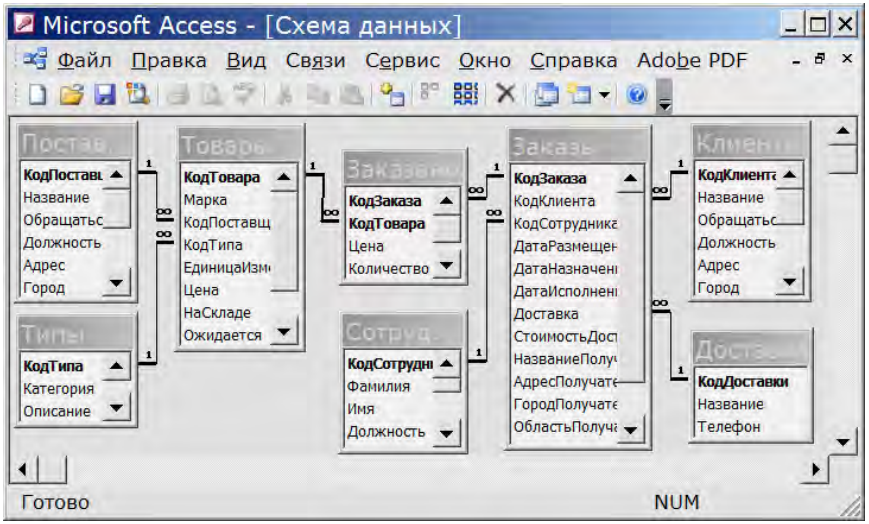

Рис. 2.4. Схема учебной БД «Борей»

4. В списке диалогового окна **Добавление таблицы** выделить таблицы БД и нажать кнопку **Добавить**. Если окно не выводится на экране, необходимо выполнить команду **Связи⇨Добавить таблицу**.

5. Закрыть диалоговое окно **Добавление таблицы.**

6. В окне **Схема данных** необходимо упорядочить главные и подчиненные таблицы следующим образом: главные расположить выше и вокруг подчиненных таблиц.

7. Для формирования связи между таблицами перетащить поле первичного ключа из главной таблицы на поле внешнего ключа подчиненной таблицы.

8. В диалоговом окне **Изменение связей** (рис. 2.5):

- проверить корректность связи (совпадение по полям);
- установить флажки в группе Обеспечение целостности данных.

9. Повторить шаги 7 – 8 для оставшихся таблиц.

10. Сохранить макет схемы данных.

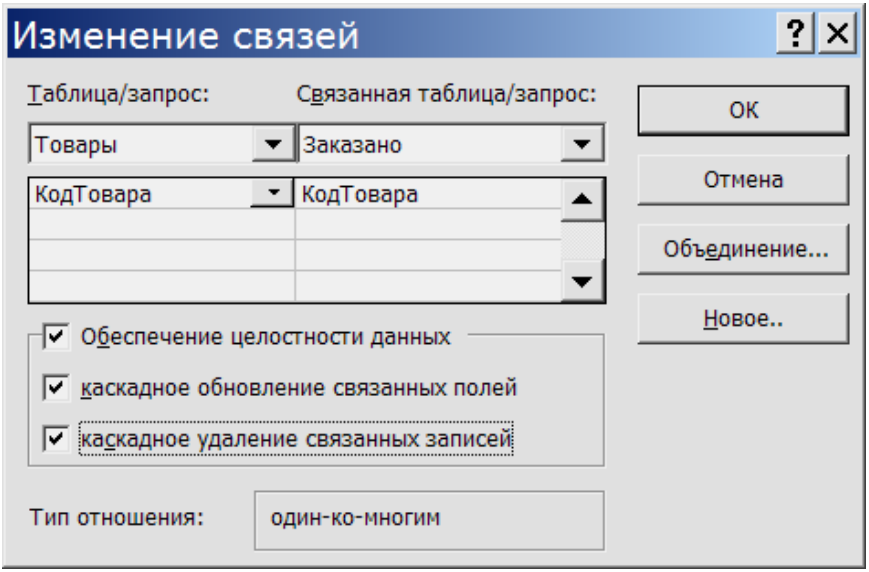

Рис. 2.5. Изменение межтабличных связей

Для успешного формирования связи между таблицами необходимо, чтобы ключевые поля (первичные и внешние) имели:

- одинаковый тип данных,
- содержимое одного типа и
- одинаковый размер поля.

Существует исключение: поле счетчика можно связать с числовым полем, соблюдая следующие требования:

Характеристика поля	**Первичный ключ (главной таблицы)**	**Внешний ключ (подчиненной таблицы)**
1. Тип данных	Счетчик	Числовой
2. Размер поля	Длинное целое	Длинное целое
3. Обязательное поле	Да, по умолчанию	Да
4. Индексированное поле	Да, совпадения не допускаются	Да, допускаются совпадения

Лекция 3. АНАЛИЗ ДАННЫХ ПРИ ПОМОЩИ ЗАПРОСОВ

1. **Назначение и классификация запросов Microsoft Access.**
2. **Способы создания запросов выбора.**
3. **Формирование условий отбора в запросах.**
4. **Вычисляемые поля в запросах.**
5. **Применение функций Microsoft Access.**
6. **Применение параметров в запросах.**
7. **Использование групповых операций.**
8. **Создание перекрестных запросов.**

3.1. Назначение и виды запросов Microsoft Access

Запрос – объект СУБД MS Access, предназначенный для анализа данных из таблиц БД.

Назначение запросов:

- выбор по условиям отбора конкретной информации из одной или нескольких взаимосвязанных таблиц;
- выполнение вычислений и представление результатов в виде таблицы (в т.ч. с использованием функций);
- группирование записей с одинаковыми значениями в полях с применением групповых (статистических) функций;
- обновление данных в исходных таблицах, добавление и удаление записей;
- создание новых таблиц БД на основе данных из существующих таблиц.

Запрос строится на основе:

- исходных таблиц БД,
- таблиц, полученных в результате выполнения запросов на создание таблиц,

- результатов других запросов (временных таблиц с результатами).

Классификация запросов Microsoft Access:

1. По методу создания:

1) *Query Be Example (QBE-**запрос)*** – запрос по образцу, строится при помощи графического средства Microsoft Access, включающего подсхему данных и бланк запроса. Для формирования запроса достаточно переместить при помощи мыши необходимые поля из таблиц и ввести условия отбора данных (условие подбора состоит из примеров данных, которые и составят результат запроса, также называется запросом типа выборки)

2) *Structured Query Language (SQL-**запрос)*** – язык структурированных запросов (для их формирования **программисты** применяют специальные инструкции и функции Microsoft Access).

QBE-запрос может быть трансформирован в SQL с помощью меню **Вид⇨Режим** SQL. А вносимые в SQL-инструкции изменения автоматически отображаются в спецификации QBE-запроса.

2. По функциональному признаку запросы можно разделить:

1) ***на запросы выбора*** (результат обработки данных существует в виде таблицы до закрытия запроса):

a) простые;
b) с условиями;
c) с вычислениями;
d) с параметрами (необходимые для запроса значения задаются пользователем в диалоговом окне при выполнении запроса);
e) с групповыми операциями (обеспечивают статистическую обработку данных);

f) перекрестные (обеспечивают компактное отображение данных с объединением однотипной информации по строкам и столбцам, на пересечении которых находится итоговое значение. Например, Продавец – Покупатель, объем операций);

2) ***запросы действия*** (запросы на изменение, приводят к изменениям данных в исходных таблицах или созданию новых таблиц, сохраненных как объект таблица)

a) на создание таблицы;
b) обновление данных;
c) добавление данных;
d) удаление данных.

Результатом обработки исходной информации по запросу может быть:

- ***для запроса выбора*** – **временная таблица**, в которую включены выбранные из БД сведения, удовлетворяющие критериям запроса. Подмножество данных, отображаемых запросом, называется **динамическим набором данных** (DYNASET, поскольку каждый раз отображает «свежие» данные из таблиц). Эта таблица **существует до закрытия запроса**;
- ***для запроса действия*** – **изменение данных** в базовых таблицах или новый объект – таблица.

Поля в таблице с результатами запроса наследуют свойства полей исходных таблиц.

Последовательное выполнение ряда запросов позволяет решать сложные задачи, не прибегая к традиционному программированию.

3.2. Способы создания запросов выбора

Создание запросов выполняется на **вкладке Запросы** окна БД. После нажатия кнопки **Создать** в диалоговом окне необходимо выбрать режим создания (рис. 3.1).

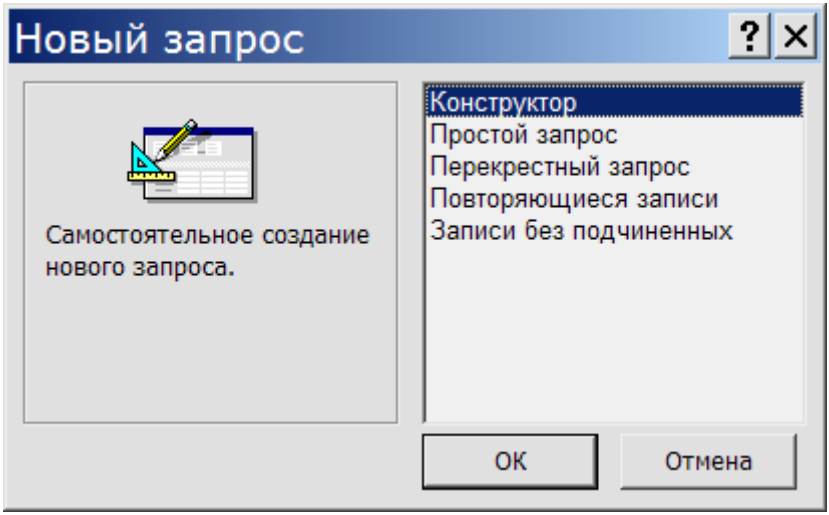

Рис. 3.1. Режимы создания запросов

Запросы выбора по образцу (QBE-запросы) могут быть созданы в двух режимах:

1. **Режим Мастера**.

В этом режиме пошаговые инструкции позволяют сформировать:

- **простой запрос** на выборку;
- **запрос для поиска повторяющихся записей** (повтор значений в заданных полях, например, клиенты из одной страны или города);
- **запрос для поиска записей без подчинения** (например, найти клиентов, не имеющих заказов);
- **перекрестный запрос**.

2. **Режим Конструктора.**

Если выбирается режим **Конструктора**, на экране отображается **окно Конструктора запроса** (рис. 3.2).

Подсхема данных содержит исходные таблицы (в виде списка полей) с установленными связями.

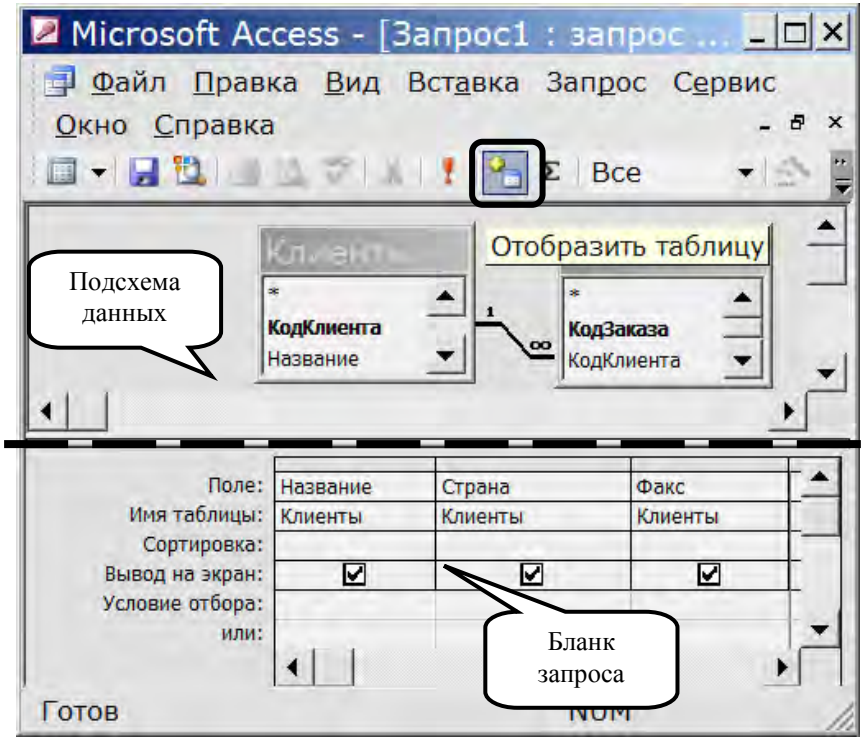

Рис. 3.2. Окно Конструктора запроса

Если связи между таблицами отсутствуют, то для добавления недостающих таблиц-связок следует нажать кнопку **Отобразить таблицу**. Кроме того, результат запроса зависит от вида соединения таблиц (внутренне или внешнее), выбранного для каждой связи в окне Параметры объединения.

Бланк запроса по образцу (нижняя панель) содержит **столбцы** (имена полей, включаемых в запрос) и **строки**:

- ***Поле*** (может содержать имя поля или выражение)*;*
- ***Имя таблицы***;
- ***Сортировка****;*
- ***Вывод на экран****;*
- ***Условие отбора / или****.*

Строка Поле определяет имена полей, включенных в запрос (динамический набор данных). Поля могут использоваться для включения в результат запроса, для вычислений, для сортировки и задания условий отбора (могут не отображаться на экране).

Добавить поля в Бланк запроса можно несколькими способами:

1. Открыть раскрывающийся список в Поле Бланка запроса.
2. Выделить поле таблицы из подсхемы данных и, удерживая левую кнопку мыши, перетащить в Бланк запроса.
3. Выделить поле таблицы из подсхемы данных и выполнить двойной щелчок левой кнопкой мыши.

Для добавления в Бланк запроса **всех полей** из таблицы необходимо выбрать * (**звездочку**) и перетащить в область Бланка запроса. Следует учитывать, что если в Бланк запроса включены все поля с использованием *, то **условия отбора задать нельзя**.

Строка Сортировка определяет порядок и поля сортировки (до 3 полей в запросе).

Строка Вывод на экран – если установлен флажок, поле выводится в результирующей таблице, если нет – не выводится, но в запросе участвует.

Строка Условие отбора (УО) позволяет задать инструкции, при помощи которых выбираются записи, в динамический набор данных.

Для полей, включенных в запрос, **дополнительно можно установить Свойства (формат, маска ввода, подпись).**

После заполнения Бланка запроса необходимо выполнить запрос, нажав на панели инструментов **Конструктор запросов** кнопку  или вызвав меню **Запрос⇨Запуск.**

При закрытии окна запроса необходимо подтвердить сохранение запроса и присвоить имя, **изменив имя по умолчанию** (Запрос 1).

3.3. Формирование условий отбора в запросах

Условие отбора (УО) представляет собой **выражение**, состоящее из **операндов** и **операторов** (табл. 3.1), в результате работы которого возвращается единственное значение.

Выражения могут также выполнять вычисления, обрабатывать текст или управлять значениями поля.

В качестве **операндов** могут использоваться **константы**, **литералы** (заданные пользователем числовые, текстовые значения, значения даты/времени), **идентификаторы объектов** БД (имена полей в таблицах, отчетах и формах в формате [Имя объекта]![Имя поля], например, [Товары]![Цена]), а также **функции**.

Таблица 3.1

Операторы MS Access

Оператор	**Пример**	**Описание**
1	2	3
Арифметические операторы (результат – число)		
+	[итог]+[надбавка]	Складывает два операнда
-	Date()-7	Считает разность двух операндов
- **(унарный)**	-123456	Меняет знак операнда на противоположный
/	[количество] / 12.55	Делит один операнд на другой
\	[коробок]\2	Делит один операнд на другой нацело
Mod	[Коробок] Mod 12	Возвращает остаток от деления на число
^	[Основание]^[Показатель]	Возводит Основание в степень Показатель
Операторы сравнения (результат – True or False)		
<	1<100	Меньше (True)
>	1>100	Больше (False)
=	100=100	Равно (True)
>=	100>=1	Больше либо равно (True)
<=	100<=20	Меньше либо равно (False)
<>	1<>100	Не равно (True)

Окончание табл. 3.1

1	2	3
Логические операторы (результат – True or False)		
And	A And D	Конъюнкция (логическое И)
Or	A Or H	Дизъюнкция (логическое ИЛИ)
Not	Not H	Логическое отрицание
Xor	A Xor H	Исключающее ИЛИ
Eqv	A Eqv H	Логическая эквивалентность
Imp	A Imp H	Логическая импликация
Оператор слияния текстовых строк (результат – строка)		
&	“Проживает по адресу: ” & [Клиенты]![Адрес]	Конкатенация (объединяет 2 текстовых значения в единую строку символов)
Операторы сравнения с образцом (результат – True or False)		
Between	Between 0 And 100	Определяет, находится ли числовое значение в определенном диапазоне значений
Is	Is Null , Is Not Null	При использовании совместно с Null определяет, является ли значение Null или Not Null
In	In («Минск», «Киев»)	Определяет, является ли строковое значение элементом списка значений
Like	Like “лав*”	Определяет, начинается ли строковое значение с указанных символов
Операторы идентификации		
«!»	Класс Объекта ! Имя Объекта	Позволяют объединить имена объектов для отбора специфических объектов или их свойств, различать имена объектов и их свойств, идентифицировать определенные поля в таблицах
«.»	Имя объекта . Метод	

Операторы сравнения позволяют производить поиск более чем по одному значению.

