МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ САМАРСКОЙ ОБЛАСТИ
ГОУ СПО «Усольский сельскохозяйственный колледж»

Методическое пособие MS Access

Система управления Базами данных (СУБД)

Тема: Создание проекта «Автогараж»

Усолье 2017 г.

Рассмотрено
На заседании цикловой комиссии математических и естественно-научных дисциплин
Протокол №____
От «___»_____2005г.
_____

Утверждаю
Зам. директора по учебной работе
_____

Автор: Чебаков Юрий Владимирович – преподаватель дисциплины «Информационные технологии в профессиональной деятельности»

Рецензент:

Учебное пособие разработано в помощь студентам дисциплины «Информационные технологии в профессиональной деятельности» для проведения практических работ по теме: «СУБД Microsoft Access», а также для самостоятельного узучения.

Базы данных

Хранение информации — одна из важнейших функций компьютера. Одним из распространенных средств такого хранения являются базы данных. *База данных* — это файл специального формата, содержащий информацию, структурированную заданным образом.

Структура базы данных

Большинство баз данных имеют *табличную структуру.* Как мы знаем, в табличной структуре адрес данных определяется пересечением строк и столбцов. В базах данных столбцы называются *тлями,* а строки — *записями.* Поля образуют *структуру базы данных,* а записи составляют информацию, которая в ней содержится.

Для того чтобы легко усвоить понятие структуры базы данных, надо представить себе пустую базу, в которой пока еще нет никаких данных. Несмотря на то, что данных в базе нет, информация в ней все-таки есть. Это структура базы, то есть набор полей. Они определяют, что будет записано в эту базу и в каком виде.

Простейшие базы данных

Простейшие базы можно создавать, не прибегая к специальным программным средствам. Чтобы файл считался базой данных, информация в нем должна иметь структуру (поля) и быть форматирована так, чтобы содержимое соседних полей легко различалось. Простейшие базы можно создавать даже в текстовом редакторе Блокнот, то есть обычный текстовый файл при определенном форматировании тоже может считаться базой данных.

Существует по крайней мере два формата текстовых баз данных:

- с заданным разделителем;
- с фиксированной длиной поля.

Несмотря на «примитивность» таких текстовых баз данных, мощные системы управления базами данных позволяют импортировать подобные файлы и преобразовывать их в «настоящие» базы данных, Поэтому если в организации пока нет системы управления базами данных, данные можно хранить в текстовом файле, а потом, когда такая система появится, данные не пропадут и будут успешно импортированы.

Таблицы баз данных легко сортируются. Эта таблица отсортирована по поля Код сотрудника. Вид сортировки в порядке возрастания.

Свойства полей. Типы полей

Поля — это основные элементы структуры базы данных. Они обладают *свойствами.* От свойств полей зависит, какие типы данных можно вносить в поле, а какие нет, а также то, что можно делать с данными, содержащимися в поле.

Например, данные, содержащиеся в поле Цена, можно просуммировать, чтобы определить итоговый результат. Суммировать данные, содержащиеся в поле Номер телефона, совершенно бессмысленно, даже если номера телефонов записаны цифрами. Очевидно, что эти поля обладают разными свойствами и относятся к разным типам.

Основным свойством любого поля является его длина. Длина поля выражается в *символах* или, что то же самое, в *знаках.* От длины поля зависит, сколько информации в нем может поместиться. Мы знаем, что символы *кодируются* одним или двумя байтами, поэтому можно условно считать, что длина поля измеряется в байтах.

Очевидным уникальным свойством любого поля является его *Имя.* Разумеется, одна база данных не может иметь двух полей с одинаковым именем, поскольку компьютер запутается, в их содержимом. Но кроме имени у поля есть еще свойство *Подпись.* Подпись — это та информация, которая отображается в заголовке столбца. Ее не надо путать с именем поля, хотя если подпись не задана, то в заголовке отображается имя поля. Разным полям, например, можно задать одинаковые подписи. Это не помешает работе компьютера, поскольку поля при том по-прежнему сохраняют разные имена.

Разные типы полей имеют разное назначение и разные свойства.

1. Основное свойство *текстового поля* — размер.
2. *Числовое поле* служит для ввода числовых данных. Оно тоже **имеет** размер, но числовые поля бывают разными, например для ввода *целых чисел* и для ввода *действительных чисел.* В последнем случае кроме размера поля задается также размер десятичной части числа.
3. Поля для ввода дат или времени имеют тип *Дата/время.* Для ввода логических данных, имеющих только два значения (Да или Нет; 0 или 1; Истина или Ложь и т. п.), служит специальный тип — *Логическое поле.* Нетрудно догадаться, что длина такого поля всегда равна 1 байту, поскольку этого более **чем** достаточно, чтобы выразить логическое значение.
4. Особый тип поля — *Денежный.* Из названия ясно, какие данные в нем хранят. Денежные суммы можно хранить и в числовом поле, но в денежном формате с ними удобнее работать. В этом случае компьютер изображает числа вместе с денежными единицами, различает рубли и копейки, фунты и пенсы, доллары и центы, в общем, обращается с ними элегантнее.
5. В современных базах данных можно хранить не только числа и буквы, но и картинки, музыкальные клипы и видеозаписи. Поле для таких объектов называется *полем объекта OLE.*
6. У текстового поля есть недостаток, связанный с тем, что оно имеет ограниченный размер (не более 256 символов). Если нужно вставить в поле длинный текст, для этого служит поле типа *MEMO.* В нем можно хранить до 65 535 символов. Особенность поля MEMO состоит в том, что реально эти данные хранятся не в поле, а в другом месте, а в поле хранится только указатель на то, где расположен текст.
7. Очень интересно поле *Счетчик.* На первый взгляд это обычное числовое поле, но оно имеет свойство автоматического наращивания. Если в базе есть такое поле, то при вводе новой записи в него автоматически вводится число, на единицу большее, чем значение того же поля в предыдущей записи. Это поле удобно для нумерации записей.

Связанные таблицы

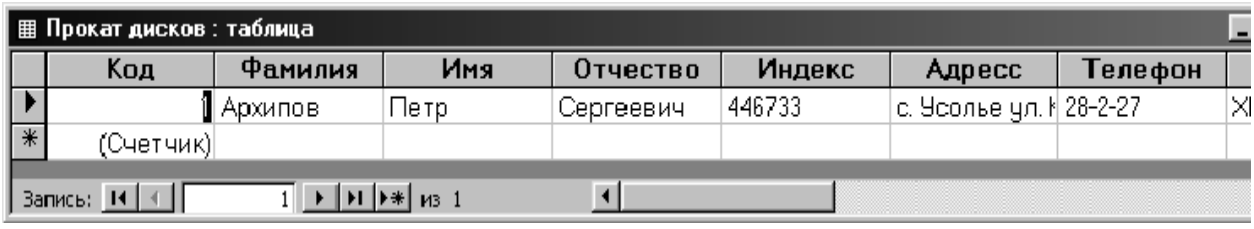
Пример, который мы привели выше, можно считать простейшими базами данных, но на самом деле это не совсем базы, а только таблицы. Если бы информация хранилась в таких простых структурах, то для работы с ней можно было бы обойтись без специальных *систем управления базами данных.* На практике приходится иметь дело с более сложными структурами, которые образованы из многих *связанных таблиц.*

Базы данных, имеющие связанные таблицы, называют также реляционными базами данных.

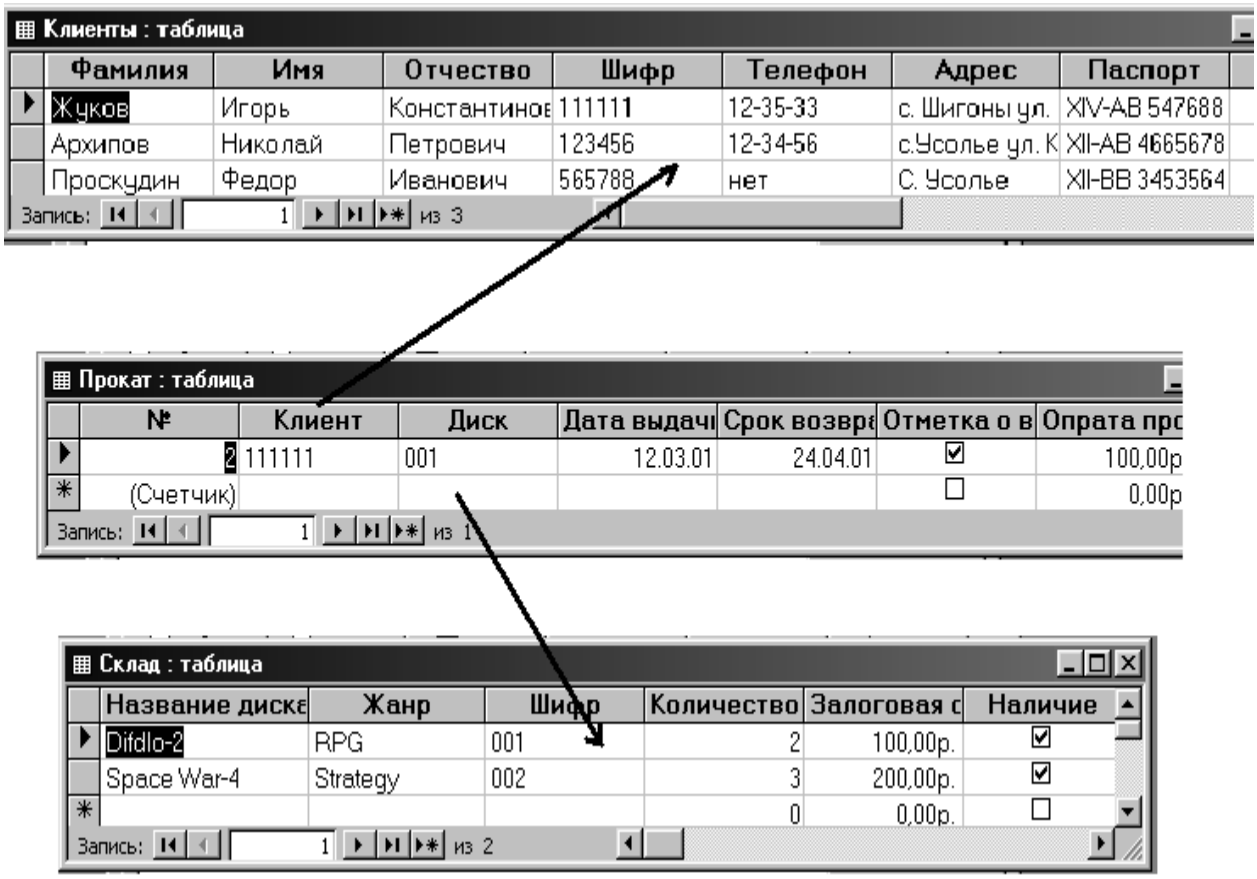
Рассмотрим пример работы малого предприятия, занимающегося прокатом компакт-дисков с компьютерными играми. Для того чтобы знать, кто какой диск взял, когда должен возвратить и сколько дисков каждого наименования осталось на складе, предприятию необходима база данных. Но если все сведения о покупателях и о дисках хранить в одной таблице, то таблица станет очень неудобной для работы. В ней начнутся повторы данных. Всякий раз когда гражданин Новиков В. П. будет брать очередной диск, придется вписывать его домашний адрес, телефон и паспортные данные. Так никто не работает. Это долго, трудно и чревато многочисленными ошибками.

Гораздо удобнее сделать несколько таблиц. В одной хранить сведения о клиентах со всеми их паспортными данными, в другой — сведения о выданных дисках, чтобы в любой момент узнать, что выдано клиенту и когда наступает срок возврата, а в третьей таблице — остаток дисков на складе, чтобы вовремя пополнять запасы. После этого отдельные поля таблиц *связывают.* Если из таблицы Прокат известно, что клиент НВП взял диск D001, то система управления базой данных мгновенно найдет в таблице Клиенты все паспортные данные этого человека, а в таблице Склад все данные об этом диске.

Разделение базы на связанные таблицы не только удобно, но иногда и необходимо. Например, для увеличения числа заказов менеджер фирмы, занимающейся прокатом компакт-дисков, решил поставить в общем зале компьютер, на котором каждый клиент может просмотреть список имеющихся дисков с иллюстрациями из игр. Если база состоит только из одной таблицы, то вместе с информацией о дисках случайный посетитель получит доступ к информации о других клиентах фирмы. Вряд ли это понравится заказчикам. Такой менеджер не только не приобретет новых клиентов, но и растеряет тех, которых имел.



Если данные в разных записях начинают повторяться, это может говорить о том, что база имеет плохую структуру. Надо подумать о том, нельзя ли разбить таблицу на группу связанных таблиц.



Если заданы связи между таблицами, то работать с разными таблицами можно, как с одной цельной базой данных.

Поля уникальные и ключевые

Создание базы данных всегда начинается с разработки структуры **ее** таблиц. Структура должна быть такой, чтобы при работе с базой требовалось вводить в нее как можно меньше данных. Если ввод каких-то данных приходится повторять неоднократно, базу делают из нескольких связанных таблиц. Структуру каждой таблицы разрабатывают отдельно.

Для того чтобы связи между таблицами работали надежно и по записи из одной таблицы можно было однозначно найти записи в другой таблице, надо предусмотреть в таблице *уникальные* поля.

Уникальное поле — это поле, значения в котором не могут повторяться.

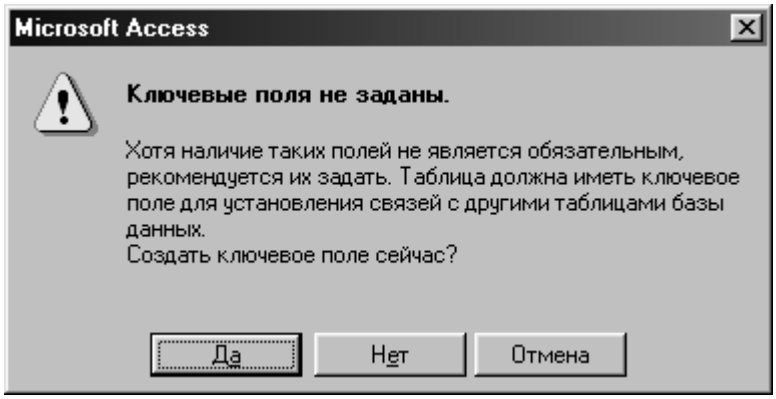
Если из таблицы Прокат известно, что клиент Новиков просрочил возврат взятого диска, то он должен уплатить штраф. Но в таблице Клиенты фирмы может быть несколько разных Новиковых, и компьютер не разберется, кто же из них должен платить штраф. Это означает, что поле Фамилия не является уникальным и потому его нельзя использовать для связи между таблицами.

Поле номера телефона — более удачный кандидат на звание *уникального поля,* но, как вы понимаете, и одним телефоном могут пользоваться несколько разных людей.

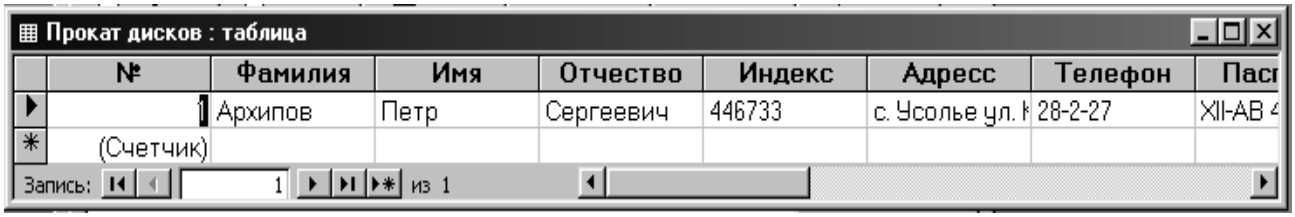
Если ни одно поле таблицы не приемлемо в качестве уникального, его можно создать искусственно. В нашем примере в таблице Клиенты фирмы создано поле Шифр. Его и использовали для связи между таблицами.

Скорее всего, поле Шифр окажется уникальным, и проблем **со** связями между таблицами не возникнет, но было бы неплохо, если бы компьютер мог просигнализировать в том случае, если вдруг записи в этом поле повторятся. Для этого существует понятие *ключевое поле.* При создании структуры таблиц одно поле (или одну комбинацию полей)

можно назначить ключевым. С ключевыми полями компьютер работает особо. Он проверяет **их** уникальность и быстрее выполняет сортировку по таким полям. Ключевое поле — очевидный кандидат для создания связей. Иногда ключевое поле называют *первичным ключом.*

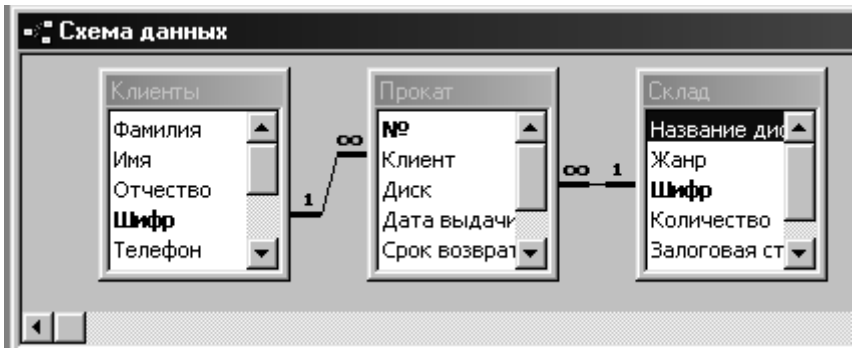


Если при создании таблицы автор не задал ключевое поле, система управления базой данных вежливо напомнит о том, что поле первичного ключа таблице не помешает.



В качестве первичного ключа в таблицах часто используют поле, имеющее тип Счетчик. Ввести два одинаковых значения в такое поле нельзя по определению, поскольку приращение значения поля производится автоматически.

Структура связей между таблицами называется *Схемой данных*



СУБД Access 9x

Системы управления базами данных (СУБД) — это программные средства, с помощью которых можно создавать базы данных, наполнять их и работать с ними. В мире существует немало различных систем управления базами данных. Многие из них на самом деле являются не законченными продуктами, а специализированными языками

программирования, с помощью которых каждый, освоивший данный язык, может сам создавать такие структуры, какие ему удобны, и вводить в них необходимые элементы управления. К подобным языкам относятся Clipper, Paradox, FoxPro и **Другие.**

В литературе можно, например, встретить как утверждение о том. что FoxPro — это язык программирования, так и утверждение, что это система управления базами данных. Надо понимать, что в последнем случае речь идет о СУБД, написанной с помощью средств языка FoxPro.

Необходимость программировать всегда сдерживала широкое внедрение баз данных в малом бизнесе. Крупные предприятия могли позволить себе сделать заказ на программирование специализированной системы «под себя». Малым предприятиям зачастую не по силам было не только решить, но даже и правильно сформулировать эту задачу.

Положение изменилось с появлением в составе пакета Microsoft Office системы управления базами данных Access. Ранние версии этой программы имели номера Access 2,0 и Access 95. Последняя версия — Access 97 входит в состав пакета Office 97. Далее мы будем говорить о программе Access 9x, имея в виду общие свойства разных версий.

С помощью Access 9x обычные пользователи получили удобное средство для создания и эксплуатации достаточно мощных баз данных без необходимости что-либо программировать. В то же время работа с Access 9x не исключает возможности программирования. При желании систему можно развивать и настраивать собственными силами. Для этого надо владеть основами программирования на языке Visual Basic.

Еще одним дополнительным достоинством Access 9x является интегрированность этой программы с Excel 9x, Word 9x и другими программами пакета Office 9x. Данные, созданные в разных приложениях, входящих в этот пакет, легко импортируются и экспортируются из одного приложения в другое.

Объекты Access 9x

Исходное окно Access 9x отличается простотой и лаконичностью. Шесть вкладок этого окна представляют шесть видов объектов, е которыми работает программа.

Таблицы — основные объекты базы данных. С ними мы уже знакомы. В них хранятся данные. Реляционная база данных может иметь много взаимосвязанных таблиц.

Запросы — это специальные структуры, предназначенные для обработки данных базы. С помощью запросов данные упорядочивают, фильтруют, отбирают, изменяют, объединяют, то есть обрабатывают.

Формы — это объекты, с помощью которых в базу вводят новые данные или просматривают имеющиеся,

Отчеты — это формы «наоборот». С их помощью данные выдают на принтер в удобном и наглядном виде.

Макросы — **это** *Макрокоманды.* Если какие-то операции с базой производятся особенно часто, имеет смысл сгруппировать несколько команд в один макрос и назначить его выделенной комбинации клавиш.

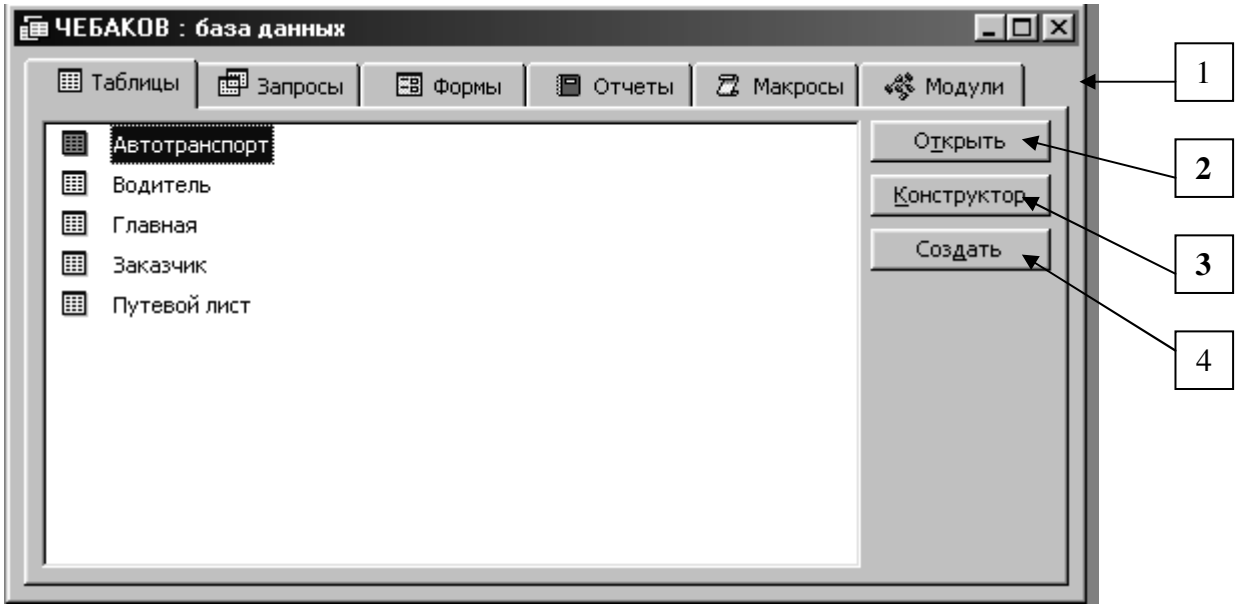
Модули — это программные процедуры, написанные на языке Visual Basic. Если стандартных средств Access не хватает для удовлетворения особо изощренных требований заказчика, программист может расширить возможности системы, написав для этого необходимые модули.

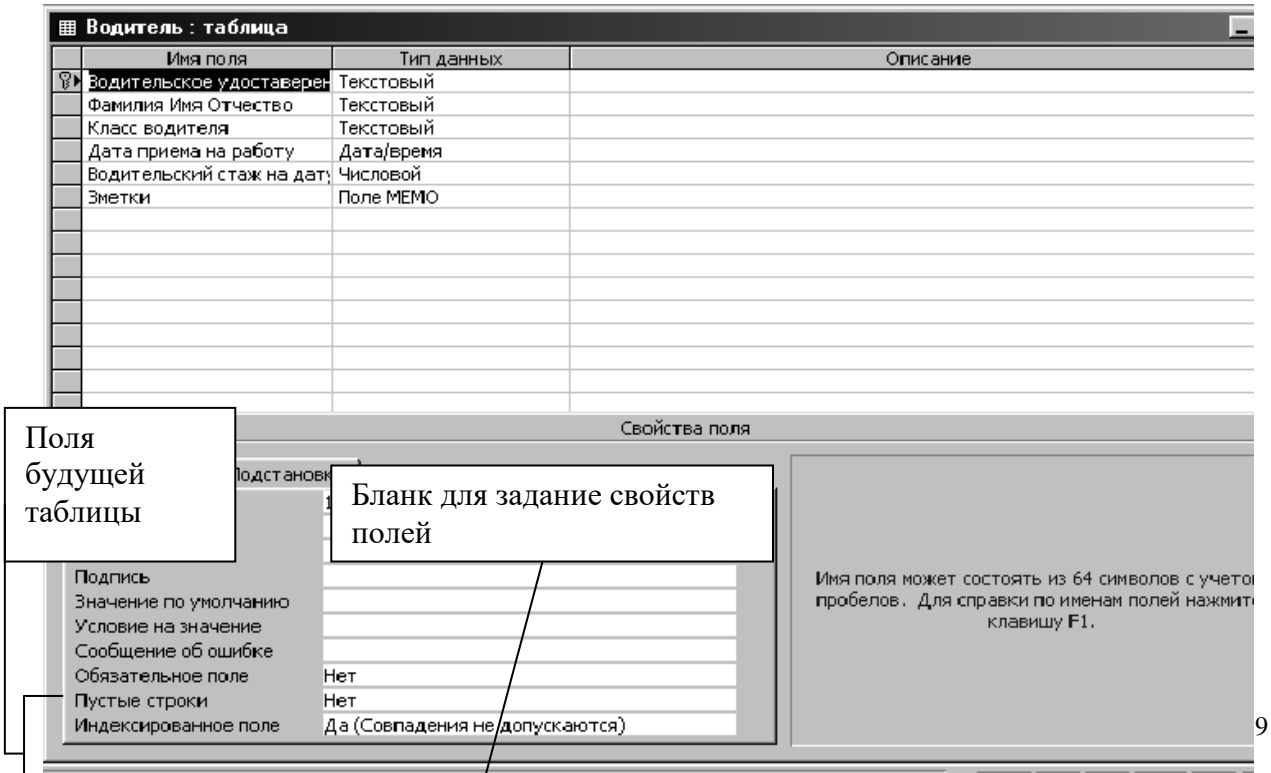
режимы работы с Access 9x

С организационной точки зрения в работе с любой базой данных есть два разных режима: *проектировочный* и *эксплуатационный* (пользовательский). Создатель базы имеет право создавать в ней новые объекты (например таблицы), задавать их структуру, менять свойства полей, устанавливать необходимые связи. Он работает со *структурой базы* и имеет полный доступ к базе. У одной базы может быть один, два или несколько разработчиков.

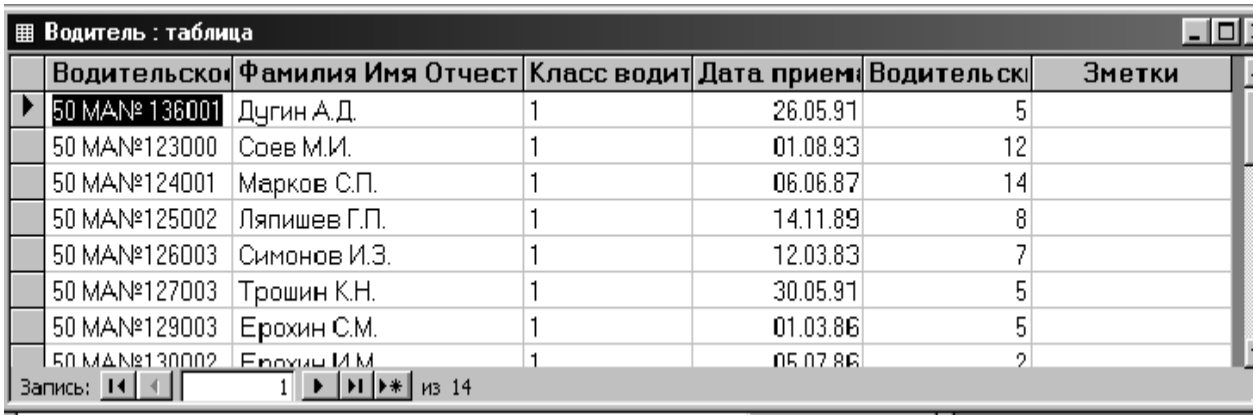
Пользователь базы — это лицо, которое наполняет ее информацией с помощью форм, обрабатывает данные с помощью запросов и получает результат в виде *результирующих таблиц* или *отчетов.* У одной базы могут быть миллионы пользователей, и, конечно, доступ к структуре базы для них закрыт.

1. Взгляните на стартовое окно базы данных. Кроме шести вкладок для основных объектов оно содержит три командные кнопки:
   Открыть, Конструктор, Создать. С их помощью и выбирается режим работы с базой.
2. Кнопка Открыть открывает избранный объект. Если это таблица, то ее можно просмотреть, внести новые записи или изменить те, что были внесены ранее.
3. Кнопка Конструктор тоже открывает избранный объект, но по-другому. Она открывает его структуру и позволяет править не содержимое, а устройство. Если это таблица, в нее можно вводить новые поля или изменять свойства существующих полей. Если это форма, в ней можно изменять или создавать *элементы управления.* Очевидно, что этот режим служит не для пользователей базы, а для ее разработчиков.
4. Действие командной кнопки Создать соответствует ее названию. Она служит для создания новых объектов. Этот элемент управления тоже предназначен для проектировщиков базы. Таблицы, запросы, формы и отчеты можно создавать несколькими разными способами: автоматически, вручную или с помощью Мастера. О достоинствах и недостатках этих методов мы поговорим при более подробном рассмотрении объектов Access 9x.





