Геннадий Гурвиц

Разработка реального приложения с использованием

Visual FoxPro 9

Профессиональный подход

Министерство транспорта Российской Федерации
Федеральное агентство железнодорожного транспорта
ГОУ ВПО «Дальневосточный государственный
университет путей сообщения»

Г.А. Гурвиц

Разработка реального приложения с использованием Microsoft Visual FoxPro 9

Хабаровск
Издательство ДВГУПС
2007

УДК 004.655(075.8)
ББК 3973.2-018.2 я73
Г 950

Рецензенты:

Кафедра «Автоматика и системотехника»
Тихоокеанского Государственного технического университета
(заведующий кафедрой доктор технических наук,
профессор *Чье Ен Ун*)

Заместитель директора по науке ФГУП ВНИИФТИ Дальстандарт
доктор технических наук,
профессор Ю.Б. Дробот

Гурвиц Г. А.
Г 950 Разработка реального приложения с использованием Microsoft Visual FoxPro 9 : учеб. пособие. – Хабаровск: Изд-во ДВГУПС, 2007. – 198 с.: ил.
ISBN 5-262-00297-8

Учебное пособие соответствует государственному образовательному стандарту направления 230200 «Информационные системы» специальности 230201 «Информационные системы и технологии».

Рассматриваются основные этапы создания реального приложения для работы с реляционными базами данных. Описывается работа с Microsoft Visual FoxPro 9.0.

Пособие предназначено для студентов, изучающих курсы «Управление данными», «Базы данных», «Безопасность баз данных» и «Корпоративные информационные системы».

УДК 004.655(075.8)
ББК 3973.2-018.2 я73

Все названия программных продуктов являются зарегистрированными торговыми марками.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими-то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на то нет письменного разрешения владельца авторских прав.

Материал, изложенный в пособии, многократно проверен автором. Но поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 5-262-00297-8

© Гурвиц Г.А.
© ГОУ ВПО «Дальневосточный государственный университет путей сообщения» (ДВГУПС), 2007

ВВЕДЕНИЕ

Реально работающее приложение лучше всего подходит для изучения проблем, с которыми не раз доводилось сталкиваться в процессе работы с новым программным продуктом. Именно на этом и сделан особый акцент в пособии. Надеюсь, что вы найдете в нем простые приемы создания приложений для работы с базами данных, позволяющие избежать тупиковых решений и ненужных усилий, обычно ведущих к напрасной потери времени. Перед вами руководство по быстрому освоению базовых возможностей Microsoft Visual FoxPro. Поэтому не ищите в нем подробных экскурсов в теорию программирования. В последнее время широкое распространение получил унифицированный язык моделирования – UML, который предназначен для описания, визуализации и документирования объектно-ориентированных систем и бизнес-процессов с ориентацией на их последующую реализацию в виде программного обеспечения. Ввиду небольшого масштаба предприятий, предложенных в качестве вариантов к курсовому проекту в данном пособии, этот язык и его реализации (CASE-инструментарии) не рассматриваются.

После установки Microsoft Visual FoxPro советую Вам заглянуть в папку: C:\Program Files\Microsoft Visual FoxPro 9\Samples\

Здесь вы найдете несколько приложений-примеров, входящих в базовую поставку Microsoft Visual FoxPro. Начинающим разработчикам будет очень полезно изучить возможности базы данных Tastrade. Поищите здесь и другие приложения, выберите те, на которые хотите сделать похожими ваши собственные, и разберитесь как они устроены. Вы удивитесь: масса вещей, требующих написания текста программы, теперь просто встроены в меню СУБД.

Уверяю Вас, что рассматриваемый в пособии Microsoft Visual FoxPro, является самой выгодной файловой СУБД среди существующих, исходя из соотношения цены и качества. Информационные системы уровня отдела предприятия, построенные с использованием Microsoft Visual FoxPro, выгодно отличаются невысокой суммарной стоимостью владения, а богатые возможности этой СУБД являются одним из самых важных критериев при выборе продукта, который будет использоваться на предприятии при построении баз данных.

В середине 90-х Microsoft Visual FoxPro 3.0 стал первым инструментальным средством корпорации Microsoft с переведенной на русский язык документацией. Несмотря на свой почтенный стаж (17 лет на рынке средств разработки и СУБД), Visual FoxPro продолжает оставаться одним из популярных инструментов. А ведь еще в конце 90-х гг. прошлого столетия многим казалось, что Visual FoxPro вскоре прекратит свое развитие, разделив учесть коллег-соперников по сегменту баз данных xBase.

В 2005 г. Microsoft выпустила очередную версию этого инструмента – Visual FoxPro 9.0, которая, по мнению экспертов, стала самым существенным обновлением данной системы со времени перехода от варианта DOS к Windows (от версии VFP 2.6 к 3.0). Это вызвало заметный энтузиазм в среде сообщества пользователей и разработчиков Visual FoxPro, которое сохраняет преданность данной платформе разработки и верит в ее возможности решения самого широкого круга задач.

По мнению Кэна Леви, менеджера группы продуктов Visual Studio Data корпорации Microsoft, Visual FoxPro 9.0 с его механизмом управления локальным курсором отлично подходит для реализации проектов с БД любых размеров. С помощью объектно-ориентированного языка, нацеленного на обработку данных, разработчики могут создавать приложения для настольных ПК, клиент-серверной среды и Web.

Одно из самых главных достоинств инструмента – его полная совместимость с предыдущими версиями Visual FoxPro и даже приложениями, написанными 20 лет назад в среде dBase II. Обладая собственным внутренним механизмом управления реляционной БД, тесной взаимосвязью между языком и данными, полноценными возможностями объектно-ориентированного программирования и широким спектром функций Misrosoft Visual FoxPro 9.0 позволяет создавать производительные, масштабируемые БД-ориентированные решения (настольные, клиент-серверные и Web) с поддержкой баз данных с таблицами объемом до 2 Гб. При этом Visual FoxPro 9.0 выгодно отличается от других инструментов Microsoft умеренными системными требованиями и высокой эффективностью разрабатываемых приложений (производительность, размеры БД и программного кода).

Visual FoxPro пока избежал участи перевода в среду .NET, он сам и создаваемые с его помощью приложения предназначены для работы в традиционной Windows с COM-архитектурой. Он не использует принцип управляемого кода, при этом язык FoxPro сохраняет высокую эффективность – на нем написаны многие компоненты самого инструмента. В то же время улучшение интеграции с .NET-приложениями – одно из главных направлений развития VFP. С помощью VFP 9.0 можно создавать Web-сервисы и COM-компоненты, при этом существенно упростилось их взаимодействие с .NET-приложениями.

Visual FoxPro последние годы применяется и для мобильных решений. Теперь на его базе можно разрабатывать и приложения для компьютеров под управлением Windows XP Tablet PC.

Значительное число новшеств связано с механизмом управления данными. Все годы существования Misrosoft Visual FoxPro особое внимание уделяется поддержке SQL-запросов и взаимодействию с MS SQL Server. С этой целью в версии 9.0 реализованы новые типы данных, сняты многие

ограничения SQL-языка, введены дополнительные типы индексов, усилена работа с удаленными данными и т. д.

Модифицированный XML Adapter обеспечивает улучшенную поддержку иерархических XML- и XSD-схем.

Visual FoxPro в силу своей предметной нацеленности всегда отличался достаточно мощной системой формирования отчетов. Эти возможности расширены за счет новых архитектурных решений, повышающих возможности управления выводом и форматирования данных.

Целый ряд улучшений и модификаций Visual FoxPro 9.0 связан с изменениями в интегрированной среде разработки, библиотеках FoxPro Foundation Classes, синтаксисе языка программирования, элементах управления и т. д. В то же время, говоря о развитии Visual FoxPro, нужно отметить, что Microsoft не считает нужным принимать радикальные шаги по повышению масштабируемости создаваемых приложений, сохраняя дистанцию между Visual FoxPro и своими стратегическими инструментами (SQL Server + Visual studio .NET).

Компания Microsoft наметила несколько основных направлений применения разработчиками Microsoft Visual FoxPro 9.0:

http://msdn.microsoft.com/vfoxpro/default.aspx

- управление данными. Можно создавать .NET-совместимые решения с иерархическими XML-структурами и Web-сервисами, а также для обеспечения обмена данными с MS SQL Server через язык SQL и новые типы данных, появившиеся в версии 9.0;
- гибкие и производительные средства разработки. Удобная среда разработки, простой и эффективный язык, разнообразные элементы визуального интерфейса, улучшенная поддержка графики;
- возможность создания всех видов БД-решений. Разработчики могут создавать и развертывать автономные и распределенные приложения для Windows Tablet PC, COM-компоненты и Web-сервисы;
- системы для построения отчетов. В VFP 9.0 реализована новая архитектура для вывода данных, что повышает гибкость формирования и просмотра отчетов, в том числе с использованием XML, HTML и графических форматов.

Microsoft Visual FoxPro сейчас не пользуется большой популярностью в США и развитых странах запада. Отечественное же сообщество разработчиков Visual FoxPro сейчас насчитывает более 30 тыс. человек. Огромное число информационных систем в настоящее время работает на основе этой СУБД. Один из менеджеров корпорации Microsoft прокомментировал сложившуюся ситуацию следующим образом: «Популярность FoxPro в России и вообще в Восточной Европе поражает наших американских коллег».

На мой взгляд, это связано с тем, что формирование российского сообщества разработчиков Microsoft началось как раз с FoxPro. Именно ему была полностью посвящена первая российская конференция Microsoft

DevCon'94. Продукт занимал заметное место и в дальнейшем, в течение всего периода проведения этого крупного ежегодного мероприятия. Рынок информационных систем в России – консервативен, а их жизненный цикл значительно превышает мировые стандарты. До сих пор некоторые крупные предприятия и организации нашего города эксплуатируют свои информационные системы, основанные на FoxPro 2.6 for DOS.

Почему Microsoft, отличающаяся достаточно жесткой линией в отношении перевода своих пользователей на новые архитектурные решения, посчитала нужным сохранить Visual FoxPro – это остается загадкой. Конечно, можно говорить о большой инсталлированной базе приложений, но их техническая поддержка может выполняться без выпуска новых версий инструментария. Ссылка на интересы сообщества FoxPro-разработчиков выглядит тоже не очень убедительно: в лучшие годы их число не превышало 200 тыс. человек. Для примера: популярность Visual Basic 6.0 оценивалась в 2001 г. на порядок выше, в 2 млн человек, но это не помешало Microsoft (несмотря на активное протесты VB-программистов) «железной рукой» поставить их перед необходимостью перехода в качественно новую, несовместимую с предыдущими версиями, среду VB.NET.

Microsoft заявила о поддержке Visual FoxPro 9.0 до декабря 2014 г.
http://msdn.microsoft.com/vfoxpro/productinfo/faq/default.aspx

После выхода в свет Visual FoxPro 9.0 группа его разработчиков приступила к созданию нового проекта с кодовым названием Sedna:
http://msdn.com/vfoxpro/roadmap

Продукт будет выпущен в первой половине 2007 г. Для его работы будет необходим Visual FoxPro 9.0. Особенностью Sedna будет функциональная совместимость с прикладными компонентами, созданными с использованием Visual Studio 2005, .NET Framework 2.0 и SQL Server 2005. Sedna создается для работы под управлением новой операционной системы Windows Vista.

По заявлениям менеджеров компании, Microsoft не планирует включить Visual FoxPro в состав Visual Studio Net. У корпорации также нет планов создать Visual FoxPro Net. Продукт останется автономным и будет работать под управлением 64-разрядной операционной системы. Sedna обеспечит усовершенствование тех компонентов Visual FoxPro, которые отвечают за COM и .NET совместимость.

Уверен, что выпускнику универсистета по специальности «Информационные системы и технологии» непременно придется столкнуться в своей работе с этим интересным продуктом. Поэтому вашему вниманию – очередной курсовой проект «Разработка реального приложения с использованием Microsoft Visual FoxPro 9».

1. ПОСТАНОВКА ЗАДАЧИ

Итак, к делу. Вам предстоит работа в информационно-аналитическом отделе дистанции гражданских сооружений. Первое ваше задание – разработка прикладного программного обеспечения деятельности отдела по учету недвижимости, находящейся на балансе предприятия. В связи с реорганизацией станционного хозяйства, объектов в ведении отдела теперь около полусотни. Квартир порядка трех тысяч, в них проживает около десяти тысяч человек. Учет недвижимости, а также отслеживание квартплаты отныне в ведении этого отдела, но это уже второй этап работы.

Занимаясь разработкой прикладного программного обеспечения деятельности различных предприятий на протяжении многих лет, могу отметить, что в 90 случаях из 100 заказчик сам не знает, чего хочет, и в 99 из 100 случаях постановку задачи приходится воспринимать на слух, в процессе работы неоднократно уточняя те или иные моменты создаваемой программы. Более того, при очередной встрече с заказчиком, связанной с демонстрацией уже выполненных этапов, очень часто открываешь для себя все новые и новые горизонты предстоящей работы, требующие существенного изменения как структуры данных, так и интерфейса будущего приложения. Но это не самый худший вариант. Иногда уже через день после итоговой встречи заказчик переосмысливает свои цели, после чего задача меняется коренным образом, и следующий визит заставляет начать всю работу заново. Именно по этой причине я рекомендую вам, внимательно выслушав заказчика, попросить его описать задачу в письменном виде, на основании чего самостоятельно сформулировать постановку задачи и еще раз обсудить ее с клиентом. Уверяю вас, если результат окажется положительным, то это будет признанием того, что ваш работодатель действительно нуждается в заказанном программном обеспечении, а самое главное знает, чего хочет.

В первую очередь на вас возложена задача компьютерного учета недвижимого имущества. Объем работы сравнительно не большой. Не радуйтесь! Ваш начальник требует, чтобы эксплуатация программного комплекса, заказанного им сегодня, началась еще вчера. Вы провели в отделе по учету недвижимости значительное время, но все, что вам удалось выяснить из разговора с персоналом, – это набор данных, которые будут храниться в электронном виде, их тип и максимальное количество в базе (табл. 1.1).

Таблица 1.1

Набор данных «Недвижимость»

№	Поле	Тип	Размер	Описание
1	Address	Текстовый	50	Адрес здания
2	District	Текстовый	15	Район города, где оно расположено
3	Land	Числовой	10	Площадь земельного участка
4	Year	Числовой	4	Год постройки здания
5	Material	Текстовый	15	Материал стен здания
6	Comment	Поле Memo	Авто	Примечания
7	Wear	Числовой	2	Износ в процентах
8	Cost	Денежный	15	Стоимость здания в рублях
9	Line	Числовой	5	Расстояние от центра города
10	Square	Числовой	10	Площадь нежилых помещений
11	Picture	Поле OLE	Авто	Фото здания
12	Kind	Числовой	1	Вид собственности
13	Elevator	Логический	1	Наличие лифта
14	Flat	Числовой	4	Номер квартиры
15	Storey	Числовой	2	Номер этажа
16	Rooms	Числовой	1	Количество комнат
17	SquareFlat	Числовой	Авто	Общая площадь квартиры
18	Dwell	Числовой	Авто	Жилая площадь квартиры
19	Branch	Числовой	Авто	Всп. площадь квартиры
20	Balcony	Числовой	Авто	Площадь балкона
21	Height	Числовой	Авто	Высота квартиры
22	Account	Числовой	5	Номер лицевого счета
23	FioHost	Текстовый	60	Ф.И.О. квартиросъемщика
24	Pasport	Поле Memo	Авто	Данные его паспорта
25	Fio	Текстовый	60	Ф.И.О. проживающего в квартире
26	Born	Числовой	4	Год рождения проживающего
27	Status	Текстовый	20	Льготы и статус проживающего

Они сведены вами в таблицу. Надеюсь, что вы предупредили работающих о том, что если какой-либо параметр отсутствует в базе данных, то извлечь его и выполнить какие-либо расчеты с его участием будет в дальнейшем невозможно.

2. НОРМАЛИЗАЦИЯ ДАННЫХ

Теперь займемся проектированием эффективной структуры данных. На сегодняшний день известны три модели данных: иерархическая, сетевая и реляционная. Так как Microsoft Visual FoxPro – это реляционная СУБД, то выбора у нас нет. Теория реляционных баз данных была разработана в начале 70-х гг. Коддом (E. F. Codd) на основе математической теории отношений. В реляционной базе данных все данные хранятся в виде таблиц, при этом все операции над базой данных сводятся к манипуляциям с таблицами. Основными понятиями в этой теории являются: таблица, строка, столбец, индекс, первичный и внешний ключи, связи. Таблица состоит из строк и столбцов и имеет уникальное имя в базе данных. База данных содержит множество таблиц, связь между которыми устанавливается с помощью совпадающих полей. В каждой из таблиц содержится информация о каких-либо объектах одного типа.

Приступая к созданию нового приложения, главное – самым тщательным образом спроектировать структуру его таблиц. Если не уделить структуре должного внимания, то в лучшем случае это может проявиться в неэффективной работе приложения, а в худшем – в невозможности реализации некоторых требований к системе в целом. И, наоборот, при хорошей организации набора таблиц будут решены не только текущие проблемы, но и потенциальные, которые в данный момент вы не могли предвидеть. В итоге структура данных является определяющим фактором успеха или провала всего приложения.

Э.Ф. Кодд доказал, что при создании таблиц и связей между ними, следуя только немногим формализованным правилам, можно обеспечить простоту манипулирования данными. Его методика получила наименование *нормализации данных*. Теория реляционных баз данных основана на концепции использования ключевых полей для определения отношений между таблицами. Чем больше таблиц, тем больше отношений требуется определить, чтобы связать их между собой. Из теории Кодда отнюдь не следует, что каждая таблица должна быть напрямую связана с любой другой таблицей. Но, поскольку каждая таблица связана хотя бы с одной таблицей в базе данных, можно утверждать, что все таблицы в базе имеют прямые или косвенные отношения друг с другом.

Мы установили, какие поля будут включены в базу данных. Следующий этап состоит в разделении их на таблицы. Конечно же, можно было бы работать с приведенной выше единственной таблицей «Недвижимость», но даже не знающим правил нормализации ясно, что для каждого проживающего в квартире не имеет смысла повторять всю информацию о здании, квартире, ответственном квартиросъемщике и лицевом счете, а при переименовании улицы – вносить исправления в тысячи записей, содержащих сведения о технических характеристиках квартиры.

Наличие повторяющейся информации приведет к неоправданному увеличению размера базы данных. В результате снизится скорость выполнения запросов. При многократном вводе повторяющихся данных возрастет вероятность ошибки.

Представьте себе ситуацию, связанную с вводом данных о проживающих на Восточном шоссе. Это пять тысяч человек. Вот несколько вариантов адреса: Шоссе Восточное, Восточное шоссе, ш. Восточное, ш-се Восточное. А сколько еще вариантов может появиться у оператора, работающего с вашей программой! О грамматических ошибках и вариантах с номером дома, запятыми и точками в адресе позволю себе умолчать. Какую информационно-поисковую систему мы получим в результате? Скорее всего: искать можно – найти нельзя! Вашему вниманию – несколько советов по включению полей в таблицы.

- Включайте поля, относящиеся только к предметной области таблицы. Поле, представляющее факт из другой предметной области, должно принадлежать другой таблице. Позже при установлении отношений между таблицами, вы увидите, как можно объединить данные нескольких полей из разных таблиц. А на этом этапе убедитесь, что каждое поле таблицы описывает только одну предметную область. Если вы обнаружите, что в разных таблицах встречается однотипная информация, значит, какие-то таблицы содержат лишние поля.
- Не включайте производные или вычисляемые данные. В большинстве случаев вам нет необходимости хранить результаты вычислений в таблице. С помощью Visual FoxPro вы всегда сможете выполнить данные вычисления в нужный момент. Не имеет смысла хранить итоговые поля в таблице.
- Включите всю необходимую информацию Довольно легко упустить важные данные. Вернитесь к собранным на первой стадии проектирования материалам. Внимательно рассмотрите бланки и отчеты и убедитесь в том, что вся интересующая вас информация может быть извлечена или вычислена из таблиц. Подумайте о вопросах, которые вы будете формулировать к базе данных. Сможете ли вы получить на них ответ, используя данные из ваших таблиц? Включили ли вы поля, в которых будут храниться уникальные данные типа кода клиента? Какие таблицы содержат информацию, обязательную для создания отчета или формы?
- Разделите информацию на наименьшие логические единицы. Вам может показаться удобным иметь одно поле для хранения полного имени клиента или одно поле для названия и описания продукта. Если вы объединяете более одной категории информации в одном поле, вам будет потом весьма непросто выделить из него отдельные факты. Постарайтесь разбить информацию на наименьшие логические части.

Воспользуемся практическими рекомендациями теории нормализации для разработки на основании таблицы «Недвижимость» многотабличной базы данных «Real Estate».

На рис. 2.1 вы видите то, что у вас должно получиться после всех манипуляций, кратко изложенных выше и предусмотренных теорией нормализации. Практический же путь к этому результату смотрите на следующих полутора десятках страниц.

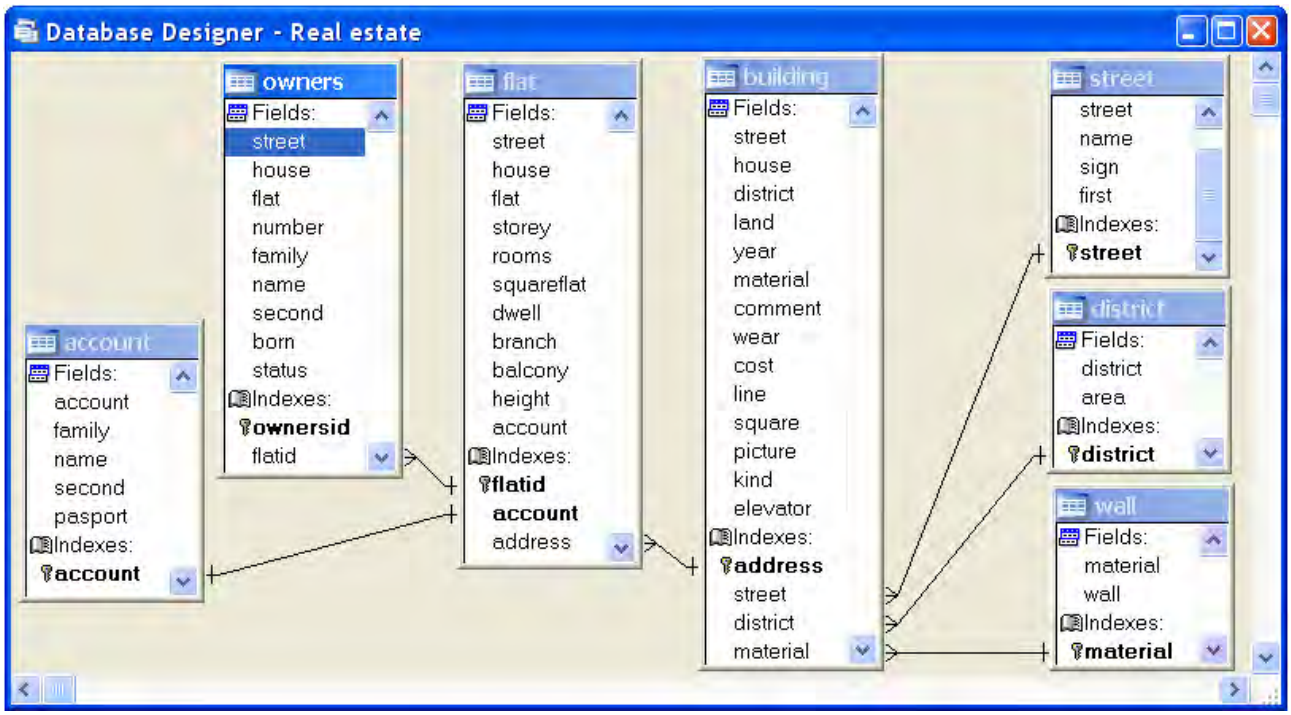


Рис. 2.1. Окончательный вид базы данных «Real Estate»

Первая нормальная форма. *Таблица находится в первой нормальной форме, если значения всех ее полей атомарные, и в ней отсутствуют повторяющиеся группы полей.*

На «заре» существования реляционных баз данных на количество полей в записи накладывались определенные ограничения. Как следствие, разработчики объединяли несколько предполагаемых полей в одно, чтобы все нужные данные поместить в одну запись. Известно, что если поле содержит несколько значений, то существенно усложняется формирование отношений между полями, считывание данных и выполнение других операций, а необходимость выполнения поиска подстрок и синтаксического анализа полей в значительной степени замедляет работу приложения. К счастью, сейчас все ограничения на количество полей в записи сняты.

Приведем наши данные к первой нормальной форме. Выделим самостоятельные группы полей и поместим их в отдельные таблицы. На первый взгляд их четыре. Это информация об адресе, здании, квартире и собственниках. Добьемся атомарности всех полей. Поле ***FioHost***, в которое записывается информация о фамилии, имени и отчестве ответственного квартиросъемщика, заменим тремя полями: ***Family***, ***Name***, ***Second***. Также поступим и с проживающими в квартире. Поле ***Address*** разобьем на

три: название, признак и порядок их следования в официальных документах. Получится следующая картина (табл. 2.1).

Таблица 2.1

Информация об адресе (Street)

№	Поле	Тип	Размер	Описание
1	Street	Числовой	4	Номер улицы
2	Name	Текстовый	30	Название улицы
3	Sign	Текстовый	10	Признак адреса
4	First	Логический	1	Порядок следования в документах

Street	Name	Sign	First
173	Воронежская	Улица	Ложь
174	Воронежский	проезд	Истина
175	Воронежское	шоссе	Истина
176	Ворошилова	Улица	Ложь

Если значением поля ***First*** является Ложь, то при формировании адреса здания в официальных документах на первое место будет поставлен признак: Улица Ворошилова, а если Истина – название: Воронежское шоссе или Воронежский проезд. Обратите внимание на заполнение поля ***Sign***. Если в поле ***First*** стоит Ложь, то значение признака пишется с большой буквы.

Обратите внимание на то, как легко будет сейчас решаться проблема переименования улицы. Допустим, что отныне Воронежское шоссе, стоящее под номером 175 в таблице ***Street,*** переименовано, например, в улицу Муравьева-Амурского. Вносим исправления только в таблицу ***Street***. Оставляем этот номер, меняем название, признак и значение поля ***First*** с Истина на Ложь. Проблема решена. Так как во всех остальных таблицах Воронежское шоссе (улица Муравьева-Амурского) фигурирует под номером 175, то никакие изменения не требуются.

Таблица 2.2

Информация о здании (Building)

№	Поле	Тип	Размер	Описание
1	Street	Числовой	4	Ссылка на номер улицы
2	House	Текстовый	4	Номер дома
3	District	Текстовый	15	Район города
4	Land	Числовой	10	Площадь земельного участка
5	Year	Числовой	4	Год постройки здания
6	Material	Текстовый	15	Материал стен здания
7	Comment	Поле Memo	Авто	Примечания
8	Wear	Числовой	2	Износ в процентах
9	Cost	Денежный	15	Стоимость здания в рублях
10	Line	Числовой	5	Расстояние от центра города
11	Square	Числовой	10	Площадь нежилых помещений
12	Picture	Поле OLE	Авто	Фото здания
13	Kind	Числовой	1	Вид собственности
14	Elevator	Логический	1	Наличие лифта

Таблица 2.3

Информация о квартире (Flat)

№	Поле	Тип	Размер	Описание
1	Street	Числовой	4	Ссылка на номер улицы
2	House	Текстовый	4	Номер дома
3	Flat	Числовой	4	Номер квартиры
4	Storey	Числовой	2	Номер этажа
5	Rooms	Числовой	1	Количество комнат
6	SquareFlat	Числовой	Авто	Общая площадь квартиры
7	Dwell	Числовой	Авто	Жилая площадь квартиры
8	Branch	Числовой	Авто	Вспомогательная площадь квартиры
9	Balcony	Числовой	Авто	Площадь балкона
10	Height	Числовой	Авто	Высота квартиры
11	Account	Числовой	5	Номер лицевого счета
12	Family	Текстовый	20	Фамилия квартиросъемщика
13	Name	Текстовый	20	Имя квартиросъемщика
14	Second	Текстовый	20	Отчество квартиросъемщика
15	Pasport	Поле Memo	Авто	Данные его паспорта

Таблица 2.4

Информация о проживающих в квартире (Owners)

№	Поле	Тип	Размер	Описание
1	Street	Числовой	4	Ссылка на номер улицы
2	House	Текстовый	4	Номер дома
3	Flat	Числовой	4	Номер квартиры
4	Number	Числовой	2	Порядковый номер проживающего
5	Family	Текстовый	20	Фамилия проживающего
6	Name	Текстовый	20	Имя проживающего
7	Second	Текстовый	20	Отчество проживающего
8	Born	Числовой	4	Год рождения проживающего
9	Status	Текстовый	20	Льготы и статус проживающего

Удовлетворение требованиям первой нормальной формы называется *структурной или синтаксической нормализацией.*

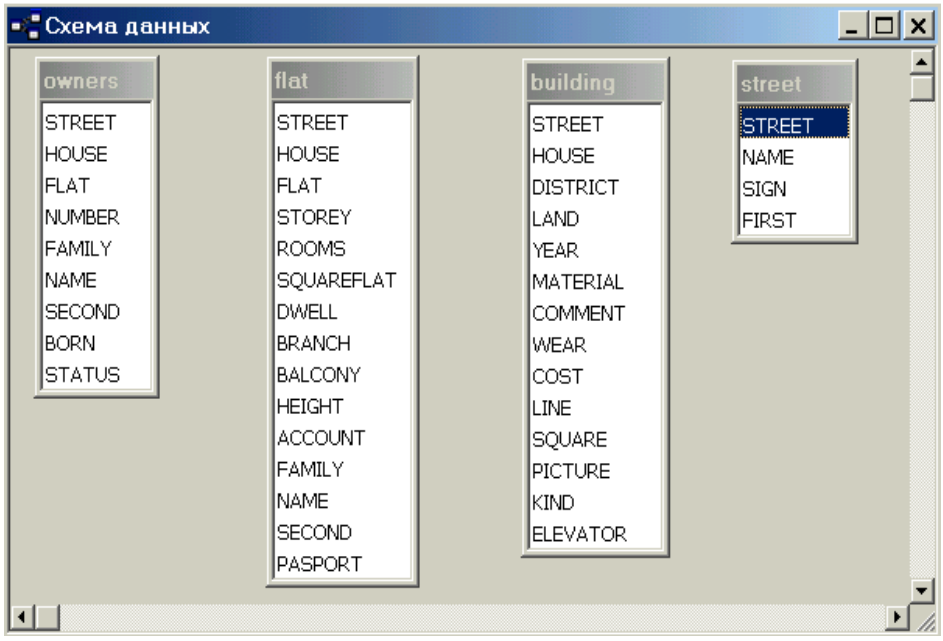


Рис. 2.2. Таблицы базы данных

Данные разделены (табл. 2.1, 2.2, 2.3, 2.4) на четыре родственные группы: улицы, здания, квартиры и проживающие (рис. 2.2). Значения всех полей этих таблиц – атомарные. Все таблицы находятся в первой нормальной форме. Однако останавливаться на этом не следует. С такими данными все еще возможно возникновение проблем. Прежде всего в базе данных много повторений значений – не внутри одной записи, а в преде-

лах одной таблицы. А там, где есть повторяющиеся значения, возможны противоречия. Посмотрите на поля ***Material*** и ***District*** таблицы ***Building***. Та же картина, которая имела место чуть раньше с названиями улиц. Варианты названий материала стен: шлакобетон, шлакобетонные, шлб, шл.бет. Уберем название материала стен и названия районов в отдельные таблицы – справочники (***Wall*** и ***District***), оставив в основной таблице ***Building*** ссылки на эти справочники. База данных примет более правильный вид (табл. на рис. 2.3).

Рис. 2.3. Таблицы базы данных в первой нормальной форме

Появились еще две таблицы: ***Wall*** и ***District*** (табл. 2.5, табл. 2.6).

Таблица 2.5

Информация о районах города (District)

№	Поле	Тип	Размер	Описание
1	District	Числовой	1	Номер района
2	Area	Текстовый	15	Название района

Таблица 2.6

Информация о материале стен здания (Wall)

№	Поле	Тип	Размер	Описание
1	Material	Числовой	1	Номер материала
2	Wall	Текстовый	15	Название материала

Структура таблицы ***Building*** несколько изменилась. Вместо описаний района и материала стен появились ссылки на соответствующую табл. 2.7.

Таблица 2.7

Окончательная структура таблицы *Building*

№	Поле	Тип	Размер	Описание
1	Street	Числовой	4	Ссылка на номер улицы
2	House	Текстовый	4	Номер дома
3	District	Числовой	1	Ссылка на район города
4	Land	Числовой	10	Площадь земельного участка
5	Year	Числовой	4	Год постройки здания
6	Material	Числовой	1	Ссылка на материал стен здания
7	Comment	Поле Memo	Авто	Примечания
8	Wear	Числовой	2	Износ в процентах
9	Cost	Денежный	15	Стоимость здания в рублях
10	Line	Числовой	5	Расстояние от центра города
11	Square	Числовой	10	Площадь нежилых помещений
12	Picture	Поле OLE	Авто	Фото здания
13	Kind	Числовой	1	Вид собственности
14	Elevator	Логический	1	Наличие лифта

Вторая нормальная форма. Таблица находится во второй нормальной форме, если она удовлетворяет условиям первой нормальной формы, и любое неключевое поле однозначно идентифицируется полным набором ключевых полей.

Настало время поговорить о ключевых полях. Мощь реляционных баз данных, таких как Microsoft Visual FoxPro, опирается на их способность быстро найти и связать данные из разных таблиц при помощи запросов, форм и отчетов. Для этого каждая таблица должна содержать одно или несколько полей, однозначно определяющих каждую запись в таблице. Такие поля называют *первичным ключом таблицы*. Если для таблицы определен первичный ключ, то Microsoft Visual FoxPro предотвращает дублирование значений полей или ввод значений Null в эти поля. В Microsoft Visual FoxPro можно выделить три типа ключевых полей: *простой ключ, составной ключ и счетчик (Integer AutoInc).* Если поле содержит уникальные значения, то его можно определить как ключевое или простой ключ. Примеры из нашей реальной жизни: идентификационный номер налогоплательщика, однозначно определяющий каждого жителя нашей страны,

номер свидетельства пенсионного фонда, кадастровый номер земельного участка, реестровый номер строения, номер автомобиля – все это уникальные номера в пределах страны. Поле ***Street*** (номер улицы) в таблице ***Street*** также можно определить как простой ключ. Этим же требованиям отвечают поля ***District*** (номер района) и ***Material*** (номер материала) таблиц ***District*** и ***Wall***. Можно смело гарантировать их уникальность в пределах нашего программного комплекса. С таблицей ***Building***, содержащей информацию о зданиях, при определении первичного ключа нужно поступить таким образом. К нашим услугам составной ключ. Связка полей – номер улицы плюс номер дома – однозначно определит положение записи, относящейся к одному зданию в этой таблице. С однозначным определением квартиры в таблице ***Flat*** (квартиры) дело состоит чуть сложнее. Составной первичный ключ выглядит так: номер улицы плюс номер дома плюс номер квартиры.

В очень редких случаях с определением первичного ключа для таблицы может сложиться тупиковая ситуация. Не отчаивайтесь, добавьте в таблицу поле и определите его тип как «Integer (AutoInc)». Все остальное Visual FoxPro сделает самостоятельно. В это поле будет автоматически вноситься уникальное число даже при работе с Вашей базой в сетевом варианте (с нескольких компьютеров одновременно).

Третья нормальная форма. *Таблица находится в третьей нормальной форме, если она удовлетворяет условиям второй нормальной формы и ни одно из неключевых полей таблицы не идентифицируется с помощью другого неключевого поля.*

Посмотрите внимательно на таблицу ***Flat*** (квартиры). Она содержит неключевое поле ***Account*** (номер лицевого счета), которое однозначно определяет ответственного квартиросъемщика (поля: ***Family***, ***Name***, ***Second*** и ***Pasport***) в этой таблице. Уберем все эти поля в еще одну таблицу ***Account*** и назначим в ней в качестве простого первичного ключа поле ***Account*** (табл. 2.8).

Таблица 2.8

Информация об ответственном квартиросъемщике (Account)

№	Поле	Тип	Размер	Описание
1	Account	Числовой	5	Номер лицевого счета
2	Family	Текстовый	20	Фамилия квартиросъемщика
3	Name	Текстовый	20	Имя квартиросъемщика
4	Second	Текстовый	20	Отчество квартиросъемщика
5	Pasport	Поле Memo	Авто	Данные его паспорта

Осталось установить связи между таблицами, и база данных будет готова к работе. Microsoft Visual FoxPro поддерживает три типа связей: один к одному, один ко многим и много к одному.

Связь «один к одному» означает, что каждой записи одной таблицы соответствует только одна запись другой таблицы и наоборот. В качестве примера рассмотрим связь между таблицами ***Flat*** и ***Account*** (рис. 2.4).

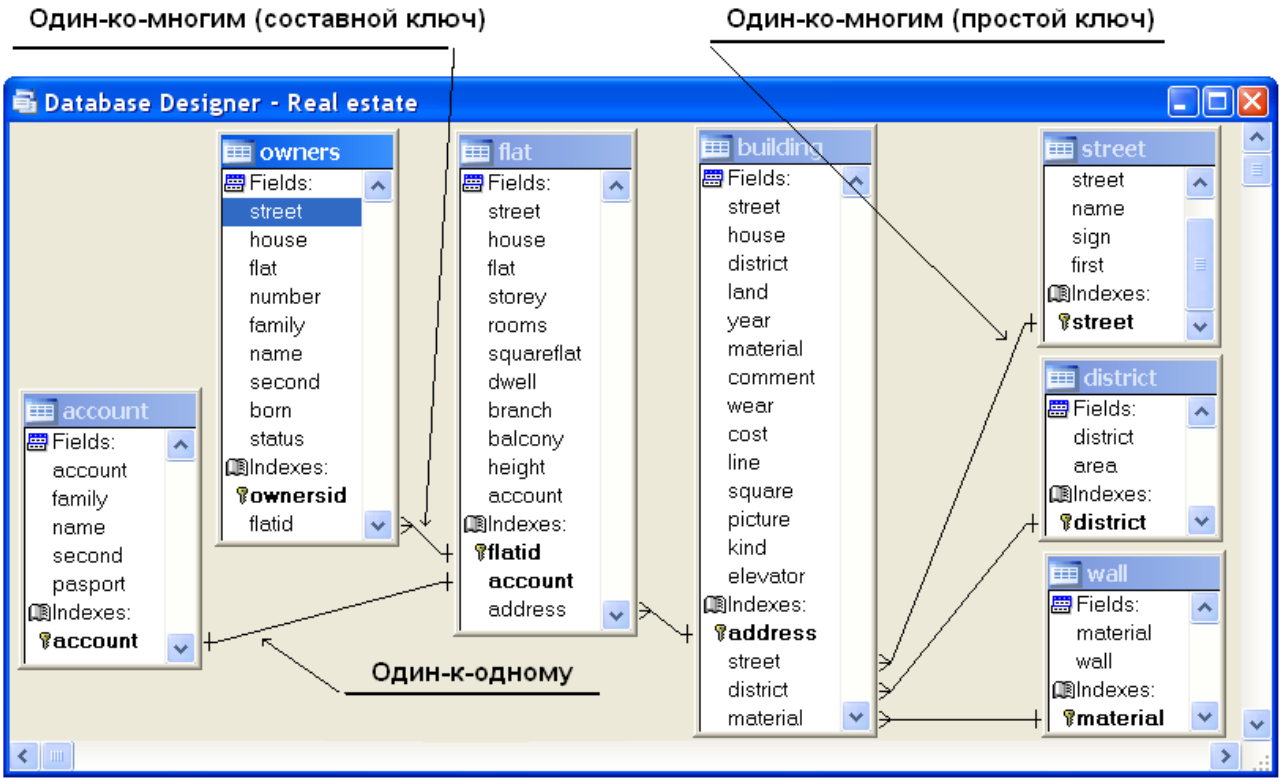


Рис. 2.4. Схема связей между таблицами

Одна квартира – один ответственный квартиросъемщик. Связь между ними поддерживается при помощи совпадающих полей ***Account***. Обратите внимание! У полей, используемых для связи, одинаковое наименование (***Account***) и тип (числовой с 5 разрядами). Всегда придерживайтесь этого правила при определении полей для связи любого типа между таблицами. Хотя, если быть более точным, связь между таблицами устанавливается на основании значений совпадающих полей, а не их наименований.

Связь «один ко многим». В качестве иллюстрации данного типа связи обратимся к таблицам ***Street*** и ***Building***. Одной улице в таблице улиц ***Street*** соответствует несколько зданий из таблицы зданий ***Building***. Связь между ними осуществляется на основании значений совпадающих полей ***Street***. Используется простой первичный ключ таблицы ***Street***. В качестве других примеров могут быть рассмотрены таблицы ***Building*** и ***Flat***, ***Flat*** и ***Owners***. Одному зданию соответствуют несколько квартир, а одной квар-

тире – несколько собственников. Для связи этих таблиц используются составные первичные ключи.

Связь «много к одному» аналогично ранее рассмотренному типу «один ко многим». Тип связи между объектами полностью зависит от вашей точки зрения. Например, если вы будете рассматривать связь между собственниками и квартирой, то получите много к одному. Несколько собственников проживают в одной квартире.

Связь «многие ко многим» возникает между двумя таблицами в тех случаях, когда одна запись из первой таблицы может быть связана более чем с одной записью из второй таблицы, а одна запись из второй таблицы может быть связана более чем с одной записью из первой таблицы. Таких связей следует избегать, так как реляционная модель не позволяет непосредственно работать с ними. Microsoft Visual FoxPro или любая другая реляционная СУБД в этом случае бесполезны. Всегда можно ввести в базу данных еще одну-две промежуточные таблицы и тем самым избежать возможных неприятностей при разработке интерфейса вашего приложения, используя понятные и безотказно работающие связи «один ко многим». Некоторые варианты заданий из этого пособия могут привести к связи «многие ко многим» между таблицами базы данных.

Обратившись к материалам подразд. 3.8, Вы увидите мое видение решения этой проблемы одного из вариантов курсового проекта.

Что за третьей нормальной формой? Если вы довели уровень нормализации таблиц вашей базы данных до третьей нормальной формы и ваша задача – разработка системы масштаба предприятия, то смело можете переходить к разработке интерфейса. Однако если вы участвуете в разработке суперхранилища данных под Oracle или DB2, то разберитесь по специальной литературе с нормальной формой Бойса-Кодда, четвертой и пятой нормальными формами.

3. РАЗРАБОТКА БАЗЫ ДАННЫХ

3.1. Запуск Microsoft Visual FoxPro

Для запуска Microsoft Visual FoxPro на вашем компьютере нажмите кнопку «Пуск» и выберите в открывшемся главном меню пункт «Все программы». В списке программ найдите «Microsoft Visual FoxPro 9» – FoxPro установлен там по умолчанию. Щелчок левой кнопкой мыши даст следующую картинку (рис. 3.1).

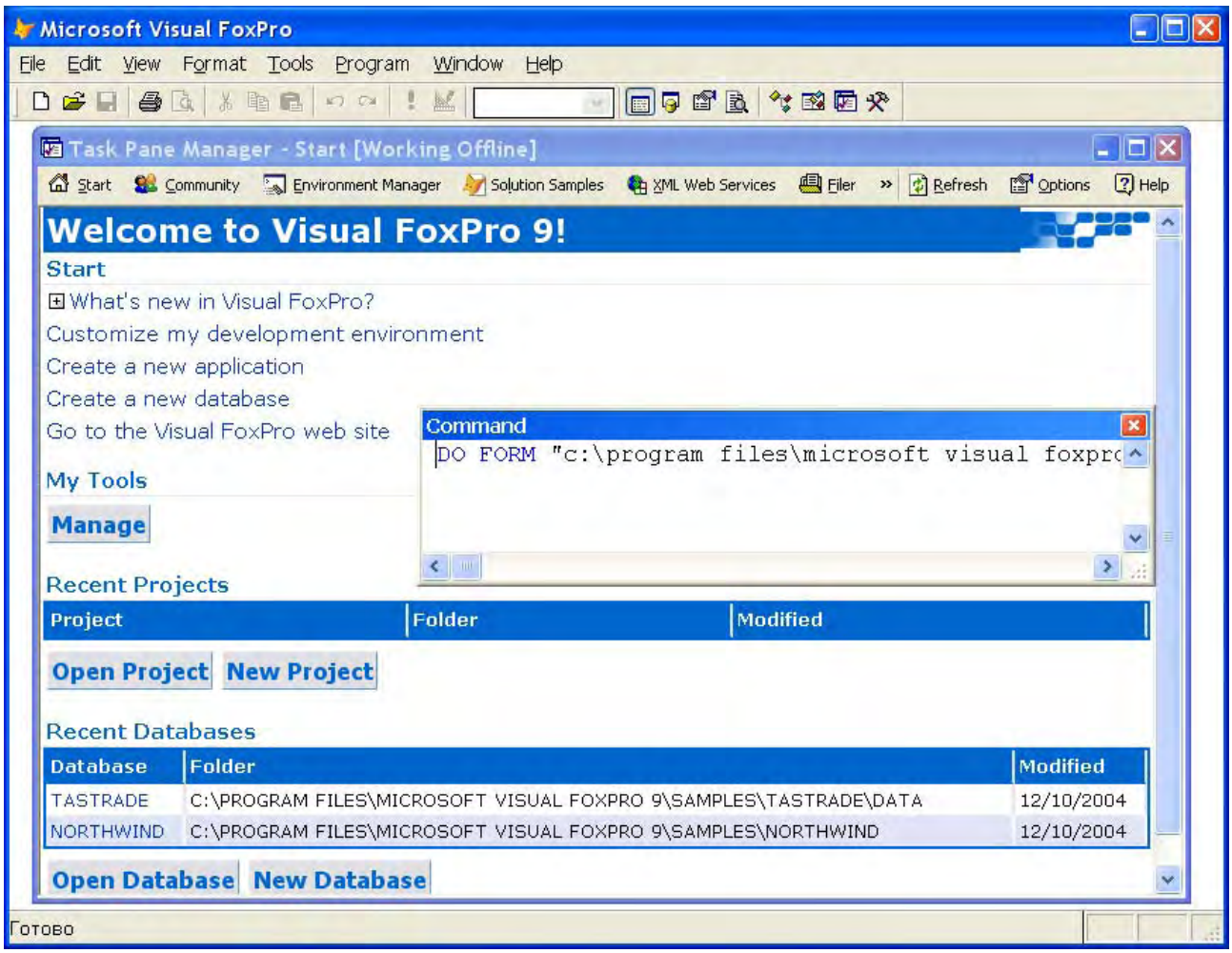


Рис. 3.1. Главное окно Microsoft Visual Fox Pro 9.0

На экране появится главное окно Visual FoxPro, окно менеджера панели задач (Task Pane Manager), в нижней части которого содержится список ранее открывавшихся баз данных, и командное окно (Command) с последними, набранными в нем командами.

Команду Visual FoxPro можно выполнить, набрав ее в окне ***Command*** и нажав на клавишу ***Enter***. Для повторного выполнения команды поместите курсор на строку с командой и нажмите ***Enter*** еще раз.

Так как в окне ***Command*** допускается редактирование, можно изменить команду инструментами редактирования, доступными в Visual FoxPro. Эти инструменты позволяют редактировать, вставлять, удалять, вырезать и помещать текст в окне ***Command***. Преимущество использования окна ***Command*** состоит в немедленном выполнении инструкций. Нет необходимости сохранять файл и запускать его как программу.

Кроме того, в окне ***Command*** сохраняются в виде команд все действия, производимые с меню и диалоговыми окнами. Вы можете скопировать и разместить эти команды в программе Visual FoxPro. После этого програм-

му можно запускать сколько угодно раз, избегая многократного повтора одних и тех же действий.

Для комфортной работы создайте на рабочем столе ярлык Microsoft Visual FoxPro 9, в его свойствах в качестве рабочей папки укажите ту, в которой будет располагаться разрабатываемое приложение и данные. В нашем примере – это папка ***Real Estate***. Не устраивайте в ней «свалку», разместив одновременно все объекты приложения: таблицы, индексы, контейнер базы данных, формы, отчеты, классы и т. д.

Советую все объекты в этой папке расположить структурировано. Примерный вид папки с приложением может быть таким (рис. 3.2).

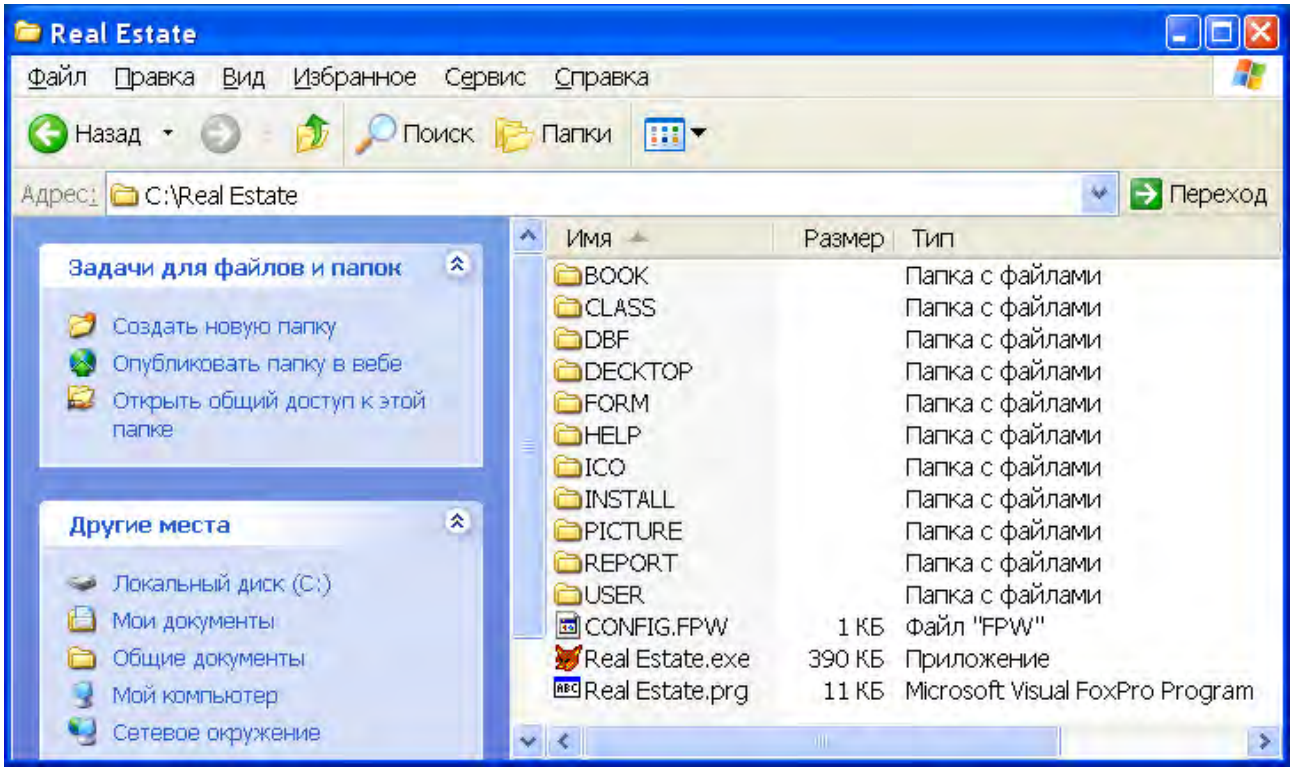


Рис. 3.2. Состав папки Real Estate

BOOK – папка, содержащая формы справочников.

CLASS – папка, содержащая пользовательские классы.

DBF – папка с данными (контейнер, таблицы, индексные файлы, поля примечаний и т. д.).

DECKTOP – в этой папке расположены картинки, используемые для оформления главного окна программного комплекса.

FORM – папка, содержащая главные формы приложения.

HELP – папка с файлами контекстуально-зависимой помощи, вызываемой при нажатии клавиши F1.

ICO – иконки для оформления интерфейса вашего приложения.

INSTALL – папка, содержащая файлы дистрибутива для инсталляции готового программного комплекса на клиентский компьютер.

PICTURE – фотографии зданий (в данном примере).

REPORT – папка, содержащая отчеты Visual FoxPro.

USER – папка, содержащая информацию о пользователях и паролях для работы с программным комплексом. В ней расположена фактически одна таблица. Это связано с тем, что в Visual FoxPro отсутствует система защиты от несанкционированного доступа средствами СУБД. Такое выделение отдельной папки позволит обеспечить защиту комплекса средствами операционной системы.

Config.fpw – файл, содержащий настройки Visual FoxPro. Его можно создать при помощи текстового редактора «Блокнот».

В большинстве случаев подойдет такой вид:

```
EXCLUSIVE=On
DELETED=ON
MULTILOCKS=ON
DATE=GERMAN
PATH=FORM,DBF,REPORT,CLASS,ICO
CENTURY=ON
RESOURCE=ON
```

3.2. Создание базы данных Visual FoxPro

В языке Visual FoxPro есть два основных пути доступа к данным и два способа их организации. Способами доступа являются: традиционная для Visual FoxPro навигация от записи к записи и более современные реляционные методы, основанные на SQL (Structured Query Language) – языке структурированных запросов. Организовать данные вы можете либо в автономные таблицы, либо в базу данных Visual FoxPro. Однако учтите: если вы оформляете их в виде автономных таблиц, то в значительной мере теряете контроль над управлением этими данными. Важно знать эти варианты, так как каждый из них имеет свои преимущества, и вы можете их сопоставлять и комбинировать. В данной книге будут использованы все четыре варианта, которые лежат в основе технологии разработки баз данных в Visual FoxPro.

Visual FoxPro объединяет в себе две основные парадигмы доступа в базах данных: навигационную и реляционную. Эти методы часто чередуют, поэтому полезно знать, что из себя представляет и в каких ситуациях подходит каждый из них. Навигационный доступ основывается на физическом или логическом порядке данных, проходя их от начала к концу или наоборот. Вы помещаете указатель где-нибудь в файле данных и получаете доступ к данным соответствующей записи. Чтобы работать с другими данными, вы переходите к той записи, где они расположены.

При навигационном доступе предпочитают использовать термины «файл», «запись» и «поле». Это потому, что в навигационных системах вы храните каждую таблицу как отдельный файл и перемещаетесь в нем от записи к записи, сверху вниз и обратно. Каждая запись содержит поля, в которых хранятся данные. Навигационный язык унаследовал многое от языка COBOL и до сих пор широко используется даже теми, кто работает с реляционными базами данных.

Реляционный доступ ссылается на данные только по их значениям и рассматривает эти данные как содержащиеся в неупорядоченных наборах. В этом случае вы применяете операцию не к какой-то части файла, а сразу ко всему набору данных. Стандартным языком реляционного доступа является SQL.

При реляционном доступе используются термины «таблица», «строка» и «столбец». В реляционной системе от вас скрыт способ хранения данных на диске. Вы просто обращаетесь к таблице, а затем к строкам и столбцам, в которых хранятся данные. Так как реляционный доступ не использует навигационных методов, вам не нужно знать ничего о физическом расположении строк в таблице.

В языке Visual FoxPro используется терминология и навигационного, и реляционного доступа. Однако лучше придерживаться реляционной терминологии, которая поможет вам разобраться в других реляционных базах данных и книгах по реляционной разработке.

Visual FoxPro использует оба названных метода доступа к данным по сложившейся традиции. Сначала применялся только навигационный метод, а позднее к нему добавился и реляционный. Чтобы понять, как и почему это произошло, рассмотрим, какие типы баз данных существуют и когда они разрабатывались.

Многообразие программного обеспечения баз данных может слегка обескураживать. Его архитектура сильно варьируется в зависимости от времени его создания, а также от технических и программных платформ, на которых это программное обеспечение основано. Важно различать системы баз данных, берущие начало от больших ЭВМ или систем UNIX, и те, которые появились только на персональных компьютерах. В свою очередь, базы данных для персональных компьютеров ведут свое происхождение от программ баз данных, которые тогда существовали. Поэтому имеет смысл посмотреть, какие же базы данных имелись к тому времени, когда впервые появились базы данных для персональных компьютеров.

Когда персональные компьютеры стали повсеместно использоваться для хранения баз данных, было принято делить большие системы баз данных на три основные категории: иерархические, сетевые и реляционные. Четвертый тип, объектно-ориентированные базы данных, появился позднее.

В иерархических системах данные организованы в древовидную структуру, где каждый узел дерева содержит определенный тип данных. В иерар-

хической базе данных, чтобы достигнуть любого узла, вы начинаете с корня, следуете указателю и спускаетесь до нужного уровня. Вы перемещаетесь по системе запись за записью, следуя включенным в них указателям, чтобы получить необходимые данные. Иерархические системы довольно эффективны, однако, они часто содержат избыточные данные. Кроме того, для получения доступа к любым данным надо проходить через корневой узел, а это не всегда удобно. Иерархические системы баз данных – такие, как IMS/VS, которые принадлежат IBM – представляют старейшее поколение систем баз данных и берут свое начало от больших ЭВМ.

В начале 1970-х гг. группа CODASYL выпустила стандарт для сетевых *баз* данных. В системах этого типа вы храните данные и имеете к ним доступ более свободный, чем в иерархических системах. В сетевых базах нет отдельного корневого узла, однако, они тоже хранят данные, используя указатели от родительских узлов к наборам подчиненных записей. В отличие от иерархических, сетевые системы легко реализуют отношения типа «многие-ко-многим». Однако обычно эти системы довольно сложны и требуют солидного программного обеспечения. В них тоже применяется навигация от записи к записи, основанная на использовании вставленных указателей. В 1970-х гг. сетевые системы баз данных приобрели популярность и стали применяться как для миникомпьютеров, так и для больших ЭВМ.

Реляционные системы баз данных исключили необходимость сложной навигации, так как данные предоставляются пользователю или программисту в виде независимых наборов. Эти наборы данных оформлены как таблицы. Они не содержат ни повторяющихся строк, ни многозначных данных в столбцах. Реляционные базы данных заменили навигацию в записях реляционной алгеброй – группой операций над таблицами. Поэтому пользователь не видит никаких вставленных указателей.

Е.Ф. Кодд, создатель реляционной модели, основывался на прикладной теории множеств, что позволило ему подготовить для своей разработки солидную теоретическую основу. Реляционная модель быстро получила признание, поскольку по своим потенциальным возможностям превосходила все остальные. Правда, до 1980-х гг. она не имела особого коммерческого успеха и редко применялась на практике. Однако впоследствии почти все новые программы баз данных создавались как реляционные или, по крайней мере, претендовали на это. Так было вплоть до появления объектно-ориентированных баз данных в 1990-х гг. Подавляющее большинство баз данных, которые используются сегодня, построены по реляционной модели.

Visual FoxPro имеет сложную родословную. Он относится к семейству программных продуктов, которые, в свою очередь, ведут свое происхождение от dBASE II. Построенный по реляционной модели, dBASE II хранил данные в независимых таблицах без видимых для пользователя указателей. Для файлов данных, где хранились эти таблицы, было выбрано рас-

ширение DBF – сокращение от DataBase File (файл базы данных). Однако для управления данными dBASE II использовал навигационные команды перемещения.

Команды навигации ссылались на *указатель записи* – маркер, который система хранит для конкретной записи в файле данных. Когда вы хотели обработать какие-либо данные, вы просто обращались к соответствующей записи и вызывали необходимые команды. Такая навигация делала dBASE II гораздо более плоской и примитивной обрабатывающей системой, чем реляционные базы данных.

Вскоре фирма Nantucket представила компилятор Clipper, эффективно клонировав dBASE III+, а Fox Software произвела свой клон FoxBase+. Все эти продукты баз данных для персональных компьютеров сначала стали известны как «семейство dBASE», а позже – как «семейство xBase».

В то самое время, когда dBASE II и его потомство продолжали распространяться в мире персональных компьютеров, компания IBM изобрела язык SQL, который предоставил пользователю несколько весьма эффективных команд-запросов. Разработчики баз данных для персональных компьютеров вскоре захотели иметь эти команды среди своих инструментов.

Следующий важный шаг вперед для разработчиков на языке Fox произошел в 1990-х гг., когда компания Fox Software ввела SQL-команду Select в версию FoxPro 2.O. Программисты и пользователи FoxPro наконец получили возможность выбирать между навигационным и реляционным доступом к данным. С тех пор была проделана большая работа по дальнейшему внедрению SQL в FoxPro.

Недавно в язык Visual FoxPro была включена концепция Контейнера базы данных. Это весьма значительное расширение реляционных функциональных возможностей для структуры файлов DBF. Контейнер базы данных дает разработчикам множество средств таких, как ссылочная целостность и хранимые процедуры. Эти средства являются стандартными в реляционных базах данных.

Если и другой способ классификации баз данных. В наши дни приложения реляционных баз данных подразделяются на «настольные» и «серверные». Настольные базы данных помещают обработчик базы данных на персональный компьютер, который запускает программное обеспечение, тогда как серверные базы данных помещают его на отдельный серверный компьютер. Конечно, эта схема неполна: стоит упомянуть, что некоторые компании, производящие серверные базы данных, теперь продвигают на рынок упрощенные версии своей продукции, которые предназначены для задач меньшего масштаба. Это так называемые настольные серверные базы данных, и их можно запускать на пользовательских персональных компьютерах.

В 1990-х гг. внимание привлек новый тип баз данных – объектно-ориентированные. Фактически объектно-ориентированные базы данных хранят методы классов, а в некоторых случаях и постоянные объекты классов – в базе данных, чтобы осуществлять беспрепятственную интеграцию между хранением данных и объектно-ориентированными языками программирования.

Кроме того, недавно появилась гибридная разновидность базы данных, которая названа *объектно-реляционной.* Практически это реляционные базы данных со значительно расширенными возможностями, что позволяет им осуществлять такие функции, которые раньше были доступны только для объектно-ориентированных баз.

Язык Visual FoxPro можно отнести либо к реляционным, либо к навигационным языкам, предназначенным для обработки файлов. Это зависит от того, какие средства Visual FoxPro вы предпочтете использовать для хранения и доступа к данным. Даже при максимально благоприятных условиях Visual FoxPro содержит меньше реляционных возможностей, чем стандартные реляционные базы данных. В последние несколько лет в новых версиях Visual FoxPro наметился уклон в сторону увеличения реляционных характеристик обработчика баз данных и языка доступа к данным.

Хотя Visual FoxPro является объектно-ориентированным языком программирования, его база данных прочно основывается на реляционной модели и не имеет особенностей объектно-ориентированной базы данных. Поэтому, чтобы лучше понять структуру и функции Контейнера баз данных Visual FoxPro, вам нужно иметь представление о реляционной модели.