Between … And позволяет выбирать значения в заданном диапазоне.

Для задания сложных условий отбора могут использоваться логические операторы – *And, Xor, Not*. Xor.

Условия отбора, размещенные в разных столбцах, на одной строке объединяются при помощи логического оператора **И** (AND), а на разных строках – **ИЛИ** (OR).

Если значение в строке УО записано **без оператора сравнения**, то подразумевается оператор "=" **(равно)**.

Результатом обработки выражения в строке УО является одно из значений True (Истина) или False (Ложь).

Условие отбора можно установить следующими способами:

1. Если поле имеет тип **числовой, денежный или счетчик**, то в строку «Условие отбора» вводится **оператор сравнения и число** без дополнительных символов (например, <455 или >=234).

2. Если поле **текстовое**, то просто вводится текст или текст с символами шаблона (* **и** ?, причем «*» заменяет любое число символов, а «?» – не более одного символа). После ввода в строке отобразится **Like "текст"**, при этом "" и Like система добавляет автоматически.

3. В поля типа **дата/время** вводится оператор сравнения и критерий отбора. Например, для отбора значений даты после 9 октября 2000 года в качестве УО вводится выражение >9.10.2000, которое преобразуется к виду >#09.10.2000# (знаки «#» система добавляет автоматически).

4. В полях с **логическим** типом данных используются **константы *Да, Нет (Истина, Ложь)*.**

5. Для поиска **пустых полей** в УО используется выражение Is Null.

Для формирования сложных выражений в условии отбора рекомендуется использовать **Мастер построения выражений** (рис. 3.3), нажав на панели инструментов **Конструктор запросов** кнопку  .

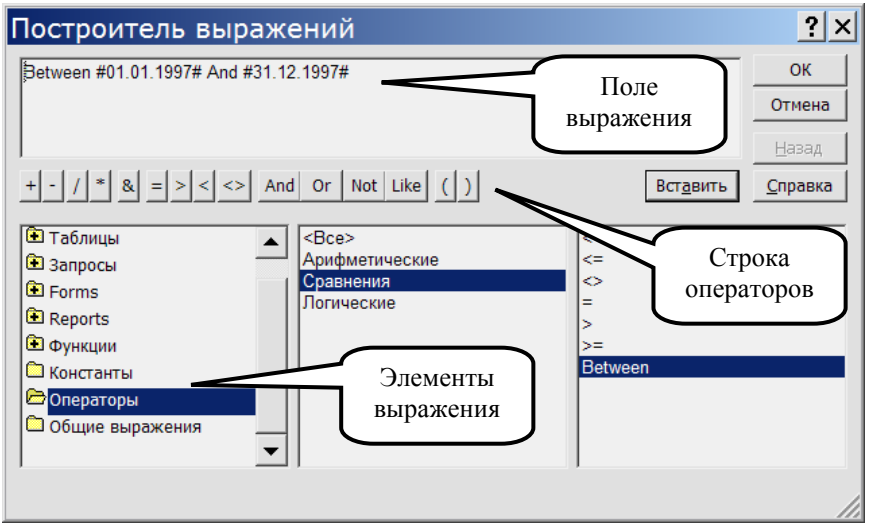

Рис. 3.3. Окно Построителя выражений

В нижней части окна Построителя находятся **элементы выражения**:

- в левом поле выводятся папки, содержащие таблицы, запросы, формы, объекты базы данных, встроенные и определенные пользователем функции, константы, операторы и общие выражения;
- в среднем поле задаются определенные элементы или типы элементов для папки, заданной в левом поле. Например, если выбрать в левом поле **Операторы**, то в среднем поле появится список всех типов операторов Microsoft Access;
- в правом поле выводится список значений (если они существуют) для элементов, заданных в левом и среднем полях. Например, если выбрать в левом поле Операторы и тип операторов в среднем, то в правом поле будет выведен список всех встроенных операторов выбранного типа.

3.4. Вычисляемые поля в запросах

В запросе могут производиться вычисления. Для этого необходимо создать новое (вычисляемое) поле, в котором будут отображаться результаты вычислений, определенных в выражении.

Результат этого поля будет **пересчитываться при каждом выполнении запроса** (на основе текущих значений в базовых таблицах).

При вычислениях используются арифметические **операторы**, **объекты** и **функции** MS Access.

Выражение можно ввести в пустую ячейку строки Поле. MS Access самостоятельно сформирует стандартное имя вычисляемого поля (рис. 3.4) – **Выражение1**(2, 3…), которое помещается перед выражением и отделяется : (двоеточием).

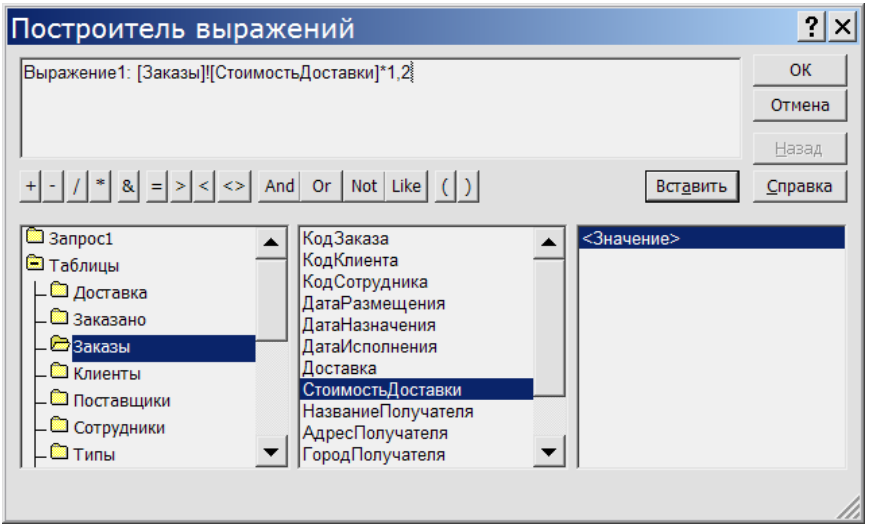

Рис. 3.4. Формирование вычисляемого поля

Имя **Выражение1** становится заголовком поля в таблице с результатами запроса.

Например, необходимо рассчитать стоимость заказов, сформированных каждым сотрудником, с учетом цены товара, проданного количества и скидки (БД Борей).

При построении запроса необходимо:

1. В качестве источника указать взаимосвязанные таблицы Сотрудники, Заказано и Заказы из БД Борей.
2. Включить в Бланк запроса поле Фамилия из таблицы Сотрудники.
3. Для формирования вычисляемого поля с собственным именем выполнить следующие действия:

- в пустом поле Бланка запроса вызвать Построитель выражений (рис. 3.5),
- ввести **имя** вычисляемого поля (уникальное),
- ввести : **(двоеточие)**,
- после : (двоеточия) построить выражение, выбирая нужные объекты, функции, операторы и значения (рис. 3.6).

Во избежание ошибок имена объектов БД (например, ссылки на поля таблицы) рекомендуется выбирать из списка в Построителе выражений, а не вводить вручную.

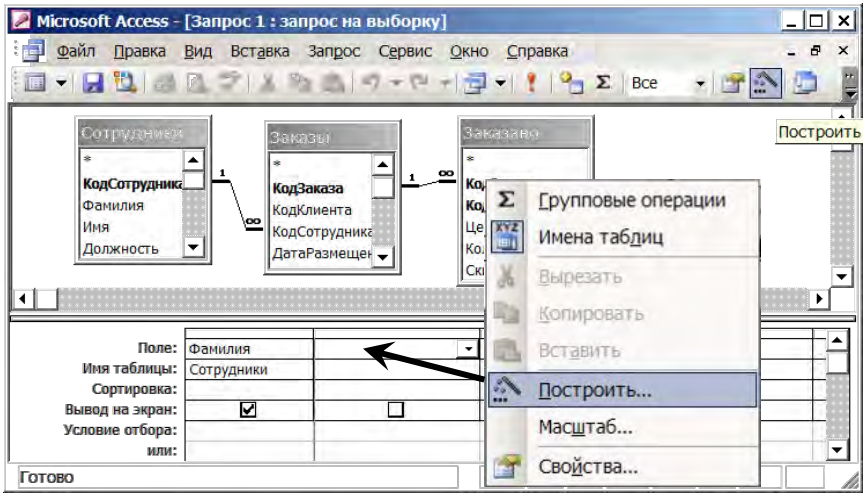

Рис. 3.5. Вызов Построителя выражения

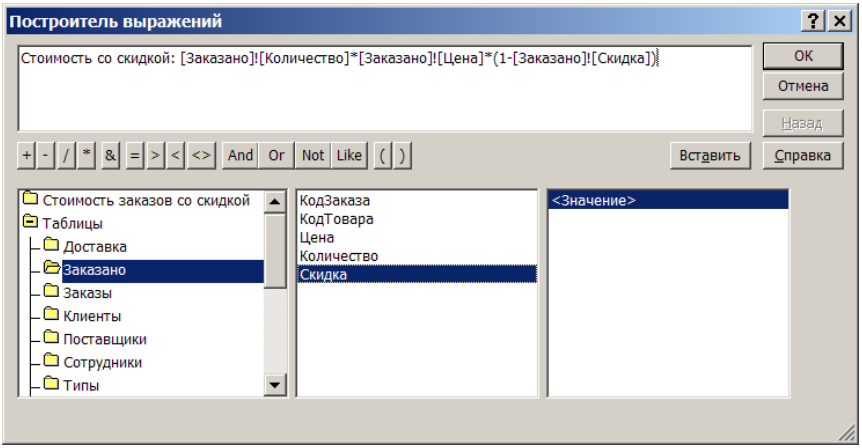

Рис. 3.6. Формирование выражения для вычисляемого поля

После закрытия окна Построителя выражения в ячейке Поле бланка запроса отобразится результат построения вычисляемого поля. Строка **Имя таблицы** для вычисляемого поля остается **пустой** (рис. 3.7).

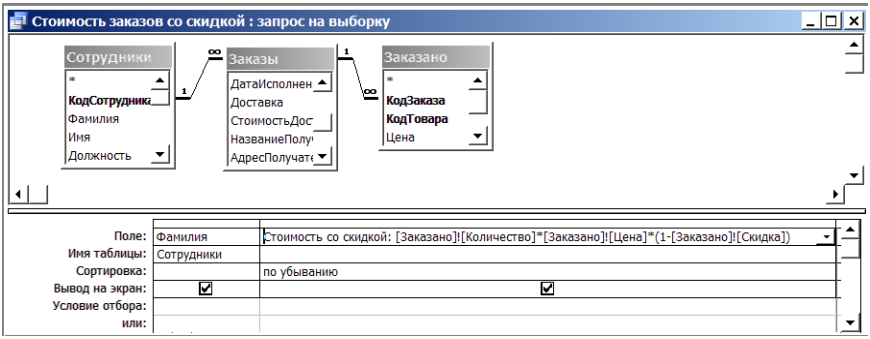

Рис. 3.7. Вычисляемое поле

Для построения текстовых выражений используется оператор конкатенации – &. Фрагменты текста заключаются в двойные кавычки "".

Для обработки текстовых значений также используются текстовые функции, доступные в Построителе выражений из раздела ***Функции/Встроенные функции, категория Текстовые***…

Например, чтобы создать в запросе поле, отображающее фамилию и инициал клиента (Петров И.), необходимо в свободном поле Бланка запроса построить формулу:

ФИО: [Клиенты]![Фамилия] & " " & Left([Клиенты]![Имя];1) & "."

В данном случае применяется текстовая функция Left, которая извлекает одну первую букву из значения в поле имени клиента – Left**([Клиенты]![Имя];1)**.

Аргументы функции, представляющие собой имена объектов БД, рекомендуется не вводить вручную, а выбирать из списка в Построителе выражений.

3.5. Применение функций Microsoft Access

Функции возвращают в выражение значение вместо имени функции. Встроенные функции Microsoft Access доступны в Построителе выражений и по функциональному назначению делятся на определенные категории (табл. 3.2).

Значение, возвращаемое функцией, определяется ее типом. Например, функция Date (), имеющая пустой список аргументов, возвращает текущую дату.

Синтаксически функция выделяется круглыми скобками, следующими сразу же за ее идентификатором. Многие функции требуют наличия аргументов, которые записываются в скобках через запятую при обращении к функции. Наличие круглых скобок после имени функции синтаксически обязательно, даже если функция не имеет аргументов.

При работе с функциями рекомендуется выбирать функции и задавать аргументы не вручную, а с помощью Построителя выражений.

Таблица 3.2

Категории функций MS Access

Категория	Назначение функций
1	2
Массивы	Используются для определения размерностей массивов при программировании на VBA
Преобразование	Применяются для преобразования одного типа данных в другой. Например, для преобразования числа в строку служит функция Str (), а обратное преобразование выполняет функция Val ()
База данных	С помощью функций для работы с объектами баз данных осуществляется обращение к объектам баз данных: таблицам, запросам, формам, отчетам, макросам и модулям. Например, функция CurrentDB () возвращает ссылку на объект Database, с которым работает Access в текущий момент
Дата/время	Используются для проведения операций со значениями даты и времени. Например, функция DateDiff () вычисляет промежуток между двумя датами, а функция Now () возвращает системную дату и время часов компьютера
По подмножеству	Статистические функции по подмножествам записей аналогичны статистическим функциям SQL, но работают с вычисляемыми значениями, а не со значениями, содержащимися в полях запросов. Примером статистической функции SQL является StDev (), а соответствующей ей статистической функцией по подмножеству записей— DStDev (). Обе они вычисляют стандартное отклонение для заданного множества значений

Продолжение табл. 3.2

1	2
Обработка ошибок	Используются для отслеживания ошибок. Например, функция Error () выводит сообщение об ошибке по ее номеру. Наличие таких функций в выражении можно считать признаком хорошего тона, поскольку используемое выражение, корректно обрабатывает ошибочный результат и предупреждает пользователя
Финансовые	Аналогичны финансовым функциям в Microsoft Excel и используют те же аргументы. Например, функция Rate () возвращает процентную ставку, необходимую для получения указанной суммы на базе исходной путем регулярных взносов за определенный срок
Общие	Используются в основном при программировании на VBA для извлечения вспомогательной информации и управления ходом выполнения программы. Например, функция Command() служит для извлечения аргументов командной строки при выполнении программы, написанной на VBA
Проверка	Функции проверяют тип аргумента. Например, Isnumeric() — возвращает True, если аргумент имеет один из числовых типов данных, и False — в противном случае; Is0bject() — возвращает True, если аргумент— объект OLE Automation, и False — в противном случае
Математические	Используются для выполнения различных математических и тригонометрических вычислений. Например вычисления логарифма Log () или синуса числа Sin ()

Окончание табл. 3.2

1	2
Сообщения	Позволяют выводить сообщения или вводить новые данные, а также устанавливать различные параметры ввода/вывода. Например, функция ввода данных Inputbox() выводит диалоговое окно
Управление	Используются для выбора значения из нескольких альтернатив. Например, функция Ilf (Expr, Truepart, Falsepart) возвращает значение выражения Truepart, если значение условного выражения Expr равно True, или значение выражения Falsepart, если значение Expr есть False. Особенность функции в том, что независимо от значения выражения Expr, вычисляется как выражение Truepart, так и Falsepart. В качестве каждого из аргументов может быть достаточно сложное выражение, например, включающее другие функции. Функция Choose () возвращает значение, соответствующее заданному положению в списке значений, а функция Switch () возвращает значение, связанное с первым из последовательности выражением, имеющим значение True
Статистические	Статистические функции SQL обобщают данные в полях для выбранных записей, например, при подведении итогов (Avg(), Count(), Sum())
Текстовые	Позволяют проводить различные операции над строками. Например, функция Trim () возвращает строку, заданную в качестве аргумента, без начальных и заключительных пробелов. Функция UCase () возвращает строковое значение, преобразованное в прописные буквы

В выражении могут использоваться несколько функций, функции могут быть аргументами других функций.

Например, в запросе можно создать поле, содержащее фамилию и инициалы, причем результат записывается прописными буквами:

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

Функции категории **Дата/время** позволяют решать различные задачи для полей, содержащих даты.

Например, с помощью функции DatePart() можно определить квартал, в котором был сделан заказ:

Квартал: DatePart ("q"; [Заказы]![ДатаРазмещения])

Вместо аргумента "q" можно использовать значения "yyyy", "m", "ww" и др., чтобы определить для исходной даты год, месяц, неделю и др.

Например, с помощью функции Month() можно определить порядковый номер месяца, в котором был сделан заказ:

Месяц: Month ([Заказы]![ДатаРазмещения])

Например, с помощью функции MonthName() можно определить название месяца, в котором был сделан заказ:

Месяц: MonthName (Month([Заказы]![ДатаРазмещения]); 0)

Например, с помощью функций IIf(), DatePart(), Date(), Year() можно определить возраст или стаж сотрудника на текущую дату (число полных лет):

Число лет: IIf (DatePart("y";Date();2)>DatePart("y";[Дата];2);
Year(Date())-Year([Дата]);Year(Date())-Year([Дата])-1)

Аналогично можно рассчитать, сколько лет осталось до пенсии сотруднику, если известны пол сотрудника, дата его рождения и возрастной порог выхода на пенсию.