Новая таблица в режиме конструктор



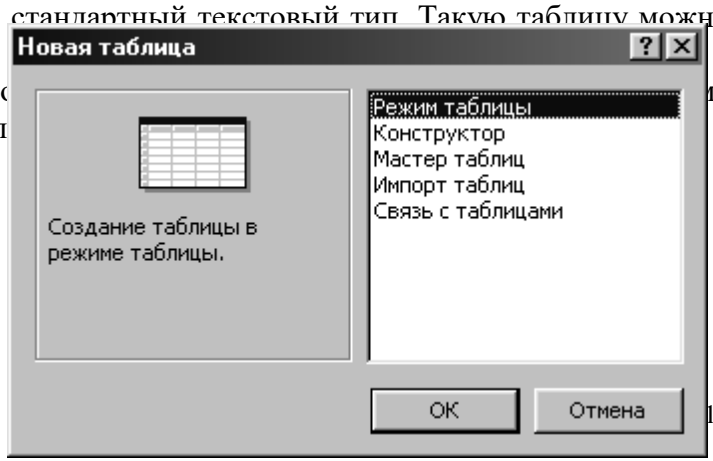
Та же таблица в режиме просмотра.

Таблицы. Создание таблиц.

Таблицы — основные объекты базы данных. Без запросов, форм, отчетов и прочего можно обойтись, но если нет таблиц, то данные ^некуда записывать, а значит, нет и базы. Создание базы начинается с создания первой таблицы.

Создание таблицы состоит в задании ее полей и назначении их свойств. Оно начинается с щелчка на кнопке Создать в окне База данных.

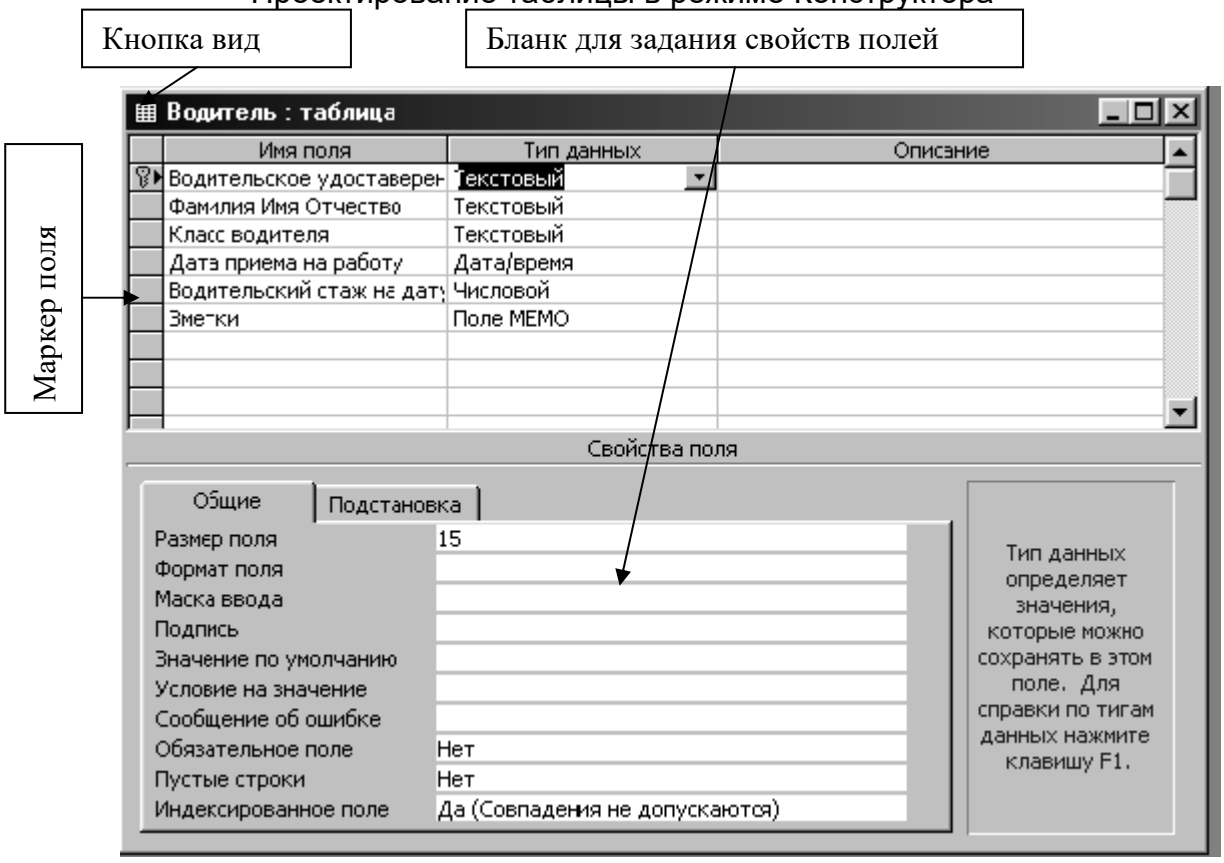
1. Есть несколько способов создания новой таблицы, отличающихся уровнем автоматизации.
2. Самый «автоматичный» способ состоит в импорте таблиц **из** другой базы, может быть, даже созданной в другой системе. В зависимости от обстоятельств из импортируемой таблицы может поступить структура полей, их названия и свойства, а также и содержимое базы. Если что-то импортируется не совсем так, как надо, необходимые правки (например в свойства полей) вносят вручную.
3. В тех случаях, когда речь идет о чужой таблице, которая находится на удаленном сервере и которую нельзя импортировать целиком, пользуются режимом Связь с таблицами. Это напоминает подключение к таблице для совместного использования ее данных.
4. Опытные разработчики пользуются Мастером таблиц. Это программа, ускоряющая создание структуры таблицы. Мастер задает ряд вопросов и, руководствуясь полученными ответами, создает структуру таблицы автоматически. Несмотря на то что этот режим служит для упрощения работы, начинающим пользоваться им не рекомендуется, поскольку, не владея всей терминологией, легко запутаться в вопросах и ответах. Первые таблицы стоит попробовать создать вручную.
5. Пункт Режим таблицы открывает заготовку, в которой все поля имеют формальные имена: Поле1, Поле2... и т. д. и один стандартный текстовый тип. Такую таблицу можно сразу наполнять информацией.
6. Наиболее универсальный ручной мето ие можно самостоятельно задать имена п



Возможные режимы создания новой таблицы.

Простейшая таблица со стандартными полями

Проектирование таблицы в режиме Конструктора



Для изменения свойств полей надо перейти в режим Конструктор щелчком на кнопке Вид. Чтобы вставить новое поле, надо установить указатель мыши на маркер поля и нажать клавишу INSERT. Чтобы удалить поле, его надо выделить и нажать клавишу DELETE. Закончив создание структуры, можно щелкнуть на кнопке Вид и перейти в Режим таблицы для заполнения ее данными.

Особенности таблиц баз данных

Прежде чем мы приступим к изучению приемов работы с таблицами баз данных, надо обратить внимание на одну особенность всех баз данных, связанную с сохранением информации. Тех, кто привык работать с другими классами программ, она поначалу обескураживает.

Обычно с документом в программах можно делать все что угодно, пока не настала пора его сохранять. Испортив неаккуратными действиями исходный документ, можно отказаться от сохранения и вернуться к работе с прежней копией. В базах данных это не

так.

Таблицы баз данных не являются самостоятельными документами. Сама база — это документ. Ей соответствует файл на диске, и мы можем сделать его копию. Структура таблиц — тоже документ. В некоторых системах она имеет отдельный файл, а в некоторых (например в Access 9x) такого файла нет, но структура таблиц входит в состав общего файла базы данных наряду с запросами, формами, отчетами и другими объектами. При изменении структуры таблицы система управления базой данных всегда выдает запрос на сохранение изменений.

Но содержание таблиц — это совсем другое дело. Его нельзя сохранить принудительной командой или, наоборот, отказаться от его сохранения. Все изменения в таблицах сохраняются автоматически *в режиме реального времени.* Режим реального времени означает, что, пока мы работаем с таблицей, происходит ее непрерывное сохранение. Как только заканчивается ввод данных в одно поле и происходит переход к следующему полю, данные немедленно записываются на жесткий диск.

Профессионалы высоко ценят эту особенность систем управления базами данных, а начинающих она иногда вводит в заблуждение. Экспериментируя с таблицами, надо знать, что все изменения, которые вносятся в их содержание, имеют необратимый характер. Нельзя что-то изменить, удалить, а потом отказаться от сохранения и вернуться к исходному варианту.

Эта особенность систем управления базами данных требует аккуратного отношения к работе с таблицами. Для экспериментов надо создавать отдельные копии базы или таблиц и работать с ними.

Надежность и безопасность баз данных

Надежность баз данных имеет особую важность. Последствия утраты документа, созданного в текстовом процессоре или графическом редакторе, можно оценить затратами времени, необходимого для его воспроизведения. Утрата базы данных может привести к остановке целой отрасли промышленности и иметь глобальные последствия. Существуют базы данных, от которых зависит движение транспорта, работа банков и промышленных предприятий. Есть базы, содержащие жизненно важные сведения медицинского характера.

Создатели систем управления базами данных не могут полагаться на то, что конкретный пользователь не забудет своевременно дать команду Сохранить. Они учитывают и то, что во время работы может произойти аварийное отключение электричества. Ни при каких условиях информация не должна теряться, поэтому все изменения данных немедленно и автоматически сохраняются на диске.

Совместное использование данных

Системы управления базами данных должны учитывать, что с базами могут одновременно работать много людей. Если бы с базами работали как с документами в текстовом процессоре, то один человек, открывший файл для редактирования, *монополизировал* бы этот файл и *блокировал* бы к нему доступ других пользователей до тех пор, пока файл не будет закрыт и сохранен.

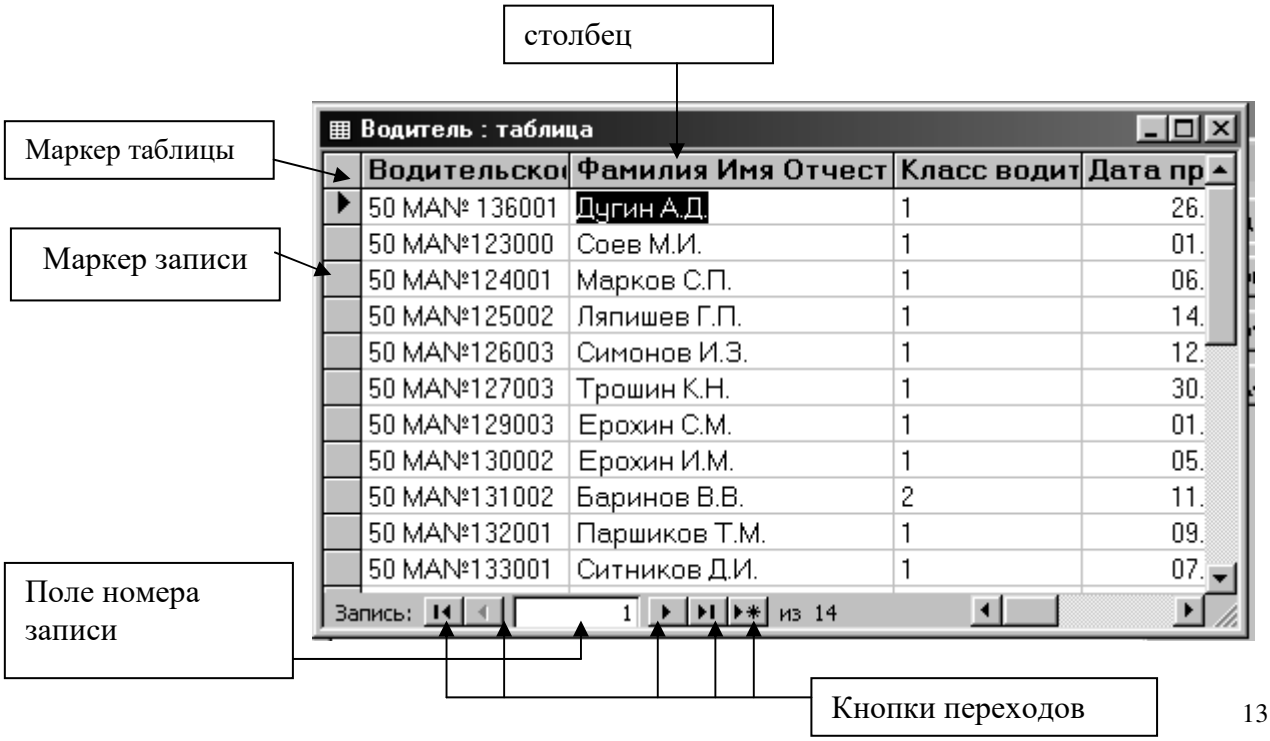
В базах данных один пользователь, вносящий изменения в базу, блокирует только одну запись, с которой он работает, причем ненадолго. Например, известно, что службы автомобильной инспекции имеют базы данных угнанных автомобилей. Тот факт, что где-то в центральной службе идет ввод новых записей об угнанных автомобилях, не мешает инспекторам на местах обращаться к базе по компьютерной сети и наводить необходимые справки. Как только ввод очередной записи завершается, она становится доступной всем инспекторам для просмотра, а некоторым (кому это положено по должности) и для редактирования.

Если в локальной или глобальной сети с одной базой работают несколько пользователей, то каждый может видеть в режиме реального времени те изменения, которые вносят в базу его коллеги.

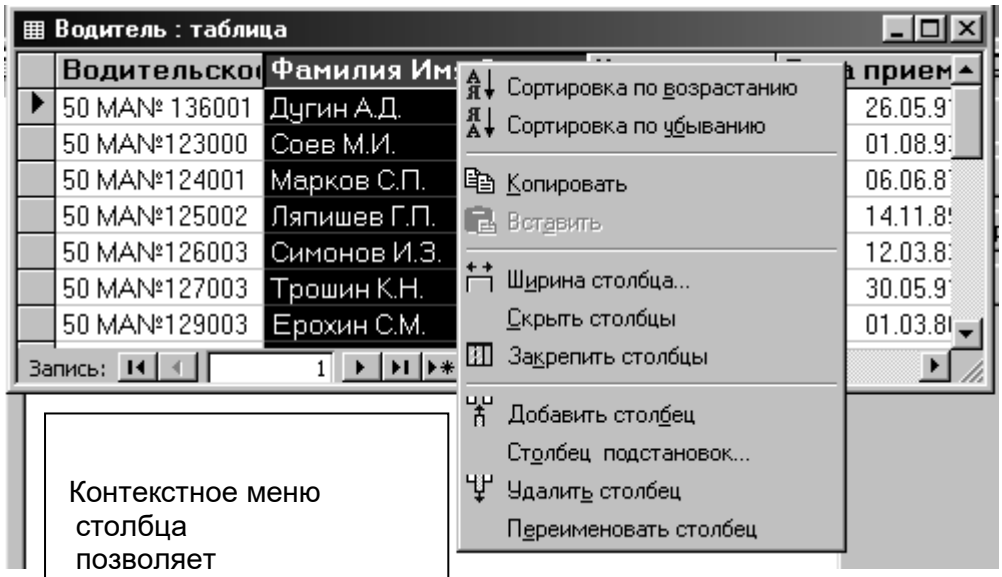
Приемы работы с таблицами баз данных

На рисунке ниже представлена типичная таблица базы данных. С ней можно работать обычными приемами управления с помощью мыши.

1. Обратите внимание на строку состояния в нижней части окна. В Access 9x эта строка называется *полем номера записи.* Это поле содержит *кнопки перехода,* с помощью которых можно эффективно перемещаться по таблице.
2. Каждая запись имеет слева кнопку *(маркер записи).* Щелчок на этом маркере выделяет всю запись и готовит ее к копированию,
   перемещению, удалению.
3. **Щелчок** правой кнопкой **на** выделенной записи открывает контекстное меню для операций с записью.
4. Маркер, находящийся в левом верхнем углу таблицы, — **это** *маркер таблицы.* Щелчок на нем выделяет всю таблицу, а правый щелчок открывает контекстное меню для операций с таблицей в целом.
5. Поля базы данных представлены в таблице *столбцами.* Каждый столбец имеет заголовок, в котором записано имя поля или **то** значение, которое задано в свойстве Подпись.
6. Если содержимое поля не полностью умещается в ячейке таблицы, столбец можно расширить. При наведении указателя мыши на границу между столбцами указатель меняет форму. Теперь границу можно перемещать методом перетаскивания, а двойной щелчок, выполненный в этот момент, автоматически устанавливает ширину столбца равной длине самого длинного значения в данном поле.
7. Щелчок на заголовке столбца выделяет весь столбец, а щелчок правой кнопкой на выделенном столбце открывает контекстное меню. В нем есть очень интересные пункты, позволяющие отсортировать записи по данному полю, вставить новый столбец, скрыть столбец и прочее.
8. Скрытый столбец не исчезает из базы, а только перестает отображаться на экране. Чтобы снова его отобразить, надо навести указатель на границу между столбцами в том месте, где был скрыт столбец, и выполнить двойной щелчок. Скрытый столбец опять станет видимым.



Контекстное меню записи позволяет удалять, копировать и перемещать записи и управлять высотой строк.



Контекстное меню столбца позволяет сортировать записи, копировать, удалять и перемещать столбцы, управлять их шириной и режимом отображения

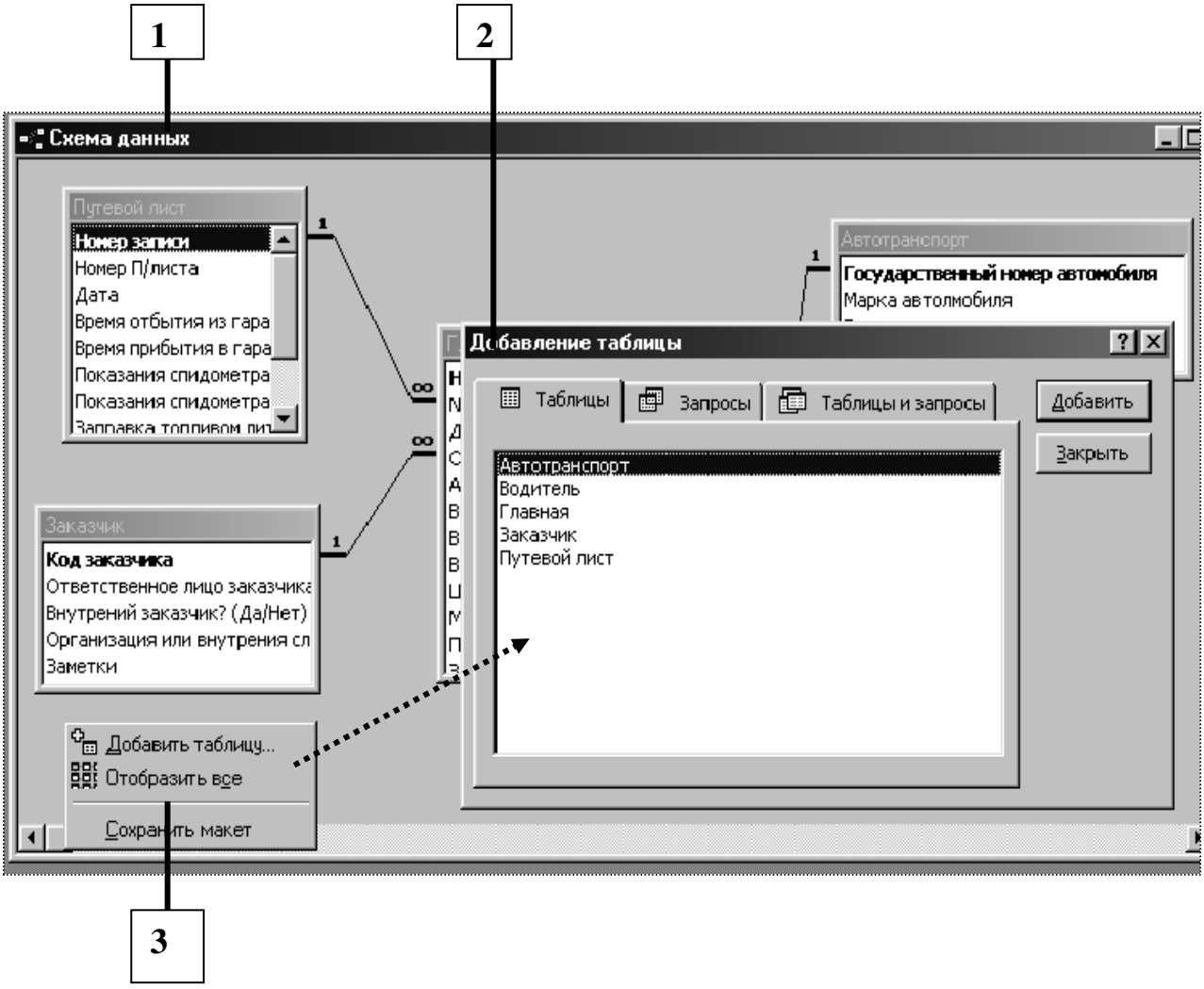
Создание связей между таблицами

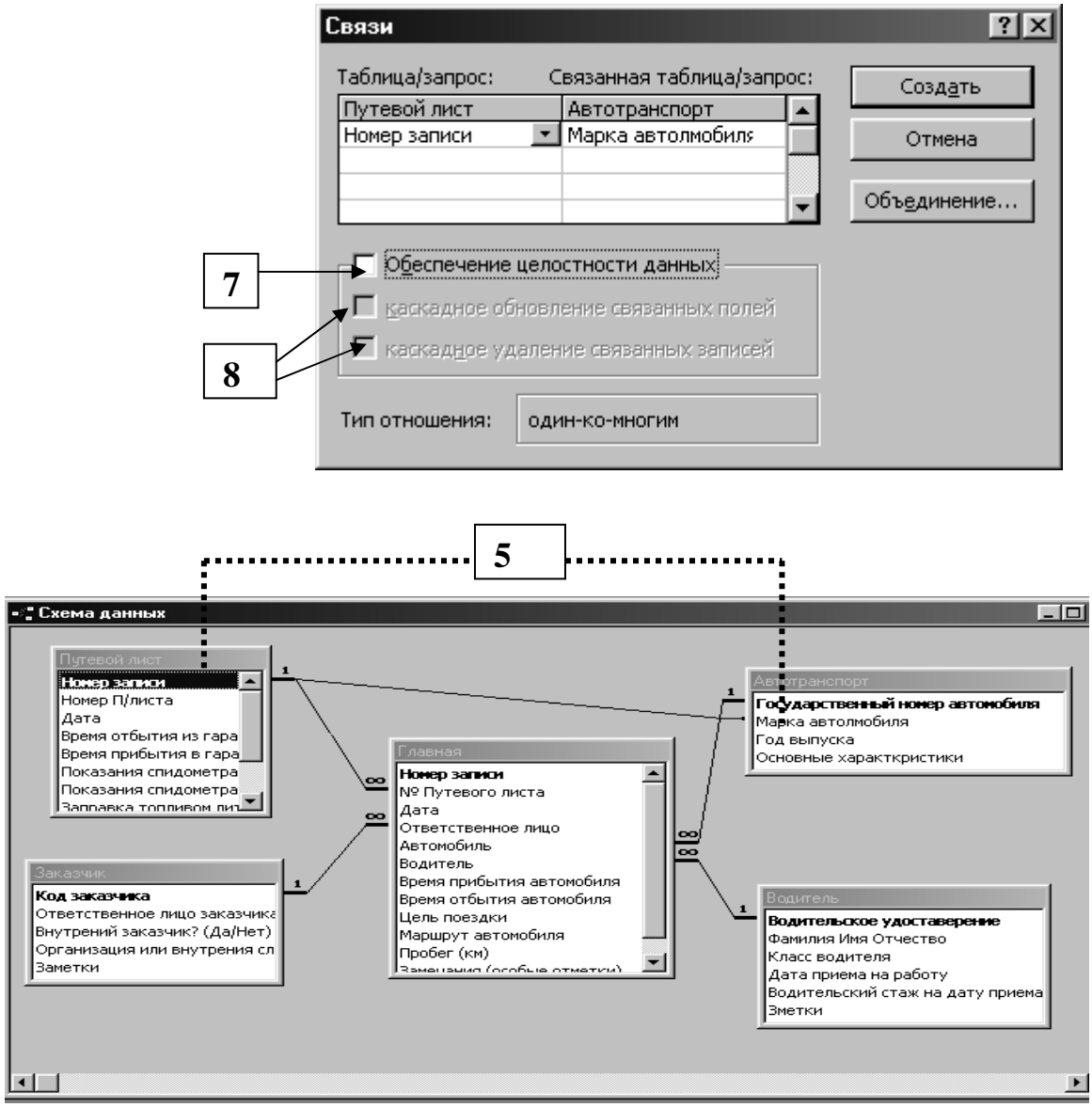
Основные преимущества систем управления базами данных реализуются при работе не с отдельными таблицами, а с группами взаимосвязанных таблиц. Для создания связей между таблицами СУБД Access 9x имеет специальное диалоговое окно, которое называется Схема данных.

1. Окно Схема данных открывают щелчком на одноименной кнопке панели инструментов или командой Сервис > Схема данных.
2. Если ранее никаких связей между таблицами базы не было, то при открытии окна Схема данных одновременно открывается окно Добавление таблицы, в котором можно выбрать нужные таблицы для включения в структуру межтабличных связей.

3. Если связи между таблицами уже были заданы, то для введения в схему данных новой таблицы надо щелкнуть правой кнопкой мыши на схеме данных и в контекстном меню выбрать пункт Добавить таблицу.
4. Введя в схему данных все таблицы, которые надо связать, можно приступать к созданию связей между полями таблиц.
5. Связь между полями устанавливают путем перетаскивания имени поля из одной в таблицы в другую на соответствующее ему связанное поле.
6. После перетаскивания открывается диалоговое окно Связи, в котором можно задать свойства образующейся связи.
7. Включение флажка Обеспечение условия целостности данных позволяет защититься от случаев удаления записей из одной таблицы, при которых связанные с ними данные других таблиц останутся без связи.

   Чтобы условие целостности могло существовать, поле основной таблицы должно обязательно быть ключевым и оба поля должны иметь одинаковый тип.
8. Флажки Каскадное обновление связанных полей и Каскадное удаление связанных записей обеспечивают одновременное обновление или удаление данных во всех подчиненных таблицах при их изменении в главной таблице. Если клиент Соколова выйдет замуж и изменит фамилию на Воронову, то придется внести изменение только в поле Фамилия таблицы Клиенты. В прочих таблицах изменения произойдут автоматически.





Диалоговое окно Схема данных наглядно отображает связи между таблицами. Чтобы удалить связь, надо щелкнуть на линии связи правой кнопкой мыши и воспользоваться командой Удалить контекстного меню.

Запросы

Предположим, что на крупном предприятии есть огромная база"' данных Кадры, содержащая подробнейшие сведения о каждом сотруднике. Кроме формальной информации база может содержать и конфиденциальную, например сведения о заработной плате. Вся эта информация хранится в базовых таблицах.

Работать с базой данных Кадры могут разные подразделения предприятия, и всем им нужны разные данные. Не все то, что положено знать службе безопасности предприятия, должно быть доступно главному врачу, и наоборот. Поэтому доступ пользователей к базовым таблицам *закрывают.*

Для доступа к данным есть другое, гораздо более гибкое и удобное средство — *запросы*. Для одной и той же таблицы можно создать множество разных запросов, каждый из которых сможет извлекать из таблицы лишь малую часть информации, но именно ту

часть, которая в данный момент необходима. У сотрудника бухгалтерии должен быть запрос, который позволит определить сколько дней в году по болезни отсутствовал тот или иной работник, но у него не должно быть запроса, позволяющего узнать, чем он болел и где лечился, а у главного врача такой запрос быть должен.

В результате работы запроса из общей исходной базы формируется *результирующая таблица,* содержащая часть общей информации, соответствующую запросу.

Важным свойством запросов является то, что при создании результирующей таблицы можно не только выбирать информацию из базы, но и обрабатывать ее. При работе запроса данные могу упорядочиваться (сортироваться), фильтроваться (отсеиваться), объединяться, разделяться, изменяться, и при этом никаких изменений в базовых таблицах может не происходить.

Результаты обработки сказываются только на содержании результирующей таблицы, а она имеет временный характер, и иногда ее даже называют моментальным снимком.

И еще одним ценным свойством запросов является их способность выполнять *итоговые вычисления.* Запрос может не только выдать результирующую таблицу, но и найти, например, среднее (наибольшее, наименьшее, суммарное и т. п.) значение по какому-то полю.