Что, кроме названия, делает конкретную базу данных реляционной? Ее соответствие реляционной модели базы данных – вот наиболее часто применяемый критерий. Иногда возникают довольно упорные дебаты по поводу того, каким условиям должна удовлетворять база данных, чтобы ее можно было считать реляционной. Е.Ф. Кодд предложил набор из 12 правил, по которым должен оцениваться продукт. В результате комплексной оценки у Visual FoxPro 6,5 балла из 12. Замечу, что у Oracle – 8,5 баллов.

Важной характеристикой реляционных СУБД является то, что для сохранности данных они используют транзакции. В транзакцию можно поместить одну или более операций над данными, так что выполняться будет либо сразу вся группа операций, либо ни одна из них. Если транзакция заканчивается неудачей, базу данных можно вернуть в исходное стабильное состояние.

Реляционные СУБД должны также обеспечивать поддержку параллельного доступа, т. е. методы блокировки. Необходимо, чтобы операции над данными, которые совершаются многими пользователями, не отменяли результатов друг друга, а база данных обеспечивала согласованное представление данных для каждого пользователя.

Перейдем к конкретным действиям по созданию базы данных. Запустите Microsoft Visual FoxPro на вашем компьютере и воспользуйтесь одним из трех способов.

- В окне менеджера панели задач (Task Pane Manager) на его первой вкладке ***Start*** выберите ссылку ***New Database***. На экране дисплея появится окно ***Create***. Перейдите в папку ***DBF***, в которой мы будем хранить данные, задайте имя базы данных – ***Real Estate***. Нажмите кнопку ***Сохранить***. База данных получила свое название и законное место на жестком диске.
- В главном меню Visual FoxPro щелкните пункт ***File*** и выберите команду ***New***. В открывшемся окне щелкните радиокнопку ***Database*** и нажмите кнопку ***New file.*** На экране дисплея появится окно ***Create***. Перейдите в папку ***DBF***, в которой мы будем хранить данные, задайте имя базы данных – ***Real Estate***. Нажмите кнопку ***Сохранить***.
- Введите в командном окне (Command) команду ***Create Database***. На экране дисплея появится окно ***Create***. Перейдите в папку ***DBF***, в которой мы будем хранить данные, задайте имя базы данных – ***Real Estate***. Нажмите кнопку ***Сохранить***.

Теперь в папке ***DBF*** появилось три файла – так называемый контейнер базы данных. Это файлы:

Real Estate.dbc
Real Estate.dct
Real Estate.dcx

Отличительной особенностью Microsoft Visual FoxPro является то, что один объект этой СУБД размещается в нескольких файлах (от одного до трех). Расширения других файлов, используемых Visual FoxPro, будут рассмотрены нами позднее в соответствующих главах.

3.3. Создание таблиц

Visual FoxPro 9.0 позволяет создавать как таблицы, входящие в базу данных, так и свободные таблицы. Это дань истории развития продукта – черта, свойственная только ему. Так уж сложилось со времен появления Visual FoxPro 3.0. Воспользуемся ей.

Будут применяться таблицы обоих типов. Свободная (вне базы данных) таблица ***User*** даст возможность отрегулировать вопрос о правах доступа к приложению (рис. 3.2), а все остальные, размещенные в контейнере ***Real Estate***, представляют реляционную базу данных.

Существует несколько способов создания таблиц в Microsoft Visual FoxPro:

1) с помощью мастера таблиц;
2) с использованием конструктора таблиц;
3) путем импорта данных из внешнего файла в текущую базу данных;
4) создание их в текущей базе данных, связанных с таблицами внешнего файла.

Первый способ – для новичков в работе с базами данных, испытывающих сильное желание немедленно создать свою первую таблицу.

Процесс создания таблицы с помощью мастера включает в себя несколько этапов. Вам будет предложено несколько десятков образцов таблиц делового и личного применения. Выбирайте нужные детали и проектируйте свою таблицу! Два последних применяются, как правило, в процессе доработки уже готового и запущенного в эксплуатацию программного комплекса. Если же вы заняты разработкой нового приложения – забудьте про все способы, кроме второго. Конструктор таблиц – вот то, что вам необходимо в этом случае!

Воспользовавшись главным меню Visual Fox Pro или окном менеджера панели задач откройте базу данных ***Real Estate***, созданную в подразд. 3.1. Появится окно конструктора базы данных (рис. 3.3). Дальше три пути на ваш выбор. Все они приведут к цели. Такое оформление характерно для многих продуктов корпорации Microsoft. Это пересекающиеся каскады.

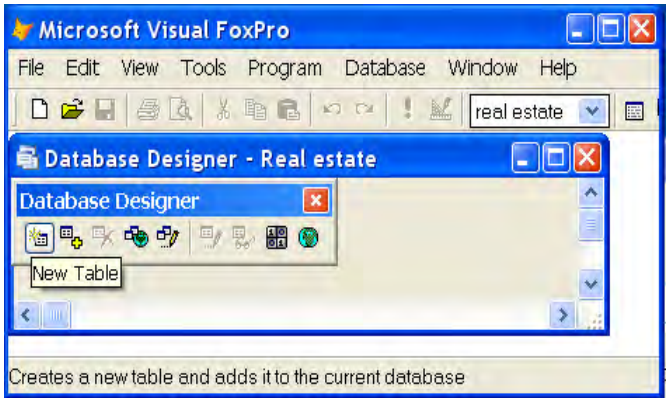


Рис. 3.3. Окно конструктора базы данных Real Estate

Каскад первый. Щелкните по первой иконке панели инструментов конструктора базы данных. Под ней вы найдете подсказку ***New Table***. Появится окно New table, позволяющее выбрать запуск мастера таблиц или конструктора таблиц. Щелкните в нем по кнопке ***New table***. Произойдет запуск конструктора таблицы (рис. 3.6).

Каскад второй. Щелкните правой кнопкой мыши в любом месте окна конструктора базы данных. Появится всплывающее меню (рис. 3.4). Выберите в нем пункт ***New table***. Появится окно с заголовком New table, позволяющее выбрать запуск мастера таблиц или конструктора таблиц.

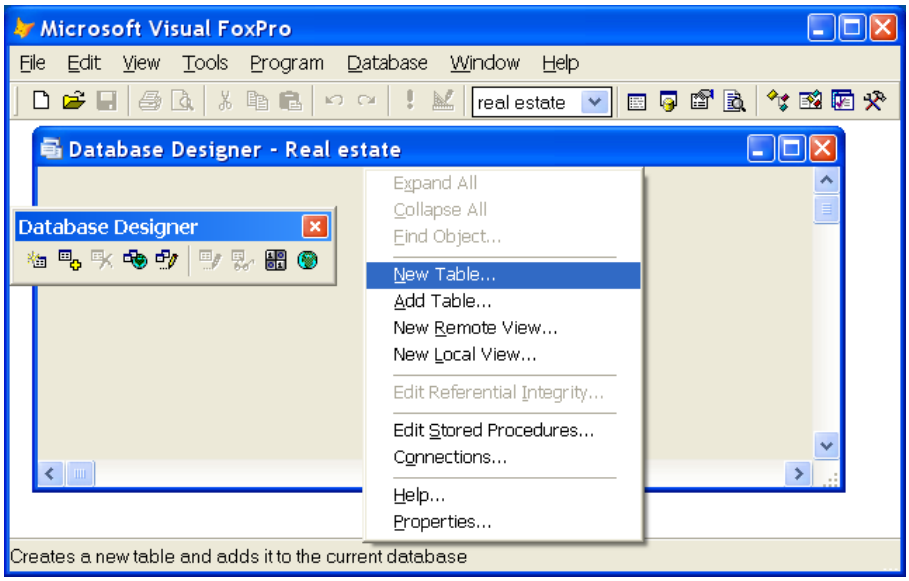


Рис. 3.4 Всплывающее меню конструктора базы данных

Щелкните в нем по кнопке ***New table***. Произойдет запуск конструктора таблицы (рис. 3.6).

Каскад третий. В главном меню Visual FoxPro выберите пункт ***Database***. Появится всплывающее подменю (рис. 3.5). Выберите в нем пункт ***New table***. Появится окно New table, позволяющее выбрать запуск мастера таблиц или конструктора таблиц. Щелкните в нем по кнопке ***New table***. Произойдет запуск конструктора таблицы (рис. 3.6).

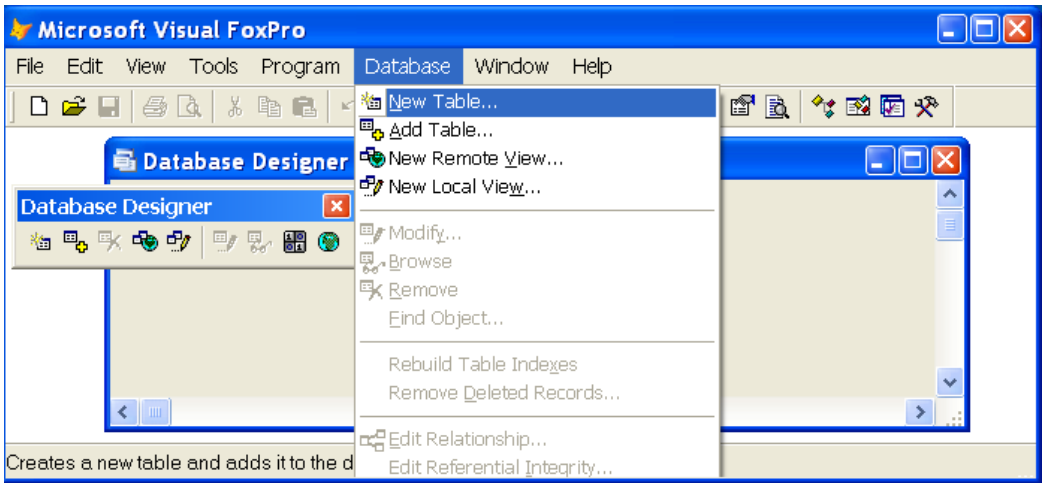


Рис. 3.5. Работа в главном меню Visual FoxPro 9.0 с базой данных

Во всех трех случаях увидим окно конструктора таблицы. Это форма с тремя вкладками: ***Fiels***, ***Indexes*** и ***Table***.

- ***Fields*** (Поля) – поля создаваемой таблицы и условия достоверности вводимых данных на уровне поля;
- ***Indexes*** (Индексы) – индексы создаваемой таблицы;
- ***Table*** (Таблица) – условия достоверности вводимых данных на уровне записи, а также триггеров добавления, удаления и модификации.

Первая вкладка (рис. 3.6) используется при определении полей таблицы. Она предназначена для ввода названия полей, типа данных и ширины поля. При занесении числовых полей задается количество десятичных знаков. Кроме основных параметров для каждого поля на этой вкладке задаются дополнительные параметры, которые помогут вам при сопровождении программного комплекса. Обратите особое внимание на ячейку ***Field comment***. Не оставляйте ее пустой! Опишите подробно назначение поля таблицы. Позже обязательно поймете важность этого совета.

Имена полей должны содержать не более 255 символов и могут включать любые комбинации символов за исключением точки, восклицательного знака и квадратных скобок. Используйте в именах полей только латинские буквы при общей длине имени до 10 символов включительно. В этом случае у Вас не будет проблем с конвертацией таблиц Microsoft Visual FoxPro в таблицы других СУБД (например, Microsoft SQL Server).

Каждое поле таблицы должно иметь уникальное имя, но в различных

таблицах можно использовать одинаковые имена полей. В табл. 3.1 приведены основные типы данных полей Microsoft Visual FoxPro 9.0.

Таблица 3.1

Типы данных полей таблиц Microsoft Visual FoxPro 9.0

Вид данных	Тип данных	Описание
Текстовый	Character	Текст или числа, не требующие проведения расчетов. Максимальная длина – 255 символов. По умолчанию длина текстового поля устанавливается равной 10 символов
	Character (binary)	Аналогичен Character. Используется в том случае, если не требуется учитывать кодовую страницу
	Varchar	Аналогичен Character. Для этого типа не происходит заполнение свободных мест незначащей информацией
	Varbinary	Шестнадцатеричные значения
Дата и время	Datetime	Любая дата от 01.01.0001 до 31.12.9999 и время от 00.00.00 a.m. до 11.59.59 p.m.
Дата	Date	Любая дата от 01.01.0001 до 31.12.9999
Денежный	Currency	Позволяет выполнять расчеты с точностью до 15 знаков в целой и до 4 знаков в дробной части
Логический	Logical	Содержит значение True (.T.) или False (.F.) Истина или ложь
Числовой	Integer	Целые числа от –2147483647 до 2147483648
	Integer (AutoInc)	Счетчик. Уникальные, последовательно возрастающие числа, автоматически вводящиеся в таблицу при добавлении каждой новой записи
	Numeric	Десятичные данные с фиксированной точкой
	Float	Десятичные данные с плавающей точкой простой точности
	Double	Десятичные данные с плавающей точкой высокой точности
Двоичное поле произвольной длины	General	Включает рисунок, фотографию, звукозапись, диаграммы, векторную графику, форматированный текст и т. п.
Текстовое поле произвольной длины	Мемо	Поля типа MEMO предназначены для хранения больших текстовых данных. Длина поля может достигать 64 Кб. Поле не может быть ключевым или индексированным
Двоичный	Blob	Двоичные символы

В правой части первой вкладки ***Fields*** конструктора таблицы (рис. 3.6) также расположены ячейки, позволяющие задать для каждого поля создаваемой таблицы свойства, которые будут задействованы при вводе данных.

Format (Формат) – задает формат отображения данных в окне Browse, отчетах и формах.

Input mask (Маска ввода) – задает шаблон для ввода данных.

Caption (Заголовок) – определяет заголовок поля.

Rule (Условие) – проверка правильности ввода данных на уровне поля создаваемой таблицы.

Message (Сообщение) – текст сообщения, которое появится на экране дисплея при неправильном вводе данных в поле.

Default value (Значение по умолчанию) – значение, вводимое в поле по умолчанию.

Display library (Показать библиотеку) – определяет имя файла библиотеки классов.

Display class (Показать класс) – определяет имя класса из выбранной библиотеки классов.

Next Value (Начальное значение) – задает начальное значение счетчика. Доступно только для поля IntegerInc.

Step (Шаг) – приращение счетчика. Доступно только для поля IntegerInc.

Создадим нашу первую таблицу ***Building***. Ее окончательная структура взята из табл. 2.7. Имейте в виду, что имя поля и его описание вводится с клавиатуры, а тип данных выбирается из списка.

Рассмотрим действия по созданию таблицы подробнее (рис. 3.6).

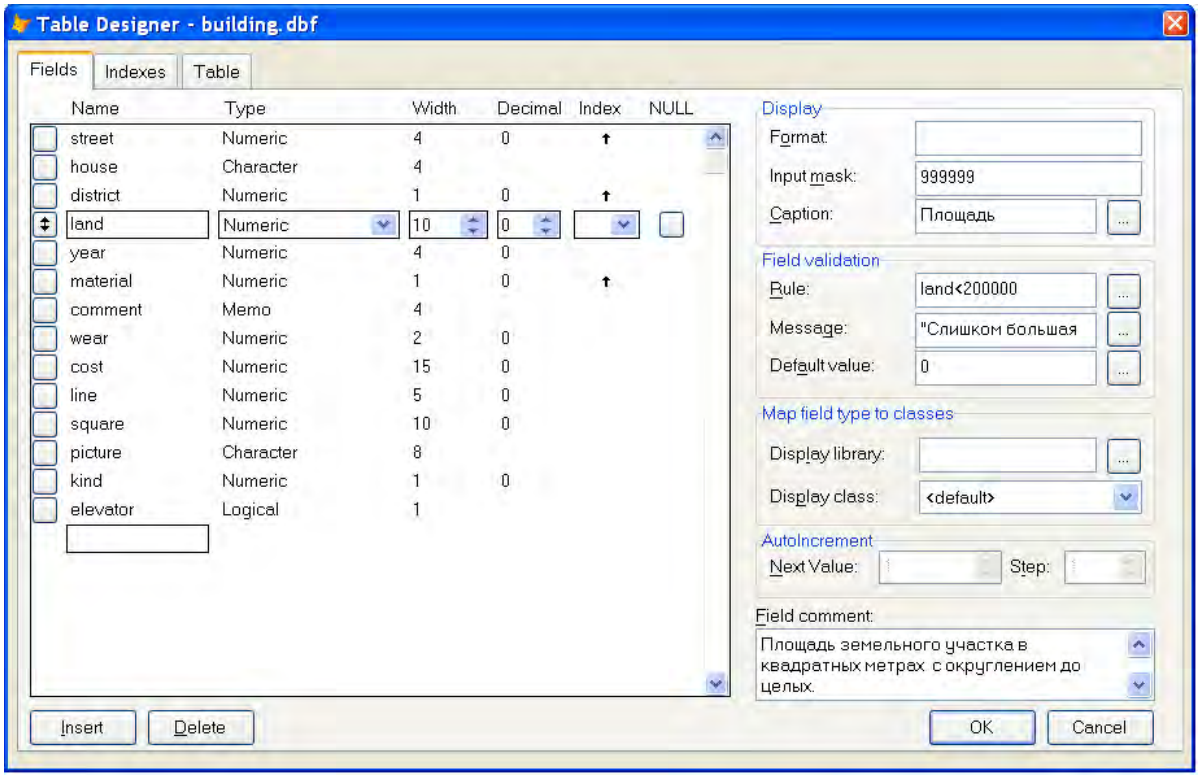


Рис. 3.6. Таблица ***Building*** в конструкторе таблиц (первая вкладка)

1. Введите в первую колонку имя первого поля: ***Street*** и нажмите клавишу <Tab>. Курсор переместится во вторую колонку ***Type***. По умолчанию будет назначен тип ***Character*** длиной 10 символов.

2. Раскройте список типов данных при помощи мыши. Выберите тип ***Numeric*** и нажмите клавишу <Tab>.

3. Заполните колонки ***Width*** (Число знаков) и ***Decimal*** (Число знаков после десятичной точки). Для ссылки на номер улицы используем четыре десятичных знака. Это дает возможность работать с 9999 улицами, что вполне достаточно для города с миллионным населением. Кстати, в Хабаровске 1248 улиц, переулков, проездов, шоссе, кварталов, площадей, бульваров и т. д.

4. Обязательно заполните ячейку ***Field comment***. Не ленитесь, пишите подробнее!

5. Повторите шаги 1–4 для всех оставшихся полей таблицы.

Остальные поля первой вкладки конструктора таблиц на этом этапе можно не заполнять. После занесения данных обо всех полях таблицы просто закройте окно конструктора таблиц. Появится диалоговое окно «***Сохранение***», запрашивающее подтверждение на сохранение структуры таблицы (рис. 3.7).

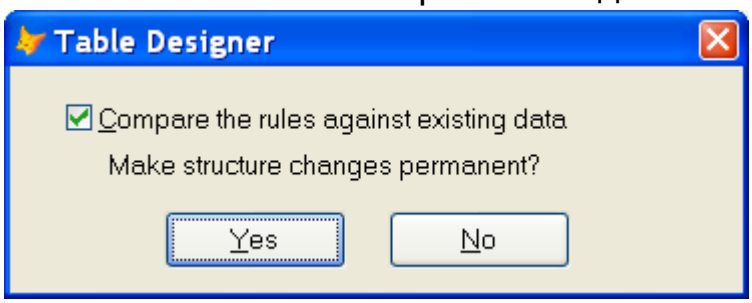


Рис. 3.7. Подтверждение сохранения

Щелкните по кнопке «Yes». Наша первая таблица появится в окне базы данных, а в папке DBF – два файла: ***Building.dbf*** (сама таблица) и ***Building.fpt*** (поле ***Memo***). Заполнять сейчас созданную таблицу начинающему пользователю категорически не рекомендуется, да так и не делается! Посмотрите на содержимое табл. 3.2. Это не отдельная таблица, она будет связана с другими таблицами базы данных ***Real Estate***. Что в ней хранится – пока загадка. Уверен, что большинство цифр Вам просто непонятны.

Таблица 3.2

Информация, содержащаяся в связанной таблице

Street	House	Flat	Storey	Rooms	Square	Dwell	Branch	Account
14	102	1	1	3	60,8	40	20	3450
14	102	2	1	4	100	70	28	1000
14	102	3	1	4	78	60	16	4321
14	102	4	2	4	90	80	5	666
14	102	5	2	3	100	95	30	778
14	102	6	10	1	200	190	8	9787
14	102	7	10	7	170	150	10	879
179	104	1	1	1	30	20	9	23210
179	104	2	1	2	42	30	11	3267

179	104	3	1	1	27	20	6	6666
179	104	4	2	4	100	90	5	4587

Аналогичным образом создадим все наши таблицы, разработанные в разд. 2: ***Flat***, ***Owners***, ***Account***, ***Street***, ***District*** и ***Wall***.

3.4. Создание первичных ключей и индексов

Одним из основных требований, предъявляемых к СУБД, является возможность быстрого поиска требуемых записей. В реляционных СУБД для реализации этого требования служат индексы. Индекс очень похож на алфавитный указатель в книге. Например, у вас в руках книга по Microsoft Visual FoxPro и вы хотите узнать о том, что написано в ней об индексах. Загляните в конец книги и найдите в предметном указателе слово «индекс». Так как указатель отсортирован по алфавиту, вы без труда найдете нужное слово и ссылки на страницы, где оно встречается в книге. Индекс работает с таблицей по такому же принципу. Он содержит отсортированные значения указанного поля таблицы и ссылки на номера записей таблицы, где эти значения находятся. При поиске записи система управления базами данных сначала просматривает индекс, что занимает совсем немного времени, так как для этого используется специальный алгоритм, находит ссылку на номер записи и по ней – нужную строчку в таблице. Таким образом, отпадает необходимость последовательного просмотра всех записей в таблице.

Индекс можно построить по полю почти любого типа. К счастью, пользователь не обязан знать, за счет чего достигается такое огромное увеличение скорости поиска. Достаточно создать индекс, а система Visual FoxPro позаботится обо всем остальном.

Посмотрим на конечный результат (рис. 3.8). На нем представлены индексы таблицы ***Flat***. Их три: ***Flat Id*** (Primary), ***Account*** (Candidate) и ***Address*** (Regular).

Рис. 3.8. Таблица ***Flat*** в конструкторе таблиц (вторая вкладка ***Indexes***)

Хочу предостеречь вас от типичной ошибки начинающего разработчика – создания индексов по всем полям таблицы для достижения максимальной скорости поиска в сложных запросах. Во-первых, в этом просто нет необходимости, а во-вторых – возникнет серьезная задержка при добавлении записей в таблицу, так как системе придется перестраивать большое число индексов одновременно.

Простой первичный ключ – это индекс, созданный по ключевому полю таблицы (Тип Visual FoxPro – Primary).

Составной первичный ключ – это индекс, созданный по ключевой связке полей таблицы (Тип Visual FoxPro также – Primary).

О том, как выбрать ключевое поле или назначить ключевую связку полей для таблицы, рассказано в разд. 2 («Вторая нормальная форма»).

Первичный ключ у любой таблицы может быть только один. Этого требует теория нормализации. Кроме первичного ключа таблица может иметь любое количество обычных индексов (Тип Visual FoxPro – Regular). Среди них могут быть и уникальные, не допускающие повторяющихся значений. Их принято называть индексы-кандидаты на роль первичного ключа (Тип Visual FoxPro – Candidate). В нашем примере такой индекс есть. Загляните в таблицу ***Flat*** (рис. 2.4) и обратите внимание на поле ***Account*** (номер лицевого счета квартиросъемщика). Это поле однозначно определяет положение любой квартиры в таблице.

А теперь о создании ключей и обычных индексов с самого начала и по порядку.

Создание простого первичного ключа. Создадим простой первичный ключ для таблицы ***Street*** (улицы). Ключевое поле, однозначно определяющее положение любой улицы в таблице улиц, также носит название ***Street*** (рис. 2.1).

1. Откройте таблицу ***Street*** в режиме конструктора. Для этого в окне базы данных ***Real Estate*** (рис. 2.1) щелкните по таблице ***Street*** правой кнопкой мыши. Появится меню. В нем пять пунктов. Выберите из них четвертый с названием ***Modify***.

2. Появится окно конструктора таблиц. Перейдите на вторую вкладку с названием ***Indexes*** (рис. 3.9). Увидите таблицу. В ней шесть колонок: ***Order***, ***Name***, ***Type***, ***Expression***, ***Filter и Collate***.

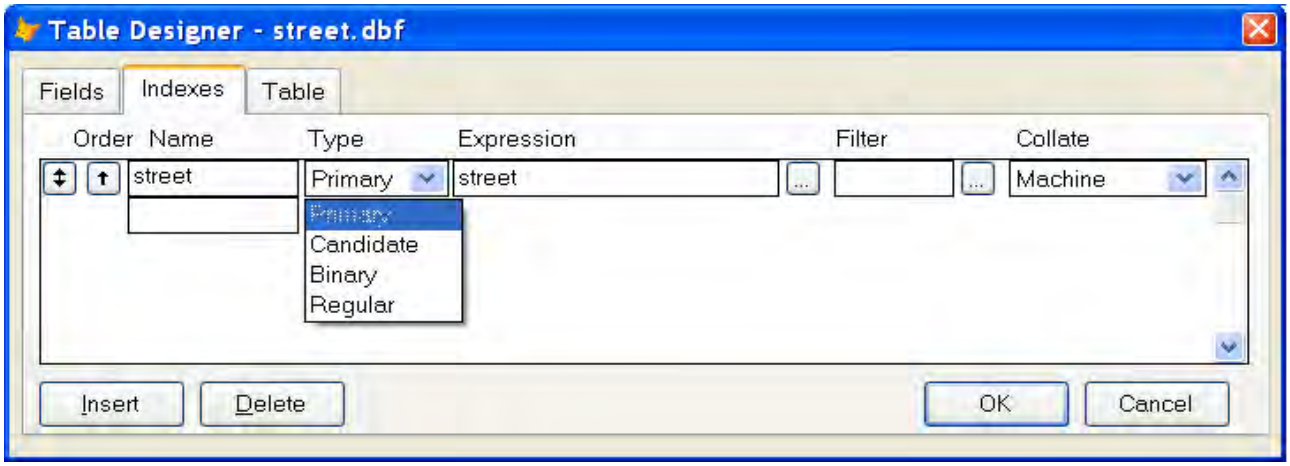


Рис. 3.9. Таблица ***Street*** в конструкторе таблиц (вторая вкладка ***Indexes***)

3. Во вторую колонку введите название индекса – ***street***.

4. Нажмите клавишу ***Tab*** для перехода в третью колонку ***Type***.

5. Откройте при помощи мыши поле со списком. В нем четыре значения: ***Primary***, ***Candidate***, ***Binary*** и ***Regular***. Выберите первое – ***Primary***.

6. В четвертую колонку введите название поля, по которому создается индекс (первичный ключ в данном случае) – ***street***. Остальные колонки заполнять не требуется.

7. Щелкните мышью по кнопке ***OK***. Visual FoxPro попросит подтвердить сделанные изменения. В появившемся окне выберите кнопку ***Yes***.

Создание составного первичного ключа. Создадим составной первичный ключ для таблицы ***Owners*** (Проживающие). Связка ключевых полей, однозначно определяющая положение любой записи в этой таблице, выглядит так: ***Street+House+Flat+Number***.

1. Откройте таблицу ***Owners*** в режиме конструктора. Для этого в окне базы данных ***Real Estate*** (рис. 2.1) щелкните по таблице ***Owners*** правой кнопкой мыши. Появится меню. В нем пять пунктов. Выберите из них четвертый с названием ***Modify***.

2. Появится окно конструктора таблиц. Перейдите на вторую вкладку с названием ***Indexes*** (рис. 3.10). Увидите таблицу. В ней шесть колонок: ***Order***, ***Name***, ***Type***, ***Expression***, ***Filter и Collate***.

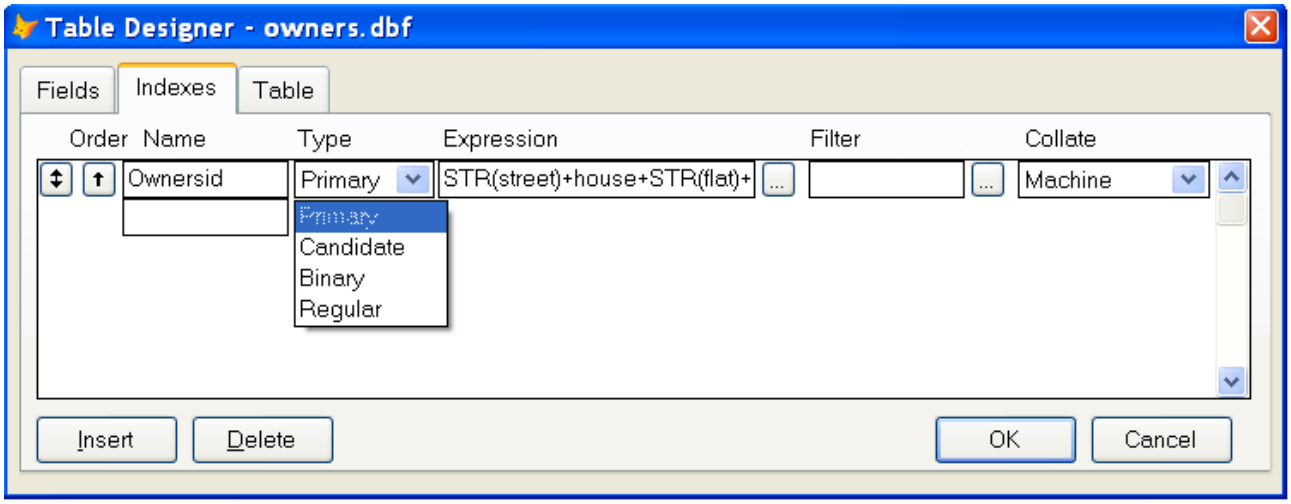


Рис. 3.10. Таблица ***Owners*** в конструкторе таблиц (вторая вкладка ***Indexes***)

3. Во вторую колонку введите название индекса – ***Ownersid***.

4. Нажмите клавишу ***Tab*** для перехода в третью колонку ***Type***.

5. Откройте при помощи мыши поле со списком. В нем четыре значения: ***Primary***, ***Candidate***, ***Binary*** и ***Regular***. Выберите первое, как и в предыдущем случае – ***Primary***.

6. В четвертую колонку с названием ***Expression*** необходимо ввести выражение для составного индекса. Воспользуемся построителем выражений. Для его запуска сделайте щелчок мышью по кнопке [...]. Она расположена справа

от колонки. Появится диалоговое окно построителя (рис. 3.11). Составные части выражения должны быть одного типа, поэтому использование функции ***STR*** для преобразования ***Numeric*** в ***Character*** – необходимость.

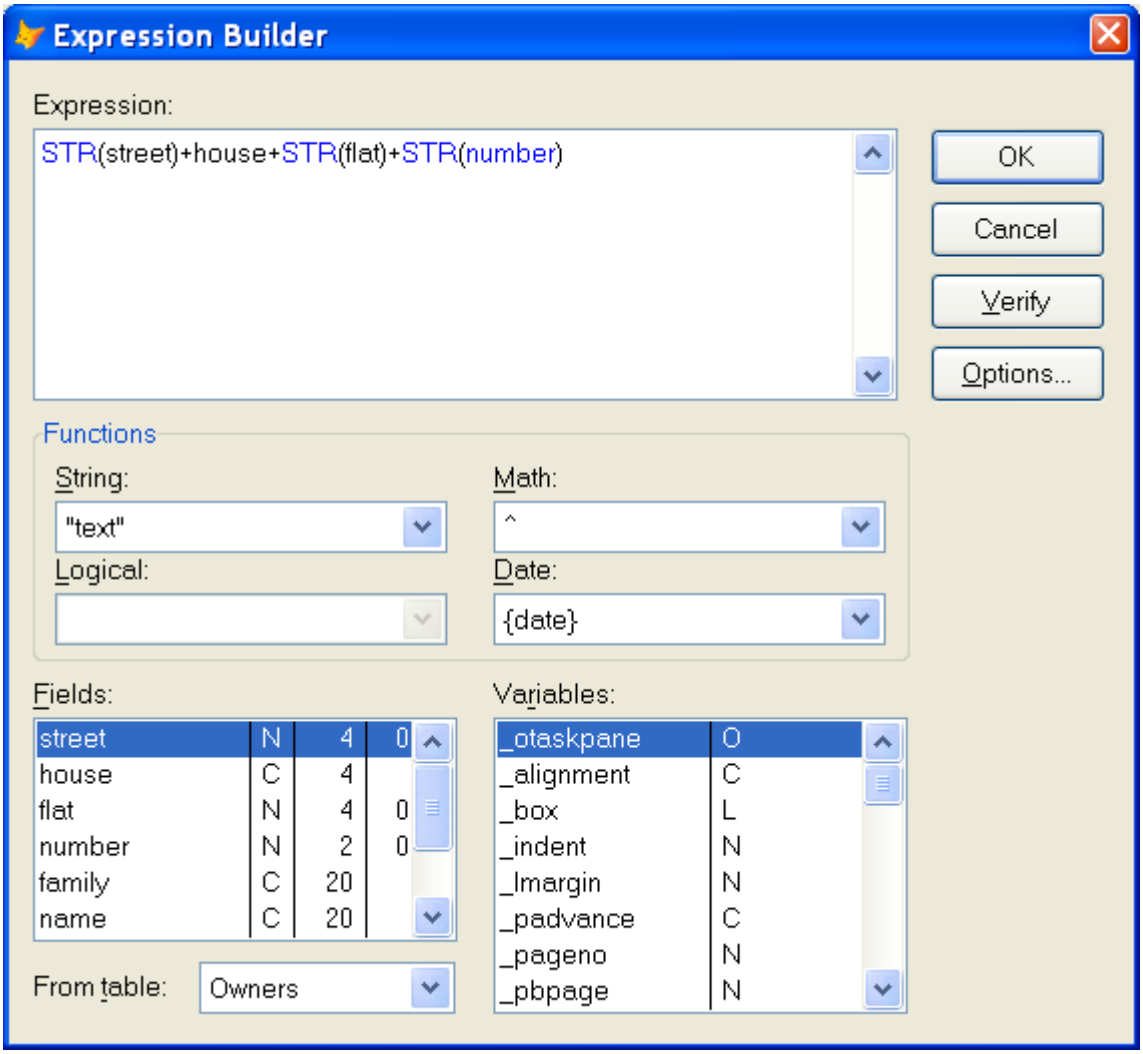


Рис. 3.11. Построитель выражений

7. Щелкните мышью по кнопке ***OK***. Visual FoxPro попросит подтвердить сделанные изменения. В окне выберите кнопку ***Yes***.

Создание обычного индекса по полю таблицы. Порядок создания как простого индекса (Regular), так и уникального (индекса-кандидата Candidate) – один и тот же и практически ничем не отличается от создания первичного ключа. Просто в пункте номер пять предыдущей инструкции выберите соответствующее значение (Regular или Candidate). Все индексы одной таблицы Visual FoxPro хранит в одном CDX-файле.

3.5. Контроль правильности ввода данных

Информация, накапливаемая в базе данных, должна обладать абсолютной достоверностью. Несоблюдение этого правила может порой при-

вести к печальным последствиям. Например, отдел комплектации не сделает вовремя заказ на поставку необходимых материалов, владелец квартиры получит квитанцию для оплаты налога на автотранспорт, которого у него никогда не было, а пенсионеру будет отказано в выдаче страхового полиса и т. д. Даже самые опытные пользователи, заполняющие таблицы, могут допустить ошибку и занести неверные данные, что, скорее всего, и произошло в перечисленных выше случаях.

Разработчик программного комплекса просто обязан помочь пользователю избежать большинства ошибок при вводе информации. Ниже приведены две возможности, которые любезно предоставили в наше распоряжение авторы Microsoft Visual FoxPro 9.0.

Добавление условия на значение поля позволяет проверить корректность данных только в одном поле, независимо от значений других полей. Рассмотрим пример, в котором на номер района наложено ограничение. Этот номер не может находиться вне диапазона от 1 до 9, даже если пользователь этого очень захочет (рис. 3.12).

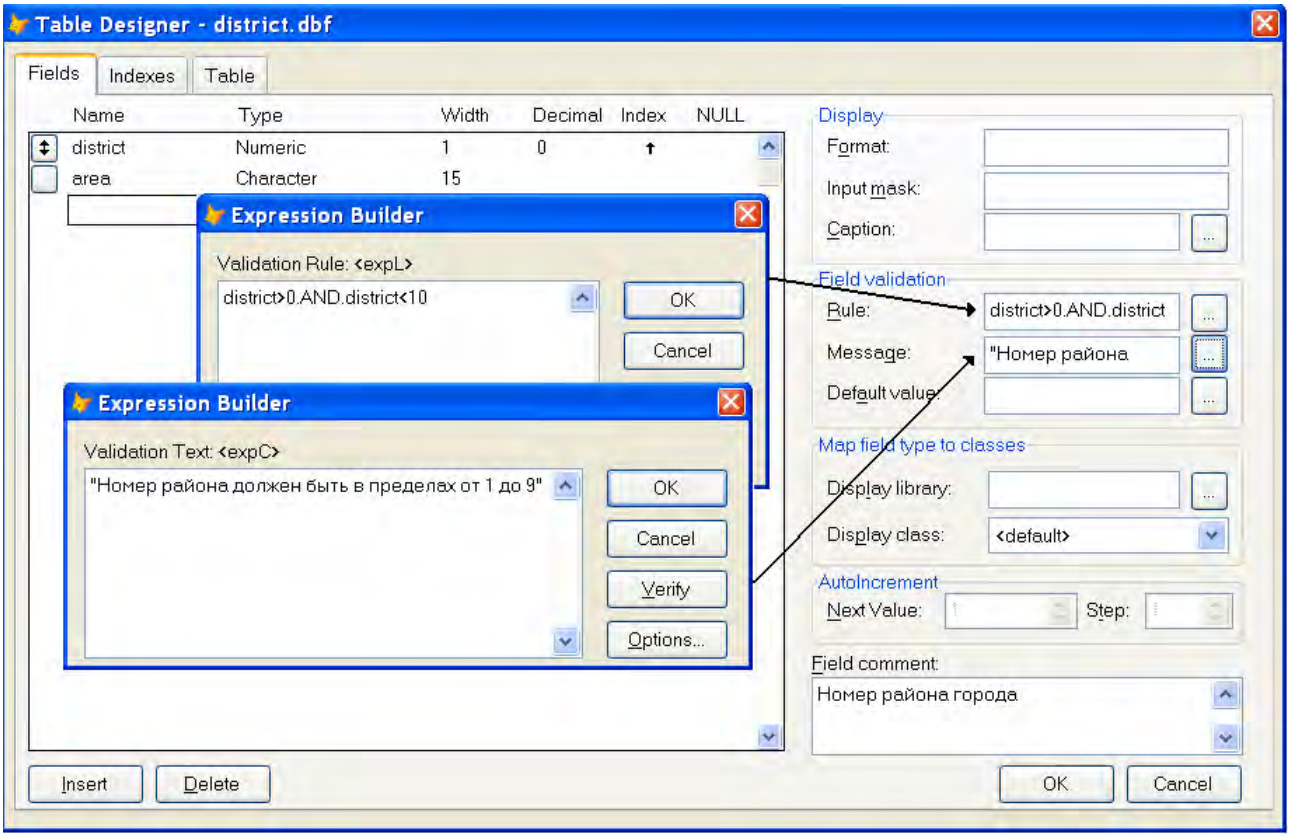


Рис. 3.12. Добавление условия на значение поля ***district*** (номер района)

Чтобы добавить условие на значение поля таблицы ***District*** (районы):

1. Откройте таблицу ***District*** в режиме конструктора. Для этого в окне базы данных ***Real Estate*** (рис. 2.1) щелкните по таблице ***District*** правой кнопкой мыши. Появится меню. В нем пять пунктов. Выберите из них четвертый с названием ***Modify***.

2. Появится окно (рис. 3.13) конструктора таблиц (вкладка ***Fields***).

3. Запустите построитель выражений для ячейки ***Rule***, выбрав пиктограмму . Наберите на клавиатуре: ***District>0 .And. District<10*** и щелкните по кнопке ***OK***.

4. Запустите построитель выражений для ячейки ***Message***, выбрав пиктограмму . Наберите на клавиатуре: ***«Номер района должен быть от 1 до 9»*** и щелкните по кнопке ***OK***. Закройте окно конструктора таблицы и подтвердите сохранение сделанных изменений.

При попытке ввода номера района, который не находится в пределах диапазона 1–9, получим сообщение об ошибке и отказ программного комплекса от записи в таблицу сделанных изменений (рис. 3.15).

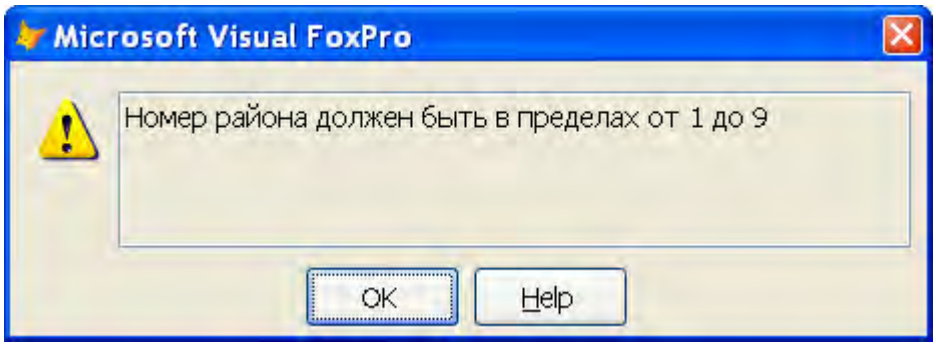


Рис. 3.13. Сообщение при ошибочных действиях оператора

Добавление условия на значение записи. Позволяет сравнить значения нескольких полей сразу. Рассмотрим пример, в котором производится проверка соответствия общей площади квартиры сумме составляющих: жилой, вспомогательной и приведенной площади балкона.

Для того чтобы добавить условие на значение записи:

1. Откройте таблицу ***Flat*** в режиме конструктора. Для этого в окне базы данных ***Real Estate*** (рис. 2.1) щелкните по таблице ***Flat*** правой кнопкой мыши. Появится меню. В нем пять пунктов. Выберите из них четвертый с названием ***Modify***.

2. Появится окно конструктора таблиц (первая вкладка ***Fields***).

3. Перейдите на третью вкладку ***Table*** (рис. 3.14).

4. Запустите построитель выражений для ячейки ***Rule***, выбрав пиктограмму . Наберите на клавиатуре: ***squareflat = dwell+ branch+ balcony*** и щелкните по кнопке ***OK***.

5. Запустите построитель выражений для ячейки ***Message***, выбрав пиктограмму . Наберите на клавиатуре: ***"Общая площадь квартиры не равна сумме составляющих"*** и щелкните по кнопке ***OK***. Закройте окно конструктора таблицы и подтвердите сохранение сделанных изменений.

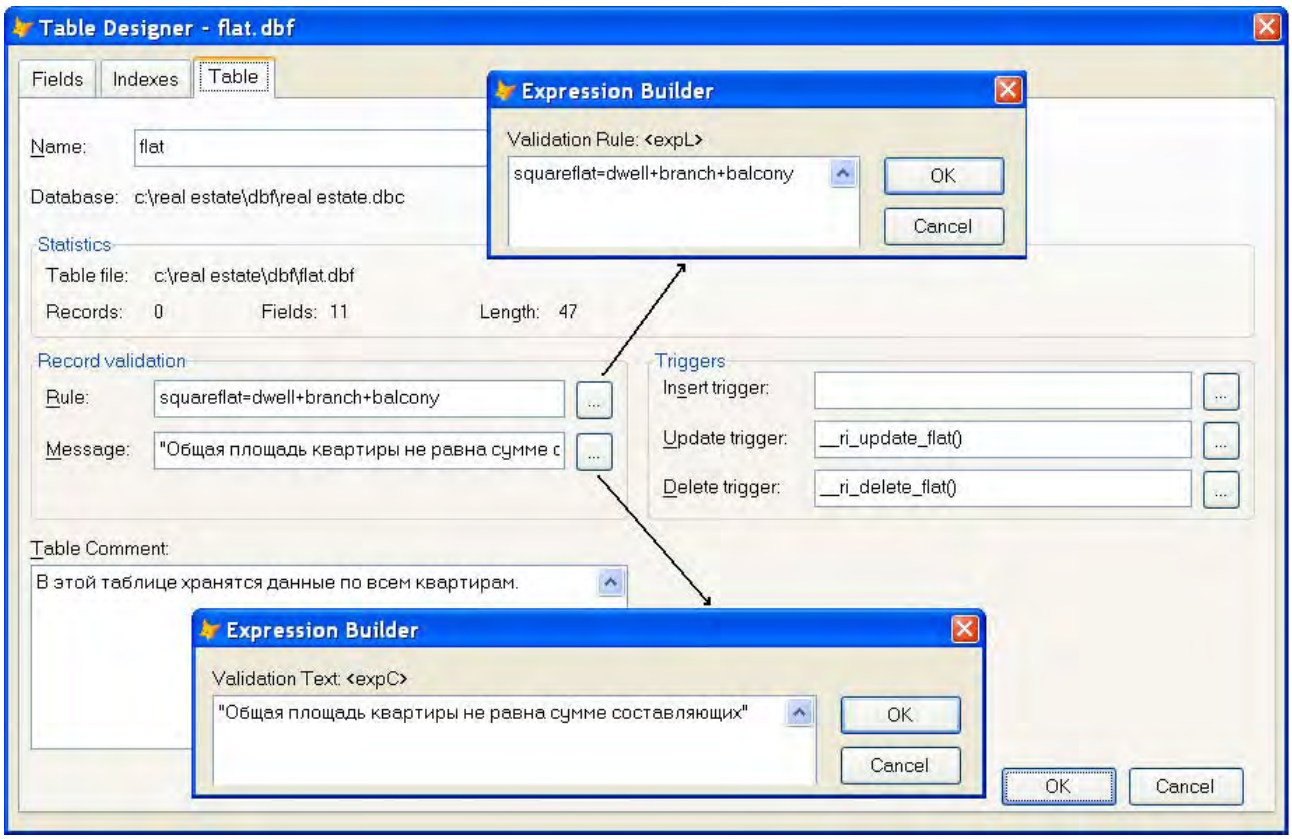


Рис. 3.14. Диалоговое окно «Свойства таблицы»

В случае появления этой ошибки при работе программного комплекса появится сообщение (рис. 3.15).

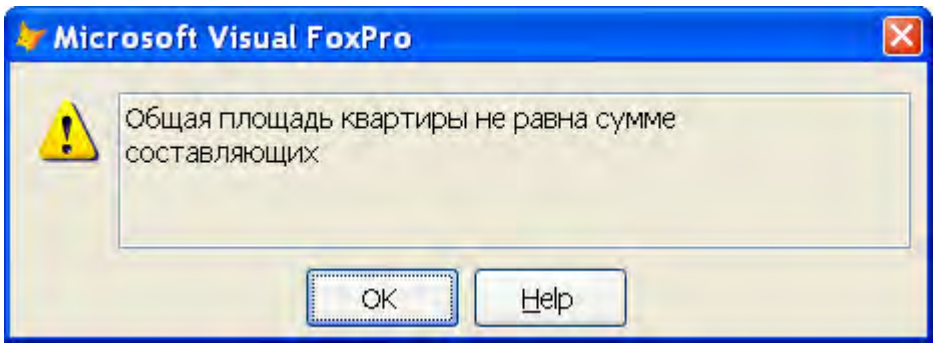


Рис. 3.15. Сообщение об ошибке

3.6. Создание связей между таблицами

Подведем итоги. База данных имеется. Таблицы доведены до третьей нормальной формы и помещены в базу. Первичный ключ есть у каждой таблицы. Индексы созданы. Типы связей между таблицами определены. Настало время создания связей между таблицами непосредственно в базе

данных. Связи между таблицами назначают и просматривают в окне базы данных (рис. 2.1). Открыть его можно следующим образом. В главном меню Visual FoxPro щелкните пункт ***File*** и выберите команду ***Open***. В открывшемся окне в ячейке ***Тип файлов*** выберите пункт ***Database (*.dbc)***. Перейдите в папку ***DBF***. Выберите нашу базу данных ***Real Estate.dbc***. Обязательно поставьте флажок ☑ в ячейке ***Open exclusive***. Если этого не сделать, то вам будут доступны не все опции по работе с базой данных.

Из двух связанных таблиц одна является главной (родительской), а другая подчиненной (дочерней). Для главной таблицы нужен индекс ***Primary*** (в окне базы данных он отмечен значком 🔑 и его имя выделено жирным шрифтом), а для подчиненной – ***Regular***.

Установим связь между таблицами ***District*** (главная) и ***Building*** (подчиненная). Поместите указатель мыши на первичный ключ главной таблицы ***District***. Нажмите левую кнопку мыши и, не отпуская ее, «перетащите» появившийся символ перечеркнутой окружности на обычный индекс ***District*** таблицы ***Building***. Значок перечеркнутой окружности превратится в маленький прямоугольник с надписью внутри. Отпустите левую кнопку мыши. Связь установлена. Имейте ввиду, что попытка проделать эти действия, начиная с подчиненной таблицы ***Building***, закончится неудачей. Значок перечеркнутой окружности никогда не превратится в прямоугольник с надписью.

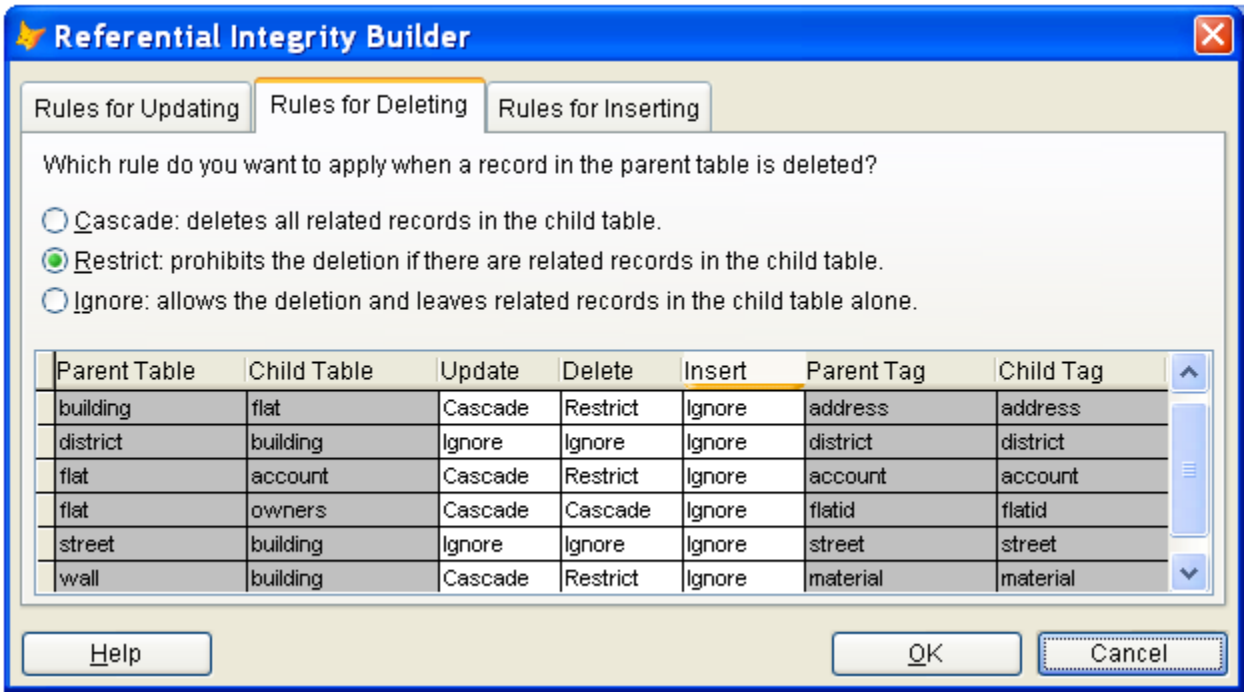
3.7. Обеспечение ссылочной целостности данных

Вернемся к рис. 2.1. Важной особенностью Microsoft Visual FoxPro является автоматическое обеспечение ссылочной целостности данных. Если на связь между таблицами наложены условия ссылочной целостности, то добавление в связанную таблицу записи, для которой нет соответствующих записей в главной таблице, становится невозможным.

Проверка целостности данных может осуществляться и программными средствами. Например, при добавлении в таблицу ***Building*** описания нового здания, вы можете проверить, имеется ли в таблице ***District*** район, в котором расположено это здание. Однако более правильным является определение условия целостности данных на уровне базы данных, так как в этом случае ни одно приложение не может нарушить целостность данных. С базой данных может работать несколько приложений, в том числе и не только ваших.

Несколько дополнительных возможностей. Откройте окно базы данных. В главном меню Visual FoxPro щелкните пункт ***File*** и выберите команду ***Open***. В открывшемся окне в ячейке ***Тип файлов*** выберите пункт ***Database (*.dbc)***. Перейдите в папку ***DBF***. Выберите нашу базу данных ***Real Estate.dbc***. Обязательно поставьте флажок ☑ в ячейке ***Open exclusive***. Если этого не сделать, то вам будут недоступны опции по работе с базой данных,

которые мы сейчас рассмотрим. Щелкните по кнопке ***OK***. Появится окно базы данных. В главном меню Visual FoxPro выберите пункт ***Database***, а в открывшемся подменю пункт ***Edit Referential Integrity***. Появится окно конструктора ссылочной целостности (рис. 3.16).



Parent Table	Child Table	Update	Delete	Insert	Parent Tag	Child Tag
building	flat	Cascade	Restrict	Ignore	address	address
district	building	Ignore	Ignore	Ignore	district	district
flat	account	Cascade	Restrict	Ignore	account	account
flat	owners	Cascade	Cascade	Ignore	flatid	flatid
street	building	Ignore	Ignore	Ignore	street	street
wall	building	Cascade	Restrict	Ignore	material	material

Рис. 3.16. Конструктор ссылочной целостности базы данных

В нижней части конструктора перечислены все связи между таблицами (каждая на отдельной строке). В первых двух столбцах приводятся названия родительской и дочерней таблиц. В следующих трех – ***Update*** (Обновить), ***Delete*** (Удалить) и ***Insert*** (Вставить) – указаны правила соблюдения целостности. В начале работы все эти три столбца содержат элемент ***Ignore*** (Игнорировать). Однако вы можете сами определить правила поведения для каждой связи и выполняемого действия. Наконец, в последних двух столбцах определены родительский и дочерний индексы, участвующие в отношении.

Имейте в виду, что модифицировать можно только содержимое трех центральных столбцов, «отвечающих» за правила ссылочной целостности. Они выделены белым. При выборе любого из этих столбцов (с помощью щелчка) появляется кнопка, на которой изображена направленная вниз стрелка. После щелчка на этой кнопке раскрывается меню, содержащее возможные варианты поведения, смысл которых расшифрован в верхней части окна конструктора.

Для каждого действия (обновления, удаления и вставки) отведена отдельная вкладка, на которой перечислены доступные варианты поведе-

ния. При обновлении ключевого значения в родительской таблице можно применить следующие правила поведения.

- ***Cascade*** (Последовательно выполнить). При выборе этой опции обновляются все «дочерние» записи в соответствии с новым значением ключа в родительской таблице, если у них совпадало старое значение «родительского» ключа.
- ***Restrict*** (Ограничить). Если в дочерней таблице есть связанные записи (т. е. существуют записи с текущим значением родительского ключа), то FoxPro запрещает обновление родительского ключа.
- ***Ignore*** (Игнорировать). При выборе этой опции система прекращает следить за соблюдением правил ссылочной целостности и разрешает обновление родительского ключа независимо от наличия связанных записей в дочерних таблицах.

Как видите, все эти опции объединены в группу переключателей. При щелчке на одном из переключателей изменяется правило целостности для выделенной связи между таблицами в соответствии с выбранной опцией переключателя. Итак, у вас есть два способа выбора правил ссылочной целостности: с помощью группы переключателей в верхней части окна построителя и с помощью меню в нижней его части.

Правила для удаления родительских записей аналогичны правилам, применяемым к операции обновления. Однако правила для вставки записей применяются только с дочерней стороны отношения. В этом случае возможны только два правила.

- ***Restrict****.* Запрещается вставка дочерней записи, если нет родительской записи с таким же значением ключа.
- ***Ignore****.* Не выполняется никаких проверок в целях сохранения ссылочной целостности, т.е. вставка заведомо разрешается.

Определив правила ссылочной целостности для каждой связи между таблицами и действия, щелкните на кнопке ***OK***, чтобы выйти из конструктора. Что касается нашего примера, то, скорее всего, нам стоит разрешить последовательное выполнение обновлений ключа в таблице ***Flat*** после того, как он обновится в таблице ***Building***. С другой стороны, следует запретить удаление записей таблицы ***Building***, если существуют связанные записи в таблице ***Flat***. И наконец, имеет смысл запретить вставку записи в таблицу ***Flat***, если в таблице ***Building*** нет записи с таким же значением ключа.

При щелчке на кнопке ***OK*** отображается диалоговое окно с вопросом о сохранении внесенных изменений и генерируется код ссылочной целостности, после чего построитель завершает свою работу. При этом в базе данных создается набор триггеров и хранимых процедур. Если в базе данных ранее были определены триггеры или хранимые процедуры, то перед их перезаписью создается резервная копия. Она помещается в файл ***Risp.old*** в текущую папку. Хранимые процедуры, созданные ранее для

других целей (например, для соблюдения правил контроля данных), придется вручную скопировать из резервной копии.

После того как построитель ссылочной целостности завершит свою работу, можно открыть окно ***Table Designer*** (рис. 3.14) и выбрать в нем третью вкладку ***Table***, чтобы просмотреть добавленные триггеры. В качестве альтернативного варианта можно просмотреть хранимые процедуры, щелкнув на кнопке ***Edit Stored Procedure*** (Редактировать хранимую процедуру), расположенной на панели инструментов ***Database Designer*** в окне конструктора баз данных. Приведем текст хранимой в базе данных процедуры ***_ri_update_flat()***. Она будет запущена на выполнение автоматически при изменении значения первичного ключа.

```
procedure __RI_UPDATE_flat
** "Referential integrity update trigger for" flat
LOCAL llRetVal
llRetVal = .t.
PRIVATE pcParentDBF,pnParentRec,pcChildDBF,pnChildRec,;
        pcParentID,pcChildID
PRIVATE pcParentExpr,pcChildExpr
STORE "" TO pcParentDBF,pcChildDBF,pcParentID,pcChildID,;
            pcParentExpr,pcChildExpr
STORE 0 TO pnParentRec,pnChildRec
IF _triggerlevel=1
  BEGIN TRANSACTION
  PRIVATE pcRIcursors,pcRIwkareas,pcRIolderror,pnerror,;
  pcOldDele,pcOldExact,pcOldTalk,pcOldCompat,PcOldDBC
  pcOldTalk=SET("TALK")
  SET TALK OFF
  pcOldDele=SET("DELETED")
  pcOldExact=SET("EXACT")
  pcOldCompat=SET("COMPATIBLE")
  SET COMPATIBLE OFF
  SET DELETED ON
  SET EXACT OFF
  pcRIcursors=""
  pcRIwkareas=""
  pcRIolderror=ON("error")
  pnerror=0
  ON ERROR pnerror=rierror(ERROR(),message(),message(1),program())
  IF TYPE('gaErrors(1)')<>"U"
    release gaErrors
  ENDIF
  PUBLIC gaErrors(1,12)
  pcOldDBC=DBC()
  SET DATA TO ("REAL ESTATE")
ENDIF first trigger
LOCAL lcParentID && parent's value to be sought in child
LOCAL lcOldParentID && previous parent id value
LOCAL lcChildWkArea && child work area handle returned by riopen
```

```
LOCAL lcChildID && child's value to be sought in parent
LOCAL lcOldChildID && old child id value
LOCAL lcParentWkArea && parentwork area handle returned by riopen
LOCAL lcStartArea
lcStartArea=select()
llRetVal=.t.
lcParentWkArea=select()
SELECT (lcParentWkArea)
pcParentDBF=dbf()
pnParentRec=recno()
lcOldParentID=OLDVAL("ACCOUNT")
pcParentID=lcOldParentID
pcParentExpr="ACCOUNT"
lcParentID=ACCOUNT
IF lcParentID<>lcOldParentID
  lcChildWkArea=riopen("account")
  IF lcChildWkArea<=0
    IF _triggerlevel=1
      DO riend WITH .F.
    ENDIF at the end of the highest trigger level
    SELECT (lcStartArea)
    RETURN .F.
  ENDIF not able to open the child work area
  pcChildDBF=dbf(lcChildWkArea)
  SELECT (lcChildWkArea)
  SCAN FOR ACCOUNT=lcOldParentID
    pnChildRec=recno()
    pcChildID=ACCOUNT
    pcChildExpr="ACCOUNT"
    IF NOT llRetVal
      EXIT
    ENDIF && not llretval
    llRetVal=riupdate("ACCOUNT",lcParentID,"FLAT")
  ENDSCAN get all of the account records
  =rireuse("account",lcChildWkArea)
  IF NOT llRetVal
    IF _triggerlevel=1
      DO riend WITH llRetVal
    ENDIF at the end of the highest trigger level
    SELECT (lcStartArea)
    RETURN llRetVal
  ENDIF
ENDIF this parent id changed
SELECT (lcParentWkArea)
pcParentDBF=dbf()
pnParentRec=recno()
lcOldParentID=OLDVAL("STR(STREET)+HOUSE+STR(FLAT)")
pcParentID=lcOldParentID
pcParentExpr="STR(STREET)+HOUSE+STR(FLAT)"
lcParentID=STR(STREET)+HOUSE+STR(FLAT)
IF lcParentID<>lcOldParentID
```

```
  lcChildWkArea=riopen("owners")
  IF lcChildWkArea<=0
    IF _triggerlevel=1
      DO riend WITH .F.
    ENDIF at the end of the highest trigger level
    SELECT (lcStartArea)
    RETURN .F.
  ENDIF not able to open the child work area
  pcChildDBF=dbf(lcChildWkArea)
  SELECT (lcChildWkArea)
  SCAN FOR STR(STREET)+HOUSE+STR(FLAT)=lcOldParentID
    pnChildRec=recno()
    pcChildID=STR(STREET)+HOUSE+STR(FLAT)
    pcChildExpr="STR(STREET)+HOUSE+STR(FLAT)"
    llRetVal=riupdate("STREET",VAL(substr(lcParentID,1,10)),;
            "flat")
    IF NOT llRetVal
      EXIT
    ENDIF && not llretval
    llRetVal=riupdate("HOUSE",substr(lcParentID,11,4),"flat")
    IF NOT llRetVal
      EXIT
    ENDIF && not llretval
    llRetVal=riupdate("FLAT",VAL(substr(lcParentID,15,10)),"flat")
    IF NOT llRetVal
      EXIT
    ENDIF && not llretval
  ENDSCAN get all of the owners records
  =rireuse("owners",lcChildWkArea)
  IF NOT llRetVal
    IF _triggerlevel=1
      DO riend WITH llRetVal
    ENDIF at the end of the highest trigger level
    SELECT (lcStartArea)
    RETURN llRetVal
  ENDIF
ENDIF this parent id changed
IF _triggerlevel=1
  do riend with llRetVal
ENDIF at the end of the highest trigger level
SELECT (lcStartArea)
RETURN llRetVal
** "End of Referential integrity Update trigger for" flat
```

Мы очень резко окунулись в код Microsoft Visual FoxPro. Не пугайтесь! Разработчики Microsoft так далеко обогнали нас, прикладников, что даже не смешно. И все-таки со временем вы разберетесь в том, что делает этот код, а сейчас – примите его к сведению. При изменении любой из таблиц, влияющих на состояние ссылочной целостности базы данных, их

индексы (или постоянные отношения) перезапускают построитель ссылочной целостности. Это приводит к пересмотру кода в соответствии с выполненными изменениями.