3.6. Применение параметров в запросах

Запросы с параметрами не являются самостоятельным типом запросов. Они расширяют возможности запросов выбора с условиями и вычислениями.

Необходимость запросов с параметрами обусловлена тем, что запросы выбора не всегда выполняются пользователем при неизменных критериях отбора. Чаще всего запросы представляют собой незначительно видоизмененные варианты базового запроса, условия которых изменяются от случая к случаю, но незначительно. Для их реализации **предусмотрены параметры**, значения которых задаются пользователем в диалоговом окне при выполнении запроса.

Параметры используются:

- в условиях отбора,
- в вычисляемых полях.

Для проектирования параметрического запроса необходимо заполнить строку УО или вычисляемое поле не конкретным значением, а названием параметра, заключенным в [ ], значение которого будет запрошено у пользователя в диалоговом окне «Введите значение параметра».

Название параметра должно быть уникальным и не должно совпадать с именами полей.

Например, параметры [Введите фамилию], [Введите курс валюты].

Если в запрос вводится несколько параметров, то порядок их появления определяется порядком расположения полей в Бланке запроса.

3.7. Использование групповых операций

В запросах могут создаваться **итоговые поля**, позволяющие применять групповые функции для проведения статистических вычислений. Например, для подсчета числа заказов или количества проданных товаров.

Для вывода в бланке запроса строки Групповая операция необходимо выполнить команду **Вид⇨Групповые операции** или нажать кнопку Σ на Панели инструментов.

В запросы с групповыми операциями обычно включаются, как минимум, **два поля**, по одному из которых выполняется группировка, а по второму применяется статистическая функция (табл. 3.3).

Данные в запросах с групповыми операциями обычно **сортируют** по полю со статистической функцией.

Для поля со статистической функцией рекомендуется задать **свойство Подпись**.

Таблица 3.3

Статистические функции MS Access

Функция	Действие
Sum	Суммирование значений определенного поля в записях, отобранных запросом
Avg	Вычисление среднего значения определенного поля в записях, отобранных запросом
Min	Вычисление минимального значения определенного поля в записях, отобранных запросом
Max	Вычисление максимального значения определенного поля в записях, отобранных запросом
Count	Вычисление количества записей, отобранных запросом, в определенном поле
First	Определение первого значения в указанном поле среди записей, отобранных запросом
Last	Определение последнего значения в указанном поле среди записей, отобранных запросом
StDev	Вычисление стандартного отклонения значений данного поля для всех записей, отобранных запросом
Var	Вычисление вариации значений данного поля для всех записей, отобранных запросом

Например, чтобы подсчитать количество заказов, сформированных каждым сотрудником фирмы (БД Борей), необходимо:

1. Создать запрос выбора, добавив в подсхему данных таблицы **Сотрудники** и **Заказы**.
2. Включить в бланк запроса поля: **Фамилия** (из таблицы Сотрудники) и **Код заказа** (из таблицы Заказы).
3. Добавить в бланк запроса строку **Групповая операция**.
4. В строке **Групповая операция** по полю **Фамилия** выбрать **Группировка**, а по полю **Код заказа** – статистическую функцию Count (рис. 3.8).
5. По полю **Код заказа** выбрать сортировку по убыванию и задать свойство **Подпись поля** «Количество заказов».

На рис. 3.9 представлены варианты выполнения запросов без изменения и с изменением свойства Подпись поля для поля со статистической функцией.

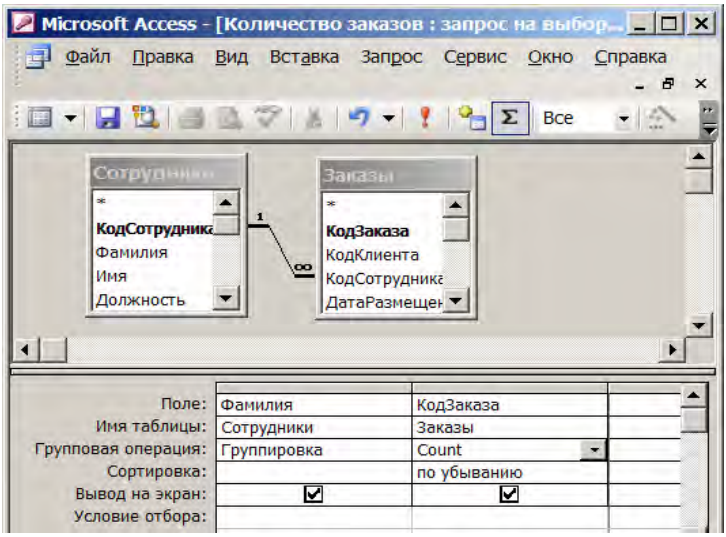

Рис. 3.8. Макет запроса с групповой операцией

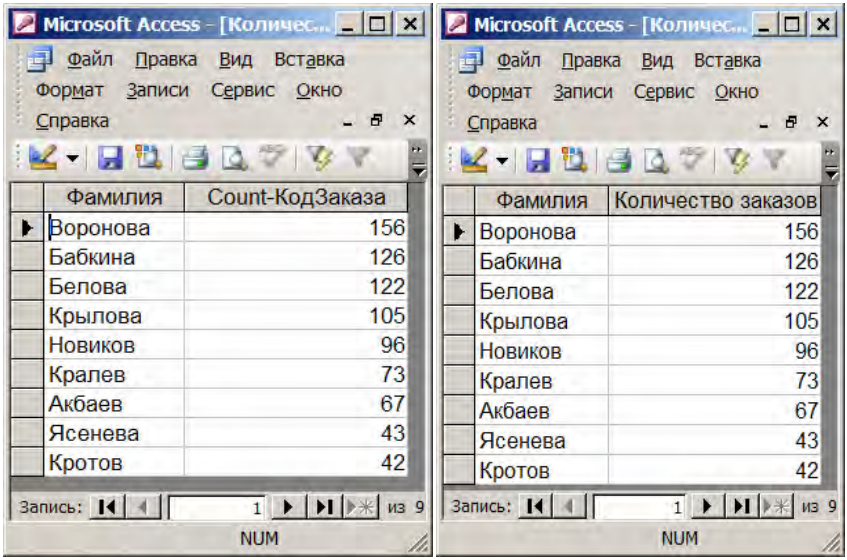

Рис. 3.9. Результаты запроса с групповой операцией

3.8. Создание перекрестных запросов

Перекрестный запрос позволяет компактно отобразить данные, используя объединение однотипной информации (в формате строк-столбцов, на пересечении которых вычисляется итог).

Перекрестные запросы обладают рядом достоинств:

- возможностью обработки значительного объема данных и вывода их в формате, который удобен для дальнейшего построения графиков и диаграмм;
- простотой и скоростью разработки сложных запросов с несколькими уровнями детализации.

К недостаткам перекрестных запросов можно отнести невозможность сортировки таблицы результатов по значениям, содержащимся в столбцах, поскольку в большинстве случаев одновременное упорядочивание данных

в столбцах по всем строкам невыполнимо. Перекрестные запросы удобны для представления данных в виде таблицы, но на их основе довольно сложно построить отчеты.

Перекрестные запросы можно создать:

- в режиме **Мастера** перекрестного запроса;
- в режиме **Конструктора**.

Создание в режиме **Мастера перекрестного запроса** требует предварительной подготовки: необходимо создать запрос выбора, содержащий поля из всех таблиц, используемых для перекрестного запроса.

В режиме **Мастера перекрестного запроса** в пошаговом режиме необходимо:

- выбрать в качестве источника запрос,
- задать поля – заголовки строк и заголовки столбцов
- выбрать значения, которые отображаются на пересечении, и итоговую функцию. Дополнительно можно вычислить итоги по строкам.

Например, необходимо создать перекрестный запрос, позволяющий подсчитать количество товаров, проданных каждым сотрудником по каждой категории (БД Борей).

Предварительно можно создать запрос выбора **Исходные данные**, включающий все необходимые поля из таблиц **Сотрудники**, **Заказы**, **Заказано**, **Товары**, **Типы**.

В режиме **Мастера перекрестного запроса** на первом шаге выбрать в качестве источника созданный запрос. Далее, при формировании макета перекрестного запроса, в качестве **заголовка строк** выбрать фамилию сотрудника, в качестве **заголовков столбцов** – название категории, а **вычисления** выполнить по полю Количество (выбрать функцию **Сумма**).

Окно Мастера перекрестного запроса со сформированным макетом приведено на рис. 3.10, результат работы запроса отображен на рис. 3.11, а макет перекрестного запроса в режиме Конструктора – на рис. 3.12.

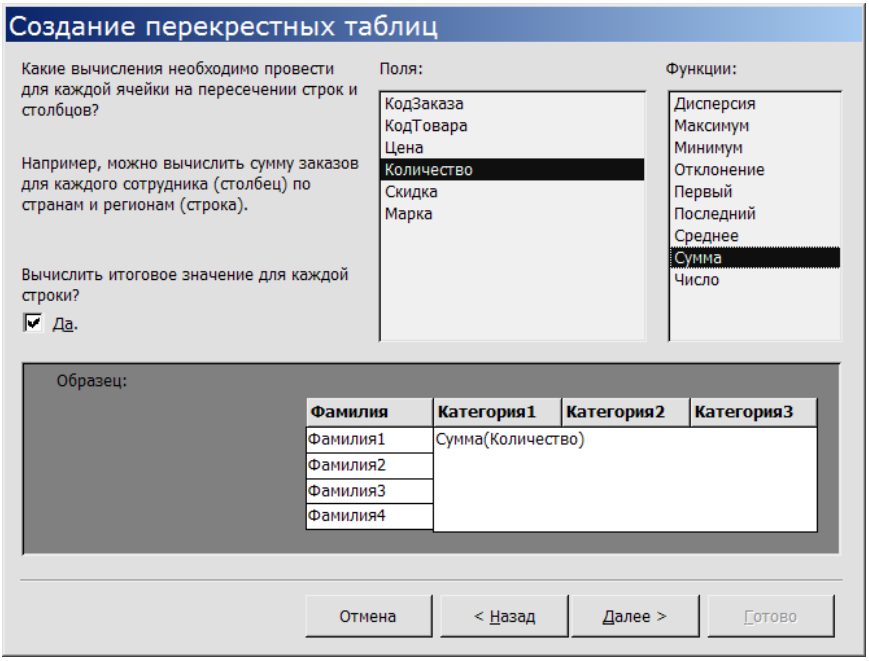

Рис. 3.10. Макет перекрестного запроса в режиме Мастера

Создание перекрестного запроса **в режиме Конструктора** не требует предварительного создания запроса и выполняется в следующем порядке:

- добавить в подсхему данных все таблицы,
- выполнить команду **Запрос⇨Перекрестный**,
- добавить в бланк запроса **3 поля**: **Фамилия** – **заголовки строк**, **Категория** – **заголовки столбцов** и **Количество** – **значения**,
- по полям **Заголовки строк** и **Заголовки столбцов** выполнить операцию **Группировка**,
- по полю **Значение** – выбрать статистическую функцию SUM (рис. 3.13).

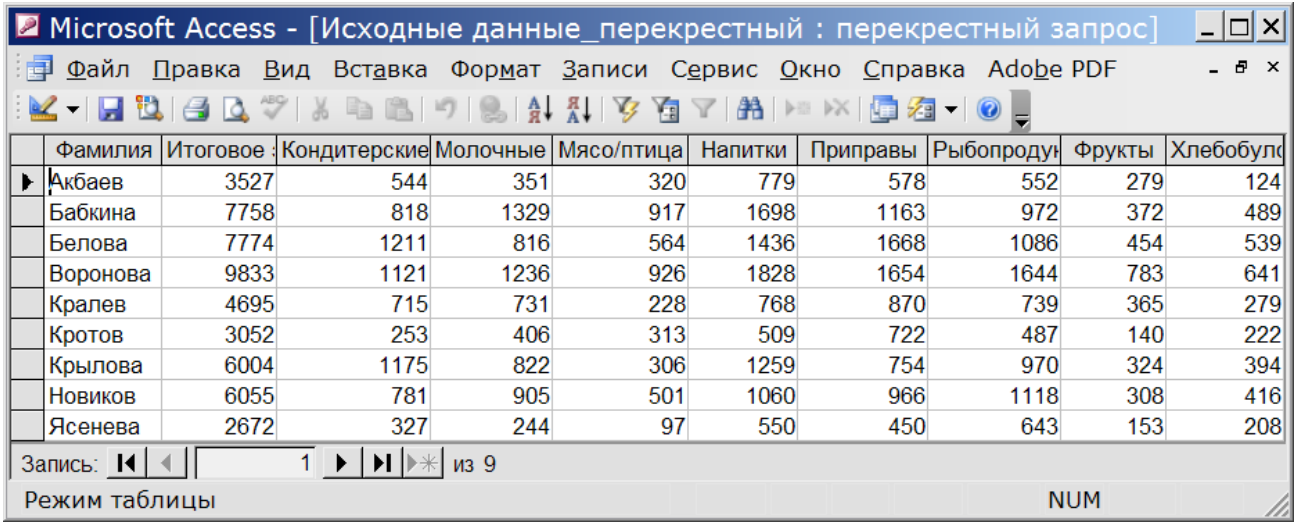

Рис. 3.11. Результат работы перекрестного запроса в режиме Таблицы

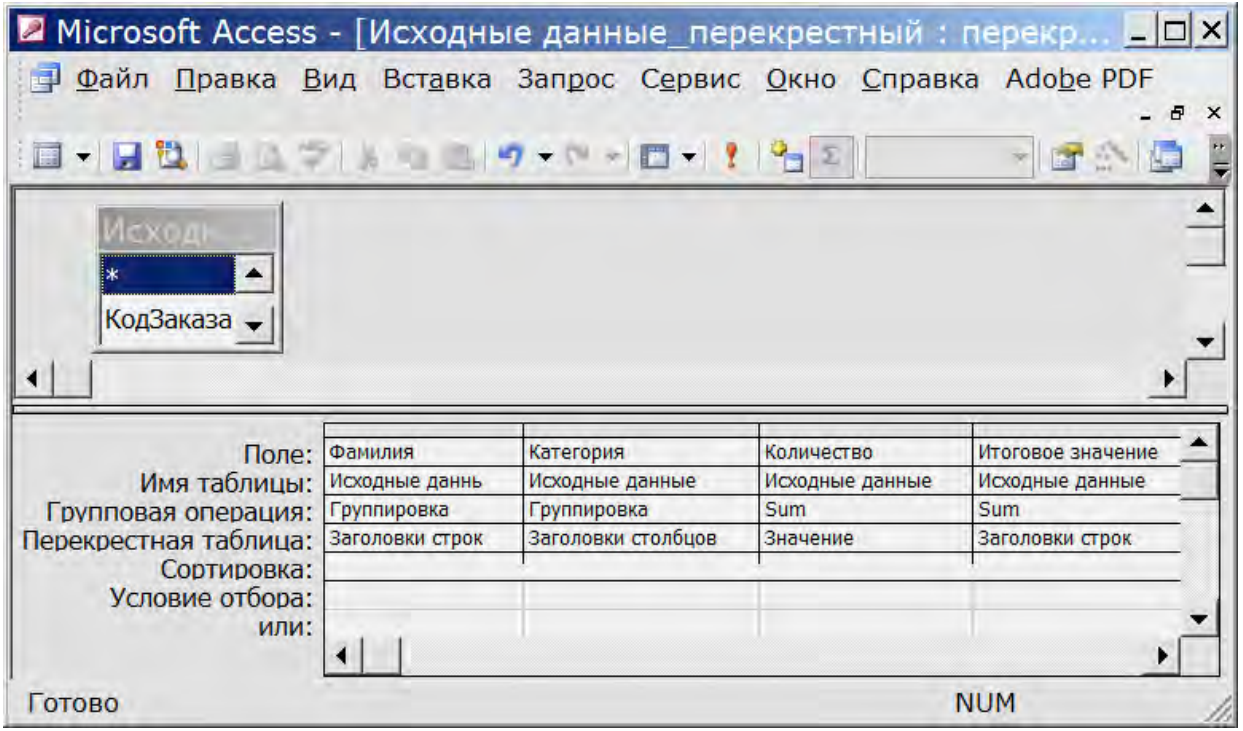

Рис. 3.12. Макет перекрестного запроса в режиме Конструктора

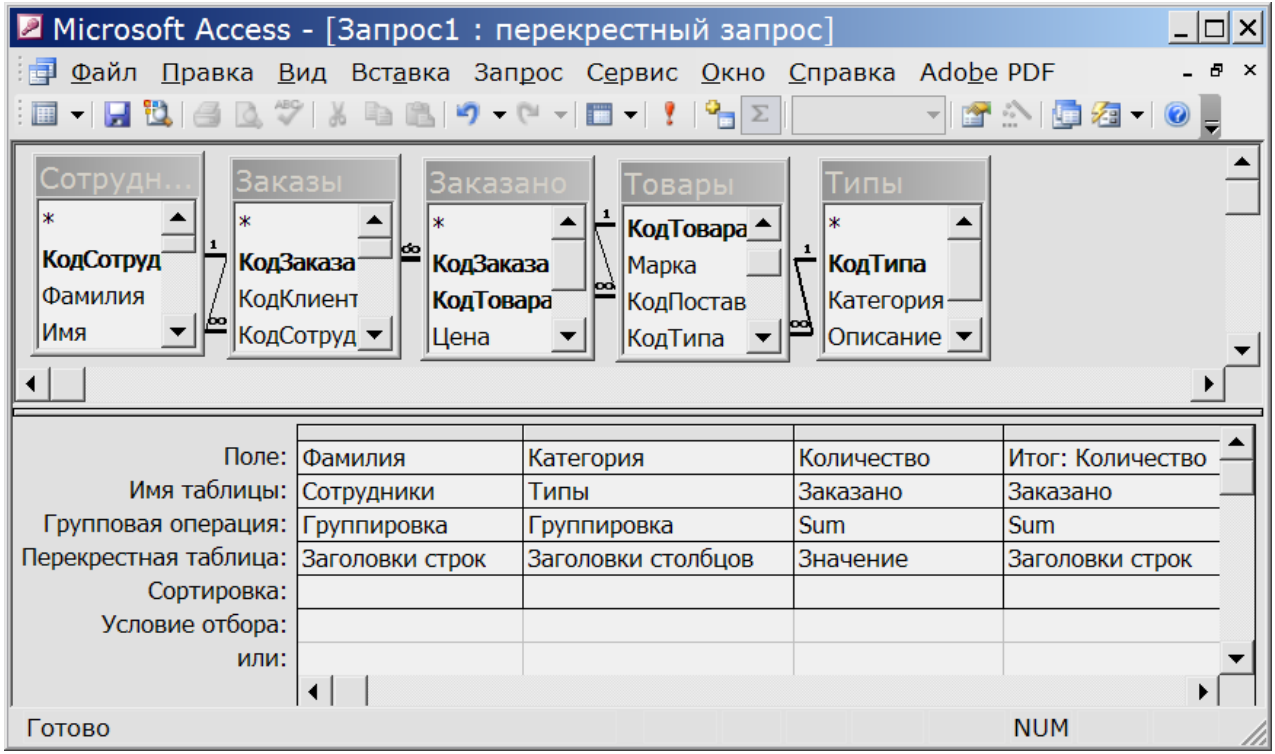

Рис. 3.13. Макет перекрестного запроса при создании в режиме Конструктора

Лекция 4. СОЗДАНИЕ ЗАПРОСОВ ДЕЙСТВИЯ

1. **Назначение и особенности запросов действия.**
2. **Понятие целостности данных и ее обеспечение.**
3. **Методика формирования запросов действия.**
4. **Запрос на создание новой таблицы.**
5. **Запрос на обновление записей.**
6. **Запрос на добавление записей.**
7. **Запрос на удаление записей.**