Начинающих пользователей баз данных часто удивляет отсутствие в таблицах элементарной возможности вставить новую запись между другими. Записи всегда добавляются только в конец базы. На вопрос, почему так происходит, ответ простой: потому что в упорядочении таблиц нет необходимости. Для этого существуют запросы. Совершенно неважно, под каким номером внесена в таблицу та или иная запись. Если нужно видеть ее в строго определенном месте (например, рядом с другими аналогичными], значит нужно создать запрос, который сгруппирует записи по заданному признаку.

Запросы на выборку

Существует немало различных видов **запросов,** но самые простые из них и, к тому же, используемые наиболее часто — это *запросы на выборку.* С них и принято начинать знакомство с созданием запросов.

Цель запроса на выборку состоит в создании *результирующей таблицы,* в которой отображаются только нужные **по** условию запроса данные из *базовых таблиц.*

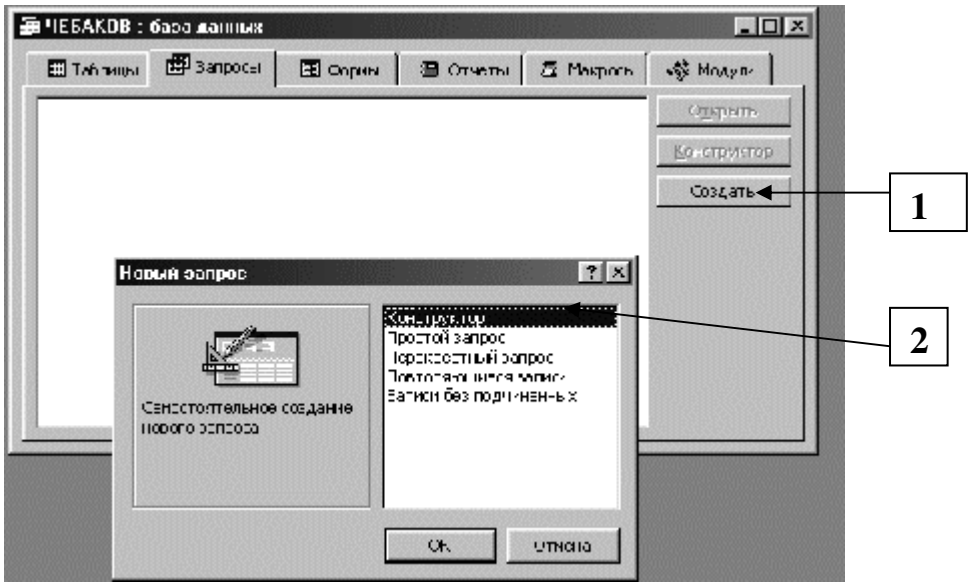
Как и другие объекты Access 9x, запросы можно создавать автоматически с помощью Мастера или вручную. И, как обычно, на этапе обучения лучше не пользоваться Мастером, чтобы почувствовать работу с запросами «кончиками пальцев».

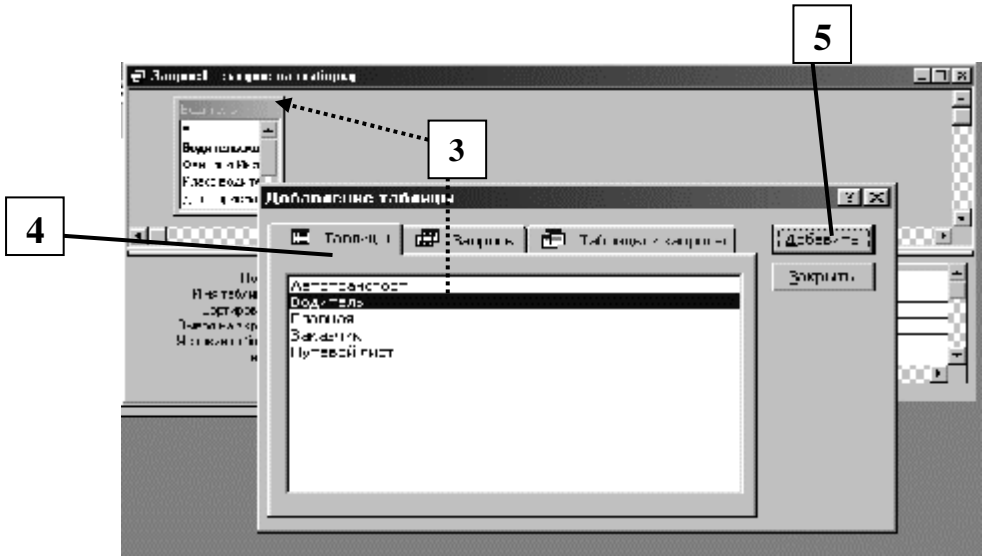
Для создания запросов к базам данных существует специальный *язык запросов.* Он называется *SQL (Structured Query Language — структурированный язык запросов).* К счастью, те, кто пользуются СУБД Access 9x, могут позволить себе не изучать этот язык. Вместо него в Access 9x есть простое средство, которое называется *бланком запроса по образцу.* С его помощью можно сформировать запрос простыми приемами, перетаскивая элементы запроса между окнами.

Выбор базовых таблиц для запроса

1. Создание запроса к базе начинается с открытия вкладки Запросы диалогового окна База данных и щелчка на кнопке Создать.
2. В открывшемся диалоговом окне Новый запрос задают ручной режим создания запроса выбором пункта Конструктор.

3. Создание запроса в режиме Конструктора начинают с выбора тех таблиц базы, на которых будет основан запрос.
4. Выбор таблиц выполняют в диалоговом окне Добавление таблицы. В нем отображаются все таблицы, имеющиеся в базе.
5. Выбранные таблицы заносят в верхнюю половину бланка запроса по образцу щелчком на кнопке Добавить.
6. В окне Добавление таблицы обратите внимание на наличие трех вкладок: Таблицы, Запросы, Запросы и таблицы. Они говорят о том, что запрос не обязательно основывать только на таблицах. Если ранее уже был создан запрос, то новый запрос можно основывать и на нем.





Какие именно таблицы использовать в качестве базовых, решает **сам** создатель запроса.

Заполнение бланка запроса по образцу

Бланк запроса по образцу — удивительно изящное и удобное средство создания запросов. Наверное, оно в немалой степени способствует тому успеху, который СУБД Access 9x имеет у потребителей.

1. Бланк запроса по образцу имеет две панели. **На** верхней панели расположены списки полей тех таблиц, на которых основывается запрос.

2. Строки нижней панели определяют структуру запроса, то есть структуру результирующей таблицы, в которой будут содержаться данные, полученные по результатам запроса.

3. Строку Поле заполняют перетаскиванием названий полей из таблиц в верхней части бланка. Каждому полю будущей результирующей таблицы соответствует один столбец бланка запроса по образцу.

4. Строка Имя таблицы заполняется автоматически при перетаскивании поля.

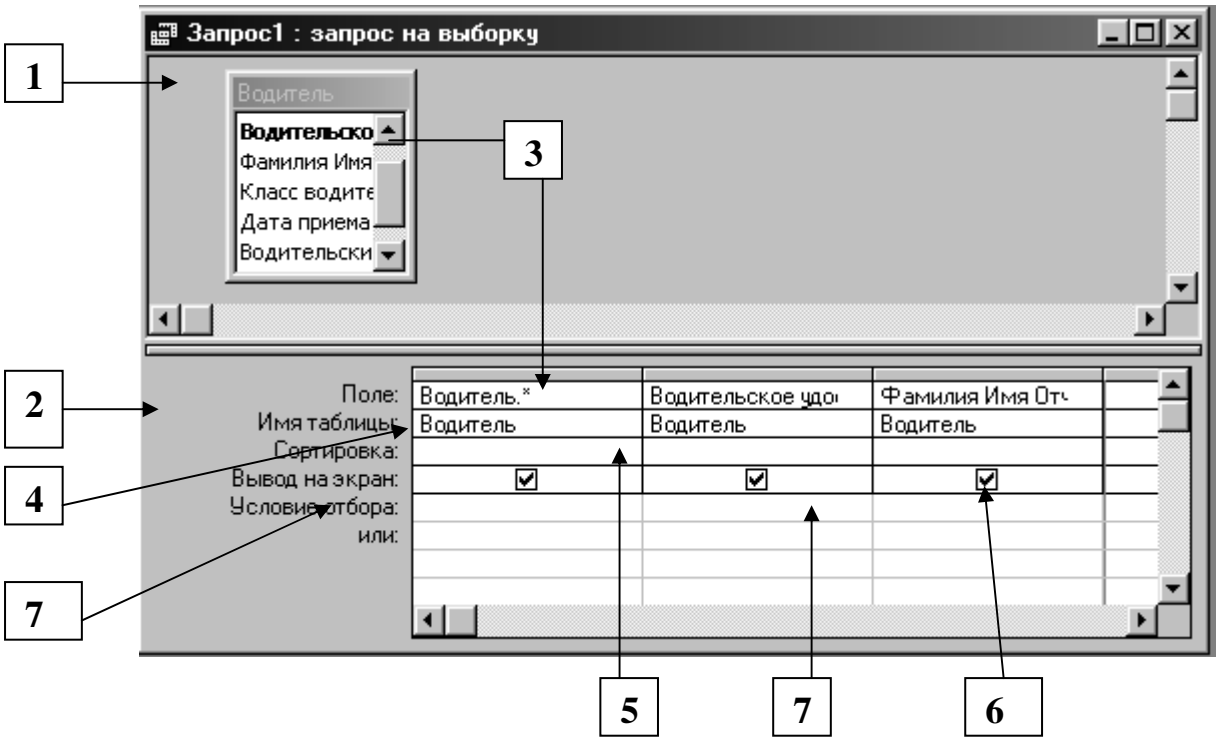
5. Если щелкнуть на строке Сортировка, появится кнопка раскрывающегося списка, содержащего виды сортировки. Если назначить сортировку по какому-то полю, данные в результирующей таблице будут отсортированы по этому полю.

б. Бывают случаи, когда поле должно присутствовать в бланке запроса по образцу, но не должно отображаться в результирующей таблице. В этом случае можно запретить его вывод на экран, сбросив соответствующий флажок.

7. Самая интересная строка в бланке запроса по образцу называется Условие отбора. Именно здесь и записывают тот критерий, по которому выбирают записи для включения в результирующую таблицу. По каждому полю можно создать свое условие отбора.

8. Запуск запроса выполняют щелчком на кнопке Вид. При запуске образуется результирующая таблица.

9. Чтобы выйти из результирующей таблицы и вернуться к созданию запроса в бланке запроса по образцу, нужно еще раз щелкнуть на кнопке Вид.



ФОРМЫ

Обычно разработчик базы данных создает структуру таблиц и запросов, но заполнением таблиц информацией он не занимается. Для этого есть специальные кадры

(обычно малоквалифицированные), выполняющие функции наборщиков. Для упрощения **их** труда разработчик базы может подготовить специальные объекты — *формы.*

Форма представляет собой некий электронный бланк, в котором имеются поля для ввода данных. Наборщик вводит данные в эти поля, и данные автоматически заносятся в таблицы базы.

Зачем нужны формы?

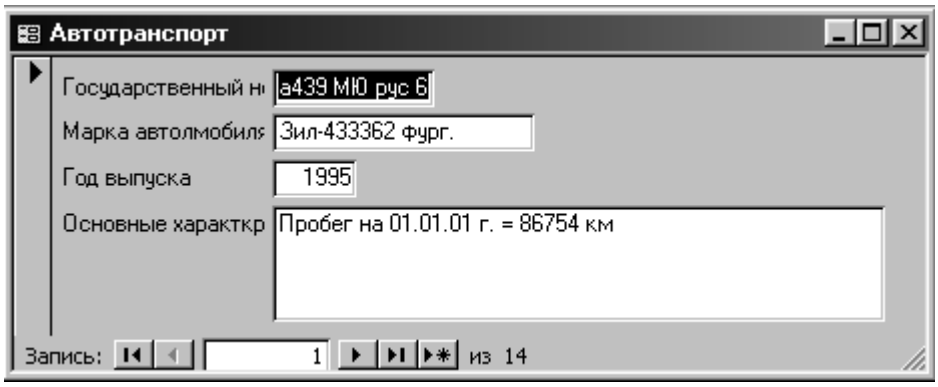
Данные в таблицу можно вносить и без помощи каких-либо форм, но существуют по крайней мере четыре причины, которые делают формы незаменимым средством ввода данных в базу.

Во-первых, малоквалифицированному персоналу нельзя предоставлять доступ к таблицам (самому ценному из того, что есть в базе). Представьте, что будет, если новичок «наведет порядок» в таблице банка, хранящей расчетные счета клиентов.

Во-вторых, разные люди могут иметь разные права доступа к информации, хранящейся в таблицах. Например, один имеет право вводить только имена и адреса клиентов, другой — только номера их расчетных счетов, а третий — только денежные суммы, хранящиеся на этих счетах. Сговор между этими людьми должен быть исключен. Для ввода данных им предоставляют разные формы, хотя данные из форм могут поступать в одну таблицу.

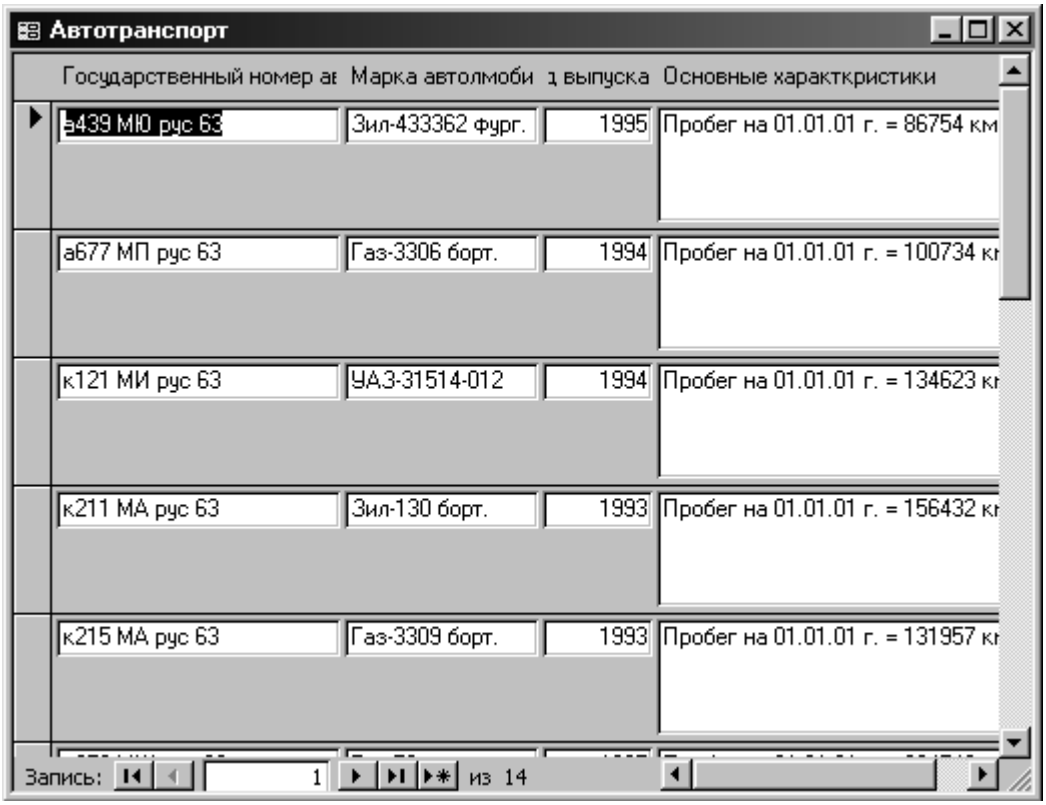
В-третьих, ввод данных в таблицу — чрезвычайно утомительное занятие. Уже после нескольких часов работы люди делают ошибки. Ввод данных в форму проще. Здесь многое можно автоматизировать. К тому же элементы управления форм настраивают таким образом, чтобы при вводе данных выполнялась их первичная проверка.

И наконец, в-четвертых, надо вспомнить, откуда берется информация для баз данных. Как правило, ее берут из бумажных бланков (анкет, заявлений, накладных, счетов, описей, ведомостей, справок и т. п.). Экранные формы можно сделать точной копией бумажных бланков, с которых происходит ввод данных. Благодаря этому во много раз уменьшается количество ошибок при вводе и значительно снижается утомляемость персонала.



Такая форма называется в столбец. В ней всегда видна одна запись.

Поля этой записи расположены в столбец. Переключение между записями выполняют с помощью кнопок перехода в нижней части формы.



Такие формы называют ленточными. В них одновременно отображается несколько записей.

Создание форм

Как и другие объекты Access 9x, формы можно создавать вручную или автоматически, причем несколькими способами. При создании таблиц и запросов мы рекомендовали на первых порах автоматическими средствами не пользоваться, чтобы вникнуть в терминологию и подготовить себя к работе с Мастером, задающим непонятные для начинающих вопросы. С формами дело обстоит иначе. Они состоят из многочисленных элементов управления, и от того, насколько аккуратно эти элементы расположены на экране, зависит внешний вид формы. Автоматические средства позволяют создавать аккуратные формы и не задают пользователю лишних 'вопросов. Начинать работу лучше с них.

Автоформы

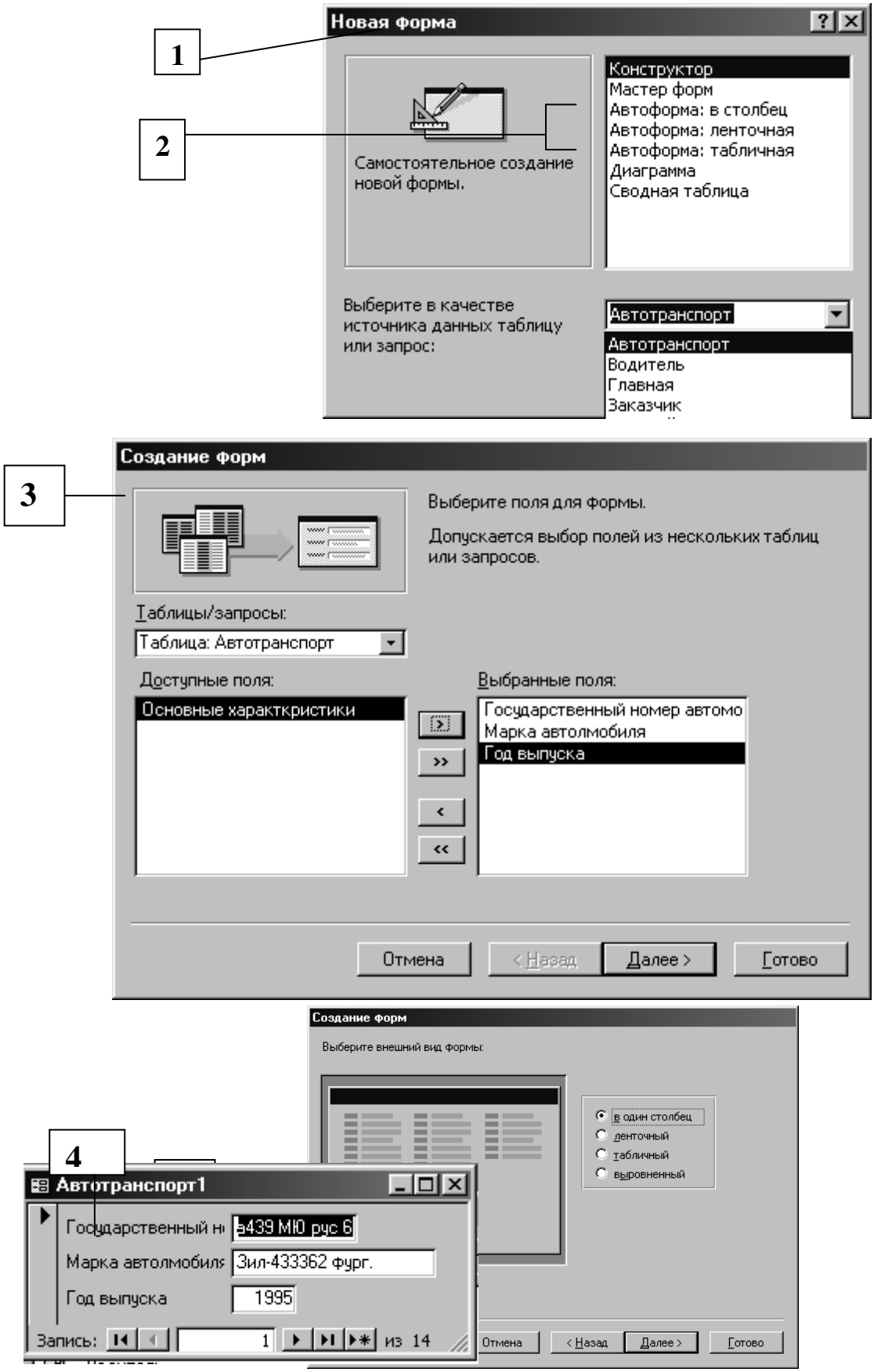
1. Автоформы — самый простой вид автоматических форм. Для создания автоформы надо открыть вкладку Формы в диалоговом окне База данных и щелкнуть на кнопке Создать — откроется окно Новая форма.
2. В диалоговом окне Новая форма выбирают в качестве источника данных для формы какую-либо таблицу или запрос, после чего создают автоформу двойным щелчком в списке выбора вида автоформы *(табличная, ленточная* или *в столбец).*

Создание формы с помощью Мастера

3. С помощью Мастера форма создается всего в четыре этапа:
   - выбор полей, данные для которых можно будет вводить в **форме;**
   - выбор внешнего вида формы (один из четырех);
   - выбор фонового рисунка формы (один из десяти);
   - задание имени формы.

   Все эти пункты достаточно хорошо объяснены в Мастере и не требуют никаких пояснений.

4. Готовую форму можно сразу же использовать для просмотра существующих записей или для ввода новых.



Структура форм

Создавая формы автоматическими средствами, можно не задумываться над их структурой, но при разработке формы вручную **со** структурой приходится иметь дело.

Структуру формы составляют ее *разделы,* а разделы содержат *элементы управления.*

Разделы формы

1. Самый простой способ познакомиться с разделами формы состоит в том, чтобы взять готовую форму, например, созданную с помощью Мастера, и посмотреть ее устройство в режиме Конструктора. Как мы уже знаем, для этого надо щелкнуть на кнопке Вид на панели управления Access 9x.

2. При просмотре в Конструкторе мы видим структуру формы. Обратите внимание на то, что рядом с ней открывается *панель элементов,* содержащая заготовки и инструменты для создания элементов управления формы.

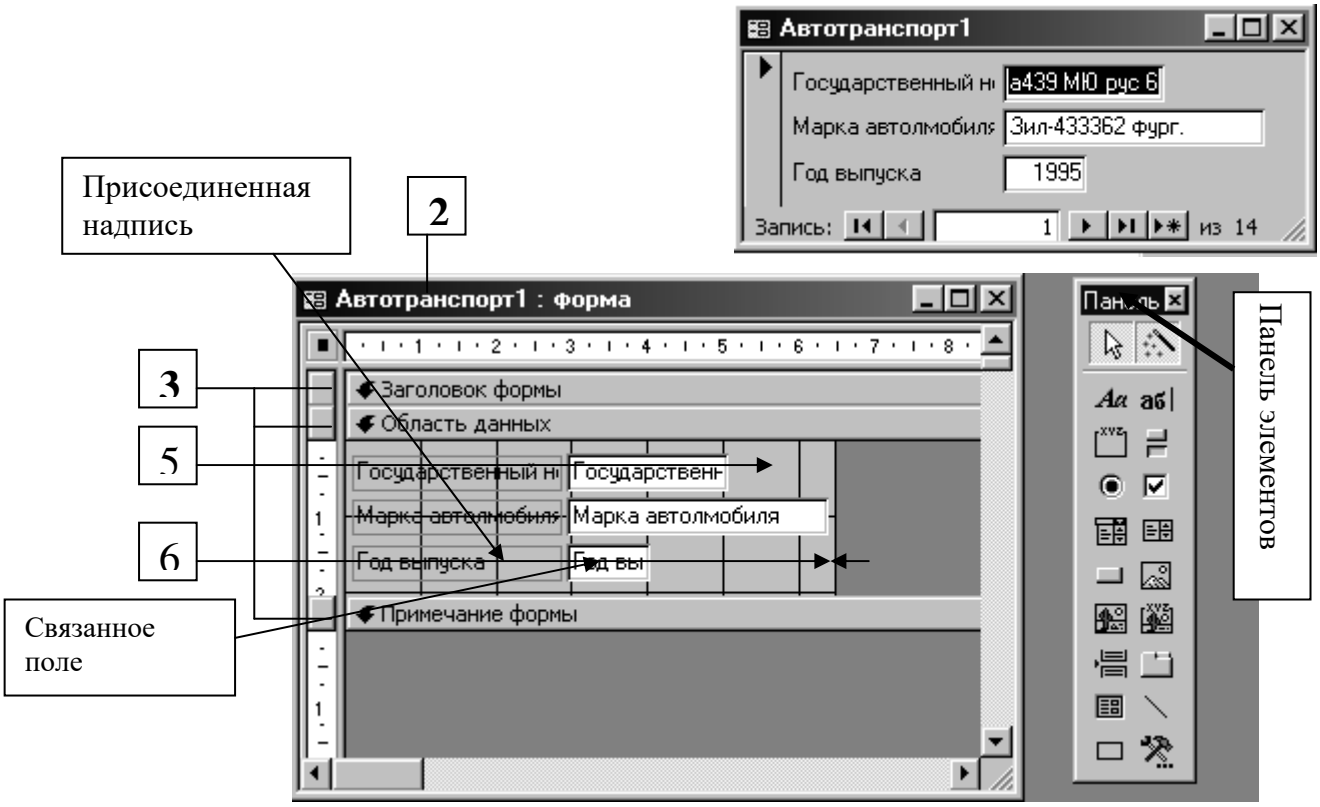
3. В структуре формы четко видны три раздела: ***раздел заголовка*** формы, *область данных* и *раздел примечания* формы.

В нашем случае заполнена только область данных. Так произошло потому, что форму создавал Мастер, который не потрудился создать и заполнить вспомогательные разделы.

4. Все, что содержится в области данных, является *элементами управления.* В нашем случае здесь присутствуют элементы управления только двух типов: *связанное поле* (то, что в него вводится, поступает и в одноименное поле таблицы, на базе которой создана форма) и *присоединенная надпись* (называется так, поскольку перемещается вместе со своим элементом управления). В нашем случае содержание присоединенной надписи совпадает с названием связанного поля, но, как вы понимаете, это можно и изменить.

5. Фоновый рисунок, лежащий под элементами управления, показывает размер *рабочего поля формы.*

6. Размеры разделов и размеры рабочего поля формы можно изменять с помощью мыши. При наведении на границу раздела указатель меняет форму. В этот момент границу можно перемещать методом перетаскивания.

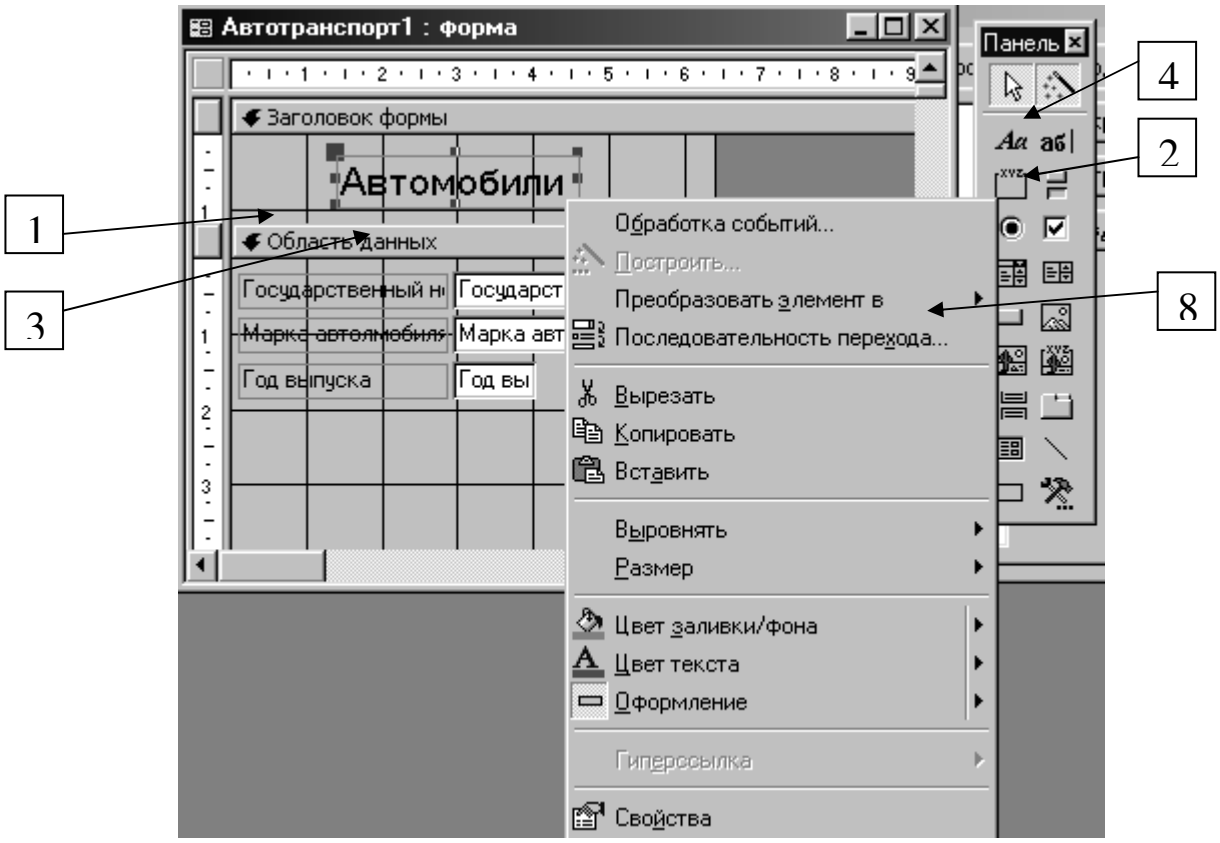


Создание надписей

Редактирование форм состоит в создании новых или изменении имеющихся элементов управления, а также в изменении их взаимного расположения.