Как видите, с помощью построителя ***Referential Integrity Builder*** можно быстро и легко добавлять общие правила ссылочной целостности к отношениям между таблицами в базе данных, не особенно беспокоясь о том, как это реализуется.

3.8. Устранение связи «многие ко многим»

В качестве примера рассмотрим функционирование фирмы «Столица» (часть варианта 8). Особенностью работы этой фирмы является посредническая деятельность – стыковка поставщиков товаров и покупателей. Один поставщик связан с несколькими покупателями. Один покупатель, в свою очередь, связан с несколькими поставщиками. По понятным причинам поставщик «не знает» покупателя и наоборот. Вот Вам предпосылка создания связи «многие ко многим» между двумя таблицами: поставщики и покупатели (рис. 3.17).

Поставщики

ИНН поставщика	Название поставщика	Штрих-код	Название товара	Дата выпуска	Цена за 1 ед.	К-во
252206156	«Океан-2»	46052461	Печенье	01.06.04	95-00	200
772654321	«Октябрь»	78690543	Печенье	12.07.04	123-50	450
772654321	«Октябрь»	78695544	Шоколад	15.05.04	43-00	500
772654321	«Октябрь»	78694455	Конфеты	14.06.04	105-00	350
278765433	«Амур пиво»	46052462	Печенье	14.07.04	97-00	600

∞ ∞

Покупатели

ИНН покупателя	Название покупателя	Название товара	Дата покупки	Цена за 1 ед.	К-во	Штрих-код
276546373	«Татьяна»	Печенье	18.07.04	105-00	100	46052462
276546373	«Татьяна»	Печенье	18.07.04	140-00	150	78690543
273894756	«Светлый-3»	Печенье	21.07.04	110-00	50	46052461

Рис. 3.17. Пример связи «многие ко многим»

Как мы уже знаем, реляционная модель не позволяет непосредственно реализовать связь типа «многие ко многим». Кроме этого, штрих-код

товара (рис. 3.18), являясь его однозначной идентификацией, не может быть назначен в качестве первичного ключа. В случае если через какое-то время будет закуплена еще одна партия такого же товара, то в таблице «поставщики» появится строчка, имеющая идентичный штрих-код. Аналогичная ситуация возможна и с покупателем, который приобретет такой же товар из другой партии. И еще одна проблема: один клиент практически никогда не покупает всю партию целиком и перед фирмой встает проблема учета остатка товара.

Рис. 3.18 Штрих-код

Все эти проблемы (реляционной модели и алгоритма) легко решаются добавлением двух промежуточных таблиц: «Поставленные товары» и «Проданные товары». На рис. 3.19 показан фрагмент схемы данных превращения «связи многие ко многим» в несколько связей «один ко многим» с добавлением этих промежуточных таблиц.

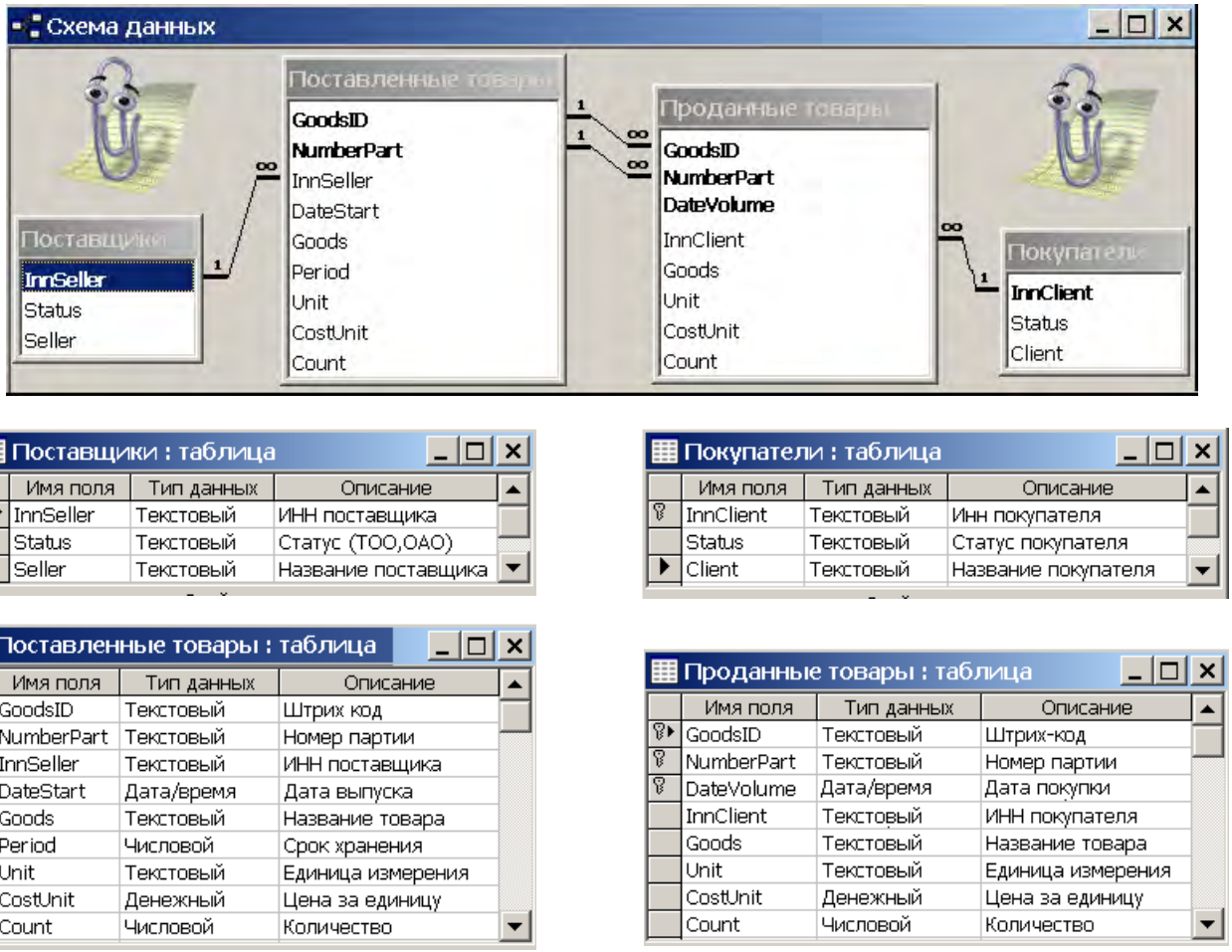


Рис. 3.19. Работающий фрагмент схемы данных

Преобразование связи «многие ко многим» в несколько связей «один ко многим» и «много к одному» можно сделать следующим образом. Вместо того чтобы рассматривать множество поставщиков, поставляющих товары

многим клиентам, будем рассматривать одного поставщика и множество поставляемых им товаров («один ко многим»). Далее будем рассматривать деление товара одной партии на множество частей, подлежащих продаже («один ко многим»). В результате получим множество товаров, приобретаемым одним покупателем («много к одному»).

Появление таблицы «проданные товары» позволяет решить внутреннюю проблему учета остатков товара в фирме «Столица».

Особенностью реализации предложенной схемы является наличие составного первичного ключа в таблице «Поставленные товары» по связке полей «штрих-код товара» плюс «номер партии» (***GoodsID + NumberPart***).

4. РАЗРАБОТКА ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ

4.1. Головной модуль

Головной модуль – это программа, управляющая ходом работы приложения. Включает в себя, как правило, настройку среды Visual FoxPro, описание глобальных переменных, создание главного меню программного комплекса и запуск обработчика событий.

Перед тем как приступить к созданию головного модуля, необходимо спроектировать систему меню. Состав меню невозможно определить без учета конкретных задач, для решения которых предназначено разрабатываемое приложение. Параллельно с определением состава данных необходимо определить те средства, которые получит пользователь при работе с программным комплексом.

Для создания меню можно воспользоваться как конструктором, так написанием кода. Мне больше нравится второй способ. Но вначале нарисуем вид меню на бумаге, обязательно обсудив его структуру с заказчиком. Пока меню не превратилось в строчки кода, внести в него изменения гораздо проще. На рис. 4.1 приведен эскиз главного меню программного комплекса ***Real Estate***, согласованный с заказчиком.

Текст головного модуля, реализующий это меню, имеет вид

```
****************************************** RealEstate.prg *******
* Учебный пример к книге                                        *
* Гурвиц Г.А. "Разработка реального приложения                  *
* с использованием Microsoft Visual FoxPro 9.0"                 *
* Учебное пособие. - Хабаровск: Изд-во ДВГУПС, 2007.            *
****************************************************************
*
* Внесение изменений в системные настройки Visual FoxPro
CLEAR MACROS          && Отмена системных назначений клавиш F1-F12
SET RESOURCE ON       && Сохранять настройки в таблице Foxuser.dbf
SET EXCLUSIVE OFF     && Базы данных доступны всем
SET MULTILOCKS ON     && Блокировка нескольких записей сразу
```

```
SET TALK OFF          && Не отображать результаты выполнения
                      && команд APPEND, AVERAGE,CALCULATE и др.
SET DATE GERMAN       && Тип общепринятой в России даты
SET CENTURY ON        && Столетие полностью
SET DELETED ON        && Помеченные к удалению записи невидимы
SET SAFETY OFF        && Не выдавать запрос на уничтожение
SET STATUS BAR OFF    && Не показывать нижнюю статусную строку
* Подключение первого файла, содержащего процедуры
* FileProc - имя файла
SET PROCEDURE TO FileProc
* Подключение второго и последующих
* SET PROCEDURE TO <имя файла> ADDITIVE
* SET PROCEDURE TO <имя файла> ADDITIVE
SET PATH TO DBF,USER,FORM,BOOK,ICO,REPORT,INFO   && Пути поиска
* Вызов процедуры описания глобальных переменных
DO ADJUSTMENT                  && Находится в файле FileProc
* Состояние памяти
ON KEY LABEL F2 DO DisplayMemory
* Распределение рабочих областей Visual FoxPro
ON KEY LABEL CTRL+F2 DO FORM AreaWork
* Модификация главного окна FOX PRO
_SCREEN.CAPTION=[Учебный пример Real Estate]
_SCREEN.ICON=[Title.ICO]
_SCREEN.MINBUTTON=.T.          && Есть кнопка свертывания
_SCREEN.MAXBUTTON=.T.          && Есть кнопка развертывания
_SCREEN.WINDOWTYPE=1           && Тип окна
ON ERROR DO ERRORHND           && Вызов процедуры обработки ошибок
* Создание папки для временных выборок
* Если папка уже имеется на компьютере, то возникнет ошибка
* с кодом 1961. Она будет перехвачена процедурой ERRORHND,
* которая находится в процедурном файле FileProc
MKDIR C:\WINNT
MKDIR C:\WINNT\TEMP
* В правом верхнем углу экрана появится приглашение к работе:
   *-----------------------------------------------------*
   * Папка для временных файлов C:\WINNT\TEMP имеется!   *
   * Можно работать!                                     *
   *-----------------------------------------------------*
* Определение размеров главного окна программного комплекса
* в зависимости от разрешения дисплея рабочей станции
DO CASE
   CASE SYSMETRIC(1)=1600         && 1600*1280 пикселей
        _SCREEN.HEIGHT=1272
        _SCREEN.WIDTH=1588
   CASE SYSMETRIC(1)=1280         && 1280*1024 пикселей
        _SCREEN.HEIGHT=970
        _SCREEN.WIDTH=1272
   CASE SYSMETRIC(1)=1024         && 1024*768 пикселей
        _SCREEN.HEIGHT=710
        _SCREEN.WIDTH=1016
   CASE SYSMETRIC(1)=800          && 800*600 пикселей
```

```
       _SCREEN.HEIGHT=540
       _SCREEN.WIDTH=795
ENDCASE
_SCREEN.WINDOWSTATE=2    && 2-Развернуть во весь экран 0-назад
_SCREEN.AUTOCENTER=.F.   && Размещение НЕ по центру экрана
_SCREEN.BORDERSTYLE=3    && Обрамление двойная линия
* Цвет фона
_SCREEN.BackColor=RGB(192,192,192)
* Размещение картинки в главном окне программного комплекса
IF FILE('C:\WINNT\TEMP\PICTURE.JPG')
   _SCREEN.PICTURE=[C:\WINNT\TEMP\PICTURE.JPG]
ENDIF
_SCREEN.FONTNAME=[ARIAL CYR]             && Шрифт
_SCREEN.FONTSIZE=9                       && Размер шрифта
_SCREEN.ICON=[HOUSE.ICO]                 && Иконка
* Подтверждение выхода из программы
* Запуск процедуры REALQUIT при закрытии окна Visual FoxPro
ON SHUTDOWN DO REALQUIT
SET SYSMENU TO           && Убрать системное меню
DO FORM LOGIN            && Запуск формы контроля доступа
IF SuperVisor=.F.
   * Идентификация не выполнена
   DO STOP               && Завершение работы
ENDIF
SET SYSMENU ON           && Работа со строкой главного меню

* Замена меню Visual FoxPro на собственное
* Определение заголовков создаваемого меню
* PROMPT "\<Текст" - текст, появляющийся в строке меню
DEFINE PAD POINT1 OF _MSYSMENU PROMPT "\<Поддержка"
DEFINE PAD POINT2 OF _MSYSMENU PROMPT "\<Информация"
DEFINE PAD POINT3 OF _MSYSMENU PROMPT "\<Здания"
DEFINE PAD POINT4 OF _MSYSMENU PROMPT "\<Справочники"
DEFINE PAD POINT5 OF _MSYSMENU PROMPT "\<Выход"
* При переходе в пункт 1 меню показать POPUP-меню "Поддержка"
ON PAD POINT1 OF _MSYSMENU ACTIVATE POPUP SUPPORT
* При переходе в пункт 2 меню показать POPUP-меню "Информация"
ON PAD POINT2 OF _MSYSMENU ACTIVATE POPUP INFORMATIONS
* При переходе в пункт 3 меню показать POPUP-меню "Здания"
ON PAD POINT3 OF _MSYSMENU ACTIVATE POPUP BUILDING
* При переходе в пункт 4 меню показать POPUP-меню "Справочники"
ON PAD POINT4 OF _MSYSMENU ACTIVATE POPUP DICTIONARY
* При выборе 5-го пункта меню запустить процедуру STOP
ON SELECTION PAD POINT5 OF _MSYSMENU DO STOP

* Описание POPUP-меню "Поддержка"
DEFINE POPUP SUPPORT MARGIN FONT [Arial Cyr],11
DEFINE BAR 1 OF SUPPORT PROMPT "\<Смена картинки";
                                  SKIP FOR ChangePicture=.F.
DEFINE BAR 2 OF SUPPORT PROMPT "Смена \<пароля";
                                  SKIP FOR ChangePassword=.F.
```

```
DEFINE BAR 3 OF SUPPORT PROMPT "\<Задержка при поиске"
DEFINE BAR 4 OF SUPPORT PROMPT "\<Удаленные записи";
                                        SKIP FOR SetDeleted=.F.
DEFINE BAR 5 OF SUPPORT PROMPT "Календарь \<ежедневник"
DEFINE BAR 6 OF SUPPORT PROMPT "\<Калькулятор"
DEFINE BAR 7 OF SUPPORT PROMPT "\<Выход"
* При выборе 1-го пункта меню запустить форму ChangPic
ON SELECTION BAR 1 OF SUPPORT DO FORM ChangPic
* При выборе 2-го пункта меню запустить форму PassWord
ON SELECTION BAR 2 OF SUPPORT DO FORM PassWord
* При выборе 3-го пункта меню запустить форму Adjust
ON SELECTION BAR 3 OF SUPPORT DO FORM Adjust
* При переходе в пункт 4 меню показать POPUP-меню SETDEL
ON BAR 4 OF SUPPORT ACTIVATE POPUP SETDEL
* При выборе 5-го пункта меню запустить процедуру
ON SELECTION BAR 5 OF SUPPORT DO CALENDAR
ON SELECTION BAR 6 OF SUPPORT DO CALCULATOR
ON SELECTION BAR 7 OF SUPPORT DO STOP

* Описание POPUP-меню "Удаленные записи"
DEFINE POPUP SETDEL MARGIN FONT [Arial Cyr],11
DEFINE BAR 1 OF SETDEL PROMPT "\<Удаленные записи видимы"
DEFINE BAR 2 OF SETDEL PROMPT "Удаленные \<записи НЕвидимы"
ON SELECTION BAR 1 OF SETDEL SET DELETE OFF
ON SELECTION BAR 2 OF SETDEL SET DELETE ON

* Описание POPUP-меню "Информация"
DEFINE POPUP INFORMATIONS MARGIN FONT [Arial Cyr],11
DEFINE BAR 1 OF INFORMATIONS PROMPT "Об \<авторе"
DEFINE BAR 2 OF INFORMATIONS PROMPT "\<О компьютере"
DEFINE BAR 3 OF INFORMATIONS PROMPT "О \<заполнении таблиц";
                                        SKIP FOR CountRecords=.F.
DEFINE BAR 4 OF INFORMATIONS PROMPT "О \<правах доступа";
                                        SKIP FOR RightAccess=.F.
DEFINE BAR 5 OF INFORMATIONS PROMPT "\<Совершенно секретно"
ON SELECTION BAR 1 OF INFORMATIONS DO FORM Author
ON SELECTION BAR 2 OF INFORMATIONS DO INFORM
ON SELECTION BAR 3 OF INFORMATIONS DO FORM Fill
ON SELECTION BAR 4 OF INFORMATIONS DO FORM Access
ON SELECTION BAR 5 OF INFORMATIONS DO ABSOLUTELY

* Описание POPUP-меню Здания
DEFINE POPUP BUILDING MARGIN FONT [Arial Cyr],11
DEFINE BAR 1 OF BUILDING PROMPT "\<Поиск здания";
                                        SKIP FOR SeekBuilding=.F.
DEFINE BAR 2 OF BUILDING PROMPT "\<Добавить здание";
                                        SKIP FOR EddBuilding=.F.
ON SELECTION BAR 1 OF BUILDING DO FORM Search
ON SELECTION BAR 2 OF BUILDING DO FORM AddBuild

* Справочники
```

```
DEFINE POPUP DICTIONARY MARGIN FONT [Arial Cyr],11
DEFINE BAR 1 OF DICTIONARY PROMPT "\<Улицы города";
                                        SKIP FOR StreetTown=.F.
DEFINE BAR 2 OF DICTIONARY PROMPT "\<Районы города";
                                        SKIP FOR DistrictTown=.F.
DEFINE BAR 3 OF DICTIONARY PROMPT "\<Материал стен";
                                        SKIP FOR MaterialWall=.F.
DEFINE BAR 4 OF DICTIONARY PROMPT "\<Работники";
                                        SKIP FOR Staff=.F.
ON SELECTION BAR 1 OF DICTIONARY DO FORM Street
ON SELECTION BAR 2 OF DICTIONARY DO FORM District
ON SELECTION BAR 3 OF DICTIONARY DO FORM Wall
ON SELECTION BAR 4 OF DICTIONARY DO FORM Employee

* Запуск обработчика событий Visual FoxPro
READ EVENTS
DO STOP             && Процедура STOP находится в файле FileProc
```

Рис. 4.1. Главное меню программного комплекса

Запустить на выполнение головной модуль можно из командного окна ***Command***, набрав команду ***Do RealEstate***. А если вы находитесь в редакторе текстов Visual FoxPro, то щелкните по пиктограмме ! с восклицательным знаком, которую легко найти в главном меню СУБД.

4.2. Обеспечение информационной безопасности приложения

Любое современное предприятие не может сегодня успешно функционировать без создания надежной системы защиты своей информации, включающей не только организационно-нормативные меры, но и технические программно-аппаратные средства контроля. Все решения по защите от несанкционированного доступа и установление личной ответственности работников принимаются руководителем предприятия, так как именно он по действующему ныне законодательству несет в первую очередь ответственность за утрату конфиденциальной информации. Не сомневайтесь – ответственность за доступ к информационной системе предприятия будет возложена на Вас.

При защите информационной системы от несанкционированного доступа могут быть использованы два принципа управления доступом к защищаемым ресурсам: дискреционный и мандатный.

Дискреционный принцип управления доступом. Каждому зарегистрированному пользователю устанавливаются права доступа к объектам системы, которые прописываются в правилах разграничения доступа (ПРД). Эти правила должен утвердить руководитель предприятия. Обязательно побеспокойтесь об этом. Данный вариант управления доступом позволяет для любого пользователя системы создать изолированную программную среду, т. е. ограничить его возможности по запуску программ, указав в качестве разрешенных к запуску только те программы, которые действительно необходимы для выполнения пользователем своих служебных обязанностей. Таким образом, программы, не входящие в этот список, пользователь запустить не сможет.

Мандатный принцип управления доступом. Принцип управления доступом к ресурсам, основанный на сопоставлении уровня конфиденциальности, присваиваемого каждому ресурсу, и полномочиях конкретного зарегистрированного пользователя по доступу к ресурсам информационной системы с заданным уровнем конфиденциальности.

Для организации мандатного управления доступом, для каждого пользователя системы устанавливается некоторый уровень допуска к конфиденциальной информации (уровень отдела, уровень департамента, уровень дирекции и т. д.), а каждому ресурсу присваивается так называемая метка конфиденциальности или код доступа. При этом разграничение доступа к информации осуществляется путем сравнения уровня допуска пользователя и метки конфиденциальности ресурса и принятии решения о предоставлении или не предоставлении доступа.

В рассматриваемом нами примере ***Real Estate*** реализован дискреционный принцип управления доступом (формы ***Login*** и ***Employee***). Внимательно изучив работу форм, вы без больших переделок сможете реализовать мандатный принцип управления доступом. Просто уберите доступ

любого пользователя (даже администратора) к объектам ***Check Box***, (вторая страница формы ***Employee***).

Наряду с контролем доступа система защиты программного комплекса должна содержать подсистему учета сделанных пользователем изменений. Один из способов реализации этой подсистемы вы найдете в главе с описанием работы формы ***Employee***.

4.2.1. Форма Login – контроль доступа к приложению

Научимся делать доступными пользователю только те пункты меню и кнопки, с которыми ему положено работать в соответствии с правилами разграничения доступа (ПРД), существующими на предприятии. Для этих целей создадим форму ***Login***. Эта форма запускается на выполнение до активации меню программного комплекса и либо вообще запрещает регистрацию работника, если его данных (фамилия и пароль) нет в базе, либо ограничивает доступ в соответствии с ПРД.

Создадим форму ***Login*** в режиме конструктора форм. Microsoft Visual FoxPro имеет в своем арсенале еще одно средство для создания формы за несколько минут – Мастер форм. Однако позволю себе порекомендовать вам не работать с ним. Это инструмент для непрофессионалов. Времени потратите немного, но и хорошего результата не получите!

Перед Вами самый распространенный вид формы (рис. 4.2), созданной в Microsoft Visual FoxPro. Обратите внимание на то, что до завершения процесса регистрации главное меню программного комплекса отсутствует. Вместо него – светлая полоса над картинкой. Это сделали две строки из ***RealEstate.prg***:

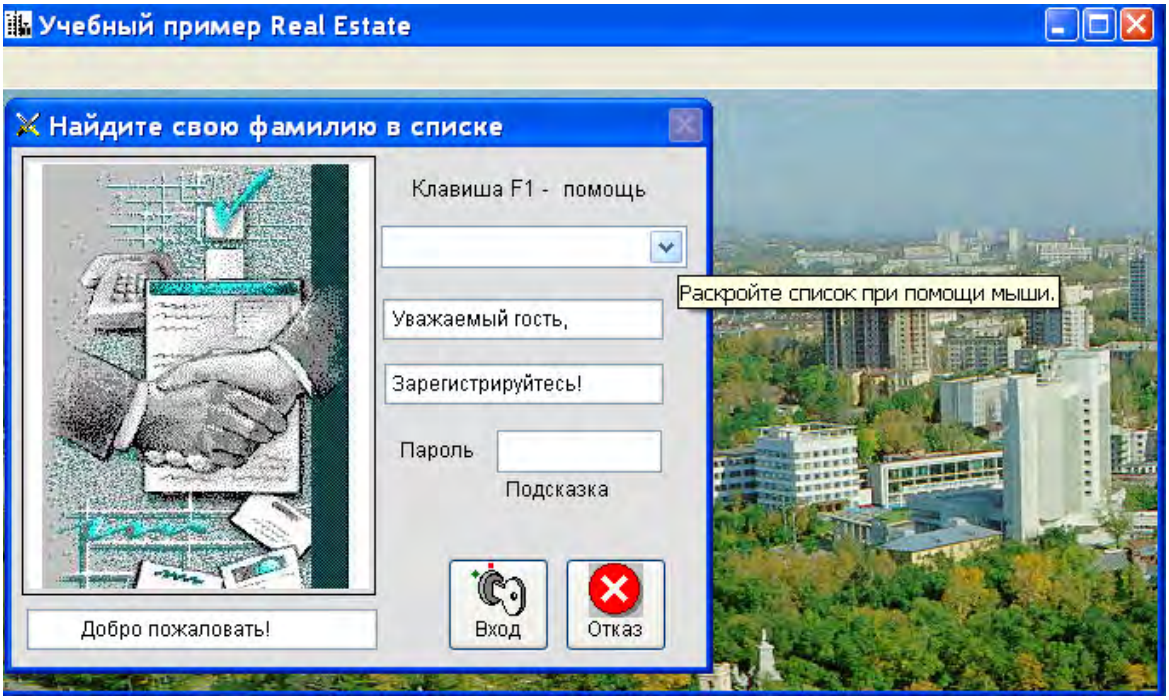


Рис. 4.2. Форма контроля доступа к программного комплексу

```
SET SYSMENU TO              && Убрать системное меню
DO FORM LOGIN               && Запуск формы контроля доступа
```

Для создания формы в главном меню Visual FoxPro щелкните пункт ***File*** и выберите команду ***New***. В открывшемся окне щелкните радиокнопку ***Form*** и нажмите кнопку ***New file.*** На экране дисплея появится окно ***Form Designer***. Это окно конструктора форм. В нем наша первая форма с именем ***Form1***. Справа увидите еще одно окно – окно ***Propertis*** (Свойства). В окне свойств несколько десятков строчек. Все содержат значения по умолчанию. В нашем случае следует изменить всего несколько из них. На рис. 4.3 показаны только измененные свойства. Остальные удалены для наглядности. Если значение свойства изменялось, то его система выделяет жирным цветом.

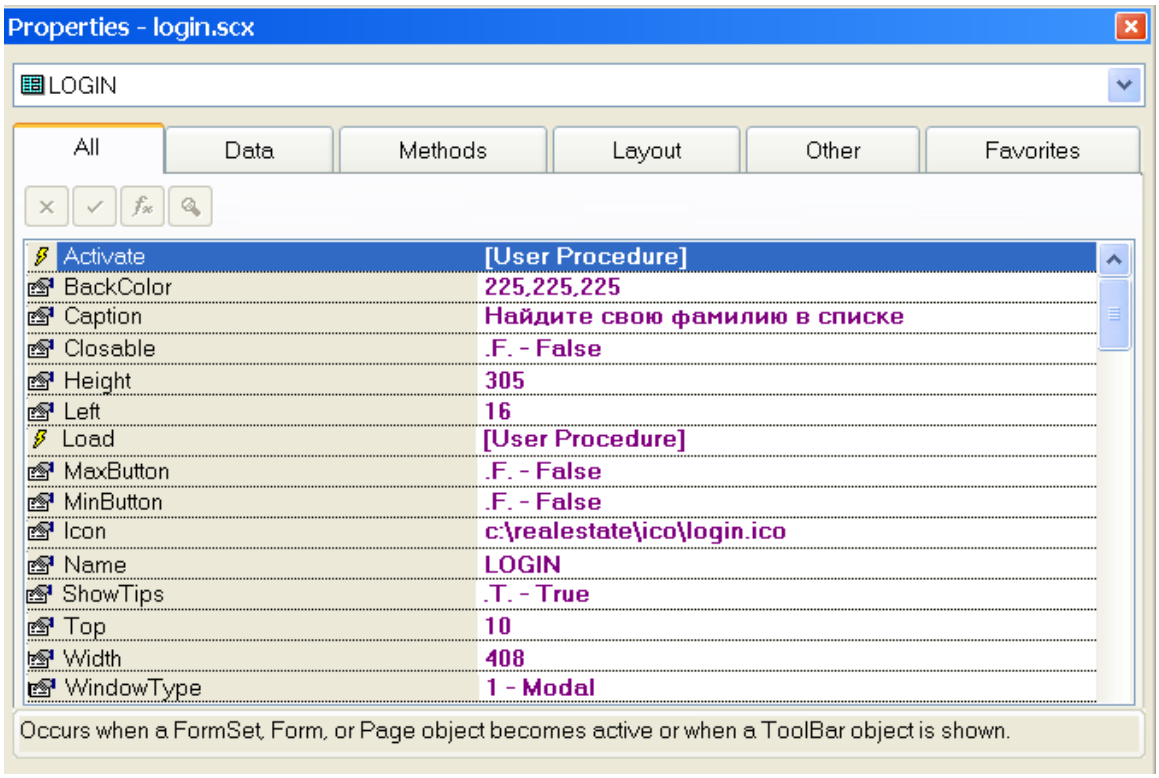


Рис. 4.3. Окно свойств формы ***Login***

Значком молнии отмечены события. Для двух из них ***Activate*** и ***Load*** написан текст (***User procedure***), который запускается на выполнение при наступлении события. ***Activate*** – появление формы на экране. Текст события ***Activate*** имеет вид

```
* Создать переменные памяти с пустыми значениями
SCATTER MEMVAR BLANK
* Дать некоторым из них начальные значения
M.LastName=  []
M.Post=      [         Добро пожаловать!]
M.FirstName= [Уважаемый гость,]
M.SecondName=[Зарегистрируйтесь!]
```

```
PAROL=       []
* Определение полного пути к картинке с приглашением
SET EXACT ON
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
   RealDirectory=[\]
ENDIF
SET EXACT OFF
* Размещение картинки-приглашения в объекте IMAGE1
THISFORM.IMAGE1.PICTURE=DISK+RealDirectory+[PHOTO\Welcome.jpg]
* Перерисовать форму
THISFORM.REFRESH
* Запуск обработчика событий Visual FoxPro
READ EVENTS
```

Текст события ***Load*** (загрузка формы) имеет вид:

```
* Установить в таблице Пользователей отображение фамилий
* по алфавиту
SELECT USER
SET ORDER TO TAG LastName
```

Форма ***Login*** работает с таблицей ***User***, которая не включена в базу данных ***Real Estate*** и является в терминах Visual FoxPro свободной таблицей. Это сделано из соображений безопасности. Права доступа к ней устанавливаются средствами операционной системы отдельно для каждого пользователя локальной вычислительной сети.

Добавим таблицу ***User*** в окружение формы. Сделайте щелчок правой кнопки мыши в любом месте окна ***Form Designer***. Появится меню. Выберите в нем третий пункт ***Data Environment***. Еще один щелчок правой кнопкой, но уже в появившемся окне ***Data Environment*** активизирует очередное меню. Выберите в нем первый пункт ***Add***. Появится окно ***Open***. Найдите в нем таблицу ***User***. Она находится в одноименной папке ***User***. Состав полей приведен в табл. 4.1.

Таблица 4.1

Состав таблицы пользователей

№	Поле	Тип	Размер	Описание
1	LastName	Character	15	Фамилия пользователя
2	FirstName	Character	12	Имя
3	SecondName	Character	10	Отчество
4	Post	Character	25	Занимаемая должность

Окончание табл. 4.1

№	Поле	Тип	Размер	Описание
5	PassWord	Character	10	Зашифрованное значение пароля
6	File	Character	8	Имя файла с фотографией
7	Access01	Logical	1	Настройка картинки в главном окне
8	Access02	Logical	1	Возможность смены своего пароля
9	Access03	Logical	1	Показ удаленных записей
10	Access04	Logical	1	Просмотр объема базы данных
11	Access05	Logical	1	Просмотр своих прав доступа
12	Access06	Logical	1	Возможность поиска зданий
13	Access07	Logical	1	Возможность добавления зданий
14	Access08	Logical	1	Возможность работы с квартирами
15	Access09	Logical	1	Работа с лицевыми счетами
16	Access10	Logical	1	Возможность работы с отчетами
17	Access11	Logical	1	Работа с адресным планом
18	Access12	Logical	1	Работа со списком районов
19	Access13	Logical	1	Доступ к материалу стен зданий
20	Access14	Logical	1	Установка прав доступа всем работникам
21	Inspector	Character	15	Фамилия предоставившего доступ
22	Date_up	Date	8	Дата последней корректировки
23	Time_up	Character	10	Время последней корректировки
24	Range	Numeric	1	Типовой код доступа

После добавления таблицы в окружение данных формы нам не нужно беспокоиться о своевременном ее открытии и закрытии. Все теперь Visual FoxPro сделает сам.

Создадим текстовое поле для отображения имени работника (рис. 4.4). Это однострочное текстовое поле, присоединенное к полю ***FirstName*** таблицы ***User***. Порядок действий следующий. Если на экране дисплея отсутствует окно списка полей таблицы ***User***, откройте окружение данных (***Data Environment***). В списке полей выделите поле ***FirstName*** (рис. 4.4). Нажмите левую кнопку мыши и «перетащите» выделенный элемент на форму. В активной области формы указатель мыши превратится в символ текстового поля. Расположение символа поля указывает верхний левый угол его метки. «Перетащите» текстовое поле в нужное место формы. У вас есть возможность изменить размеры как элемента, так и метки при помощи клавиатуры. Выделите нужный объект и добейтесь необходимых резуль-

татов. При помощи клавиатуры такие действия выполняются гораздо точнее, чем при использовании мыши. Используйте для этого сочетание клавиш ***Shift + стрелка*** (вверх, вниз, вправо и влево).

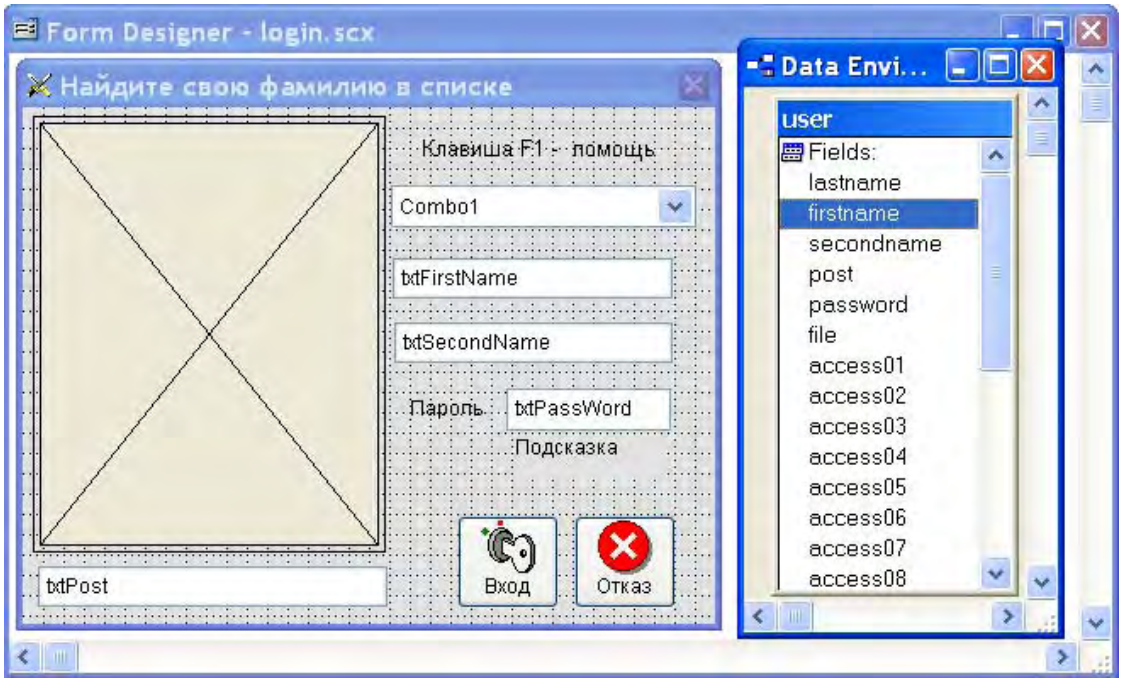


Рис. 4.4. Форма ***Login*** в конструкторе

Для выбора фамилии работника нам понадобится ***Combo Box*** (поле со списком). Найдите его на панели ***Form Controls*** (элементы управления формы). Она показана на рис. 4.5. Если панель отсутствует на экране –

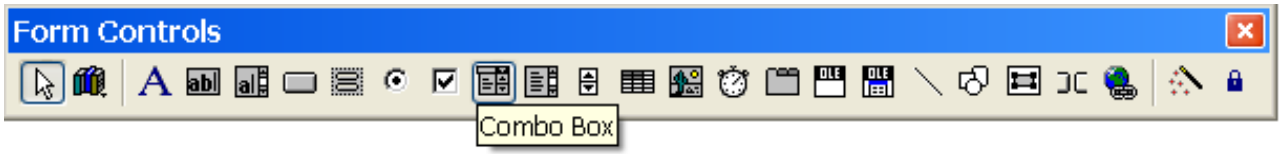


Рис. 4.5. Панель управления ***Form Controls***

выберите в главном меню Visual FoxPro пункт ***View***, а в открывшемся подменю пункт ***Toolbars***. Откроется окно ***Toolbars***. Сделайте отметку напротив названия панели – ***Form Controls*** и щелкните кнопку ***OK***.

Выберите на панели значок ***Combo Box***, а в нужном месте активной области формы при помощи левой кнопки мыши отведите место для этого объекта. Теперь можно запустить построитель. Он ускорит нашу работу. Несколько окон, около десятка ответов и мы у цели. Для запуска построителя щелкните по созданному на форме новому объекту правой кнопкой мыши. Появится подменю. Выберите в нем пятый пункт ***Builder***.

На рис. 4.6 показано окно свойств объекта ***Combo1***. Значения этих свойств – результат работы построителя. Впрочем, построитель можно и не запускать. Выполните назначения самостоятельно – результат тот же.

На рисунке показаны только измененные свойства. Остальные удалены для наглядности.

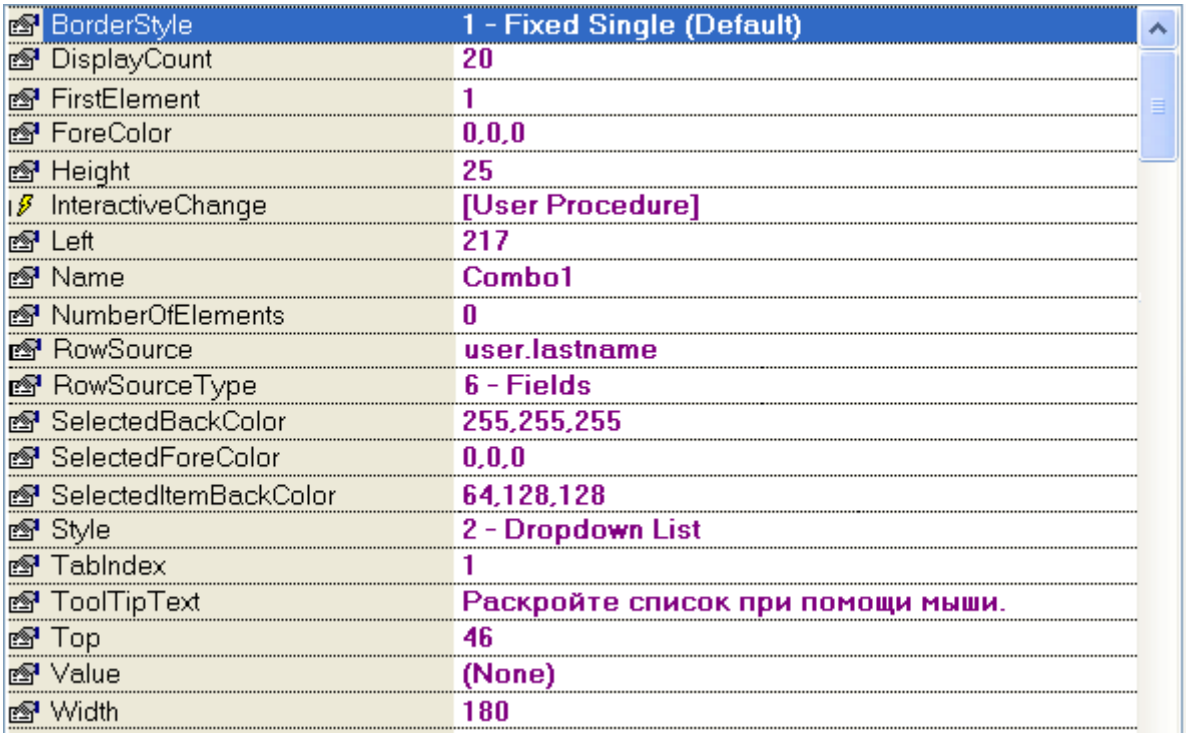


Рис. 4.6. Окно свойств поля со списком ***Combo1***

Если пользователь изменит выбранное в поле значение, то наступит событие ***InteractiveChange***. Немедленно будет запущена процедура (***User Procedure***). Текст ее приведен ниже:

```
* Определение местоположения фотографий
SET EXACT ON
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
   RealDirectory=[\]
ENDIF
SET EXACT OFF
* Есть фотография - значение переменной FileName
FileName=DISK+RealDirectory+[PHOTO\]+ALLTRIM(User.File)+[.jpg]
* Нет фотографии - значение переменной NoFileName
NoFileName=DISK+RealDirectory+[PHOTO\]+[NoFoto.jpg]
IF FILE(FileName)
   * Если фотография имеется в папке PHOTO
   * Разместить ее на месте картинки IMAGE1
   THISFORM.IMAGE1.PICTURE=FileName
ELSE
   * Если нет - взять картинку NoFoto.jpg
   THISFORM.IMAGE1.PICTURE=NoFileName
ENDIF
```

```
SCATTER MEMVAR
* Сохраняем фамилию в глобальной переменной на время сеанса
FAMILY=M.LastName
* Расшифровка пароля и занесение его как подсказку
* в демонстрационных целях под полем пароля
THISFORM.LABEL3.CAPTION=UnKod(M.Password)
THISFORM.REFRESH
```

Пароль пользователя хранится в зашифрованном виде в поле ***Pass-Word*** таблицы ***User***. Для его зашифровки используется процедура-функция ***CrKod***, а для расшифровки ***UnKod***. Находятся они в процедурном файле ***FileProc***. Текст файла приведен в разд. 7.

Последний этап – добавление в форму двух кнопок ***Вход*** и ***Отказ***. Найдите на панели ***Form Controls*** (элементы управления формы) (рис. 4.5) значок кнопки ***Command Button***. В нужном месте активной области формы при помощи левой кнопки мыши отведите место для этого объекта. В окне свойств укажите ***Picture*** (расположение файла с иконкой), высоту ***Height***, ширину ***Width*** (точные размеры в пикселах) и ***Caption*** (название кнопки). Понадобится текст для обработки события ***Click*** (щелчок по кнопке). Он приведен ниже:

```
*- Кнопка Вход
SCATTER MEMVAR   && Создание переменных памяти
Parol=ALLTRIM(Parol)
IF LEN(ALLTRIM(M.PassWord))=0
   * Если в таблице USER был стерт пароль (Например через ODBC)
   SuperVisor=.F.   && Идентификация не выполнена
   =MESSAGEBOX('Пароль в таблице идентификации '+;
               'отсутствует. Операция сравнения паролей '+;
               'не может быть выполнена. Обратитесь к '+;
               'администратору! ',48,' Внимание!')
ELSE
   IF LEN(PAROL)=0
      =MESSAGEBOX('Вы забыли ввести пароль.',48,' Внимание!')
      * Установка курсора в поле TEXT5 и возврат в форму
      THISFORM.TEXT5.SetFocus
      RETURN
   ENDIF
   Parol=CrKod(Parol)            && Зашифровка введенного пароля
   SET EXACT ON                  && Точное соответствие
   IF Parol=  M.PassWord         && Пароль
      SuperVisor=.T.             && Идентификация выполнена
      FAMILY= M.LastName         && Фамилия работника
      * Права доступа к пунктам меню и кнопкам форм
      ChangePicture=  M.Access01 && Смена картинки главного окна
      ChangePassword= M.Access02    && Смена пароля
      SetDeleted=     M.Access03    && Удаленные записи
      CountRecords=   M.Access04    && Заполнение таблиц
```

```
      RightAccess=    M.Access05    && Просмотр прав доступа
      SeekBuilding=   M.Access06    && Поиск зданий по параметрам
      AddBuilding=    M.Access07    && Занесение нового здания
      WorkFlats=      M.Access08    && Обработка квартир
      AccountWork=    M.Access09    && Работа с лицевым счетом
      WordExcel=      M.Access10    && Работа с внешними отчетами
      StreetTown=     M.Access11    && Работа с улицами
      DistrictTown=   M.Access12    && Работа с районами
      MaterialWall=   M.Access13    && Материал стен зданий
      Staff=          M.Access14    && Работа с пользователями
      =MESSAGEBOX('Инспектор '+ALLTRIM(FAMILY)+;
           '!  Доступ разрешен.',48,'Результат идентификации')
      * Проверка срока действия пароля (100 - число дней)
      IF DATE()>M.Date_Up+100
         =MESSAGEBOX(' Срок действия пароля истек! '+;
                  'Его необходимо сменить! ',48,' Внимание!')
         DO FORM ChangPas     && Запуск формы смены пароля
      ENDIF
   ELSE
      SuperVisor=.F.   && Идентификация не выполнена
      =MESSAGEBOX('Доступ запрещен.',;
                  48,' Результат идентификации')
   ENDIF
ENDIF
THISFORM.Release   && Закрыть форму LOGIN
CLEAR EVENTS       && Остановить обработчик событий Visual FoxPro
*- Конец процедуры

*- Кнопка Отказ
SuperVisor=.F.   && Идентификация не выполнена
THISFORM.Release && Закрыть форму LOGIN
CLEAR EVENTS     && Остановить обработчик событий Visual FoxPro
*- Конец процедуры
```

4.2.2. Форма Access – просмотр прав доступа

При создании форм назначения прав доступа (***Employee***) и просмотра прав доступа (***Access***) воспользуемся такой возможностью, предоставляемой Visual FoxPro, как работа с классами. Поэтому сначала немного об объектно-ориентированном программировании (ООП).

Вспомним его основные термины. Объектно-ориентированное программирование можно сравнить с поваренной книгой, содержащей список рецептов. В терминах ООП эти рецепты названы классами, а сама поваренная книга – библиотекой классов. По рецептам (т. е. классам) можно готовить блюда, которые называются объектами или экземплярами объектов.

В поваренной книге есть раздел «супы». В этом разделе описаны рецепты десятков супов: овощных, грибных, рыбных и т. д. Рецепты какого-то одного раздела названы подклассами. Каждый рецепт в поваренной книге

содержит список необходимых продуктов и список инструкций по приготовлению блюда. Продукты известны как свойства класса, а инструкции – как методы.

По этим «рецептам» в «поваренной книге» можно начать «готовить» свои программы. Однако объектно-ориентированное программирование предоставляет одну дополнительную возможность, которая может только сниться тем, кто использует поваренные книги. Каждый раз, когда вы вносите изменения в рецепт (класс), все приготовленные по нему блюда (объекты) немедленно тоже изменяются. Это явление называется наследованием. Настоящая мощь ООП заключается в возможности повторно использовать различные компоненты. Вместо разработки отдельной программы по каждому отдельному вопросу разработчики могут сосредоточить усилия на создании набора повторно используемых объектов, составляющих основу прикладной программы.

Создадим свой первый класс – визуальный класс с именем ***Worker***. Он будет использоваться при создании как минимум двух форм, что должно несколько облегчить наши труды по разработке приложения. Для его создания можно использовать конструктор форм. Создание нового класса с помощью конструктора форм сводится к созданию новой формы или выбору необходимых деталей уже существующей и последующему выбору пункта ***File*** и ***Save As Class*** главного меню Visual FoxPro. При этом открывается окно диалога ***Save Class***, запрашивающее дополнительную информацию о сохраняемом классе. Окно ***Save Class*** позволяет выбрать сохраняемую информацию. Это может быть набор выделенных объектов, вся форма или весь набор форм. После определения сохраняемой информации нужно дополнительно ввести имя, под которым надо сохранить класс, имя библиотеки классов, в которую следует записать определение класса (файл библиотеки имеет расширение ***vcx***) и описание записываемого класса.

Но мы воспользуемся конструктором визуальных классов. В главном меню Visual FoxPro ***File*** выберите ***New***, выделите ***Class*** и выберите ***New File***. Появится окно (рис. 4.7.) В диалоговом окне ***New Class*** задайте имя класса ***Worker***, укажите его базовый класс ***Container*** и имя библиотеки ***Worker.vcx*** , куда он будет записан. Эта библиотека может содержать несколько классов. Ограничимся пока одним. Сделаем щелчок по кнопке ***OK***. Класс создан.

Для выбора класса сначала следует выбрать библиотеку, а затем выделить модифицируемый класс и нажать на него два раза мышью или нажать на кнопку **Open.** После того как класс выбран, активизируется окно конструктора классов ***Class Designer***, которое позволяет изменять свойства и методы класса (рис. 4.8).

Рис. 4.7. Создание нового визуального класса

Конструктор классов кроме этого позволяет разработчику определять пиктограммы и другие элементы описания, которые будут появляться при использовании класса в конструкторе форм.

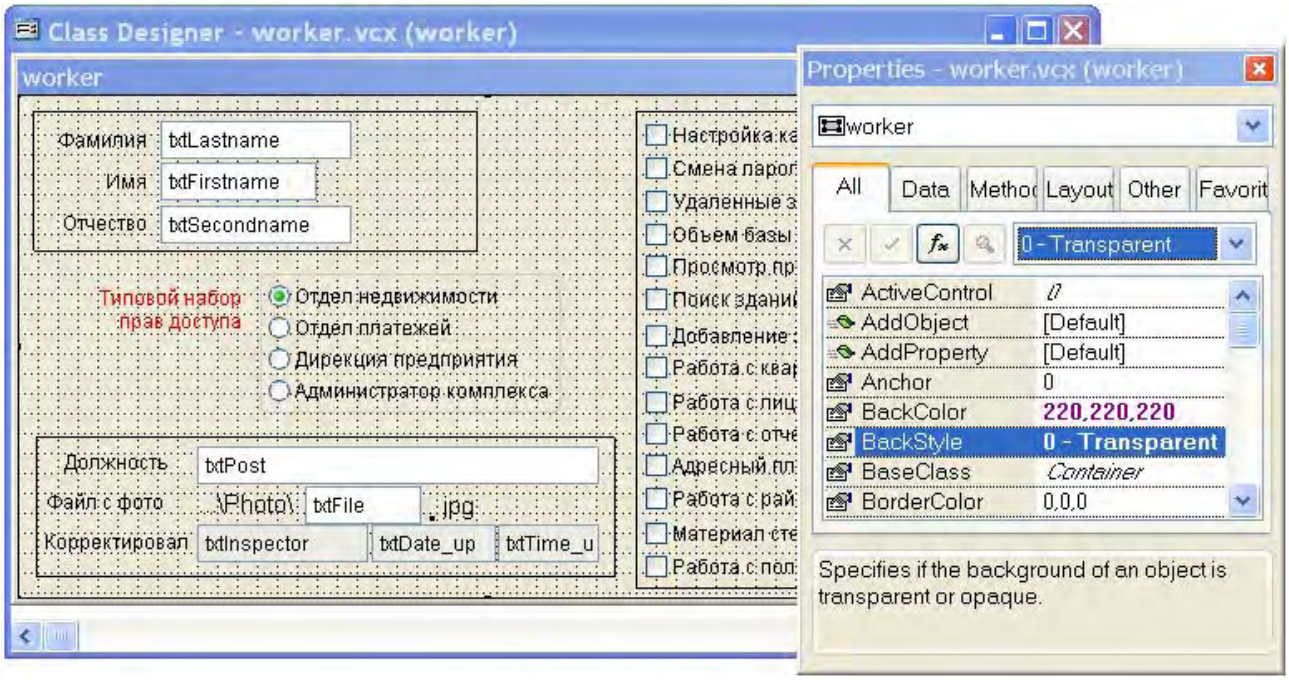


Рис. 4.8. Класс ***Worker*** в конструкторе классов

Поскольку основа приложения будет использована в течение всего процесса разработки приложения, она должна способствовать этому процессу. Значительную часть времени разработчик уделяет написанию кода отладки. Использование объектно-ориентированного метода позволяет вводить возможности отладки на самом низком уровне. Это не слишком сильно усложняет процесс создания программы, но благодаря наследованию позволяет в дальнейшем отлаживать любой фрагмент программного комплекса.

После создания класса приступим к разработке формы ***Access***. В окружении данных формы (***Data Environment***) разместим таблицу ***User***.

Вы уже умеете это делать. Добавить класс в форму не сложно. Откройте панель ***Form Controls*** (элементы управления формы). Она показана на рис. 4.9. Если панель отсутствует на экране – выберите в главном меню Visual FoxPro пункт ***View***, а в открывшемся подменю пункт ***Toolbars***. Откроется окно ***Toolbars***. Сделайте отметку напротив названия панели – ***Form Controls*** и щелкните кнопку ***OK***.

Выберите на панели значок ***View Classes***. Откроется меню, выберите в нем первый пункт ***Add***. В открывшемся диалоговом окне ***Open*** найдите библиотеку классов ***Worker.vcx*** и класс ***Worker***. После щелчка по ***OK*** на панели ***Form Controls*** появится пиктограмма класса  , а сама панель станет значительно меньше. Исчезнут стандартные классы (***Standard***). Щелкните по пиктограмме. В нужном месте активной области формы при помощи левой кнопки мыши отведите место для класса ***Worker***.

Рис. 4.9. Панель ***Form Controls*** (элементы управления формы)

Форма ***Access*** и ее окружение данных в конструкторе форм показана на рис. 4.10.

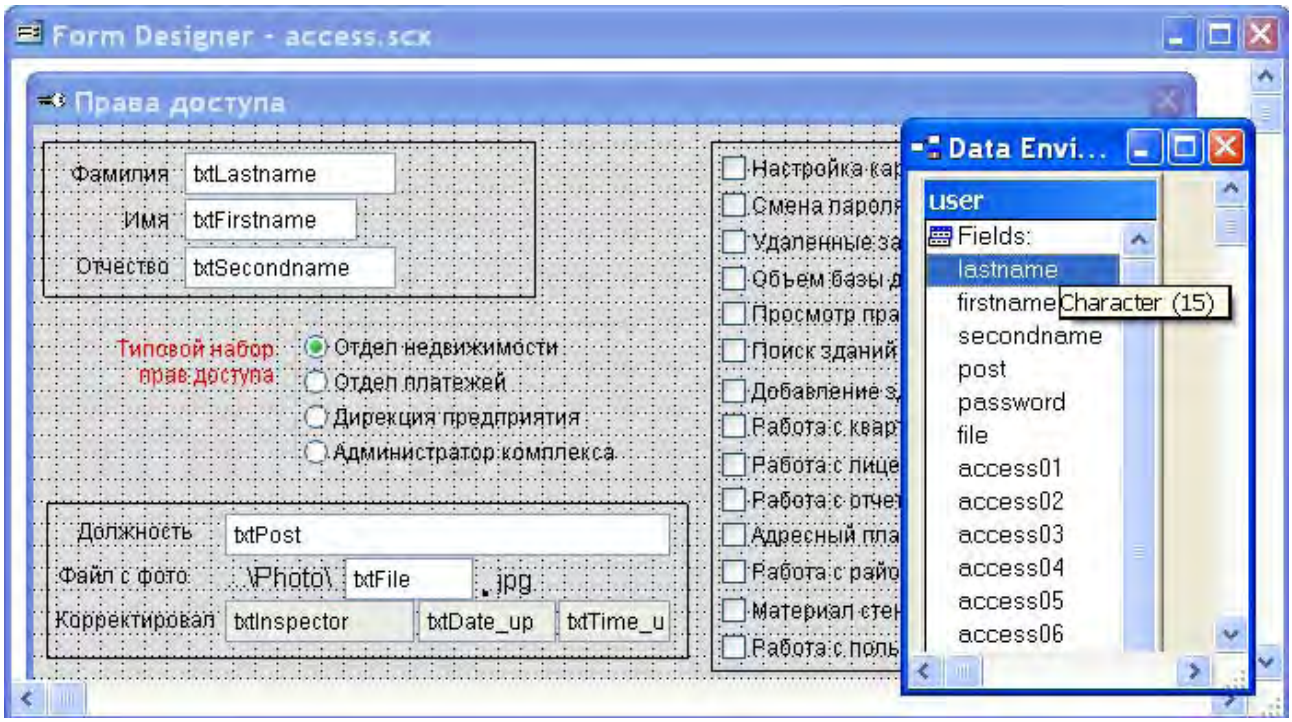


Рис. 4.10. Форма ***Access*** и ее окружение данных в конструкторе форм

4.2.3. Форма Employee – назначение прав доступа

Приступим к разработке формы ***Employee***. Это будет двухстраничная форма, на первой вкладке которой – список работников предприятия, а на второй – подробности по выбранному работнику. В окружении данных формы (***Data Environment***) разместим таблицу ***User***. Вы уже умеете это делать. Добавить объект ***Page Frame*** не сложно. Откройте панель ***Form Controls*** (элементы управления формы). Она показана на рис. 4.9. Если панель отсутствует на экране – выберите в главном меню Visual FoxPro пункт ***View***, а в открывшемся подменю пункт ***Toolbars***. Откроется окно ***Toolbars***. Сделайте отметку напротив названия панели – ***Form Controls*** и щелкните кнопку ***OK***.

Выберите на панели значок ***Page Frame***, а в нужном месте активной области формы при помощи левой кнопки мыши отведите место для этого объекта. Форма ***Employee*** с активной первой страницей и окружением данных в конструкторе форм показана на рис. 4.11.

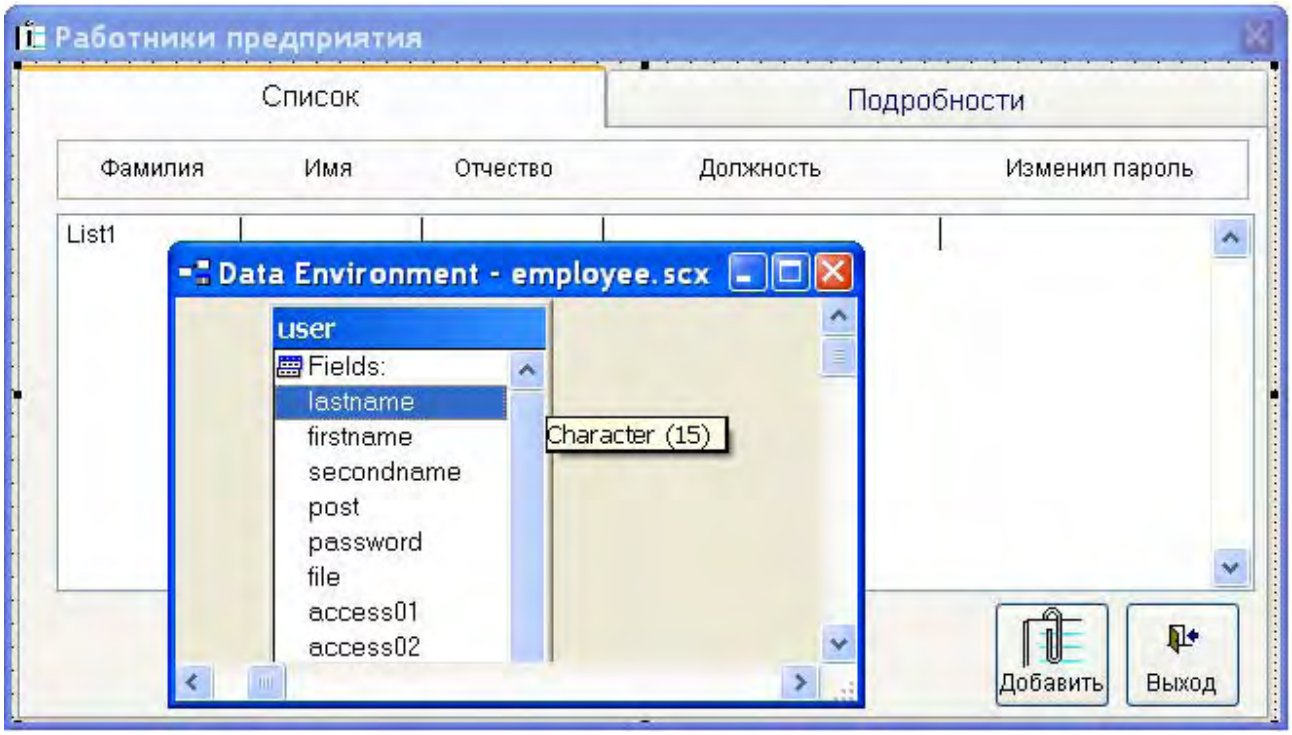


Рис. 4.11. Форма ***Employee*** в конструкторе форм

Добавим код, который будет запушен на выполнение при наступлении события ***Load*** формы:

```
* Описание глобальной переменной IND (Индикатор)
* Признак выбора работника из списка на первой странице формы
PUBLIC IND
IND=0           && Начальное значение
```

```
* Отображение фамилий работников предприятия по алфавиту
SELECT USER
SET ORDER TO TAG LastName
```

Код события ***Activate*** первой страницы формы ***Employee***:

```
* Если в процессе работы с формой выбор работника уже был
* сделан, (IND=1 или IND=2)то при наступлении этого события
* значение Индикатора принять равным единице
IF IND>=1
   IND=1
ENDIF
* Сделать активным поле со списком LIST1
THISFORM.PAGEFRAME1.PAGE1.LIST1.SETFOCUS
* Перерисовать форму
THISFORM.PAGEFRAME1.PAGE1.LIST1.REFRESH
```

Событие ***Interactive Change*** поля со списком ***List1***:

```
IND=1 && Выбор сделан
* Переход на вторую страницу формы при помощи кода не нужен
```

Событие ***DblClick*** поля со списком ***List1***:

```
IND=1  && Выбор в поле List1 сделан
* Перейти на вторую страницу формы
THISFORM.PAGEFRAME1.ACTIVEPAGE=2
```

Событие ***Click*** кнопки ***Добавить***:

```
IND=2  && Была выбрана кнопка Добавить
* Перейти на вторую страницу формы
THISFORM.PAGEFRAME1.ACTIVEPAGE=2
```

Перейдем на вторую страницу ***Page Frame1*** формы ***Employee*** и разместим на ней класс ***Worker***. Для размещения класса в форме необходимо проделать следующее. Выберите на панели ***Form Controls*** (элементы управления формы) значок ***View Classes*** (рис. 4.9). Откроется меню, выберите в нем первый пункт ***Add***. В открывшемся диалоговом окне ***Open*** найдите библиотеку классов ***Worker.vcx*** и класс ***Worker***. После щелчка по кнопке ***OK*** окна ***Open*** на панели ***Form Controls*** появится пиктограмма класса ***Worker***, а сама панель станет значительно меньше. Исчезнут стандартные классы (***Standard***). Новый вид панели показан на рис. 4.12.