4.1. Назначение и особенности запросов действия

С помощью запросов действия пользователь может создавать новые таблицы, обновлять, добавлять или удалять группы записей в базовых таблицах.

Особенность запросов действий в том, что они **не создают динамического набора данных,** а вносят изменения в существующие таблицы БД или создают новые таблицы (объекты) БД на вкладке Таблица.

Отличительной особенностью запросов действия является невозможность отмены результата их выполнения.

4.2. Понятие целостности данных и ее обеспечение

Для корректной работы запросов действий в БД для всех межтабличных связей в схеме БД должен быть установлен **параметр Обеспечение целостности данных** и обеспечивающие эту целостность **каскадное удаление** и **каскадное обновление записей** (**Сервис⇨Схема данных**, **КЗМ** на связи, **Изменить связь**…, рис.4.1).

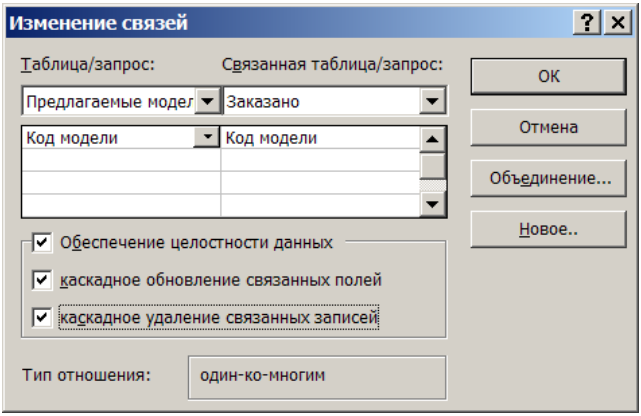

Рис. 4.1. Параметры обеспечения целостности данных

Целостность данных означает систему правил Microsoft Access для поддержания установленных межтабличных связей при вводе, редактировании и удалении записей.

Целостность данных обеспечивает корректную обработку данных, защиту от случайного удаления или изменения связанных данных.

Установить целостность данных можно только в том случае, если выполнены следующие условия:

- связанное поле главной таблицы является ключевым полем или имеет уникальный индекс;
- связанные поля имеют один тип данных. Возможны **два исключения**:
  - поле Счетчика может быть связано с числовым полем, если свойство Размер поля для обоих полей имеет значение «Длинное целое»;
  - поле Счетчика можно связать с числовым полем, если свойство Размер поля для обоих полей имеет значение «Код репликации»;
- обе таблицы принадлежат одной базе данных Microsoft Access. Для связанных таблиц из баз данных других форматов установить целостность данных невозможно.

Если установлен только флажок **Обеспечение целостности данных**, то накладываются следующие **ограничения**:

1. **Невозможно ввести в поле внешнего ключа** связанной таблицы значение, не содержащееся в ключевом поле главной таблицы. Однако в поле внешнего ключа **возможен ввод пустых значений,** показывающих, что записи не являются связанными.

Например, нельзя сохранить запись, регистрирующую заказ для несуществующего клиента, но можно создать запись для заказа, который пока не отнесен ни к одному из клиентов, если ввести пустое значение в поле «КодКлиента».

2. **Не допускается удаление записи из главной таблицы**, если существуют связанные с ней записи в подчиненной таблице.

Например, невозможно удалить запись из таблицы «Сотрудники», если в таблице «Заказы» имеются заказы, относящиеся к данному сотруднику.

3. **Невозможно изменить значение ключевого поля в главной таблице**, если существуют записи, связанные с данной.

Например, невозможно изменить КодСотрудника в таблице «Сотрудники», если в таблице «Заказы» имеются заказы, относящиеся к этому сотруднику.

Если установлен **только флажок Обеспечение целостности данных**, то любая попытка выполнить действие, нарушающее одно из перечисленных выше правил, приведет к выводу на экран предупреждения, а **само действие выполнено не будет**.

Чтобы преодолеть ограничения на удаление или изменение связанных записей, сохраняя при этом целостность данных, следует использовать средства поддержки целостности данных – установить флажки:

- **Каскадное обновление связанных полей и**
- **Каскадное удаление связанных записей.**

Если установлен **флажок Каскадное обновление связанных полей**, то при изменении ключевого поля главной таблицы автоматически изменяются и соответствующие значения связанных записей. Microsoft Access выполняет каскадное обновление без вывода предупреждающих сообщений.

Если установлен флажок **Каскадное удаление связанных записей**, то при удалении записи в главной таблице удаляются и все связанные записи в подчиненной таблице. Если записи удаляются из таблицы или формы при установленном флажке Каскадное удаление связанных записей, Microsoft Access **выводит предупреждение о возможности удаления связанных записей**.

Если записи из таблицы удаляются с помощью **запроса на удаление** записей, то удаление выполняется автоматически **без вывода предупреждения**.

4.3. Методика формирования запросов действия

Запросы действия рекомендуется создавать в **режиме Конструктора** в следующем порядке:

1. Сформировать запрос выбора, выбрав необходимые поля и задав условия отбора.
2. Выполнить запрос выбора, проверив корректность заданных условий (**Запрос⇨Запуск**, отобразить динамический набор данных).
3. Вернуться в режим Конструктора (**Вид⇨Конструктор**).
4. Задать требуемый **вид запроса действия в меню Запрос** (рис. 4.2):
   - **Создание таблицы**,
   - **Обновление**,
   - **Добавление**,
   - **Удаление**.

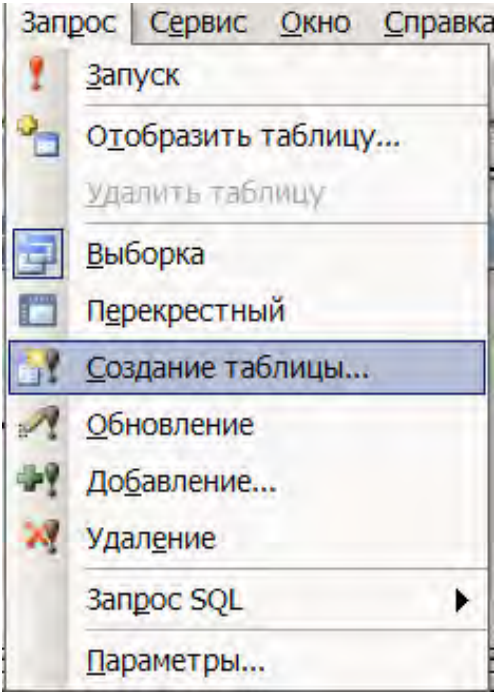

Рис. 4.2. Выбор вида запроса действия в меню Запрос

Для запроса на создание таблицы необходимо дополнительно указать имя создаваемой таблицы (и имя внешней БД, если таблица создается не в текущей БД).

5. Выполнить запрос действия (**Запрос⇨Запуск**).
6. В диалоговом окне будет указано, какие именно действия выполняются в БД по запросу, и запрошено подтверждение на выполнение.
7. Чтобы просмотреть изменения после выполнения запроса, необходимо открыть таблицы или создать запросы выбора.

4.4. Запрос на создание новой таблицы

Применяется **для архивирования** старых записей БД или **сохранения резервных копий** таблиц.

Из динамического набора формируется новая таблица − постоянный объект БД, сохраняемый на вкладке Таблицы.

Сохранить новую таблицу можно как в текущей БД, так и в новой БД (в момент выполнения запроса на создание таблицы **новая БД должна быть закрыта**).

Чтобы посмотреть **результат**, необходимо перейти в окне БД **на вкладку Таблицы** и открыть созданную таблицу.

4.5. Запрос на обновление записей

Применяется для изменения значений в полях группы записей, отобранных по заданному условию.

После выполнения действий 1–4 из п. 4.2 в Бланке запроса появится **строка Обновление**, в которую вводятся новые значения для полей.

В качестве новых значений могут выступать не только **константы**, но и **вычисляемые выражения**.

4.6. Запрос на добавление записей

Запрос используется для добавления всех или отобранных по условию записей одной таблицы к другой. Может применяться в тех случаях, когда макеты таблиц имеют некоторые различия по количеству и составу полей.

После выполнения действий 1−4 из п. 4.2 в диалоговом окне будет запрошено имя и местоположение таблицы, в которую необходимо добавить отобранные записи, а в Бланке запроса появится **строка Добавление**, в которую автоматически вводятся **имена полей** запроса, совпадающие с именами таблицы.

Можно исключить добавляемые поля из запроса, в этом случае в целевой таблице они окажутся пустыми.

4.7. Запрос на удаление записей

Запрос используется для удаления отобранных по условию записей из таблицы.

После выполнения действий 1−4 из п. 4.2, в Бланке запроса появится **строка Удаление**, в которой автоматически появляется значение «Условие».

В строку «**Условие**» можно ввести конкретное **значение** или **параметр**.

Для удаления записей, содержащих незаполненные поля, используется условие отбора Is Null.

Внимание! После выполнения запроса операцию удаления нельзя отменить.

Лекция 5. РАБОТА С ФОРМАМИ В MICROSOFT ACCESS

1. Назначение и виды форм, способы их создания.
2. Работа с данными в окне форм.
3. Работа с формами в окне Конструктора.
4. Работа с элементами управления (ЭУ).
5. Особенности создания и настройки кнопочных форм.

5.1. Назначение и виды форм, способы их создания

Форма – объект СУБД MS Access, предназначенный для наглядного отображения, ввода и редактирования данных из таблиц (табличная форма), а также для управления объектами базы данных (кнопочная форма).

Формы, как правило, соответствуют формам первичных документов — источников данных для загрузки БД. При этом

выполняется важная особенность технологии работы с базой данных — однократный ввод данных

Формы бывают:

- **табличные** (рис. 5.1)
  - **однотабличные**,
  - **многотабличные** (подчиненные и связанные);
- **кнопочные** (рис. 5.2).

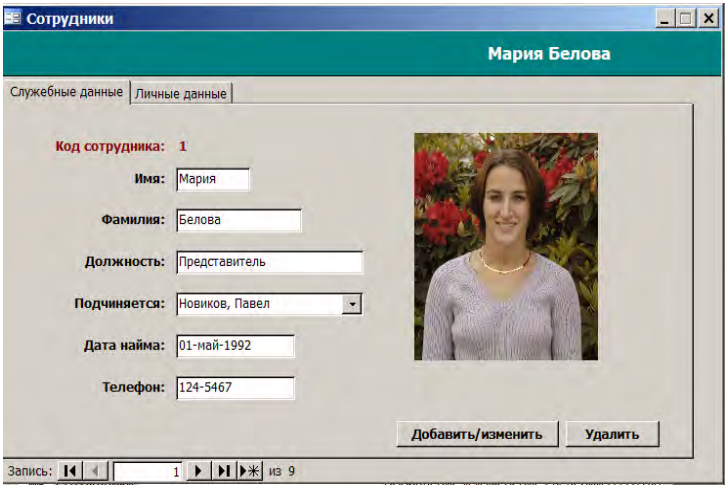

Рис. 5.1. Пример табличной формы

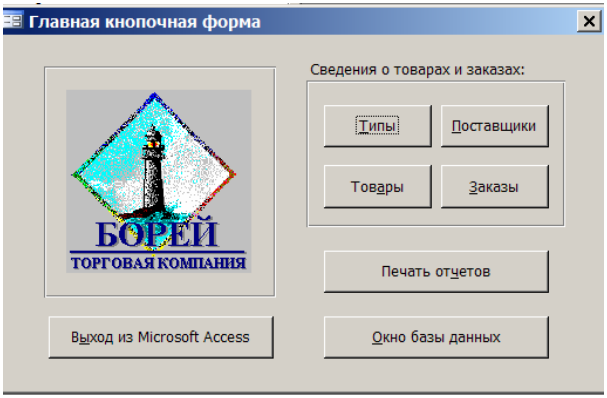

Рис. 5.2. Пример кнопочной формы

Табличная форма позволяет изменить вид отображения данных. В табличной форме можно просматривать, редактировать и удалять записи.

Форму можно создать на основе любой таблицы или запроса. Порядок следования полей в форме не обязательно совпадает с порядком полей в таблице БД.

Для создания формы, использующей данные из многих таблиц, в качестве основы используется **запрос** (например, исходные данные), включающий необходимые данные из связанных таблиц.

Способы создания формы:

1. На вкладке Таблицы или Запросы выбрать объект, для которого будет создаваться форма. На панели инструментов выбрать **кнопку Автоформа** или **Новый объект**, или выполнить команду **Вставка⇨Автоформа** (будет создана **форма в столбец**, которая включает все поля из источника).

2. На вкладке Формы нажать кнопку **Создать**. В диалоговом окне **Новая форма** выбирают источник формы (запрос или таблица). Далее указывается режим создания формы:

- **Конструктор**;
- **Мастер форм**;
- **Автоформа:**
  - **в столбец** (одна запись в столбце, подпись, значение),
  - **ленточная** (все записи в строках, но подписи только в заголовке над значениями, как таблица, но оформление другое – ширина столбцов позволяет увидеть название полей),
  - **табличная** (как таблица, все записи в строках);
- **Диаграмма**;
- **Сводная таблица**.

Режим *Автоформа* представляет самый простой путь создания формы, при котором выбирается один из стандартных макетов, в форму включаются **все** поля источника.

Название построенной Автоформы совпадает с названием источника (таблицы или запроса).

Режим *Мастер форм* позволяет использовать набор базовых макетов и их элементов, которые можно комбинировать по желанию пользователя.

Шаги *Мастера форм*:

1. Выбор режима Мастер форм и **источника формы** (запроса или таблицы БД).
2. Выбор **полей**, отображаемых в форме (из указанного источника или других доступных объектов БД, даже из нескольких таблиц).
3. Выбор **внешнего вида формы** – порядка расположения полей:
   - в один столбец,
   - ленточный,
   - табличный,
   - выровненный (поля одной записи отображаются более компактно, чем при расположении в столбец.)
4. Выбор **стиля формы** – дизайна (глобус, ель и другие из стандартного набора). Стиль может быть изменен в режиме Конструктора по команде **Формат⇨Автоформат** (изменяется текущий стандартный Стиль на основе формы).
5. Ввод **Имени формы** (по умолчанию совпадает с именем источника).

После завершения работы Мастера создается форма, в которой размещены заданные поля из таблицы-источника. Если включена подчиненная форма, то для нее выделена область, в которой указано только присвоенное ей имя.

По умолчанию созданная форма открывается в режиме Формы. Можно на последнем шаге Мастера установить флажок «Изменить макет» и перейти в режим Конструктора для редактирования макета формы.

5.2. Работа с данными в окне форм

Переключение между режимами отображения формы выполняется в **меню Вид⇨Конструктор** (Форма или Таблица) или при помощи кнопок на **ПИ**.