При рассмотрении приемов создания новых элементов управления мы воспользуемся тем фактом, что Мастер, создавший форму, не заполнил ее раздел заголовка.

1. Перетащив вниз разделительную границу между заголовком и областью данных, мы можем освободить вверху достаточно места для создания крупной надписи.
2. На панели элементов существует специальный элемент управления для создания заголовков, который называется Надпись.
3. Щелкнув на нем, а потом на форме, мы получаем текстовую рамку, в которую можно вводить произвольный текст. При вводе текста не надо заботиться о его форматировании. Неважно, как он выглядит и где расположен. Закончив ввод, надо нажать клавишу ENTER, после чего можно приступать к оформлению текста.
4. Для форматирования элемента управления его надо сначала выделить. Для этого служит инструмент Выбор объектов.
5. При выделении элемента управления вокруг него образуется рамка с восемью маркерами (по углам и по центрам сторон рамки). Рамку можно растягивать или сжимать методом перетаскивания границ. При наведении на маркер указатель мыши меняет форму, принимая изображение открытой ладони. В этот момент рамку можно перемещать.
6. Особую роль играет левый верхний маркер рамки. При наведении на него указатель мыши принимает форму указательного пальца. О роли этого маркера мы расскажем чуть позже.
7. Когда объект выделен, можно изменять параметры шрифта, метод выравнивания текста и другие элементы форматирования. Это выполняют обычными средствами форматирования, доступными через соответствующую панель инструментов Access.
8. Если щелкнуть на выделенном элементе правой кнопкой мыши, откроется его контекстное меню, в котором имеются дополнительные возможности изменения оформления. В нашем случае, например, применено Оформление с тенью.

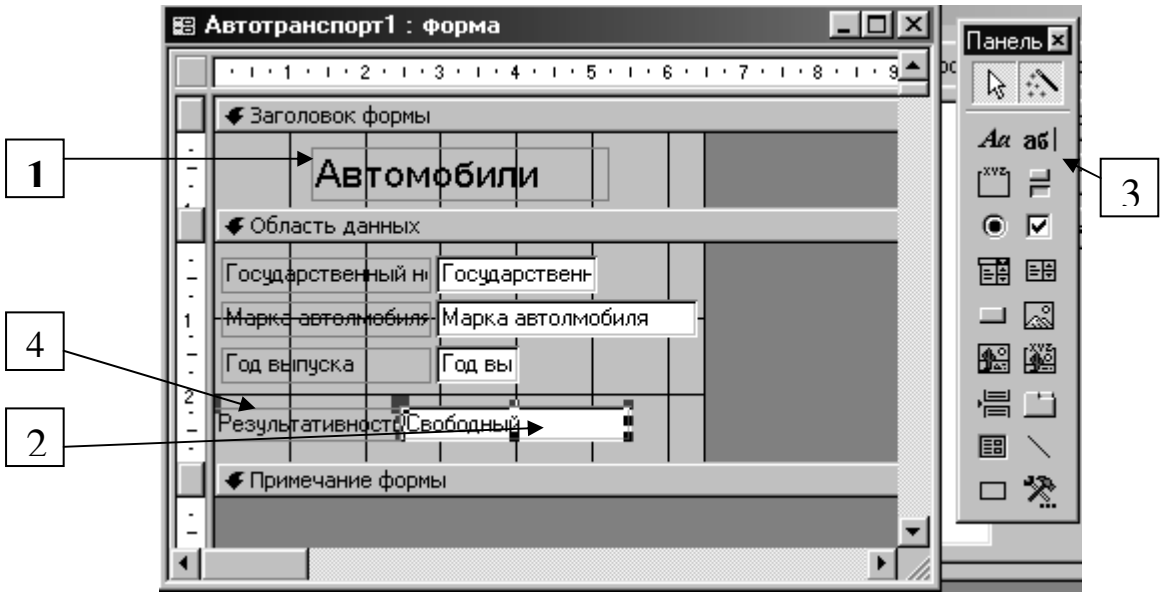


Создание и редактирование связанных полей

1. Заголовок таблицы, который мы только что создали, не связан ни с одним из полей таблицы. Поэтому элемент управления Надпись еще называют *свободным полем.* Текст, введенный в него, остается неизменным независимо от того, какую запись в этот момент просматривают в форме.

2. Совсем иначе обстоит дело с элементами управления, в которых отображается содержимое полей таблицы. Такие элементы управления называют *связанными полями.*
3. Для их создания служит элемент Поле на панели элементов.
4. При создании связанного поля вместе с ним одновременно образуется еще один элемент управления — *присоединенная надпись.* Она перемещается вместе со связанным полем и образует с ним единое целое.
5. Обратите внимание на то, что, что слово «Результативность» в присоединенной надписи записано без последней буквы. Это не ошибка. Просто Мастер, создававший форму, сделал **это** неаккуратно, и связанное поле «наехало» на присоединенную надпись.
6. Оторвать поле от присоединенной надписи позволяет уже упомянутый маркер, расположенный в левом верхнем углу. При наведении на него указатель мыши принимает форму указательного пальца. В этот момент связанное поле можно оторвать **от** присоединенной надписи и перемещать отдельно.

   Перемещать элементы управления и изменять их размеры с помощью мыши не слишком удобно. Гораздо удобнее использовать для этой цели курсорные клавиши в комбинации с клавишами SHIFT или CTRL В первом случае происходит изменение размеров элемента управления, а во втором — изменение его расположения.
7. Чтобы элементы управления располагались в форме ровными рядами, существуют специальные команды выравнивания. Сначала надо выделить группу элементов управления с помощью инструмента Выбор объектов (группа выбирается при нажатой клавише SHIFT), а потом дать команду формат > Выровнять и выбрать метод выравнивания.



Прочие элементы управления формы

При создании формы вручную элементы управления размещают на ней так, как удобно проектировщику. Созданные элементы управления формы выравнивают с помощью команды Формат > Выровнять.

Кроме рассмотренных выше элементов управления Надпись и Поле, существует еще несколько полезных элементов управления.

1. ***Переключатели.*** С ними можно связать команды, например, выполняющие фильтрацию.

2. ***Флажки.*** Действуют аналогично переключателям, но в отличие от них, допускают множественный выбор. Удобны для управления режимами сортировки данных.
3. ***Список.*** Может содержать фиксированный набор значений или значения из заданного поля одной из таблиц. Позволяет не вводить данные, а выбирать их из списка.
4. ***Поле со списком***. Применяется так же, как и список, но занимает меньше места в форме, поскольку список открывается только после щелчка на раскрывающей кнопке.
5. ***Командные кнопки***. С каждой из них можно связать какую-либо полезную команду, например команду поиска записи, перехода между записями и другие.
6. ***Вкладки.*** Позволяют разместить много информации на ограниченной площади. На вкладках размещают другие элементы управления.
7. ***Поле объекта OLE.*** Служит для размещения внешнего объекта, соответствующего принятой в Windows концепции связывания и внедрения объектов. Объектом, как правило, является иллюстрация, например фотография, но это может быть и видеозапись, и музыкальный фрагмент, и голосовое сообщение.

Существуют два типа полей для размещения объектов **OLE:**

Свободная рамка объекта и Присоединенная рамка объекта. В первом случае рамка не связана ни с каким полем таблиц базы данных. Объект, находящийся в ней, выполняет роль иллюстрации и служит для оформления формы. С Присоединенной рамкой связано одно из полей таблицы. В ней отображается содержимое этого поля. Это содержимое может меняться при переходе от одной записи к другой.

Отчеты

Напомним функции основных объектов базы данных:

- таблицы служат для хранения данных;
- запросы служат для выбора данных из таблиц, а также для автоматизации операций по обновлению и изменению таблиц;
- формы служат для упрощения операций ввода данных в таблицы, но могут быть использованы и для просмотра результатов .работы запросов на экране.

Из основных объектов нам осталось рассмотреть только *отчеты.* Отчеты во многом похожи на формы и тоже позволяют получить результаты работы запросов в наглядной форме, но только не на экране, а в виде распечатки на принтере. Таким образом, в результате работы отчета создается *бумажный документ.*

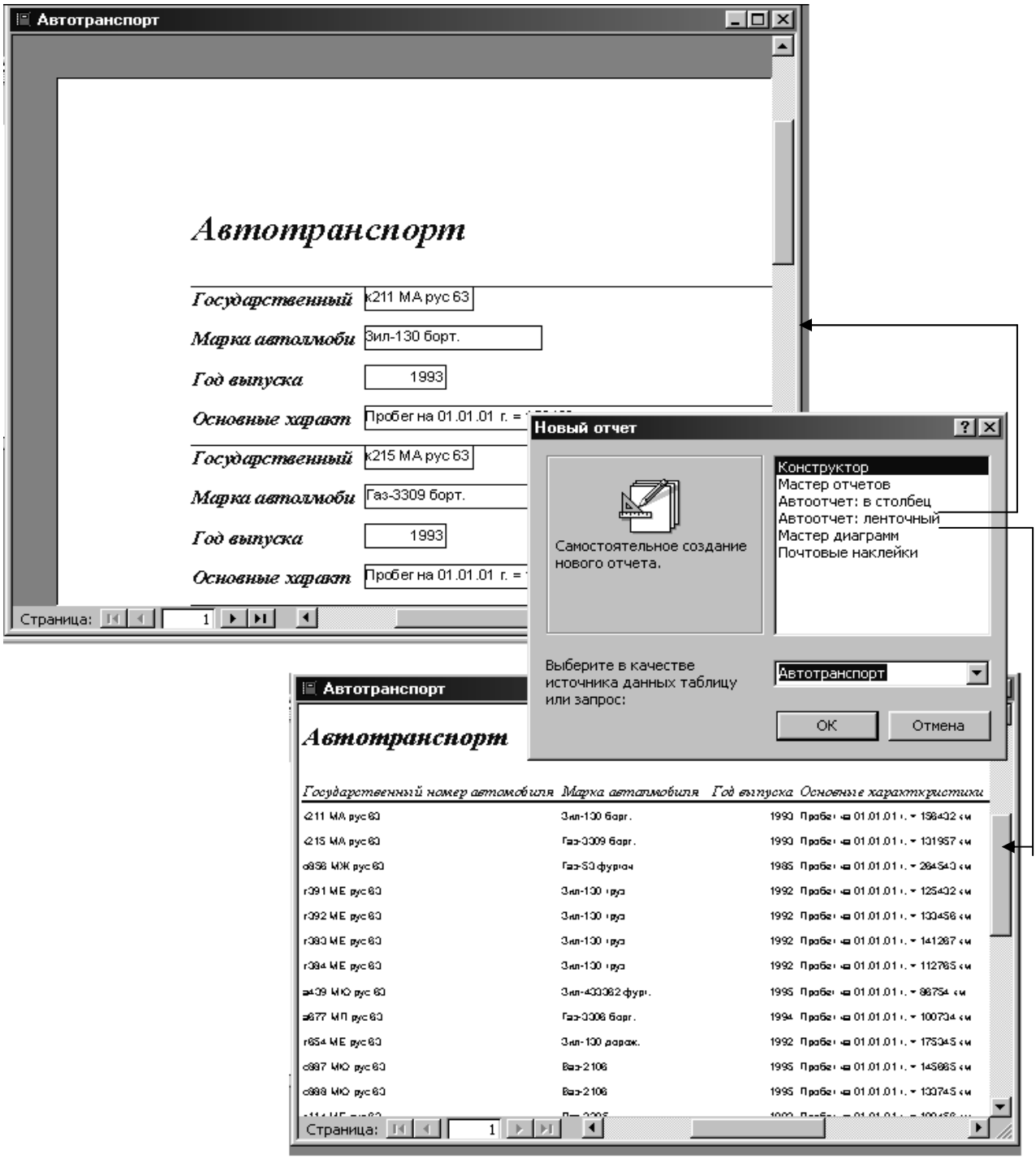
Автоотчеты

Большая часть того, что было сказано о формах, относится и к отчетам. Выбрав в диалоговом окне База данных вкладку Отчеты и щелкнув на кнопке Создать, мы получаем диалоговое окно Новый отчет, позволяющее создать отчет автоматически *(автоотчет), с* помощью Мастера или вручную.

Точно так же, как и с формами, с отчетами удобнее знакомиться в режиме автоматического создания. Создайте на основе любой таблицы автоотчет в *столбец* или *ленточный.* Операция настолько проста, что сводится к одному щелчку левой кнопки мыши.

Отчеты предназначены для вывода информации на принтер, поэтому для расчета расположения данных на печатной странице программа Access 9x должна «знать» все необходимое об особенностях принтера. Эти данные Access получает от операционной системы. Соответственно, принтер в системе должен быть установлен.

При отсутствии принтера отчеты создавать все-таки можно. Достаточно выполнить программную установку с помощью команды операционной системы: Пуск ^ Настройка > Принтеры > Установка принтера, после чего зарегистрировать драйвер принтера, либо взяв его с гибкого диска, либо выбрав один из драйверов, прилагающихся к самой операционной системе.



Структура отчета

Как и формы, отчеты состоят из разделов, а разделы могут содержать элементы управления. Но, в отличие от форм, разделов в отчетах больше, а элементов управления, наоборот, меньше.

Со структурой отчета проще всего ознакомиться, создав какой-либо автоотчет, а затем открыв его в режиме Конструктора.

1. Как видно из рисунка, структура отчета состоит из пяти разделов: *заголовка отчета, верхнего колонтитула, области данных, нижнего колонтитула* и *примечания отчета.* По сравнению с формами новыми являются разделы верхнего и нижнего колонтитулов.

2. Раздел *заголовка* служит для печати общего заголовка отчета.

3. Раздел *верхнего колонтитула* можно использовать для печати подзаголовков, если отчет имеет сложную структуру и занимает много страниц. Здесь можно также помещать и *колонцифры* (номера страниц), если это не сделано в нижнем колонтитуле.

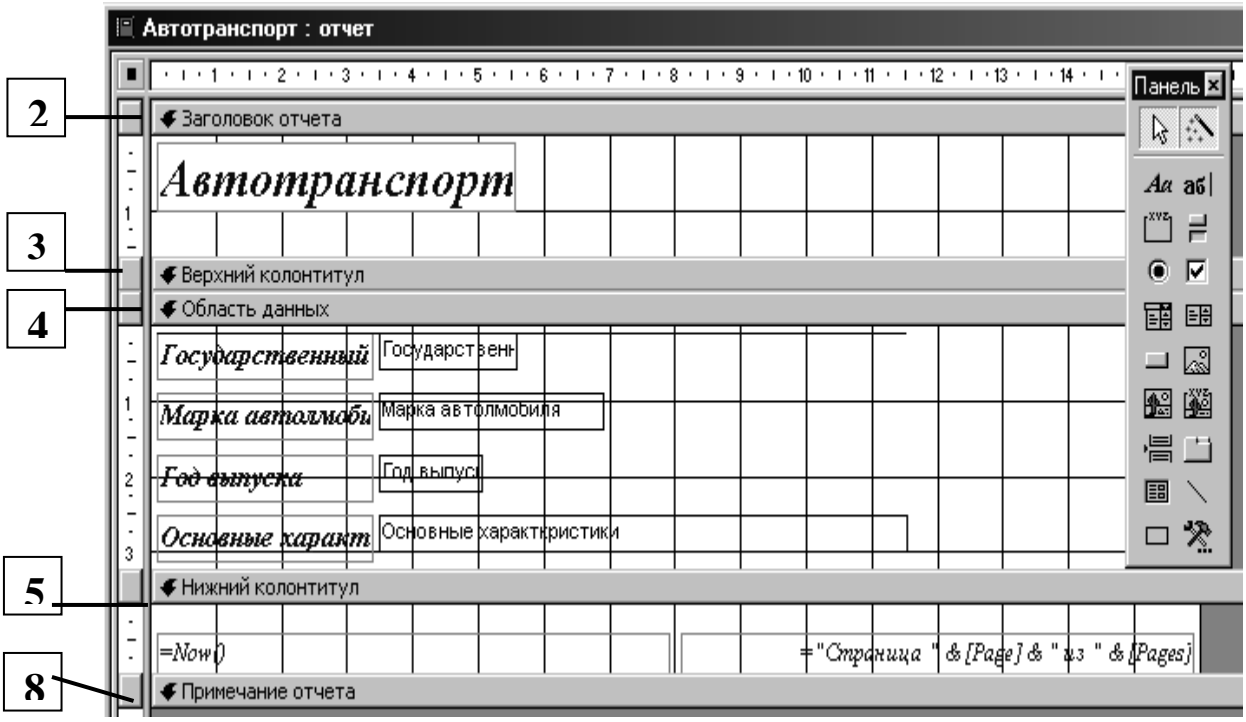
4. В *области данных* размещают элементы управления, связанные с содержимым полей таблиц базы. В эти элементы управления выдаются данные из таблиц для печати на принтере. Порядок размещения и выравнивания элементов управления тот же, что и при создании структуры форм.

5. Раздел *нижнего колонтитула* используют для тех же целей, что и раздел верхнего колонтитула. В нашем случае в нем размещены два элемента управления.

6. В первом элементе управления выводится текущая дата. Для этого использована встроенная в Access 9x функция Now(). Она возвращает текущую дату и помещает ее в поле, а отчет воспроизводит ее при печати.

7. **Во** втором элементе управления выводится номер страницы и общее количество страниц. Для их определения использованы встроенные функции Page() и Pages(). Тот текст, который записан в кавычках, воспроизводится «буквально», а оператор & служит для «склеивания» текста, заключенного в кавычки, со значениями, возвращаемыми функциями. Оператор & называется *оператором конкатенации.*

8. Раздел *примечания* используют для размещения дополнительной информации. В нашем примере он не использован,



Закрепление пройденного

- *Системы управления базами данных (СУБД)* — это специальные программные средства, предназначенные для работы с файлами баз данных (файлами специального формата, содержащими информацию, структурированную заданным образом).
- Современные СУБД позволяют хранить в виде файлов данные любых типов: числовые, текстовые, графические, звуковые, видео и прочие.
- Данные в базах хранятся в виде *таблиц.*
- Каждая таблица имеет *структуру.*
- Структура таблицы определяется составом ее *полей* и их *свойствами.* Важнейшими свойствами полей являются: *тип поля* и *размер поля.* Для хранения разных *типов данных* используют *поля соответствующих типов.*

- Данные, хранящиеся в таблице, **можно** изменять, удалять;

сортировать, фильтровать, размножать и выполнять с ними другие операции.

- Для автоматизации операций по работе с данными, в частности, для отбора нужных данных, применяют специальные объекты, которые называются *запросами.*
- В СУБД Access 9х запросы создают с помощью специального *бланка запроса по образцу.*
- С помощью *запросов на выборку* производят выбор данных **из** базы, их обработку, выполнение итоговых вычислений и другие операции. По результатам работы запроса создается временная *результирующая таблица.*
- На основе результирующей таблицы, можно создавать новые таблицы или изменять существующие. Для этого служат *запросы на изменение,*
- Для ввода данных в таблицы или для просмотра данных в наглядной форме служат специальные объекты, называемые *формами.* Формы — экранные объекты
- Структура форм состоит из *разделов* и *элементов управления.* Проектирование формы состоит в размещении элементов управления на бланке формы и в задании связей между **этими** элементами и полями таблиц или запросов базы данных.
- Создание форм можно выполнять автоматически (автоформы), полуавтоматически (с помощью *Мастера)* или вручную (в режиме *Конструктора).*
- Размещение элементов управления на бланке **формы авто**матизировано. В большинстве случаев при создании нового элемента запускается *программа-Мастер, с* помощью которой происходит настройка свойств элемента управления.
- Для создания печатных документов, которые содержат информацию из базовых таблиц или из результирующих таблиц, полученных по результатам работы запросов, служат специальные объекты — ***отчеты.***
- Отчеты отличаются от форм тем, что предназначены **не** для ввода данных, а только для вывода, а также тем, что создают не экранные, а печатные документы.
- Структура отчетов, как и форм, состоит из разделов и элементов управления. Проектирование отчета состоит в создании структуры его разделов и в размещении элементов управления внутри этих разделов, а также в задании связей между этими элементами и полями таблиц или запросов базы данных.
- Создание отчетов может выполняться автоматически *(автоотчеты),* полуавтоматически (с помощью *Мастера)* или вручную (в режиме *Конструктора).*
- *Таблицы, запросы, формы* и *отчеты* являются основными объектами базы данных. Их разрабатывает разработчик базы. Пользователь базы использует эти объекты без вмешательства в их структуру.
- Разработчик базы данных имеет также два типа дополнительных объектов: *макросы* и *модули.* Эти объекты создают в тех случаях, когда стандартных средств управления базой данных оказывается недостаточно для выполнения операций, необходимых заказчику системы. С помощью *макросов* создают *макрокоманды,* упрощающие наиболее утомительные операции с базой, а с помощью *модулей,* написанных на языке программирования Visual Basic, создают программные процедуры для выполнения нестандартных операций.

Практическая работа № 1

Тема: Анализ данных и разработка структуры данных. Создание структуры реляционной БД. Создание базы данных (БД).

Цель: Приобрести и закрепить практические навыки по созданию БД.

Часть I: Анализ информации, создание структуры данных и определение связей между данными.

Порядок выполнения:

1. Теоретическая часть:

Анализ первоначальных данных: В создаваемой БД нет пока ни одного объекта. Начнем с проработки ее структуры данных. Целью нашей работы является создать БД для учета транспортных средств автогаража. Основой для анализа данных будут записи из Путевых листов, которые будут вносится в таблицы и обрабатываться в запросах и отчетах.

Разберем все данные, которые будем вносить из Путевого листа:

- Номер П/листа;
- Дата;
- Лицо (организация), в чьем распоряжении был автомобиль;
- Автомобиль;
- Водитель;
- Время прибытия автомобиля;
- Время отбытия автомобиля;
- Цель поездки;
- Маршрут автомобиля;
- Пробег (км).

При этом будем использовать следующие повторяющиеся справочные данные:

- Лицо (организация), в чьем распоряжении был автомобиль;
- Автомобиль;
- Водитель;

Эти данные необходимо иметь в виде отдельных таблиц для более подробной информации о них. Так для пункта Лицо (организация), в чьем распоряжении был автомобиль необходимо записать следующие данные:

- Код заказчика;
- Ответственное лицо заказчика (кто подписывает П/лист);
- Внутренний заказчик (Да/Нет);
- Организация или Внутренняя служба (наименование);

Для пункта автомобиль будут необходимы следующие данные:

- Государственный номер автомобиля;
- Марка автомобиля;
- Год выпуска;
- Основные характеристики;

Для пункта Водитель будут необходимы следующие данные:

- Водительское удостоверение;
- Фамилия Имя Отчество
- Класс водителя;

- Дата приема на работу;
- Водительский стаж на дату приема на работу.

Затем мы должны продумать то, что на одном П/листе может быть несколько заказчиков к тому же надо данные для учета топлива. Для этого введем еще одну справочную таблицу с полями:

- Дата;
- Номер путевого листа;
- Время отбытия из гаража;
- Время прибытия в гараж;
- Показания спидометра при отбытии;
- Показания спидометра при прибытии;
- Заправлено топливом литров;
- Марка топлива;
- Замечания механика.

Мы можем анализировать информацию более подробно, но в рамках учебной работы этих данных будет достаточно.

Формирования таблиц: Теперь проведем анализ данных для создания полей таблиц:

1) Таблица Главная:

Наименования поля	**Тип данных**	**Число символов**	**Ключевое поле**
Номер записи	Счетчик		Первичный
№ Путевого листа	Число		
Дата	Дата/время	Формат	
Ответственное лицо (Лицо (организация), в чьем распоряжении был автомобиль)	Число		
Автомобиль	Текст	15	
Водитель	Текст	15	
Время прибытия автомобиля	Дата/время		
Время отбытия автомобиля	Дата/время		
Цель поездки	Текст	80	
Маршрут автомобиля	Текст	100	
Пробег (км)	Числовой		
Замечания (особые отметки)	Поле MEMO		

2) Заказчик:

Наименование поля	**Тип данных**	**Число символов**	**Ключевое слово**
Код заказчика	Счетчик		Первичный
Ответственное лицо заказчика (кто подписывает П/лист)	Текст	30	
Внутренний заказчик (Да/Нет)	Логический		
Организация или внутренняя служба (наименование)	Текст	50	
Заметки	Поле МЕМО		

3) Автотранспорт:

Наименование поля	Тип данных	Число символов	Ключевое слово
Государственный номер автомобиля	Текст	15	Первичный
Марка автомобиля	Текст	25	
Год выпуска	Числовое		
Основные характеристики	Поле МЕМО		

4) Водитель:

Наименование поля	Тип данных	**Число символов**	**Ключевое слово**
Водительское удостоверение	Текст	15	Первичный
Фамилия Имя Отчество	Текст	50	
Класс водителя	Текст	12	
Дата приема на работу	Дата/время	Формат	
Водительский стаж на дату приема	Число		
Заметки	Поле МЕМО		

5) Путевой лист:

Наименование поля	Тип данных	Число символов	Ключевое слово
Номер записи	Счетчик		Первичный
Номер Путевого листа	Текст	15	
Дата	Дата/Время	Формат	
Время отбытия из гаража	Дата/Время	Формат	
Время прибытия в гараж	Дата/Время	Формат	
Показания спидометра при отбытии	Число		
Показания спидометра при прибытии	Число		
Заправка топливом литров	Число		
Марка топлива	Текст	5	
Замечание механика	Поле МЕМО		

Определение связей: Чтобы данные из таблиц взаимодействовали как нам надо, мы должны создать реляционную модель нашей БД. И так, операционной таблицей будет Главная (в ней будет заносится оперативная информация), особое место у нас занимает таблица Путевые листы: в ней ведется оперативный учет, но она, прежде всего, служит справочником для Главной. Все другие таблицы будут носить справочный характер. Поэтому каждая запись справочной таблицы будет нести уникальную информацию, уникальность которой будет заключена в ключевых полях, отмеченных как «Первичный ключ». Это либо номер определенного документа, либо порядковый номер записи. Для исключения лишней информации в главной таблице, мы будем вносить в нее только значения наших ключей из справочной таблиц. Тогда для создания целостности информации и возможности делать ссылки на справочные записи необходимо установить связь программными средствами. Тип связи мы определим как «Один-ко-многим», где одной записи в справочной таблице будут соответствовать много записей из Главной.

Установим эти связи в следующем виде:

№	Таблица-источник	Поле-источник	Поле в главной таблице
1	Ответственное лицо	Код заказчика	Ответственное лицо
2	Автомобиль	Государственный номер автомобиля	Автомобиль
3	Водитель	Водительское удостоверение	Водитель
4	Путевой лист	Номер записи	№ Путевого листа

Данные в связанных полях должны иметь один формат. Если в справочной таблице ключевое поле имеет формат Счетчик, то в главной – Число, если в справочной – Текст, тогда соответственно и в главной тоже Текст. Для текстовых полей должно совпадать и количество символов.

Часть II. Создание БД.

4. **Практическая часть:**

2.1 Создание БД. Практика: Запустить Microsoft Access. В появившимся окне создания базы данных выберем Новая база (Рис.1) и нажмем кнопку OK.

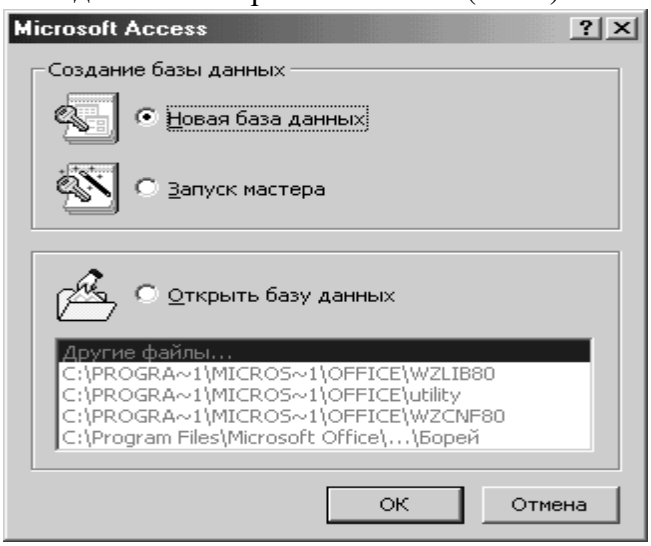


Рис.1

В новом окне нам необходимо присвоить имя новой БД и сохранить ее в нужной папке(Рис.2). **Присвоим имя файла свою фамилию и выберем паку Практика на диске С:** в (она указывается в верхней части окна под пунктом Папка). Тип файла оставим по умолчанию Базы данных (.mdb). Нажмем кнопку Создать – БД будет создана .