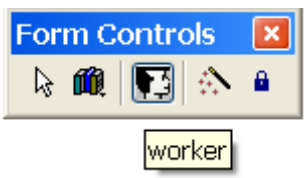


Рис. 4.12. Новый вид панели Form Controls

Щелкните по пиктограмме класса. В нужном месте активной области формы при помощи левой кнопки мыши отведите место для класса ***Worker***. После того,

как вы отпустите левую кнопку мыши, класс появится в форме. Добавим справа от класса четыре кнопки (рис. 4.13) и займемся написанием кода для обработки событий.

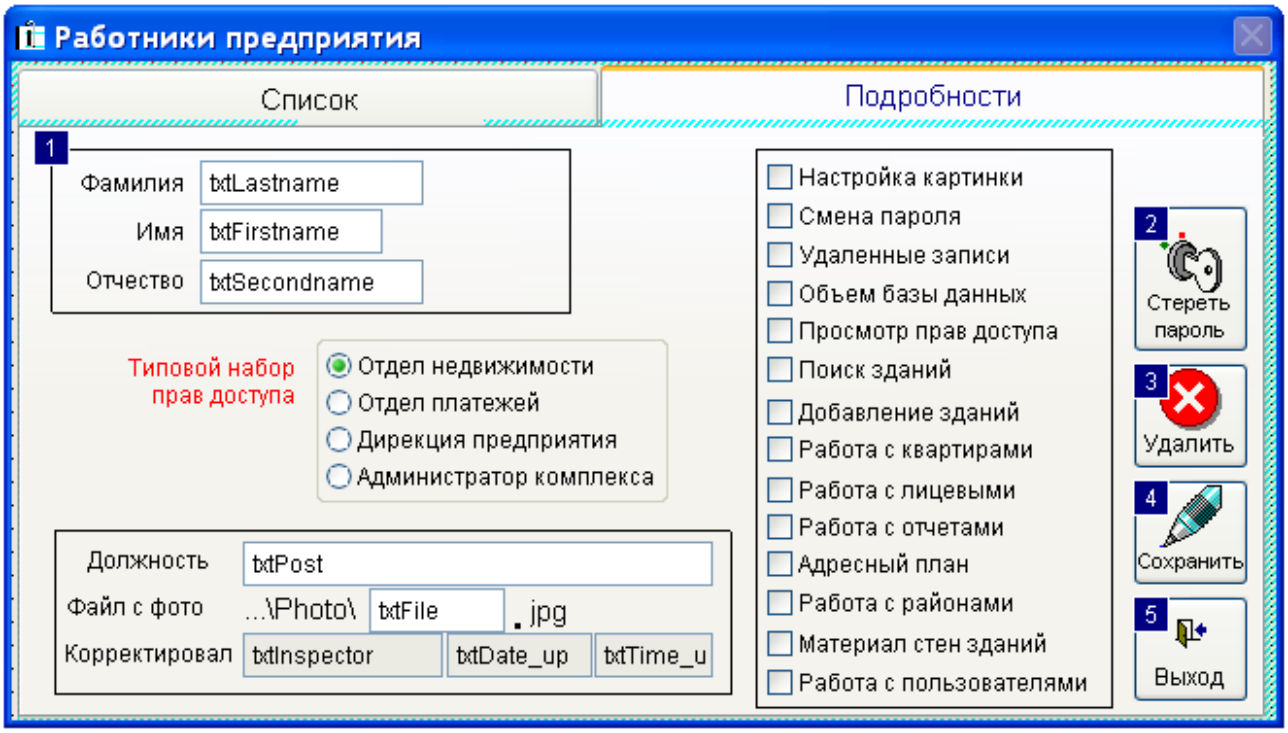


Рис. 4.13. Вторая страница формы Employee

Код события ***Activate*** второй страницы формы ***Employee***:

```
DO CASE
   CASE IND=0           && Выбор не сделан
     =MESSAGEBOX('Ни один работник не выбран! ',;
                 48,' Внимание!')
     * Вернуться на первую страницу формы
     THISFORM.PAGEFRAME1.ACTIVEPAGE=1
   CASE IND=1           && Выбор сделан
     * Кнопка Стереть пароль доступна
     THISFORM.PAGEFRAME1.PAGE2.COMMAND2.ENABLED=.T.
     * Надпись на кнопке 3 - Сохранить
     THISFORM.PAGEFRAME1.PAGE2.COMMAND3.CAPTION=[Сохранить]
     * Кнопка Удалить доступна
     THISFORM.PAGEFRAME1.PAGE2.COMMAND4.ENABLED=.T.
     * Создать переменные памяти
     SCATTER MEMVAR
   CASE IND=2         && Занесение нового работника
     * Кнопка Стереть пароль недоступна
     THISFORM.PAGEFRAME1.PAGE2.COMMAND2.ENABLED=.F.
     * Надпись на кнопке 3 - Записать
     THISFORM.PAGEFRAME1.PAGE2.COMMAND3.CAPTION=[Записать]
     * Кнопка Удалить недоступна
```

```
     THISFORM.PAGEFRAME1.PAGE2.COMMAND4.ENABLED=.F.
     * Создать переменные памяти с пустыми значениями
     SCATTER MEMVAR BLANK
     THISFORM.PAGEFRAME1.PAGE2.Worker1.TxtLastName.SETFOCUS
ENDCASE
* Перерисовать вторую страницу формы
This.Refresh
```

Добавим код события ***Interactive Change*** переключателя ***Optiongroup1*** класса ***Worker***. В отличие от формы ***Access*** форма ***Employee*** предназначена не только для просмотра, но и для корректировки прав доступа. Текст этого события будет храниться и работать только в форме ***Employee***.

```
* Общие права доступа для всех категорий работников
ThisForm.PageFrame1.Page2.Worker1.chkAccess01.Value= .T.
ThisForm.PageFrame1.Page2.Worker1.chkAccess02.Value= .T.
ThisForm.PageFrame1.Page2.Worker1.chkAccess03.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess04.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess05.Value= .T.
ThisForm.PageFrame1.Page2.Worker1.chkAccess06.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess07.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess08.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess09.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess10.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess11.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess12.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess13.Value= .F.
ThisForm.PageFrame1.Page2.Worker1.chkAccess14.Value= .F.
* Индивидуальные права доступа
DO CASE
 CASE THIS.Value=1    && Отдел недвижимости
    * Поиск зданий
    ThisForm.PageFrame1.Page2.Worker1.chkAccess06.Value=.T.
    * Добавление зданий
    ThisForm.PageFrame1.Page2.Worker1.chkAccess07.Value=.T.
    * Работа с квартирами
    ThisForm.PageFrame1.Page2.Worker1.chkAccess08.Value=.T.
    * Работа с отчетами
    ThisForm.PageFrame1.Page2.Worker1.chkAccess10.Value=.T.
    * Работа с адресным планом
    ThisForm.PageFrame1.Page2.Worker1.chkAccess11.Value=.T.
    * Работа со справочником материалов
    ThisForm.PageFrame1.Page2.Worker1.chkAccess13.Value=.T.
 CASE THIS.Value=2   && Отдел платежей
    * Поиск зданий
    ThisForm.PageFrame1.Page2.Worker1.chkAccess06.Value=.T.
    * Работа с квартирами
    ThisForm.PageFrame1.Page2.Worker1.chkAccess08.Value=.T.
    * Работа с лицевыми счетами
    ThisForm.PageFrame1.Page2.Worker1.chkAccess09.Value=.T.
```

```
 CASE THIS.Value=3   && Дирекция предприятия
    * Поиск зданий
    ThisForm.PageFrame1.Page2.Worker1.chkAccess06.Value=.T.
    * Работа с квартирами
    ThisForm.PageFrame1.Page2.Worker1.chkAccess08.Value=.T.
    * Работа с отчетами
    ThisForm.PageFrame1.Page2.Worker1.chkAccess10.Value=.T.
 CASE THIS.Value=4   && Администратор комплекса
    * Удаленные записи
    ThisForm.PageFrame1.Page2.Worker1.chkAccess03.Value=.T.
    * Объем базы данных
    ThisForm.PageFrame1.Page2.Worker1.chkAccess04.Value=.T.
    * Работа с формой прав доступа
    ThisForm.PageFrame1.Page2.Worker1.chkAccess14.Value=.T.
ENDCASE
```

Особое место среди кнопок, размещенных на второй странице формы, занимает кнопка ***Стереть пароль***. Она нужна только администратору информационной системы в случае, если работник забыл свой пароль. Программный код для этой кнопки предусматривает не только стирание пароля, но и занесение нового (5 нулей рядом) в зашифрованном виде. Это сделано на тот случай, если злоумышленник через ODBC, подключив любого клиента, или установив на рабочей станции Visual FoxPro, получит доступ к таблице ***User***. Уничтожение пароля одного из пользователей ему ничего не даст, так как при регистрации под чужим именем сработает код кнопки ***Вход*** формы ***Login*** и на экране нарушитель получит сообщение (рис. 4.14).

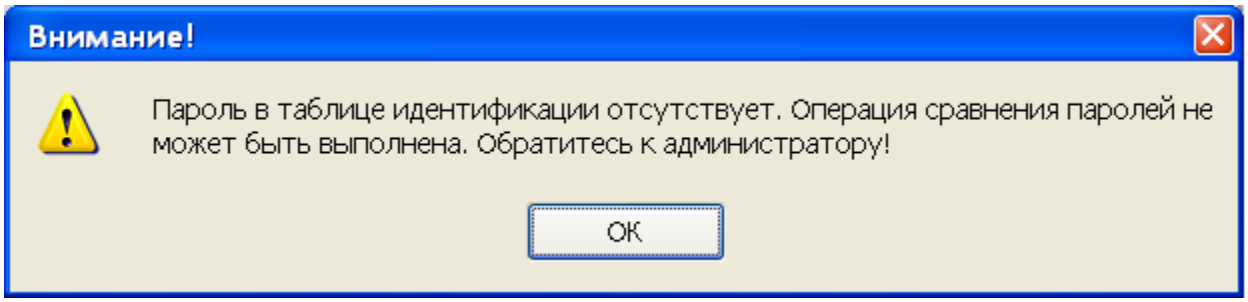


Рис. 4.14. Попытка стереть пароль, минуя программный комплекс

Попытка подсмотреть пароль в таблице ***User*** также ничего не даст. Он хранится там в зашифрованном виде.

Код события ***Click*** кнопки ***Стереть пароль*** имеет вид

```
lnMsgResult=MESSAGEBOX('Подтвердите свои действия',;
                       36,' Удаление пароля ')
IF lnMsgResult=6      && Кнопка Да
   * Стереть пароль - занести пять нулей
   M.PassWord=[00000]
   M.PassWord=CrKod(M.PassWord)  && Зашифровка пароля
   M.Inspector=FAMILY            && Фамилия удалившего пароль
```

```
   M.Date_Up=DATE()              && Дата удаления пароля
   M.Time_Up=TIME()              && Время удаления пароля
   * Занести значения переменных памяти в поля текущей записи
   GATHER MEMVAR
   * Перейти на первую страницу формы
   THISFORM.PAGEFRAME1.ACTIVEPAGE=1
ENDIF
```

Код события ***Click*** кнопки ***Удалить*** имеет вид

```
lnMsgResult=MESSAGEBOX('Подтвердите свои действия',;
            36,' Удаление работника ')
IF lnMsgResult=6      && Кнопка Да
   * Удаление текущей записи
   DELETE
   * Перейти к первой записи в таблице User
   GOTO TOP
   * Перейти на первую страницу формы
   THISFORM.PAGEFRAME1.ACTIVEPAGE=1
ENDIF
```

Кнопка ***Сохранить*** в случае занесения нового работника имеет название ***Записать*** (смотри событие ***Activate*** второй страницы формы). Для нового работника, кроме учетной информации и прав доступа заносится еще и пароль – опять же 00000 в зашифрованном виде. Смена начального пароля отныне на совести «новичка».

Код события ***Click*** кнопки ***Сохранить*** имеет вид

```
* Выполнение присваивания значений переменным памяти
* Контроль ввода фамилии
M.LastName=ThisForm.PageFrame1.Page2.Worker1.TxtLastName.Value
IF LEN(ALLTRIM(M.LastName))=0
   =MESSAGEBOX(' Вы забыли про фамилию!',;
            48,' Ошибка')
   ThisForm.PageFrame1.Page2.Worker1.TxtLastName.SetFocus
   RETURN
ENDIF
* Контроль ввода имени
M.FirstName=ThisForm.PageFrame1.Page2.Worker1.TxtFirstName.Value
IF LEN(ALLTRIM(M.FirstName))=0
   =MESSAGEBOX(' Вы забыли про имя работника!',;
            48,' Ошибка')
   ThisForm.PageFrame1.Page2.Worker1.TxtFirstName.SetFocus
   RETURN
ENDIF
* Контроль ввода отчества
M.SecondName=;
           ThisForm.PageFrame1.Page2.Worker1.TxtSecondName.Value
IF LEN(ALLTRIM(M.SecondName))=0
```

```
   =MESSAGEBOX(' Вы забыли про отчество работника!',;
            48,' Ошибка')
   ThisForm.PageFrame1.Page2.Worker1.TxtSecondName.SetFocus
   RETURN
ENDIF
* Контроль ввода должности
M.Post=ThisForm.PageFrame1.Page2.Worker1.TxtPost.Value
IF LEN(ALLTRIM(M.Post))=0
   =MESSAGEBOX(' Вы забыли про должность работника!',;
            48,' Ошибка')
   ThisForm.PageFrame1.Page2.Worker1.TxtPost.SetFocus
   RETURN
ENDIF
* Контроль ввода кода доступа
M.Range=ThisForm.PageFrame1.Page2.Worker1.OptionGroup1.Value
IF M.Range=0
   =MESSAGEBOX(' Вы забыли про код доступа!',;
            48,' Ошибка')
   RETURN
ENDIF
lnMsgResult=MESSAGEBOX('Подтвердите свои действия',;
     36,' Запись на сервер ')
IF lnMsgResult=6      && Кнопка Да
 M.Inspector=FAMILY     && Фамилия выполнившего ввод информации
 M.Date_Up=DATE()       && Дата ввода информации
 M.Time_Up=TIME()       && Время ввода информации
 * Права доступа
 M.Access01=ThisForm.PageFrame1.Page2.Worker1.chkAccess01.Value
 M.Access02=ThisForm.PageFrame1.Page2.Worker1.chkAccess02.Value
 M.Access03=ThisForm.PageFrame1.Page2.Worker1.chkAccess03.Value
 M.Access04=ThisForm.PageFrame1.Page2.Worker1.chkAccess04.Value
 M.Access05=ThisForm.PageFrame1.Page2.Worker1.chkAccess05.Value
 M.Access06=ThisForm.PageFrame1.Page2.Worker1.chkAccess06.Value
 M.Access07=ThisForm.PageFrame1.Page2.Worker1.chkAccess07.Value
 M.Access08=ThisForm.PageFrame1.Page2.Worker1.chkAccess08.Value
 M.Access09=ThisForm.PageFrame1.Page2.Worker1.chkAccess09.Value
 M.Access10=ThisForm.PageFrame1.Page2.Worker1.chkAccess10.Value
 M.Access11=ThisForm.PageFrame1.Page2.Worker1.chkAccess11.Value
 M.Access12=ThisForm.PageFrame1.Page2.Worker1.chkAccess12.Value
 M.Access13=ThisForm.PageFrame1.Page2.Worker1.chkAccess13.Value
 M.Access14=ThisForm.PageFrame1.Page2.Worker1.chkAccess14.Value
   IF THIS.CAPTION=[Записать]
      * При занесении нового работника пароль - пять нулей
      M.PassWord=[00000]
      M.PassWord=CrKod(M.PassWord)
      APPEND BLANK
   ENDIF
 GATHER MEMVAR
 * Переход на первую страницу формы
 THISFORM.PAGEFRAME1.ACTIVEPAGE=1
ENDIF
```

4.2.4. Форма Password – изменение пароля

В правилах разграничения доступа (ПРД), действующих на предприятии или в организации должен быть указан срок действия пароля пользователя. По истечении этого срока программный комплекс должен предложить работнику сменить пароль (смотри событие ***Click*** кнопки ***Вход*** формы ***Login***). Для этих целей создадим форму ***Password***. Она также может быть запущена работником на выполнение раньше истечения контрольного срока (пункт меню ***Поддержка*** – ***Смена пароля***).

Форма ***Password*** показана на рис. 4.15. В окне свойств формы представлены только измененные свойства. Остальные для наглядности удалены.

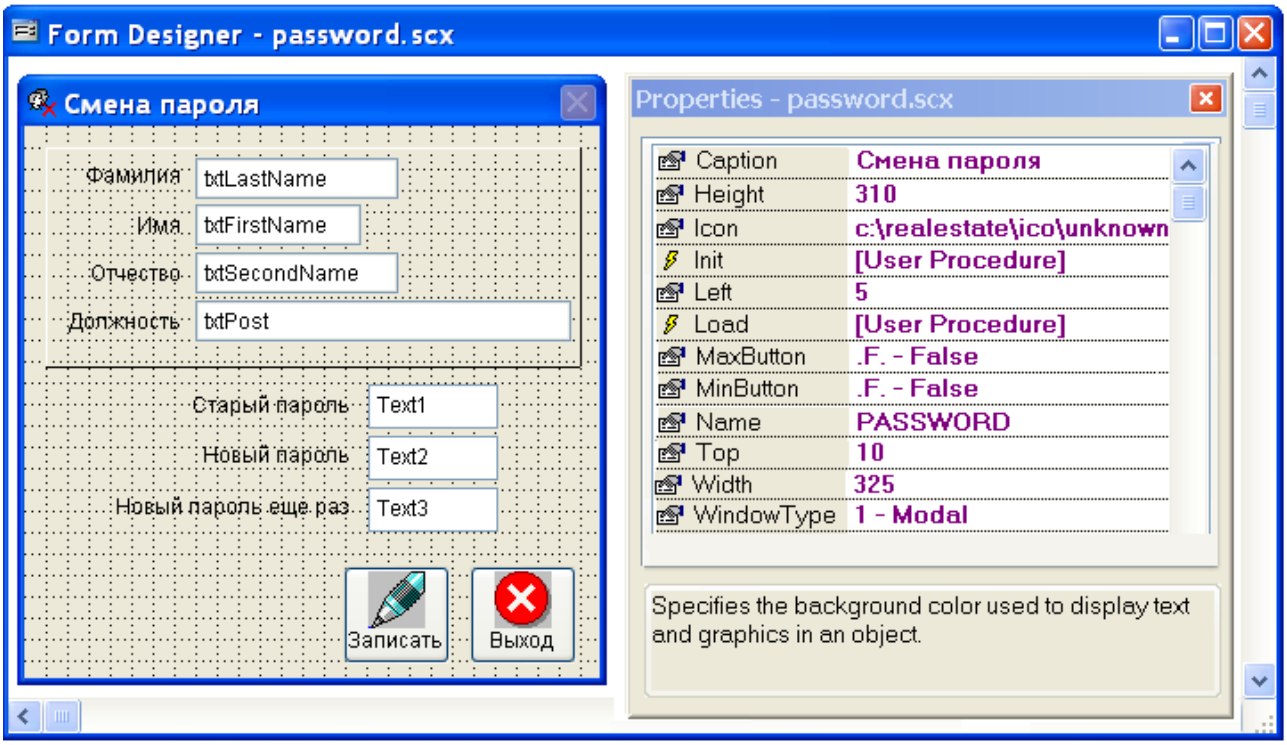


Рис. 4.15. Форма ***Password*** в конструкторе форм

Добавим код, который будет запушен на выполнение при наступлении события ***Load*** формы ***Password***:

```
* Назначение индексного файла для поиска
SELECT USER
SET ORDER TO TAG LastName
* Поиск в таблице User учетной записи работника
SEEK FAMILY
```

Код события ***Init*** формы ***Password*** имеет вид

```
* Временные переменные
PRIVATE Old,New1,New2
```

```
* Начальное значение старого пароля и двух значений
* нового пароля
STORE [ ] TO Old,New1,New2
```

Код события ***Click*** кнопки ***Записать*** имеет вид

```
* Создание переменных памяти
SCATTER MEMVAR
* Сравнение по полному соответствию
SET EXACT ON
IF UnKod(M.PassWord)#ALLTRIM(Old)
   =MESSAGEBOX('Неправильно введен старый пароль ',;
               48,'Ошибка!')
   SET EXACT OFF
   * Сделать активным поле Старый пароль
   THISFORM.TEXT1.SetFocus
   RETURN
ENDIF
SET EXACT OFF
* Удаление концевых пробелов в значениях пароля
New1=ALLTRIM(New1)
New2=ALLTRIM(New2)
SET EXACT ON
IF New1#New2
   =MESSAGEBOX('Два значения нового пароля не совпадают ',;
               48,'Ошибка!')
   SET EXACT OFF
   THISFORM.TEXT2.SetFocus
   RETURN
ENDIF
SET EXACT OFF
IF LEN(New1)<=4
   =MESSAGEBOX('Длина нового пароля должна быть '+;
                ' не менее 5 символов ',48,'Ошибка!')
   * Сделать активным поле Новый пароль
   THISFORM.TEXT2.SetFocus
   RETURN
ENDIF
New3=CrKod(New1)  && Зашифрованный новый пароль
SET EXACT ON
IF ALLTRIM(New3)=ALLTRIM(M.PassWord)
   =MESSAGEBOX('Новый и старый пароль совпадают ',;
               48,'Ошибка!')
   SET EXACT OFF
   * Сделать активным поле Старый пароль
   THISFORM.TEXT2.SetFocus
   RETURN
ENDIF
SET EXACT OFF
M.PassWord=New3          && Зашифрованный новый пароль
```

```
M.Inspector=FAMILY        && Фамилия изменившего пароль
M.Date_Up=Date()          && Дата изменения пароля
M.Time_Up=Time()          && Время изменения пароля
* Сброс значений переменных памяти в поля записи
GATHER MEMVAR
* Закрытие формы
THISFORM.Release
```

4.3. Создание основных форм приложения

Мы разрабатываем приложение для его применения в локальной вычислительной сети. С ним будет работать несколько человек одновременно. Настало время выбрать стратегию урегулирования многопользовательских конфликтов. Они могут возникать, когда пользователи мешают друг другу. Обычно это происходит в том случае, если сразу несколько человек пытаются получить доступ к одним и тем же данным и изменить их. Конфликт, как правило, вызван тем, что два или более пользователя имеют доступ к одним и тем же данным и могут одновременно редактировать их. Это снижает доверие к базе данных, в частности, еще и потому, что в нее иногда вносится противоречивая информация.

4.3.1. Разработка многопользовательского приложения

Visual FoxPro хорошо работает в многопользовательских средах. Рассмотрим ситуации, в которых возникают конфликты, и те инструменты Visual FoxPro, которые помогают их предотвратить или разрешить. Сначала разберем механизмы блокировки, которые предоставляет Visual FoxPro. После этого перейдем к изучению стратегии для отдельных таблиц (буферизация строк и таблиц).

Если ваше приложение обслуживает локальную сеть, Intranet или Web-сервер, нельзя исключать возможность того, что сразу несколько пользователей захотят редактировать одни и те же данные. Конечно, если вам повезет, этого не произойдет. Однако вероятность подобных столкновений увеличивается по мере того, как растет число пользователей системы. Чем их больше, тем чаще возникают конфликты. Если вы не хотите полагаться на случай, то вставьте в свое приложение коды, благодаря которым оно сможет должным образом урегулировать возникающие проблемы.

Управление межпользовательскими конфликтами в Visual FoxPro состоит в следующем: задаются различные команды установок и блокировок, которые будут служить «арбитражем» для конкурирующих запросов. Блокировка снабжает пользователя «замком», который не дает другим пользователям изменять его данные. Хотелось бы, чтобы это происходило автоматически, без применения кода. До определенной степени это действительно возможно. Visual FoxPro предоставляет несколько режимов

блокировки, которые требуют минимум кодов. Также VIisual FoxPro автоматически управляет запросами и по умолчанию посылает пользователю сообщения при конфликтах. Однако не существует стратегии, которая годилась бы для всех возможных ситуаций, поэтому лучше изучить все доступные команды Visual FoxPro.

Причины конфликтов между пользователями. Конфликты возможны, когда несколько пользователей одновременно стремятся получить доступ к одной и той же части базы данных. В этом случае ваше приложение должно «решить», кому из них отдать предпочтение.

Обстоятельства, при которых возникают конфликты между пользователями. Конфликты могут возникнуть как минимум в трех ситуациях: при редактировании и запросе данных, а также сопровождении базы данных.

- *Конфликты при редактировании.* Случается, что два или более пользователя одновременно пытаются редактировать одни и те же данные. Возможно ли сделать это в вашем приложении? Или оно «разрешит» редактирование одному из пользователей, а остальные будут ждать своей очереди?
- *Конфликты при запросе*. Эта ситуация возникает, когда один из пользователей запускает длительный запрос или отчет, и важно, чтобы данные в нем были непротиворечивы. Если вы позволите другим пользователям в это время редактировать данные, в них могут быть введены новые значения, которые сделают результаты обрабатываемого запроса недействительными.
- *Конфликты при сопровождении*. Иногда необходимо переиндексировать таблицы или создать резервную копию базы данных, а эти действия могут совершаться только одним пользователем. Что, если в это время в системе будут другие пользователи, которые помешают вашей программе получить эксклюзивный доступ к нужным таблицам? Способы разрешения подобных конфликтов необходимо предусмотреть в любом приложении баз данных, так как подобные столкновения практически неизбежны.

При разработке стратегии разрешения конфликтов между пользователями надо учитывать:

- тип используемых данных;
- требования к эффективности работы приложения;
- особые требования пользователя;
- вероятность конфликтов;
- требования к организации сопровождения приложения.

Тип используемых данных. Иногда типы данных настолько отличаются, что вам не удастся выработать единую стратегию. Кроме того, они могут оказаться разными для разных приложений. Нередко базы данных требуют очень аккуратного обращения. В этом случае нельзя допускать, чтобы пользователь вносил в них какие-либо изменения и писал поверх того, что написано другим. Если вы это разрешите, данные станут противоречивы-

ми, а пользователь запутается. Однако данные не всегда так уязвимы, и если один пользователь станет писать поверх изменений, сделанных другим, то это не обязательно вызовет серьезные последствия.

Требования к эффективности работы приложения. Иногда при выборе стратегии разрешения конфликта важны соображения эффективности. Допустим, ваше приложение работает медленно, когда к определенной области данных обращается сразу много пользователей. Поэтому для увеличения скорости можно выбрать стратегию, которая сведет к минимуму накладные расходы сети. Частые блокировки сети приводят к дополнительным издержкам, а повторяющиеся сообщения о блокировке создают впечатление медлительности приложения и мешают пользователям вводить данные.

Особые требования пользователя. У каждого пользователя свои особенности характера. Возможно, он потребует, чтобы в любой момент времени доступ к определенному фрагменту информации имел только один человек или процесс. Например, у одного из ваших клиентов может быть деловое правило, гласящее, что с лицевым счетом ***Account*** в каждый момент времени работает только один инспектор. Тогда ваше приложение должно отвечать этому условию. Другой клиент может предъявить прямо противоположное требование. Поэтому всегда старайтесь выяснить, есть ли у пользователя специальные пожелания.

Вероятность конфликтов. В некоторых случаях она крайне невелика. Например, вряд ли в многотысячном списке квартир ***Flats*** два пользователя захотят одновременно редактировать одну и ту же запись. А вот в маленьких таблицах, которые постоянно используются, конфликты гораздо более вероятны.

Требования к организации сопровождения приложения. При обсуждении блокировки часто забывают, что каждое рабочее приложение должно обладать некоторыми средствами сопровождения. В их функции входят диагностика, обновление и переиндексация. Часто такие действия требуют эксклюзивного доступа к системе. Например, это необходимо при переиндексации и упаковке таблицы. Аналогично при диагностике и глобальных изменениях в базе данных может понадобиться эксклюзивный доступ к базе в целом. Во всех этих случаях для других пользователей доступ должен быть запрещен. Многие приложения осуществляют сопровождение, запуская специальные программы в ненапряженный период (например, около полуночи). При этом используется программа запуска сопровождения.

Итак, когда один из пользователей редактирует некоторые данные, другие пользователи не должны иметь к ним доступа. Если вы приняли такое решение, можете использовать *блокировку* – инструмент, который есть во всех базах данных. Взаимодействуя с вашей операционной системой и/или локальной вычислительной сетью, Visual FoxPro устанавливает и

снимает блокировки на основании запросов, которые поступают от запущенных экземпляров ваших приложений.

Необходимо гарантировать согласованность записи в базу данных. Поэтому Visual FoxPro, проводя изменения в некотором наборе данных, должен блокировать его. Возможно, это целая таблица, структуру которой вы меняете. Возможно, это заголовок таблицы, в котором обновляется количество строк таблицы. Возможно, это отдельная строка, которую нужно изменить. Обработчик базы данных Visual FoxPro гарантирует, что в конкретный момент времени работать с определенным множеством данных может только один пользователь, даже если вы не задаете такое требование специально.

4.3.2. Типы блокировок в Visual FoxPro

На сегодняшний день в базах данных принято использовать три вида блокировок: эксклюзивную, общую и блокировку изменений.

- *Эксклюзивная блокировка.* Пользователь, установивший эксклюзивную блокировку некоторых данных, имеет эксклюзивные права чтения этих данных и записи в них. Другие пользователи не могут не только писать в эти данные, но даже и читать их. Visual FoxPro обеспечивает такую блокировку только на уровне таблицы и базы данных. При работе с Visual FoxPro нельзя эксклюзивно блокировать отдельную строку или набор строк в таблице.
- *Общая блокировка.* Установленная пользователем, позволяет ему и другим пользователям читать данные, но не делать в них записи. Этот тип блокировки обычно используется для отчетов и запросов, чтобы гарантировать непротиворечивость данных. В Visual FoxPro отсутствует.
- *Блокировка изменений.* Позволяет только одному пользователю писать в заблокированные данные. Другие пользователи могут читать их, несмотря на блокировку, но не писать в них. Visual FoxPro обеспечивает блокировку изменений на уровне строк и таблиц. Блокировка изменений в Visual FoxPro бывает двух типов – *пессимистическая и оптимистическая*. При оптимистической блокировке никакие строки или таблицы не блокируются до момента обновления, а при обновлении – только ненадолго. Все это благодаря буферизации. На период внесения изменений в строку или таблицу эти объекты сохраняются в специальном буфере. После завершения корректировки сбрасываются обратно на диск.

Чтобы понять различные команды блокировки в Visual FoxPro, рассмотрим их в порядке сужения сферы их действия, т. е. того, на что они распространяются. Блокировка может распространяться на всю базу данных, отдельную таблицу, строки данных таблицы или на столбец таблицы.

- *Блокировка на уровне базы данных.* При открытии база данных в Visual FoxPro всегда имеет определенный статус блокировки: либо эксклюзивный, либо общий. Команда ***Open Database Real Estate*** открывает базу данных как эксклюзивную или как общую, в зависимости от значения установки ***Set Exclusive***. Эту установку можно задать либо с помощью команды ***Set Exclusive***, либо последовательно открыв диалоговые окна ***Tools***, ***Options*** и отметив флаг ***Exclusive*** во вкладке ***File Locations*** (Расположения файлов). Если значением ***Exclusive*** является ***On***, вышеприведенная команда будет открывать базу данных эксклюзивно.
- *Блокировка на уровне таблицы*. При блокировке на уровне таблицы вы можете использовать эксклюзивную блокировку или блокировку изменений. Открытие таблицы для эксклюзивного доступа очень похоже на открытие базы данных. Вы просто должны запустить команду ***Use*** с ключевым словом ***Exclusive***. Как и в случае с базой данных, такая блокировка не позволит другим пользователям ни изменять, ни даже читать данные в таблице. Вы снимаете эксклюзивную блокировку таблицы, когда закрываете ее посредством команды ***Use***.
- *Блокировка на уровне строк.* Это наиболее распространенный тип блокировки. Другое название – блокировка записей. Visual FoxPro сам блокирует изменение строки во время выполнения некоторых команд. Visual FoxPro блокирует таблицы и строки с помощью собственного обработчика базы данных совместно с операционной системой. Точный метод программистам не раскрывается.
- *Блокировка столбца*. В Visual FoxPro не поддерживается.

Для установки буферизации в программе используется функция ***Cursorsetprop***. Обязательно установите режим буферизации как один из параметров. Например, вы можете указать режим пессимистической блокировки строк, задавая в качестве параметра функции ***Cursorsetprop()*** значение 2:

```
SELECT Street
=Cursorsetprop ("Buffering", 2, "Street")
```

Другими возможными значениями являются:
1 – отключение буферизации;
2 – включение пессимистической буферизации строк;
3 – включение оптимистической буферизации строк;
4 – включение пессимистической буферизации таблиц;
5 – включение оптимистической буферизации таблиц.

4.3.3. Форма Street – справочник улиц

Мы перешли к созданию многопользовательских форм. Этот раздел поможет вам пройти через все этапы создания такой формы. На конкретном примере рассмотрим последствия применения буферизации, обра-

ботку ошибок и сообщения пользователю при выбранном типе буферизации. Самыми распространенными операциями в многопользовательских формах являются **Add** (Добавление), **Edit** (Редактирование) и **Delete** (Удаление). Многопользовательские формы, которые включают буферизацию, должны иметь дело с каждой из этих операций.

Добавление требует наименьших усилий. Если при включенной буферизации пользователь добавляет строку, Visual FoxPro помещает ее в буфер, и никто другой не может редактировать эту строку до обновления таблицы. Таким образом, конфликта между пользователями не возникнет.

На этапе редактирования при оптимистической буферизации конфликты могут возникать, если один из пользователей пытается редактировать и обновлять строку в то время, когда ее уже редактирует кто-то другой. Подчеркну, что столкновения такого рода вероятны только при оптимистической буферизации, но не при пессимистической. При подобном конфликте Visual FoxPro позволяет решить вопрос пользователю, который обновляет строку: Visual FoxPro спрашивает у него, записать ли строку поверх или пользователь отредактирует ее заново.

Конфликт, который возникает при удалении, является несколько более сложным. Возможно, хотя и маловероятно, что два пользователя захотят удалить одну и ту же строку в одно и то же время. Поскольку они оба хотят совершить одинаковое действие, это проблемы не составляет. Однако не исключено, что один из пользователей попытается удалить строку, которую другой в это время редактирует. Поэтому лучше всего задать в процедуре, отвечающей за удаление, следующую функцию: отследить тот факт, что другой пользователь работает над строкой, и если это так, не выполнять удаления.

На рис. 4.16 показано, как будет выглядеть эта форма в режиме разработки. Одновременно показаны обе страницы формы ***Street***.

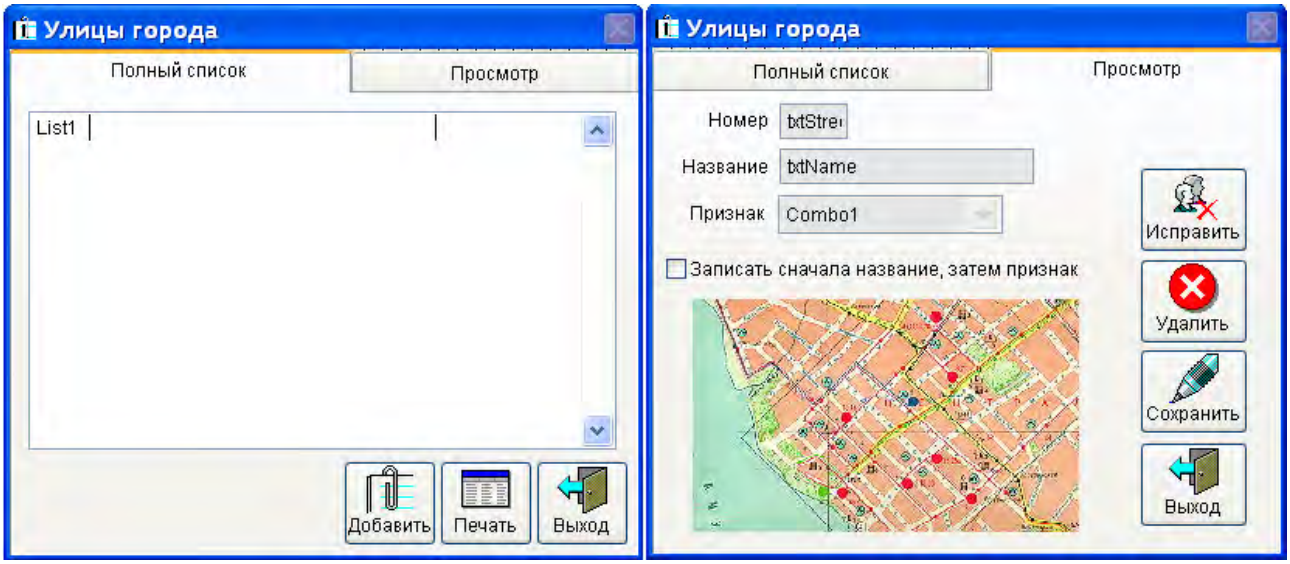


Рис. 4.16. Обе страницы многопользовательской формы ***Street*** в конструкторе

В окружение данных формы добавлена одна таблица – ***Street***, набор полей которой представлен в табл. 2.1. Поле со списком ***List1*** привязано к трем полям ***Street***, ***Name*** и ***Sign***. Изучите внимательно тексты процедур, приведенных ниже. Надеюсь, что достаточное количество комментариев к ним, ответит на все ваши вопросы.

Текст события ***Activate*** второй страницы формы:

```
* Если форма в режиме просмотра
IF This.Caption=[Просмотр]
   * Гасим кнопку сохранить
   This.Command2.Enabled=.F.
   * Делаем недоступными поля одной записи в таблице Street
   This.TxtStreet.Enabled=.F.   && Номер улицы
   This.TxtName.Enabled=  .F.   && Название улицы
   This.Combo1.Enabled=   .F.   && Признак адреса
   This.ChkFirst.Enabled= .F.   && Порядок следования
ENDIF
* Работаем с таблицей Street
SELECT STREET
* Перерисовать вторую страницу формы
This.refresh
```

Текст события ***Click*** кнопки ***Добавить***:

```
Select Street
=CURSORSETPROP("Buffering",3)
* В памяти компьютера создается буфер строки
* Все изменения  заносятся пока только в буфер
* 3 - буферизация строки. Блокировка оптимистическая
* Запись блокируется только на время записи ее на диск
* Добавляем пустую запись (в данном случае - в буфер)
APPEND BLANK
* Делаем активной вторую страницу формы
ThisForm.PageFrame1.ActivePage=2
* Первую страницу делаем недоступной
ThisForm.PageFrame1.page1.Enabled=.F.
* Меняем заголовок второй страницы Просмотр на Добавление
ThisForm.PageFrame1.page2.Caption=[Добавление]
* Заменяем кнопку Выход на Отказ
ThisForm.PageFrame1.page2.Command1.Caption=[Отказ]
* Делаем недоступной кнопку Удалить
ThisForm.PageFrame1.page2.Command3.Enabled=.F.
* Делаем недоступной кнопку "Исправить"
ThisForm.PageFrame1.page2.Command4.Enabled=.F.
* Кнопку сохранить делаем доступной
ThisForm.PageFrame1.page2.Command2.Enabled=.T.
ThisForm.PageFrame1.page2.Label2.Caption=[Буферизация включена]
* Делаем доступными поля одной записи
ThisForm.PageFrame1.page2.TxtStreet.Enabled=.T.
```

```
ThisForm.PageFrame1.page2.TxtName.Enabled=  .T.
ThisForm.PageFrame1.page2.Combo1.Enabled=   .T.
ThisForm.PageFrame1.page2.ChkFirst.Enabled= .T.
* Устанавливаем курсор на поле "Номер"
ThisForm.PageFrame1.page2.txtStreet.SetFocus
```

Текст события ***Click*** кнопки ***Исправить***:

```
SELECT Street
=CURSORSETPROP("Buffering",3)
* В памяти компьютера создается буфер строки
* Все изменения  заносятся пока только в буфер
* 3 - буферизация строки. Блокировка оптимистическая
* Запись блокируется только на время записи ее на диск
* Первую страницу делаем недоступной
ThisForm.PageFrame1.page1.Enabled=.F.
* Меняем заголовок второй страницы Просмотр на Корректировка
ThisForm.PageFrame1.page2.Caption=[Корректировка]
* Заменяем кнопку "Выход" на "Сброс"
ThisForm.PageFrame1.page2.Command1.Caption=[Сброс]
* Делаем доступной кнопку "Сохранить"
ThisForm.PageFrame1.page2.Command2.Enabled=.T.
&& Делаем недоступной кнопку "Удалить"
ThisForm.PageFrame1.page2.Command3.Enabled=.F.
ThisForm.PageFrame1.page2.Label2.Caption=[Буферизация включена]
* Делаем доступными поля одной записи
ThisForm.PageFrame1.page2.TxtStreet.Enabled=.T.
ThisForm.PageFrame1.page2.TxtName.Enabled=  .T.
ThisForm.PageFrame1.page2.Combo1.Enabled=   .T.
ThisForm.PageFrame1.page2.ChkFirst.Enabled= .T.
```

Текст события ***Click*** кнопки ***Удалить***:

```
InMsgResult=MESSAGEBOX('Действительно хотите удалить?',;
            52,'Удаление')
* Вывод информационного окна с заголовком "Удаление"
* Текст в окне 'Действительно хотите удалить?'
* 52 = 4+48+0
* 4 - в информационном окне кнопки "Да" и "Нет"
* 48 - картинка восклицательного знака
* 0 - по умолчанию кнопка "Да"
IF InMsgResult=6   && Если выбрана кнопка "Да"
   SELECT STREET
   DELETE      && Удаление
   GOTO TOP    && Переход к первой записи
   * Делаем активной первую страницу формы
   ThisForm.PageFrame1.ActivePage=1
   * Обновление первой страницы формы
   ThisForm.PageFrame1.Page1.Refresh
   * Делаем активным поле со списком первой страницы
```

```
   ThisForm.PageFrame1.Page1.List1.SetFocus
   * Обновление поля со списком
   ThisForm.PageFrame1.Page1.List1.Refresh
ENDIF
```

Текст события Click кнопки Сохранить:

```
* Проверка "Все поля записи должны быть заполнены"
* Номер улицы
IF EMPTY(Street.Street)=.T.
   =MESSAGEBOX('Вы забыли ввести номер улицы!',48,'Ошибка!')
   ThisForm.PageFrame1.Page2.TxtStreet.Setfocus
   RETURN
ENDIF
* Название улицы
IF EMPTY(Street.Name)=.T.
   =MESSAGEBOX('Вы забыли ввести название улицы!',48,'Ошибка!')
   ThisForm.PageFrame1.Page2.TxtName.Setfocus
   RETURN
ENDIF
* Признак адреса
IF EMPTY(Street.Sign)=.T.
   =MESSAGEBOX('Вы забыли ввести признак адреса!',48,'Ошибка!')
   ThisForm.PageFrame1.Page2.Combo1.Setfocus
   RETURN
ENDIF
=TABLEUPDATE()  && Сброс из буфера в таблицу
=CURSORSETPROP("Buffering",1) && Буферизация выключена
ThisForm.PageFrame1.page1.Enabled=.T.
ThisForm.PageFrame1.page2.Caption=[Просмотр]
ThisForm.PageFrame1.page2.Command1.Caption=[Выход]
* Перечисленные ниже объекты доступны
ThisForm.PageFrame1.page2.Command3.Enabled= .T.
ThisForm.PageFrame1.page2.Command4.Enabled= .T.
ThisForm.PageFrame1.page2.TxtStreet.Enabled=.T.
ThisForm.PageFrame1.page2.TxtName.Enabled=  .T.
ThisForm.PageFrame1.page2.Combo1.Enabled=   .T.
ThisForm.PageFrame1.page2.ChkFirst.Enabled= .T.
* Убираем надпись "Буферизация выключена"
ThisForm.PageFrame1.page2.Label2.Caption=[]
SELECT Street
GOTO TOP
* Активна первая страница формы
ThisForm.PageFrame1.ActivePage=1
* Перерисовать первую страницу формы
ThisForm.PageFrame1.Page1.Refresh
* Сделать активным поле со списком
ThisForm.PageFrame1.Page1.List1.SetFocus
* Перерисовать поле со списком
ThisForm.PageFrame1.Page1.List1.Refresh
```

Текст события ***Destroy*** формы ***Street***:

```
DO CASE
   CASE ThisForm.PageFrame1.page2.Command1.Caption=[Отказ]
        * Пользователь передумал заносить новую улицу
        SELECT Street
        DELETE             && Удаление изменений в буфере
        =TABLEREVERT()     && Отказ от записи на диск
        ThisForm.Release   && Закрытие формы

   CASE ThisForm.PageFrame1.page2.Command1.Caption=[Сброс]
        * Пользователь передумал выполнить корректировку
        SELECT STREET
        =TABLEREVERT()     && Отказ от записи на диск
        ThisForm.Release   && Закрытие формы

   CASE ThisForm.PageFrame1.page2.Command1.Caption=[Выход]
        ThisForm.Release
ENDCASE
```

Обратите внимание на надпись ***Label2*** второй страницы формы. Она используется для индикации состояния буферизации. При создании формы ее свойство ***Caption*** представляет собой строку нулевой длины. В процессе работы оно может меняться на ***Буферизация включена*** и ***Буферизация выключена***.

Остановимся подробнее на создании элемента ***Combo1*** (поле с раскрывающимся списком) второй страницы формы. Оно предназначено для выбора признака адреса. Для его создания можно использовать построитель, а можно назначить свойства вручную. На рис. 4.17 приведен набор этих свойств. Значения по умолчанию для наглядности удалены.

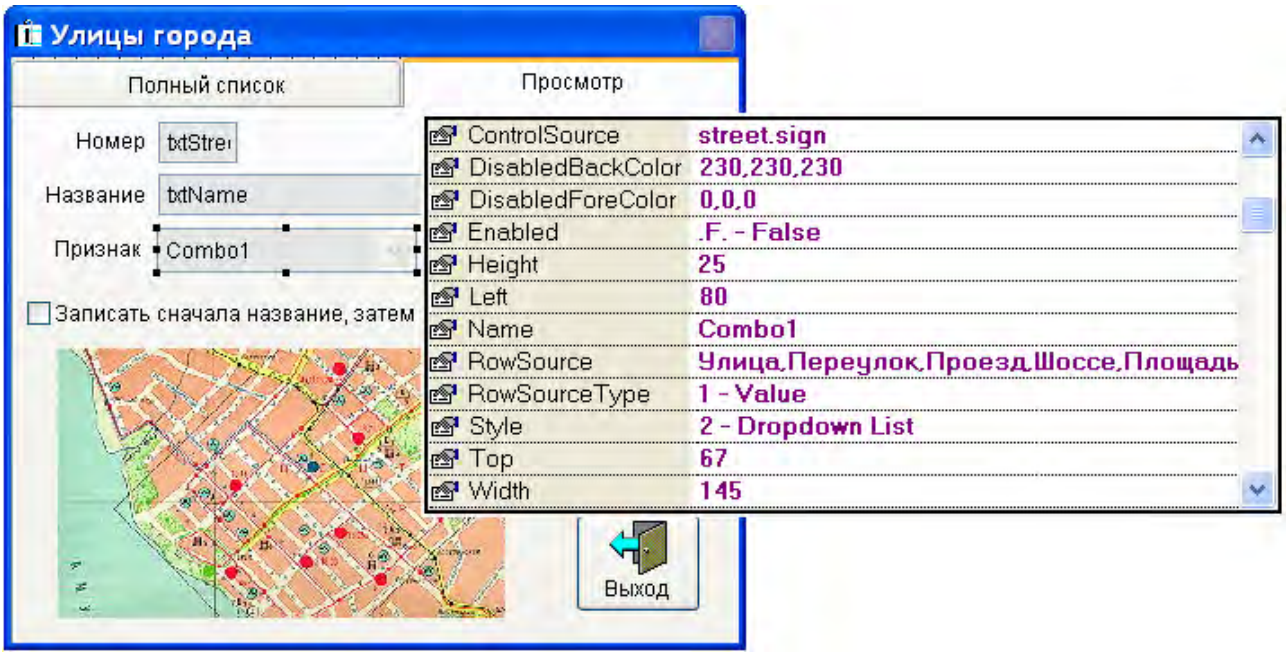


Рис. 4.17. Значения свойств элемента Combo1 второй страницы формы

Рассмотрим процесс обработки ошибок, возникающих при одновременной корректировке одной записи несколькими пользователями в таблице ***Street***. В этом случае методика, предложенная в приведенном выше коде, даст ошибку с номером 1585. Ее следует перехватить и обработать. Фрагмент обработки имеет вид

```
* Обработка ошибки с кодом 1585
lnMsgResult=MESSAGEBOX('Пока Вы занимались корректировкой, '+;
          'эту запись уже кто-то изменил. '+;
          'Записать Ваши изменения поверх?',20,' Внимание!')
IF lnMsgResult=6      && Кнопка Да
   * Сбросить буфер на диск
   =TABLEUPDATE(.T.,.T.)
ELSE
   * Отказ от записи на диск
   =TABLEREVERT()
ENDIF
```

Процедура перехвата носит название ***ErrorHnd*** и находится в процедурном файле ***FileProc***. Полный текст этого файла приведен в разд. 7.

4.3.4. Создание формы поиска здания

Технология создания многопользовательских форм, описанная в предыдущем разделе, хорошо подходит для работы со справочниками. Справочник – небольшая таблица. Все строки ее несложно отобразить на экране и выбрать требуемую. Для работы с основными таблицами такая технология не пригодна. Попробуйте найти нужную фамилию в телефонной книге, зная только номер телефона и район города. Все ваши иллюзии по поводу быстрого решения этой задачи очень скоро рассеются. Круг объектов, которые должны быть найдены, необходимо на первом этапе значительно сузить. Для этой цели предназначена форма ***Search*** (рис. 4.18). По существу, это построитель запроса к базе данных.

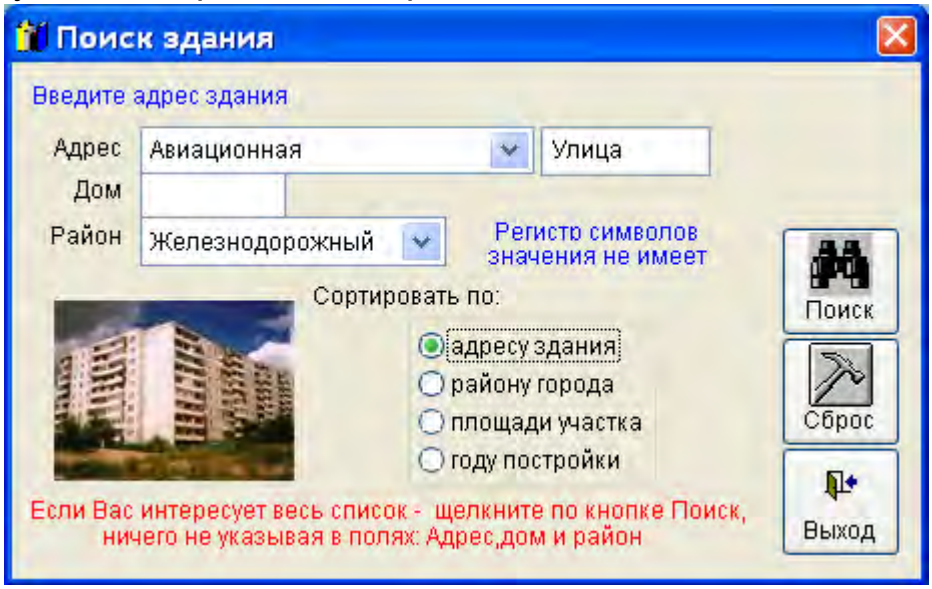


Рис. 4.18. Формирование запроса к базе данных

В этом случае (рис. 4.18) будет сформирован запрос, результатом которого станет выборка, содержащая все здания участка Авиационной улицы (номер 23), находящиеся только в железнодорожном районе (номер 2) города. Приведем текст запроса на языке SQL (Structured Query Language – язык структурированных запросов), который будет сформирован в результате работы формы ***Search***. Visual FoxPro «говорит» на своем диалекте SQL. Этот диалект отличается от стандарта примерно так же, как отличается английский от американского английского.

```
SELECT Building.Street,Street.Name,Street.Sign,Street.First,;
       Building.House,Building.District,District.Area,;
       Building.Land,Building.Year,Building.Material,Wall.Wall,;
       Building.Comment,Building.Wear,Building.Cost,;
       Building.Line,Building.Square,Building.Picture,;
       Building.Kind,Building.Elevator;
         FROM Building,Street,District,Wall;
         INTO TABLE C:\WINNT\TEMP\cBuilding;
           WHERE Building.Street=Street.Street;
               AND Building.District=District.District;
               AND Building.Material=Wall.Material;
               AND Building.Street=23 AND Building.District=2;
           ORDER BY Name,Sign,House
```

Язык запросов *SQL*, официально признан стандартизованным языком в ANSI (American National Standards Institute – Американский национальный институт стандартов). С помощью SQL очень удобно разрабатывать приложения по работе с базами данных и формировать запросы из этих приложений. Временем появления языка SQL можно считать начало 1970-х гг., когда Е. Ф. Кодд опубликовал свои первые работы по реляционной алгебре.

Мы не будем подробно описывать язык структурированных запросов. Этот раздел – всего лишь краткое введение, которое знакомит пользователя с основами SQL.

Чтобы употреблять SQL в приложениях, прежде всего необходимо понять, что он собой представляет и каким образом устроен. Как уже говорилось, это стандартизованный язык, который создан на основе английского. Язык SQL используется для создания баз данных и работы с ними. Он позволяет описывать сложные запросы при помощи очень небольшого кода. Чрезвычайно важной характеристикой SQL является стандартизованный подход к работе с базами данных. Итак, SQL – это стандартизованный язык доступа к данным, и он поддерживается большинством мощных систем баз данных, которые существуют в настоящее время на рынке, вклю-

чая Microsoft SQL-Server, ORACLE и DB2. Синтаксис и структура языка SQL определяются стандартом ANSI.

Язык SQL – богатый язык с большим количеством разнообразных команд. Однако все они относятся к одной из двух больших категорий: язык определения данных (DDL) и язык управления данными (DML).

- *Язык определения данных,* как ясно из названия, используется для определения баз и таблиц данных. С его помощью создается структура таблиц базы данных, определяются их «бизнес-правила» и связи.
- *Язык управления данными* позволяет дополнять, обновлять, формировать запросы и удалять данные из ваших таблиц.

На практике язык определения данных применяется не очень часто. Как правило, он используется для создания первоначальной структуры базы данных и изредка – для ее изменений. Гораздо чаще употребляется язык управления данными, и это вполне естественно: ведь предназначением любой базы данных является именно работа с данными в рамках своей структуры.

Как вы уже знаете, создавать таблицы и управлять ими можно с помощью навигационных команд языка Visual FoxPro. Так, команда ***Replace*** позволяет обновлять данные, а ***Copy*** и ***Seek*** – запрашивать их из таблиц. Это достаточно мощные команды, но для их оптимального использования требуется глубокое и детальное знание таблиц. Также нужны значительное время и усилия, которые необходимо затратить при разработке приложений. Здесь и проявляются преимущества SQL. Этот язык построен так, что позволяет обрабатывать данные целыми наборами. Поэтому при грамотном подходе можно с помощью всего нескольких строчек кода создавать оптимизированные запросы. Еще одним достоинством SQL является его совместимость. При работе с базами в архитектуре клиент-сервер ваши запросы могут поддерживаться самыми различными платформами, а не только Visual FoxPro. Наконец, последнее преимущество – это естественный синтаксис, близкий к синтаксису английского языка. Более того, логика SQL соответствует той логике, которую можно ожидать при работе с базами данных. Итак, преимущества языка SQL по сравнению с навигационным инструментарием Visual FoxPro следующие:

- способность работать с набором записей при создании оптимизированных запросов;
- совместимость со множеством платформ;
- естественный синтаксис, близкий к синтаксису английского языка.

Для работы с основными таблицами базы данных ***Real Estate*** мы будем использовать преимущественно язык SQL. На рис. 4.19 приведена форма поиска здания ***Search***, расположенная в конструкторе форм.

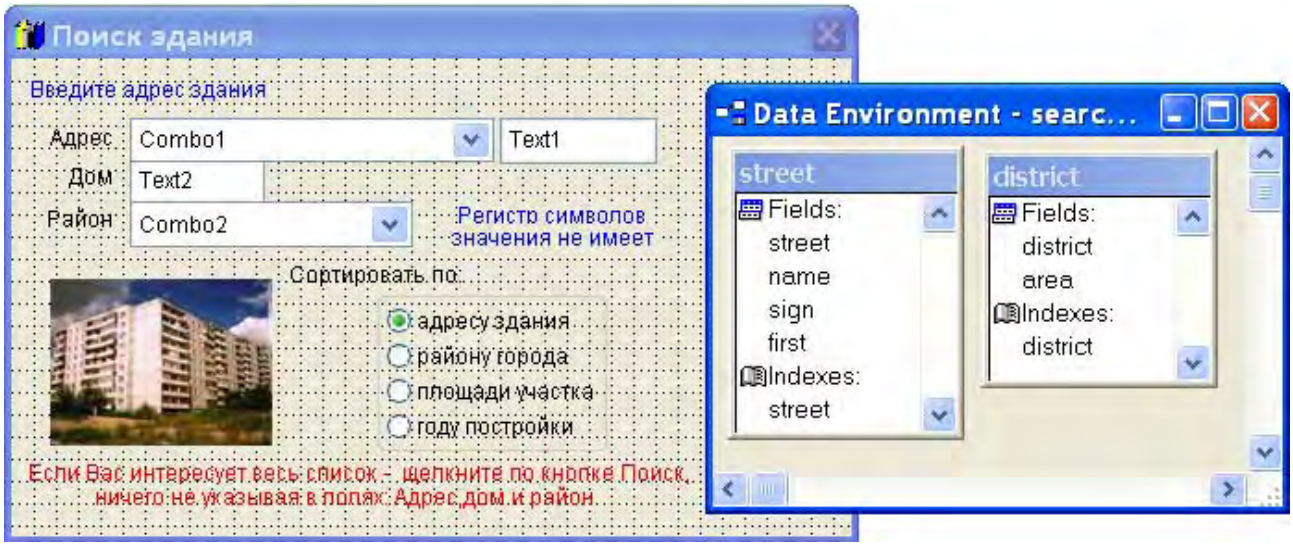


Рис. 4.19. Форма ***Search*** в конструкторе форм

Код события ***Init*** формы ***Search*** имеет вид

```
* Выбранные значения параметров поиска
PUBLIC SelectStreet, SelectDistrict,SelectSign,;
       SelectHouse,SortSelect
* SelectStreet -    номер выбранной улицы
* SelectDistrict - номер выбранного района
* SelectSign -     выбранный признак адреса
* SortSelect -     порядок сортировки записей в выборке
* Начальные значения
STORE 0 TO SelectStreet, SelectDistrict
STORE [ ] TO SelectSign, SelectHouse
SortSelect=1
```

Код события ***Destroy*** (“последний вздох”) формы ***Search***:

```
* Убрать глобальные переменные из памяти
RELEASE SelectStreet,SelectSign,SelectHouse,;
        SelectDistrict,SortSelect
* Закрыть временную таблицу-выборку
IF USED ('cBuilding')
   USE IN cBuilding
ENDIF
```

Код события ***InteractiveChange*** поля со списком ***Combo1***:

```
* Обновление признака адреса в поле Text1
* Взять значение в поле Sign таблицы Street
ThisForm.Text1.Value=Street.Sign
* Перерисовать объект Text1
ThisForm.Text1.Refresh
```

Код события ***Click*** кнопки ***Сброс***:

```
*- Кнопка Сброс
* Сброс ранее введенных значений
STORE 0 TO SelectStreet, SelectDistrict
STORE [ ] TO SelectSign, SelectHouse
SortSelect=1
* Перерисовать форму
ThisForm.Refresh
```

Код события ***Click*** кнопки ***Поиск***:

```
*- Кнопка Поиск
Wait 'Ждите! Ваш запрос обрабатывает сервер.' WINDOW NOWAIT
* Номер улицы
IF SelectStreet=0
   * Значение символьной переменной - пустая строка
   * К оператору SELECT ничего добавлено не будет
   Sstreet=[]
ELSE
   * Строка в квадратных скобках будет добавлена
   * к оператору SELECT как значение &Sstreet
   * Обратите внимание на знак & (амперсанд)
   Sstreet=[and Building.Street=SelectStreet]
ENDIF
* Номер дома
* Уберем концевые пробелы из введенного номера дома
SelectHouse=ALLTRIM(SelectHouse)
IF LEN(SelectHouse)=0
   Shouse=[]
ELSE
   * Строка в квадратных скобках будет добавлена
   * к оператору SELECT как значение &Shouse
   Shouse=[and Building.House==SelectHouse]
   * Знак == означает: в точности равно.
   * Если поставить просто знак равенства
   * Будут найдены все здания, номера которых
   * начинаются с введенных символов.
ENDIF
* Район
IF SelectDistrict=0
   Sdistrict=[]
ELSE
   * Строка в квадратных скобках будет добавлена
   * к оператору SELECT как значение &Sdistrict
   Sdistrict=[and Building.District=SelectDistrict]
ENDIF
* Сортировка
* Строка в квадратных скобках будет добавлена
```

```
* к оператору SELECT как значение &Ssort
DO CASE
   CASE SortSelect=1
        * По адресу здания
        Ssort=[Name,Sign,House]
   CASE SortSelect=2
        * По району города
        Ssort=[Area]
   CASE SortSelect=3
        * По площади участка
        Ssort=[Land]
   CASE SortSelect=4
        * По году постройки
        Ssort=[Year]
ENDCASE
* Выполнение запроса
SELECT Building.Street,Street.Name,Street.Sign,Street.First,;
       Building.House,Building.District,District.Area,;
       Building.Land,Building.Year,Building.Material,Wall.Wall,;
       Building.Comment,Building.Wear,Building.Cost,;
       Building.Line,Building.Square,Building.Picture,;
       Building.Kind,Building.Elevator;
         FROM Building,Street,District,Wall;
         INTO TABLE C:\WINNT\TEMP\cBuilding;
           WHERE Building.Street=Street.Street;
               AND Building.District=District.District;
               AND Building.Material=Wall.Material;
               &Sstreet &Shouse &Sdistrict;
           ORDER BY &Ssort
SELECT cBuilding
* Во временной выборке находится информация по зданиям,
* отвечающим условиям составленного запроса.
* Запись содержит полный набор полей из всех таблиц:
* Building, Street, District и Wall
* Полученную выборку можно конвертировать в Excel, Access и др.
* без дополнительной обработки
WAIT 'Готово!' WINDOW NOWAIT
IF RECCOUNT()=0
   =MESSAGEBOX('Зданий, отвечающих условиям вашего запроса, '+;
           'в базе нет. Повторите запрос, изменив требования.';
           ,48,'Внимание')
   RETURN
ENDIF
* Временная таблица-выборка хранится в файлах
* cBuilding.dbf и cBuilding.fpt в папке C:\WINNT\TEMP
* Во втором хранится Memo-поле Building.Comment
* Запуск формы для работы со зданиями
DO FORM Building
```

4.3.5. Форма Building – форма для работы со зданиями

После получения таблицы-выборки, содержащей сведения по зданиям, попавшим в запрос, сконструируем форму для работы с этой таблицей. Это будет двухстраничная форма, на первой вкладке которой список зданий, а на второй – подробности по выбранному зданию. В окружении данных формы (***Data Environment***) разместим таблицу ***cBuilding*** и три таблицы-справочника ***Street*** (улицы), ***District*** (районы), ***Wall*** (материал стен). Для этого сделайте щелчок правой кнопки мыши в любом месте окна ***Form Designer***. Появится меню. Выберите в нем третий пункт ***Data Environment***. Еще один щелчок правой кнопкой, но уже в появившемся окне ***Data Environment*** активизирует очередное меню. Выберите в нем первый пункт ***Add***. Появится окно ***Open***. Найдите в нем таблицу ***cBuilding***. Она находится папке ***WINNT\TEMP*** и появилась там в результате работы формы ***Search***. Аналогичным образом добавьте в окружение данных таблицы ***Street***, ***District*** и ***Wall***.