При работе с формой в режиме Формы используются:

1. Кнопки перехода между записями в окне формы.
2. Кнопки на ПИ или команды меню для работы с записями:

- создать новую запись (**Вставка⇨Новая запись**);
- удалить запись (**Правка⇨Удалить**);
- найти запись по ее содержимому (**Правка⇨Найти**);
- фильтры (удобны для анализа данных в табличной форме, **Записи⇨Фильтр**);
- сортировка (**Записи⇨Сортировка**);
- сохранить (**Записи⇨Сохранить запись**).

Для выбранного поля (курсор в поле) можно:

1. Применить условное форматирование **Формат⇨Условное форматирование** (формат для невыполнения условий и до 3 условий для поля).
2. Выполнить форматирование выводимых данных в поле, изменив:

- **шрифт** – размер шрифта, цвет шрифта, выравнивание данных в поле
- **внешний вид поля** – границу поля, вид оформления, цвет заливки, цвет границы.

5.3. Работа с формами в окне Конструктора

Форма в режиме Конструктора (рис. 5.3) имеет следующую структуру:

- **область данных – обязательная область**, в которой размещаются элементы управления для оформления и вывода данных;

- **верхний/нижний колонтитул** – необязательная область, содержит элементы управления, текст или данные, которые выводятся при печати формы на каждой странице. В режиме Формы колонтитулы не отображаются;
- **заголовок/примечание формы** – необязательная область, содержит элементы управления, текст или данные, которые выводятся при печати формы на первой/последней странице соответственно.

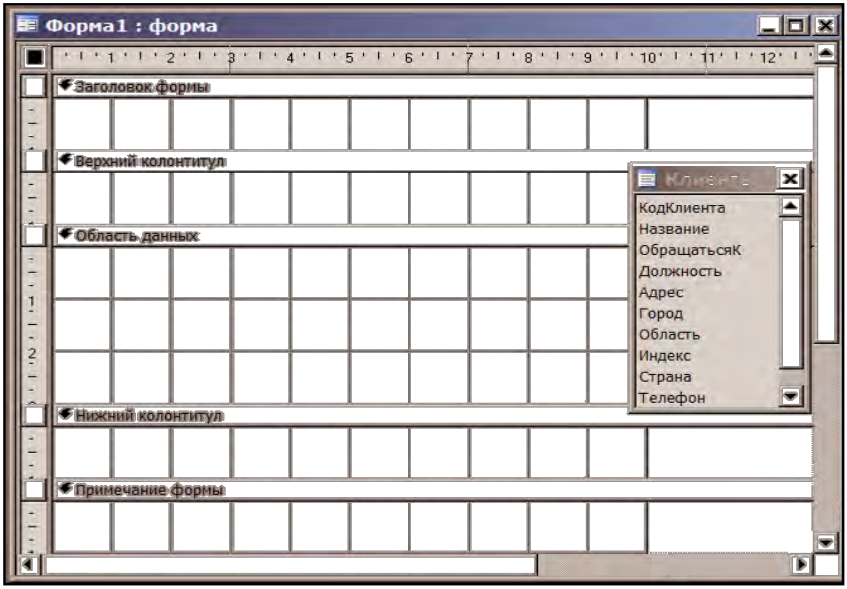

Рис. 5.3. Структура формы в режиме Конструктора

Для отображения или сокрытия указанных разделов формы (кроме области данных) используется меню **Вид**.

Работа с формой в режиме **Конструктора** связана с командами в меню **Вид**:

Свойства позволяет отобразить список свойств для выделенного объекта.

Список полей позволяет отобразить или скрыть список только тех полей или запроса, для которых проектируется форма.

Последовательность переходов позволяет установить порядок перехода при нажатии клавиш «Tab» или «Shift+Tab».

Программа отображает программы, связанные с формой.

Группа команд: Линейка, Сетка, Панель элементов – позволяет вывести на экран указанные элементы интерфейса.

Колонтитулы, Заголовок/Примечание формы добавляют соответствующие разделы формы.

В начале работы над формой рекомендуется выбрать область данных и **задать размер формы**. Работа с формой в целом заключается в ее выделении и изменение размеров.

Способы выделения формы:

1. Меню **Правка⇨Выделение формы**.
2. В левом верхнем углу макета формы значок «выделение формы».

Изменение размера – курсором мыши на границе области (курсор в виде двунаправленной стрелки).

Работа с областями формы:

1. Выделение области – щелкнуть «мышкой» «область данных».
2. Использовать клавиши «Tab» и «Shift+Tab».

Строка заголовка выделенной области затемняется.

5.4. Работа с элементами управления

Элементами управления (ЭУ) называются **объекты** в формах, отчетах или на страницах доступа к данным, используемые для оформления, отображения данных или выполнения других действий.

Все ЭУ в формах/отчетах делятся на три типа:

1. **Связанные** (присоединенные) ЭУ – связаны с данными в таблице или запросе и предназначены для вывода данных из таблиц БД.

2. **Свободные** ЭУ – независимы от данных в таблице, обычно используются для оформления.

3. **Вычисляемые** ЭУ – создаются при помощи выражений, используют для вычислений данные из таблиц.

При работе с формами необходимо отобразить на экране **Панель элементов** (**Вид⇨Панель элементов**, рис. 5.4).

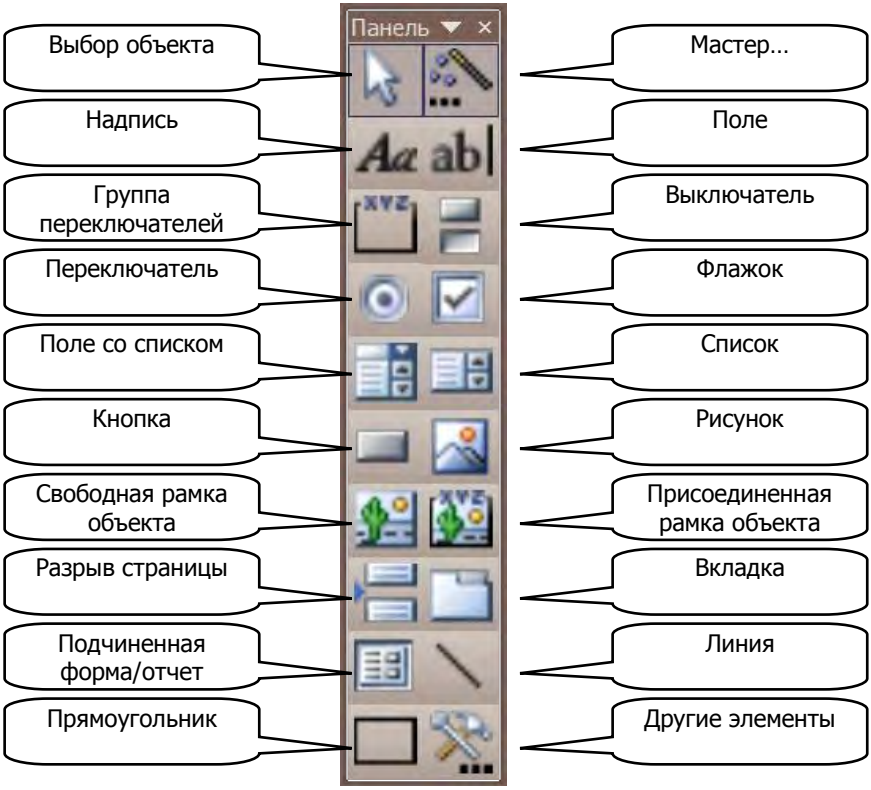

Рис. 5.4. Кнопки ЭУ Панели элементов

На Панели элементов рекомендуется активизировать следующие кнопки:

- **Выбор объекта** обеспечивает работу с ЭУ,

- **Мастер** позволяет автоматически задавать свойства для вновь создаваемых ЭУ.

Описание основных ЭУ приведено в табл. 5.1.

ЭУ Поле – добавляет в форму текстовое поле, которое используется не только для отображения данных из таблиц, но и для вывода **вычисляемых значений**.

Вычисляемое выражение задается в меню **Вид⇨Свойства⇨Данные…**. В режиме формы значения в вычисляемых полях нельзя отредактировать.

Для добавления в форму **поля (связанного ЭУ)** необходимо выполнить **Вид⇨Список полей,** выделить в списке нужное поле и перетащить его в область данных. Для выделения всех полей в списке полей необходимо выбрать ***.**

Для выделения ЭУ (смежных/несмежных) использовать ЛКМ+Shift или Ctrl (рис. 5.5).

Меню Формат в режиме Конструктора формы позволяет выполнять следующие действия над ЭУ (рис. 5.6):

1. Выровнять ЭУ.
2. Задать авторазмер.
3. Задать интервал между объектами.
4. Задать шрифт надписи (если надпись выделена).

Значения свойств для связанного ЭУ в форме могут не соответствовать значениям тех же свойств поля базовой таблицы, с которым связан ЭУ. Если значения различны, то значения в форме или отчете обычно перекрывают значения в таблице, но применимы только к текущей форме или отчету.

Рекомендуется задавать **в базовой таблице или запросе**, а не в форме или отчете, значения для свойств ЭУ: **Формат**, **Число десятичных знаков**, **Маска ввода**, **Условие на значение**, **Сообщение об ошибке** и **Значение по умолчанию**.

В этом случае можно быть уверенным, что для поля заданы нужные значения свойств независимо от того, добавляется ли оно к форме или отчету. Это обеспечивает согласованность настроек при добавлении полей в форму/отчет.

Таблица 5.1

Характеристики элементов управления

Элемент управления	Кнопка	Назначение
Надпись *Label*		Отображение произвольного текстового выражения (пояснения)
Поле *TextBox*		Ввод текстовой информации, которая может преобразовываться в числа и даты
Поле со списком *ComboBox*		Хранение списка значений, из которого пользователь может выбрать или ввести с клавиатуры только одно значение
Список *ListBox*		Хранение списка значений, из которого пользователь может выбрать одно или несколько значений
Флажок *CheckBox*		Выбор из нескольких возможных вариантов
Переключатель *OptionButton*		Выбор одного из нескольких взаимоисключающих вариантов
Рамка *Frame*		Визуальная группировка элементов управления
Кнопка *Command Button*		Выполнение некоторых действий (макросов) при нажатии кнопки
Набор вкладок *TabStrip*		Создание нескольких вкладок в диалоговом окне
Рисунок *Image*		Отображение в форме графических файлов форматов: BMP, GIF, JPG, ICO, WMF

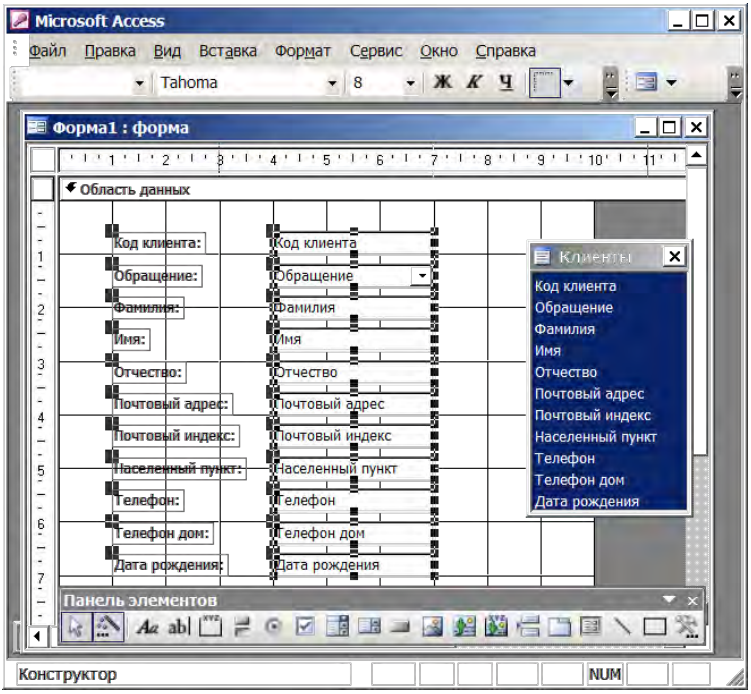

Рис. 5.5. Выделение ЭУ

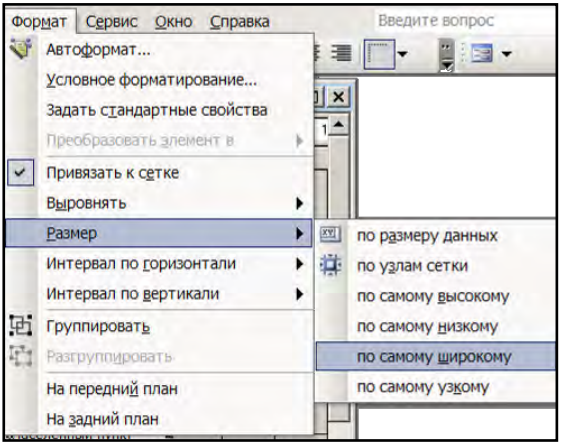

Рис. 5.6. Форматирование ЭУ

При создании присоединенных ЭУ в формах и отчетах путем перетаскивания полей из списка полей Microsoft Access копирует некоторые свойства из поля базовой таблицы или запроса для создаваемого ЭУ.

Например, если для поля «Цена» в таблице «Товары» свойство **Формат поля** имеет значение **Денежный**, а свойство **Число десятичных знаков –** значение **Авто**, то при создании присоединенного поля в форме путем перетаскивания поля «Цена» из списка полей Microsoft Access автоматически установит для указанных свойств создаваемого ЭУ такие же значения.

Свойства **Значение по умолчанию**, **Условие на значение** и **Сообщение об ошибке не наследуются**. Microsoft Access не устанавливает для этих свойств ЭУ те же значения, что и для свойств базового поля. Однако **значения этих свойств проверяются** при создании ЭУ, основанного на поле, для которого эти свойства установлены. Если свойство **Условие на значение** установлено для поля и для ЭУ, основанного на этом поле, Access проверяет оба условия на значение. Если свойство **Значение по умолчанию** установлено для поля и для ЭУ, основанного на этом поле, **свойство ЭУ подавляет свойство поля**.

В табл. 5.2 перечислены свойства, **наследуемые** для присоединенных ЭУ при перетаскивании поля из списка полей в формы или отчеты.

Существует возможность изменения значений любых наследуемых свойств в окне свойств ЭУ. Изменение значения свойства ЭУ в форме не влияет на значение этого свойства для поля в базовой таблице или запросе. Аналогично, если изменить значение свойства для поля в таблице или запросе **после создания формы** с использованием этого поля, значение **свойства ЭУ не обновляется**, обновление необходимо выполнить вручную.

Таблица 5.2

Наследуемые свойства некоторых присоединенных ЭУ

Присоединенный ЭУ	Наследуемые свойства
Поле	**Формат поля**, **Число десятичных знаков**, **Маска ввода** **Текст строки состояния** (из свойства **Описание**)
Список	Все свойства, указанные на вкладке **Подстановка** в режиме Конструктора таблиц, **Текст строки состояния** (из свойства **Описание**)
Поле со списком	Все свойства, указанные на вкладке **Подстановка** в режиме Конструктора таблиц, **Маска ввода**, **Текст строки состояния** (из свойства **Описание**)

5.5. Особенности создания и настройки кнопочных форм

Кнопочные формы являются элементами интерфейса БД и предназначены для управления объектами БД. **Главная кнопочная форма** (Панель переключений) обычно создается для того, чтобы ограничить круг задач, выполняемых рядовым пользователем в БД.

Особенность создания кнопочных форм заключается в том, что до разработки кнопочной формы должны быть созданы все объекты (таблицы, запросы, отчеты, табличные формы), к которым будет обращаться кнопочная форма.

Существуют два способа создания кнопочных форм:

- в режиме **Конструктора**;
- с помощью **Диспетчера кнопочных форм**.

Порядок создания в режиме Конструктора:

1. На вкладке Формы нажать кнопку Создать, выбрать режим **Конструктор** (не указывать источник формы).

2. Самостоятельно сформировать **макет формы** (рис. 5.7), разместить кнопки и назначить кнопкам действия (макросы, процедуры обработки событий).

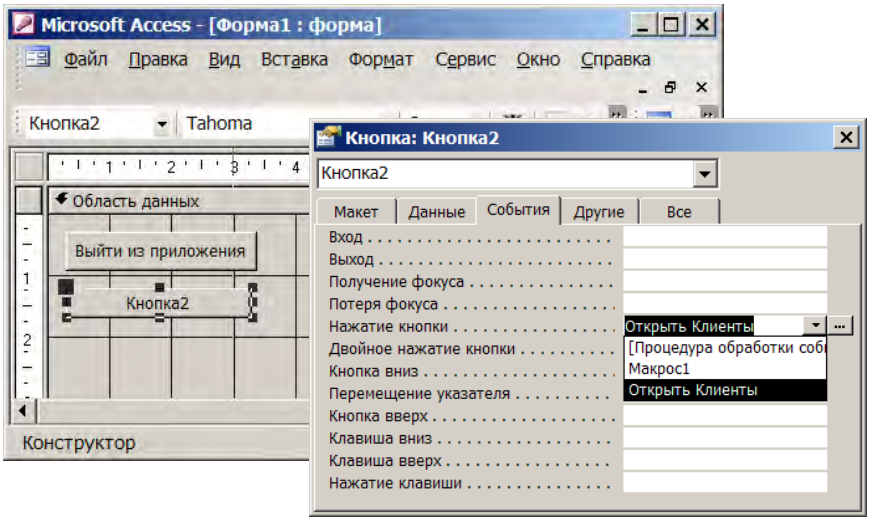

Рис. 5.7. Создание макета кнопочной формы

3. Настроить **Свойства формы** на вкладке Макет (рис. 5.8):

- задать подпись,
- разрешить режим формы (остальные режимы запретить),
- убрать полосы прокрутки,
- убрать область выделения,
- убрать кнопки перехода,
- убрать кнопки размеров окна,
- подобрать размеры формы и задать тип границы – окна диалога или тонкую.