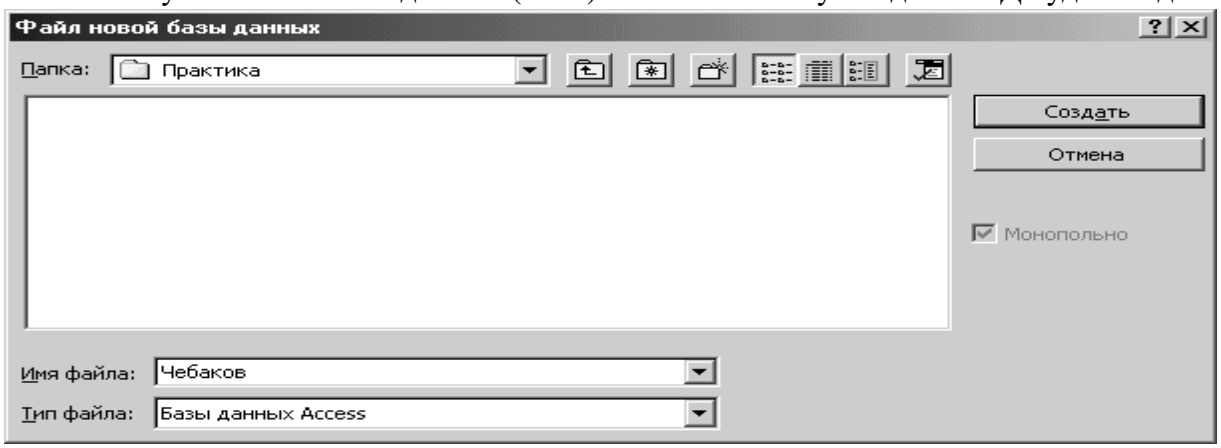


Рис.2

Часть III: Создание БД Путевые листы с помощью конструктора.

4. **Создание таблиц:**

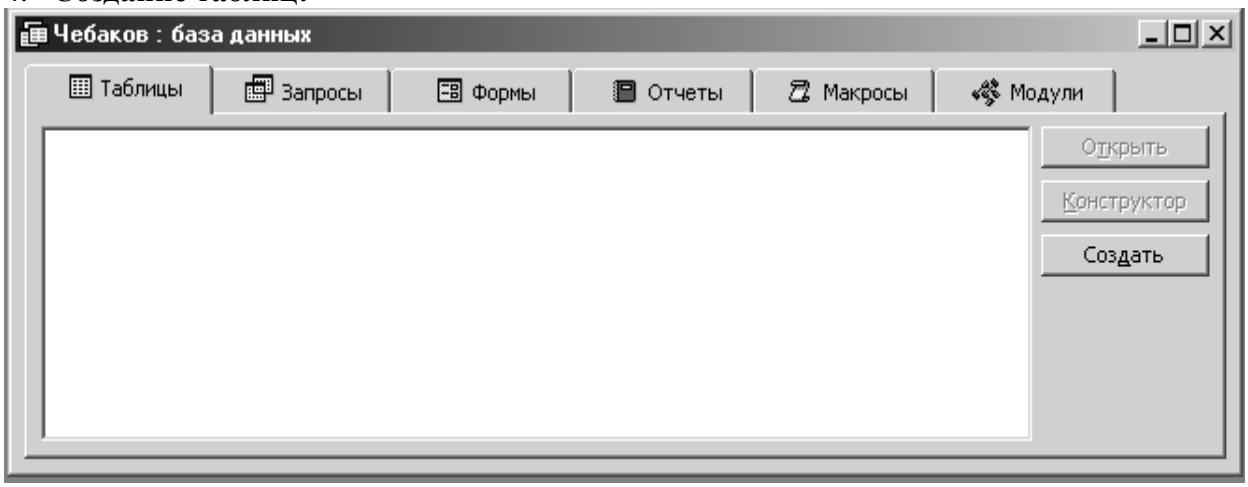


Рис.3

Когда мы откроим БД, открывается окно БД (Рис.3). В окне БД выберем вкладку Таблицы. Нажмем кнопку Создать. Появится окно с выбором методов создания таблицы:

- Режим таблицы;
- Конструктор;
- Мастер таблиц;
- Импорт таблиц;
- Связь с таблицами.

Выберем режим конструктор. В полях конструктора мы будем заносить данные для таблицы Главная (Рис.4).

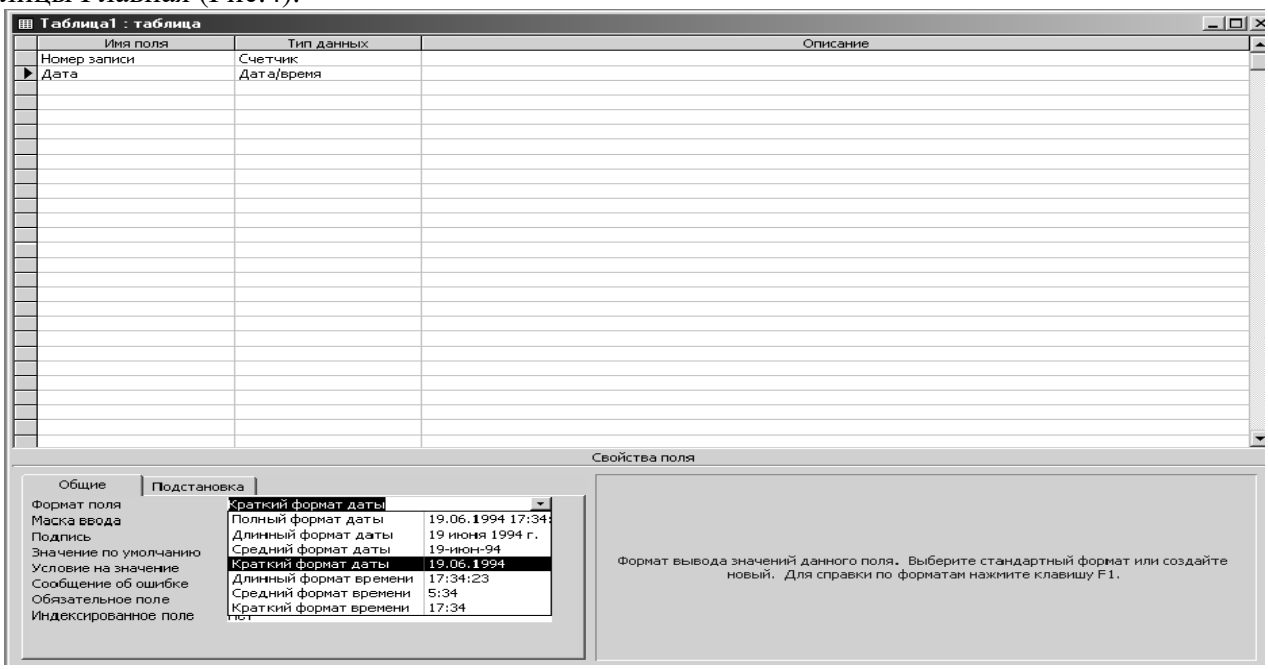


Рис.4

Вся информация для названий, типов и длины полей дана в п.1.2 Части I Практической работе №1. В поле «Имя поля» вносим название полей. В «Типы данных» заносим типы полей.

- **Для текстовых полей в нижней части Конструктора на вкладке Общие укажем длину поля, в соответствии с ранее указанной.**
- **Для Даты/времени укажем формат поля (Рис.4) и Маску ввода.**
- **Для числа укажем длину поля либо длинное целое либо целое.**

Не забывайте устанавливать первичный ключ в таблице. Для этого помечаем поле которое будет ключевым. Затем выбираем на панели инструментов БД значок ключа и нажимаем на его.

После заполнения полей выйдем из Конструктора: нажмем на крестик в верхнем правом углу. Конструктор высветит окно для сохранения таблицы. Наберем в окошечке имени таблицы вместо Таблица 1 – Таблица Главная.

Аналогично создадим следующие таблицы:

- Заказчик;
- Автотранспорт;
- Водители;
- Путевые листы.

Часть IV: Создание связей между таблицами.

4. Связывание таблиц:

На панели инструментов БД нажмем на кнопку Схема данных БД и создания связей. В этом окне (Рис.5) выберем каждую таблицу нажатием кнопки Добавить.

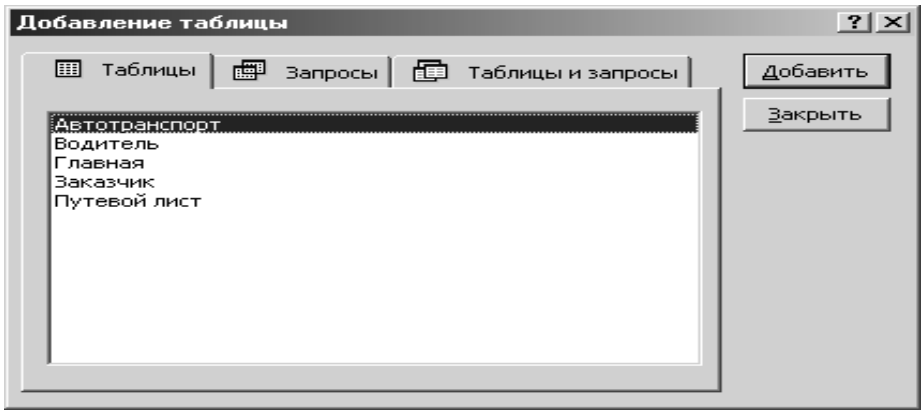


Рис.5

Затем нажмем кнопку Закрыть и приступим к редактированию связей.

1. На рисунке 6 показаны таблицы БД, расставленные так, чтобы было удобно рассматривать связи.
2. Из таблицы Путевой лист захватим указателем мышки поле «Номер Записи» и перетащить его в таблицу Главная на поле «№ Путевого листа».

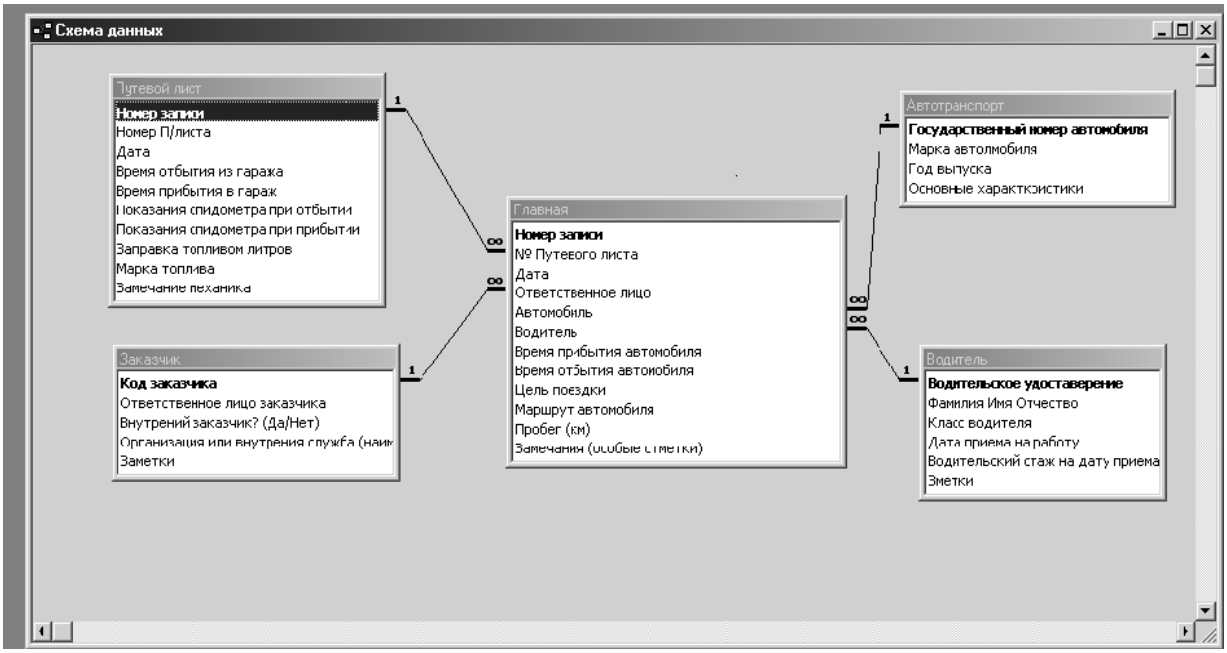


Рис.6

В появившемся диалоговом окне (Рис.7)

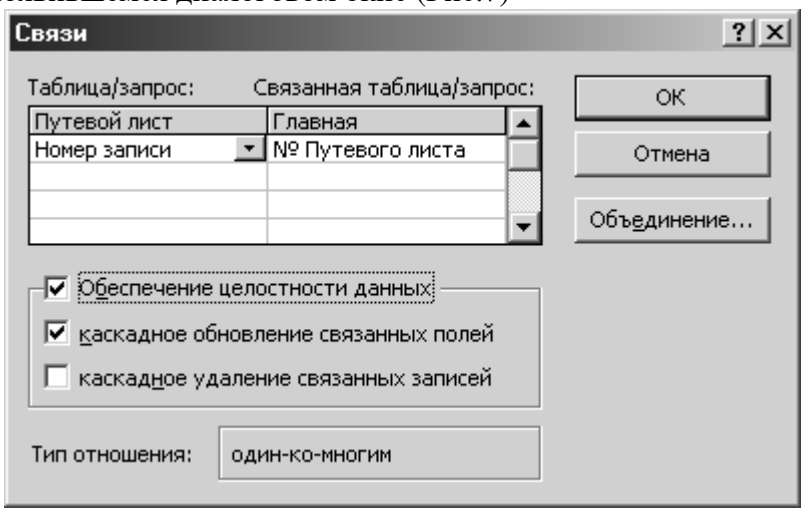


Рис.7

Установим флажки для пунктов «Обеспечение целостности данных» и «Каскадное обновление связанных полей» Нажмем кнопку Ok. На схеме появится связь один-ко-многим (1-∞) (Рис.5). Все следующие связи установим тем же образом.

Практическая работа №2

Тема: Создание запросов базы данных (БД) с помощью Мастера, их редактирование и внесение данных.

Цель: Приобрести и закрепить практические навыки по созданию базы данных.

Часть I: Создание запроса с помощью Мастера.

Порядок выполнения:

1. Создание запросов :

Создание простого запроса: Выберем в окне БД вкладку Запросы. Нажмем кнопку Создать. Появится окно Новый запрос (Рис.8).

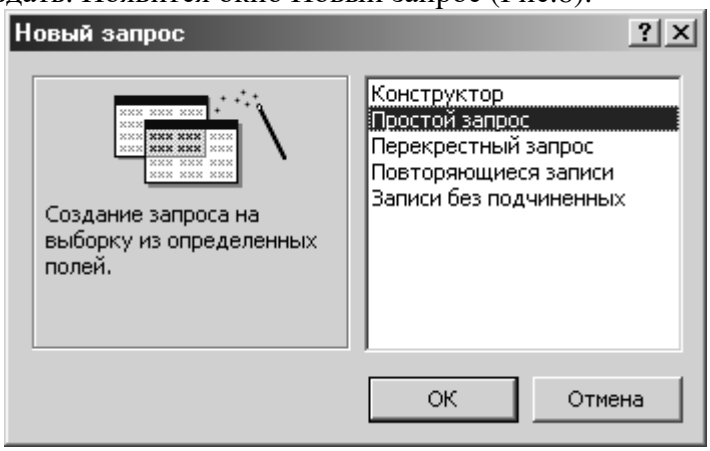


Рис.8

В данном окне мы выберем из меню Простой запрос и нажмем Ok (Все данные запроса мы будем выбирать из одной таблицы Главная). В открывшемся окне (рис.9) выберем нужную нам таблицу Главная. Затем, помечая с помощью курсора нужное поле (черная метка), указателем > перетащим в правое нижнее окно следующие поля:

- Дата;
- № Путевого листа;
- Ответственное лицо;
- Водитель;
- Автомобиль;
- Пробег;
- Цель поездки.

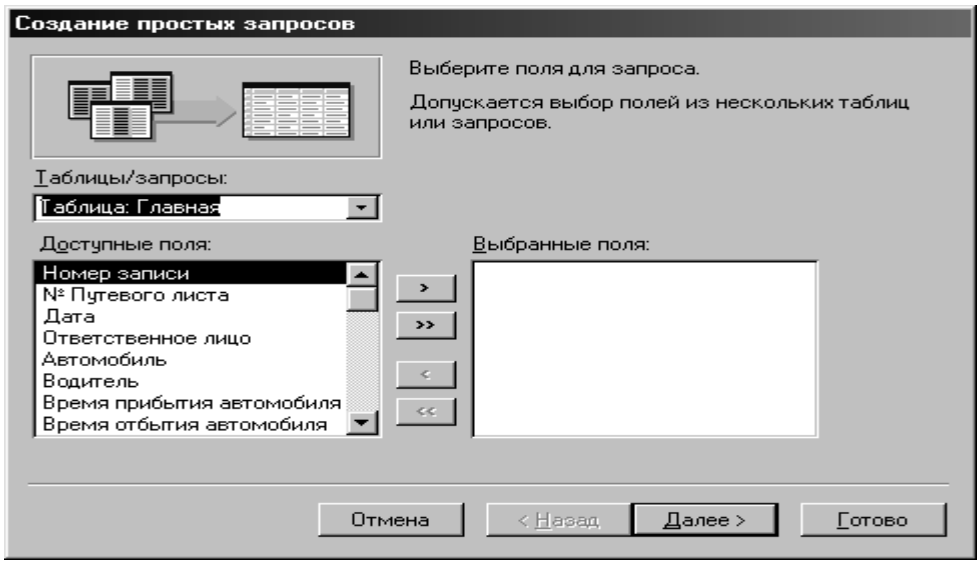


Рис.9

Нажмем кнопку Далее >.
Откроется следующее окно (фрагмент см. на рис.10).

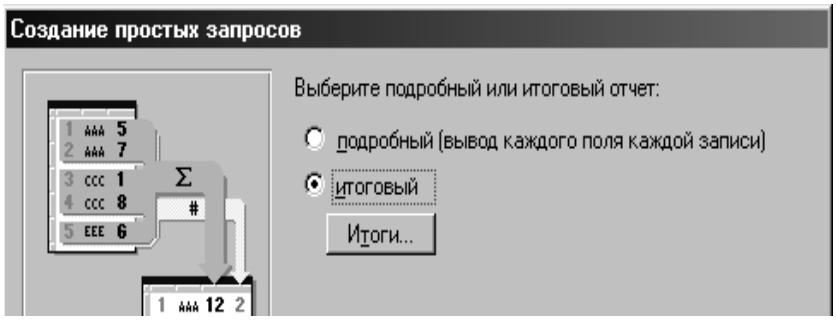


Рис.10

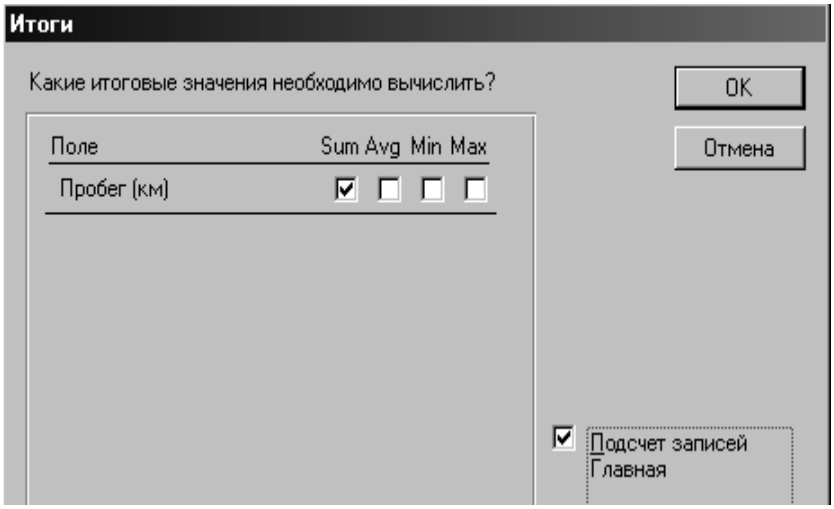


Рис.11

Выбираем переключатель «Итоговый» и нажимаем кнопку Итоги для подробного описания какие итоговые значения надо получить. На рисунке 11 показано, по какому полю можно получать итоговые значения.

Для нашего случая это одно поле Пробег (имеет числовой формат). Можно выбрать для него значение Sum – получение суммы для отобранных полей. В правом нижнем углу формы отметим флажок Подсчет записей Главная, чтобы вести отсчет отобранных записей. Введем указанные изменения и нажмем кнопку Ok. После этого появится окно (Рис.10).

Нажмем кнопку Далее>. Новое окно изображено на рисунке 12. Выбираем интервал группировки дат по Дням, чтобы отслеживать записи ежедневно. Нажмем кнопку Далее>.

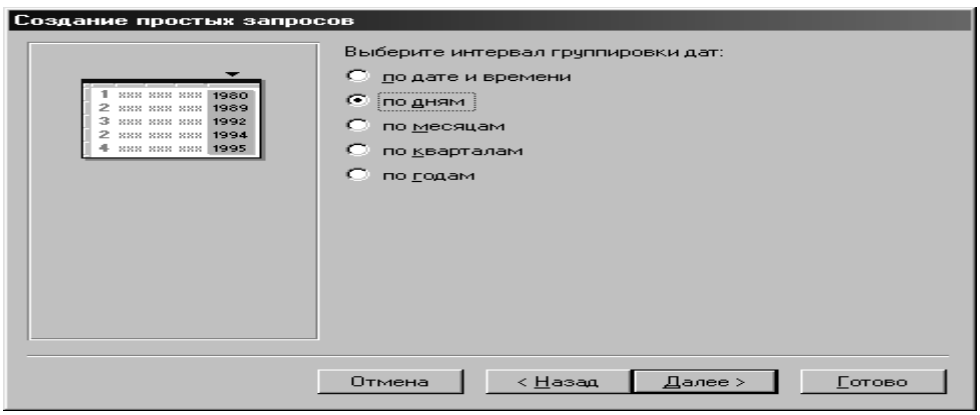


Рис.12

Новое окно изображено на рисунке 13, Присвоим запросу имя Главная. Запрос, затем установим переключатель в значение «Открытие результатов запроса». Флажок по выводу справки оставим выключенным. Нажмем кнопку Готово и… запрос готов.

Запрос откроется в виде таблицы данных. Пока мы не заполняли таблицы данных в запросе не будет.

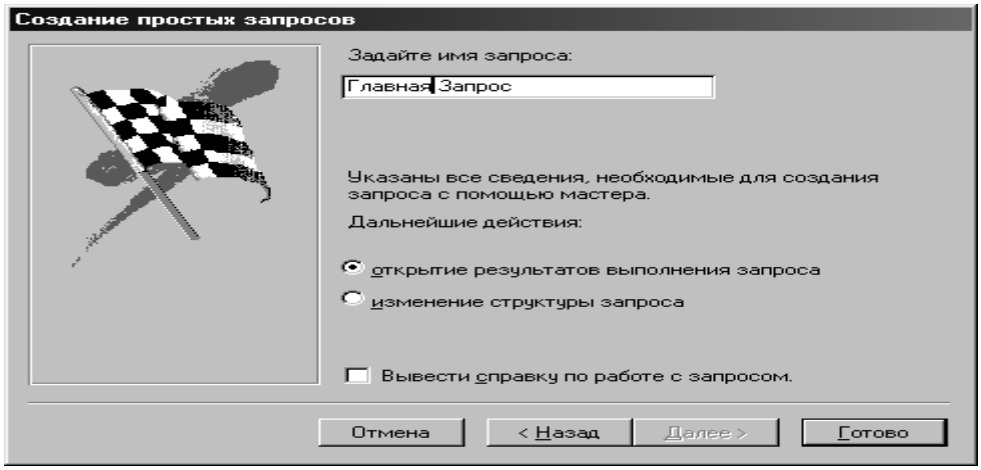


Рис.13

Часть II: Ввод данных в режиме таблиц.

2. Ввод данных:

Заполнение справочных таблиц: Запустить свой файл. На вкладке Таблицы откроим таблицу Заказчик и внесем данные прямо в поля (рис.14):

Заказчик : таблица

Код заказчи	Ответственн	Внутрений з	Организация или внутрения	Заметки
1	Петров Н.М.	☑	Директор	
2	Васильев А.Б.	☑	Энергетика	
3	Шухова Т.А.	☑	Водозаборные пункты	
4	Банинка Г.Ф.	☑	Бухгалтерия	
5	Сомов П.Д.	☑	Котельная	
6	Хахлов М.Н.	☑	Автогараж	
7	Маркин А.А.	☑	Эксплуатация тепловых сетей	
8	Савенков И.Г.	☑	Слесаря	
9	Пупков Б.И.	☑	Очистительные сооружения	
10	Лихой В.В.	☑	Эксплуатация жилого фонда	
11	Чибис Т.Я.	☑	Стройцех	
12	Дорогин Л.Н.	☐	Школа	
13	Лещенко Я.И.	☐	Администрация с/о	
* (Счетчик)		☐		

Рис.14

На вкладке Таблицы откроем таблицу Автотранспорт и внесем данные прямо в поля (Рис.15):

Автотранспорт : таблица

Государственн	Марка автол	Год выпуска	Основные характкристики
а439 МЮ рус 50	Зил-433362 фур	1995	Пробег на 01.01.01 г. = 86754 км
а677 МП рус 50	Газ-3306 борт.	1994	Пробег на 01.01.01 г. = 100734 км
к121 Ми рус 50	УАЗ-31514-012	1994	Пробег на 01.01.01 г. = 134623 км
к211 МА рус 50	Зил-130 борт.	1993	Пробег на 01.01.01 г. = 156432 км
к215 МА рус 50	Газ-3309 борт.	1993	Пробег на 01.01.01 г. = 131957 км
о856 МЖ рус 50	Газ-53 фургон	1985	Пробег на 01.01.01 г. = 264543 км
с887 МЮ рус 50	Ваз-2106	1995	Пробег на 01.01.01 г. = 145665 км
с888 МЮ рус 50	Ваз-2106	1995	Пробег на 01.01.01 г. = 133745 км
т114 МЕ рус 50	Паз-3205	1992	Пробег на 01.01.01 г. = 100456 км
т383 МЕ рус 50	Зил-130 груз	1992	Пробег на 01.01.01 г. = 141267 км
т384 МЕ рус 50	Зил-130 груз	1992	Пробег на 01.01.01 г. = 112765 км
т391 МЕ рус 50	Зил-130 груз	1992	Пробег на 01.01.01 г. = 125432 км
т392 МЕ рус 50	Зил-130 груз	1992	Пробег на 01.01.01 г. = 133456 км
т654 МЕ рус 50	Зил-130 дорож.	1992	Пробег на 01.01.01 г. = 175345 км
		0	

Рис.15

На вкладке Таблицы откроем таблицу Водитель и внесем данные в поле (Рис.16): Справочные данные внесены. Теперь мы можем работать с операционной таблицей, но для этого необходимо создать форму и получить возможность внесения данных прямо из справочных таблиц.

Водитель : таблица

Водительско	Фамилия Имя Отчест	Класс водит	Дата прием	Водительск	Зметки
50 МАМ 136001	Дугин А.Д.	1	26.05.91	5	
50 МАМ123000	Соев М.И.	1	01.08.93	12	
50 МАМ124001	Марков С.П.	1	06.06.87	14	
50 МАМ125002	Ляпишев Г.П.	1	14.11.89	8	
50 МАМ126003	Симонов И.З.	1	12.03.83	7	
50 МАМ127003	Трошин К.Н.	1	30.05.91	5	
50 МАМ129003	Ерохин С.М.	1	01.03.86	5	
50 МАМ130002	Ерохин И.М.	1	05.07.86	2	
50 МАМ131002	Баринов В.В.	2	11.06.96	2	
50 МАМ132001	Паршиков Т.М.	1	09.08.88	6	
50 МАМ133001	Ситников Д.И.	1	07.05.80	3	
50 МАМ134002	Базаркин Г.П.	1	25.11.91	4	
50 МАМ135001	Строганов Н.И.	1	03.03.83	3	
50 МАМ29003	Мисюк С.Г.	1	23.01.93	4	
				0	

Рис.16

Практическая работа №3

Тема: Создание форм базы данных (БД) с помощью Мастера, их редактирование и внесение данных.

Цель: Приобрести и закрепить практические навыки по созданию базы данных.

Часть I: Создание формы с помощью Мастера.