Добавить объект ***Page Frame*** не сложно. Откройте панель ***Form Controls*** (элементы управления формы). Она показана на рис. 4.9. Если панель отсутствует на экране – выберите в главном меню Visual FoxPro пункт ***View***, а в открывшемся подменю пункт ***Toolbars***. Откроется окно ***Toolbars***. Сделайте отметку напротив названия панели - ***Form Controls*** и щелкните кнопку ***OK***.

Выберите на панели значок ***Page Frame***, а в нужном месте активной области формы при помощи левой кнопки мыши отведите место для этого объекта. Форма ***Building*** с активной первой страницей и окружением данных в конструкторе форм показана на рис. 4.20.

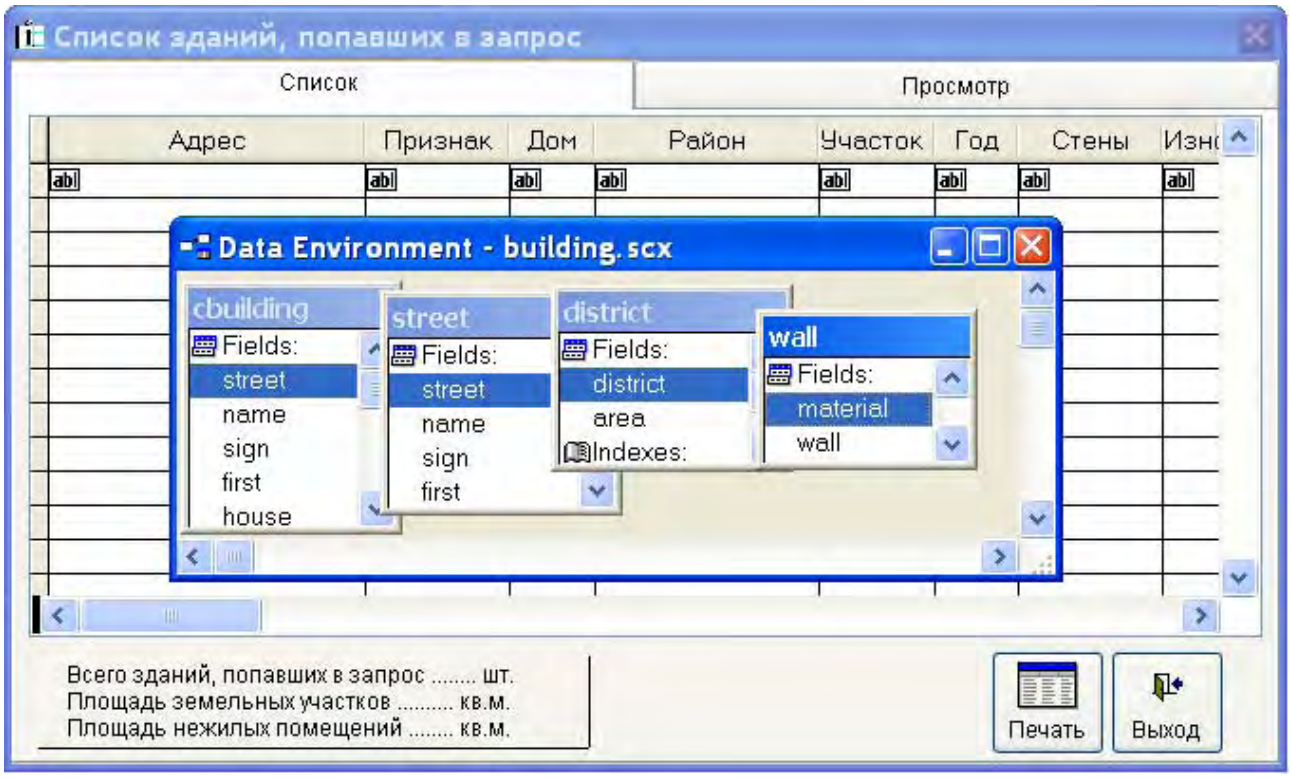


Рис. 4.20. Форма ***Building*** в конструкторе форм

В формах, рассмотренных ранее (***Employee***, ***Street, District, Wall***) для отображения нескольких записей мы использовали ***List Box*** (поле со списком). У этого объекта один существенный недостаток – отсутствие горизонтальной линейки прокрутки. Следовательно, увидеть в списке можно лишь те колонки, которые поместились в форме. Разместим на первой странице формы альтернативу объекту ***List Box***. Это ***Grid*** (сетка). Возможности ***Grid*** впечатляют. Вы можете установить любой ячейке сетки, связанной с таблицей, свои индивидуальные свойства. Да и общие свойства объекта ***Grid*** отличаются большим разнообразием. Изменением этих свойств мы сможем заставить ***Grid*** заменить, да еще и на более высоком уровне объект ***List Box***. Используем для этого ***Builder*** (построитель).

Выберите на панели ***Form Controls*** (элементы управления формы) значок  (Grid), а в нужном месте активной области формы при помощи левой кнопки мыши отведите место для этого объекта. Для запуска построителя сделайте по объекту ***Grid1*** щелчок правой кнопки мыши. Появится меню. Выберите в нем пятый пункт – ***Builder*** (рис. 4.21). Откроется диалоговое окно ***Grid Builder***. В нем четыре вкладки. Нам необходимо заполнить три из них (***Grid Items***, ***Style*** и ***Layout***).

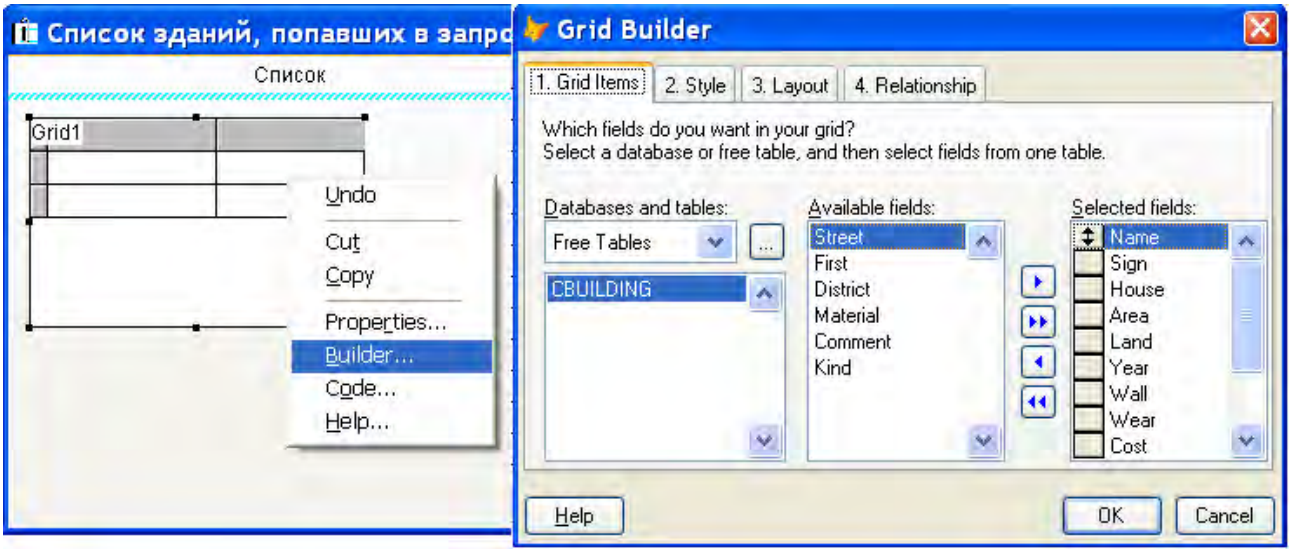


Рис. 4.21. Запуск построителя ***Grid Builder***

При помощи первой вкладки определяют набор полей, которые будут размещены в сетке, и последовательность их размещения. Не нажимайте клавишу ***Enter***. Это приведет к преждевременному завершению работы построителя. Придется все начинать сначала. Вторая и третья вкладки показаны на рис. 4.22.

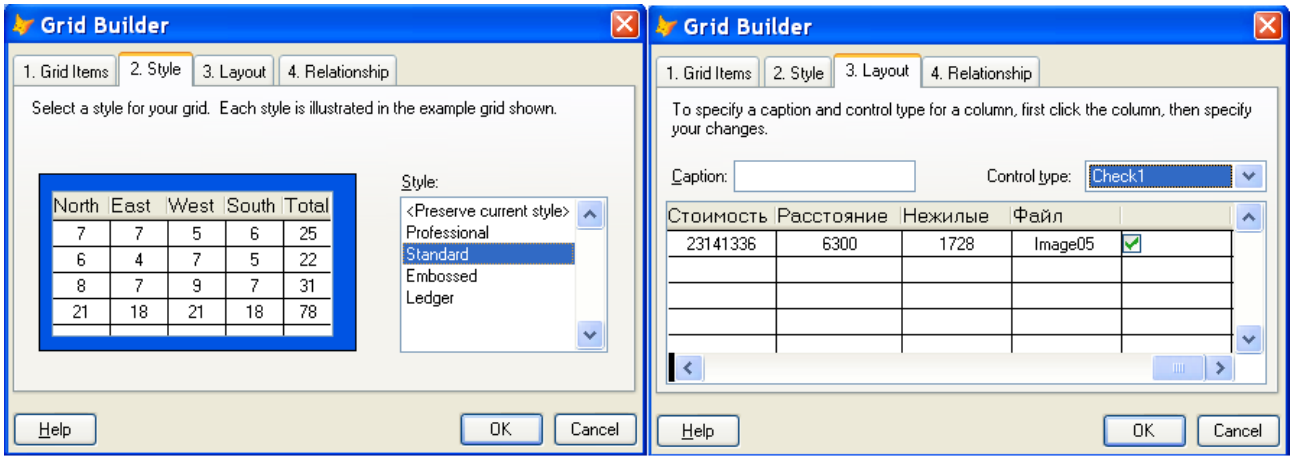


Рис. 4.22. Продолжение работы с построителем ***Grid Builder***

Вторая вкладка ***Style*** – стиль оформления. Выберите стандартный. Третья – надписи колонок и тип их содержимого. Обратите внимание на последнюю колонку ***Elevator*** (наличие лифта в здании). На рисунке она еще не имеет надписи. Если поставить в поле ***Control type*** тип ***Check Box***, то вместо малопонятных пользователю значений .T. или .F. в колонке будет отображаться красивый значок с птичкой внутри ☑, если лифт в здании есть.

Для завершения работы построителя сделайте щелчок по кнопке ***OK***. Объект ***Grid1*** появится на первой странице формы в законченном виде. Откройте окно ***Properties*** этого объекта. Нам необходимо изменить значения некоторых свойств: сделать сетку доступной только для чтения, изменить вид курсора и разрешить пользователю выделять строку только целиком. Значения этих свойств приведены в табл. 4.2.

Таблица 4.2

Свойства объекта *Grid*

Номер	Название свойства	Значение
1	HightLightStyle	2 – Current row highlighting enable with visual persistence
2	Mouse Pointer	1 - Arrow
3	Read Only	.T. -True
4	AllowCellSelection	.F. -False
5	RecordMark	.T. - True

В начале этой главы мы научилисть делать недоступными пункты меню приложения в соответствии с правилами разграничения доступа (ПРД), действующими на предприятии. Ниже приведен код, позволяющий распространить ПРД и на другие объекты приложения.

Код события ***Activate*** формы ***Building***:

```
* Обработка прав доступа к кнопкам Печать и Квартиры
* Гашение кнопки Печать
IF WordExcel=.F.
   * Доступ к просмотру отчетов запрещен
   ThisForm.PageFrame1.Page1.Command2.Enabled=.F.
ENDIF
* Гашение кнопки Квартиры
IF WorkFlats=.F.
   * Просмотр информации по квартирам запрещен
   ThisForm.PageFrame1.Page2.Command2.Enabled=.F.
ENDIF
```

Код события ***Init*** формы ***Building***:

```
* Глобальные переменные для сохранения параметров адреса
* у выбранного здания. Необходимы в случае корректировки адреса
PUBLIC SelectStreetAddress,SelectHouseAddress,IndBuilding
* Номер улицы выбранного здания
SelectStreetAddress=0
* Номер дома выбранного здания
SelectHouseAddress=[]
* Признак выбора здания для просмотра
IndBuilding=0
```

Код события ***Destroy*** формы ***Building***:

```
* Освободить глобальные переменные
RELEASE SelectStreetAddress,SelectHouseAddress,IndBuilding
```

На первой странице формы размещены три надписи. Их имена ***Label11***, ***Label12*** и ***Label13***. Возьмите на вооружение методику замены заголовка надписи на реальные цифры количества зданий, попавших в запрос. Смысл этой методики вы увидите из текста события ***Activate***.

Код события ***Activate*** первой страницы формы:

```
* Сброс значения признака выбора здания для просмотра
IndBuilding=0
PRIVATE CountBuilding,LandAll,CountUnder
SELECT cBuilding
* Подсчет числа зданий попавших в запрос
COUNT TO CountBuilding ALL
* Подсчет общей площади земельных участков
SUM cBuilding.Land TO LandAll
* Подсчет общей площади нежилых помещений
SUM cBuilding.Square TO CountUnder
```

```
* Отображение данных по выборке на первой странице формы
ThisForm.PageFrame1.Page1.Label11.Caption=;
    [В запрос попало зданий  ]+;
    ALLTRIM(STR(CountBuilding,4))+[ шт.]
ThisForm.PageFrame1.Page1.Label12.Caption=;
    [Общая площадь участков ]+ALLTRIM(STR(LandAll,13,1))+[ кв.м.]
ThisForm.PageFrame1.Page1.Label13.Caption=;
    [Площадь нежилых помещений ]+;
    ALLTRIM(STR(CountUnder,13,1))+[ кв.м.]
* Перерисовать первую страницу формы
ThisForm.PageFrame1.Page1.Refresh
```

Код события ***Init*** сетки ***Grid1***:

```
* Выполнить раскрашивание строк Сетки через одну
* Цвет фона
THISFORM.PageFrame1.Page1.grid1.SETALL("dynamicBackColor",;
  "IIF(MOD(RECNO( ), 2)=0, RGB(255,255,255), RGB(234,234,234))";
  , "Column")
* Цвет текста
THISFORM.PageFrame1.Page1.grid1.SETALL("dynamicForeColor",;
  "IIF(MOD(RECNO( ), 2)=0, RGB(0,0,0), RGB(255,0,0))", "Column")
```

Код события ***DblClick*** сетки ***Grid1***:

```
* Здание для просмотра выбрано
IndBuilding=1
* Сделать активной вторую страницу
ThisForm.PageFrame1.ActivePage=2
```

Код события ***Click*** сетки ***Grid1***:

```
* Здание для просмотра выбрано
IndBuilding=1
```

Код события ***Activate*** второй страницы формы:

```
* Выбрано ли здание для просмотра?
IF IndBuilding=0
     * Выбор не сделан
     =MESSAGEBOX('Ни одно здание не выбрано! ',;
                 48,' Внимание!')
     * Вернуться на первую страницу формы
     THISFORM.PAGEFRAME1.ACTIVEPAGE=1
ENDIF
* Номер улицы выбранного здания
SelectStreetAddress=cBuilding.Street
* Номер дома выбранного здания
SelectHouseAddress=cBuilding.House
* Делаем недоступной кнопку "Записать"
ThisForm.PageFrame1.page2.Command4.Enabled=.F.
* Делаем недоступным Класс Building1
ThisForm.PageFrame1.page2.Building1.Enabled=.F.
* Определение местоположения фотографии здания
```

```
SET EXACT ON
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
   RealDirectory=[\]
ENDIF
SET EXACT OFF
* Есть фотография - значение переменной FileName
FileName=DISK+RealDirectory+[Picture\]+;
ALLTRIM(cBuilding.Picture)+[.jpg]
* Нет фотографии - значение переменной NoFileName
NoFileName=DISK+RealDirectory+[Picture\]+[NoFoto.jpg]
IF FILE(FileName)
   * Если фотография имеется в папке Picture
   * Разместить ее на месте картинки IMAGE1
   THIS.IMAGE1.PICTURE=FileName
ELSE
   * Если нет - взять картинку NoFoto.jpg
   THIS.IMAGE1.PICTURE=NoFileName
ENDIF
* Перерисовать страницу формы
THIS.Refresh
```

Вторая страница формы ***Building*** (рис. 4.23) содержит информацию по выбранному для просмотра зданию. Основное место отведено под класс ***Building1***. На рисунке он выделен. Этот же визуальный класс используется также в форме ***AddBuild*** (Занесение нового здания).

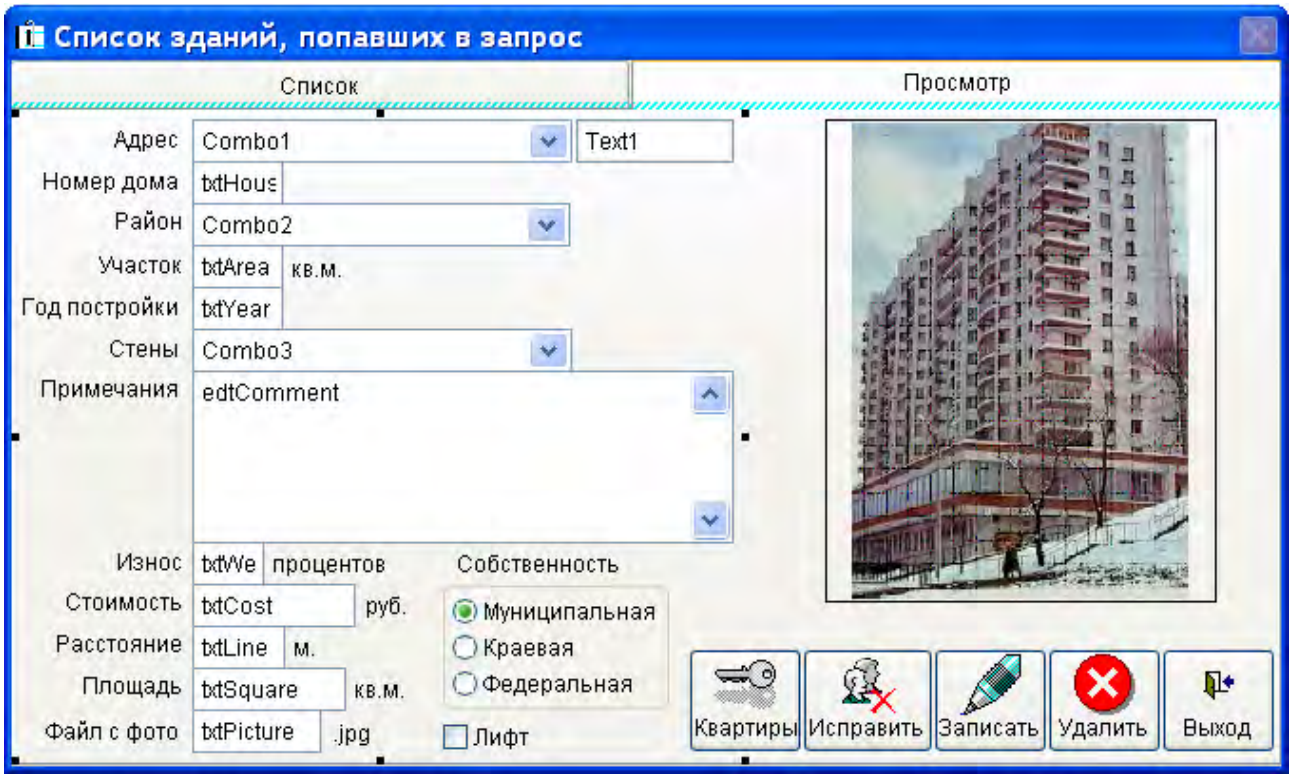


Рис. 4.23. Вторая страница формы ***Building***

Обратите внимание на событие, которое наступит при потере фокуса объектом ***txtPicture*** класса ***Building1***. Немедленно после занесения имени файла, содержащего фотографию здания, его изображение появится на месте объекта ***Image1***.

Код события ***LostFocus*** объекта ***txtPicture*** класса ***Building1***:

```
* Определение местоположения фотографии здания
SET EXACT ON
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
   RealDirectory=[\]
ENDIF
SET EXACT OFF
* Есть фотография - значение переменной FileName
FileName=DISK+RealDirectory+[Picture\]+;
ALLTRIM(cBuilding.Picture)+[.jpg]
* Нет фотографии - значение переменной NoFileName
NoFileName=DISK+RealDirectory+[Picture\]+[NoFoto.jpg]
IF FILE(FileName)
   * Если фотография имеется в папке Picture
   * Разместить ее на месте картинки IMAGE1
   THISFORM.PAGEFRAME1.PAGE2.IMAGE1.PICTURE=FileName
ELSE
   * Если нет - взять картинку NoFoto.jpg
   THISFORM.PAGEFRAME1.PAGE2.IMAGE1.PICTURE=NoFileName
ENDIF
```

Если пользователь сделает щелчок левой кнопкой мыши по фотографии здания, то сработает код события ***Click*** объекта ***Image1***. Этот код создавался на протяжении десяти лет, пополняясь с появлением новых операционных систем. Одни стандартные просмотрщики изображений заменялись на другие, только один ***Paint*** оставался неизменным. Именно на нем я и остановил свой выбор. Будем надеяться, что Windows Vista, которую Microsoft обещает выпустить в 2007 г., не расстанется с ***Paint*** и код будет работать.

Код события ***Click*** объекта ***Image1***:

```
IF LEN(ALLTRIM(cBuilding.Picture))=0
    =MESSAGEBOX('В архиве отсутствует фото этого здания',;
          48,' Внимание')
    RETURN
ENDIF
SET EXACT ON
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
```

```
   RealDirectory=[\]
ENDIF
SET EXACT OFF
FileName1=DISK+RealDirectory+[PICTURE\]+;
ALLTRIM(cBuilding.Picture)+[.jpg]
* Поиск просмотрщика картинок
FileVeiwver=[ ]
* WINDOWS 95
IF FILE ('C:\WINDOWS\WANGIMG.EXE')
   FileVeiwver='C:\WINDOWS\WANGIMG.EXE '
ENDIF
* WINDOWS 98
IF FILE ('C:\WINDOWS\KODAKIMG.EXE')
   FileVeiwver='C:\WINDOWS\KODAKIMG.EXE '
ENDIF
* WINDOWS NT 4.0 WorkStation
IF FILE ('C:\Program Files\Windows
NT\Accessories\ImageVue\wangimg.exe')
   FileVeiwver='C:\Program Files\Windows
NT\Accessories\ImageVue\wangimg.exe'
ENDIF
* WINDOWS 2000
IF FILE ('C:\Program Files\Windows
NT\Accessories\ImageVue\kodakimg.exe')
   FileVeiwver='C:\Program Files\Windows
NT\Accessories\ImageVue\kodakimg.exe'
ENDIF
* WINDOWS XP и другие ОС Выбираем PAINT
IF FILE ('C:\Windows\System32\Mspaint.exe')
   FileVeiwver='C:\Windows\System32\Mspaint.exe'
ENDIF
IF FileVeiwver=[ ]
    =MESSAGEBOX('На Вашем компьютере отсутствует'+;
     ' просмотрщик изображений!',48,' Внимание')
    RETURN
ENDIF
IF FILE('&FileName1')
   * Запуск просмотрщика изображений
   RUN /N1 &FileVeiwver &FileName1
ELSE
      =MESSAGEBOX('Пропал файл с изображением этого здания!',;
          48,' Внимание')
ENDIF
```

Код события ***Click*** кнопки ***Исправить***:

```
*- Кнопка Исправить
* Первую страницу делаем недоступной
ThisForm.PageFrame1.page1.Enabled=.F.
* Меняем заголовок второй страницы "Просмотр" на "Корректировка"
ThisForm.PageFrame1.page2.Caption=[Корректировка]
```

```
* Делаем доступной кнопку "Записать"
ThisForm.PageFrame1.page2.Command4.Enabled=.T.
* Делаем недоступной кнопку "Удалить"
ThisForm.PageFrame1.page2.Command1.Enabled=.F.
* Делаем недоступной кнопку "Квартиры"
ThisForm.PageFrame1.page2.Command2.Enabled=.F.
* Меняем заголовок кнопки "Выход" на "Отказ"
ThisForm.PageFrame1.page2.Command3.Caption=[Отказ]
* Делаем доступным Класс Building1
ThisForm.PageFrame1.page2.Building1.Enabled=.T.
```

Код события ***Click*** кнопки ***Записать***:

```
*- Кнопка Записать
lnMsgResult=MESSAGEBOX('Сейчас результаты корректировки '+;
               'будут записаны на диск.',52,'Подтвердите!')
IF lnMsgResult=6
   SELECT cBuilding
   SCATTER MEMO MEMVAR
  * Корректировка в основной базе
  IF .NOT. USED ('Building')
     USE Building IN 0
  ENDIF
  SELECT Building
  * Выбираем индекс для быстрого поиска
  SET ORDER TO TAG ADDRESS
  * Индексированный поиск здания в таблице
  * SelectStreetAddress - номер улицы, на которой стоит здание
  * SelectHouseAddress - номер дома
  SEEK STR(SelectStreetAddress)+SelectHouseAddress
  * Если записей мало, можно обойтись простым поиском
  *  SET EXACT ON
  *  LOCATE FOR Street=cBuilding.Street AND House=cBuilding.House
  *  SET EXACT OFF
  IF FOUND()
     GATHER MEMO MEMVAR
  ELSE
     =MESSAGEBOX('Нет доступа к таблице зданий '+;
       ' или в таблице отсутствует здание, данные которого '+;
       'Вы редактировали.',48,'А вот Вам и проблема!')
  ENDIF
  THISFORM.Release
ENDIF
```

Код события ***Click*** кнопки ***Удалить***:

```
lnMsgResult=MESSAGEBOX('Подтвердите!',52,'Удаление!')
IF lnMsgResult=6
```

```
  * Удаление в основной базе
  IF .NOT. USED ('Building')
     USE Building IN 0
  ENDIF
  SELECT Building
  * Выбираем индекс для быстрого поиска
  SET ORDER TO TAG ADDRESS
  * Индексированный поиск здания в таблице зданий
  * SelectStreetAddress - номер улицы, на которой стоит здание
  * SelectHouseAddress - номер дома
  SEEK STR(SelectStreetAddress)+SelectHouseAddress
  IF FOUND()
     * Если здание найдено - удалить его
     DELETE
  ENDIF
  THISFORM.Release
ENDIF
```

Код события ***Click*** кнопки ***Квартиры***:

```
*- Кнопка Квартиры
* Выборка всех квартир дома в C:\WINNT\TEMP\cFlat.dbf
SELECT Flat.Street,Flat.House,;
       Flat.Flat,Flat.Storey,Flat.rooms,;
       Flat.SquareFlat,Flat.Dwell,Flat.Branch,;
       Flat.Balcony,Flat.Height,Flat.Account;
         FROM Flat;
         WHERE Flat.Street=SelectStreetAddress;
              AND Flat.House==SelectHouseAddress;
         INTO TABLE 'C:\WINNT\TEMP\cFlat.dbf';
         ORDER BY Flat
* Запуск формы для работы с квартирами
DO FORM Flat
```

4.3.6. Занесение нового здания

Новостройки и здания, переданные на баланс предприятия, находятся в ведении специального подразделения. С целью защиты информации от несанкционированного доступа, функции работы со зданиями в проекте ***Real Estate*** разделены на две группы: занесение нового здания и корректировка информации по уже существующим.

Для занесения нового здания предназначена форма ***AddBuild*** (рис. 4.24). Это одностраничная форма на основе класса ***Building***. В окружении данных этой формы только справочники ***Street***, ***District*** и ***Wall***. Таблица-выборка создается с одной пустой записью при наступлении события ***Load*** (загрузка формы). Буферизация не используется. При закрытии формы выборка удаляется.

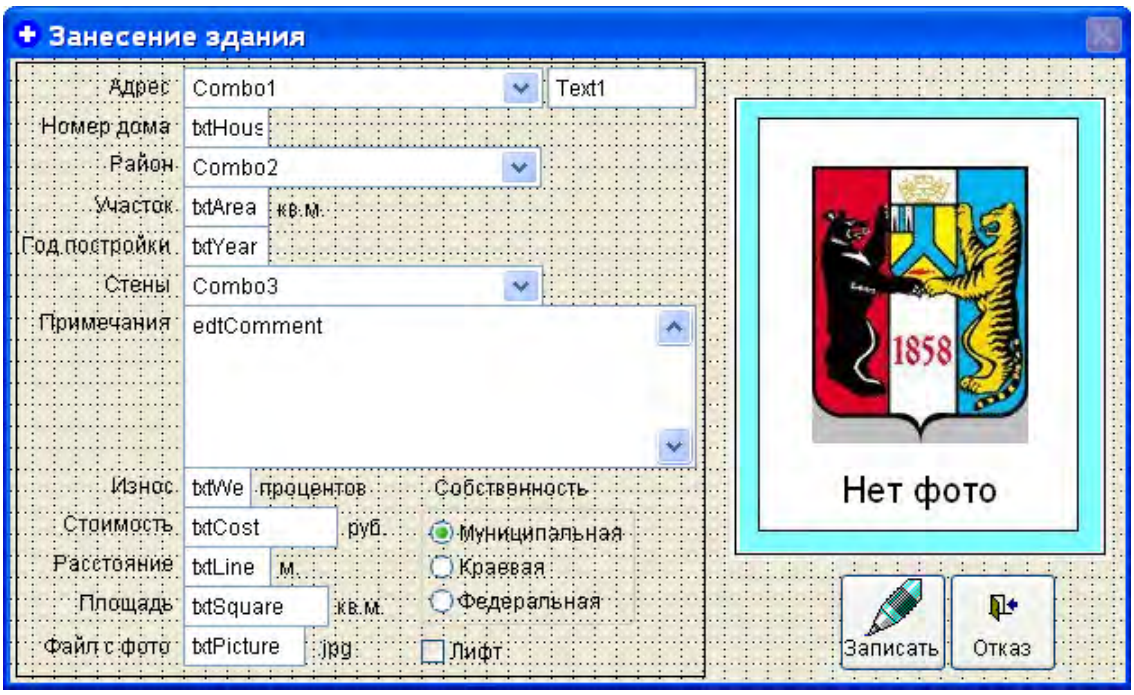


Рис. 4.24. Форма ***AddBuild*** в конструкторе форм

Код события ***Load*** формы ***AddBuild***:

```
* Если в папке C:\WINNT\TEMP\ остались файлы-выборки
* Закрыть таблицу, если она открыта
IF USED ('cBuilding')
   USE IN cBuilding
ENDIF
* Удаление таблицы
IF FILE ('C:\WINNT\TEMP\cBuilding.DBF')
   DELETE FILE 'C:\WINNT\TEMP\cBuilding.DBF'
ENDIF
* Удаление полей Memo
IF FILE ('C:\WINNT\TEMP\cBuilding.FPT')
   DELETE FILE 'C:\WINNT\TEMP\cBuilding.FPT'
ENDIF
* Открываем основную таблицу зданий
IF .NOT. USED ('Building')
   USE building IN 0
ENDIF
SELECT Building
* Копируем структуру во временную таблицу-выборку
COPY STRUCTURE TO C:\WINNT\TEMP\cBuilding.DBF
* Закрываем основную таблицу
USE
* Открываем временную таблицу-выборку
IF .NOT. USED ('cBuilding')
   USE 'C:\WINNT\TEMP\cBuilding' IN 0
ENDIF
```

```
SELECT cBuilding
* Добавляем пустую запись
APPEND BLANK
```

Код кнопки ***Записать*** формы ***AddBuild***:

```
*- Кнопка Записать
IF cBuilding.Street=0
  =MESSAGEBOX('Вы забыли ввести название улицы!',;
              48,'Ошибка!')
  ThisForm.Building1.Combo1.Setfocus
  RETURN
ENDIF
IF LEN(ALLTRIM(cBuilding.House))=0
  =MESSAGEBOX('Вы забыли ввести номер дома!',;
              48,'Ошибка!')
  ThisForm.Building1.TxtHouse.Setfocus
  RETURN
ENDIF
* Есть ли уже здание по такому адресу?
IF .NOT. USED ('Building')
   USE Building IN 0
ENDIF
SELECT Building
SET ORDER TO TAG ADDRESS
* Индексированный поиск здания в таблице
* cBuilding.Street - номер улицы, на которой стоит здание
* cBuilding.House - номер дома
SEEK STR(cBuilding.Street)+cBuilding.House
IF FOUND()
  =MESSAGEBOX('Здание по этому адресу уже есть!',;
              48,'Ошибка!')
  ThisForm.Building1.Combo1.Setfocus
  SELECT cBuilding
  RETURN
ENDIF
SET DELETED OFF
SEEK STR(cBuilding.Street)+cBuilding.House
SET DELETED ON
IF FOUND()
  =MESSAGEBOX('Здание по этому адресу уже есть среди '+;
              'удаленных! Необходимо выполнить PACK',;
              48,'Ошибка!')
  ThisForm.Building1.Combo1.Setfocus
  SELECT cBuilding
  RETURN
ENDIF
SELECT cBuilding
```

```
IF cBuilding.District=0
  =MESSAGEBOX('В каком районе расположено здание?',;
              48,'Ошибка!')
  ThisForm.Building1.Combo2.Setfocus
  RETURN
ENDIF
lnMsgResult=MESSAGEBOX('Сейчас данные о новом здании '+;
               'будут записаны в базу.',52,'Подтвердите!')
IF lnMsgResult=6
   SELECT cBuilding
   SCATTER MEMO MEMVAR
   * Запись в основную базу
   IF .NOT. USED ('Building')
      USE Building IN 0
   ENDIF
   SELECT Building
    APPEND BLANK
   GATHER MEMO MEMVAR
   THISFORM.Release
ENDIF
```

Перед записью в основную таблицу зданий выполняется проверка на наличие в ней объекта, имеющего такой же адрес. Поиск выполняется также и среди записей, помеченных на удаление. Visual FoxPro не производит физического удаления записей из таблиц. Это дает возможность вернуть ошибочно удаленную запись на место. Для снятия метки с записей выбранной таблицы, помеченных на удаление предназначена команда RECALL [Scope] [FOR lExpression1] [WHILE lExpression2].

Хранение в базе данных удаленной информации требуется не всегда. Существует два способа удаления отмеченных записей. Первый применяется ко всей базе данных целиком. Пусть это будет наша ***Real Estate***. Наберите в командном окне ***Commmand***:

```
OPEN DATABASE "c:\realestate\dbf\real estate.dbc" EXCLUSIVE
MODIFY DATABASE
```

В главном меню Visual FoxPro выберите ***Database*** и ***Clean up Database***. Второй применим к одной таблице. Пусть этой таблицей будет ***Building***. Наберите в командном окне ***Command***:

```
USE c:\realestate\dbf\building.dbf EXCLUSIVE
PACK
```

В обоих случаях база данных или таблица должны быть открыты в режиме Exclusive. Для выполнения операций упаковки объектов требуется исключительно монопольный доступ.

4.3.7. Форма Flat – работа с квартирами

Создадим форму, которая полностью обеспечивает все операции по работе с основной таблицей: добавление, удаление и корректировку записей, а также вызов форм продолжения каскада (Поиск – Здание – Квартира – Проживающие – Лицевой счет – Договор приватизации). Применение классов для ее создания не оправдано. Остановимся на уже известных нам Page Frame, Grid, Combo Box и Text Box.

Начинать работу с формой ***Flat*** можно только после создания временной таблицы-выборки C:\WINNT\TEMP\cFlat.dbf. Добавим ее в окружение данных этой формы. Для этого сделайте щелчок правой кнопки мыши в любом месте окна ***Form Designer***. Появится меню. Выберите в нем третий пункт ***Data Environment***. Еще один щелчок правой кнопкой, но уже в появившемся окне ***Data Environment*** активизирует очередное меню. Выберите в нем первый пункт ***Add***. Появится окно ***Open***. Найдите в нем таблицу ***cFlat***. Она находится папке ***WINNT\TEMP*** и появилась там в результате обработки события ***Click*** кнопки ***Квартиры*** второй страницы формы ***Building***. Вид первой страницы формы ***Flat*** в процессе работы программного комплекса показан на рис. 4.25. Вторая страница формы, но в конструкторе форм, показана на рис. 4.26.

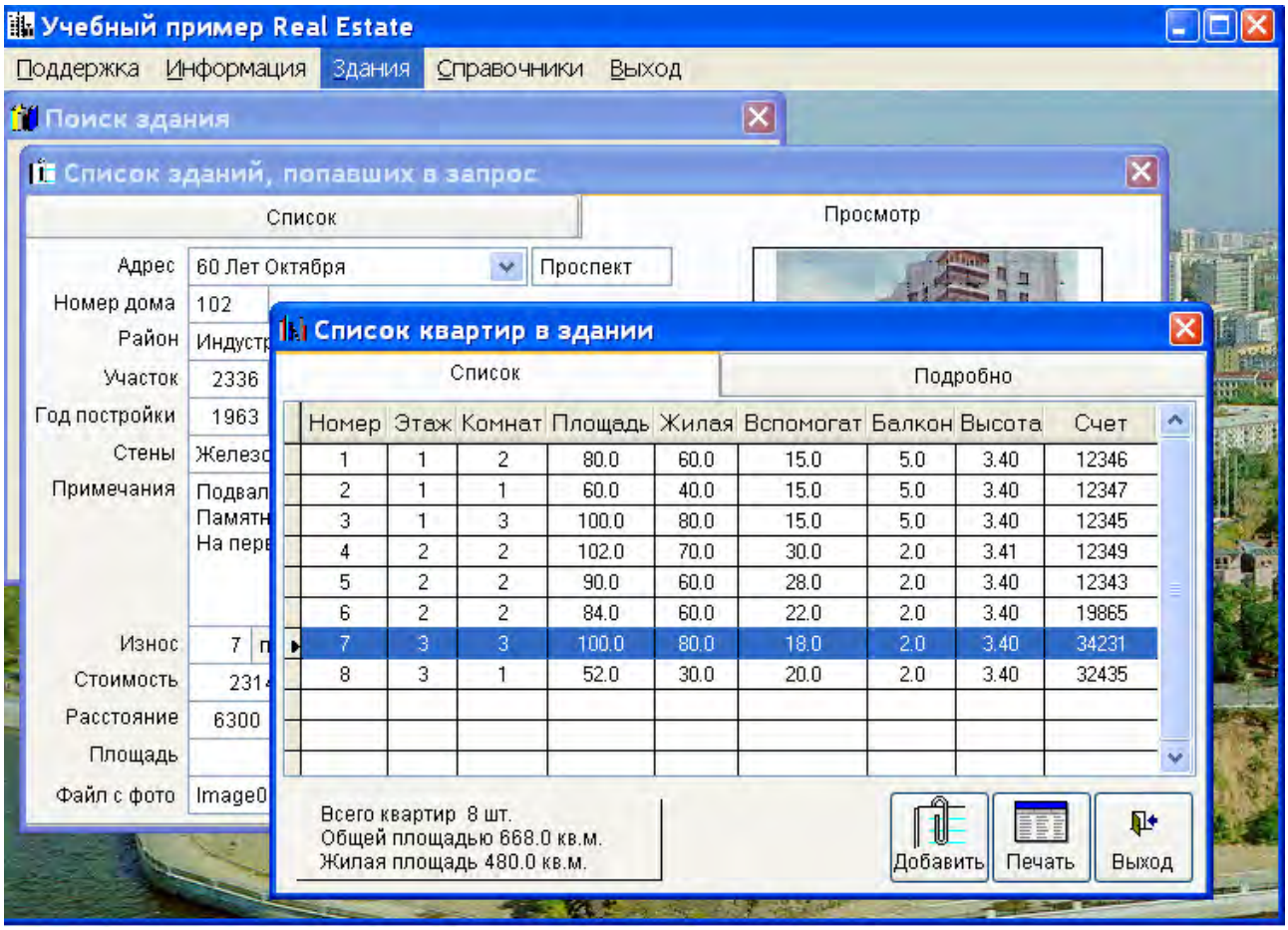


Рис. 4.25. Работа с квартирами в учебном комплексе Real Estate

Код события ***Init*** формы ***Flat***:

```
* Создание глобальных переменных
PUBLIC SelectFlat     && Номер выбранной квартиры
SelectFlat=0
PUBLIC SelectAccount  && Номер выбранного счета
SelectAccount=0
PUBLIC IndFlat        && Тип действия с квартирой
IndFlat=0
```

Код события ***Activate*** формы ***Flat***:

```
* Обработка прав доступа к объектам формы
* Гашение кнопки Печать
IF WordExcel=.F.
   ThisForm.PageFrame1.Page1.Command2.Enabled=.F.
ENDIF
* Гашение кнопки Счет
IF AccountWork=.F.
   ThisForm.PageFrame1.Page1.Command5.Enabled=.F.
ENDIF
```

Код события ***Activate*** первой страницы формы ***Flat***:

```
* Сброс признака выбора квартиры
IndFlat=0
PRIVATE CountFlat,AllSquareFlat,AllDwell
SELECT cFlat
* Подсчет количества квартир в здании
COUNT TO CountFlat ALL
* Подсчет общей площади квартир
SUM cFlat.SquareFlat TO AllSquareFlat
* Подсчет жилой площади квартир
SUM cFlat.Dwell TO AllWwell
* Отображение данных по выборке на первой странице формы
ThisForm.PageFrame1.Page1.Label11.Caption=;
    [Всего квартир  ]+ALLTRIM(STR(CountFlat,4))+[ шт.]
ThisForm.PageFrame1.Page1.Label12.Caption=;
    [Общей площадью ]+ALLTRIM(STR(AllSquareFlat,13,1))+[ кв.м.]
ThisForm.PageFrame1.Page1.Label13.Caption=;
    [Жилая площадь ]+ALLTRIM(STR(AllWwell,13,1))+[ кв.м.]
ThisForm.PageFrame1.Page1.Refresh
```

Код события ***Click*** кнопки ***Добавить*** первой страницы формы ***Flat***:

```
*- Кнопка Добавить
* Признак занесения данных по новой квартире
IndFlat=1
* Сделать активной вторую страницу формы
ThisForm.PageFrame1.ActivePage=2
```

Код события ***Activate*** второй страницы формы ***Flat***:

```
DO CASE
   CASE IndFlat=0
        * Выбор не сделан
        =MESSAGEBOX('Ни одна квартира для просмотра не выбрана! '+;
                    'Для выбора - двойной щелчок мышью.',48,'Ошибка!')
        ThisForm.PageFrame1.ActivePage=1
   CASE IndFlat=1
         * Добавить квартиру
         APPEND BLANK
         * Номер квартиры доступен
         ThisForm.PageFrame1.Page2.txtFlat.Enabled=.T.
         * Номер лицевого счета доступен
         ThisForm.PageFrame1.Page2.txtAccount.Enabled=.T.
         * Передать фокус на номер квартиры
         ThisForm.PageFrame1.Page2.txtFlat.SetFocus
         * Сделать кнопку Жильцы недоступной
         ThisForm.PageFrame1.Page2.Command2.Enabled=.F.
         * Сделать кнопку Удалить недоступной
         ThisForm.PageFrame1.Page2.Command4.Enabled=.F.
         * Сделать кнопку Счет недоступной
         ThisForm.PageFrame1.Page2.Command5.Enabled=.F.
         * Сделать кнопку Договор приватизации недоступной
         ThisForm.PageFrame1.Page2.Command6.Enabled=.F.
         THIS.Refresh
   CASE IndFlat=2
         * Просмотреть Квартиру
         * Номер квартиры недоступен
         ThisForm.PageFrame1.Page2.txtFlat.Enabled=.F.
         * Номер лицевого счета недоступен
         ThisForm.PageFrame1.Page2.txtAccount.Enabled=.F.
         * Обновить эту страницу формы
         THIS.Refresh
ENDCASE
```

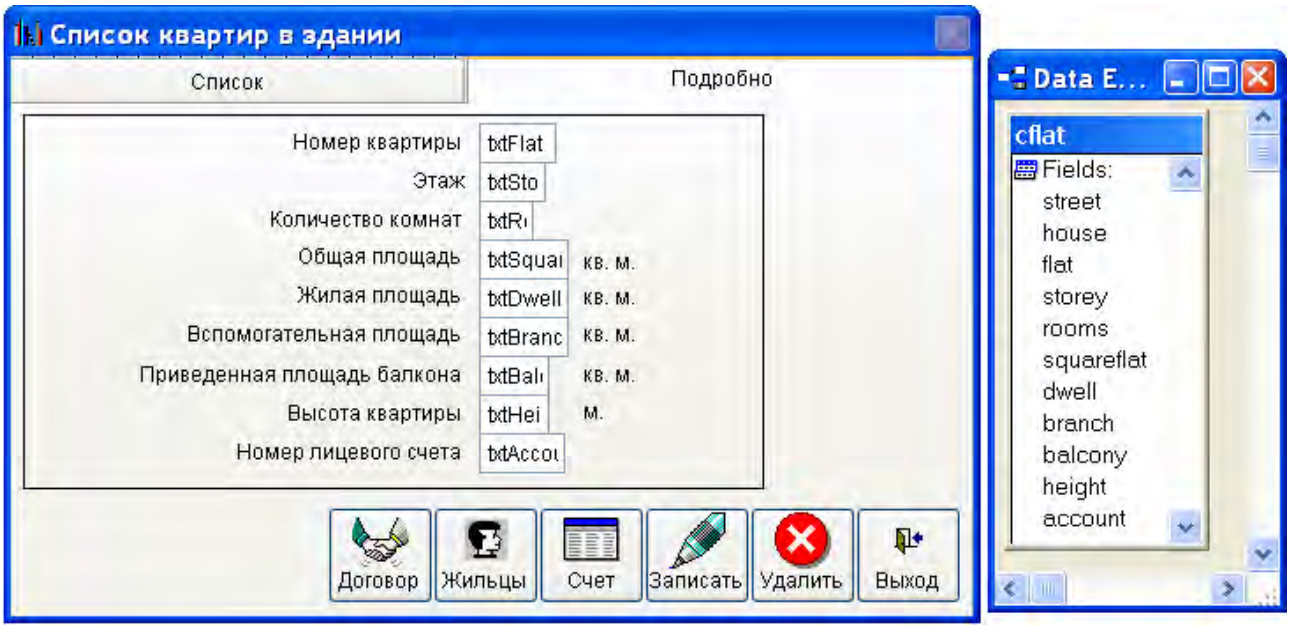


Рис. 4.26. Вторая страница формы ***Flat*** в конструкторе форм

Код события ***Click*** кнопки ***Записать*** второй страницы формы ***Flat***:

```
*- Кнопка записать
* Проверка введенных значений
IF cFlat.Flat=0
   =MESSAGEBOX('Вы забыли ввести номер квартиры!',48,'Ошибка!')
   ThisForm.PageFrame1.Page2.txtFlat.Setfocus
   RETURN
ENDIF
IF cFlat.SquareFlat=0
         =MESSAGEBOX('Вы забыли про общую площадь!',48,'Ошибка!')
         ThisForm.PageFrame1.Page2.txtSquareFlat.Setfocus
         RETURN
ENDIF
IF cFlat.Account=0
         =MESSAGEBOX('Введите номер лицевого счета',48,'Ошибка!')
         ThisForm.PageFrame1.Page2.txtAccount.Setfocus
         RETURN
ENDIF
DO CASE
   CASE IndFlat=1
      * Добавление новой квартиры
      * Есть ли уже квартира с таким номером
      IF .NOT. USED ('Flat')
         USE Flat
      ENDIF
      SELECT Flat
      SET ORDER TO TAG FlatID
      SEEK STR(SelectStreetAddress)+SelectHouseAddress+;
           STR(cFlat.Flat)
      IF FOUND()
         =MESSAGEBOX('Квартира с таким номером уже есть!',;
                     48,'Ошибка!')
         SELECT cFlat
         ThisForm.PageFrame1.Page2.txtFlat.Setfocus
         RETURN
      ENDIF
      SET DELETED OFF
      SEEK STR(SelectStreetAddress)+SelectHouseAddress+;
           STR(cFlat.Flat)
      SET DELETED ON
      IF FOUND()
         =MESSAGEBOX('Квартира с таким номером уже есть среди '+;
                     'удаленных!',48,'Ошибка!')
         SELECT cFLAT
         ThisForm.PageFrame1.Page2.txtFlat.Setfocus
         RETURN
      ENDIF
      * Есть ли уже квартира с таким лицевым счетом
      IF .NOT. USED ('Flat')
         USE Flat
      ENDIF
```

```
   SELECT Flat
   SET ORDER TO TAG Account
   SEEK cFlat.Account
   IF FOUND()
      =MESSAGEBOX('Квартира с таким номером счета уже есть!',;
            48,'Ошибка!')
      SELECT cFlat
      ThisForm.PageFrame1.Page2.txtAccount.Setfocus
      RETURN
   ENDIF
   SELECT cFlat
   lnMsgResult=MESSAGEBOX('Сейчас данные о новой квартире '+;
            'будут записаны в базу.',52,'Подтвердите!')
   IF lnMsgResult=6     && Кнопка Да
      SELECT cFlat
      SCATTER MEMO MEMVAR
      M.Street=SelectStreetAddress
      M.House=SelectHouseAddress
      * Запись в основную базу
      IF .NOT. USED ('Flat')
         USE Flat IN 0
      ENDIF
      SELECT Flat
      APPEND BLANK
      GATHER MEMO MEMVAR
      THISFORM.Release
   ENDIF
CASE IndFlat=2
   * Редактирование квартиры
   lnMsgResult=MESSAGEBOX('Сейчас результаты корректировки '+;
            'будут записаны на диск.',52,'Подтвердите!')
   IF lnMsgResult=6
      SELECT cFlat
      SCATTER MEMO MEMVAR
      * Корректировка в основной базе
      IF .NOT. USED ('Flat')
           USE Flat IN 0
      ENDIF
      SELECT Flat
      SET ORDER TO TAG FlatID
      SEEK STR(SelectStreetAddress)+SelectHouseAddress+;
           STR(cFlat.Flat)
      IF FOUND()
           GATHER MEMO MEMVAR
      ELSE
           =MESSAGEBOX('Нет доступа к таблице квартир, '+;
              'или в таблице отсутствует квартира, '+;
              ' данные которой Вы редактировали.';
              ,48,'А вот Вам и проблема!')
      ENDIF
      SELECT cFlat
      THISFORM.Release
```

```
      ENDIF
ENDCASE
```

Код события ***Click*** кнопки ***Удалить*** второй страницы формы ***Flat***:

```
*- Кнопка Удалить
lnMsgResult=MESSAGEBOX('Подтвердите!',52,'Удаление!')
IF lnMsgResult=6
  * Удаление в основной базе
  IF .NOT. USED ('Flat')
     USE Flat IN 0
  ENDIF
  SELECT Flat
  * Индексированный поиск
  SET ORDER TO TAG FlatID
  SEEK STR(SelectStreetAddress)+SelectHouseAddress+;
       STR(cFlat.Flat)
  IF FOUND()
     DELETE     && Удаление
  ELSE
     =MESSAGEBOX('Нет доступа к таблице квартир, '+;
                 'или в таблице отсутствует квартира, '+;
                 'данные которой Вы хотите удалить.';
                 ,48,'А вот Вам и проблема!')
  ENDIF
  THISFORM.Release
ENDIF
```

Код события ***Click*** кнопки ***Жильцы*** второй страницы формы ***Flat***:

```
*- Кнопка Жильцы
* Номер выбранной квартиры
SelectFlat=cFlat.Flat
* Остальные параметры адреса смотри:
* событие Activate второй страницы формы Building
* SelectStreetAddress - номер улицы
* SelectHouseAddress  - номер дома
SELECT * FROM Owners;
        WHERE Street=SelectStreetAddress;
              AND House=SelectHouseAddress;
              AND Flat=SelectFlat;
        INTO TABLE 'C:\WINNT\TEMP\cOwners.dbf';
        ORDER BY Number
SELECT cOwners
DO FORM Owners
```

Код события ***Click*** кнопки ***Счет*** второй страницы формы ***Flat***:

```
*- Кнопка Счет
* Номер выбранного лицевого счета
SelectAccount=cFlat.Account
```

```
SELECT * FROM Account;
         WHERE Account=SelectAccount;
         INTO TABLE 'C:\WINNT\TEMP\cAccount.dbf'
SELECT cAccount
DO FORM Account
```

4.3.8. Форма Account – лицевой счет

Отличительная особенность формы (рис. 4.27) заключается в работе с таблицей, содержащей одну единственную запись. Если выполняется заведение лицевого счета, то эта запись – с пустыми полями. За выбор режима работы формы отвечает код события ***Load***:

```
PUBLIC IndAccount
* Работа с таблицей-выборкой
SELECT cAccount
IF RECCOUNT()=0
   * Если запись в таблице отсутствует
   * Заведение лицевого счета
   IndAccount=1                    && Признак создания счета
   SCATTER MEMO MEMVAR
   M.Account=SelectAccount
   APPEND BLANK
   GATHER MEMO MEMVAR
ELSE
   * Корректировка лицевого счета
   IndAccount=2                     && Признак корректировки
ENDIF
```

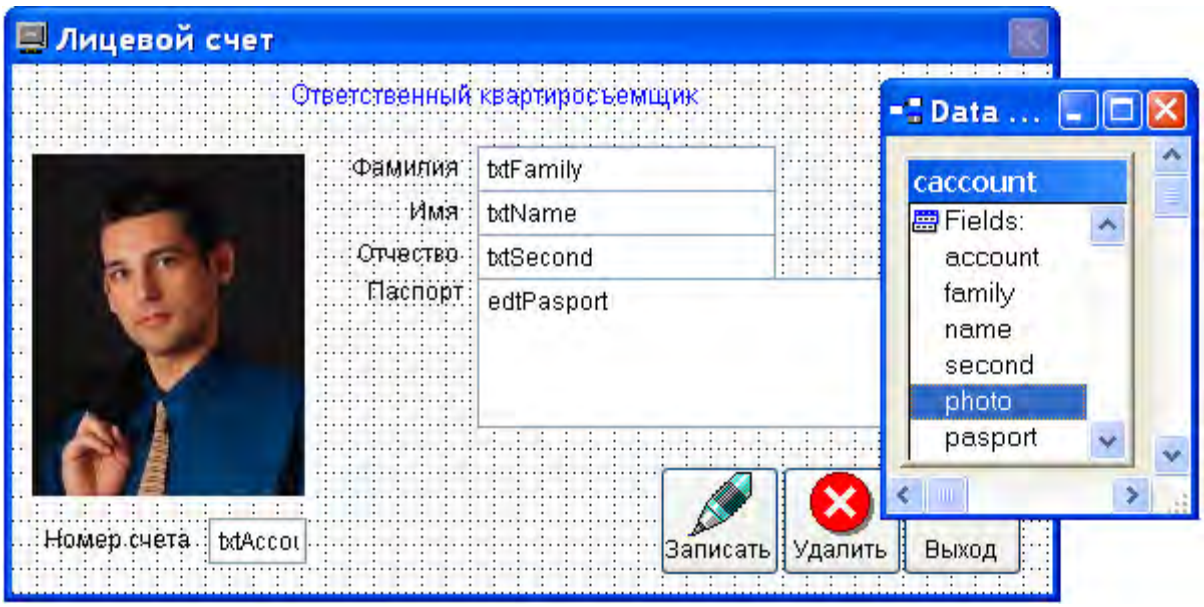


Рис. 4.27. Форма для работы с лицевым счетом

Работа с изображениями в этой форме выполнена по каноническим принципам Visual FoxPro. Фотографии не хранятся в отдельных файлах, как мы делали это ранее, а заносятся в поле ***General***. Если быть более точным, то в поле ***General*** хранятся ссылки на место фотографии в файле ***cAccount.fpt***. В этом же файле хранятся поля ***Memo***. У нашей таблицы это поле ***pasport***.

Откройте таблицу ***cAccount*** в окне просмотра ***Browse***. Для этого в командном окне ***Command*** наберите:

```
USE c:\winnt\temp\caccount.dbf EXCLUSIVE
BROWSE
```

На экране дисплея появится окно ***Browse*** (рис. 4.28).

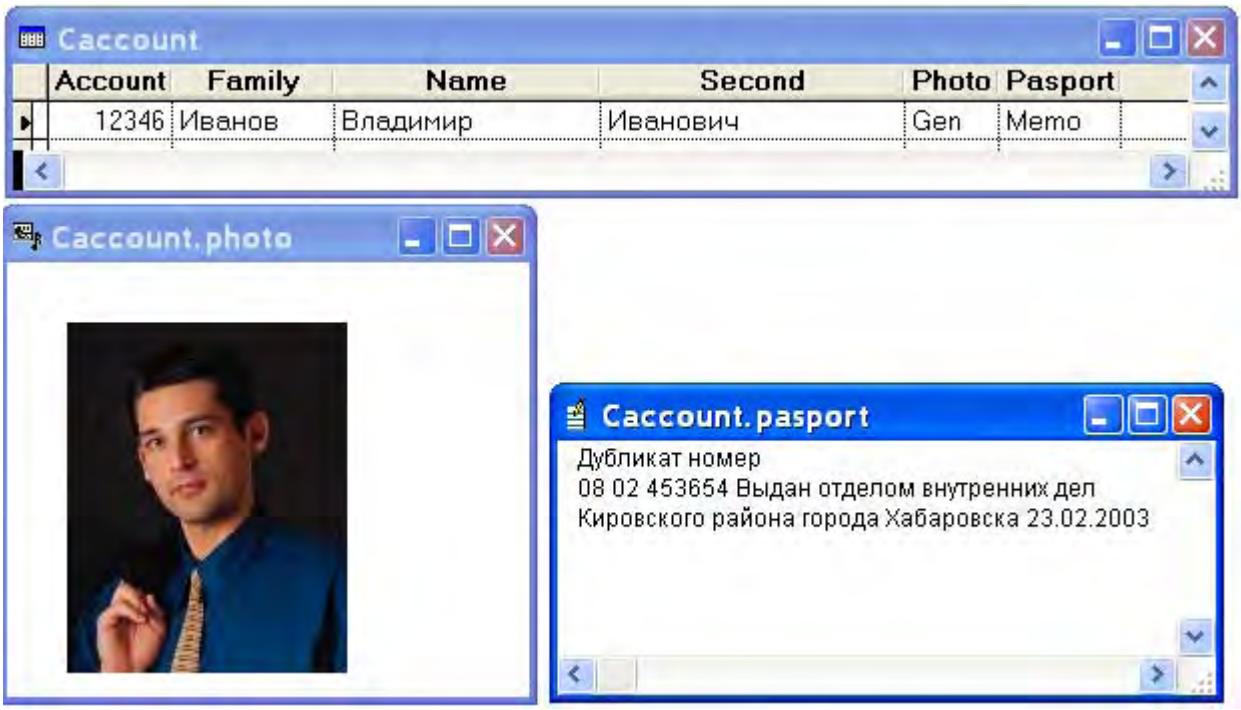


Рис. 4.28. Таблица-выборка ***cAccount*** в режиме просмотра

Если в поле типа ***General*** содержится информация, то в соответствующей ячейке мы увидим слово ***Gen***, начинающееся с большой буквы, если информации нет – ***gen*** с маленькой буквы. Сделайте двойной щелчок по этому слову и увидите содержимое ячейки – фотографию. Все сказанное также справедливо и для поля типа ***Memo***.