Рис. 5.8. Свойства кнопочной формы

4. Настроить **параметры запуска формы** при открытии БД (**Сервис⇨Параметры запуска**, рис. 5.9).

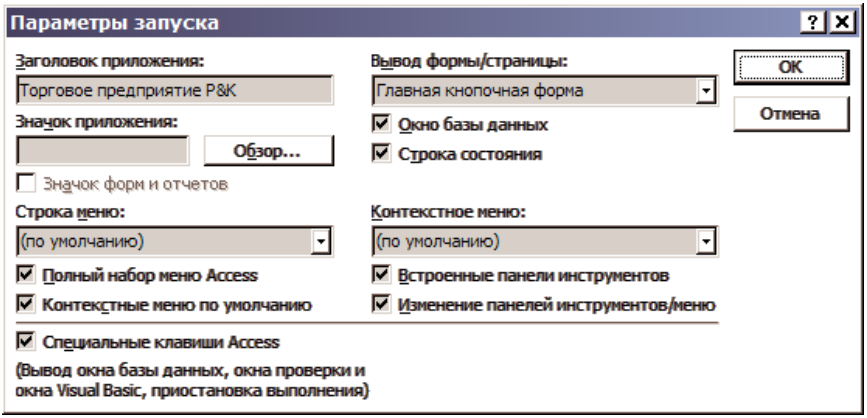

Рис. 5.9. Настройка параметров запуска БД

Диспетчер кнопочных форм вызывается в меню **Сервис⇨Служебные программы⇨Диспетчер кнопочных форм и** позволяет параллельно создавать кнопочные формы различного уровня (главные и подчиненные).

Лекция 6. РАЗРАБОТКА ОТЧЕТОВ В MICROSOFT ACCESS

1. **Общие сведения.**
2. **Создание и настройка отчета.**
3. **Просмотр отчета.**
4. **Печать отчета.**

6.1. Общие сведения

Отчет – объект СУБД MS Access, предназначенный для подготовки и вывода на печать итоговых документов в удобном для пользователя виде.

Отчет – средство организации данных для вывода на печать.

До начала работы с отчетами необходимо убедиться, что установлен драйвер принтера.

Назначение отчетов:

1. Готовить итоговые документы, группируя данные по различным признакам.
2. Создавать почтовые наклейки.
3. Представлять данные на диаграмме (Excel).
4. Представлять данные в виде сводной таблицы Excel.

Основой для создания отчета могут служить таблицы и запросы.

В MS Access предусмотрена стандартная **структура отчета** (рис. 6.1), которая предполагает наличие основной части и колонтитула.

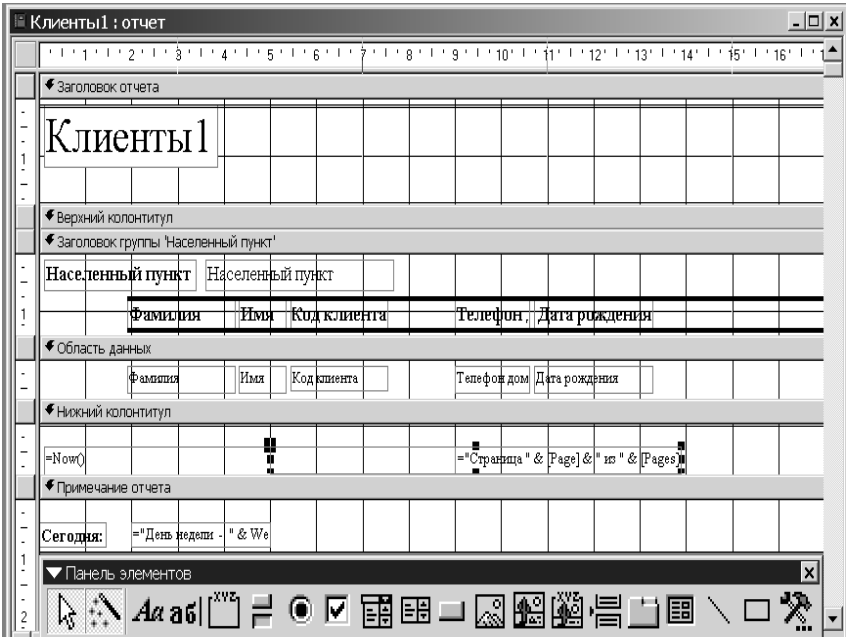

Рис. 6.1. Структура отчета в режиме Конструктора

Для создания связи между отчетом и исходными данными, так же как и в формах, используются ЭУ.

6.2. Создание и настройка отчета

Способы создания отчета:

1. **Автоотчет**, который включает все поля и записи из базовой таблицы или запроса.

2. **Мастер отчетов** – серия диалоговых окон, в которых запрашиваются необходимые данные и позволяют создавать макет отчета.

3. Создание вручную в режиме **Конструктора**.

Порядок создания Автоотчета:

1) на вкладке Отчеты нажать кнопку **Создать**.

2) в диалоговом окне «Новый отчет» указать **вид Автоотчета**:

- в столбец,
- ленточный – поля отдельной записи образуют строку,
- а также указать **источник данных** - таблицу или запрос;

3) ОК.

Создание с помощью мастера:

1) на вкладке Отчеты нажать кнопку **Создать**;
2) в диалоговом окне выбрать режим **Мастер отчетов** и указать **источник данных** – таблицу или запрос;
3) в следующем окне **указать поля**, которые должны быть отражены в отчете. Можно добавить поля из другой таблицы или запроса (выбрав из списка в этом же диалоговом окне);
4) в следующем диалоговом окне можно добавить **уровни группировки**;
5) задать **сортировку записей** (до 4 полей сортировки);
6) в следующем окне выбрать **макет** и **ориентацию страницы** (можно задать опцию «настроить ширину полей» для размещения на одной странице);
7) в следующем окне выбрать стиль (фон и элементы оформления заголовков, подписи данных элементов данных и др.);
8) на последнем шаге мастера необходимо задать **имя отчета**. Может быть выбрана опция «Просмотр» или «Изменение структуры».

После завершения работы мастера создается отчет и по умолчанию выводится на экран в режиме Предварительного просмотра. Может быть выбрана опция «Изменение структуры», которая обеспечит переход в режим Конструктора.

Создание отчета в режиме Конструктора:

1) на вкладке Отчеты нажать кнопку **Создать**;

2) в диалоговом окне выбрать режим **Конструктора** и и указать **источник данных** – таблицу или запрос, на основе которого строится отчет.

В режиме Конструктора вывести на экран список полей источника по команде из меню **Вид⇨Список полей**.

С помощью мыши перетащить необходимые поля из списка в требуемые области отчета (рис. 6.1).

Для работы с отчетом в режиме Конструктора используются Панель элементов и панели инструментов Конструктор отчетов и Формат (форма/отчет) (рис. 6.2).

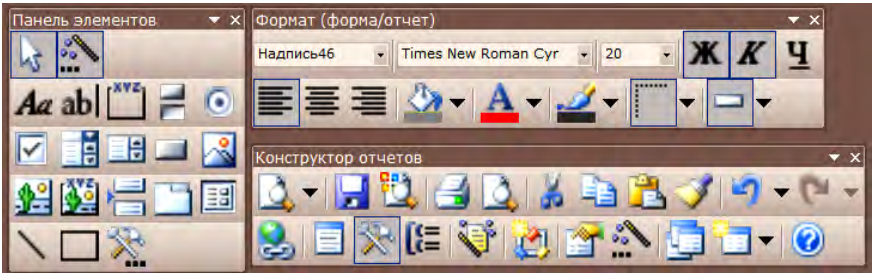

Рис. 6.2. Набор инструментов для работы с ЭУ отчета

С помощью Панели элементов можно добавлять **свободные и вычисляемые ЭУ** в отчет.

Например, можно **расположить в колонтитуле номер страницы отчета**, выполнив последовательность действий:

1) в режиме Конструктора добавить в область колонтитула ЭУ Текстовое поле;

2) для добавленного ЭУ вызвать контекстное меню, команда Свойства поля;

3) на вкладке Данные, в строке Данные вызвать Построитель выражений;

4) в Построителе выражений в категории элементов выбрать Общие выражения, в подкатегории выбрать Номер страницы и нажать кнопку Вставить (рис. 6.3).

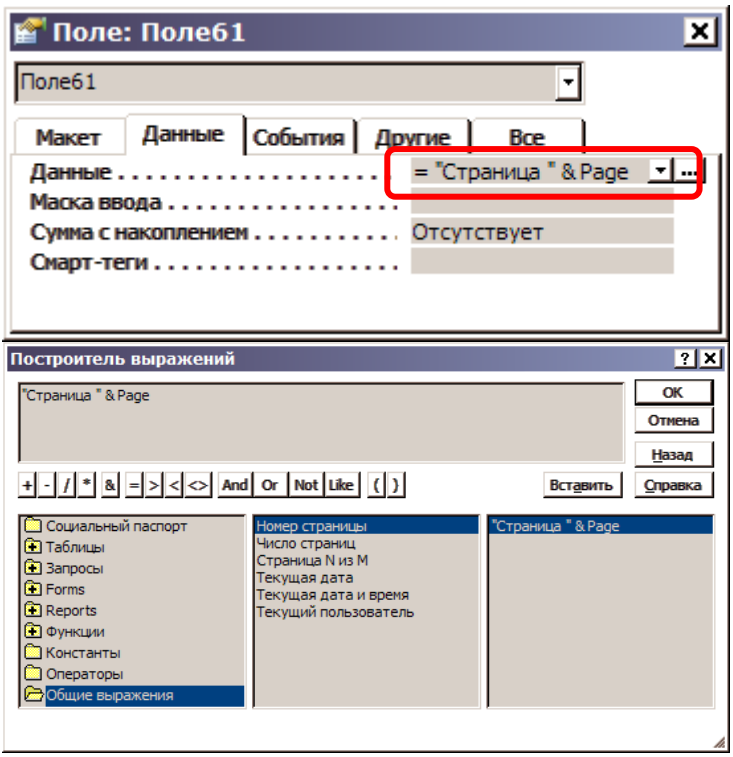

Рис. 6.3. Добавление номера страницы в отчет

Для изменения **внешнего вида ЭУ** необходимо выделить ЭУ и использовать **ПИ Формат (форма/отчет)** или контекстное меню для изменения шрифта, цвета и др.

Для изменения **формата** отображенных данных в ЭУ необходимо выделить ЭУ, вызвать контекстное меню **Свойства** и задать необходимые значения.

Для перемещения элемента управления, изменения размера и выравнивания необходимо выделить ЭУ и использовать меню **Формат** или контекстное меню.

Для изменения вида отчета можно воспользоваться кнопкой **Автоформат** на **ПИ Конструктор отчетов**.

Для изменения порядка группировки используется соответствующая кнопка на **ПИ Конструктор отчетов**.

В отчет можно добавить диаграмму (Microsoft Graph) в режиме Конструктора по команде **Вставка⇨Диаграмма**.

6.3. Просмотр отчета

Существует два режима просмотра отчета:

1) просмотр образца макета – в режиме **Конструктора** по команде **Вид⇨Образец** (рис. 6.4). При этом выводятся только поля, которые необходимы для заполнения каждого элемента;

2) просмотр полного содержимого отчета (рис. 6.5) – **Вид⇨Предварительный просмотр**.

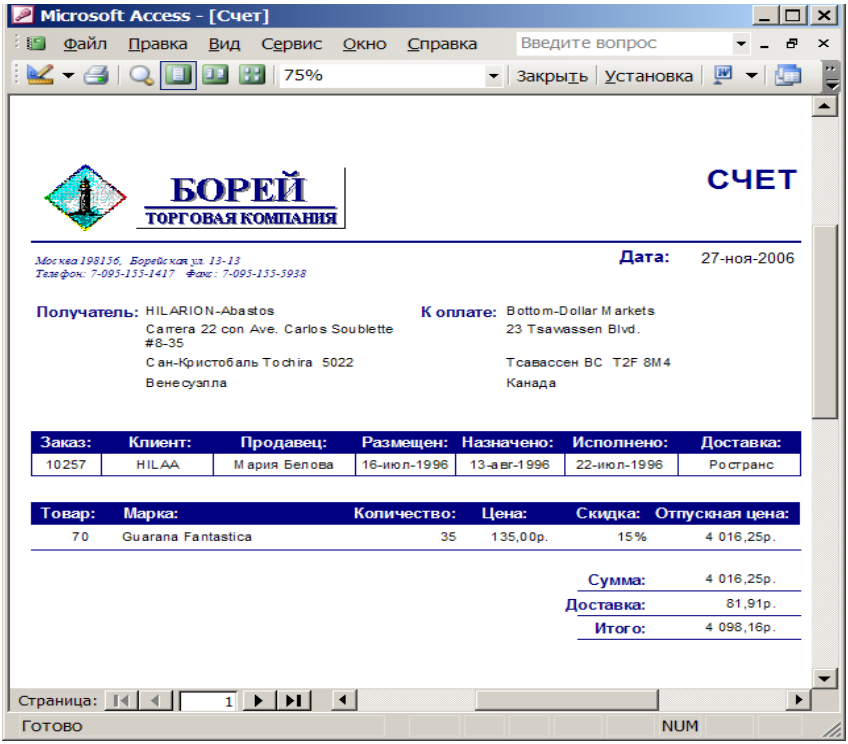

Рис. 6.4. Просмотр отчета в режиме Образец

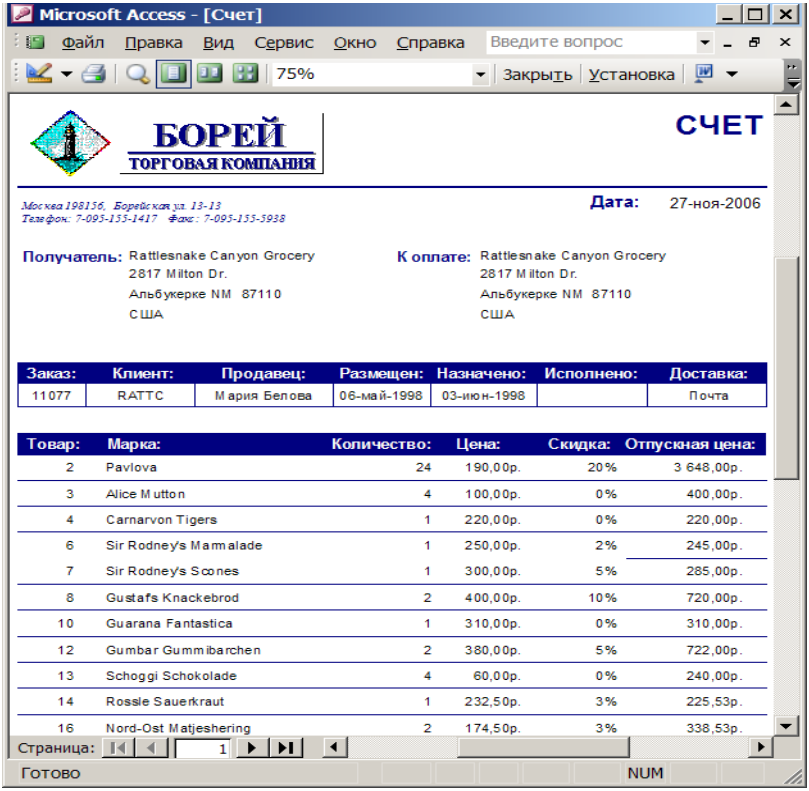

Рис. 6.5. Предварительный просмотр отчета

6.4. Печать отчета

До печати отчета обязательно задать формат и ориентацию бумаги, размеры полей в меню **Файл⇨Параметры страницы**. MS Access сохраняет настройки печати вместе с отчетом, поэтому их устанавливают один раз.

Для вывода отчета на принтер из любого окна БД или режима просмотра выполнить команду **Файл⇨Печать**.

Вывод отчета в **файл** MS Word (имя_отчета.RTF) выполняется по команде **Сервис⇨Связи с** Office⇨ **Публикация в** MS Word.

Вывод отчета в **файл снимка** (имя_отчета.SNP) можно выполнить по команде **Файл⇨Экспорт**, выбрать **тип файла Снимок.**snp. В результате будет создан файл с расширением .SNP, содержащий графическую копию каждой страницы отчета MS Access с высокой точностью воспроизведения (двумерного макета, графики и других внедренных объектов).

Лекция 7. СПЕЦИАЛЬНЫЕ ПРИЕМЫ РАБОТЫ С СУБД MICROSOFT ACCESS

1. Работа с макросами.
2. Взаимодействие MS Access с приложениями MS Office.
3. Использование механизма Слияния в MS Office Word для подготовки рассылки.
4. Утилиты MS Access.
5. Настройка среды MS Access.