Порядок выполнения:

1. Создание формы:

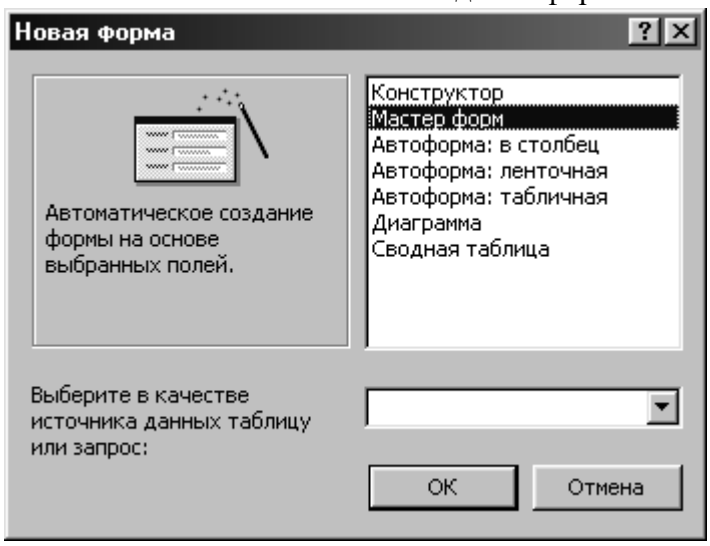


Рис.17

1.1 Запуск Мастера форм: Запустим редактор. Откроем БД и в окне БД выберем вкладку Формы. В вашей БД формы нет, поэтому для вас есть один выбор: Создать Форму. Нажмем кнопку Создать и перед нами откроется окно, показанное на рис.17 Выберем из меню вариант Мастер форм. Нажмем кнопку **Ok.** Откроется окно на Рис.18.

1.2 связывание новой формы с таблицей для ввода данных. В открывшемся окне (Рис.18) мы выбираем «Таблица Главная» в верхнем окошечке слева.

В окне ниже появится список полей данной таблицы. Так как мы создаем форму, для ввода данных в эту таблицу, то эта форма должна быть связана со всеми полями таблицы Главная. Нажмем кнопку с двойной стрелкой >> для переноса сразу всех полей в форму. Имена всех полей перенесутся. Нажмем кнопку далее >. Откроется следующее диалоговое окно.

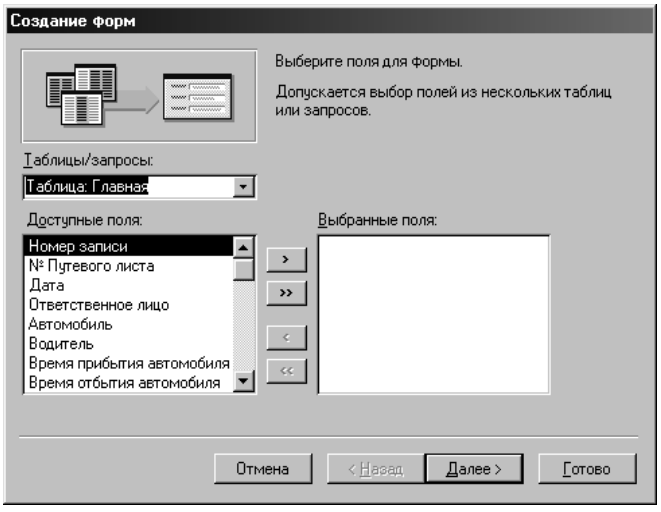


Рис.18

1.3 Определение внешнего вида формы:

На рис.19 в открывшемся окне мы должны выбрать вид формы.
Для нашего случая (ввод данных в виде одной записи) необходимо выбрать выключатель «в один столбец» (в в правой половине окна). Нажмем кнопку Далее>. Откроется следующее диалоговое окно.

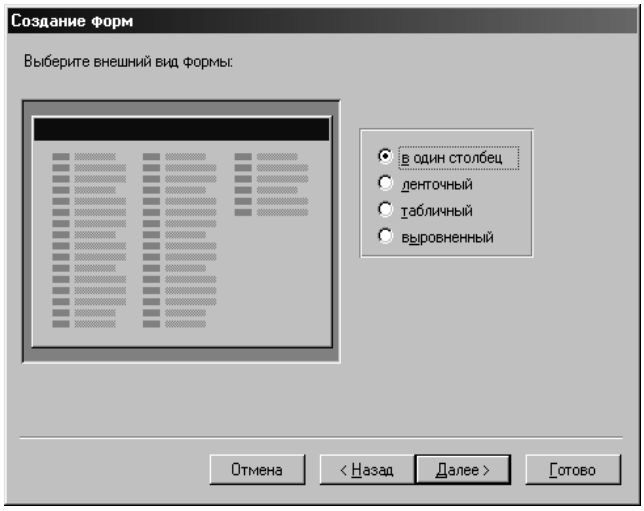


Рис.19

В следующем окошке мы определим стиль оформления самой формы (Стандартное меню этого окна показано на Рис.20). Выберем обычный стиль. Нажмем кнопку Далее>. Откроется окно, в котором мы выберем имя формы – Главная и переключатель Открытые формы для просмотра или ввода данных. Нажмем кнопку Готово. Откроется форма для ввода значений в таблицу Главная.

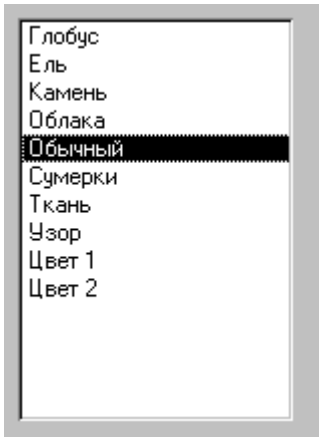


рис. 20

Такая форма не совсем удобная для ввода, поэтому мы должны ее отредактировать. Редактирование форм проводится в Конструкторе.

Часть II: Редактирование формы с помощью конструктора.

2. Внесение изменений:

2.1 Открыть Конструктор формы: В окне БД вашей программы откроем вкладку формы и, пометим форму Главная, нажмем форму Конструктор, На Рис.21 показан формы в режиме Конструктор.

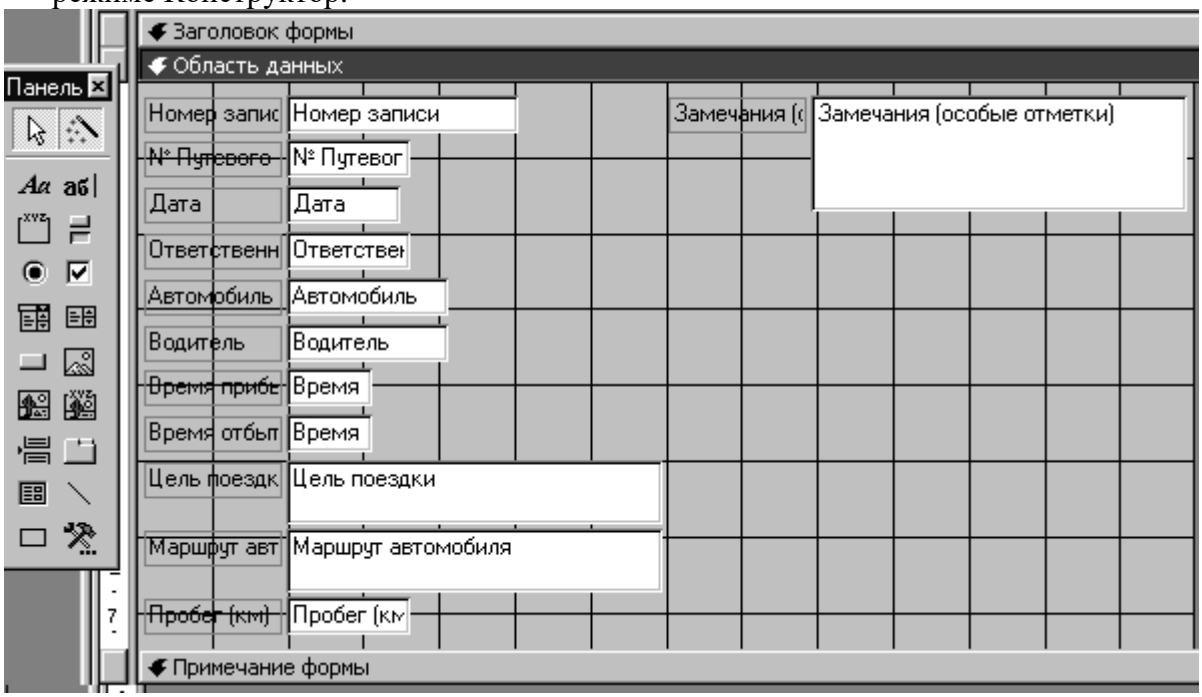


Рис.21

Просмотр полей формы: В открывшемся Конструкторе формы курсором мышки подведем под окошечко поля и двойным щелчком вызовем Инспектор объекта (Рис.22), в котором даны все свойства выделенного поля (объекта). Для поля Номер записи закроем доступ к его изменению. В этом поле будет проводиться автоматический подсчет записей (Счетчик). Закроим окно Инспектор объекта.

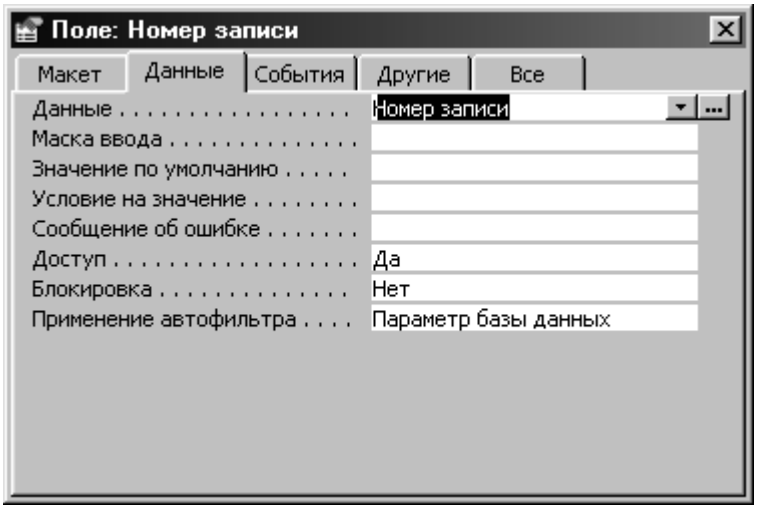


Рис.22

Замена простых полей на поля со списком для тех данных, которые мы будем вносить из справочной таблице: На панели объектов необходимо нажать кнопку Мастера объектов, чтобы запускались Мастера при размещении объектов на форму. На панели объектов выберем объект Поле со списком и поместить его на форму. При этом откроется Мастера поля со списком (Рис.23). Зададим способ, по которому мы будем создавать список справочных данных: отметим выключатель Таблица или запрос содержат значения, которые использует поле со списком (Рис.23). Нажмем кнопку Далее>.

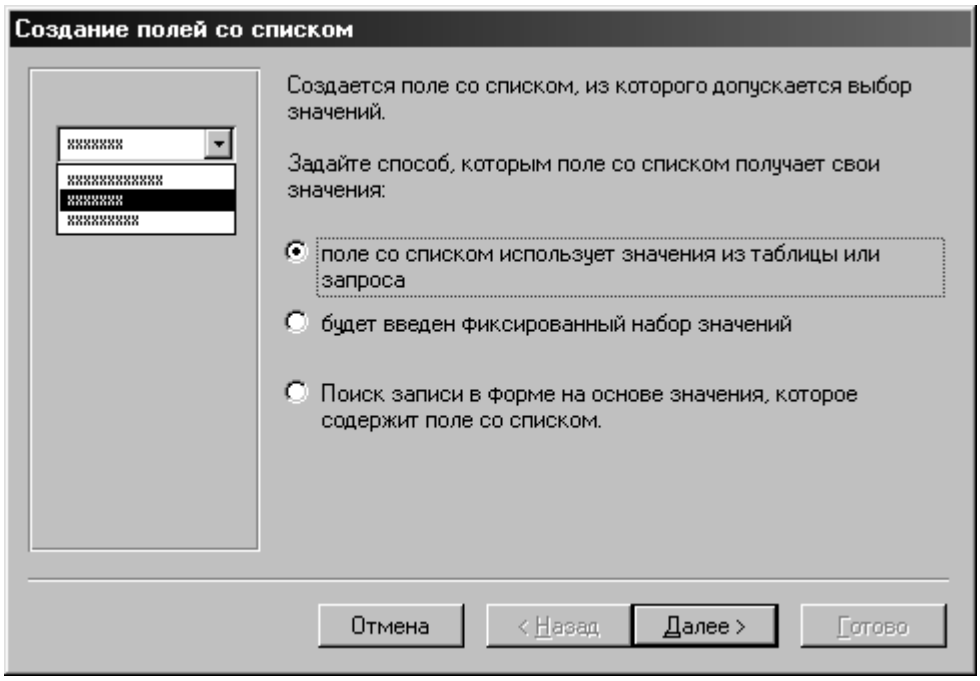


Рис.23

В открывшемся окне (Рис.24) выбираем выключатель Таблица (внизу окна в группе Показатели) и отмечаем нужную таблицу: Путевой лист. Нажмем кнопку Далее>.

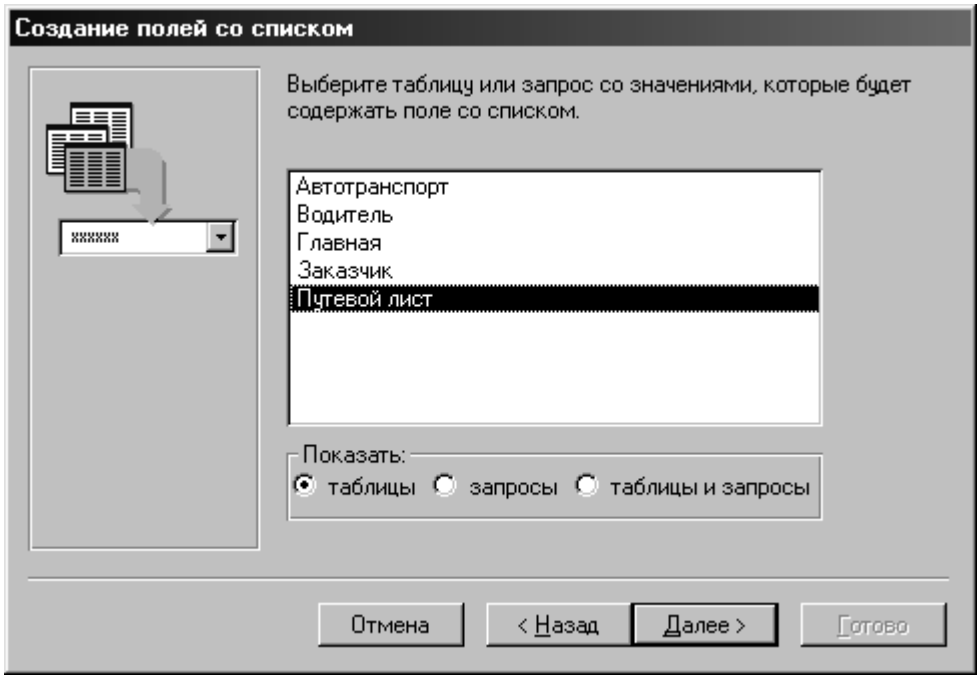


Рис.24

Перейдем для выбора нужных полей в следующем окне. Перетаскиваем два поля (Рис.25) из левого окошка в правое кнопкой > в последовательности:

- № Путевого листа
- Дата.

В данном случае поле Дата выбирается для детализации информации: Чтобы выбрать номер путевого листа, необходимо знать за какое число этот путевой лист. Нажмем кнопку Далее>.

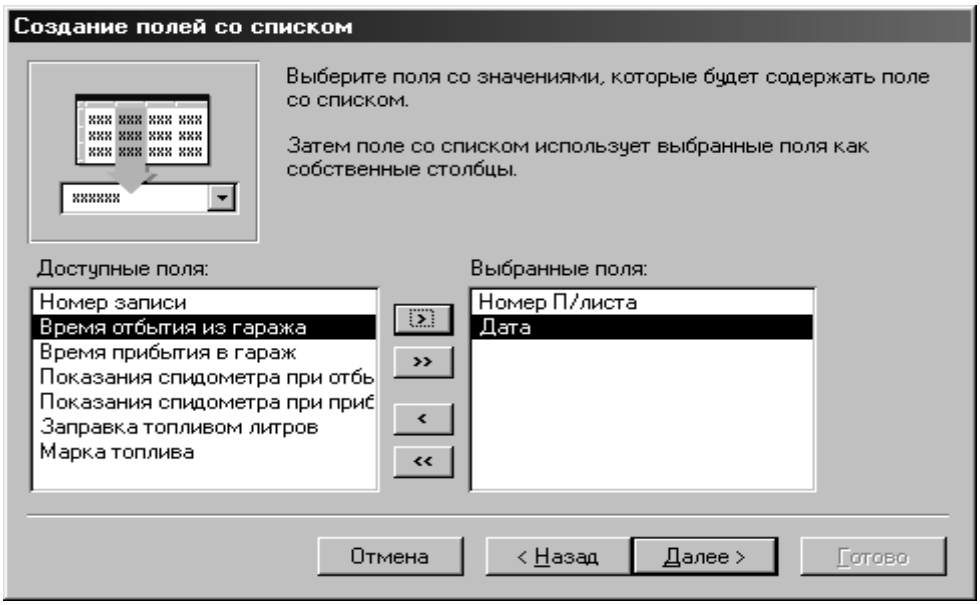


Рис.25

В следующем окне установим ширину столбцов, которые будут выводится в списке. Отметим флажок Скрыть ключевой столбец. Нажмем кнопку Далее> (Рис.26).

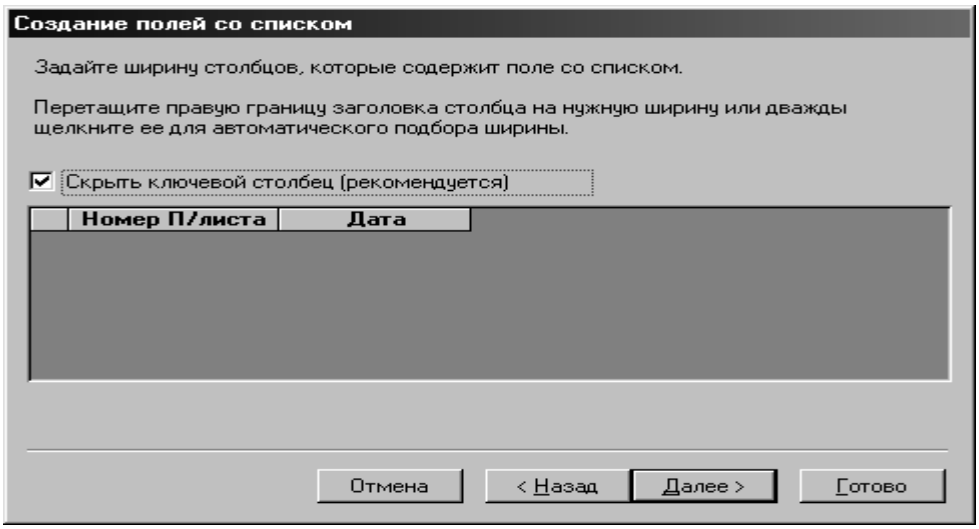


Рис.26

В следующем окне отметим выключатель Сохранить в поле и выберем поле из таблицы

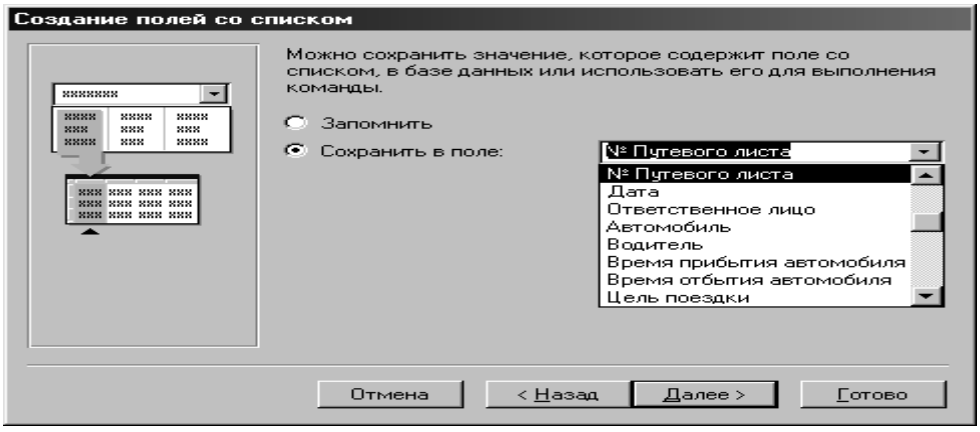


Рис.27

Главная: № Путевого листа (Рис.27).

Нажмем кнопку Далее> и перейдем к окну, в котором определим подпись к данному полю формы. На рисунке 28 показана это окно. Затем нажмем кнопку Готово и … Мы создали поле со списком для переноса № Путевого листа из таблицы Путевой лист в таблицу Главная.

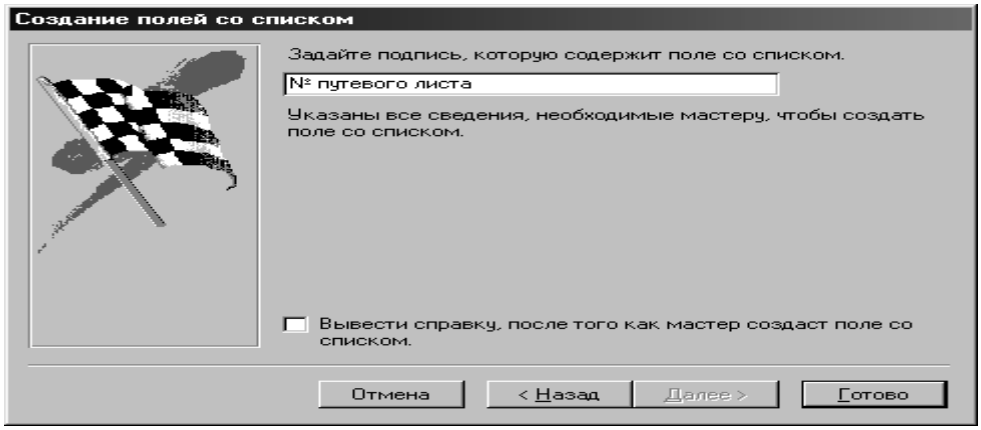


Рис.28

Создание поля со списками для переноса данных из полей таблиц Автотранспорт, Водители, Заказчики: Выполняются эти действия **идентично описанному выше**. Отметим лишь те пары полей, которые будут указаны в списке для справочной информации.

- ***Заказчик: Номер заказчика (связанное) и Ответственное лицо (Справочное);***
- ***Водитель: № Удостоверения (связанное) и Ф.И.О. водителя (Справочное);***
- ***Автотранспорт: Государственный номер (связанное) и Марка автомобиля (Справочное).***

Старые поля, созданные Мастером формы можно не удалять, удалив только их надписи. Они пригодятся для контроля ввода данных через поля со списком, Удобней всего их разместить справа от поля со списком одного имени. Текст в этих полях можно выделить цветом или полужирным шрифтом (Рис. 29).

Для автоматизации ввода времени Время отправление и Время прибытия необходимо

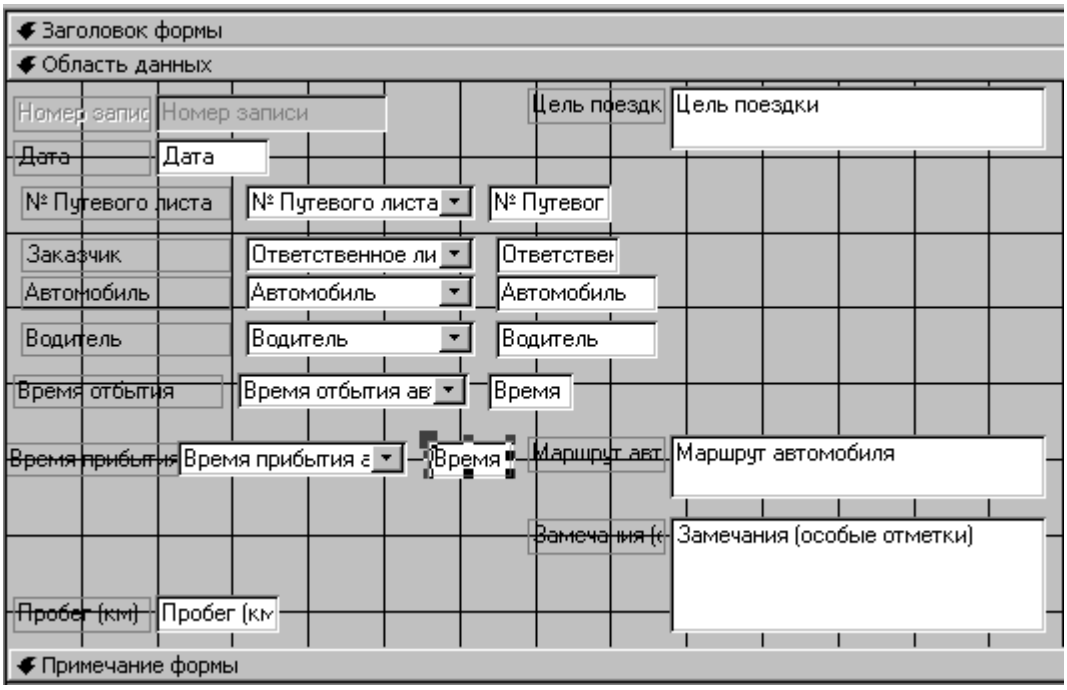


Рис.29

создать поля списком с фиксированным набором данных (Рис. 29). Например с интервалом 10 минут от 6:00 до 18:00 часов (как наиболее употребляемый интервал времени для работы).

Спланируем поля как показана на Рис.29. Форма для ввода данных в таблицу Главная практически готова.

Практическая работа №4

Тема: Ввод в форму элементов управления и создание автоматизации управления формой.

Цель: Приобрести и закрепить практические навыки в построении форм.

Часть I: Ввод в форму элементов управления.

Порядок выполнения:

1. Анализ и проектирование элементов управления:

Управление формами осуществляется с помощью командных кнопок. Есть несколько способов задания команд. Выберем для этого Мастер Кнопок.

Разделы команд	Команды	Кнопки	Дополнения
Переходы по записям	Первая запись	Начало	
	Предыдущая запись	Назад	
	Следующая запись	Вперед	
	Последняя запись	Конец	
Обработка записей	Добавление записей	Новая	
	Восстановление записи	Отменить	
	Сохранить запись	Сохранить	
	Удаление записи	Удалить	
Работа с формой	Закрыть форму	Закрыть	Для работы с другими объектами приложения
Приложение	Выход из приложения	Выход	

1.1 Определение функции для управления Формой:

Прежде чем начать проектировать кнопки, мы должны определиться в тех функциях, которые они будут выполнять. Мы должны контролировать ввод записей и возможность их изменять, создать удобное передвижение по записям. Дополнительно включим команду по закрытию формы.

1.2 Открытие формы в режиме Конструктора и подготовка ее к работе: Мы находимся в нашей программе в окне БД, затем откроем вкладку формы и, активизировать форму Главная, нажмем кнопку Конструктор.

Затем перейдем в нижнюю часть формы и курсором мышки раздвинем Область данных формы на два шага сетки. На эту свободную часть формы будем помещать кнопки.

1.3 Проектирование кнопки: В режиме Конструктора формы на панели элементов нажмем кнопку Кнопка. Затем поместим кнопку на форму. После этого откроется окно Мастера кнопок (Рис.30).

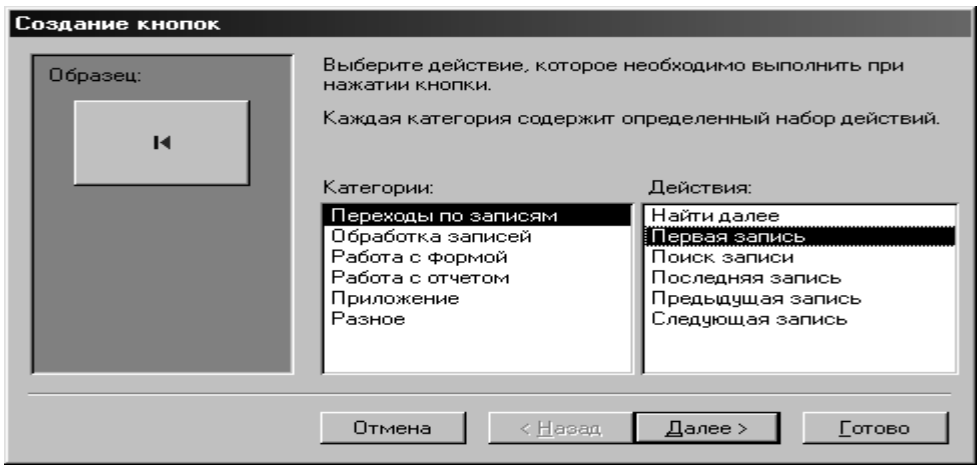


Рис.30

В соответствии с таблицей выбираем раздел Переходы по записям в левой окне формы. В правом выбираем команду Первая запись. Нажмем кнопку Далее>.