Сделайте активным окно с фотографией. В главном меню Visual FoxPro выберите цепочку: Edit – Точечный рисунок Object – Открыть. Запустится ***Paint***. Делайте с фотографией все, что необходимо. Кроме этого работает буфер обмена Windows. Его также можно использовать для занесения нового изображения. Цепочка в этом случае короче: Edit – Paste.

Код события ***Activate*** формы ***Account***:

```
IF IndAccount=1
   * Кнопка "Удалить" недоступна
   THISFORM.Command2.Enabled=.F.
ENDIF
```

Код события ***Destroy*** формы ***Account***:

```
* Удаление глобальной переменной из памяти
RELEASE IndAccount
```

Код события ***Click*** кнопки ***Записать***:

```
*- Кнопка Записать
* Проверка введенных значений
IF LEN(ALLTRIM(cAccount.Family))=0
  =MESSAGEBOX('Вы забыли ввести фамилию!',;
              48,'Ошибка!')
  ThisForm.txtFamily.Setfocus
  RETURN
ENDIF
IF LEN(ALLTRIM(cAccount.Name))=0
  =MESSAGEBOX('Вы забыли ввести имя!',;
              48,'Ошибка!')
  ThisForm.txtName.Setfocus
  RETURN
ENDIF
IF LEN(ALLTRIM(cAccount.Second))=0
  =MESSAGEBOX('Вы забыли ввести отчество!',;
              48,'Ошибка!')
  ThisForm.txtSecond.Setfocus
  RETURN
ENDIF
DO CASE
   CASE IndAccount=1
      * Добавление нового квартиросъемщика
      lnMsgResult=MESSAGEBOX('Сейчас данные о квартиросъемщике '+;
                'будут записаны в базу.',52,'Подтвердите!')
      IF lnMsgResult=6     && Кнопка Да
         SELECT cAccount
         SCATTER MEMO MEMVAR
         * Запись в основную базу
         IF .NOT. USED ('Account')
            USE Account IN 0
         ENDIF
         SELECT Account
         APPEND BLANK
         GATHER MEMO MEMVAR
         THISFORM.Release
      ENDIF
```

```
   CASE IndFlat=2
      * Редактирование данных квартиросъемщика
      lnMsgResult=MESSAGEBOX('Сейчас результаты корректировки '+;
               'будут записаны на диск.',52,'Подтвердите!')
      IF lnMsgResult=6
         SELECT cAccount
         SCATTER MEMO MEMVAR
         * Корректировка в основной базе
         IF .NOT. USED ('Account')
              USE Account IN 0
         ENDIF
         SELECT account
         SET ORDER TO TAG Account
         SEEK cAccount.Account
         IF FOUND()
              GATHER MEMO MEMVAR
         ELSE
         =MESSAGEBOX('Нет доступа к таблице квартиросъемщиков, '+;
                 'или в таблице отсутствует запись, '+;
                 ' данные которой Вы редактировали.';
                 ,48,'А вот Вам и проблема!')
         ENDIF
         * Работаем с таблицей-выборкой
         SELECT cAccount
         THISFORM.Release
      ENDIF
ENDCASE
```

Код события ***Click*** кнопки ***Удалить***:

```
*- Кнопка Удалить
lnMsgResult=MESSAGEBOX('Подтвердите!',52,'Удаление!')
IF lnMsgResult=6
  * Удаление в основной базе
  IF .NOT. USED ('Account')
     USE Account IN 0
  ENDIF
  SELECT Account
  * Индексированный поиск
  SET ORDER TO TAG Account
  SEEK cAccount.Account
  IF FOUND()
     DELETE     && Удаление
  ELSE
     =MESSAGEBOX('Нет доступа к таблице квартиросъемщиков, '+;
                 'или в таблице отсутствует запись, '+;
                 'данные которой Вы хотите удалить.';
                  ,48,'А вот Вам и проблема!')
  ENDIF
  * Удаление формы из памяти
  THISFORM.Release
ENDIF
```

4.4. Создание процедур и форм поддержки

4.4.1. Форма Adjust – задержка при пошаговом поиске

Вернемся к форме ***Building***, к ее второй странице. В рабочем состоянии она показана на рис. 4.23. Обратите внимание на поле ***Адрес***. Это объект ***Combo Box*** (раскрывающееся поле). В нем представлен список улиц города. Их больше тысячи. Несмотря на расположение в алфавитном порядке, выбор нужной улицы – весьма трудоемкая операция. Если при создании ***Combo Bo***x поставить в качестве значения свойства ***IncrementalSearch*** истину (.T.- True), то пользователь получит возможность применить пошаговый поиск. Что это означает? Запустите приложение ***Real Estate***. Добейтесь появления на экране второй страницы формы ***Building***. Заголовок ее окна: «Список зданий, попавших в запрос». Щелкните по кнопке ***Исправить***.

Раскройте Combo Box – ***Адрес***. Указатель мыши обязательно должен оставаться на месте щелчка. Дальше работаем с клавиатурой. Допустим, нам необходимо найти в списке улицу Серышева. Набираем на клавиатуре С-е-р-ы и т. д. По мере ввода символов компьютер при каждом приращении в комбинации набранных букв ищет соответствующее им слово. Иначе говоря, сначала ищется первое слово, начинающееся на С, далее первое слово, начинающееся на Се, далее первое слово, начинающееся на Сер и т. д. Теперь подробнее.

После нажатия клавиши С Вы увидите в списке следующую картину:

```
Садовый переулок
Салтыкова-Щедрина Улица
Санаторная Улица
Санаторный переулок
```

После нажатия клавиши е содержимое списка изменится:

```
Севастопольская Улица
Севастопольский переулок
Северная Улица
Седова переулок
```

После нажатия клавиши р появится следующее:

```
Серова Улица
Серпуховский переулок
Серышева Улица
Сеченова Улица
```

Нужная нам улица появилась в поле ***Combo Box***. Щелкните по ней мышью. Список закроется. На экране останется выбранное значение: ***Серышева***. При пошаговом поиске компьютер не отличает больших букв от маленьких. Поэтому можете работать в любом регистре, но русский шрифт должен быть включен обязательно!

Внимание! Между нажатиями на клавиши у Вас только полсекунды. Замешкались, в нашем примере, с буквой е, увидите улицы на эту букву.

Промедлили – не беда. Начните набор сначала: С-е-р-ы и т. д. Только побыстрее. А теперь о правиле «полсекунды». Время реакции у каждого человека индивидуально. Пользователь должен иметь возможность настроить компьютер под себя. Для этого и предназначена форма ***Adjust*** (рис. 4.29).

Основной элемент формы – объект ***Spinner***, ***ControlSource*** которого –

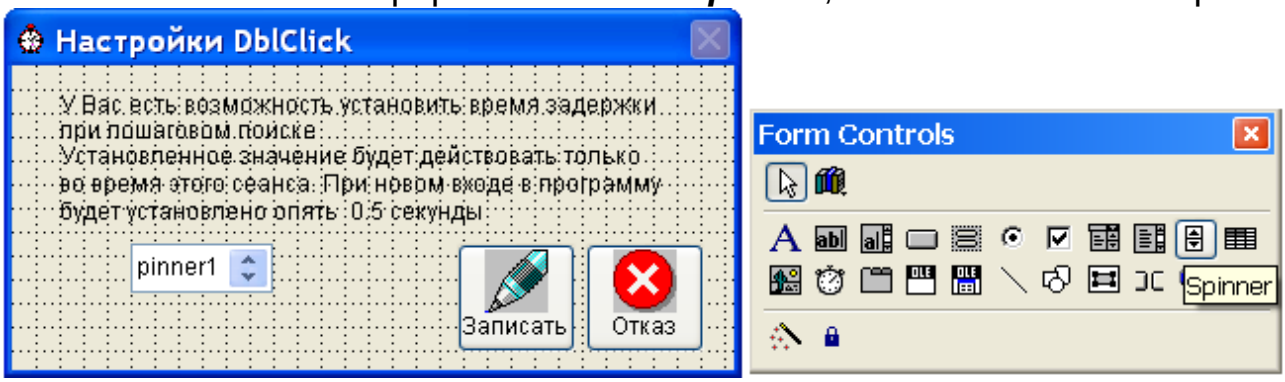


Рис. 4.29. Форма ***Adjust*** в конструкторе форм

системная переменная ***_dblclick***. Время задержки при пошаговом поиске и время, в течение которого два сделанных щелчка мыши, превращаются в двойной щелчок, это одна и та же переменная! Распорядитесь ее значением, в зависимости от того, какая цель поставлена. Приведем код события ***Click*** кнопки ***Записать***:

```
*- Кнопка Записать
IF .NOT. BETWEEN(_dblclick,0.25,1.25)
  =MESSAGEBOX('Давайте ограничимся диапазоном 0.25-1.25 секунды';
              ,48,'Предложение!')
  ThisForm.Spinner1.SetFocus
  RETURN
ENDIF
ThisForm.Release
```

4.4.2. Просмотр состояния памяти

Переменные существуют, пока выполняется приложение или продолжается сеанс Visual FoxPro, в котором они были созданы. Чтобы задать область видимости переменной, употребляются ключевые слова LOCAL, PRIVATE и PUBLIC.

- Ключевое слово LOCAL создает локальные переменные, которые могут использоваться и модифицироваться только в той программе, где были созданы, и которые недоступны в программах более высокого или более низкого уровня. Локальные переменные и массивы освобождаются при прекращении выполнения содержащей их программы.
- Ключевое слово PRIVATE скрывает от текущей программы переменные и массивы, определенные в вызывающей программе. Имена таких пе-

ременных можно использовать в текущей программе, не влияя на исходные переменные. Когда программа, содержащая объявление с ключевым словом PRIVATE, оканчивается, все скрытые переменные и массивы снова становятся доступными.

- Ключевое слово PUBLIC определяет глобальные переменные или массивы. Глобальные переменные и массивы могут использоваться и модифицироваться из любой программы, запускаемой в текущем сеансе Visual FoxPro. Все переменные и массивы, создаваемые в окне Command, являются глобальными.

Если переменная имеет то же имя, что и поле, Visual FoxPro всегда отдает предпочтение имени поля.

Для того чтобы знать какие переменные находятся в памяти в данный момент и чему равно их значение, воспользуйтесь процедурой DisplayMemory. Для вызова ее в нужный момент времени включите в текст головного модуля строчку: ON KEY LABEL F2 DO DisplayMemory. Теперь при нажатии клавиши F2 всегда можно получить карту состояния памяти (рис. 4.30).

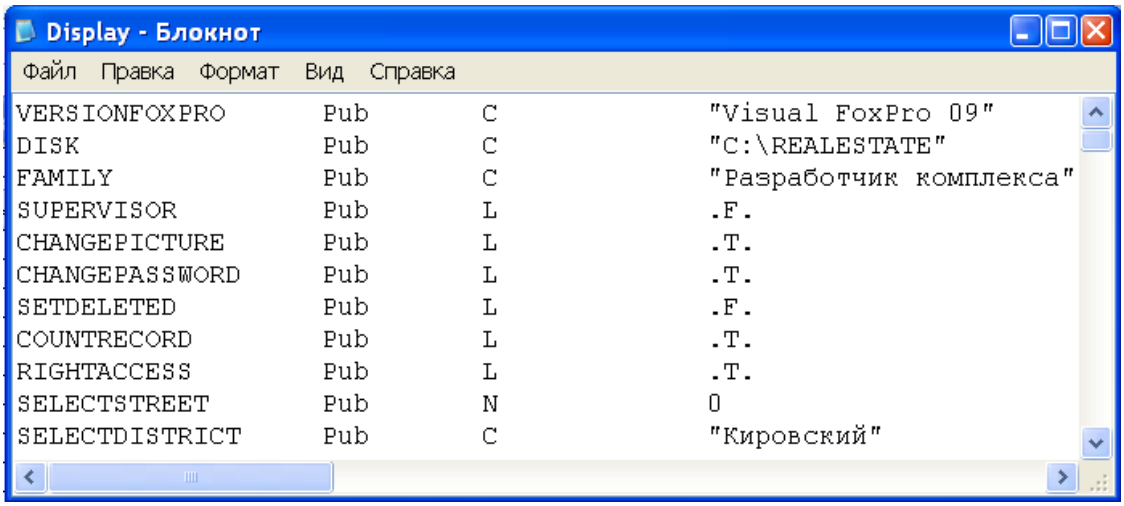


Рис. 4.30. Карта состояния памяти

Текст процедуры размещен в процедурном файле ***FileProc***:

```
PROCEDURE DisplayMemory                    && Состояние памяти
PRIVATE FileName
* Имя временного файла - Display.txt
FileName='C:\WINNT\TEMP\Display.txt'
* Разместить состояние памяти в файле
DISPLAY MEMORY TO FILE &FileName NOCONSOLE
* Просмотреть при помощи программы Блокнот
RUN /N1 NOTEPAD.EXE &FileName
* Удалить временный файл
DELETE FILE &FileName
RETURN
```

4.4.3. Информация о рабочей станции

Для правильной настройки программного комплекса при работе в локальной вычислительной сети необходимо убедиться в том, что пути поиска компонентов и их размещение указаны правильно. В этом вам поможет процедура ***Inform***, результаты работы которой показаны на рис. 4.31. Она также размещена в процедурном файле ***FileProc***.

Текст процедуры ***Inform***:

```
PROCEDURE Inform      && Информация о компьютере
DEFINE WINDOW INFO FROM 2,5 TO 27,135;
       TITLE ' Информация ';
       FONT "COURIER NEW",10;
       FLOAT CLOSE ZOOM GROW MINIMIZE;
       STYLE "B"
```

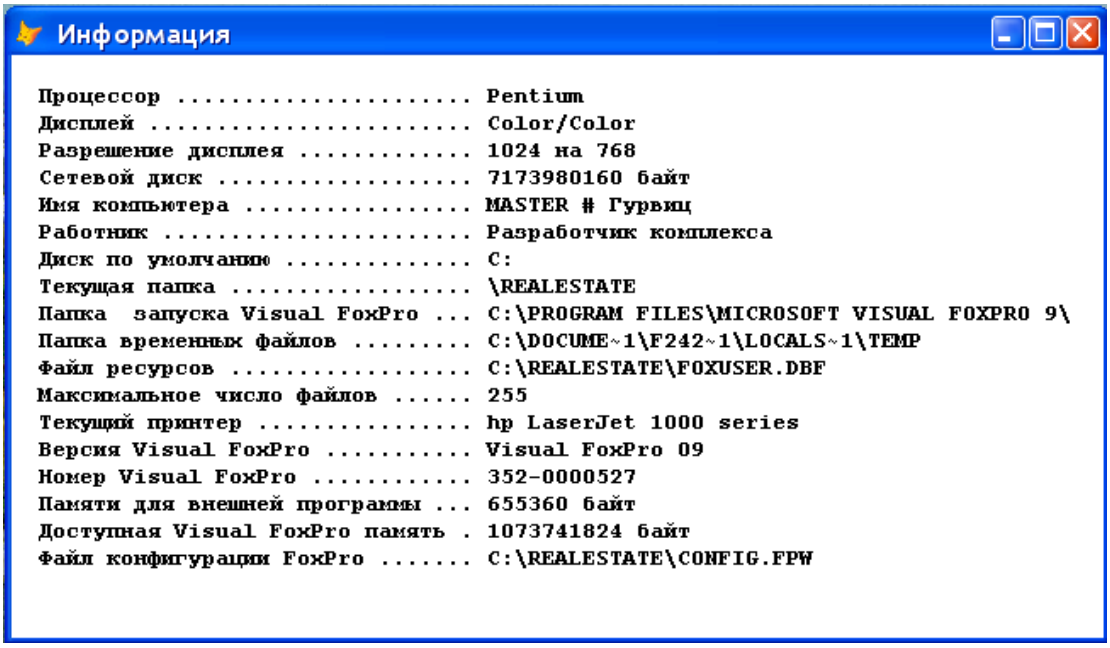


Рис. 4.31. Информация о рабочей станции

```
MOVE WINDOW INFO CENTER
ACTIVATE WINDOW INFO
? [  Процессор ..................... ]+SYS(17)
? [  Дисплей ....................... ]+SYS(2006)
? [  Разрешение дисплея ............ ]+;
                ALLTRIM(STR(SYSMETRIC(1),4,0))+;
                [ на ]+ALLTRIM(STR(SYSMETRIC(2),4,0))
? [  Сетевой диск .................. ]+SYS(2020)+[ байт]
? [  Имя компьютера ................ ]+SYS(0)
? [  Работник ...................... ]+FAMILY
? [  Диск по умолчанию ............. ]+SYS(5)
? [  Текущая папка ................. ]+SYS(2003)
? [  Папка  запуска Visual FoxPro ... ]+SYS(2004)
```

```
? [  Папка временных файлов ......... ]+SYS(2023)
? [  Файл ресурсов ................... ]+SYS(2005)
? [  Максимальное число файлов ...... ]+SYS(2010)
? [  Текущий принтер ................. ]+SYS(6)
? [  Версия Visual FoxPro ............ ]+PADR(VERSION(),16)
? [  Номер Visual FoxPro ............. ]+SYS(9)
? [  Памяти для внешней программы ... ]+SYS(12)+[ байт]
? [  Доступная Visual FoxPro память . ]+SYS(1001)+[ байт]
? [  Файл конфигурации FoxPro ....... ]+SYS(2019)
? [ ]
RETURN
```

4.4.4. Информация о заполнении базы данных

Хорошую помощь при составлении отчетов о проделанной работе может оказать запуск формы ***Fill***. Результаты ее работы показаны на рис. 4.32. В окружение данных этой формы ничего помещать не надо. Необходимые выборки выполняются в процессе работы и удаляются после закрытия формы.

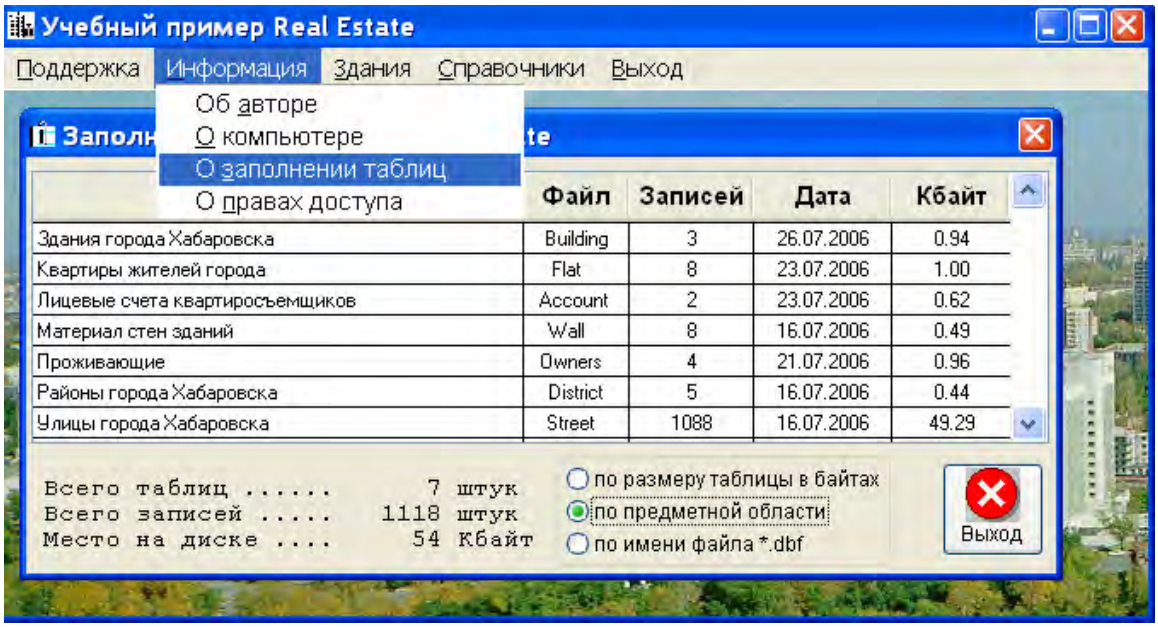


Рис. 4.32. Информация о заполнении таблиц базы данных «Real Estate»

Код события ***Load*** формы ***Fill***:

```
* Создание временной таблицы Fill
SELECT * FROM INFO INTO TABLE 'C:\WINNT\TEMP\FILL'
SELECT INFO
USE
* Добавляем поля в таблицу-выборку
ALTER TABLE 'C:\WINNT\TEMP\FILL' ;
     ADD COLUMN NumRec N(8);
```

```
     ADD COLUMN DateModify D;
     ADD COLUMN AllField N(2);
     ADD COLUMN LenRec N(3);
     ADD COLUMN AllByte N(13,2)
* Создание глобальных переменных
PUBLIC nSumRecord,nSumByte
STORE 0 TO nSumRecord,nSumByte
SELECT FILL
SCAN
   SCATTER MEMVAR
   NameTable=[']+ALLTRIM(M.DBF)+[')]
   NameSelect=ALLTRIM(M.DBF)
   IF .NOT. USED &NameTable
      USE &NameTable IN 0
   ENDIF
   SELECT &NameSelect
   M.NumRec=RECCOUNT()
   M.DateModify=LUPDATE()
   M.AllField=FCOUNT()
   M.LenRec=RECSIZE()
   M.AllByte=(RECCOUNT()*RECSIZE()+HEADER()+1)/1024
   USE
   SELECT FILL
   GATHER MEMVAR
ENDSCAN
* Подсчет количества записей и размера таблиц
SUM NUMREC TO nSumRecord ALL
SUM ALLBYTE TO nSumByte ALL
nSumByte=nSumByte
* Создание индексов
INDEX ON ALLBYTE TAG ALLBYTE OF 'C:\WINNT\TEMP\FILL' DESCENDING
INDEX ON DBF TAG DBF OF 'C:\WINNT\TEMP\FILL'
INDEX ON INFO TAG INFO OF 'C:\WINNT\TEMP\\FILL'
SET ORDER TO TAG ALLBYTE
GOTO TOP
```

Код события ***Activate*** формы ***Fill***:

```
ThisForm.Label1.Caption=[Всего таблиц ......]+;
                        STR(RECCOUNT(),7)+[ штук]
ThisForm.Label2.Caption=[Всего записей .....]+;
                        STR(nSumRecord,7)+[ штук]
ThisForm.Label3.Caption=[Место на диске ....]+;
                        STR(nSumByte,7)  +[ Кбайт]
ThisForm.Command1.SetFocus
```

Код события ***Destroy*** формы ***Fill***:

```
* Освобождение глобальных переменных
RELEASE nSumRecord,nSumByte
* Удаление временных выборок
```

```
CLOSE TABLES ALL
IF FILE ('C:\WINNT\TEMP\Fill.dbf' )
   DELETE FILE 'C:\WINNT\TEMP\Fill.dbf'
ENDIF
IF FILE ('C:\WINNT\TEMP\Fill.bak' )
   DELETE FILE 'C:\WINNT\TEMP\Fill.bak'
ENDIF
IF FILE ('C:\WINNT\TEMP\Fill.cdx' )
   DELETE FILE 'C:\WINNT\TEMP\Fill.cdx'
ENDIF
```

4.4.5. Информация о рабочих областях

Каждую таблицу Visual FoxPro открывает в новой рабочей области. Попытка дважды открыть таблицу заканчивается сообщением об ошибке: «File is in use». Что делать? Нажмите Ctrl + F2 и увидите окно (рис. 4.33). Для того чтобы комбинация клавиш сработала, в текст головного модуля должна быть добавлена строчка:

ON KEY LABEL Ctrl+F2 DO FORM AreaWork

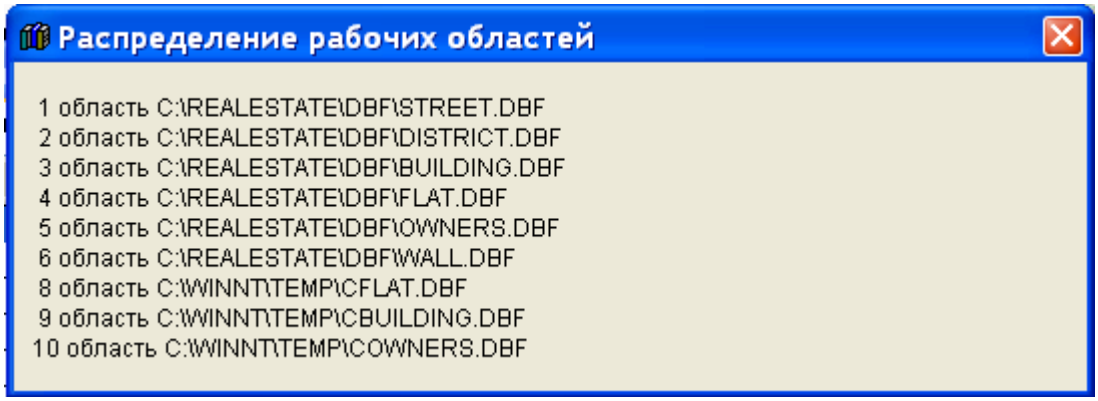


Рис. 4.33. Информация о распределении рабочих областей

Это запуск формы ***AreaWork***. Форма не содержит никаких объектов, кроме заголовка и иконки. Текст события ***Activate*** имеет вид

```
* Сканирование максимум двадцати областей
* В больших программных комплексах следует цифру увеличить
FOR I=1 TO 20
    IF LEN(ALLTRIM(DBF(I)))#0
       ? [   ]+STR(I,2)+[ область ]+DBF(I)
    ENDIF
ENDFOR
* Отключить на время работы формы вызов по CTRL+F2
ON KEY LABEL CTRL+F2
```

В код события ***Destroy*** поместите возврат комбинации клавиш:

```
* Вернуть назад вызов формы AreaWork по CTRL+F2
```

```
ON KEY LABEL CTRL+F2 DO FORM AreaWork
```

4.4.6. Смена картинки главного окна

Экран компьютера целый день перед глазами работника предприятия. Унылая картина. Не правда ли? Чтобы хоть как-то разнообразить ее, дайте пользователю возможность выбора фона главного окна программного комплекса. Лучше, если картинок будет побольше. На рис. 4.34 показан рабочий вид формы ***ChangPic***.

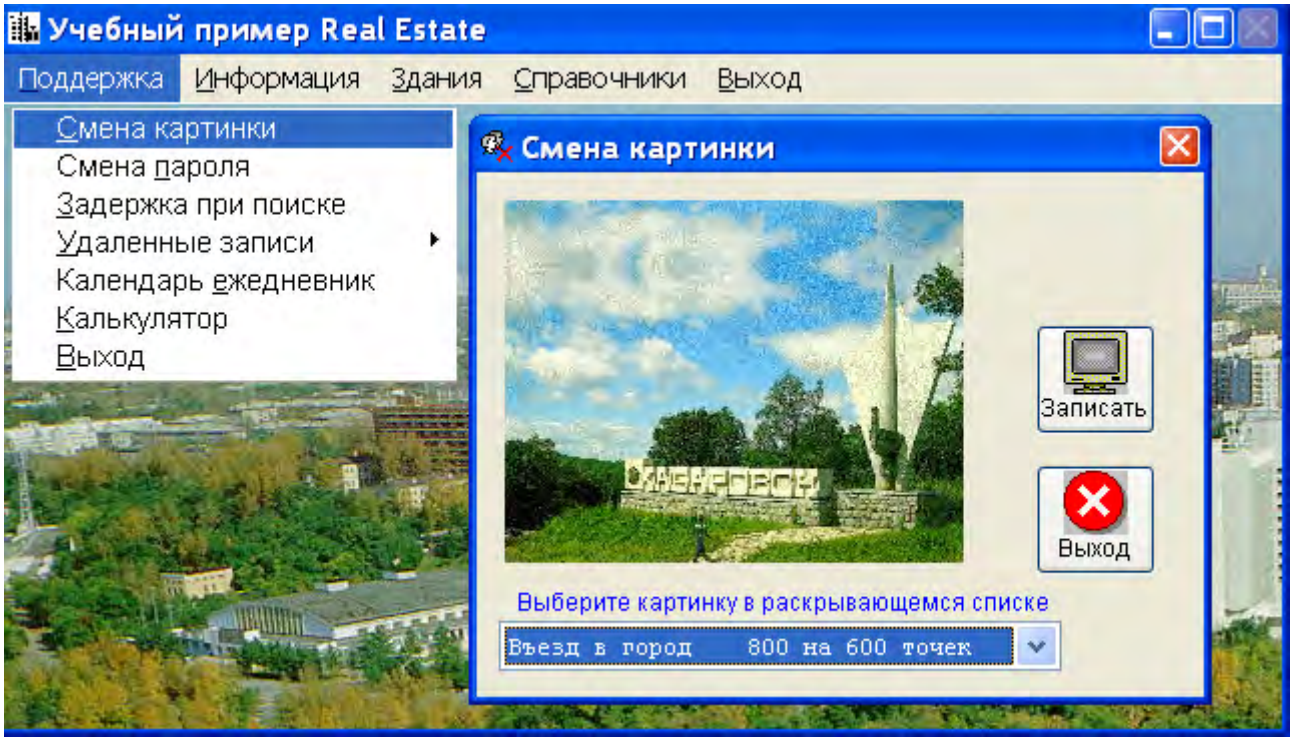


Рис. 4.34. Смена картинки главного окна программного комплекса

Код события ***Activate*** формы ***ChangePic***:

```
* Для преемственности в работе во время
* одного сеанса имя картинки - глобальное
Public NamePicture
NamePicture=[Въезд в город    800 на 600 точек]
* Определение полного пути к папке с картинками
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
   RealDirectory=[\]
ENDIF
THISFORM.IMAGE1.PICTURE=DISK+RealDirectory+[\DECKTOP\khab.jpg]
* Описание массива Title
* Он используется в RowSourceType (5-Array) Combo1
* Число элементов - по числу картинок
DIMENSION Title(28)
Title(1) =[Въезд в город     800 на 600 точек]
Title(2) =[Физкультура       800 на 600 точек]
Title(3) =[ХабИИЖТ           800 на 600 точек]
```

```
Title(4) =[Манеж            1024 на 768 точек]
* Два десятка строк в целях экономии места пропущено
Title(26)=[Мост            1280 на 1024 точек]
Title(27)=[Сказочный город 1280 на 1024 точек]
Title(28)=[Без картинки]
THISFORM.COMBO1.Refresh
```

Код события ***InteractiveChange*** объекта ***Combo1***:

```
* Определение полного пути к папке с картинками
* Переменная Shot содержит полный путь к папке
IF ALLTRIM(SYS(2003))=[\]
   RealDirectory=[]
ELSE
   RealDirectory=[\]
ENDIF
Shot=DISK+RealDirectory+[\DECKTOP\]
SET EXACT OFF
* Неполное соответствие для того, чтобы
* не писать далее название картинок целиком
* Например
* CASE THIS.VALUE=[Физкультура]
* Вместо
* CASE THIS.VALUE=[Физкультура       800 на 600 точек]

DO CASE
   CASE THIS.VALUE=[Физкультура]
        THISFORM.IMAGE1.PICTURE=Shot+[SPORTS.JPG]
   CASE THIS.VALUE=[Въезд в город]
        THISFORM.IMAGE1.PICTURE=Shot+[KHAB.JPG]
   CASE THIS.VALUE=[ХабИИЖТ]
        THISFORM.IMAGE1.PICTURE=Shot+[RailWay.JPG]
   CASE THIS.VALUE=[Манеж]
        THISFORM.IMAGE1.PICTURE=Shot+[Area.JPG]
   * Два десятка строк пропущено
   CASE THIS.VALUE=[Железная дорога]
        THISFORM.IMAGE1.PICTURE=Shot+[RAILROAD.JPG]
   CASE THIS.VALUE=[Без картинки]
        THISFORM.IMAGE1.PICTURE=Shot+[NO.JPG]
ENDCASE
```

Код события ***Click*** кнопки ***Записать***:

```
* Если картинка для главного окна комплекса назначена
* удалить ее
IF FILE('C:\WINNT\TEMP\Picture.jpg')
   DELETE FILE('C:\WINNT\TEMP\Picture.jpg')
ENDIF

IF NamePicture=[Без картинки]
   * Вывод только на экран
```

```
   _SCREEN.PICTURE=[]
ELSE
   * Картинку на экран и в файл C:\WINNT\TEMP\Picture.jpg
   _SCREEN.PICTURE=THISFORM.IMAGE1.PICTURE
   FilePicture=THISFORM.IMAGE1.PICTURE
   COPY FILE &FilePicture TO C:\WINNT\TEMP\Picture.jpg
ENDIF
* Форму перерисовать
THISFORM.Release
```

5. СОЗДАНИЕ ОТЧЕТОВ

5.1. Создание отчета Visual FoxPro

Основная сфера применения форм – обеспечение возможности просмотра отдельных или небольших групп связанных записей. Отчеты же представляют собой наилучшее средство отображения информации из базы данных в виде печатного документа. Разработка отчета очень похожа на разработку формы. Также используется панель управления (***Report Controls***), окружение данных (***Data Environment***) и окно свойств (***Field Properties***). В этом разделе мы построим относительно несложный отчет, пройдя шаг за шагом всю цепочку его создания.

«Заставим» наш программный комплекс выдавать на печать всю информацию по зданиям, попавшим в запрос в результате работы формы ***Search***. Отчет будет запускаться из формы ***Building*** после щелчка по кнопке ***Печать***, которая расположена на первой странице формы (рис. 5.1).

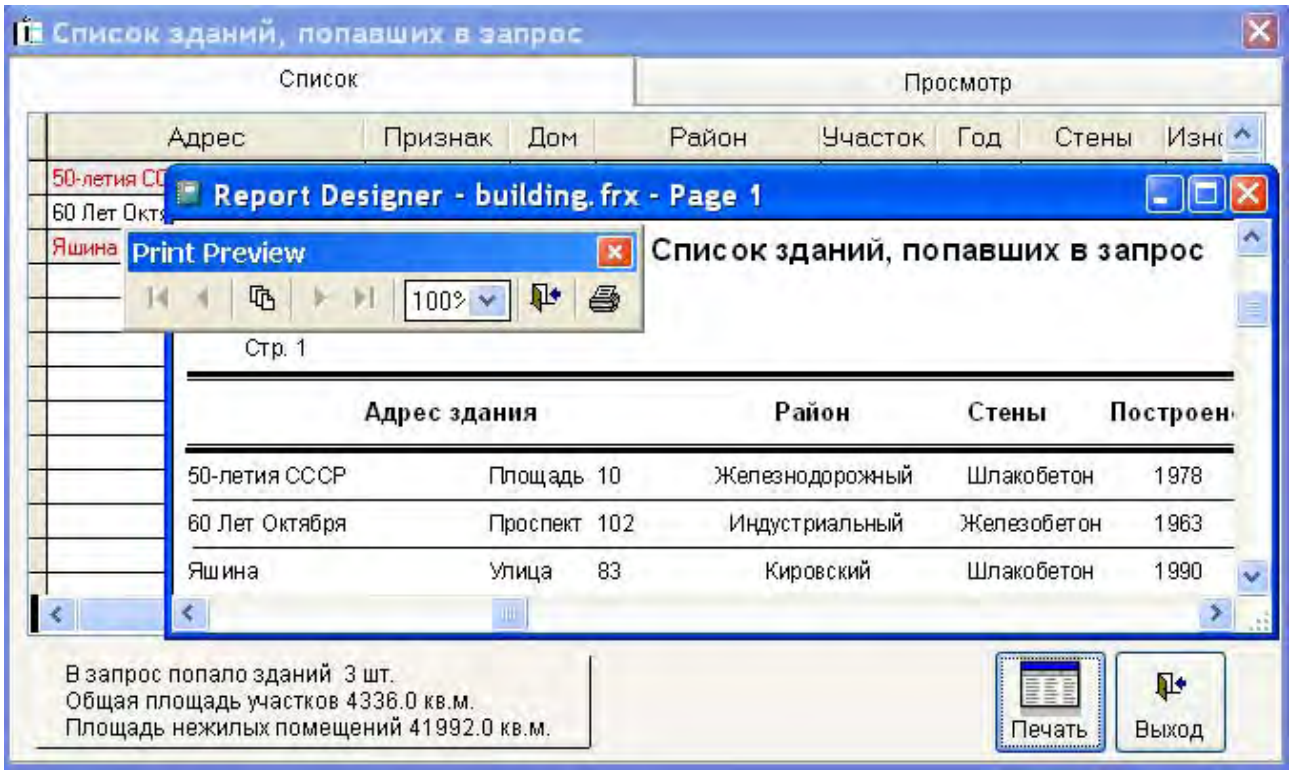


Рис. 5.1. Отчет Visual FoxPro по зданиям, попавшим в запрос

Код события ***Click*** кнопки ***Печать***:

```
*- Кнопка Печать
* Запуск отчета на выполнение
REPORT FORM Building NOEJECT NOCONSOLE PREVIEW
* Подтверждение вывода на печать, если пользователь
* забыл это сделать, используя панель Print Preview
lnMsgResult=MESSAGEBOX('Выводить на принтер',52,'Печать!')
IF lnMsgResult=6      && Кнопка Да
   REPORT FORM Building NOEJECT NOCONSOLE TO PRINTER PROMPT
ENDIF
```

Для создания отчета ***Building*** в главном меню Visual FoxPro щелкните пункт ***File*** и выберите команду ***New***. В открывшемся окне щелкните радиокнопку ***Report*** и нажмите кнопку ***New file.*** На экране дисплея появится окно ***Report Designer***. Это окно конструктора отчетов. В нем наш первый отчет с именем ***Report1***. Для работы с конструктором отчетов используются панели инструментов ***Report Designer*** (конструктор отчета) и ***Report Controls*** (элементы управления отчета), а также пункты меню ***Report*** (отчет) главного меню Visual FoxPro (рис. 5.2).

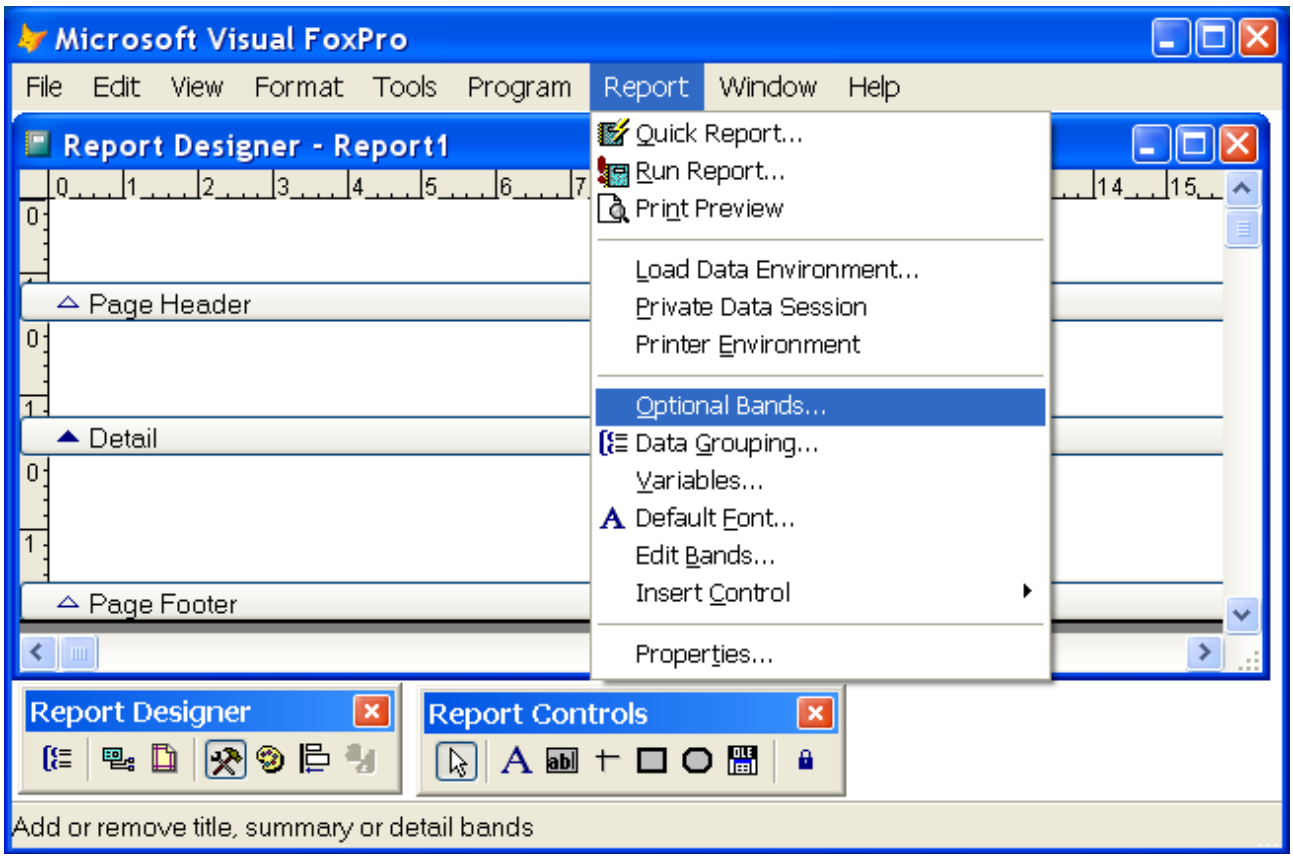


Рис. 5.2. Конструктор отчетов Visual FoxPro

В начале работы окно конструктора отчетов содержит три полосы: ***Page Header*** (верхний колонтитул), ***Detail*** (подробности) и ***Page Footer*** (нижний колонтитул). Полосы ограничены разделительными строками. Всего в отчете может быть семь полос. Каждая полоса предназначена для того, чтобы определить, когда и где будут напечатаны размещенные в ней объекты. Для отображения дополнительных полос в отчете выберите в меню ***Report*** пункт ***Optional Bands*** (рис. 5.2). Назначения всех полос отчета Visual FoxPro описаны в табл. 5.1.

Таблица 5.1

Назначение полос отчета Visual FoxPro

Название	Назначение полосы
Title (Титул)	Информация, которая будет напечатана перед основным отчетом. Размещается только на первой странице.
Page Header (Верхний колонтитул)	Данные этой полосы будут напечатаны на каждой странице. Как правило, это название отчета, номер страницы и дата создания отчета.
Group Header (Верхняя группа)	Верхние полосы сгруппированных данных. Выводятся перед самими данными.
Detail (Подробности)	Данные полей таблицы. Будет выведено столько данных, сколько строк имеет таблица.
Group Footer (Нижняя группа)	Нижние полосы сгруппированных данных
Page Footer (Нижний колонтитул)	Итоговые данные по текущей странице. Выводятся по каждой странице.
Summary (Итог)	Итоги по отчету. Выводятся один раз на последней странице.

Для выбора ширины отчета выберите в меню ***Report*** пункт ***Properties,*** а в появившемся окне ***Report Properties*** кнопку ***Page Setup***. Там же можно установить ориентацию бумаги (книжная или альбомная).

Следующий этап – определение окружения данных отчета. Добавим таблицу ***cBuilding*** в окружение отчета. Сделайте щелчок правой кнопки мыши в любом месте окна ***Report Designer***. Появится меню. Выберите в нем четвертый пункт ***Data Environment***. Еще один щелчок правой кнопкой, но уже в появившемся окне ***Data Environment*** активизирует очередное меню. Выберите в нем первый пункт ***Add***. Появится окно ***Open***. Найдите в нем таблицу ***cBuilding***. Она находится в папке ***C:\WINNT\TEMP***.

Размещение текстовой, графической информации и полей таблиц из окружения данных отчета выполняется точно также как и для форм.

5.2. Передача данных в Microsoft Word

В последних версиях Visual FoxPro компания Microsoft значительно усовершенствовала поддержку модели COM (Component Objects Model) – модели объектных компонентов. Это стандарт, регламентирующий обмен информацией между приложениями. Основная идея этого стандарта Microsoft заключается в том, что любой COM-объект может взаимодействовать с другим COM-объектом независимо от того, в какой среде он разработан. Мы теперь можем напрямую обращаться к объектам Microsoft Word из Microsoft Visual FoxPro.

Воспользуемся этой возможностью для генерации договора приватизации квартиры. Вернемся к рис. 4.26. На второй странице формы ***Flat*** расположена кнопка ***Договор***. Код события ***Click*** этой кнопки имеет вид:

```
*- Кнопка Договор
* Номер выбранной квартиры
SelectFlat=cFlat.Flat
* Остальные параметры адреса смотри:
* событие Activate второй страницы формы Building
* SelectStreetAddress - номер улицы
* SelectHouseAddress  - номер дома

* Список жильцов для приватизации
SELECT * FROM Owners;
        WHERE Street=SelectStreetAddress;
              AND House=SelectHouseAddress;
              AND Flat=SelectFlat;
        INTO TABLE 'C:\WINNT\TEMP\cOwners.dbf';
        ORDER BY Number
IF RECCOUNT()=0
  =MESSAGEBOX('В квартире нет проживающих!',48,'Ошибка!')
  RETURN
ENDIF
DO FORM Treaty  && Генерация договора
```

Если в квартире есть проживающие, то на выполнение будет запущена форма ***Treaty*** (рис. 5.3). Эта форма предназначена для сбора дополнительных данных, которые наряду с имеющимися в базе данных Visual FoxPro будет переданы в Microsoft Word. На рис. 5.4 показан окончательный вид договора приватизации.

Код события ***Init*** формы ***Treaty*** имеет вид

```
PUBLIC SelectDateTreaty,SelectChief,SelectText,;
       SelectMemorial,SelectTypeFlat,SelectTypeKind,;
       FirstStringSource

* Дата заключения договора
SelectDateTreaty=DATE()
* Фамилия подписавшего договор
SelectChief=1         && Начальник группы
* Первая строчка договора
SelectText=[]
```

```
FirstStringSource=4    && Первой строчки нет

* Здание - памятник архитектуры
SelectMemorial=.F.     && Нет
* Тип квартиры
SelectTypeFlat=1       && Отдельная
* Вид собственности
SelectTypeKind=2       && В равных долях
```

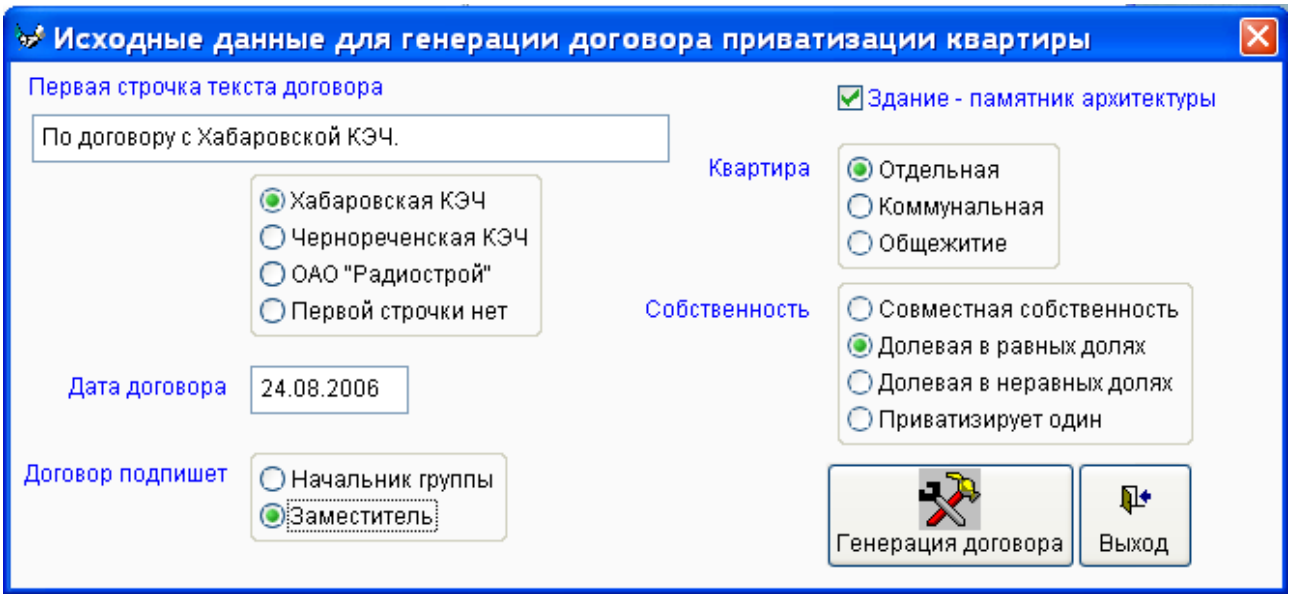


Рис. 5.3. Сбор дополнительных данных для генерации договора

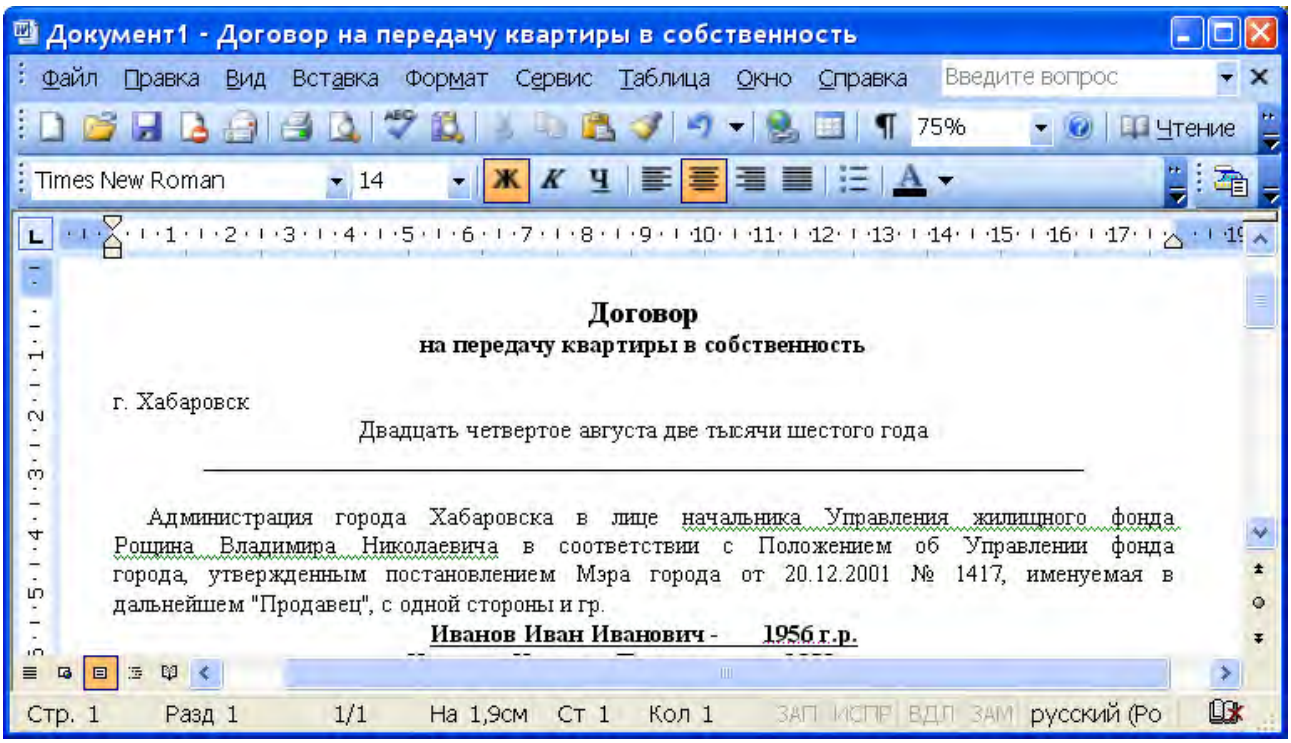


Рис. 5.4. Окончательный вид сгенерированного договора приватизации

Код события ***Click*** кнопки ***Генерация договора*** формы ***Treaty***:

```
WAIT 'Ждите! Идет передача данных в Microsoft Word' WINDOW NOWAIT
* Решение "проблемы" русского языка
DO CASE
   CASE cFlat.Rooms=1
        SelectRoom =[одной комнаты]
        SelectRoom1=[однокомнатной]
        SelectRoom2=[одну комнату]
   CASE cFlat.Rooms=2
        SelectRoom =[двух комнат]
        SelectRoom1=[двухкомнатной]
        SelectRoom2=[две комнаты]
   CASE cFlat.Rooms=3
        SelectRoom =[трех комнат]
        SelectRoom1=[трехкомнатной]
        SelectRoom2=[три комнаты]
   OTHERWISE
        SelectRoom =STR(cFlat.Rooms,2)+[ комнат]
        SelectRoom1=STR(cFlat.Rooms,2)+[ комнатной]
        SelectRoom2=STR(cFlat.Rooms,2)+[ комнат]
ENDCASE
DO CASE
   CASE SelectChief=1
        ChiefShot=[Рощин В.Н.]
        ChiefLong=[Рощина Владимира Николаевича]
   CASE SelectChief=2
        ChiefShot=[Симонова Л.И.]
        ChiefLong=[Симоновой Людмилы Ивановны]
ENDCASE
* Дата прописью
STORE [ ] TO DayText,MonthText,YearText
* Процедура находится в процедурном файле FileProc
DO Detail WITH SelectDateTreaty,DayText,MonthText,YearText
* Запущен ли Word?
ON ERROR oWord=.NULL.
* В случае возникновения ошибки в следующей строке считаем,
* что объекта oWord нет
oWord=GetObject(, "WORD.Application")
IF ISNULL(oWord)
   * Word не запущен
   ErrorWord=.T.             && Word на компьютере есть
   ON ERROR ErrorWord=.F.    && Word на компьютере нет
   * ErrorWord=.F. в случае возникновения ошибки
   * в следующей строке при запуске Word
   oWord=CREATEOBJECT("WORD.Application")  && Запускаем Word
   * Вернуть назад стандартную процедуру обработки ошибок
   ON ERROR DO ERRORHND
   IF ErrorWord=.F.
      =MESSAGEBOX('На Вашем компьютере отсутствует '+;
                  'Microsoft Word',48,'Ошибка!')
```

```
      RETURN
   ENDIF
ELSE
  =MESSAGEBOX('Microsoft Word уже запущен! Найдите его '+;
              ' на Панели задач внизу экрана',48,'Ошибка!')
  RETURN
ENDIF
* Константы Microsoft Word
#DEFINE True .T.
#DEFINE False .F.
#define wdOrientPortrait  0
#define wdToggle  9999998
#DEFINE wdUnderlineNone   0
#DEFINE wdUnderlineSingle   1
#DEFINE wdAlignParagraphLeft   0
#DEFINE wdAlignParagraphCenter   1
#DEFINE wdAlignParagraphRight   2
#DEFINE wdAlignParagraphJustify   3
#DEFINE wdAlignParagraphDistribute   4
#DEFINE wdAlignParagraphJustifyMed   5
#DEFINE wdAlignParagraphJustifyHi   7
#DEFINE wdAlignParagraphJustifyLow   8
#DEFINE wdAllowOnlyRevisions   0
oWord.Visible=.T.
* Заголовок окна Word
oWord.Caption=[Договор на передачу квартиры в собственность ]
oWord.Documents.Add
* Ориентация книжная бумага А4
* Поля (1 см = 28 пунктов)
WITH oWord.ActiveDocument.PageSetup
     .LineNumbering.Active = False
     .Orientation = wdOrientPortrait
     .LeftMargin = 48
     .RightMargin = 40
     .TopMargin =56
     .BottomMargin=56
ENDWITH
* Включить расстановку переносов
WITH oWord.ActiveDocument
     .AutoHyphenation = True
     .HyphenateCaps = True
     .ConsecutiveHyphensLimit = 0
ENDWITH

WITH oWord.Selection
     .Font.Name = "Times New Roman"
     .Font.Size = 14
     .Font.Bold = wdToggle
     .ParagraphFormat.Alignment = wdAlignParagraphCenter
     .TypeText ("Договор")
     .TypeParagraph
```

```
.Font.Size = 12
lcText=[на передачу квартиры в собственность]
.TypeText (lcText)
IF SelectMemorial=.T.
   * Если здание - памятник
   .TypeParagraph
   lcText=[в жилом доме-памятнике истории и культуры]
   .TypeText (lcText)
ENDIF
.TypeParagraph
.TypeParagraph
.Font.Size = 12
.Font.Bold = wdToggle
.ParagraphFormat.Alignment = wdAlignParagraphJustify
.TypeText ("г. Хабаровск")
.TypeParagraph
.ParagraphFormat.Alignment = wdAlignParagraphCenter
lcText=ALLTRIM(DayText+MonthText+YearText)  && Дата прописью
.TypeText (lcText)
.TypeParagraph
* Первая черта
lcText=REPLICATE([_],70)
.TypeText (lcText)
.TypeParagraph
* Первая строчка
IF LEN(ALLTRIM(SelectText))#0
   .TypeParagraph
   .ParagraphFormat.Alignment = wdAlignParagraphJustifyMed
   lcText=[       ]+ALLTRIM(SelectText)
   .TypeText (lcText)
ENDIF
.TypeParagraph
.ParagraphFormat.Alignment = wdAlignParagraphJustifyMed
lcText=[     Администрация города Хабаровска в лице]+;
      [ начальника Управления жилищного фонда ]+ChiefLong+;
      [ в соответствии с Положением об Управлении фонда ]+;
      [города, утвержденным постановлением Мэра города от ]+;
      [20.12.2001 № 1417, именуемая в дальнейшем ]+;
      ["Продавец", с одной стороны и гр.]
.TypeText(lcText)
SELECT cOwners
.TypeParagraph
.Font.Bold = wdToggle
.Font.Underline = wdUnderlineSingle
.ParagraphFormat.Alignment = wdAlignParagraphCenter
SCAN
  lcText=ALLTRIM(cOwners.Family)+[ ]+;
    ALLTRIM(cOwners.Name)+[ ]+;
    ALLTRIM(cOwners.Second)+[ - ]+STR(cOwners.Born)+[ г.р.]
    .TypeText (lcText)
    .TypeParagraph
```

```
ENDSCAN
IF RECCOUNT()>1
   * Печатать, если собственников более одного
   DO CASE
      CASE SelectTypeKind=1
           lcText=[(совместная собственность)]
      CASE SelectTypeKind=2
           lcText=[(в равных долях)]
      CASE SelectTypeKind=3
           lcText=[(долевая собственность)]
   ENDCASE
   .TypeText (lcText)
ENDIF
.TypeParagraph
.TypeParagraph
.Font.Bold = wdToggle
.Font.Underline = wdUnderlineNone
.ParagraphFormat.Alignment = wdAlignParagraphJustify
lcText=[именуемый в дальнейшем "Покупатель", заключили ]+;
       [настоящий договор о нижеследующем:]
.TypeText (lcText)
.TypeParagraph
IF SelectTypeFlat=1
   * Квартира отдельная
   lcText=[       1. "Продавец" передал в собственность, ]+;
   [а "Покупатель" приобрел квартиру, ]+;
   [состоящую из ]+ALLTRIM(SelectRoom)+[ общей площадью ]+;
   ALLTRIM(STR(cFlat.SquareFlat,5,1))+;
   [ кв. м., в том числе жилой ]+;
   ALLTRIM(STR(FLAT.Dwell,5,1))+[ кв. м., по адресу: ]
ELSE
   * Квартира коммунальная
   lcText=[       1. "Продавец" передал в собственность, ]+;
   [а "Покупатель" приобрел часть коммунальной квартиры ]+;
   [общей площадью ]+ALLTRIM(STR(cFlat.SquareFlat,5,1))+;
   [ кв. м., в том числе жилой ]+;
   ALLTRIM(STR(cFlat.Dwell,5,1))+[ кв. м. ]+;
   [Данная доля включает ]+ALLTRIM(SelectRoom2)+;
   [ площадью ]+ALLTRIM(STR(cFlat.SquareFlat,5,1))+;
   [ кв. м., в том числе жилой ]+;
   ALLTRIM(STR(cFlat.Dwell,5,1))+[ кв.м. и часть помещений]+;
   [ общего пользования квартиры, ]+;
   [пропорционально занимаемой жилой площади, по адресу: ]
ENDIF
.TypeText (lcText)
lcText=[г. Хабаровск, ]
SELECT cBuilding
* Порядок следования в адресе
IF cBuilding.First=.F.
   * Признак адреса стоит первым
   RightAddress=ALLTRIM(cBuilding.Sign)+[ ]+;
```

```
                ALLTRIM(cBuilding.Name)
ELSE
   * Признак адреса стоит вторым
   RightAddress=ALLTRIM(cBuilding.Name)+[ ]+;
                ALLTRIM(cBuilding.Sign)
ENDIF
lcText=lcText+RightAddress+;
       [, дом ]+ALLTRIM(SelectHouse)+[, кв. ]+;
       ALLTRIM(STR(SelectFlat))+[.]
.Font.Underline = wdUnderlineSingle
.Font.Bold = wdToggle
.TypeText (lcText)
.TypeParagraph
.Font.Bold = wdToggle
.Font.Underline = wdUnderlineNone
.ParagraphFormat.Alignment = wdAlignParagraphJustifyMed
lcText=[       2. "Покупатель" приобрел право ]+;
[собственности с момента государственной регистрации ]+;
[права в едином государственном реестре Хабаровским ]+;
[краевым учреждением юстиции.]
.TypeText (lcText)
.TypeParagraph
lcText=[       3. Права и обязанности, возникающие ]+;
[из настоящего договора, "Покупателю" разъяснены.]
.TypeText (lcText)
.TypeParagraph
lcText=[       4. В случае смерти "Покупателя" все права ]+;
[и обязанности по настоящему договору ]+;
[переходят к его наследникам на общих основаниях.]
.TypeText (lcText)
.TypeParagraph
lcText=[       5. Пользование квартирой производится ]+;
["Покупателем" применительно к Правилам ]+;
[пользования жилыми помещениями, содержания жилого дома ]+;
[и придомовой территории в РСФСР.]
IF SelectMemorial=.T.
   lcText=lcText+[ и Положению ]+;
   [об охране и использовании памятников истории и ]+;
   [культуры от 16.09.1982 № 865.]
ENDIF
.TypeText (lcText)
.TypeParagraph
.ParagraphFormat.Alignment = wdAlignParagraphJustifyMed
lcText=[       6. Настоящий договор составлен в трех ]+;
[экземплярах, из которых один выдается ]+;
["Покупателю", один - "Продавцу", один остается в ]+;
[Хабаровском краевом учреждении юстиции ]+;
[по государственной регистрации прав на недвижимое ]+;
[имущество и сделок с ним.]
.TypeText (lcText)
IF SelectMemorial=.T.
```

```
.TypeParagraph
lcText=[       7. "Покупатель" обязан заключить Охранное ]+;
[свидетельство по использованию квартиры ]+;
[в доме-памятнике со специально уполномоченным ]+;
[государственным органом охраны памятников по ]+;
[установленной форме.]
.TypeText (lcText)
ENDIF
.TypeParagraph
.TypeParagraph
.ParagraphFormat.Alignment = wdAlignParagraphCenter
.Font.Bold = wdToggle
lcText=[Адреса сторон:]
.TypeText (lcText)
.TypeParagraph
.TypeParagraph
lcText=[Подпись "Продавца"]
.ParagraphFormat.Alignment = wdAlignParagraphJustify
.Font.Bold = wdToggle
.TypeText (lcText)
.TypeParagraph
IF RECCOUNT()>1
   lcText=[Подписи "Покупателя"]
ELSE
   lcText=[Подпись "Покупателя"]
ENDIF
lcText=REPLICATE([ ],95)+lcText
.TypeText (lcText)
.TypeParagraph
.TypeParagraph
.Font.Name = "Courier new"
.Font.Size = 12
.ParagraphFormat.Alignment = wdAlignParagraphJustify
lcText=[_________________ ]
.TypeText (lcText)
.Font.Bold = wdToggle
lcText=ALLTRIM(ChiefShot)
.TypeText (lcText)
.TypeParagraph
.ParagraphFormat.Alignment = wdAlignParagraphRight
SELECT cOwners
SCAN
 lcText=LEFT(ALLTRIM(cOwners.Name),1)+[.]+;
        LEFT(ALLTRIM(cOwners.Second),1)+[.]+;
        ALLTRIM(cOwners.Family)
 LenText=LEN(ALLTRIM(lcText))
 LineText=Replicate('_',30-LenText)+[ ]
 .Font.Bold = wdToggle
 .TypeText (lineText)
 .Font.Bold = wdToggle
 .TypeText (lcText)
```

```
      .TypeParagraph
      .TypeParagraph
    ENDSCAN
ENDWITH
* Переход в начало документа
oWord.ActiveWindow.ActivePane.VerticalPercentScrolled = 0
WAIT 'Договор готов!' WINDOW NOWAIT
```

При написании текста для Microsoft Word на VBA (Visual Basic for Application) используется значительное количество системных констант. Visual Basic их «знает», а Visual FoxPro 9.0 – нет. К счастью все они собраны пользователями Visual FoxPro в отдельный файл ***Word.h*** (он находится на компакт диске).