7.1. Работа с макросами

Макрос – последовательность инструкций (макрокоманд) на языке Visual Basic, которая записывается единожды и исполняется многократно.

Макрос используется для выполнения часто повторяющихся операций и для повышения эффективности работы с БД.

В приложениях MS Office макрос обычно создается путем записи последовательных нажатий клавиш. В MS Access используется принципиально другой способ: макросы заносятся в специальный список, а выполняются либо из окна

БД (вкладка Макросы), либо назначаются другим объектам БД (например, кнопкам).

Вкладка Макросы имеет 3 кнопки:

- Запуск (вместо Открыть),
- Конструктор,
- Создать.

Создание Макроса выполняется в специальном окне Макрос (рис. 7.1), включающем по умолчанию три области.

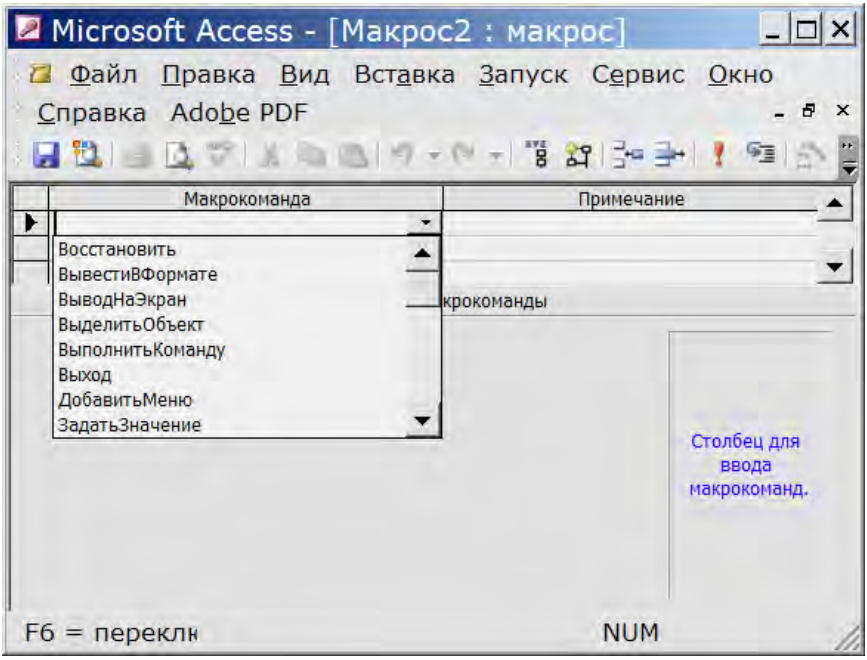

Рис. 7.1. Окно Конструктора макроса

Макрокоманда: каждая располагается на отдельной строке в порядке выполнения и выбирается из стандартного списка.

Примечание: краткое описание Макроса (отображается при вызове КЗМ для макроса).

Аргументы Макрокоманды: Эта область становится доступной только после выбора макрокоманды из списка.

В списке аргументов указывают, к какому объекту применить макрос, и задают условия выполнения действий.

Большинство макрокоманд требует **задать значения аргументов** (рис. 7.2).

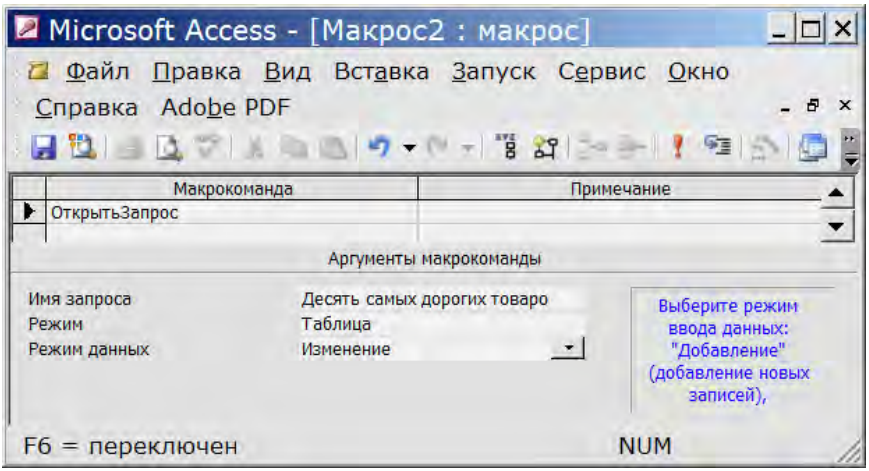

Рис. 7.2. Пример макрокоманды с аргументами

После закрытия окна Конструктора необходимо задать **имя макроса**. По умолчанию Макрос 1…

Например,

для создания макроса Открыть_Таблицу_Клиенты, необходимо в режиме Конструктора заполнить поля:

Макрокоманда:

1) **ОткрытьТаблицу** (выбрать из списка) и задать **Аргументы**:

- ***Имя*** – Клиенты,
- ***Режим*** – Таблица (конструктор, просмотр),
- ***Режим данных*** – Изменение (добавление, только чтение);

2) **Сигнал** (без аргументов);

3) **Восстановить** (без аргументов).

После записи макрос можно редактировать как любой другой объект: удалять и добавлять строки с макрокомандами.

Выполнение макроса может осуществляться одним из следующих способов:

1) на вкладке Макрос окна БД выполнить двойной щелчок или нажать кнопку Запуск;

2) из командного меню: **Сервис⇨Запуск макроса** (из списка в диалоговом окне выбрать имя Макроса);

3) при разработке в окне Макрос нажать кнопку ! на ПИ или вызвать меню **Макрос⇨Запуск**;

4) из формы или отчета (при открытии или закрытии, если назначить макрос соответствующей команды на **вкладке Событие** для выбранной области или элемента управления).

Например, для текстового поля можно назначить Макрос для свойств **Вход** и/или **Выход** (выполняется при выборе поля или отмене);

5) при помощи ЭУ **Кнопка** в форме

При использовании ЭУ **Кнопка**, можно для свойства **Нажатие кнопки** задать макрос;

6) при открытии БД автоматически выполняется макрос, сохраненный под именем AutoExec.

Например, при работе с БД необходимо регулярно выполнять следующие действия:

1) открыть таблицу Клиенты в режиме Таблица;
2) развернуть окно во весь экран;
3) переместить курсор к новой записи.

В этом случае целесообразно создать макрос AutoExec.

Для этого необходимо в режиме Конструктора макросов в области **Макрокоманда** последовательно выбрать команды:

1. **ОткрытьТаблицу**, задать аргументы **Имя**, **Режим и Режим данных**.
2. **Развернуть**.
3. **Следующая Запись**.
4. **На запись**, задать аргументы

- тип объекта: Таблица,
- имя объекта: Клиенты,
- запись: Новая.

5. **Файл ⇨Сохранить и** задать имя макроса AutoExec.

Чтобы проверить работоспособность созданного макроса AutoExec, необходимо заново открыть БД. Следует помнить, что при открытии БД сначала выполняются **параметры запуска** (из меню **Сервис⇨Параметры запуска**), а затем – макрос AutoExec (поэтому между ними не должно быть противоречий).

Чтобы **отменить** автоматическое выполнение при открытии БД можно **переименовать или удалить макрос** AutoExec.

Чтобы **пропустить** выполнение макроса AutoExec при открытии БД, необходимо удерживать клавишу Shift.

Дополнительные возможности при создании макроса:

1. Можно сформировать Макрогруппу и присвоить имена макросам, входящим в нее.
2. Можно добавить в макрос макрокоманды, выполняемые по условию.
3. Можно выполнить отладку макроса по шагам (для обнаружения ошибок, в специальном окне) **Макрос⇨По шагам**.

Создание Макрогрупп полезно, когда макрос назначается кнопкам. Добавить в окно Конструктора столбец Имя макроса можно по команде **Вид⇨Имена Макросов** или кнопка «Имена макросов».

При использовании **Макрогрупп** на вкладке Макрос появляются имена Макрогрупп. Макрогруппе присваивается имя окна макроса, например, AutoExec.

Для использования условных Макрокоманд необходимо добавить **столбец Условие** в окно Конструктора макроса по команде **Вид⇨Условие** (рис. 7.3). Если в столбце Условие

введено выражение, то Макрокоманда выполняется только тогда, когда Условие Истинно.

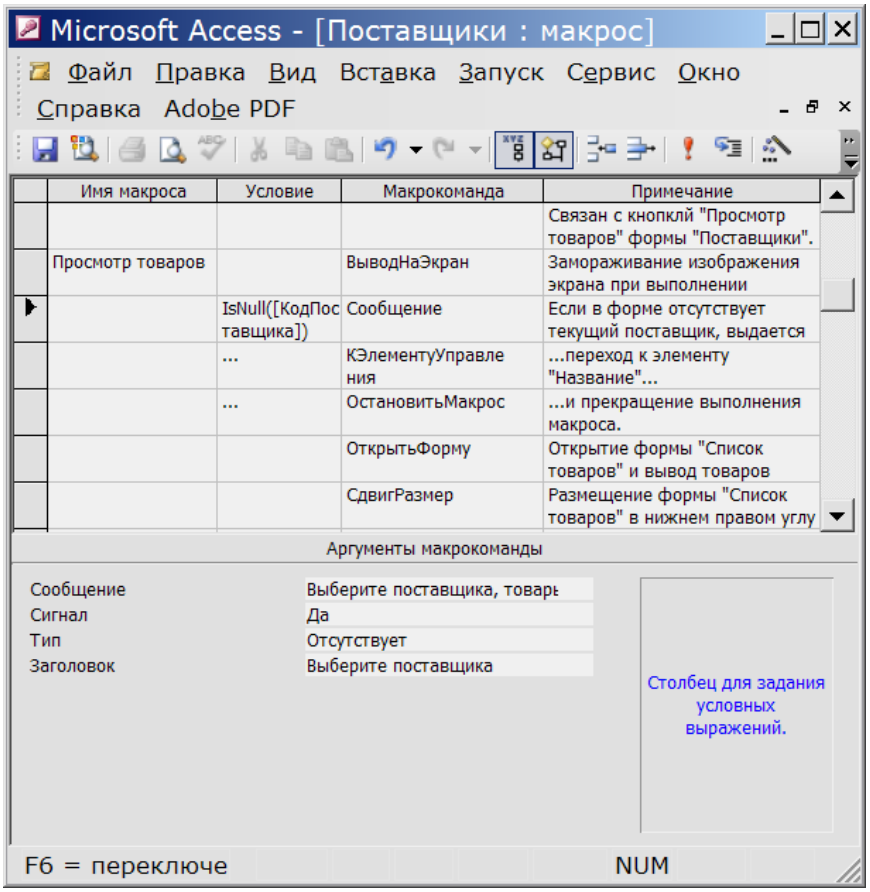

Рис. 7.3. Окно Конструктора макросов с макрогруппой и условием

7.2. Взаимодействие MS Access с приложениями MS Office

Взаимодействие MS Access с приложениями MS Office обеспечивают:

1. **Буфер обмена**.
2. **Технология** OLE (тип данных – Поле объекта OLE).
3. **Технология** ODBC – набор функций, позволяющих обрабатывать информацию из БД на SQL-серверах без установки MS Access (MS Query).
4. **Команды меню Сервис⇨Связи с** Office:
   - **Слияние в** MS Word (для таблиц и запросов);
   - **Публикация в** MS Word (для таблиц, запросов, форм и отчетов);
   - **Анализ в** MS Excel (для таблиц, запросов, форм и отчетов).

Документы, полученные по команде **Сервис⇨Связи с** Office, по умолчанию сохраняются в папке **Мои Документы**.

В MS Access существует возможность **конвертирования данных** – преобразования из одного офисного формата представления данных в другой.

Для конвертирования используется меню:

1) **Файл⇨Внешние данные⇨Импорт**;

2) **Файл⇨Экспорт**.

Набор доступных форматов зависит от выбранного объекта БД и параметров установки MS Access (рис. 7.4).

7.3. Использование механизма Слияния в MS Office Word для подготовки рассылки

Слияние – стандартный механизм приложений MS Office, который используется:

- для создания документов на бланке,
- для подготовки почтовых наклеек и конвертов,

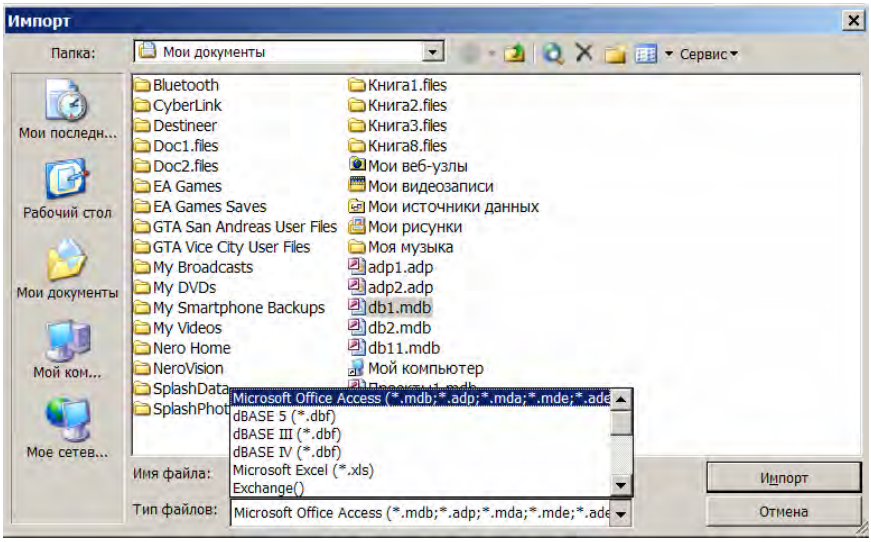

Рис. 7.4. Набор доступных форматов для импорта

- для создания каталогов,
- а также при массовой рассылке сообщений по факсу или по электронной почте.

В процессе слияния различают три типа документов: основной документ (документ на бланке), источник данных и документ слияния.

Основной документ содержит:

- ***постоянную часть*** (текст письма), данные, неизменные во всех производных документах при слиянии (например, обратный адрес отправителя или собственно текст письма),
- ***переменную часть*** – реквизиты получателя как поля слияния.

Источник данных – файл, содержащий сведения, предназначенные для объединения с основным документом (например, список имен и адресов получателей из таблицы

Word, Excel или Access). Для использования сведений из источника данных необходимо к нему подключиться.

Документ с результатом слияния – многостраничный документ на бланке, почтовая наклейка, конверт или каталог для каждой строки (записи) из источника данных.

Для подготовки серийного письма, на базе таблицы или запроса MS Access необходимо выполнить действия:

1. Выделить источник слияния в БД на вкладке Таблицы или Запросы.
2. Выполнить команду **Сервис⇨Связи с Office⇨Слияние в MS Office Word**.
3. Выбрать вариант слияния (рис. 7.5).

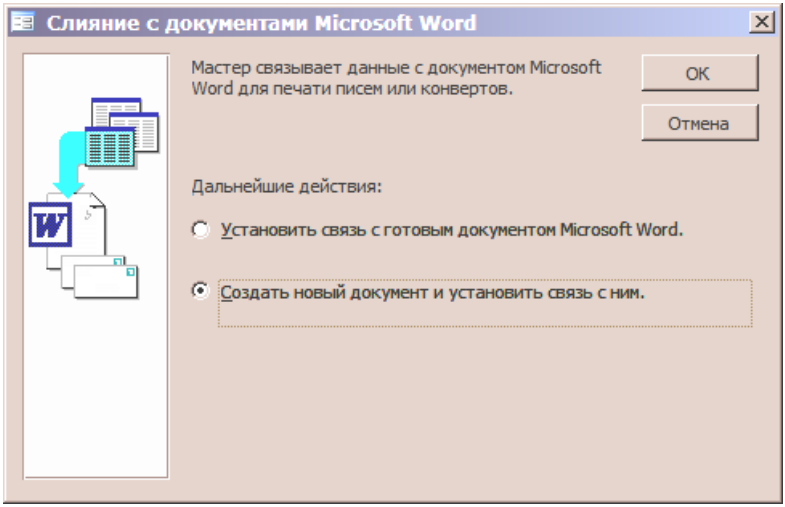

Рис. 7.5. Выбор варианта Слияния

4. В MS Word создать или отредактировать образец письма для рассылки (постоянную часть).
5. Добавить поля слияния из источника данных (с помощью ПИ Слияние, рис. 7.6).