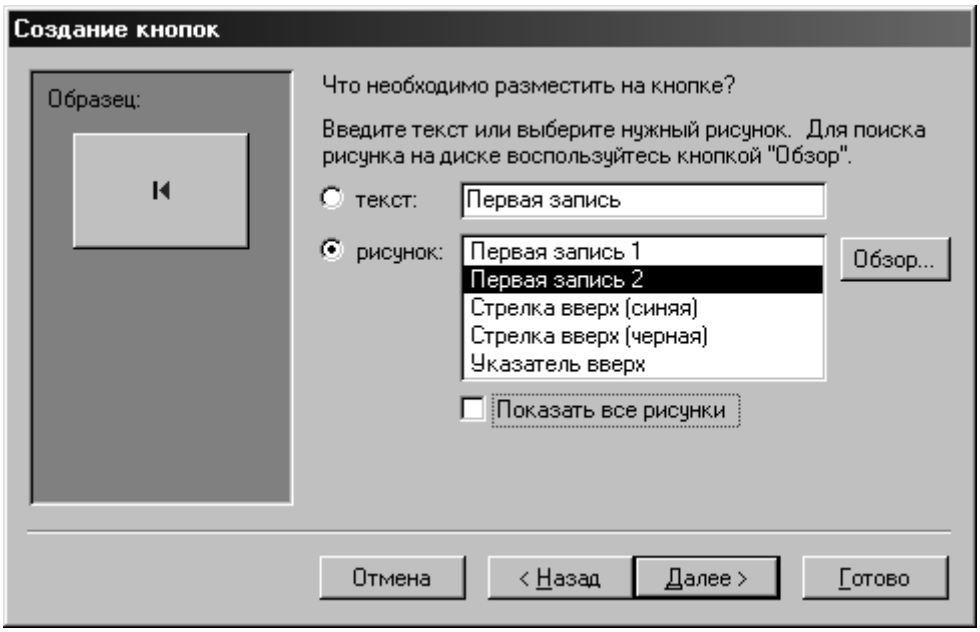


Рис.31

В следующем окне выбираем как нам обозначить кнопку, Установим переключатель на Рисунок и выберем рисунок Первая запись (вторая строчка сверху Рис.31). Нажмем кнопку Далее>.

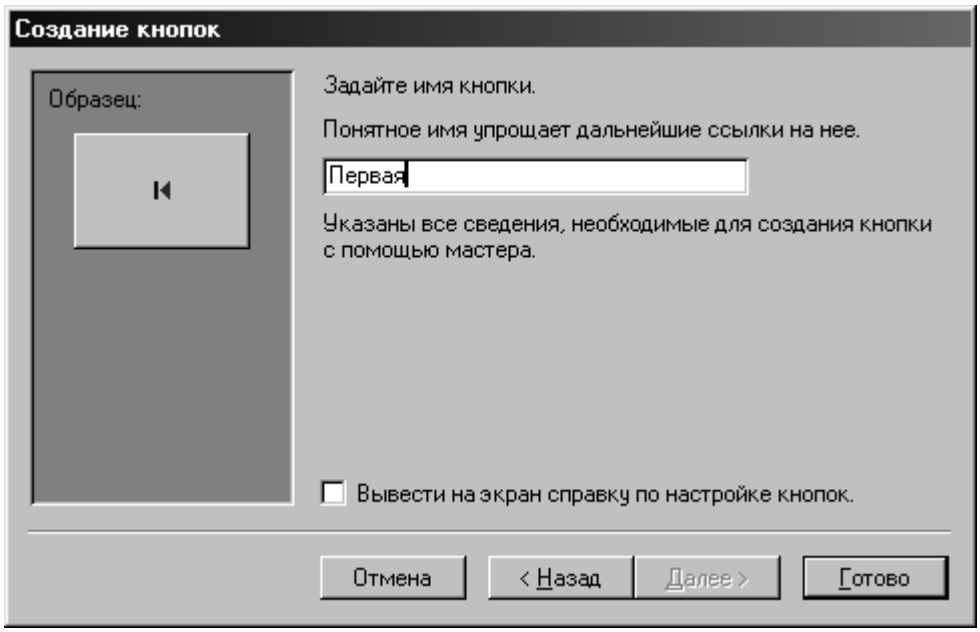


Рис.32

В следующем окне(рис.32) мы задаем наименование кнопки. В верхнем поле введем наименование «Первая» . Нажмем кнопку Готово. Кнопка Первая создана. Ее вид показан на Рис.33

Далее мы создаем все последующие кнопки описанные в таблице, используя Мастер кнопок. Затем ваша задача расположить эти кнопки на форме для удобства пользования. Окончательный вид формы данных в таблицу Главная показана на рисунке 33.

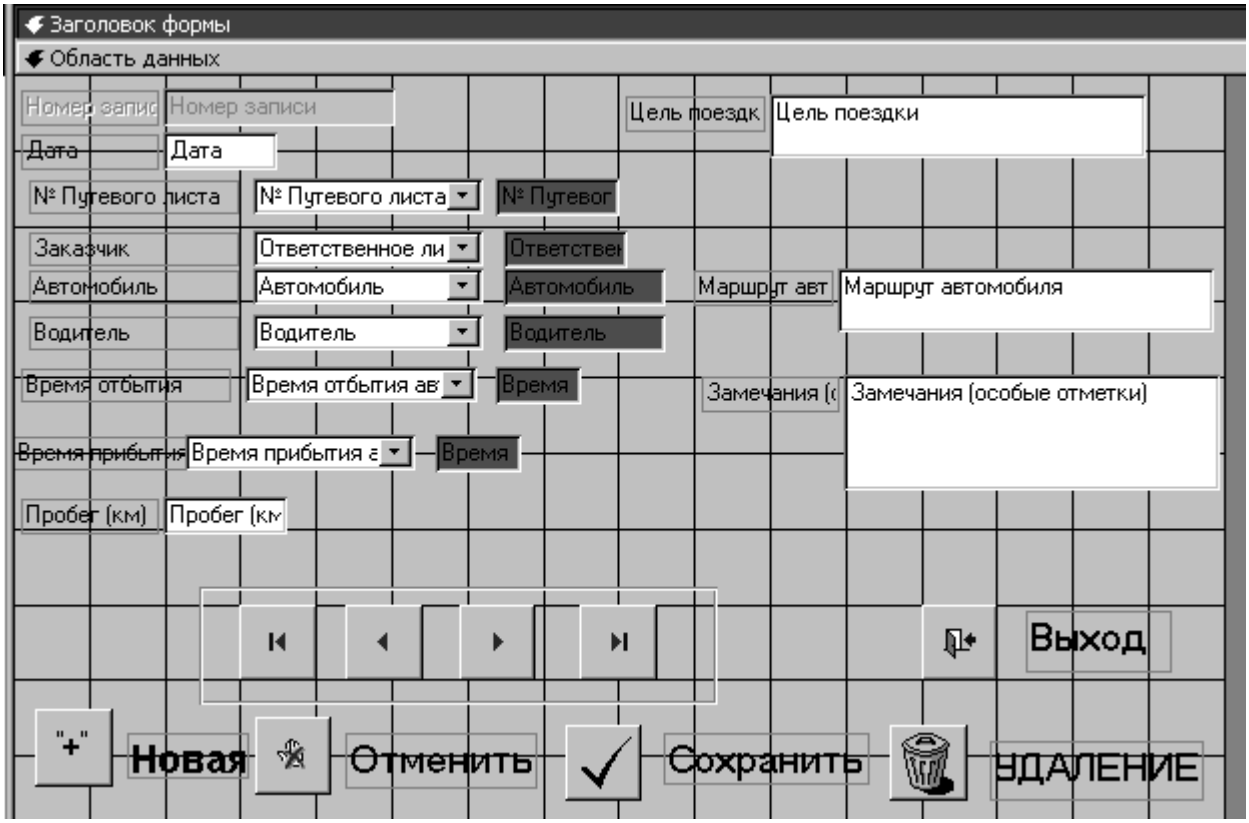


Рис.33

Часть II: Создание формы для ввода путевых листов.

2. Анализ и проектирование элементов управления формы Путевые листы:

2.1 Разбор структуры данных, которые мы будем вводить через форму: Когда мы сделали Главную форму, мы можем добавить в наш проект форму для ввода данных в таблицу Путевые листы. Это необходимо потому, что путевые листы вносятся каждодневно на каждый автомобиль и поэтому эти данные для удобства лучше вносить через форму.

Примечание: В таблице путевые листы не указаны автомобиль и водитель потому, что номер путевого листа будет уникальным для каждого автомобиля на ту дату, которая проставлена в таблице.

Номер записи автоматически прибавляется с добавлением записи в таблицу, поэтому как и в Главной форме мы будем его только отслеживать, но не изменять.

Напомним структуру таблицы Путевые листы:

Наименование поля	Тип данных	Число символов	Ключевое слово
Номер записи	Счетчик		Первичный
Номер Путевого листа	Текст	15	
Дата	Дата/Время	Формат	
Время отбытия из гаража	Дата/Время	Формат	
Время прибытия в гараж	Дата/Время	Формат	
Показания спидометра при отбытии	Число		
Показания спидометра при прибытии	Число		
Заправка топливом литров	Число		
Марка топлива	Текст	5	
Замечание механика	Поле MEMO		

2.2 Создание основы формы с помощью Мастер форм: Откроем в окне БД вкладку Формы и нажмем кнопку Создать. Затем в открытом окне (Рис.17) выберем Мастер форм и нажмем кнопку **Ok.** Затем в открывшемся окне (Рис.34) выбираем в верхнем поле Таблица Путевые листы, а из нижнего кнопкой >> перенесем все поля. Нажимаем кнопку Далее> далее… На рисунке 19 показано окно выбора внешнего вида формы. Выбираем переключателем вариант В один столбец.

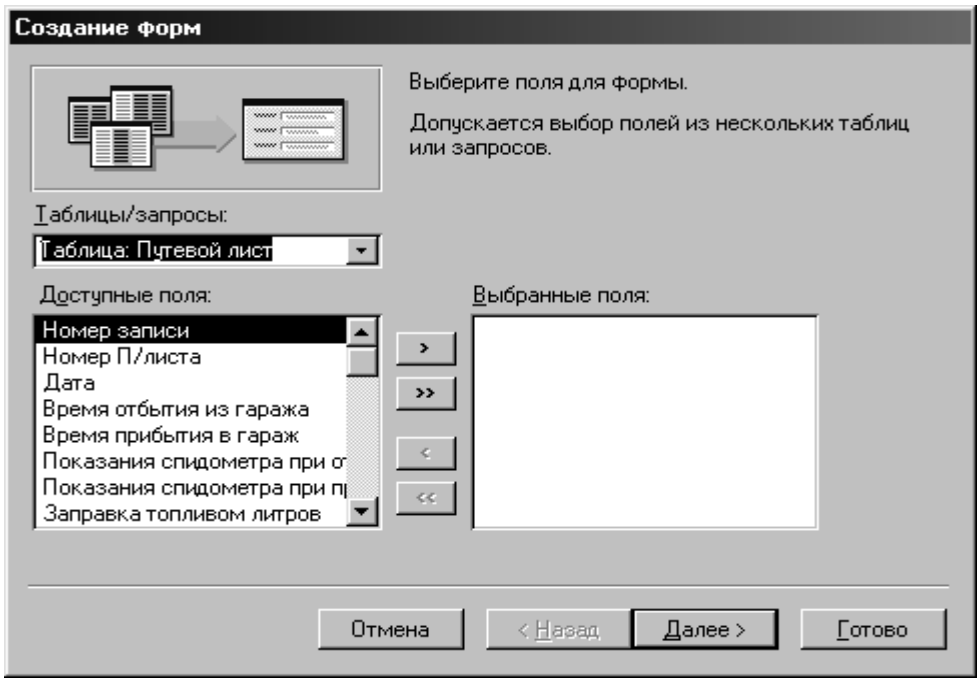


Рис.34

Нажимаем кнопку Далее>. Затем выбираем стиль оформления (Рис. 20) Обычный и нажимаем кнопку Далее>. В последнем окне выберем имя формы - Путевые листы. Нажимаем кнопку Готово при нажатом переключателе – Открытие формы для просмотра или ввода данных. Затем откроем форму в режиме Конструктора и откорректируем созданные Мастером данные. Где надо, растянем заголовок поля для того, чтобы полностью читались наименования полей. Выровняем поля и добавим кнопки:

- Начало
- Назад
- Вперед
- Конец
- Добавить
- Отменить
- Принять
- Удалить
- Закрыть форму.

Для полей «Марка топлива», «Время пребывания» и «Время отбытия из гаража» лучше всего сделать поле со списком (фиксированный набор значений).

Окончательный вид формы показан на рисунке 35.

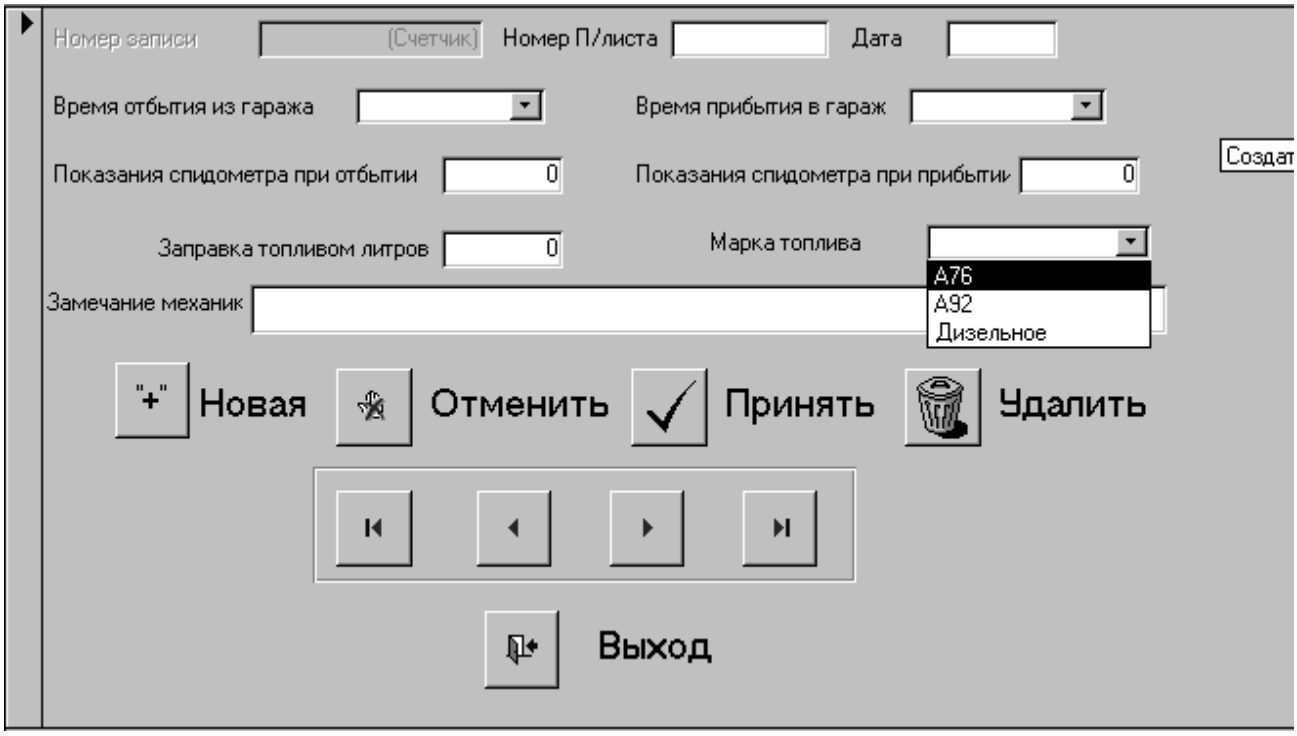

Номер записи
(Счетчик)
Номер П/листа
Дата
Время отбытия из гаража
Время прибытия в гараж
Показания спидометра при отбытии
0
Показания спидометра при прибытии
0
Заправка топливом литров
0
Марка топлива
A76
A92
Дизельное
Замечание механик
"+"
Новая
Отменить
Принять
Удалить
Выход

Рис.35

Практическая работа №5

Тема: Создание формы для управления автономной БД (Главная кнопочная форма).
Цель: Приобрести и закрепить практические навыки в построении алгоритма управления БД и создание для этого форм.

Часть I: Построение алгоритма управления автономной БД. Построение Первой кнопочной формы.

1. Построение структуры управление БД:

1.1 Анализ схемы управления БД:

Схема форм проекта «Автогараж»

Рис.36

На рисунке 36 показана схема форм вашего проекта. Первая кнопочная форма предназначена для управления вводом данных в БД. На ней будут расположены кнопки для открытия форм ввода данных и формы Справочные данные. Из формы справочные данные будут открываться те формы, через которые будут вводиться данные нерегулярно, по мере изменения или дополнения информации. Так как эта информация будет содержаться в нескольких строках, то есть необходимость создать эти формы в табличном виде для просмотра всей введенной информации.

Все управление вводом данных, как указывалось выше, будет помещено на формы Первая кнопочная, Отчеты и Справочные данные. Подробнее распишем какие функции будут переданы каждой форме.

Первая кнопочная:

1. Открытие формы Главная;
2. Открытие формы Путевые листы;
3. Открытие формы Справочные данные;
4. Открытие формы Отчеты;
5. Выход из приложения.

Отчеты:

1. Запуск отчета №1;
2. Запуск отчета №2;
3. Закрытие формы Отчет

Справочные данные:

1. Открытие формы Автотранспорт;
2. Открытие формы Водители;

3. Открытие формы Заказчики;
4. Закрытие формы Справочные данные.

1.2. Создание форм, управляющих БД: В окне БД (Рис.5) откроем вкладку Формы и нажмем кнопку Создать. Откроется окно Новая форма (Рис.17), в котором выберем из меню Конструктор. Откроется новая форма в режиме Конструктора. Не редактируя, сохраним ее под именем Отчеты. ***Так же создадим форму Справочные данные***.
После чего мы приступим к созданию формы Первая кнопочная.
Для этого создадим с помощью Конструктора простую форму, из которой будем размещать кнопки. Когда мы поместим на форму кнопку, то откроется Мастер кнопок. Выберем для первой Категорию Работа с формой и Действие Открытие формы (Рис.36)

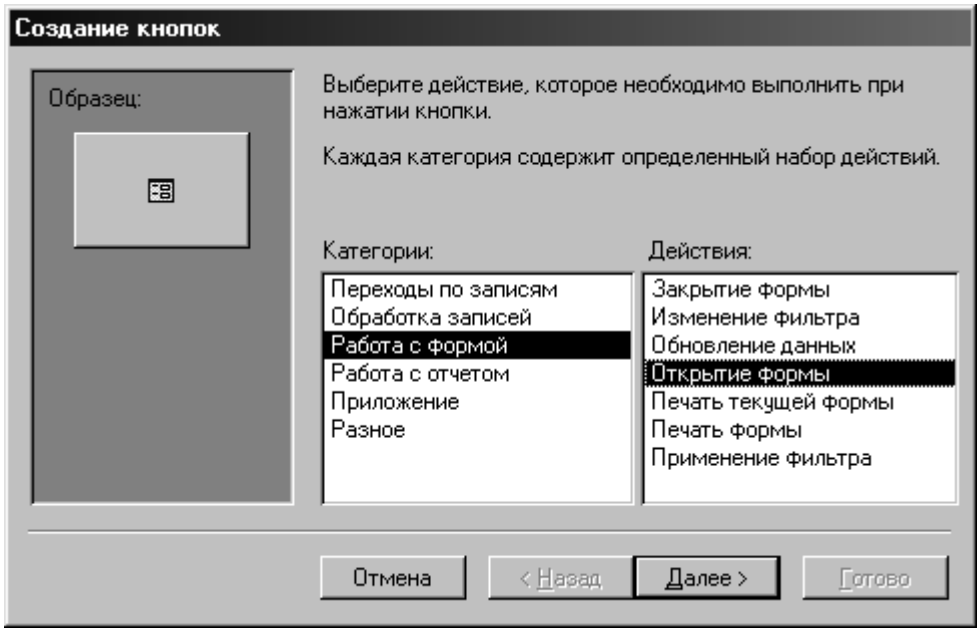


Рис.36

.Нажмем кнопку Далее>. В следующем окне (Рис.37) выберем форму Главная и нажмем кнопку Далее>.

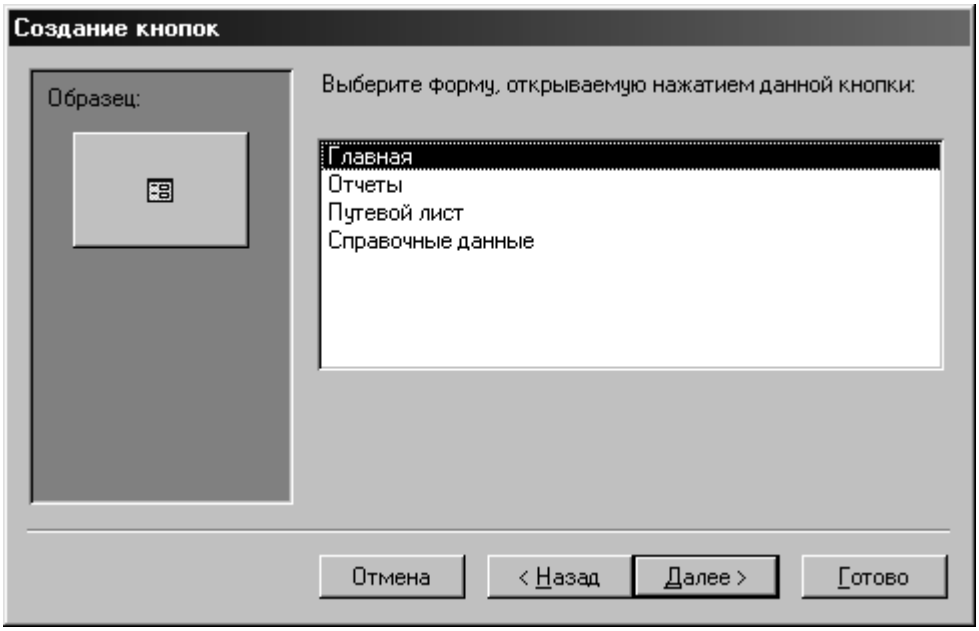


Рис.37

На рисунке 38 показано следующее окно Мастера кнопок, где мы отметим переключатель открыть форму и показать все данные.

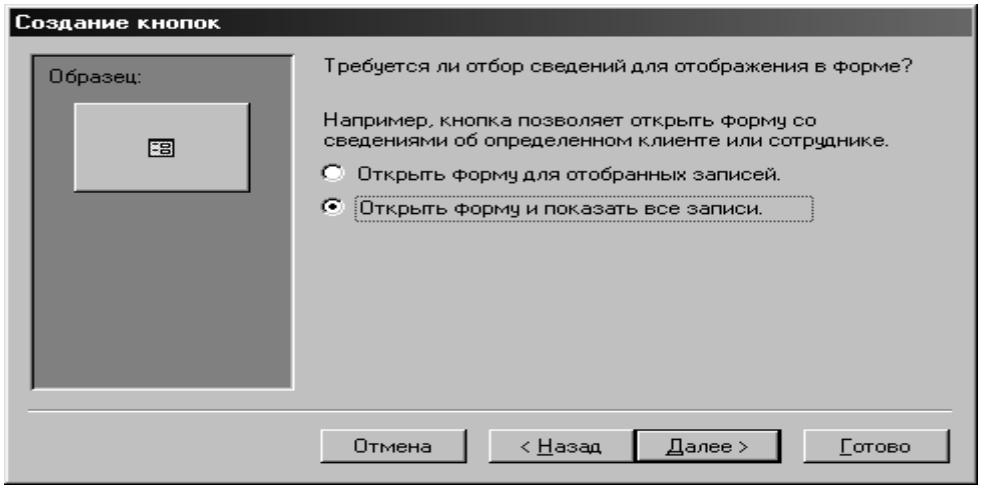


Рис.38

Нажмем кнопку Далее> и перейдем к следующему окну..
В этом окне (Рис.39) определим текст на кнопке:
Включим переключатель Текст и введем в поле справа: Ввод новых данных. Нажмем кнопку Далее>

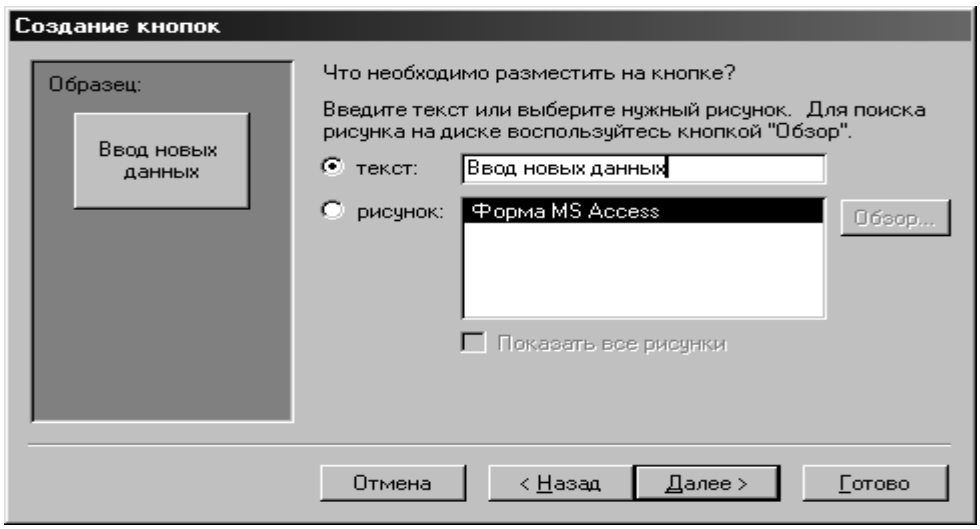


Рис.39

В следующем окне зададим имя кнопки Кнопка 1 и нажмем кнопку Готово. Мы получили первую кнопку, с помощью которой сможем открыть форму Главная для ввода данных в таблицу Главная. Таким образом мы создаем остальные кнопки.

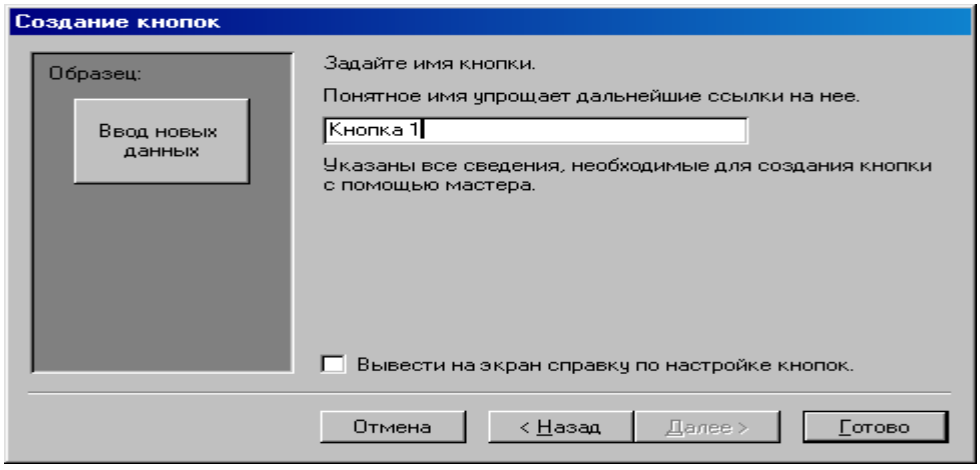


Рис.40

Рис.41

Поместим на форму Рисунок и текст, расположим их как показано на рисунке 41. Текст на кнопках и сами кнопки тоже отредактируем (сделаем их крупнее). Рисунок из ***c:\Program Files\Microsoft Office\office\Bitmaps\Dbwiz***

Часть II: Построение формы Отчеты и Формы Справочные данные.

2. Построение форм, как составных частей системы управления БД:

Для построения формы Справочные данные сначала надо построить формы, в которые будут вносится эти данные. Как уже отмечалось эти формы будут иметь табличный (ленточный) вид.

2.1 Создание формы Автотранспорт. В режиме окно БД откроем вкладку Формы и нажмем кнопку создать. В появившемся окне (Рис.17) выберем в верхнем поле Автоформа: Ленточная, а в нижнем поле Автотранспорт. Нажмем **Ok** и, введем имя формы Автотранспорт, получим нашу форму. Затем войдем в эту форму в режиме Конструктор и отредактируем ее. Добавим кнопки Закрыть, Новая, Отменить, Удалить и Сохранить. Внешний вид формы показан на рисунке 42.