5.3. Передача данных в Microsoft Excel

Отчеты, выполненные в конструкторе отчетов Visual FoxPro, не смотря на свою универсальность, значительно уступают таблицам Microsoft Excel в плане комфортности работы с отчетом. Рассмотрим на примере лицевого счета квартиросъемщика формирование отчета Excel из Visual FoxPro. Добавим кнопку ***Передать в Excel*** в форму ***Account*** (рис. 5.5).

Рис. 5.5. Форма лицевого счета квартиросъемщика

Код события ***Click*** кнопки ***Передать в Excel***:

```
WAIT 'Ждите! Идет передача в Microsoft Excel' WINDOW NOWAIT
* Запущен ли Microsoft Excel на компьютере?
ON ERROR oExcel=.NULL.
* В случае возникновения ошибки в следующей строке считаем,
* что объекта oExcel нет
```

```
oExcel=GetObject(,"EXCEL.Application")
IF ISNULL(oExcel)
   * Excel не запущен
   ErrorExcel=.T.            && Excel на компьютере есть
   ON ERROR ErrorExcel=.F.   && Excel на компьютере нет
   * ErrorExcel=.F. в случае возникновения ошибки
   * в следующей строке при запуске Excel
   oExcel=CREATEOBJECT("EXCEL.Application")  && Запускаем Excel
   * Вернуть назад стандартную процедуру обработки ошибок
   ON ERROR DO ERRORHND
   IF ErrorExcel=.F.
      =MESSAGEBOX('На Вашем компьютере отсутствует '+;
                  'Microsoft Excel',48,'Ошибка!')
      RETURN
   ENDIF
ELSE
  =MESSAGEBOX('Microsoft Excel уже запущен! Найдите его '+;
              'на Панели задач внизу экрана',48,'Ошибка!')
  RETURN
ENDIF
* Локальные переменные
NachAll=0      && Начислено за месяц
YmAll=0        && Уменьшено за месяц
DonaAll=0      && Доначислено за месяц
YpAll=0        && Уплачено за месяц
ReturnAll=0    && Возвращено за месяц
NachPeniAll=0  && Начислено пени за месяц
SpPeni=0       && Списано пени за месяц
UpPeni=0       && Уплачено пени за месяц
SaldoAll=0     && Сальдо на начало месяца
OstPenu=0      && Остаток пени
OstSht=0       && Остаток штрафа
* Константы Microsoft Excel
#DEFINE True .T.
#DEFINE False .F.
#DEFINE xlLandscape   2
#DEFINE xlPaperA4   9
#DEFINE xlCenter   -4108
#DEFINE xlBottom   -4107
#DEFINE xlDiagonalDown   5
#DEFINE xlDiagonalUp   6
#DEFINE xlAutomatic   -4105
#DEFINE xlNone   -4142
#DEFINE xlEdgeLeft   7
#DEFINE xlContinuous   1
#DEFINE xlThin   2
#DEFINE xlAutomaticScale   -4105
#DEFINE xlEdgeTop   8
#DEFINE xlEdgeBottom   9
#DEFINE xlEdgeRight   10
#DEFINE xlInsideVertical   11
```

```
#DEFINE xlInsideHorizontal   12
#DEFINE xlLeft   -4131
#DEFINE xlTop   -4160
* Сделать окно Microsoft Excel видимым
oExcel.application.Visible=.T.
* Размеры окна и его место на экране дисплея
* oExcel.Application.Top = 65
* oExcel.Application.Left = 10
* oExcel.Application.Width = 470
* oExcel.Application.Height = 290
* Убираем панели инструментов
* Стандартная
oExcel.Application.CommandBars("Standard").Visible = False
* Форматирование
oExcel.Application.CommandBars("Formatting").Visible = False
* Добавляем рабочую книгу
oExcel.WorkBooks.Add
* При закрытии рабочей книги без сохранения
* ошибка формироваться не будет
oExcel.DisplayAlerts=False
* Масштаб изображения 75%
oExcel.ActiveWindow.Zoom = 75
* Заголовок окна Excel
oExcel.Caption=[Лицевой счет]
* Ориентация альбомная поля по 1.5 см Бумага А4
oExcel.ActiveSheet.PageSetup.LeftMargin=42
oExcel.ActiveSheet.PageSetup.RightMargin=42
oExcel.ActiveSheet.PageSetup.TopMargin =42
oExcel.ActiveSheet.PageSetup.BottomMargin=42
oExcel.ActiveSheet.PageSetup.Orientation = xlLandscape
oExcel.ActiveSheet.PageSetup.PaperSize = xlPaperA4
* Заголовок отчета
oExcel.Range("C1").Select
oExcel.ActiveCell.Font.Bold = True
oExcel.ActiveCell.Font.Size = 9
oExcel.ActiveCell.FormulaR1C1 =[Лицевой счет квартиросъемщика]
oExcel.Range("B2").Select
oExcel.ActiveCell.VerticalAlignment = xlTop
oExcel.ActiveCell.Font.Size = 7
AddressFlat=[Адрес квартиры: г. Хабаровск, ]
SELECT cBuilding
* Порядок следования в адресе
IF cBuilding.First=.F.
   * Признак адреса стоит первым
   RightAddress=ALLTRIM(cBuilding.Sign)+[ ]+;
                ALLTRIM(cBuilding.Name)
ELSE
    * Признак адреса стоит вторым
    RightAddress=ALLTRIM(cBuilding.Name)+[ ]+;
                 ALLTRIM(cBuilding.Sign)
ENDIF
```

```
AddressFlat=AddressFlat+RightAddress+[, дом ]+;
      ALLTRIM(SelectHouseAddress)+[, кв. ]+;
      ALLTRIM(STR(SelectFlat))+[.]
oExcel.ActiveCell.FormulaR1C1 =AddressFlat
oExcel.Range("A3").Select
oExcel.ActiveCell.VerticalAlignment = xlBottom
oExcel.ActiveCell.Font.Size = 4
oExcel.ActiveCell.FormulaR1C1 =[Copyright © 2006 ]+;
       [Программный комплекс "Учебный пример Real Estate" ]+;
       [написал Гурвиц Геннадий Александрович тел. 35-91-33 ]+;
       [Использованы продукты Microsoft Corporation: ]+;
       [Microsoft Visual FoxPro 9.0 Service Pack 2, Microsoft ]+;
       [Visual Basic for applications, Microsoft Office 2003]
* Шрифт и центровка для всей таблицы
* oExcel.Cells.HorizontalAlignment = xlCenter
* oExcel.Cells.VerticalAlignment = xlBottom
* Надписи колонок
oExcel.Range("A4").Select
oExcel.ActiveCell.FormulaR1C1 = "Дата"
oExcel.Range("B4").Select
oExcel.ActiveCell.FormulaR1C1 = "Операция"
oExcel.Range("C4").Select
oExcel.ActiveCell.FormulaR1C1 = "Начислено"
oExcel.Range("D4").Select
oExcel.ActiveCell.FormulaR1C1 = "Уменьшено"
oExcel.Range("E4").Select
oExcel.ActiveCell.FormulaR1C1 = "Доначис."
oExcel.Range("F4").Select
oExcel.ActiveCell.FormulaR1C1 = "Уплачено"
oExcel.Range("G4").Select
oExcel.ActiveCell.FormulaR1C1 = "Возврат"
oExcel.Range("H4").Select
oExcel.ActiveCell.FormulaR1C1 = "Сальдо"
oExcel.Range("I4").Select
oExcel.ActiveCell.FormulaR1C1 = "Нач. пени"
oExcel.Range("J4").Select
oExcel.ActiveCell.FormulaR1C1 = "Количество дней пени"
oExcel.Range("K4").Select
oExcel.ActiveCell.FormulaR1C1 = "Сп. пени"
oExcel.Range("M4").Select
oExcel.ActiveCell.FormulaR1C1 = "Ост.пени"
oExcel.Range("N4").Select
oExcel.ActiveCell.FormulaR1C1 = "Ост.штр"
* Устанавливаем ширину колонок
oExcel.Columns("A:A").ColumnWidth = 8
oExcel.Columns("B:B").ColumnWidth = 29
oExcel.Columns("C:C").ColumnWidth = 7
*- Фрагмент пропущен
oExcel.Columns("N:N").ColumnWidth = 6
* Для всех заголовков столбцов жирный шрифт
oExcel.Rows("4:4").Font.Bold = True
```

```
oExcel.Rows("4:4").RowHeight = 15
oExcel.Rows("4:4").Font.Size = 7
oExcel.Rows("4:4").VerticalAlignment = xlCenter
oExcel.Rows("4:4").Interior.ColorIndex = 8
oExcel.Rows("4:4").HorizontalAlignment = xlCenter
* Обводим линиями шапку таблицы
oExcel.RANGE("A4:N4").Select
oExcel.Selection.Borders(xlDiagonalUp).LineStyle = xlNone
* Левые вертикальные линии
With oExcel.Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
EndWith
* Верхние горизонтальные линии
With oExcel.Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
EndWith
* Нижние горизонтальные линии
With oExcel.Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
EndWith
* Правые вертикальные линии
With oExcel.Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
EndWith
* Правая вертикальная в последней ячейке
With oExcel.Selection.Borders(xlEdgeRight)
     .LineStyle = xlContinuous
     .Weight = xlThin
     .ColorIndex = xlAutomatic
EndWith
* Открытие таблицы-выборки лицевого счета
IF .NOT. USED('pAccount')
   USE pAccount in 0
ENDIF
SELECT pAccount
nRow=5  && Дальнейший вывод с пятой строки
SignSaldoStart=0
SldRussia=0
SldKray=0
SCAN
 DO CASE
  CASE pAccount.Contents=[Сальдо старт]
       SignSaldoStart=1
```

```
      nRow=nRow+1 && Номер текущей строчки
      DO WRITE    && Вывод строки счета. Находится в FileProc
      oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
                ALLTRIM(STR(nRow,3))).Font.Size = 7
      oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
                ALLTRIM(STR(nRow,3))).RowHeight = 13.2
      oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
           ALLTRIM(STR(nRow,3))).HorizontalAlignment = xlCenter
      oExcel.Range([A]+ALLTRIM(STR(nRow,3))+[:]+;
                [N]+ALLTRIM(STR(nRow,3))).Select
      * Левые вертикальные линии
      With oExcel.Selection.Borders(xlEdgeLeft)
              .LineStyle = xlContinuous
              .Weight = xlThin
              .ColorIndex = xlAutomatic
      EndWith
      * Верхние горизонтальные линии
      With oExcel.Selection.Borders(xlEdgeTop)
              .LineStyle = xlContinuous
              .Weight = xlThin
              .ColorIndex = xlAutomatic
      EndWith
      * Нижние горизонтальные линии
      With oExcel.Selection.Borders(xlEdgeBottom)
              .LineStyle = xlContinuous
              .Weight = xlThin
              .ColorIndex = xlAutomatic
      EndWith
      * Правые вертикальные линии
      With oExcel.Selection.Borders(xlInsideVertical)
              .LineStyle = xlContinuous
              .Weight = xlThin
              .ColorIndex = xlAutomatic
      EndWith
      * Правая вертикальная в последней ячейке
      With oExcel.Selection.Borders(xlEdgeRight)
              .LineStyle = xlContinuous
              .Weight = xlThin
              .ColorIndex = xlAutomatic
      EndWith
  CASE pAccount.Contents=[В т.ч Аренда]
         SldRussia=pAccount.Saldo
  CASE pAccount.Contents=[В т.ч НДС]
      SldKray=pAccount.Saldo
      IF SignSaldoStart=1
         * Добавление строчки после Сальдо старт
         nRow=nRow+1
         oExcel.Range([C]+ALLTRIM(STR(nRow,3))).Select
         oExcel.ActiveCell.FormulaR1C1=;
                       [Начальное сальдо (Оплата):  ]+;
                       ALLTRIM(STR(SldRussia,13,2))+;
```

```
                [ руб.     Начальное сальдо (НДС):  ]+;
                ALLTRIM(STR(SldKray,13,2))+[ руб.]
   oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
           ALLTRIM(STR(nRow,3))).Font.Size = 7
   oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
           ALLTRIM(STR(nRow,3))).Interior.ColorIndex = 35
   nRow=nRow+1
   oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
           ALLTRIM(STR(nRow,3))).Interior.ColorIndex = 35
   StartRow=nRow
ENDIF
IF SignSaldoStart=0
   oExcel.Range([A]+ALLTRIM(STR(StartRow+1,3))+[:N]+;
          ALLTRIM(STR(nRow,3))).Select
   oExcel.Selection.Borders(xlDiagonalUp).LineStyle=;
                                                    xlNone
   With oExcel.Selection
           .HorizontalAlignment = xlCenter
   EndWith
   With oExcel.Selection.Font
           .Size = 7
   EndWith
   * Левые вертикальные линии
   With oExcel.Selection.Borders(xlEdgeLeft)
           .LineStyle = xlContinuous
           .Weight = xlThin
           .ColorIndex = xlAutomatic
   EndWith
   * Верхние горизонтальные линии
   With oExcel.Selection.Borders(xlEdgeTop)
           .LineStyle = xlContinuous
           .Weight = xlThin
           .ColorIndex = xlAutomatic
   EndWith
   * Нижняя горизонтальная линия в последней строке
   With oExcel.Selection.Borders(xlEdgeBottom)
           .LineStyle = xlContinuous
           .Weight = xlThin
           .ColorIndex = xlAutomatic
   EndWith
   * Правые вертикальные линии
   With oExcel.Selection.Borders(xlInsideVertical)
           .LineStyle = xlContinuous
           .Weight = xlThin
           .ColorIndex = xlAutomatic
   EndWith
   * Правая вертикальная в последней ячейке
   With oExcel.Selection.Borders(xlEdgeRight)
           .LineStyle = xlContinuous
           .Weight = xlThin
           .ColorIndex = xlAutomatic
```

```
        EndWith
        * Нижние горизонтальные линии в каждой строке
        * Не выводить, если строчка всего одна
        IF StartRow+1#nRow
              With oExcel.Selection.Borders(xlInsideHorizontal)
                .LineStyle = xlContinuous
                .Weight = xlThin
                .ColorIndex = xlAutomatic
              EndWith
         ENDIF
         * Добавление строчки после Сальдо
         nRow=nRow+1
         oExcel.Range([B]+ALLTRIM(STR(nRow,3))).Select
         oExcel.ActiveCell.FormulaR1C1=[Итого за месяц]
         oExcel.Range([C]+ALLTRIM(STR(nRow,3))).Select
         oExcel.ActiveCell.FormulaR1C1=NachAll
         * - Фрагмент пропущен
         oExcel.Range([L]+ALLTRIM(STR(nRow,3))).Select
         oExcel.ActiveCell.FormulaR1C1=UpPeni
         oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
                ALLTRIM(STR(nRow,3))).Font.Size = 7
         oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
                ALLTRIM(STR(nRow,3))).Interior.ColorIndex = 35
         oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
          ALLTRIM(STR(nRow,3))).HorizontalAlignment = xlCenter
         * Еще одна строчка сводных данных
         nRow=nRow+1
         StartRow=nRow
         oExcel.Range([A]+ALLTRIM(STR(nRow,3))).Select
         oExcel.ActiveCell.FormulaR1C1=[На начало месяца]+;
         [ общее сальдо составляет:  ]+;
         ALLTRIM(STR(SaldoAll,13,2))+[ руб. В т.ч]+;
         [ Оплата:  ]+ALLTRIM(STR(SldRussia,13,2))+;
         [ руб.  В т.ч  НДС:  ]+ALLTRIM(STR(SldKray,13,2))+;
         [ руб. ]+[   Остаток пени: ]+;
         ALLTRIM(STR(OstPenu,13,2))+[ руб. Остаток штрафа: ]+;
         ALLTRIM(STR(OstSHT,13,2))+[ руб.]
         oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
                ALLTRIM(STR(nRow,3))).Font.Size = 7
         oExcel.Rows(ALLTRIM(STR(nRow,3))+[:]+;
             ALLTRIM(STR(nRow,3))).Interior.ColorIndex = 35
       ENDIF
  CASE pAccount.Contents=[Сальдо      ]
       SignSaldoStart=0
       nRow=nRow+1
       NachPeniAll=NachPeniAll+pAccount.Npe   && Начислено пени
       SaldoAll=pAccount.Saldo     && Сальдо на начало месяца
       OstPenu=pAccount.OstPE      && Остаток пени
       OstSht=pAccount.OstSH       && Остаток штрафа
```

```
        DO WRITE  && Вывод строки счета
    OTHERWISE
        nRow=nRow+1
        DO WRITE  && Находится в процедурном файле FileProc
        NachAll=NachAll+pAccount.Nach          && Начислено
        YmAll=YmAll+pAccount.YmYB              && Уменьшено
        * Фрагмент пропущен
        UpPeni=UpPeni+pAccount.RaYpvo          && Уплачено пени
    ENDCASE
ENDSCAN
oExcel.Range("A1").Select  && Переход в начало отчета
* Устанавливаем защиту рабочего листа и книги
ProtectList=[oExcel.ActiveSheet.Protect("]+TIME()+[")]
ProtectBook=[oExcel.ActiveWorkbook.Protect("]+TIME()+[")]
&ProtectList
&ProtectBook
WAIT 'Таблица готова!' WINDOW NOWAIT
```

Вид готового отчета показан на рисунке. 5.6.

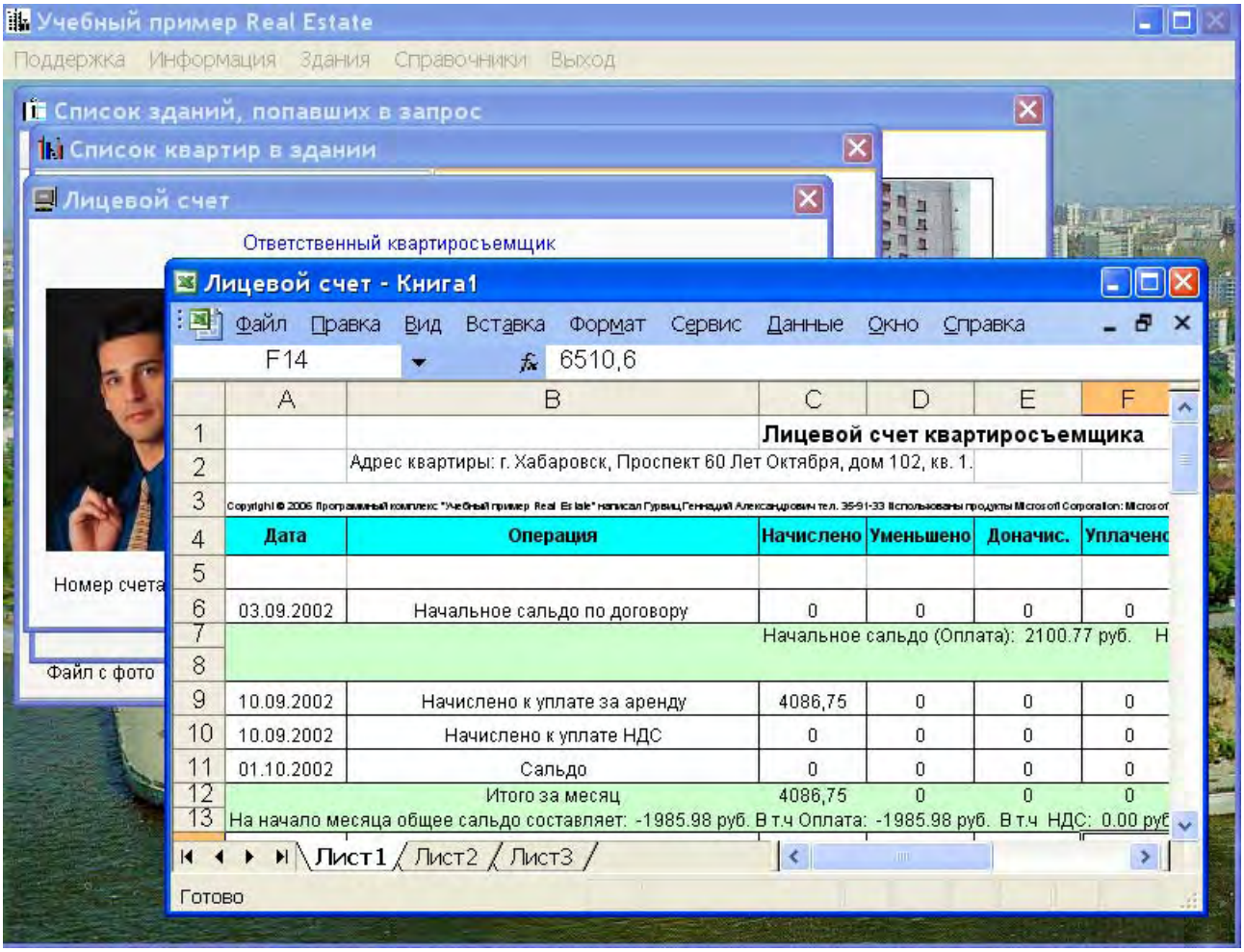


Рис. 5.6. Окончательный вид отчета, переданного в Microsoft Excel

6. СОЗДАНИЕ СИСТЕМЫ ОПЕРАТИВНОЙ СПРАВКИ

Существует множество программных продуктов, предназначенных для разработки собственной системы оперативной справки. Рассмотрим один из них – HTML Help Workshop. Это самостоятельный продукт фирмы Microsoft. Его версия 1.1 входила в состав Visual FoxPro 6.0. Сейчас он не является составной частью Visual FoxPro. Самую последнюю версию можно найти на Web-сервере http://www.microsoft.com/workshop/author/htmlhelp.

В этом разделе рассмотрена работа с HTML Help Workshop 1.3. Обязательно прочтите файл ReadMe, входящий в состав продукта. До сих пор в HTML Help Workshop имеется ряд проблем. В этом файле разработчики разъясняют способы их решения.

Для отображения системы оперативной справки HTML Help Workshop использует Internet Explorer. После компиляции системы справки HTML Help Workshop создает файл с расширением ***.chm***. Это файл контекстно-зависимой справки. Находясь в любом месте программного комплекса, пользователь может получить помощь, нажав клавишу F1.

Рассмотрим последовательно все этапы создания файла справки к нашему приложению: ***RealEstate.chm***.

Этап 1. Для каждой страницы оперативной справки создайте отдельный HTML-файл. Все страницы поместите в отдельную папку с именем HTML папки HELP. Для создания HTML-файлов можно использовать Microsoft Word. Побеспокойтесь о фоновом рисунке и фоновом звуке каждой страницы. Это позволит качественно поднять уровень оформления системы справки с минимальными затратами. Для назначения фона страницы в главном меню Microsoft Word выберите пункт ***Формат***, в появившемся меню – пункт ***Фон***. Появится окно. Выберите в нем второй пункт ***Способы заливки***. Следующее окно даст возможность назначить фоновый рисунок из коллекции Microsoft Word или выбрать из файлов рисунков, имеющихся в распоряжении разработчика. Для назначения фонового звука страницы-мелодии, которая будет звучать все время, пока пользователь видит ее содержимое – в главном меню Microsoft Word выберите пункт ***Вид***, в появившемся меню – пункт ***Панели инструментов***. В раскрывшемся списке панелей сделайте щелчок по пункту ***Web-компоненты***. Панель появится на экране (рис. 6.1). Выберите пиктограмму Фоновый звук. В появившемся окне назначьте файл.

Рис. 6.1. Панель Веб-компоненты Microsoft Word

Созданную страницу сохраните в формате HTML. Для этого в главном меню Microsoft Word выберите пункт ***Файл***, в появившемся меню – пункт ***Сохранить как Веб-страницу***. Укажите тип файла: Веб-страница. Обратите внимание! По умолчанию в окне указан тип: Веб-страница в одном файле. Выполните правильное назначение. Установите гиперссылки между HTML-файлами. Ссылок должно быть столько, сколько необходимо для толковой работы.

Этап 2. Запустите HTML Help Workshop. В его главном окне выберите пункт ***File***. В открывшемся меню пункт ***New***. Появится окно ***New***. Выберите в нем первый пункт ***Project*** и щелкните по кнопке ***OK***. HTML Help Workshop предложит ввести имя проекта и включить в него уже имеющиеся у разработчика компоненты проекта. У нас пока ничего нет. Ограничимся именем: ***RealEstate.hhp***.

Этап 3. Откройте файл проекта ***RealEstate.hhp***. На первой вкладке с именем ***Project*** увидите семь пиктограмм (рис. 6.2).

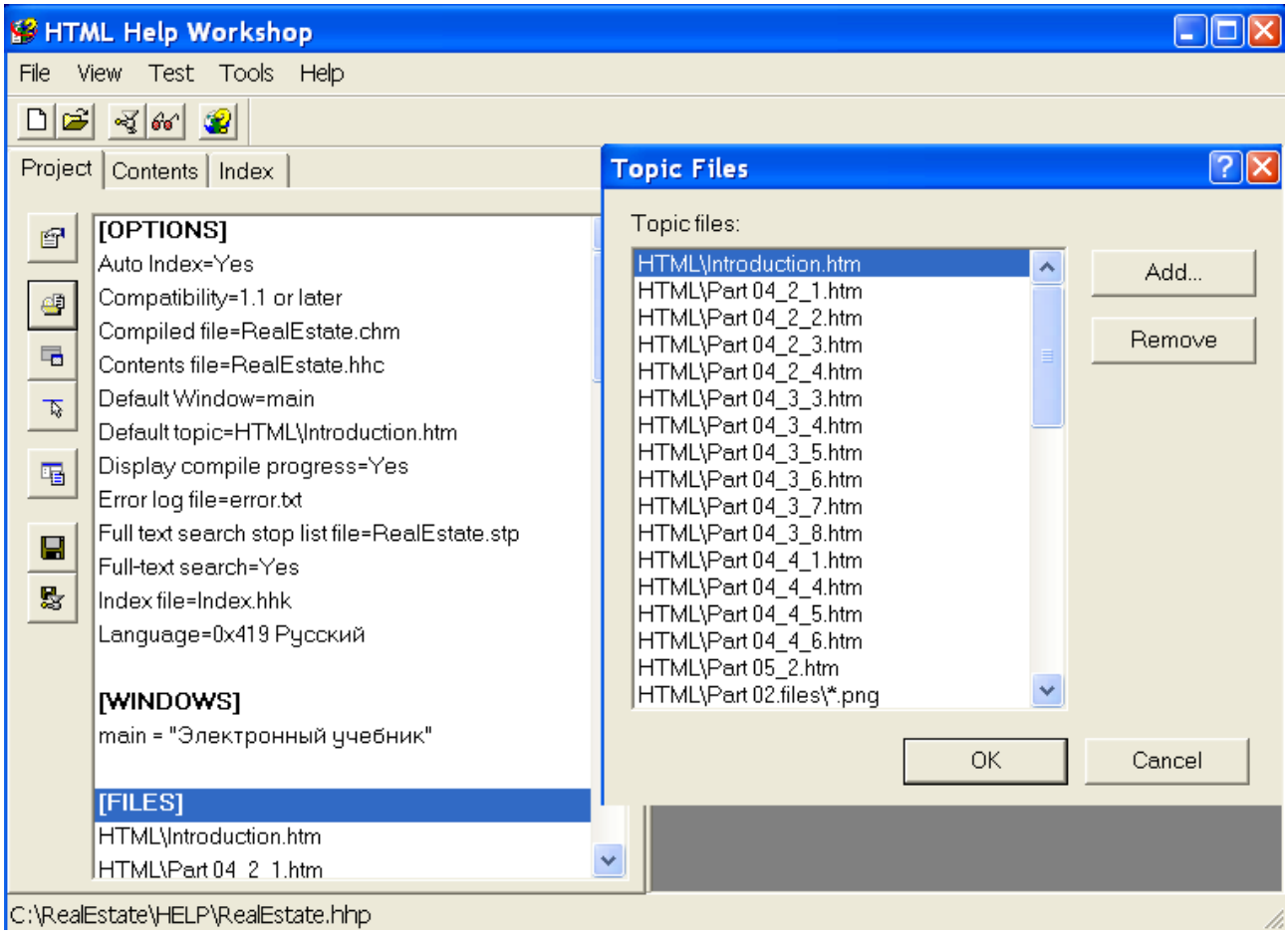


Рис. 6.2. Проект RealEstate.hhp в окне HTML Help Workshop

Выберите вторую пиктограмму Add/Remove topic files (включение в проект HTML-страниц). Добавьте в проект все созданные на первом этапе HTML-страницы.

Определимся с первой пиктограммой Change project options (Определение параметров проекта).

Щелчок по ней откроет окно ***Options*** с четырьмя вкладками. На вкладке General (Общие) определите:

- заголовок окна справки;
- файл страницы, которая будет отображена в окне при первом запуске системы оперативной справки;
- язык и шрифт.

На вкладке ***Files*** укажите название и расположение файла скомпилированной справки, а также файлов с содержанием тем справки и с указателями.

Вид вкладки ***Compiler*** (Компилятор) показан на рис. 6.3. Отметьте флажками необходимые объекты и действия.

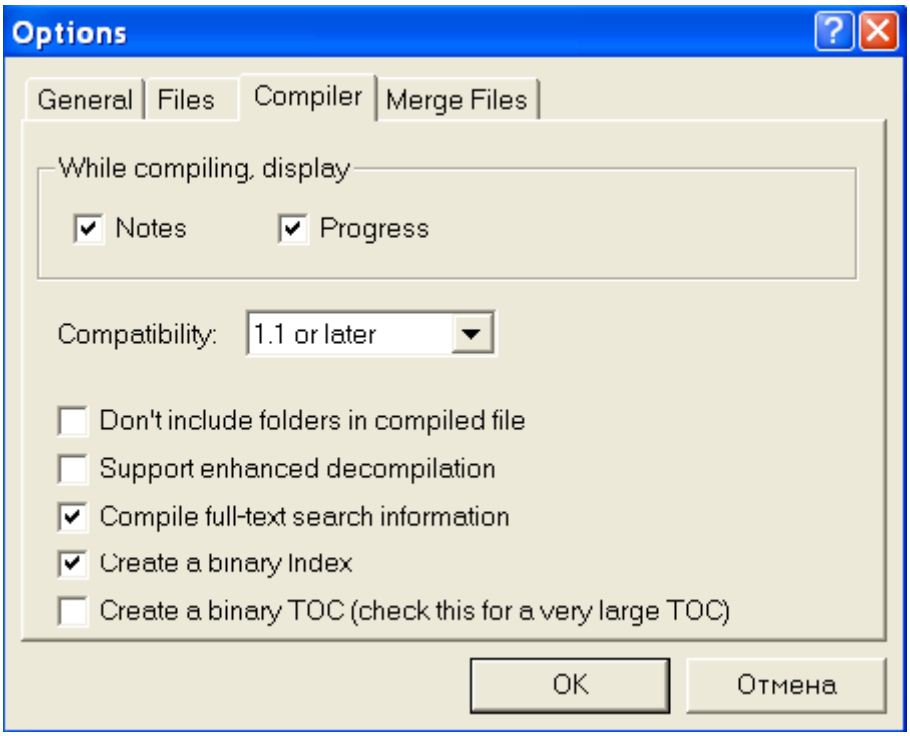


Рис. 6.3. Вкладка опций компилятора

Этап 4. Для создания содержания справочной системы перейдем на вторую вкладку HTML Help Workshop с названием ***Contents*** (Содержание). Она содержит 11 пиктограмм (рис. 6.4). Первая пиктограмма ***Contents properties*** предназначена для выбора внешнего вида вкладки содержания справочной системы. Выберите тип заголовка (значок папки или раскрытой книги). Можно также назначить свои значки.

Для добавления заголовка щелкните по второй пиктограмме ***Insert a heading***. Для добавления страницы предназначена третья пиктограмма

 ***Insert a page***. Содержание тем справочной системы можно расположить в иерархическом виде. Допускается добавление заголовков нескольких уровней вложенности.

Для изменения уровня заголовка или страницы воспользуйтесь пиктограммами со стрелками.

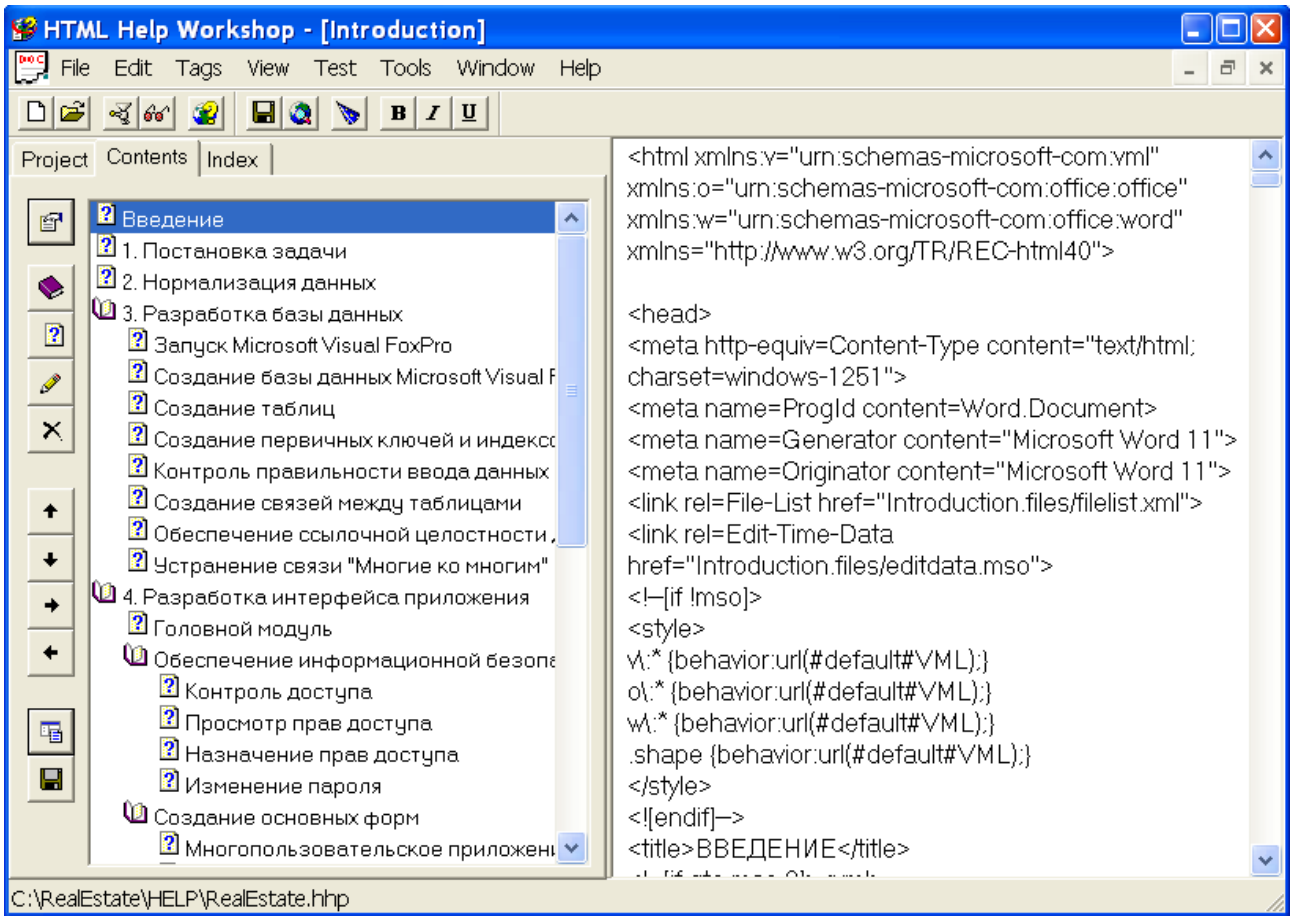


Рис. 6.4. Вторая вкладка HTML Help Workshop с названием ***Contents*** (Содержание)

После четырех этапов мы создали все основные компоненты справки. Отсутствуют только средства контекстно-зависимой справочной системы. Однако даже в таком виде наша оперативная справка работоспособна. Откомпилируйте ее, щелкнув по последней пиктограмме  ***Save all files and compile*** первой вкладки ***Project*** продукта HTML Help Workshop. При запуске на выполнение файла ***RealEstate.chm*** всегда будет отображаться назначенная стартовой HTML-страница ***Introduction*** (Введение).

Для создания справочной системы, работающей совместно с приложением, необходимо связать объекты Visual FoxPro нашего приложения с файлами HTML-страниц. Как правило, это сами формы и объекты, расположенные в них. Посредниками в этом деле выступают псевдонимы тем,

константы ***HelpContextID*** (индексы тем) и файл связи ***RealEstate.h***. Вид цепочки связи показан на рис. 6.5.

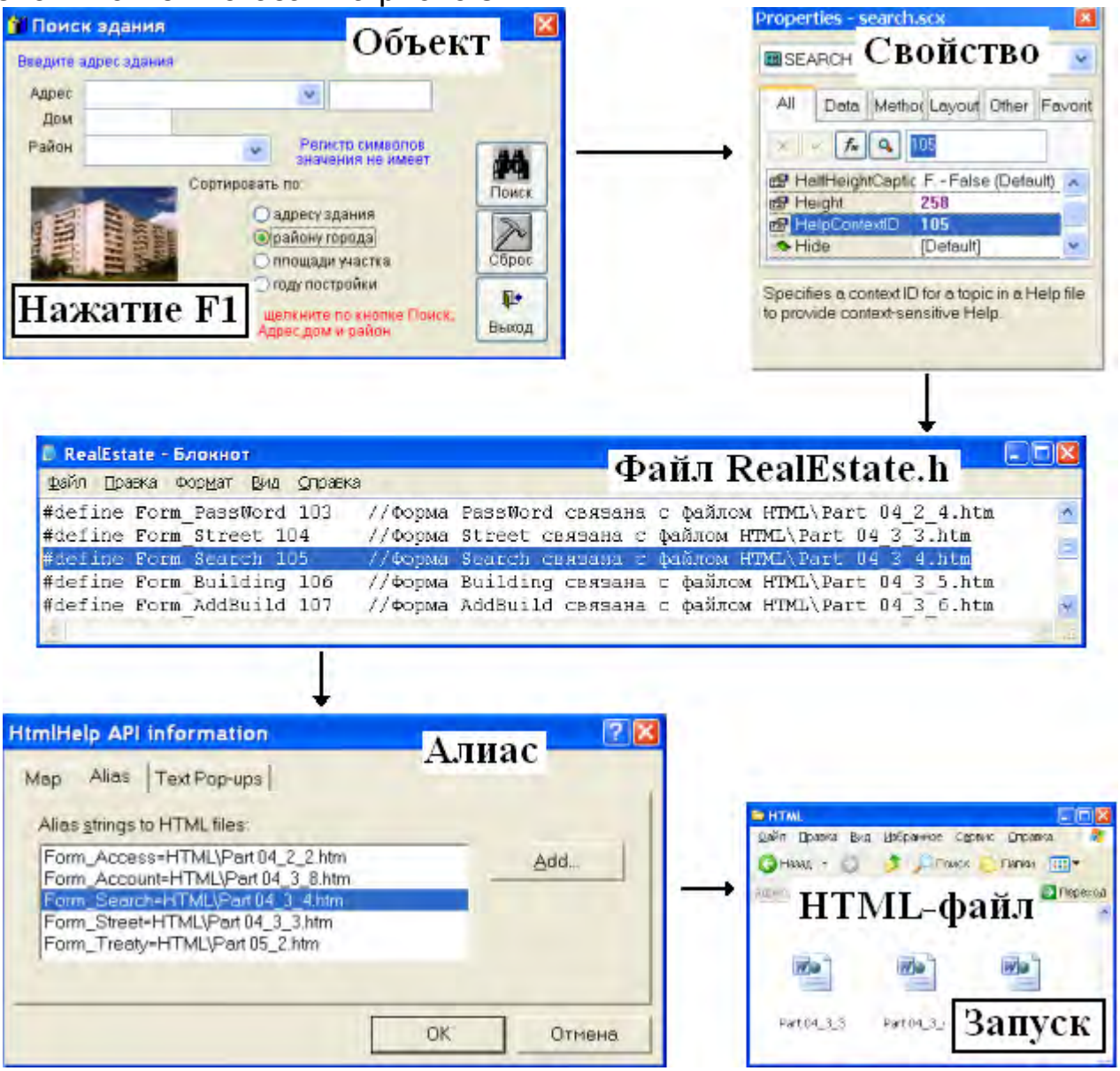


Рис. 6.5. Связь между нажатием клавиши F1 и запуском нужной HTML-страницы

Этап 5. Для назначения псевдонимов тем, необходимых для создания контексно-зависимой справочной системы, откройте окно HtmlHelp API information (рис. 6.6), выбрав четвертую пиктограмму  первой вкладки ***Project*** продукта HTML Help Workshop. Перейдите на вторую вкладку ***Alias*** (Псевдоним). После нажатия ***Add*** (Добавить) откроется окно ***Alias***. Укажите в нем Псевдоним, имя файла и комментарий. Имя псевдонима вводится с клавиатуры. Имя HTML-файла выбирается из списка. Если файла в списке нет, вернитесь к третьему этапу и добавьте в список файлов отсутствующий.

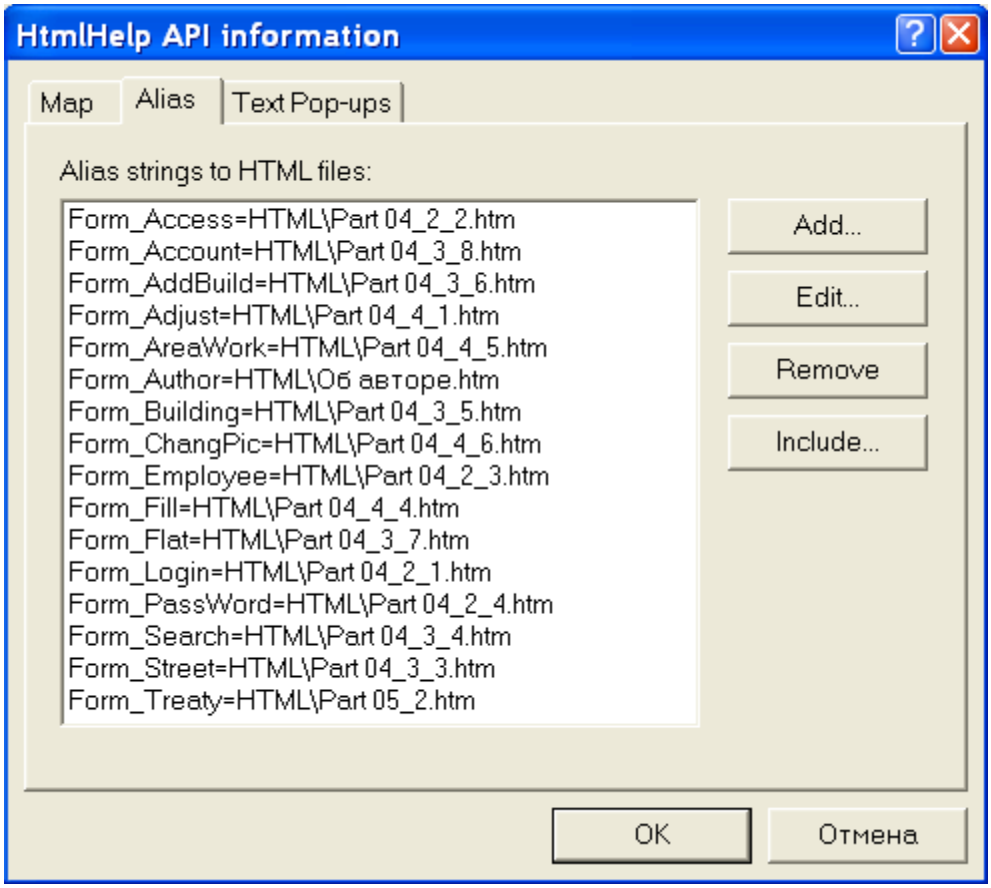


Рис. 6.6. Вторая вкладка окна HtmlHelp API Information

Имя псевдонима HTML-файла лучше всего назначать осмысленно. Лениться не следует. Даже в небольшом файле справки можно запутаться с именами объектов и соответствующими им файлами. Про имена типа ***A1*** или ***Aaa*** следует забыть. Два примера имени псевдонима:

```
Form_Building                  (Форма просмотра списка зданий)
Form_Building_Combo1_Street    (Список улиц в форме зданий)
```

Этап 6. Разберемся с индексами тем (свойство ***HelpContextID*** объекта Visual FoxPro). Запустите Microsoft Visual FoxPro. Выберите объект. Это может быть Form, CheckBox, ComboBox, CommandButton, EditBox, Grid, Image, Label, ListBox, Page и др. Установите в качестве значения этого свойства выбранного объекта уникальное число. Сколько объектов вы хотите связать с файлами страниц – столько чисел. Каких – не важно, главное, чтобы они не повторялись. В примере ***RealEstate*** они пронумерованы по порядку, начиная с сотни.

Добавьте в головной модуль строчку для вызова контекстно-зависимой справочной системы:

```
SET HELP TO RealEstate.chm
```

Добавьте в текст процедуры завершения работы программного комплекса ***STOP*** (находится в процедурном файле ***FileProc*** разд. 7) строку вызова контекстно-зависимой справочной системы по продукту Visual FoxPro:

```
SET HELP TO
```

Этап 7. Назначим связи между псевдонимами и индексами тем. Информация об этом хранится в Map-файле. Это простой текстовый файл с расширением ***.h***. В нашем примере – ***RealEstate.h***. Для его создания применим стандартную программу Windows – ***Блокнот***. Несколько строчек из файла ***RealEstate.h***:

```
#define Form_Login 100      //Форма Login связана с Файлом HTML\Part 04_2_1.htm
#define Form_Access 101     //Форма Access связана с файлом HTML\Part 04_2_2.htm
#define Form_Street 104     //Форма Street связана с файлом HTML\Part 04_3_3.htm
```

Комментарий после числа, который начинается со знака ***//***, можно вообще не писать.

Осталось включить имя этого файла в проект ***RealEstate.hhp***. Запустите HTML Help Workshop. Откройте проект. Выберите четвертую пиктограмму  первой вкладки ***Project***. Появится окно HtmlHelp API information (рис. 6.7). Добавьте Map-файл ***RealEstate.h.***

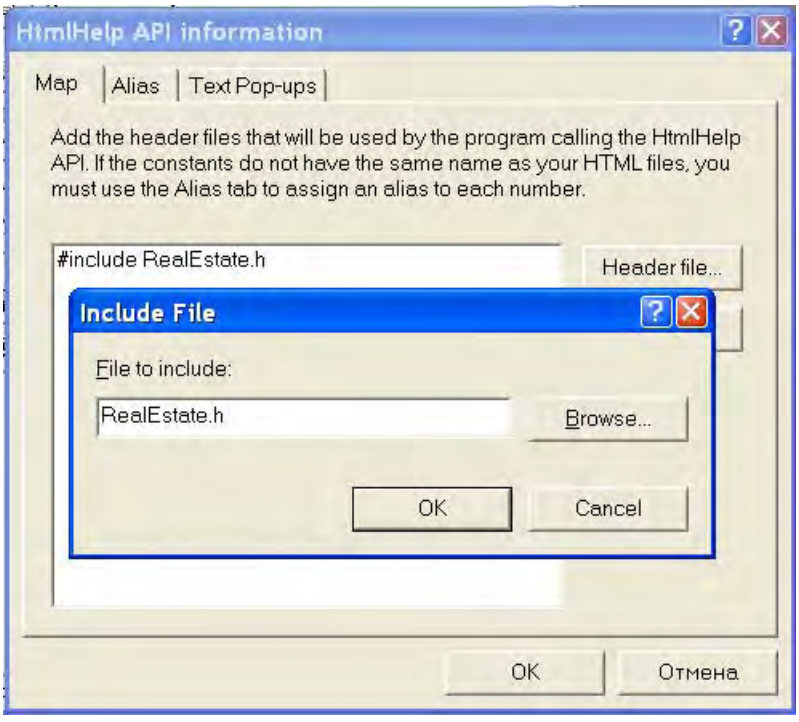


Рис. 6.7. Первая вкладка окна HtmlHelp API Information

Этап 8. Откомпилируйте файл справки, щелкнув по последней пиктограмме  ***Save all files and compile*** первой вкладки ***Project*** (рис. 6.2).

Предупреждение. HTML Help Workshop версии 1.3 (Сборка 4.74.8702.0) не включает в проект файлы рисунков Microsoft Word 2003 с расширением ***.png*** (Portable Network Graphics). PNG — это формат графических файлов, поддерживаемый многими веб-обозревателями. Он обеспечивает сжатие и хранение графических изображений без потери графических данных при распаковке изображения. Формат PNG позволяет записывать сведения о прозрачности изображений, а также сведения для управления яркостью изображения на различных компьютерах. Данный формат используется для сохранения самых разных графических данных, от небольших изображений (таких, как маркеры списков и объявления) до высококачественных фотографий. PNG-файлы хранятся в папках поддержки HTML-страниц. Для решения этой проблемы откройте файл проекта ***RealEstate.hhp*** в программе ***Блокнот*** и добавите строчки в любое место раздела [FILES]:

```
[FILES]
HTML\Part 02.files\*.png
HTML\Part 03_3.files\*.png
HTML\Part 04_1.files\*.png
HTML\Part 04_2.files\*.png
HTML\Part 04_2_4.files\*.png
HTML\Part 04_3_3.files\*.png
HTML\Part 05_2.files\*.png
HTML\Part 06.files\*.png
```

7. РАБОТА С ПРОЦЕДУРНЫМИ ФАЙЛАМИ

Процедуры языка Visual FoxPro, написанные разработчиком, удобно хранить в процедурных файлах. В них процедуры располагают последовательно, ничем не отделяя друг от друга. Подключение к головному модулю демонстрирует следующий текст, расположенный в его начале:

```
* Подключение первого файла, содержащего процедуры
* FileProc - имя файла
SET PROCEDURE TO FileProc
* Подключение второго и последующих
* SET PROCEDURE TO <имя файла> ADDITIVE
* SET PROCEDURE TO <имя файла> ADDITIVE
```

Учебный программный комплекс ***RealEstate*** имеет в своем составе один процедурный файл с именем ***FileProc***. Его текст с комментариями и с некоторыми сокращениями приведен ниже:

```
************************************************************
* Процедурный файл к учебному примеру Real Estate          *
```

```
* Содержит ряд процедур, необходимых для работы                    *
* программного комплекса                                           *
*********************************************************************

PROCEDURE STOP             && Завершение работы комплекса
* Возвращение настроек однопользовательского режима
* Это режим разработчика программного комплекса
  ON KEY LABEL ESCAPE       && Вернуть к действию клавишу ESC
  DEACTIVATE WINDOW ALL
  RELEASE WINDOW ALL
  SET RESOURCE ON           && Сохранять настройки
                            && в таблице Foxuser.dbf
  _SCREEN.BACKCOLOR=RGB(255,255,255) && Вернуть стандартный фон
  _SCREEN.WINDOWSTATE=2              && Развернуть во весь экран
  _SCREEN.MINBUTTON=.T.              && Есть кнопка свертывания
  _SCREEN.MAXBUTTON=.T.              && Есть кнопка развертывания
  _SCREEN.PICTURE=[]                 && Убрать картинку
  _SCREEN.ICON=[]                    && Убрать иконку
  * Вернуть заголовок экрана
  _SCREEN.CAPTION=[Microsoft Visual FoxPro]
  * Вернуть файл справки Visual FoxPro
  SET HELP TO
  ON SHUTDOWN               && Спец. команды, выполняющиеся
                            && после выхода из Visual FoxPro
  RELEASE ALL EXTENDED      && Удалить из памяти переменные
  SET EXCLUSIVE ON          && Монопольный доступ
  SET DELETED OFF           && Удаленные записи видимы
  SET TALK ON               && Выводить результаты
                            && выполнения команд на экран
  SET SAFETY ON             && Отображать предупреждающее сообщение
                            && перед перезаписью файла
  SET STATUS BAR ON         && Отображать нижнюю строку экрана
  ON ERROR                  && Вернуть стандартную процедуру
                            && обработки ошибок
  SET SYSMENU TO DEFAULT    && Вернуть системное меню FoxPro
  CLOSE DATABASES ALL       && Закрыть базы данных
  CLOSE TABLES ALL          && Закрыть свободные таблицы
  CANCEL                    && Завершение выполнения Fox-программы
RETURN

PROCEDURE REALQUIT          && Подтверждение выхода из FoxPro
LOCAL lnMsgResult
lnMsgResult=;
     MESSAGEBOX('Вы действительно хотите выйти из программы?',;
                20,' Выход из программы ')
IF lnMsgResult=6       && Кнопка Да
    QUIT               && Завершение работы среды Visual FoxPro
ENDIF
RETURN

PROCEDURE ERRORHND             && Процедура обработки ошибок
```

```
ERRORCODE=ERROR()              && Определение кода ошибки
ON KEY LABEL F10               && Отмена действовавших назначений F10
* Это сообщение повторяется при многих ошибках
MsgLevel='Через три секунды запрос будет повторен.'+CHR(13)+;
    CHR(13)+'Если это сообщение быстро не исчезло,'+CHR(13)+;
    'обратитесь к системному администратору.'+CHR(13)+;
    'Возможно нарушена связь с сервером.'
DO CASE
   CASE ERRORCODE=108
        ON KEY LABEL F10 DO CANCELWAIT
        WAIT  'Ждите! Нет доступа к внешней памяти.'+CHR(13)+;
              'Или установлен статус только для чтения!'+;
               CHR(13)+MsgLevel+CHR(13) WINDOW NOWAIT
        DO NETDELAY      &&  Ожидание
        ON KEY LABEL F10
        * Возвращает управление и выполняет вызов повторно
        RETRY
   CASE ERRORCODE=109
        ON KEY LABEL F10 DO CANCELWAIT
        WAIT  'Ждите! С записью работает Ваш коллега.';
        WINDOW NOWAIT
        DO NETDELAY      &&  Ожидание
        ON KEY LABEL F10
   CASE ERRORCODE=130
        ON KEY LABEL F10 DO CANCELWAIT
        WAIT  'Ждите! Запись не заблокирована.'+CHR(13)+MsgLevel;
        WINDOW NOWAIT
        DO NETDELAY      &&  Ожидание
        ON KEY LABEL F10
        * Возвращает управление и выполняет вызов повторно
        RETRY
   CASE ERRORCODE=202
        =MESSAGEBOX('Папка C:\WINNT\TEMP отсутствует!',;
        48,' Внимание!')
        * Эту папку создает программный комплекс
        * для временного хранения таблиц-выборок
        RETURN TO MASTER
   CASE ERRORCODE=1002
        lnMsgResult=MESSAGEBOX('Отсутствует дискета.'+;
               'Повторить чтение?',20,' Внимание!')
        IF lnMsgResult=6       && Кнопка Да
           RETRY
        ELSE
           RETURN TO MASTER
        ENDIF
   CASE ERRORCODE=1105
        =MESSAGEBOX(' Нет прав доступа к файлу на запись. '+;
        'Или без предупреждения выключен сервер, или '+;
        'поврежден сетевой кабель!',;
        48,' Ошибка!')
        DO STOP
```

```
   CASE ERRORCODE=1585
        lnMsgResult=;
            MESSAGEBOX('Пока Вы занимались корректировкой, '+;
            'эту запись уже кто-то изменил. '+;
            'Записать Ваши изменения поверх?',20,' Внимание!')
        IF lnMsgResult=6      && Кнопка Да
            =TABLEUPDATE(.T.,.T.)
        ELSE
            =TABLEREVERT()
        ENDIF
   CASE ERRORCODE=1961
        * Создаваемая папка уже существует
        WAIT 'Папка для временных файлов C:\WINNT\TEMP имеется!'+;
             CHR(13)+' Можно работать!' WINDOW NOWAIT
   CASE ERRORCODE=1884
         * Запись с таким номером уже существует
         =MESSAGEBOX('Запись с таким номером уже существует. '+;
                     'Возможно как помеченная на удаление и '+;
                     'невидимая из этого приложения! '+;
                     'Попробуйте открыть базу данных в режиме '+;
                     'Exclusive и запустить команду Clean Up';
                     ,48,'Фатальная ошибка!')
         DELETE             && Удаление изменений в буфере
         =TABLEREVERT()     && Отказ от записи на диск
   * В случае возникновения других ошибок
   OTHERWISE
         lnMsgResult=MESSAGEBOX(' Код ошибки '+STR(ERRORCODE,4)+;
         [    ]+MESSAGE()+[     ]+MESSAGE([1])+;
         '    Прекратить работу?',20,' Найдена ошибка ')
         IF lnMsgResult=6      && Кнопка Да
            DO STOP
         ENDIF
ENDCASE
RETURN

PROCEDURE NETDELAY           && Процедура ожидания
PRIVATE DTDELAY,DTSECOND
DTDELAY=SECONDS()+3          && Задержка 3 секунды
DTSECOND=SECONDS()
DO WHILE DTSECOND<DTDELAY
   DTSECOND=SECONDS()
ENDDO
SET ESCAPE OFF
WAIT CLEAR
WAIT '' WINDOW TIMEOUT 0.1
SET ESCAPE ON
RETURN

PROCEDURE CANCELWAIT          && Конец ожидания
ON KEY LABEL F10
WAIT CLEAR
```

```
DO STOP
RETURN TO MASTER

PROCEDURE ADJUSTMENT          && Описание глобальных переменных
PUBLIC VersionFoxPro          && Версия FoxPro
VersionFoxPro=PADR(VERSION(),16)
PUBLIC DISK                   && Папка, из которой запушен комплекс
DISK=SYS(5)+SYS(2003)
* Задержка при пошаговом поиске в секундах
_DblClick=0.5
PUBLIC FAMILY                 && Фамилия работника
FAMILY=[Разработчик комплекса]
PUBLIC SuperVisor             && Признак идентификации
Supervisor = .F.

* Глобальные переменные для доступа к пунктам меню и кнопкам форм
PUBLIC ChangePicture, ChangePassword,SetDeleted,CountRecords,;
      RightAccess,SeekBuilding,EddBuilding,WorkFlats,AccountWork,;
      WordExcel,StreetTown,DistrictTown,MaterialWall,Staff
STORE.T.TO ChangePicture, ChangePassword,SetDeleted,CountRecords,;
     RightAccess,SeekBuilding,EddBuilding,WorkFlats,AccountWork,;
     WordExcel,StreetTown,DistrictTown,MaterialWall,Staff
RETURN

PROCEDURE CALCULATOR              && Калькулятор
ACTIVATE WINDOW CALCULATOR
MOVE WINDOW CALCULATOR CENTER
*RUN /N4 C:\WINDOWS\CALC.EXE
RETURN

PROCEDURE CALENDAR                && Календарь
ACTIVATE WINDOW CALENDAR
MOVE WINDOW CALENDAR CENTER
RETURN

Function CrKod               && Зашифровка пароля
PARAMETERS cPassword
PRIVAT cLetter,cEncryptedPassword
cPassword=RTRIM(cPassword)
cEncryptedPassword=[]
FOR I=1 TO LEN(cPassword)
    cLetter=SUBSTR(cPassword,i,1)
    cEncryptedPassword=cEncryptedPassword+CHR(ASC(cLetter)-I)
NEXT I
RETURN cEncryptedPassword


Function UnKod               && Расшифровка пароля
PARAMETERS cEncryptedPassword
PRIVAT cLetter,cPassword
cEncryptedPassword=RTRIM(cEncryptedPassword)
```

```
cPassword=[]
FOR I=1 TO LEN(cEncryptedPassword)
  cLetter=SUBSTR(cEncryptedPassword,i,1)
  cPassword=cPassword+CHR(ASC(cLetter)+I)
NEXT I
RETURN cPassword

PROCEDURE DisplayMemory                    && Состояние памяти
PRIVATE FileName
* Имя временного файла - Display.txt
FileName='C:\WINNT\TEMP\Display.txt'
* Разместить состояние памяти в файле
DISPLAY MEMORY TO FILE &FileName NOCONSOLE
* Просмотреть при помощи программы Блокнот
RUN /N1 NOTEPAD.EXE &FileName
* Удалить временный файл
DELETE FILE &FileName
RETURN

PROCEDURE Detail        && Дата прописью
PARAMETERS SelectDateTreaty,DayText,MonthText,YearText
DAYS=DAY(SelectDateTreaty)
DO CASE
   CASE DAYS=1
        DayText=[Первое]
   CASE DAYS=2
        DayText=[Второе]
   *- Фрагмент пропущен
   CASE DAYS=31
        DayText=[Тридцать первое]
ENDCASE
MONTHS=MONTH(SelectDateTreaty)
DO CASE
   CASE MONTHS=1
        MonthText=[ января]
   CASE MONTHS=2
        MonthText=[ февраля]
   *- Фрагмент пропущен
   CASE MONTHS=12
        MonthText=[ декабря]
ENDCASE
YEARS=YEAR(SelectDateTreaty)
DO CASE
   CASE YEARS=2000
        YearText=[ двухтысячного года]
   CASE YEARS=2001
        YearText=[ две тысячи первого года]
   CASE YEARS=2002
        YearText=[ две тысячи второго года]
   * Фрагмент пропущен
   CASE YEARS=2010
```

```
         YearText=[ две тысячи десятого года]
   OTHERWISE
         YearText=[]
ENDCASE
RETURN

PROCEDURE WRITE              && Вывод строки лицевого счета
      * Вызывается из процедуры ExcelAccount (см. выше)
      oExcel.Range([A]+ALLTRIM(STR(nRow,3))).Select
      oExcel.ActiveCell.FormulaR1C1=DTOC(pAccount.Data_zap)

      oExcel.Range([B]+ALLTRIM(STR(nRow,3))).Select
      DO CASE
         CASE pAccount.CONTENTS='Сальдо старт'
              cAcContents=[Начальное сальдо по договору ]
         CASE pAccount.CONTENTS='Упл.Аренда  '
              cAcContents=[Уплачено за аренду]
        *- Фрагмент пропущен
         CASE pAccount.CONTENTS='Конец догов.'
              cAcContents=[Окончание срока действия договора]
         OTHERWISE
             cAcContents=pAccount.Contents
      ENDCASE
      oExcel.ActiveCell.FormulaR1C1=cAcContents
      oExcel.Range([C]+ALLTRIM(STR(nRow,3))).Select
      oExcel.ActiveCell.FormulaR1C1=pAccount.nach
      oExcel.Range([D]+ALLTRIM(STR(nRow,3))).Select
      oExcel.ActiveCell.FormulaR1C1=pAccount.ymyb
      * Фрагмент пропущен
      oExcel.Range([N]+ALLTRIM(STR(nRow,3))).Select
      oExcel.ActiveCell.FormulaR1C1=pAccount.ostsh
RETURN
```

8. ЧТО СОДЕРЖИТСЯ НА КОМПАКТ-ДИСКЕ

Компакт диск содержит все папки и файлы, которые были созданы в процессе работы над учебным примером Real Estate (рис. 8.1). Всего 15 папок и 4 файла.