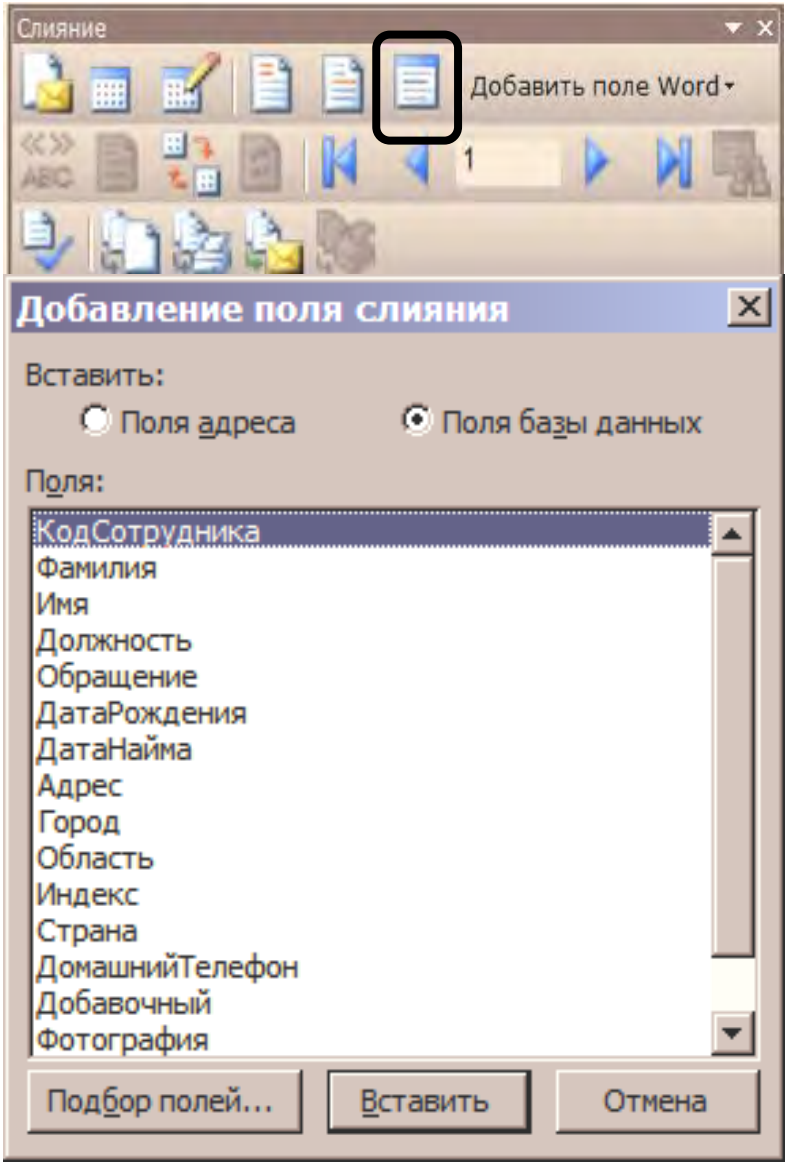

Рис. 7.6. Добавление полей Слияния

6. При необходимости добавить поля слияния MS Word (например, If … Then … Else, рис. 7.7).

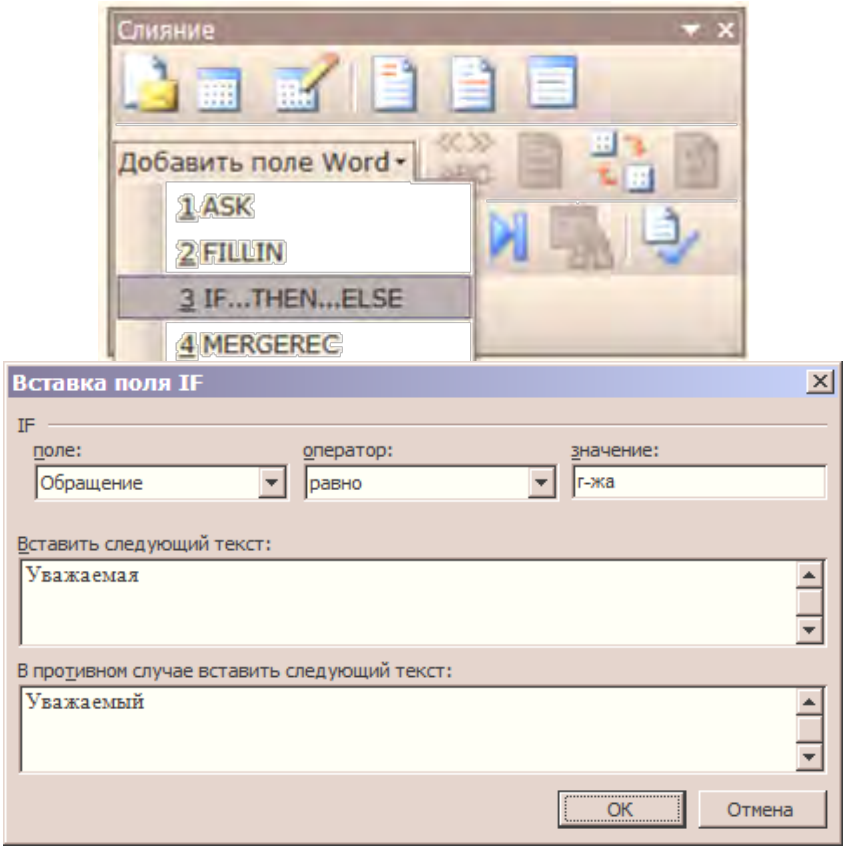

Рис. 7.7. Добавление полей MS Word

7. Проверить полученный документ (рис. 7.8).
8. Завершить создание рассылки (рис. 7.9).

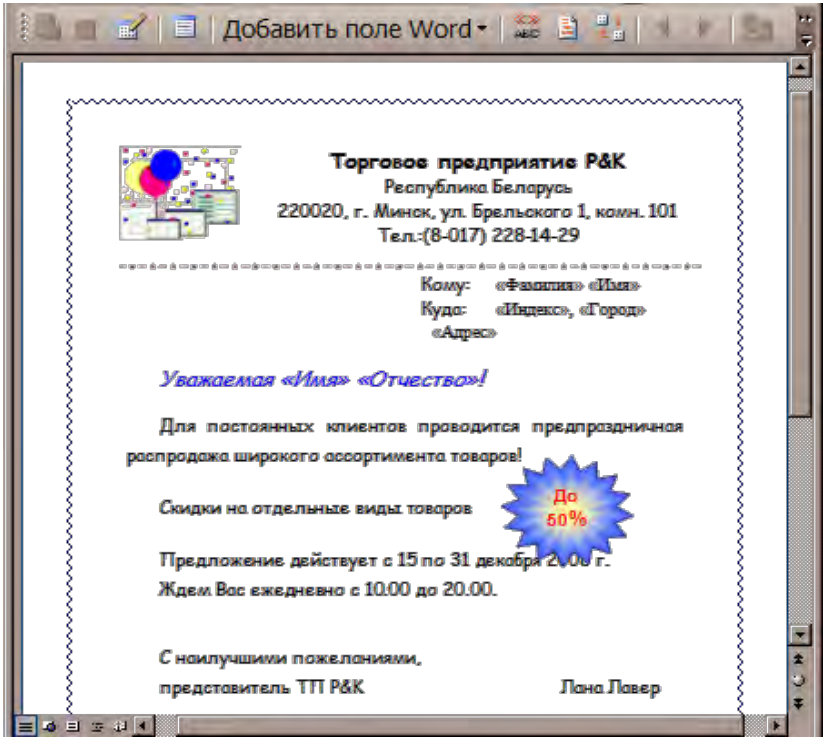

Рис. 7.8. Основной документ с полями слияния

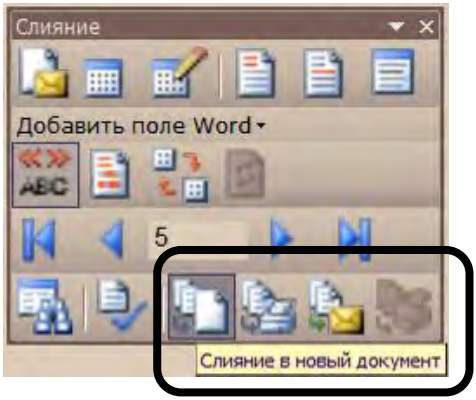

Рис. 7.9. Завершение процесса слияния

7.4. Утилиты MS Access

Утилиты – служебные программы MS Access, расширяющие возможности пользователя по настройке, управлению и обслуживанию БД. По функциональному признаку утилиты можно разделить на две группы: для работы с объектами БД и работы с БД.

1. **Для работы с объектами БД** (доступны из меню Правка, Вид, Вставка):

- **Правка** – позволяет переименовать, удалить, вырезать, копировать, вставить выбранный объект БД;
- **Вид** – позволяет изменить представление и упорядочивание объектов на вкладках БД;
- **Вставка** – позволяет запустить режим создания объекта (равносильно нажатию кнопки Создать на вкладке объекта БД).

2. **Для работы с БД** (как правило, доступны из меню **Сервис**):

- **преобразование** БД;
- **анализ БД** (таблиц, быстродействия, Архивариус – подготовка описания объектов БД к печати);
- **защита** (с помощью пароля или шифрования);
- **настройка** (среды и параметров запуска).

Утилиты для преобразования БД обеспечивают совместимость разных версий БД и вызываются в меню **Сервис⇨ Служебные программы…** (рис. 7.10).

При операции преобразования обычно используются два диалоговых окна: одно – для выбора исходного файла БД, другое – для файла назначения (желательно задать другое имя).

Чтобы избежать ошибок и потери БД при преобразовании, нельзя указывать одно и то же имя для новой (преобразованной) БД.

Лучше удалить старую БД после успешного выполнения преобразования.

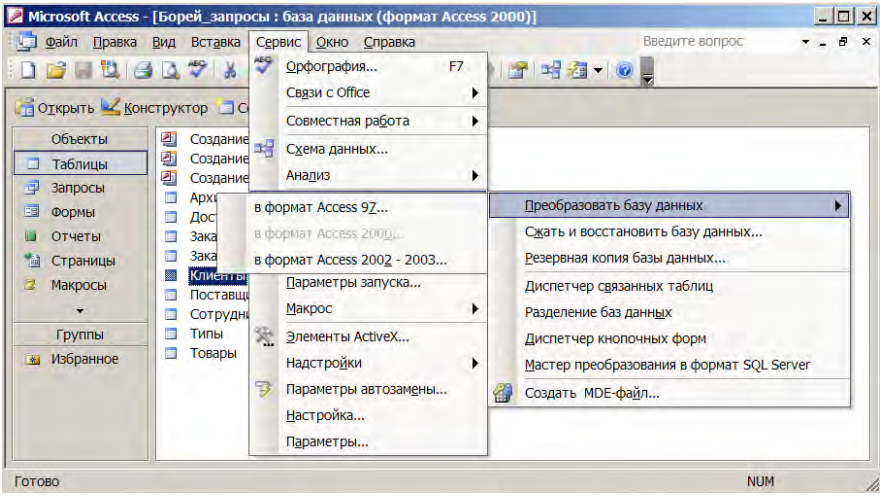

Рис. 7.10. Утилиты для работы с БД

Утилита Сжать и восстановить БД используется, если необходимо физически сжать БД: удалить свободные участки (удаленные объекты) и уменьшить размер файла. При работе с БД, особенно на этапе конструирования объектов, размер БД постоянно возрастает и даже при удалении некоторых объектов не уменьшается. Это связано с тем, что объекты физически остаются в файле БД, т.к. их реальное удаление из файла и сокращение объема файла требуют физического перемещения всех объектов в файле БД.

Для выполнения этой операции необходимо:

- закрыть БД;
- вызвать утилиту Сжать и восстановить БД;
- выбрать файл БД для сжатия и имя файла назначения (можно задать одно и то же имя, но для безопасной работы можно задать другое имя для сжатой БД).

Эта же утилита выполняет восстановление поврежденной БД при неустойчивой работе БД (например, при аварийном завершении работы с Access). При открытии поврежденной БД

обычно выполняется автоматический запрос на восстановление.

Весьма полезной утилитой для работы с БД является **Мастер преобразования в формат** SQL-**сервер**.

Большинство приложений баз данных со временем разрастается, и число их пользователей также растет. В этом случае целесообразно выполнить преобразование БД Microsoft Access в формат Microsoft SQL Server с целью оптимизации производительности, масштабируемости, безопасности, надежности, способности к восстановлению и доступности базы данных и приложения.

Такое преобразование представляет собой перенос некоторых или всех объектов базы данных из базы данных Microsoft Access (.mdb) в новую или существующую базу данных Microsoft SQL Server или новый проект Microsoft Access (.adp).

7.5. Настройка среды MS Access

Настройка рабочего места СУБД MS Access выполняется с использованием меню **Вид** и **Сервис**.

В меню **Вид** выполняется управление панелями инструментов.

В меню **Сервис⇨Параметры** выполняется настройка параметров на соответствующих вкладках (рис. 7.11).

В меню **Сервис⇨Параметры запуска** выполняется настройка параметров приложения:

- **заголовок** приложения и **значок** приложения;
- **строка меню** (может быть создана собственная, ограничивающая задачи пользователя);
- **вывод формы/страницы** (сначала открывается эта форма, а затем выполняется макрос AutoExec, если был создан).

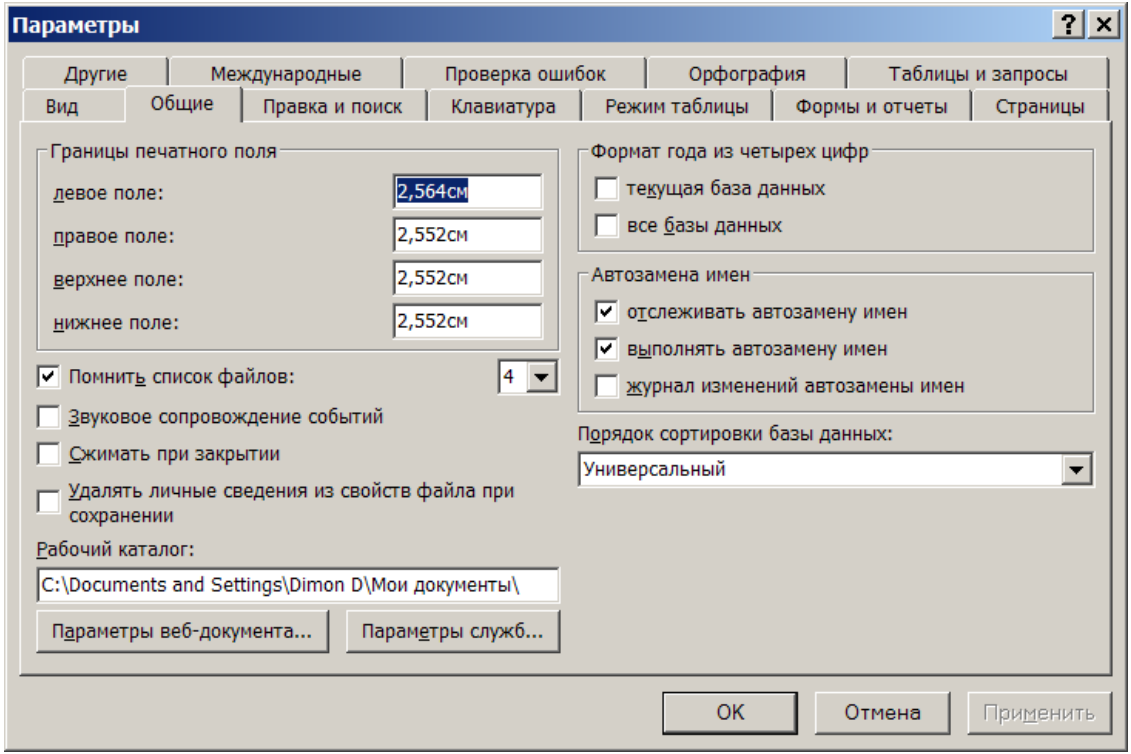

Рис. 7.11. Настройка параметров среды СУБД MS Access (вкладка Общие)

Литература

1. Экономическая информатика: учебник для вузов / под ред. В.В. Евдокимова. – СПб.: Питер Паблишинг, 1997. – 592 с.

2. Информатика: Базовый курс: учебник для вузов / под ред. С.В. Симоновича. – СПб.: Питер, 2001. – 640 с.

3. Основы экономической информатики: учебное пособие / А.Н. Морозевич [и др.]; под общ. ред. А.Н. Морозевича. – Минск: БГЭУ, 1998. – 438 с.

4. Конноли, Т. Базы данных Проектирование, реализация и сопровождение. Теория и практика / Т. Конноли. – М.: Изд. дом «Вильямс». 2000. – 1120 с.

5. Дженнингс, Р. Использование Microsoft Access 2002. Специальное издание / Р. Дженнингс. – М.: Изд. дом «Вильямс», 2002. – 1012 с.

6. Дженнингс, Р. Эффективная работа с Microsoft Access. / Р. Дженнингс. – СПб.: Питер, 1999.

7. Михеева, В.Д. Microsoft Access 2002 / В.Д. Михеева, И.А. Харитонова. – СПб.: BHV – Санкт-Петербург, 2002. – 1008 с.

Учебное издание

ЛАВРЕНОВА Ольга Анатольевна

СЕТЕВЫЕ ТЕХНОЛОГИИ И БАЗЫ ДАННЫХ

Курс лекций
для студентов специальности 1-27 01 01
«Экономика и организация производства»

В 2 частях

Ч а с т ь 1

ОСНОВЫ РАБОТЫ С РЕЛЯЦИОННЫМИ БАЗАМИ ДАННЫХ И СУБД

Редактор Л.Н. Шалаева
Компьютерная верстка О.А. Лавреновой

Подписано в печать 09.09.2009.
Формат 60×84 1/16. Бумага офсетная.
Отпечатано на ризографе. Гарнитура Таймс.
Усл. печ. л. 6,28. Уч.-изд. л. 4,96. Тираж 100. Заказ 1269.

Издатель и полиграфическое исполнение:
Белорусский национальный технический университет.
ЛИ № 02330/0494349 от 16.03.2009.
Проспект Независимости, 65. 220013, Минск.