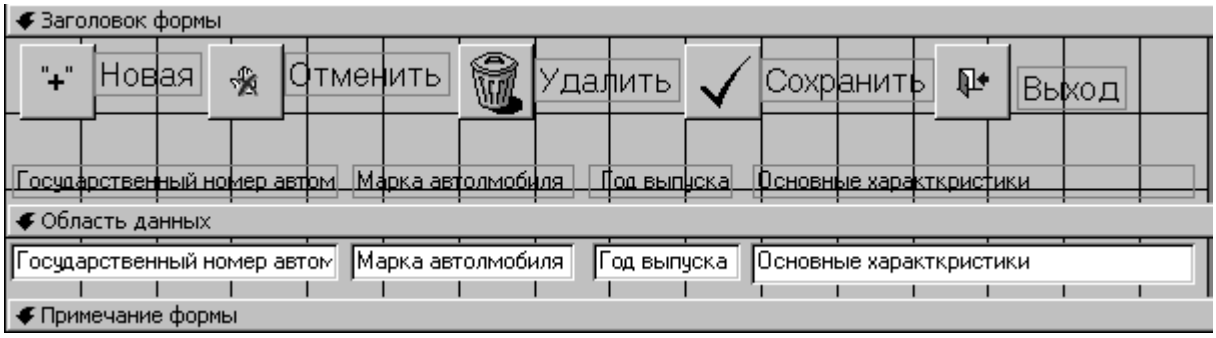


Рис.42

2.*2* ***Создать по данному принципу формы Водитель и Заказчик: Внешний вид формы показаны на рисунках 43 и44.***

Рис.43

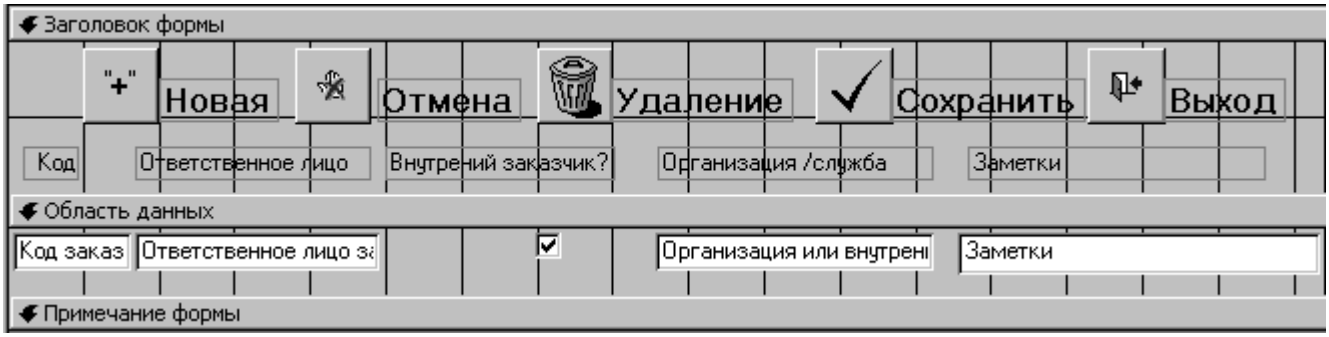


Рис.44

2.3 Приступим к созданию формы Справочные данные:
Откроем созданную ранее форму Справочные данные в режиме конструктора, чтобы затем на нее поместить управляющие кнопки и немного оформить ее внешне.
Разместим на форме кнопки для открытия форм Водители, Автотранспорт, Заказчики и Закрыть форму. Для кнопок, которыми мы будем открывать формы, мы выберем текст, а для закрытия формы значок. Заключим функциональные элементы (кнопки) в прямоугольную область, затем поместим на форму рисунок. Рисунок из ***c:\Program Files\Microsoft Office\office\Bitmaps\Dbwiz.*** Внешний вид формы показан на рисунке 45.

Рис.45

Практическая работа № 6

Тема: Создание отчетов на основе запросов.
Цель: Приобрести и закрепить практические навыки в построении отчетов на основе запроса.

Часть I: Построение отчета с помощью Мастера отчетов.

1. Создание отчета:

1.1 Создание отчета для вывода данных на печать: В окне БД откроем вкладку Отчеты и нажмем Создать. Откроется окно создания отчета. Выберем в этом окне Мастер отчетов. На рисунке 46 показано, что мы выбираем Запрос Главная Запрос (в нижнем поле окна). Нажмем кнопку **Ok.**

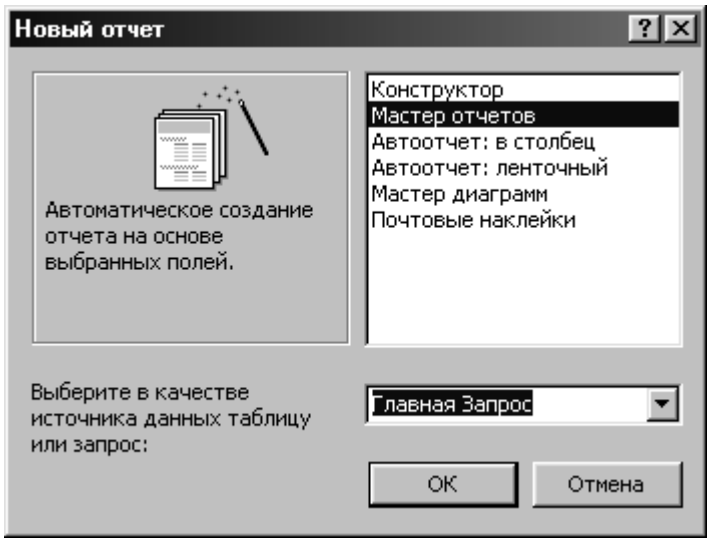


Рис.46

В следующем окне выберем поля из запроса (Рис.47) из левого нижнего поля в правое нижнее с именем Выбранные поля. Для этого нажмем кнопку >>. Перейдем в следующее окно кнопкой Далее>.

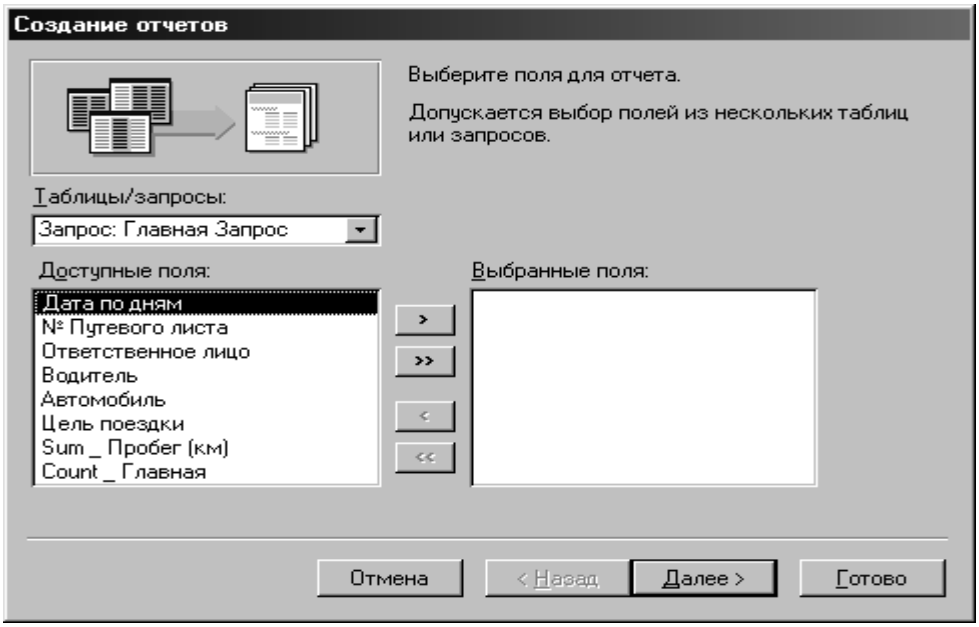


Рис.47

Далее определимся с группировкой данных в отчете. Это значит, что данные будут объединяться в группы по определенным признакам (уровням группировки) и отсортировываться по этим признакам в порядке сортировки. Например для нашего случая будут отобраны все записи за конкретный месяц, затем сгруппируются под общий заголовок все записи, которые принадлежат одному заказчику, внутри которых они распределяются по группам записей принадлежащих конкретным водителям (Рис.48). На рисунке 48 в левом поле окна даны поля отчета, по которым мы можем объединять записи в группы.

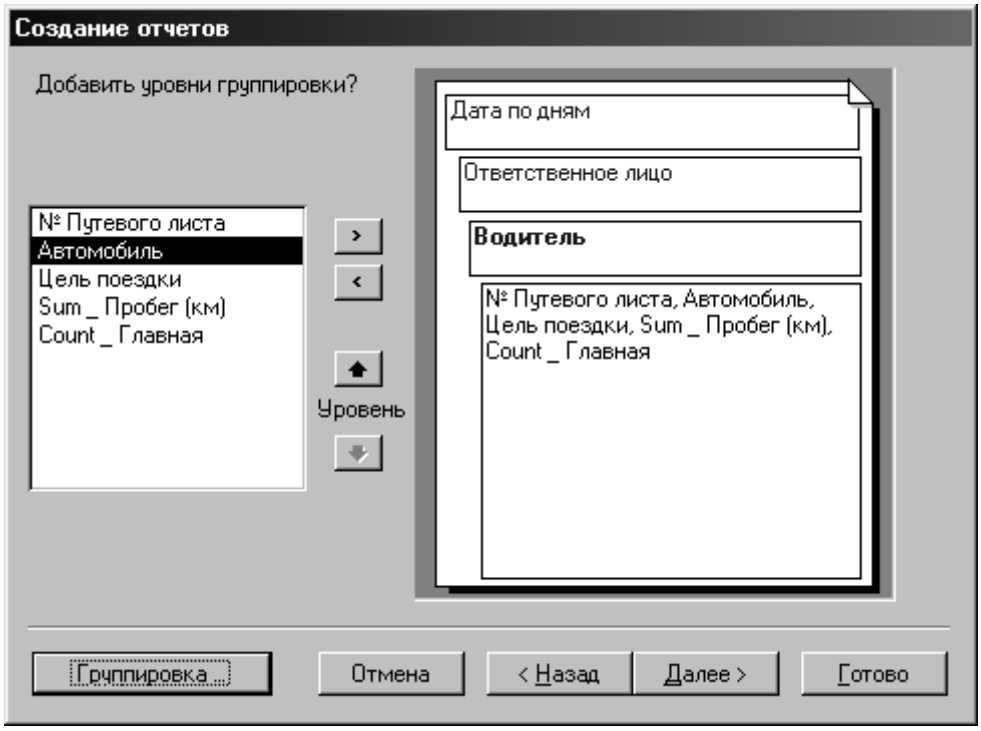


Рис.48

Объединим записи по 3-м уровням:
Даты по месяцам;
Ответственное лицо (заказчик);
Водитель.
Для этого из левого поля мы помечаем поля курсором и перетаскиваем стрелкой эти поля в правое поле окна в порядке убывания уровней. Сформируем отчет вида, указанного в правом окне рисунка 48. Нажмем кнопку Группировка. Откроется форма, показанная на рисунке 49. В ней мы ничего изменять не будем, лишь просмотрим интервалы группировок. Вернемся обратно, нажав кнопку **Ок**. Нажмем кнопку Далее>.

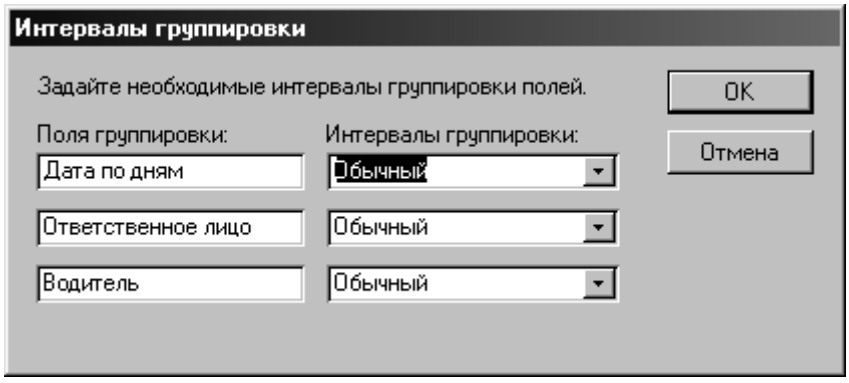


Рис.49

В следующем окне можно определить порядок сортировки полей отчета. Мы оставим это по умолчанию. Но нажмем кнопку Итоги… В нашем случае возможны итоги по одному полю: Километраж. Отметим флажок Sum, который обозначает Сумма (Рис.51). Переключатель в правой части окна установим в Данные и итоги. Нажмем кнопку **Ок**. И затем кнопку Далее

показанная на рисунке 50. В ней мы ничего изменять не будем, лишь Нажмем кнопку Далее>.

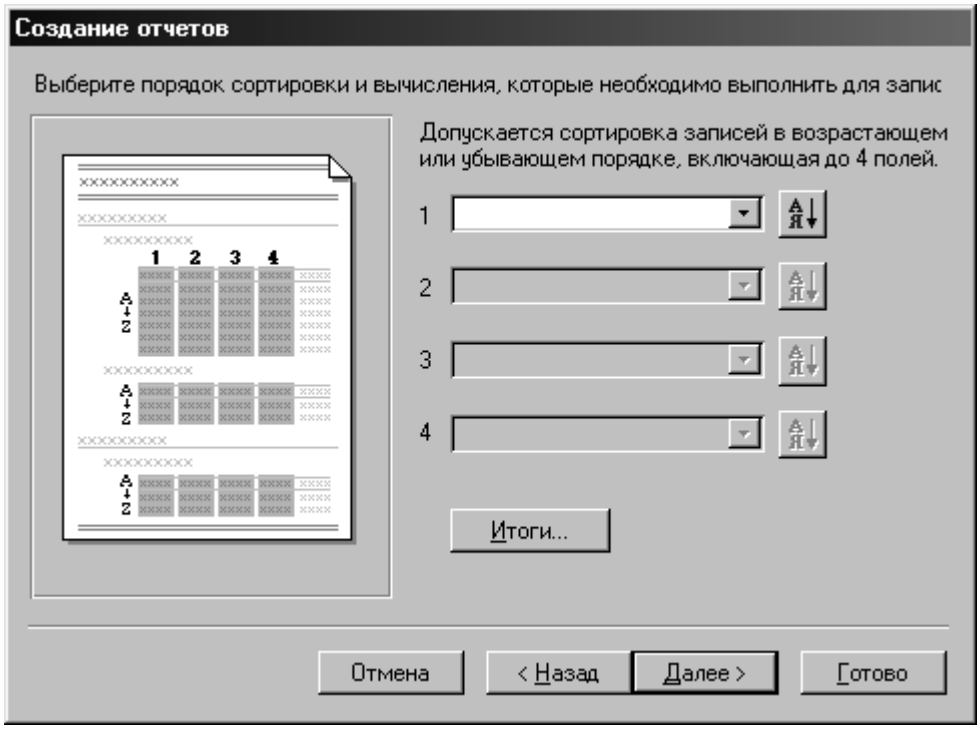


Рис.50

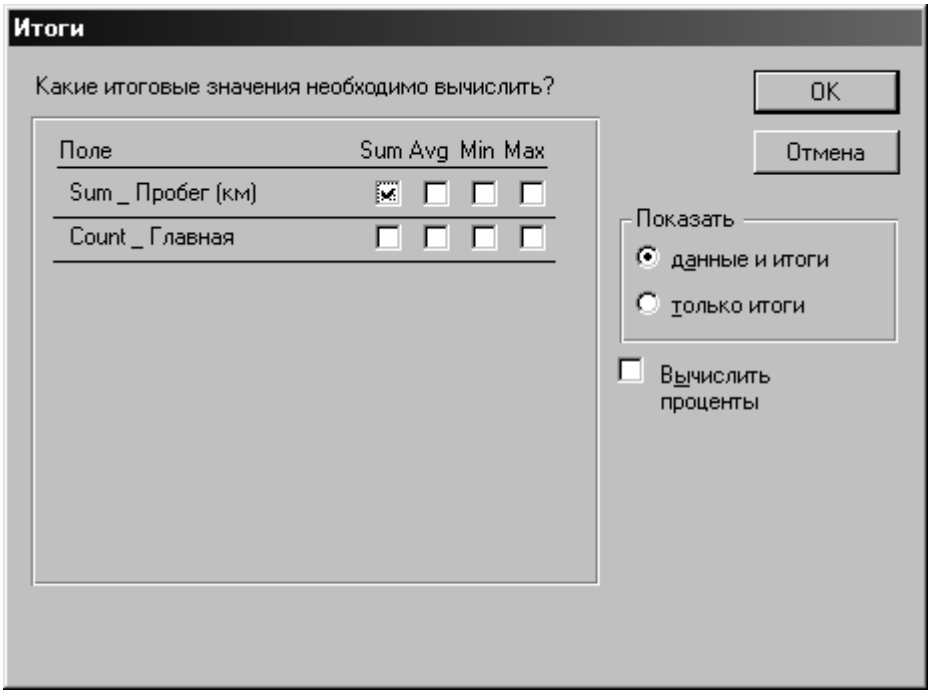


Рис.51

В следующем окне определим внешний вид отчета. Далее удобства его прочтения и ориентации его печатной формы на странице установим переключатель Макет в положение Структура 1 и переключатель Ориентация в Книжная. Отметим флажок Настроить ширину полей для размещения на одной странице. Нажмем кнопку Далее>(Рис.52).

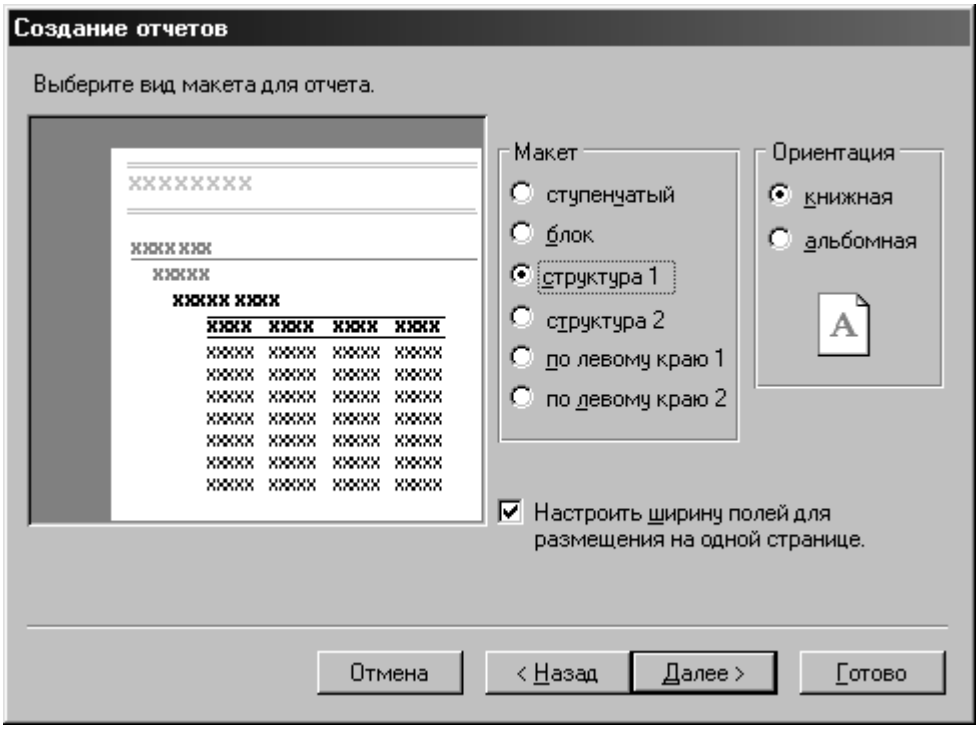


Рис.52

Далее мы определим стиль оформления отчета. Выбираем в правом поле окна вариант Строгий. Нажмем кнопку Далее>.

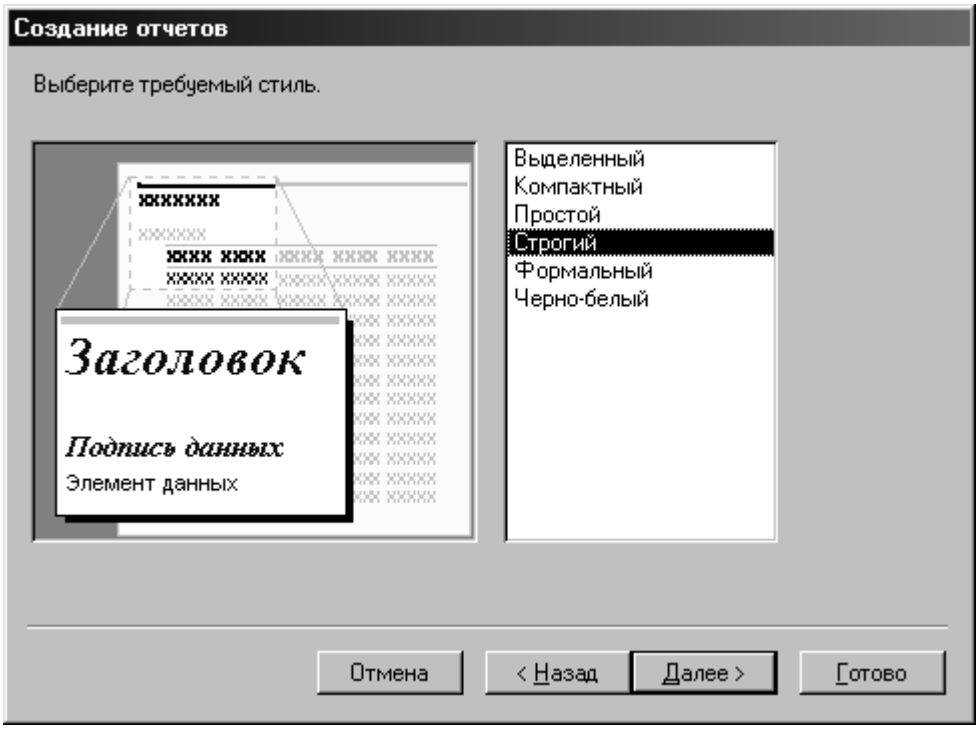


Рис.53

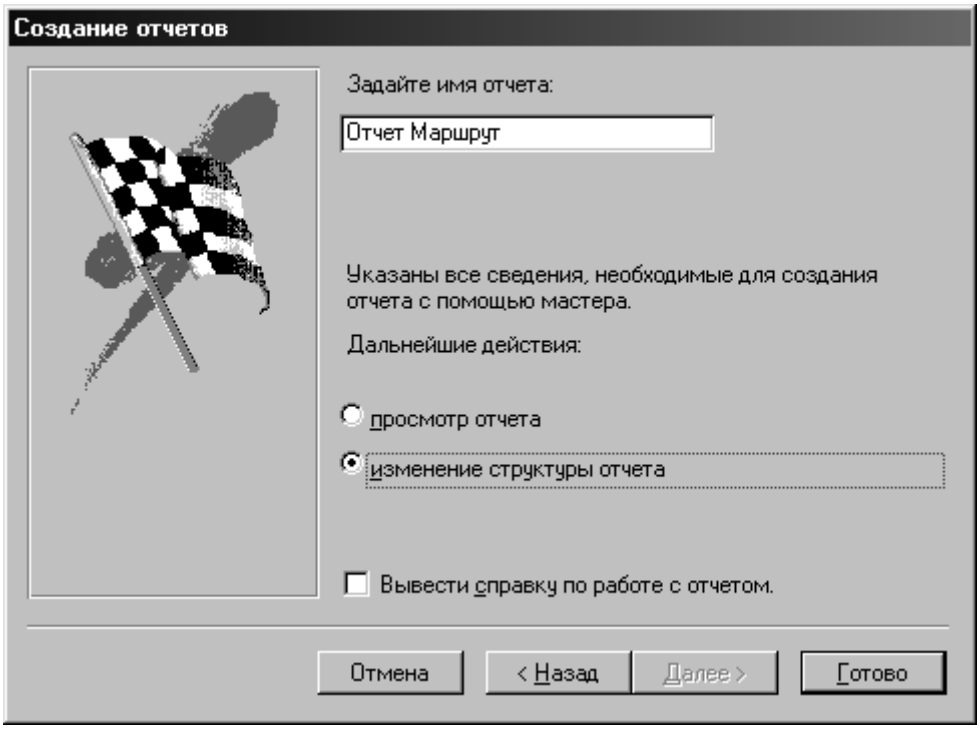


Рис.54

В последнем окне мы дадим заголовок отчету: Отчет Маршрут. Установим переключатель в положение Изменение структуры отчета для редактирования его в Конструкторе. Нажмем кнопку Готово.

В конструкторе откорректируйте отчет по своему усмотрению. Можно передвинуть поля влево, так как заголовки групп будут располагаться выше общей информации и не будут мешаться. Можно убрать рамки, некоторые линии и «поджать» разделы отчета, они занимают на отчете лишнее пространство. Присмотрите названия заголовков полей, их можно подсократить или заменить на более лаконичные (например «Номер записи» на «№»).

Часть II: Построение формы отчетов и создание автозапуска Первой кнопочной формы.

2. Окончательная стадия создания системы управления БД:

2.1 Ввод элементов управления в форму Отчет: В открытом окне БД (Рис.55) выбираем вкладку Формы и форму Отчеты, которую мы уже создали ранее. Эта форма пустая и нам необходимо создать на ней командные кнопки для просмотра и печати отчетов.

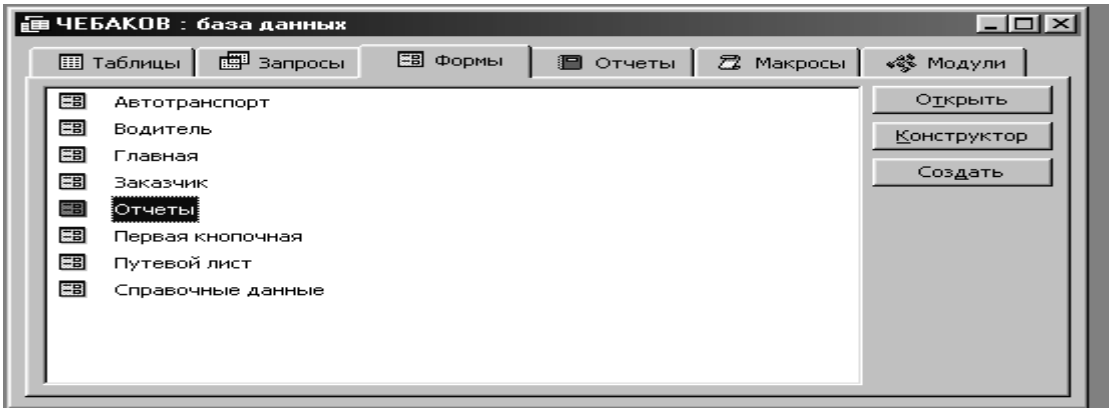


Рис.55

Откроем форму в режиме Конструктора. На панели инструментов нажмем кнопку для запуска мастеров и выберем на этой панели компонент Кнопка . Поместим его на форму. Запустится Мастер, который проведет нас по всем этапам разработки кнопки:
Категории: Работа с отчетом;
Действия: Просмотр отчета;
Выбрать отчет для просмотра: Отчет Маршрут;
Разместить на кнопке: Рисунок
Имя кнопки: «Отчет 1».

Рис.56

Расположим на форме надпись для кнопки: Просмотр отчета по Маршруту автотранспорта. Поместим вокруг прямоугольник и внизу кнопку Закрыть форму. Результат работы показан на рис. 56. Мы можем добавлять кнопки по мере добавления отчетов в БД.

2.2. <u>Создание автозапуска БД</u>: Теперь нам осталось сделать автозапуск Первой кнопочной формы и БД готова для нормальной работы.
Для этого мы создадим один Макрос. Откроем в окне БД вкладку Макросы и нажмем кнопку Создать.

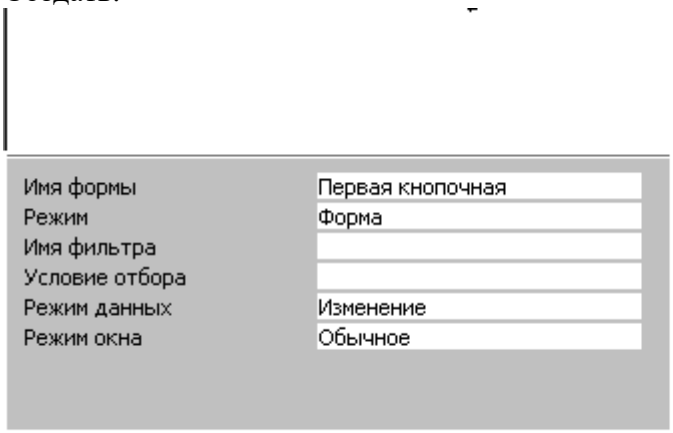


Рис.58

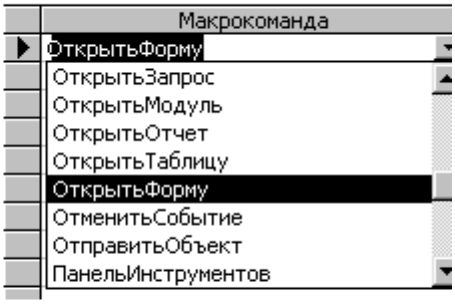


Рис.57

В открывшемся окне выберем Макрокоманду Открыть форму (Рис 57)
На Рис. 58 показана нижняя часть окна (Аргументы макрокоманд), где мы указываем название формы которую будет открывать этот макрос. Режим данных выберем Изменение. Все остальное оставим по умолчанию. Далее мы закроем Макрос и сохраним его под именем “Autoexec”. Это означает, что данный Макрос будет запускаться при запуске нашей БД.