BOOK – папка, содержит формы справочников.

CLASS – папка, содержит пользовательские классы.

DBF – папка с данными (контейнер, таблицы, индексные файлы, поля примечаний и т. д.).

DECKTOP – в этой папке расположены картинки, используемые для оформления главного окна программного комплекса.

FORM – папка, содержащая главные формы приложения.

HELP – папка с файлами контекстуально-зависимой помощи, вызываемой при нажатии клавиши F1.

ICO – иконки для оформления интерфейса вашего приложения.

INSTALL – папка, содержащая файлы дистрибутива для инсталляции готового программного комплекса на клиентский компьютер.

PHOTO – фотографии работников предприятия.

PICTURE – фотографии зданий (в данном примере).

PROGRAM – содержит процедурные файлы.

REPORT – папка, содержащая отчеты Visual FoxPro.

USER – папка, содержащая информацию о пользователях и паролях для работы с программным комплексом

VBA – папка, содержащая файлы с константами VBA для Microsoft Excel и Microsoft Word.

RealEstate.pjt и RealEstate.pjx – файлы проекта Visual FoxPro.

RealEstate.exe – исполняемый файл приложения.

Config.fpw – файл, содержащий настройки Visual FoxPro.

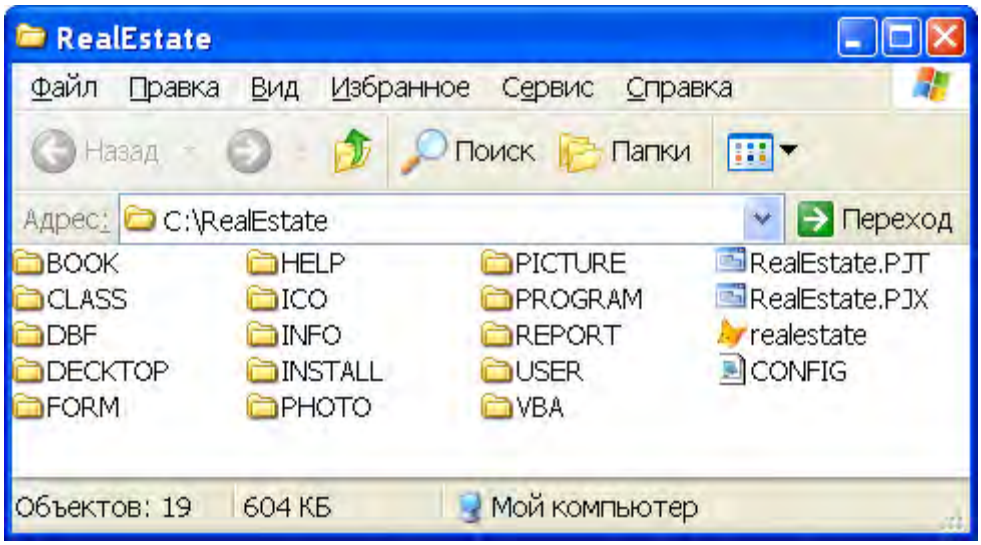


Рис. 8.1. Состав компакт-диска Real Estate

9. ЗАДАНИЯ НА КУРСОВОЙ ПРОЕКТ

Выполните разработку программного комплекса в соответствии с требованиями, подтверждающими квалификацию разработчика прикладного программного обеспечения. Программный комплекс не должен терять работоспособности при некорректных действиях пользователя. В интерфейсе должны использоваться только термины, понятные пользователю. Появление англоязычных сообщений СУБД недопустимо. Язык диалога должен категорически исключить компьютерный сленг.

Вариант 1. Разработать прикладное программное обеспечение деятельности депо по ремонту пассажирских вагонов. Депо выполняет несколько видов ремонта. Деповской ремонт – после пробега вагоном

450 тыс. км или два года эксплуатации (что наступит раньше). ТО-2 – подготовка вагона к зимним или летним условиям эксплуатации. ТО-3 – текущее обслуживание – после пробега 150 тыс. км или один год эксплуатации. Текущий ремонт – круглосуточно, при котором ремонтируются вагоны всех дорог России. Основные причины поступления вагона в текущий ремонт: неисправность колесной пары, неисправность буксового узла и т. д. Каждый вагон имеет уникальный номер. Тип вагона также имеет значение при ремонте: купейный, СВ, плацкартный, почтовый, багажный. Каждый вагон приписан к дирекции по обслуживанию пассажиров (ДОП-1, ДОП-2, ДОП-3 и т. д.). Текущий ремонт выполняют ремонтные бригады в четыре смены. Для выполнения остальных ремонтов привлекается, как правило, одна бригада. За высокое качество ремонта члены бригады получают премию.

Таблица 9.1

Набор данных к варианту 1

№	Поле	Тип	Размер	Описание
1	RegNumber	Числовой	10	Регистрационный номер вагона
2	RegName	Текстовый	60	Приписка вагона к дороге
3	RegChief	Текстовый	20	Приписка вагона к дирекции
4	Type	Текстовый	20	Тип вагона (купейный, СВ, и т. д.)
5	TypeYear	Числовой	4	Год выпуска вагона
6	TypeRepair	Текстовый	39	Тип ремонта
7	Picture	Поле OLE	Авто	Фотография вагона
8	Money	Денежный	15	Стоимость ремонта
9	Bonus	Логический	1	Качество ремонта (отличное/по нормам)
10	BonusPercent	Числовой	2	Премия в процентах (общая)
11	DateStart	Дата/Время	Авто	Начало ремонта
12	DateStop	Дата/Время	Авто	Окончание ремонта
13	Reason	Текстовый	40	Причина поступления в ремонт
14	External	Логический	1	Внешняя/Местная железная дорога
15	BankExternal	Текстовый	60	Банк внешней железной дороги
16	InnExternal	Числовой	10	ИНН внешней железной дороги
17	AddressExternal	Текстовый	80	Юридический адрес внешней ж/дороги
18	FIOchief	Текстовый	40	ФИО бригадира
19	Base	Текстовый	15	Образование бригадира (ВУЗ)
20	FIOworker	Текстовый	40	ФИО работника
21	BaseWorker	Числовой	15	Образование работника (ВУЗ)
22	YearWorker	Числовой	2	Стаж работы
23	SpecialWorker	Текстовый	30	Основная специальность работника
24	BonusWorker	Денежный	15	Премия в рублях работнику
25	Comment	Поле Memo	Авто	Примечания (за что премия)
26	NumberBankKart	Текстовый	60	Номер карты для перечисления З/П

Вариант 2. Разработать прикладное программное обеспечение деятельности ремонтно-эксплуатационного локомотивного депо. Депо выполняет несколько видов ремонта: текущий ремонт (ТР), средний ремонт (СР), техническое обслуживание (ТО) и внеплановый ремонт. При внеплановом ремонте локомотив снимается с рейса и заменяется резервным, поэтому сроки внепланового ремонта должны быть минимальными, а сам ремонт проводится порой в четыре смены. Каждый локомотив имеет уникальный номер и приписан к определенному локомотивному депо. Технология ремонта зависит от типа локомотива (пассажирский или грузовой). Для выполнения первых трех видов ремонта привлекается, как правило, одна бригада. За высокое качество выполненных работ члены бригады получают дополнительное вознаграждение (квартальная премия, месячная премия, 13 и 14 зарплата). За переработку (сверхурочные) также выплачиваются дополнительные суммы.

Таблица 9.2

Набор данных к варианту 2

№	Поле	Тип	Размер	Описание
1	RegNumber	Числовой	10	Регистрационный номер локомотива
2	RegName	Текстовый	60	Приписка локомотива к депо
3	Kind	Текстовый	20	Марка локомотива (ВЛ-80с, ВЛ-80р и т. д.)
4	Type	Текстовый	20	Тип локомотива (грузовой, пассажирский)
5	TypeYear	Числовой	4	Год выпуска локомотива
6	TypeRepair	Текстовый	39	Тип ремонта
7	Picture	Поле OLE	Авто	Фотография локомотива
8	Money	Денежный	15	Стоимость ремонта
9	Bonus	Логический	1	Качество ремонта (отличное/по нормам)
10	BonusPercent	Числовой	2	Премия в процентах (общая)
11	DateStart	Дата/Время	Авто	Начало ремонта
12	DateStop	Дата/Время	Авто	Окончание ремонта
13	Reason	Текстовый	40	Причина поступления в ремонт
14	External	Логический	1	Внешнее/Местное депо
15	BankExternal	Текстовый	60	Банк внешнего депо
16	InnExternal	Числовой	10	ИНН внешнего депо
17	AddressExternal	Текстовый	80	Юридический адрес внешнего депо
18	FIOchief	Текстовый	40	ФИО бригадира
19	Base	Текстовый	15	Образование бригадира (ВУЗ)
20	FIOworker	Текстовый	40	ФИО работника
21	BaseWorker	Числовой	15	Образование работника (ВУЗ)
22	YearWorker	Числовой	2	Стаж работы
23	SpecialWorker	Текстовый	30	Основная специальность работника
24	BonusWorker	Денежный	15	Премия в рублях работнику
25	Comment	Поле Memo	Авто	Примечания (за что премия)
26	NumberBankKart	Текстовый	60	Тип премии

Вариант 3. Разработать прикладное программное обеспечение деятельности судоходной компании «Балтика». Эта крупная компания занимается перевозками грузов между континентами. В ее собственности несколько десятков судов различного класса и грузоподъемности. К услугам этой компании обращаются тысячи клиентов из различных стран мира.

На судне может находиться несколько партий грузов для различных грузополучателей из различных стран и городов. Одна партия груза может состоять из нескольких разновидностей грузов. У одной партии груза может быть только один отправитель и только один получатель.

Судно следует по маршруту. Маршрут разрабатывается главным менеджером компании и проходит через несколько портов. В очередном порту назначения производится лишь частичная погрузка и выгрузка грузов, и судно следует дальше.

Таблица 9.3

Набор данных к варианту 3

№	Поле	Тип	Размер	Описание
1	RegNumber	Числовой	10	Регистрационный номер судна
2	Name	Текстовый	60	Название судна
3	Skipper	Текстовый	60	ФИО капитана судна
4	Type	Текстовый	15	Тип судна (танкер, сухогруз)
5	Capacity	Числовой	10	Грузоподъемность судна
6	Year	Числовой	4	Год постройки судна
7	Picture	Поле OLE	Авто	Фотография судна
8	Dockyard	Текстовый	15	Порт приписки
9	CustomValue	Числовой	10	Таможенный номер партии груза
10	DepartureDate	Дата	Авто	Дата убытия груза
11	ArriveDate	Дата	Авто	Дата прибытия груза
12	Origin	Текстовый	20	Пункт отправления
13	Destination	Текстовый	20	Пункт назначения
14	CustomClearance	Логический	1	Необходимость таможенной декларации
15	Number	Числовой	4	Номер груза в партии
16	Shipment	Текстовый	30	Название груза
17	DeclareValue	Числовой	8	Заявленная величина груза
18	Unit	Текстовый	10	Единица измерения груза
19	InsureValue	Числовой	8	Застрахованная величина груза
20	Sender	Текстовый	30	Отправитель груза
21	INNsender	Числовой	10	ИНН отправителя груза
22	BankSender	Текстовый	60	Банк отправителя груза
23	AddressSender	Текстовый	80	Юридический адрес отправителя груза
24	Consignee	Текстовый	30	Получатель груза
25	INNconsignee	Числовой	10	ИНН получателя груза
26	BankConsignee	Текстовый	60	Банк получателя груза
27	AddressConsign	Текстовый	80	Юридический адрес получателя груза
28	Comment	Поле Memo	Авто	Примечания

Вариант 4. Разработать прикладное программное обеспечение деятельности учреждения юстиции. По существующему законодательству на это учреждение возложена обязанность регистрации прав юридических и физических лиц на недвижимое имущество (здания, квартиры, земельные участки). В этом задании вам необходимо разработать лишь часть программного комплекса, обеспечивающего регистрацию прав граждан на квартиры. Имейте в виду! В здании несколько квартир. В одной квартире – несколько собственников, причем в базе данных должна храниться история перехода квартиры от одних собственников к другим. Кадастровый номер здания однозначно определяет его среди других зданий города. Смело используйте его в качестве первичного ключа таблицы зданий.

Таблица 9.4

Набор данных к варианту 4

№	Поле	Тип	Размер	Описание
1	Kadastr	Текстовый	20	Кадастровый номер здания
2	Address	Текстовый	60	Адрес здания
3	District	Текстовый	15	Район города
4	Land	Числовой	10	Площадь земельного участка
5	Year	Числовой	4	Год постройки здания
6	Material	Текстовый	15	Материал стен здания
7	Base	Текстовый	15	Материал фундамента
8	Comment	Поле Memo	Авто	Примечания
9	Wear	Числовой	2	Износ в процентах
10	Flow	Числовой	2	Число этажей в здании
11	Line	Числовой	5	Расстояние от центра города
12	Square	Числовой	10	Площадь нежилых помещений
13	Picture	Поле OLE	Авто	Фото здания
14	Flats	Числовой	3	Количество квартир в здании
15	Elevator	Логический	1	Наличие лифта
16	Flat	Числовой	4	Номер квартиры
17	Storey	Числовой	2	Номер этажа
18	Rooms	Числовой	1	Количество комнат
19	SquareFlat	Числовой	Авто	Общая площадь квартиры
20	Dwell	Числовой	Авто	Жилая площадь квартиры
21	Branch	Числовой	Авто	Вспомогательная площадь квартиры
22	Balcony	Числовой	Авто	Площадь балкона
23	Height	Числовой	Авто	Высота квартиры
24	Record	Числовой	2	Номер записи о праве собственности
25	Document	Текстовый	60	Документ на право собственности
26	DateDoc	Дата	Авто	Дата документа о собственности
27	FioHost	Текстовый	60	Ф.И.О. собственника
28	Passport	Поле Memo	Авто	Данные его паспорта
29	Part	Числовой	Авто	Принадлежащая ему доля, %
30	Born	Числовой	4	Год рождения собственника

Вариант 5. Разработать прикладное программное обеспечение деятельности малого научно-внедренческого предприятия «Квадро». Это предприятие занимается прокладкой компьютерных сетей и разработкой программных комплексов для организаций нашего города.

Численность работников в «Квадро» – примерно 80 человек. Одновременно находится в разработке до 30 проектов. Один разработчик может участвовать в нескольких проектах одновременно, но зарплата его от этого не зависит. Одна организация может заказать в «Квадро» несколько разработок. Стоимость каждого проекта оговаривается отдельно. При досрочном выполнении работы заказчик перечисляет научно-внедренческому предприятию определенный, заранее оговоренный процент премии.

Таблица 9.5

Набор данных к варианту 5

№	Поле	Тип	Размер	Описание
1	EmployeeID	Числовой	3	Идентификатор работника
2	EmployeeName	Текстовый	60	ФИО работника
3	Address	Текстовый	60	Домашний адрес
4	District	Текстовый	15	Район города
5	Experience	Числовой	2	Опыт работы по специальности
6	Year	Числовой	4	Год рождения
7	Language	Текстовый	15	Базовый язык программирования
8	Base	Текстовый	15	Образование (вуз)
9	Comment	Поле Memo	Авто	Примечания
10	Salary	Денежный	15	Зарплата
11	Bonus	Денежный	15	Премия
12	GrossSalary	Денежный	15	Полная зарплата
13	Exempt	Денежный	15	Льготы
14	Picture	Поле OLE	Авто	Фото работника
15	ProjectID	Числовой	3	Идентификатор проекта
16	ProjectName	Текстовый	40	Название проекта
17	ProjectStart	Дата	Авто	Дата начала проекта
18	ProjectStop	Дата	Авто	Дата окончания проекта
19	Chief	Текстовый	60	Руководитель проекта
20	Customer	Текстовый	60	Заказчик проекта
21	Cost	Числовой	Авто	Стоимость разработки
22	Phone	Текстовый	10	Телефон заказчика
23	Bank	Текстовый	60	Банк заказчика
24	Account	Текстовый	20	Номер счета в банке
25	INN	Текстовый	10	ИНН заказчика
26	AddressCust	Текстовый	60	Адрес заказчика
27	FioWorker	Текстовый	60	Ответственный от заказчика
28	PhoneWorker	Текстовый	10	Телефон ответственного
29	BonusAll	Числовой	Авто	Премия, %, при досрочном выполнении
30	EmployeeStart	Дата	Авто	Начало участия работника в проекте
31	EmployeeStop	Дата	Авто	Конец участия работника в проекте

Вариант 6. Разработать прикладное программное обеспечение деятельности ООО «Киновидеопрокат». Это предприятие фактически контролирует демонстрацию кинофильмов в кинотеатрах города. Отдел маркетинга, изучив ситуацию на рынке кинофильмов, принимает решение о покупке тех или иных кинолент. Отдел закупок претворяет эти решения в жизнь, причем лента может быть куплена как у производителя, так и у посредника.

Отдел аренды киновидеопроката сдает закупленные фильмы кинотеатрам города в аренду. Так как всегда закупается только одна копия фильма, он не может демонстрироваться одновременно в нескольких кинотеатрах. У одного поставщика может быть куплено несколько фильмов. Также несколько лент может быть в аренде у одного кинотеатра одновременно.

Таблица 9.6

Набор данных к варианту 6

№	Поле	Тип	Размер	Описание
1	Provider	Текстовый	40	Поставщик кинофильма
2	INN	Текстовый	10	ИНН поставщика кинофильма
3	Address	Текстовый	60	Юридический адрес поставщика
4	Bank	Текстовый	60	Банк поставщика кинофильма
5	Account	Текстовый	20	Номер счета в банке
6	Sign	Логический	1	Признак посредника
7	Film	Текстовый	20	Название кинофильма
8	Script	Текстовый	60	Автор сценария
9	Comment	Поле Memo	Авто	Краткое содержание фильма
10	Producer	Текстовый	60	Режиссер-постановщик
11	Company	Текстовый	40	Компания-производитель
12	Year	Числовой	4	Год выхода на экран
13	Expense	Денежный	15	Затраты на производство
14	Cost	Денежный	15	Стоимость приобретения
15	Translate	Логический	1	Наличие дублирования
16	Cinema	Текстовый	20	Название кинотеатра
17	INNcinema	Текстовый	10	ИНН кинотеатра
18	AddressCinema	Текстовый	60	Адрес кинотеатра
19	Chief	Текстовый	60	Директор кинотеатра
20	Owner	Текстовый	60	Владелец кинотеатра
21	BankCinema	Текстовый	60	Банк кинотеатра
22	Phone	Текстовый	10	Телефон кинотеатра
23	District	Текстовый	15	Район города
24	AccountCinema	Текстовый	20	Номер счета кинотеатра в банке
25	Capacity	Числовой	4	Число посадочных мест
26	DateStart	Дата	Авто	Дата начала демонстрации фильма
27	DateStop	Дата	Авто	Окончание демонстрации
28	PhoneWorker	Текстовый	10	Телефон ответственного
29	Worker	Текстовый	60	Ответственный от кинотеатра
30	Summa	Денежный	15	Сумма оплаты за аренду ленты
31	Tax	Денежный	15	Пени за несвоевременный возврат

Вариант 7. Разработать прикладное программное обеспечение деятельности предприятия LADA-сервис. Эта крупная компания занимается продажей автомобилей марки ВАЗ в нашем городе. Она имеет несколько филиалов в разных районах. Автомобиль может быть продан как со склада компании, так и на заказ с завода-изготовителя по предоплате. Покупатель может заказать модель, цвет, тюнинг и оговорить срок поставки заказанного автомобиля. Одновременно с новыми авто на площадках компании имеется большой выбор подержанных автомобилей, как отечественных, так и иностранных. Покупателем может быть как физическое лицо, так и организация. В первом случае – расчет наличными, во втором – через банк. Расчет производится в рублях.

Таблица 9.7

Набор данных к варианту 7

№	Поле	Тип	Размер	Описание
1	IDfilial	Числовой	1	Регистрационный номер филиала
2	Filial	Текстовый	20	Название филиала предприятия
3	InnFilial	Текстовый	10	ИНН филиала предприятия
4	Chief	Текстовый	60	Руководитель филиала
5	Capacity	Числовой	3	Число стояночных мест на площадке
6	Address	Текстовый	60	Адрес филиала предприятия
7	Phone	Текстовый	10	Номер телефона филиала
8	Brand	Текстовый	15	Марка автомобиля
9	Model	Текстовый	15	Модель автомобиля
10	BodyID	Текстовый	20	Номер кузова
11	EngineID	Текстовый	20	Номер двигателя
12	BodyModel	Текстовый	20	Модель кузова
13	Picture	Поле OLE	Авто	Фотография автомобиля
14	Volume	Числовой	5	Объем двигателя
15	Power	Числовой	3	Мощность двигателя, л.с.
16	Helm	Логический	1	Руль (правый/левый)
17	Drive	Логический	1	Привод на все колеса
18	DateStart	Дата	Авто	Дата появления в продаже
19	Cost	Денежный	15	Стоимость автомобиля
20	New	Логический	1	Новый/подержанный
21	Year	Числовой	4	Год выпуска автомобиля
22	Distance	Числовой	6	Пробег автомобиля, км
23	Type	Текстовый	15	Тип кузова автомобиля
24	Client	Текстовый	60	Покупатель автомобиля
25	Sign	Логический	1	Признак покупателя (юр/физ. лицо)
26	Bank	Текстовый	60	Банк покупателя
27	Account	Текстовый	20	Номер счета в банке
28	Comment	Поле Memo	Авто	Примечания
29	Customer	Текстовый	60	Заказчик
30	Price	Денежный	15	Стоимость заказанного автомобиля
31	StartDate	Дата	Авто	Дата заказа

Вариант 8. Разработать прикладное программное обеспечение торгово-посреднической фирмы «Столица». Бизнес этого предприятия предельно прост: «покупай дешевле – продавай дороже», или состыкуй продавца и покупателя и получи «комиссионные». Основной упор фирма делает на закупки продуктов питания в других регионах страны и за рубежом – там, где они производятся и стоят дешевле, чем в нашем регионе.

Часть продукции может быть закуплена и у местных продавцов. В этом случае фирма получает прибыль за счет того, что крупные партии товара стоят дешевле, чем мелкие.

Имейте в виду, что товар не может быть продан дешевле, чем он куплен.

Таблица 9.8

Набор данных к варианту 8

№	Поле	Тип	Размер	Описание
1	Seller	Текстовый	60	Фирма – продавец товара
2	InnSeller	Текстовый	10	ИНН продавца
3	Country	Текстовый	15	Страна продавца
4	Chief	Числовой	60	Руководитель фирмы
5	Address	Числовой	60	Юридический адрес фирмы
6	Phone	Текстовый	10	Телефон руководителя
7	Manager	Текстовый	60	Главный менеджер фирмы
8	PhonePlus	Текстовый	10	Телефон отдела продаж
9	Bank	Текстовый	60	Банк продавца
10	Account	Текстовый	20	Номер счета в банке
11	GoodsID	Числовой	10	Штрих-код товара
12	Goods	Текстовый	30	Название товара
13	Picture	Поле OLE	Авто	Фото товара
14	Category	Текстовый	15	Категория товара (кофе, печенье)
15	DateStart	Дата	Авто	Дата изготовления товара
16	Period	Числовой	4	Срок хранения товара, дн.
17	Manufacturer	Текстовый	60	Изготовитель товара
18	Unit	Текстовый	10	Единица измерения
19	CostUnit	Денежный	15	Цена за единицу
20	Count	Числовой	Авто	Количество товара
21	Client	Текстовый	60	Покупатель товара
22	InnClient	Текстовый	10	ИНН покупателя
23	Director	Текстовый	60	Руководитель фирмы-покупателя
24	PhoneDir	Текстовый	10	Телефон директора
25	AddressClient	Текстовый	60	Юридический адрес фирмы
26	BankClient	Текстовый	60	Банк покупателя
27	AccountClient	Текстовый	20	Номер счета в банке
28	Volume	Числовой	Авто	Количество купленного товара
29	CostUnitVol	Денежный	15	Цена за единицу
30	DateVolume	Дата	Авто	Дата покупки товара
31	Comment	Поле Мемо	Авто	Примечания

Вариант 9. Разработать прикладное программное обеспечение деятельности отдела гарантийного ремонта товаров фирмы «Народная торговая компания». Это предприятие – лидер продаж кондиционеров, телевизоров и другой бытовой техники в городе. Хорошо известно, что техника часто выходит из строя, причем уже в период гарантийного срока, а в этом случае продавец товара должен бесплатно отремонтировать его. Ежедневно в отдел гарантийного ремонта обращаются несколько десятков человек, купивших технику в этой компании. Вы, скорее всего, также побывали в отделе гарантийного ремонта, что очень поможет Вам при разработке программного обеспечения.

Таблица 9.9

Набор данных к варианту 9

№	Поле	Тип	Размер	Описание
1	IDfilial	Числовой	1	Регистрационный номер филиала
2	Filial	Текстовый	20	Название филиала предприятия
3	InnFilial	Текстовый	10	ИНН филиала предприятия
4	Chief	Текстовый	60	Руководитель филиала
5	Capacity	Числовой	3	Количество работающих на ремонте
6	Address	Текстовый	60	Адрес филиала предприятия
7	Phone	Текстовый	10	Номер телефона филиала
8	GoodsID	Текстовый	15	Штрих-код товара
9	Goods	Текстовый	40	Название товара или прибора
10	Categoty	Текстовый	20	Категория (утюг, миксер)
11	Country	Текстовый	20	Страна-производитель
12	Company	Текстовый	40	Изготовитель
13	Picture	Поле OLE	Авто	Фотография товара или прибора
14	INNcompany	Текстовый	10	ИНН изготовителя
15	AdddressComp	Текстовый	60	Адрес изготовителя
16	DateStart	Дата	Авто	Дата изготовления товара
17	Period	Числовой	4	Гарантийный период
18	DateBuy	Дата	Авто	Дата покупки
19	Cost	Денежный	15	Стоимость товара
20	Fax	Текстовый	12	Номер факса компании
21	PhoneCompany	Текстовый	12	Телефон компании
22	Email	Текстовый	20	Адрес электронной почты компании
23	Web	Текстовый	20	Адрес WEB-страницы
24	CostRepair	Денежный	15	Стоимость ремонта
25	CustomerID	Числовой	5	Идентификатор покупателя
26	Customer	Текстовый	60	Покупатель
27	AddressCust	Текстовый	60	Адрес покупателя
28	Comment	Поле Memo	Авто	Примечания (что было сделано)
29	Sign	Логический	1	Признак покупателя (юр/физ. лицо)
30	Guarantee	Числовой	5	Оставшийся гарантийный срок
31	StartDate	Дата	Авто	Дата приемки в ремонт
32	StopDate	Дата	Авто	Дата получения

Вариант 10. Разработать прикладное программное обеспечение деятельности отдела учета личного состава батальона железнодорожных войск. Это фактически отдел кадров воинской части. Батальон расквартирован на отдельной территории. В батальоне несколько рот, в каждой роте несколько взводов, каждый взвод состоит из трех отделений.

В мирное время батальон занимается изучением техники и поддержанием ее в рабочем состоянии. Часть технических ресурсов «законсервирована». Поддержание такой техники в отличном состоянии также входит в обязанности личного состава батальона. В настоящее время существует три вида службы: срочная, сверхсрочная и по контракту. Каждый офицер части имеет удостоверение личности, которое заменяет паспорт, а военнослужащий срочной службы – военный билет.

Таблица 9.10

Набор данных к варианту 10

№	Поле	Тип	Размер	Описание
1	Number	Текстовый	15	Номер воинской части
2	Battalion	Текстовый	30	Название батальона
3	Commander	Текстовый	50	ФИО командира батальона
4	Rank	Текстовый	20	Воинское звание командира
5	ViceCommander	Текстовый	50	ФИО зам. командира батальона
6	Rank2	Текстовый	20	Воинское звание зам. командира
7	Photo	Поле OLE	Авто	Фото командира батальона
8	Commander2	Текстовый	50	ФИО командира роты
9	Rank3	Текстовый	20	Звание командира роты
10	Passport	Текстовый	20	Удостоверение личности
11	CompanyID	Числовой	1	Номер роты
12	CompanyName	Текстовый	20	Название роты
13	Photo2	Поле OLE	Авто	Фото командира роты
14	PlatoonID	Числовой	1	Номер взвода
15	Commander3	Текстовый	50	ФИО командира взвода
16	Rank4	Текстовый	20	Звание командира взвода
17	Photo3	Поле OLE	Авто	Фото командира взвода
18	PlatoonName	Текстовый	20	Название взвода
19	DepartmentID	Числовой	1	Номер отделения
20	Soldier	Текстовый	50	ФИО военнослужащего
21	Post	Текстовый	20	Должность (командир/солдат)
22	Rank5	Текстовый	20	Звание военнослужащего
23	Start	Дата	Авто	Начало службы
24	Stop	Дата	Авто	Конец службы
25	Kind	Логический	1	Вид службы (срочная или контрактная)
26	Address	Текстовый	60	Адрес проживания
27	Phone	Текстовый	15	Домашний телефон
28	Comment	Поле Memo	Авто	Примечание
29	Birth	Числовой	4	Год рождения

Вариант 11. Разработать прикладное программное обеспечение деятельности отдела учета домовладений «Бюро технической инвентаризации». В состав домовладения входят земельный участок и несколько строений. Их называют литерами: жилой дом, летняя кухня, гараж, колодец, забор и т. д.

Для жилого дома составляется экспликация, в которой указываются данные по каждому помещению. Экспликация может быть составлена и для других крупных строений. В ее состав входит: номер квартиры, номер помещения на плане, этаж, назначение помещения, площадь, высота и т. п. Для вспомогательных литер (забор, тротуар, колодец) экспликация не заполняется.

Таблица 9.11

Набор данных к варианту 11

№	Поле	Тип	Размер	Описание
1	Number	Числовой	5	Уникальный номер домовладения
2	Block	Текстовый	20	Номер квартала
3	Address	Текстовый	60	Адрес домовладения
4	District	Текстовый	15	Район города
5	Inventory	Дата	Авто	Дата инвентаризации домовладения
6	Land	Числовой	Авто	Площадь земельного участка
7	Actual	Числовой	Авто	Фактическая площадь участка
8	BuildUp	Числовой	Авто	Площадь застройки
9	Yard	Числовой	Авто	Площадь двора
10	Green	Числовой	Авто	Площадь озеленения
11	Garden	Числовой	Авто	Площадь огорода
12	Bad	Числовой	Авто	Неудобья
13	Picture	Поле OLE	Авто	Фотография домовладения
14	Light	Логический	1	Освещение
15	WaterPipe	Логический	1	Водопровод
16	Heating	Логический	1	Отопление
17	Comment	Поле Memo	Авто	Примечания
18	Letter	Числовой	2	Номер сооружения (литеры)
19	Contents	Текстовый	20	Назначение сооружения
20	Type	Числовой	1	Тип литеры (осн./вспомогательная)
21	MySelf	Логический	1	Возведено самовольно
22	Year	Числовой	4	Год постройки
23	SquareAll	Числовой	4	Общая площадь литеры
24	Inhabited	Числовой	4	Жилая площадь
25	Wear	Числовой	2	Износ в процентах
26	Wall	Текстовый	15	Материал стен
27	Cost	Денежный	15	Инвентаризационная стоимость литеры
28	Storeys	Числовой	Авто	Этажность
29	NumberSign	Числовой	2	Номер помещения в экспликации
30	Prescribe	Текстовый	20	Назначение помещения
31	SquareRoom	Числовой	Авто	Площадь помещения
32	HighRoom	Числовой	Авто	Высота помещения
33	Storey	Числовой	Авто	Этаж, на кот. расположено помещение

Вариант 12. Разработать прикладное программное обеспечение деятельности отдела кадров университета. В отделе кадров университета находятся данные всех сотрудников: от преподавателя до ректора, и их трудовой деятельности. Наряду с такими данными, как специальность сотрудника и занимаемая должность, обязательно учитываются сведения об ученой степени сотрудника (кандидат наук, доктор) и ученом звании (доцент, профессор). Также в отделе кадров хранится информация о трудовой деятельности сотрудника: о предыдущих местах работы, сроке работы и предприятии. Отдел кадров занимается подготовкой трудовых договоров с преподавателями после избрания их по конкурсу на очередной срок. Также в его ведении находятся сведения о наложении взысканий на сотрудников и их поощрениях. Взыскания в трудовую книжку не заносятся, а хранятся в электронном виде.

Таблица 9.12

Набор данных к варианту 12

№	Поле	Тип	Размер	Описание
1	PersonID	Числовой	5	Регистрационный номер сотрудника
2	Name	Текстовый	40	ФИО сотрудника
3	Department	Текстовый	40	Название кафедры, на которой работает
4	Institute	Текстовый	40	Название института (департамента)
5	Birth	Дата	Авто	Дата рождения сотрудника
6	Place	Текстовый	20	Место рождения
7	Address	Текстовый	60	Домашний адрес сотрудника
8	Phone	Текстовый	15	Домашний телефон сотрудника
9	Education	Текстовый	40	Оконченный ВУЗ
10	Year	Числовой	4	Год окончания ВУЗа
11	Speciality	Текстовый	30	Специальность сотрудника
12	Picture	Поле OLE	Авто	Фотография сотрудника
13	DegreeYes	Логический	1	Ученая степень (есть/нет)
14	Degree	Числовой	1	Ученая степень сотрудника
15	Rank	Числовой	1	Ученое звание сотрудника
16	Post	Текстовый	20	Занимаемая должность
17	Comment	Поле Memo	Авто	Примечания
18	Passport	Текстовый	20	Номер паспорта
19	PassportDate	Дата	Авто	Дата выдачи паспорта
20	Region	Текстовый	40	Кем выдан паспорт
21	WorkBegin	Дата	Авто	Дата начала трудовой деятельности
22	WorkEnd	Дата	Авто	Дата окончания трудовой деятельности
23	Work	Текстовый	20	В качестве кого работал
24	WorkPlace	Текстовый	20	Название предприятия
25	WorkAddress	Текстовый	60	Адрес предприятия
26	WorkPhone	Текстовый	15	Телефон предприятия
27	Reason	Текстовый	30	Причина увольнения
28	Penalty	Поле Memo	Авто	Сведения о взысканиях
29	Rewards	Поле Memo	Авто	Сведения о награждениях

Вариант 13. Разработать прикладное программное обеспечение деятельности биржи труда. На биржу труда обращаются люди, не сумевшие самостоятельно устроиться на работу, но все ещё желающие найти работу по специальности. Организации предоставляют бирже список свободных вакансий. Каждый обратившийся ставится на учет. В день обращения ему предлагается список вакансий. Если свободных вакансий нет или они не устраивают ищущего работу, то ему будет предложено подождать пока подходящее свободное место работы не появится. Зарегистрированный на бирже получает пособие по безработице до тех пор, пока не будет трудоустроен. После этого его данные переносятся в архив, и выплата ему пособия прекращается.

Таблица 9.13

Набор данных к варианту 13

№	Поле	Тип	Размер	Описание
1	JoblessID	Числовой	5	Регистрационный номер безработного
2	LastName	Текстовый	20	Фамилия безработного
3	FirstName	Текстовый	20	Имя безработного
4	Patronymic	Текстовый	20	Отчество безработного
5	Age	Числовой	2	Возраст безработного
6	Passport	Текстовый	20	Номер паспорта
7	PassportDate	Дата	Авто	Дата выдачи паспорта
8	Region	Текстовый	40	Кем выдан паспорт
9	Address	Текстовый	60	Адрес безработного
10	Phone	Текстовый	15	Телефон безработного
11	Picture	Поле OLE	Авто	Фотография безработного
12	StudyPlace	Текстовый	60	Название оконченного ВУЗа
13	StudyAddress	Текстовый	60	Адрес оконченного учебного заведения
14	StudyType	Текстовый	15	Тип образования (высшее и т. д.)
15	Registrar	Текстовый	15	Фамилия регистрирующего
16	RegDate	Дата	Авто	Дата постановки на учет
17	Payment	Денежный	15	Величина пособия
18	Experience	Логический	1	Опыт работы по специальности (да/нет)
19	Comment	Поле Memo	Авто	Примечания
20	ArchivesDate	Дата	Авто	Дата перевода в архив
21	Archivist	Текстовый	15	Фамилия удалившего в архив
22	JobID	Числовой	1	Номер вакансии
23	JobType	Текстовый	20	Тип вакансии (техническая, экономич.)
24	JobName	Текстовый	20	Название вакансии
25	JobGiver	Текстовый	20	Работодатель
26	Place	Текстовый	60	Адрес работодателя
27	Mobile	Текстовый	15	Телефон работодателя
28	District	Текстовый	15	Район, в котором предлагается работа
29	Money	Денежный	15	Примерный размер зарплаты
30	More	Поле Memo	Авто	Особые требования к работнику

Вариант 14. Разработать прикладное программное обеспечение деятельности отдела учета квартир «Бюро технической инвентаризации».

В нашем городе имеется 6000 зданий, в которых расположено 199000 квартир. Помещений в этих квартирах – 1 500 000 шт. Кадастровый номер здания является уникальным. Используйте его в качестве простого первичного ключа таблицы зданий. Можете работать и с составным первичным ключом (адресом здания), но в данном случае – это не лучший вариант.

Таблица 9.14

Набор данных к варианту 14

№	Поле	Тип	Размер	Описание
1	Kadastr	Текстовый	20	Кадастровый номер здания
2	Address	Текстовый	60	Адрес здания
3	District	Текстовый	15	Район города
4	Land	Числовой	10	Площадь земельного участка
5	Year	Числовой	4	Год постройки здания
6	Material	Текстовый	15	Материал стен здания
7	Base	Текстовый	15	Материал фундамента
8	Comment	Поле Memo	Авто	Примечания
9	Wear	Числовой	2	Износ в процентах
10	Flow	Числовой	2	Количество этажей в здании
11	Line	Числовой	5	Расстояние от центра города
12	Square	Числовой	10	Площадь квартир
13	Picture	Поле OLE	Авто	Фото здания
14	Flats	Числовой		Количество квартир в здании
15	Elevator	Логический	1	Наличие лифта
16	Flat	Числовой	4	Номер квартиры
17	Storey	Числовой	2	Номер этажа
18	Rooms	Числовой	1	Количество комнат
19	Level	Логический	1	Квартира в двух уровнях
20	SquareFlat	Числовой	Авто	Общая площадь квартиры
21	Dwell	Числовой	Авто	Жилая площадь квартиры
22	Branch	Числовой	Авто	Вспомогательная площадь квартиры
23	Balcony	Числовой	Авто	Площадь балкона
23	Height	Числовой	Авто	Высота квартиры
25	Record	Числовой	2	Номер помещения в квартире
26	SquareRoom	Числовой	Авто	Площадь помещения
27	Size	Текстовый	40	Размеры помещения в плане
28	Name	Текстовый	30	Назначение (кухня, ниша …)
29	Decoration	Текстовый	60	Отделка (паркет, обои …)
30	HeightRoom	Числовой	Авто	Высота помещения
31	Socket	Числовой	2	Число розеток в помещении
32	Sections	Числовой	2	Число элементов в батарее отопления

Вариант 15. Разработать прикладное программное обеспечение деятельности аптечного склада. Аптечный склад занимается оптовой продажей лекарств больницам и аптекам города. В его ассортименте – тысячи наименований лекарств, а также различных аптечных принадлежностей (градусники, шприцы, бинты и т. д.) Возможна продажа лишь тех лекарств, которые одобрены Минздравом РФ, т. е. имеют регистрационный номер Минздрава РФ. Поступающие лекарства сопровождаются документами – приходными накладными ведомостями. Покупатель получает счет-фактуру на выбранный товар, оплачивает сумму, указанную в ней, и после оплаты получает выходную накладную ведомость, по которой получает выбранный товар.

Таблица 9.15

Набор данных к варианту 15

№	Поле	Тип	Размер	Описание
1	GoodsID	Числовой	10	Регистрационный номер товара в базе
2	Name	Текстовый	40	Название товара
3	International	Текстовый	40	Международное название лекарства
4	Begin	Дата	Авто	Дата производства
5	End	Дата	Авто	Годен до
6	Yes	Логический	1	Одобрено Минздравом РФ (да/нет)
7	RF	Текстовый	20	Регистрационный номер Минздрава РФ
8	Producer	Текстовый	60	Данные о производителе
9	Instructions	Поле Memo	Авто	Инструкция к лекарству
10	Batch	Текстовый	20	Вид упаковки
11	Seller	Текстовый	20	Название поставщика
12	Address	Текстовый	60	Адрес поставщика
13	Phone	Текстовый	15	Телефон поставщика
14	INN	Текстовый	10	ИНН поставщика
15	Sign	Логический	1	Признак посредника
16	Date	Дата	Авто	Дата поступления на склад
17	Price	Денежный	10	Цена товара
18	GoodsInvoice	Числовой	4	Номер приходной накладной ведомости
19	ClientID	Числовой	5	Номер покупателя
20	Company	Текстовый	25	Название покупателя
21	Address	Текстовый	60	Адрес покупателя
22	Phone	Текстовый	15	Телефон покупателя
23	CountNumber	Числовой	4	Номер счет-фактуры
24	DateStart	Дата	Авто	Дата выписки счет-фактуры
25	Sum	Денежный	15	Сумма к уплате
26	Cash	Логический	1	Оплата наличными (да/нет)
27	Worker	Текстовый	60	Выдавший счет-фактуру
28	Invoice	Числовой	4	Номер выходной накладной
29	INNClient	Текстовый	10	ИНН покупателя
30	Seller	Текстовый	15	Фамилия продавца

Вариант 16. Разработать прикладное программное обеспечение деятельности отдела учета нежилых помещений «Бюро технической инвентаризации».

В 2000 г. в нашем городе была проведена сплошная инвентаризация, в ходе которой было выявлено 16000 нежилых помещений. Это магазины (встроенные, пристроенные и отдельно стоящие), офисы, учреждения, мастерские и т. д. Помещение может состоять из отдельных частей (кабинет, проходная, коридор). Составных частей помещений выявлено 265 000 шт.

В одном здании может быть несколько помещений, а помещение может состоять из нескольких частей. Любое здание имеет уникальный кадастровый номер, однозначно определяющий его положение в городе.

Таблица 9.16

Набор данных к варианту 16

№	Поле	Тип	Размер	Описание
1	Kadastr	Текстовый	20	Кадастровый номер здания
2	Address	Текстовый	60	Адрес здания
3	District	Текстовый	15	Район города
4	Land	Числовой	10	Площадь земельного участка
5	Year	Числовой	4	Год постройки здания
6	Material	Текстовый	15	Материал стен здания
7	Base	Текстовый	15	Материал фундамента
8	Comment	Поле Memo	Авто	Примечания
9	Wear	Числовой	2	Износ в процентах
10	Flow	Числовой	2	Количество этажей в здании
11	Line	Числовой	5	Расстояние от центра города
12	Square	Числовой	10	Площадь нежилых помещений
13	Picture	Поле OLE	Авто	Фото здания
14	Hall	Числовой	3	Количество помещений в здании
15	Elevator	Логический	1	Наличие лифта
16	HallNum	Числовой	4	Номер помещения
17	Storey	Числовой	2	Номер этажа
18	Rooms	Числовой	1	Количество составных частей
19	Level	Логический	1	Помещение в двух уровнях
20	SquareHall	Числовой	Авто	Общая площадь помещения
21	Branch	Числовой	Авто	Вспомогательная площадь помещения
22	Balcony	Числовой	Авто	Площадь балкона
23	Height	Числовой	Авто	Высота помещения
24	Record	Числовой	2	Номер составной части помещения
25	SquarePart	Числовой	Авто	Площадь составной части
26	Size	Текстовый	40	Размеры сост. части в плане
27	NamePart	Текстовый	30	Назначение (кабинет, ниша ...)
28	Decoration	Текстовый	60	Отделка (паркет, обои …)
29	HeightPart	Числовой	Авто	Высота составной части
30	Socket	Числовой	2	Число розеток в помещении
31	Sections	Числовой	2	Число элементов в батарее отопления

Вариант 17. Разработать прикладное программное обеспечение деятельности отдела учета налогообложения физических лиц городской налоговой инспекции. По существующему законодательству некоторые категории граждан должны представить в налоговую инспекцию декларацию о полученных доходах. Налоговый инспектор должен проверить ее, занести в базу данных и выписать платежное извещение на уплату подоходного налога с доходов физического лица. Лица, заполнившие декларацию, должны доплатить в бюджет некоторую сумму. С 2002 г. шкала налогообложения – линейная (13 % со всей заработанной суммы за год), но лицам, затратившим средства на обучение, покупку лекарств и т. д., из бюджета должна быть возвращена некоторая сумма, рассчитываемая по специальной методике.

Таблица 9.17

Набор данных к варианту 17

№	Поле	Тип	Размер	Описание
1	INN	Текстовый	13	Идентификационный номер
2	LastName	Текстовый	20	Фамилия налогоплательщика
3	FirstName	Текстовый	20	Имя налогоплательщика
4	Patronymic	Текстовый	20	Отчество налогоплательщика
5	Document	Текстовый	80	Документ, удостоверяющий личность
6	Serial	Текстовый	10	Серия документа
7	Number	Текстовый	20	Номер документа
8	Date	Дата	Авто	Дата выдачи
9	Region	Текстовый	30	Кем выдан документ
10	Born	Дата	Авто	Дата рождения
11	Picture	Поле OLE	Авто	Фотография налогоплательщика
12	DateTax	Дата	Авто	Дата заполнения декларации
13	NumberTax	Текстовый	12	Номер декларации
14	Address	Текстовый	80	Адрес налогоплательщика
15	District	Текстовый	20	Район города, где проживает
16	DistrictTax	Текстовый	20	Инспекция, где стоит на учете
17	TaxNumber	Текстовый	4	Номер налоговой инспекции
18	Enterprise	Текстовый	40	Организация, выплатившая сумму
19	InnEnterprise	Текстовый	10	ИНН организации
20	AddressWorks	Текстовый	30	Адрес организации
21	Chief	Текстовый	60	ФИО главного бухгалтера
22	Phone	Текстовый	10	Телефон для связи
23	SumAll	Денежный	15	Полученная в организации сумма
24	SumTax	Денежный	15	Величина подоходного налога
25	SumPension	Денежный	15	Отчисления в пенсионный фонд
26	ExemptType	Текстовый	60	Название льготы
27	Exempt	Денежный	15	Сумма льготы
28	Comment	Поле Memo	Авто	Примечания

Вариант 18. Разработать прикладное программное обеспечение деятельности телеателье «Спектр».

Эта организация занимается послегарантийным ремонтом теле-, радиоаппаратуры отечественного и импортного производства. Клиенты этого телеателье – жители и организации нашего города и близлежащих сел. Расчет с физическими лицами ведется наличными, а с организациями – через банк. Выдача отремонтированной техники производится после полной оплаты выполненного ремонта.

Отремонтированное изделие получает гарантию. Если в течение гарантийного срока произойдет поломка изделия, то повторный ремонт выполняется за счет телеателье. Если брак допустил мастер, то часть суммы удерживается из его зарплаты. Клиент, обратившийся к услугам ателье несколько раз с ремонтом разной аппаратуры, получает дисконтную карту, дающую право на скидку при ремонте очередного изделия.

Таблица 9.18

Набор данных к варианту 18

№	Поле	Тип	Размер	Описание
1	CustomerID	Числовой	4	Идентификатор заказчика
2	CustomerType	Логический	1	Тип заказчика (физ./юр. лицо)
3	CustomerFio	Текстовый	60	ФИО заказчика (для физ. лица)
4	CustomerName	Текстовый	60	Название заказчика (для юр. лица)
5	CustomerInn	Текстовый	13	ИНН заказчика (для юр. лица)
6	Chief	Текстовый	40	Руководитель (для юр. лица)
7	Phone	Текстовый	10	Телефон заказчика
8	Address	Текстовый	60	Адрес заказчика
9	Bank	Текстовый	60	Банк заказчика (для юр. лица)
10	District	Текстовый	15	Район заказчика
11	Discont	Текстовый	5	Номер дисконтной карты
12	MasterID	Числовой	2	Идентификатор мастера
13	MasterFio	Текстовый	60	ФИО мастера
14	Experience	Числовой	2	Опыт работы по специальности
15	Defect	Числовой	2	Число некачественных ремонтов
16	RepairAll	Числовой	4	Число отремонтированных изделий
17	TypeID	Числовой	5	Идентификатор заказа
18	Type	Текстовый	15	Тип изделия (телевизор, радио и т. д)
19	Country	Текстовый	15	Страна-производитель
20	Company	Текстовый	40	Фирма-изготовитель
21	Picture	Поле OLE	Авто	Фотография изделия
22	Age	Числовой	2	Возраст изделия в годах
23	DateStart	Дата	Авто	Дата приема в ремонт
24	DateStop	Дата	Авто	Дата выдачи из ремонта
25	Summa	Денежный	15	Стоимость ремонта
26	Period	Числовой	2	Срок гарантии
27	Guarantee	Логический	1	Гарантийный ремонт (да/нет)
28	Comment	Поле Memo	Авто	Примечания

Вариант 19. Разработать прикладное программное обеспечение деятельности отдела заселения муниципальных общежитий администрации города.

В ведении администрации города находится несколько десятков общежитий. Раньше они принадлежали предприятиям города, а теперь, после банкротства предприятий, все эти общежития переданы муниципальным властям. В последние годы бесплатные квартиры гражданам города практически не предоставляются, а количество малоимущих жителей, нуждающихся в жилье, растет. Хоть как-то улучшить жилищные условия этой категории граждан позволяет наличие муниципальных общежитий. Получить четкую картину их заселения позволит данное программное обеспечение. База данных отдела содержит информацию об общежитиях, комнатах общежитий и проживающих.

Таблица 9.19

Набор данных к варианту 19

№	Поле	Тип	Размер	Описание
1	Hostel	Числовой	5	Номер общежития
2	Address	Текстовый	60	Адрес общежития
3	District	Текстовый	15	Район города, в котором расположено
4	Picture	Поле OLE	Авто	Фотография общежития
5	Owner	Текстовый	20	Балансодержатель
6	Rooms	Числовой	4	Комнат в общежитии
7	Beds	Числовой	5	Количество койко-мест в общежитии
8	RoomID	Числовой	4	Номер комнаты
9	Square	Числовой	10	Площадь комнаты
10	Comment	Поле Memo	Авто	Примечания
11	RoomBeds	Числовой	2	Количество койко-мест в комнате
12	Type	Числовой	1	Тип комнаты (одноместная и т.д.)
13	Storey	Числовой	2	Номер этажа
14	Lodger	Числовой	5	Регистрационный номер жильца
15	Name	Текстовый	40	ФИО жильца
16	Passport	Текстовый	20	Номер паспорта
17	PassportDate	Дата	Авто	Дата выдачи паспорта
18	Region	Текстовый	40	Кем выдан паспорт
19	Work	Текстовый	20	Место работы или учебы
20	Children	Логический	1	С детьми (да/нет)
21	DocumentID	Числовой	5	Номер документа на заселение
22	Document	Текстовый	20	Название документа на заселение
23	Begin	Дата	Авто	Начало действия документа
24	Giver	Текстовый	20	Кем выдан документ на заселение
25	DocComment	Поле Memo	Авто	Комментарий
26	Payment	Денежный	Авто	Плата за проживание в месяц
27	Settlement	Дата	Авто	Дата заселения в общежитие
28	End	Дата	Авто	Дата выселения из общежития
29	Reason	Поле Memo	Авто	Причина выселения

Вариант 20. Разработать прикладное программное обеспечение деятельности Государственной автомобильной инспекции по безопасности дорожного движения города.

База данных ГИБДД содержит сведения обо всех транспортных средствах города и их владельцах. В нее заносятся сведения о технических осмотрах транспортных средств и об угонах. Описание угнанного автомобиля не удаляется из базы данных. Истории переходов транспортных средств от одних владельцев к другим не накапливаются. Сведения об автомобилях, снятых с учета, навсегда удаляются из базы данных.

Таблица 9.20

Набор данных к варианту 20

№	Поле	Тип	Размер	Описание
1	OwnerID	Числовой	6	Идентификатор владельца
2	OwnerType	Логический	1	Тип владельца (физ./юр. лицо)
3	OwnerFio	Текстовый	60	ФИО владельца (для физ. лица)
4	OwnerName	Текстовый	60	Название организации
5	OwnerInn	Текстовый	10	ИНН организации
6	Chief	Текстовый	60	Руководитель организации
7	Phone	Текстовый	10	Телефон
8	Address	Текстовый	60	Адрес владельца автомобиля
9	District	Текстовый	15	Район города
10	Number	Текстовый	10	Государственный знак автомобиля
11	Brand	Текстовый	15	Марка автомобиля
12	Model	Текстовый	15	Модель автомобиля
13	BodyID	Текстовый	20	Номер кузова
14	EngineID	Текстовый	20	Номер двигателя
15	BodyModel	Текстовый	20	Модель кузова
16	Color	Текстовый	20	Цвет автомобиля
17	Volume	Числовой	5	Объем двигателя
18	Comment	Поле Memo	Авто	Примечания
19	Power	Числовой	3	Мощность двигателя в л.с.
20	Helm	Логический	1	Руль (правый/левый)
21	Drive	Логический	1	Привод на все колеса
22	Year	Числовой	4	Год выпуска автомобиля
23	TypeBody	Текстовый	15	Тип кузова автомобиля (седан, купе)
24	DrivingAway	Логический	1	Находится в угоне
25	DateAway	Дата	Авто	Дата угона
26	DateReturn	Дата	Авто	Дата возврата владельцу
27	DateSee	Дата	Авто	Дата технического осмотра
28	Inspector	Текстовый	60	ФИО инспектора, проводившего осмотр
29	YearTax	Денежный	15	Годовой налог на автомобиль
30	YearNumber	Денежный	15	Оплата за знак технического осмотра
31	Work	Денежный	15	Оплата за технический осмотр
32	Distance	Числовой	5	Пробег на дату осмотра
33	Okey	Логический	1	Технический осмотр пройден
34	Reason	Поле Memo	Авто	Причины, по которым осмотр не пройден

Вариант 21. Разработать прикладное программное обеспечение для ведения реестра имущества университетского городка.

В состав имущества входит несколько зданий. В зданиях располагаются аудитории, кафедры, лаборатории, вычислительные центры, деканаты и т. д. Любое помещение университета относится к какому-либо подразделению. Все движимое имущество, находящееся в помещении, состоит на балансе материально ответственного лица.

Каждая аудитория закреплена за определенной кафедрой университета, так же в ведении кафедр находятся и лаборатории.

По истечении определенного времени имущество, находящееся в помещениях, списывается. Архив списанного имущества не ведется.

Таблица 9.21

Набор данных к варианту 21

№	Поле	Тип	Размер	Описание
1	Kadastr	Числовой	2	Регистрационный номер здания
2	BuildingName	Текстовый	20	Название здания (корпуса) университета
3	Land	Числовой	Авто	Площадь земельного участка
4	Address	Текстовый	60	Адрес здания
5	Year	Числовой	4	Год постройки
6	Material	Текстовый	15	Материал стен здания
7	Wear	Числовой	2	Износ в процентах
8	Flow	Числовой	2	Число этажей в здании
9	Picture	Поле OLE	Авто	Фотография здания
10	Comment	Поле Memo	Авто	Дополнительные сведения по зданию
11	HallID	Текстовый	5	Номер аудитории
12	Square	Числовой	Авто	Площадь аудитории
13	Windows	Числовой	1	Количество окон
14	Heating	Числовой	3	Число элементов в батареях отопления
15	Target	Текстовый	15	Назначение (лекционная, кафедра ...)
16	Department	Текстовый	15	Принадлежность к кафедре (подразд.)
17	Chief	Текстовый	30	Материально ответст. за аудиторию
18	DepartmentID	Числовой	3	Идентификатор кафедры
19	DepartmentName	Текстовый	15	Название кафедры
20	Boss	Текстовый	40	Заведующий кафедрой
21	Phone	Текстовый	10	Телефон кафедры
22	OfficeDean	Текстовый	30	Принадлежность кафедры к деканату
23	ChiefID	Числовой	3	Идентификатор материально отв-го
24	AddressChief	Текстовый	60	Дом. адрес материально ответственного
25	Experience	Числовой	4	Год его начала работы в университете
26	UnitID	Числовой	3	Идентификатор единицы имущества
27	UnitName	Текстовый	30	Название единицы имущества
28	DateStart	Дата	Авто	Дата постановки на учет
29	Cost	Денежный	15	Стоимость единицы имущества
30	CostYear	Числовой	4	Год переоценки
31	CostAfter	Денежный	15	Стоимость после переоценки
32	Period	Числовой	4	Срок службы единицы имущества

Вариант 22. Разработать прикладное программное обеспечение деятельности туристической компании «Вояж». Эта компания формирует туристические группы для заграничных поездок и обеспечивает им полную поддержку на маршруте. Количество туристов в группе заранее известно и ограничено.

Маршрут группы может пролегать через несколько городов страны назначения. Экскурсии в несколько стран одновременно не проводятся.

При обращении в «Вояж» группы из нескольких человек компания предоставляет скидку, которая зависит от количества туристов в группе. Вместе с группой следует представитель компании, который несет полную ответственность за качество услуг, предоставляемых компанией.

При возникновении каких-либо неудобств на маршруте, возникших по вине компании, турист получает назад заранее оговоренную в контракте сумму.

Таблица 9.22

Набор данных к варианту 22

№	Поле	Тип	Размер	Описание
1	ClientID	Числовой	5	Идентификатор клиента
2	LastName	Текстовый	20	Фамилия клиента
3	FirstName	Текстовый	20	Имя клиента
4	Patronymic	Текстовый	20	Отчество клиента
5	Document	Текстовый	80	Документ, удостоверяющий личность
6	Serial	Текстовый	10	Серия документа
7	Number	Текстовый	20	Номер документа
8	Date	Дата	Авто	Дата выдачи
9	Region	Текстовый	30	Кем выдан документ
10	Born	Дата	Авто	Дата рождения
11	Picture	Поле OLE	Авто	Фотография клиента
12	Pasport	Логический	1	Наличие заграничного паспорта
13	RouteID	Числовой	3	Идентификатор маршрута
14	RouteName	Текстовый	30	Название маршрута
15	Country	Текстовый	20	Название страны
16	Period	Числовой	2	Срок пребывания
17	Worker	Текстовый	20	Представитель на маршруте
18	Cost	Денежный	15	Стоимость путевки
19	Exempt	Денежный	15	Скидка
20	Return	Денежный	15	Неустойка
21	DateStart	Дата	Авто	Дата вылета
22	Town	Текстовый	15	Пункт маршрута
23	Count	Числовой	2	Срок пребывания в пункте маршрута
24	Hotel	Текстовый	15	Название гостиницы
25	StartDate	Дата	Авто	Дата прибытия в пункт маршрута
26	StopDate	Дата	Авто	Дата убытия
27	Type	Числовой	1	Класс гостиницы (***,****)
28	Comment	Поле Memo	Авто	Экскурсионная программа

Вариант 23. Разработать прикладное программное обеспечение деятельности регистратуры ведомственной поликлиники «Эскулап».

Работники регистратуры организуют запись пациентов на прием к врачам поликлиники. Так как поликлиника ведомственная, медицинское обслуживание работников предприятия – бесплатное (за счет средств предприятия).

«Посторонние» пациенты также могут воспользоваться услугами поликлиники, полностью оплатив затраты на лечение. Определение стоимости лечения и выдача платежных документов для таких больных входит в круг обязанностей работников регистратуры. Врач ведет прием всегда в одном кабинете. Приемные дни занесены в расписание работы поликлиники. На каждого пациента в регистратуре заводится карточка. В начале приема карточки больных, записавшихся на прием, доставляются работником регистратуры в кабинет врача.

Таблица 9.23

Набор данных к варианту 23

№	Поле	Тип	Размер	Описание
1	DoctorID	Числовой	2	Идентификационный номер врача
2	LastName	Текстовый	20	Фамилия врача
3	FirstName	Текстовый	20	Имя врача
4	Patronymic	Текстовый	20	Отчество врача
5	Room	Числовой	3	Номер кабинета
6	University	Текстовый	40	Образование (университет)
7	Type	Текстовый	20	Специализация (терапевт, лор...)
8	Experience	Числовой	2	Стаж работы
9	Phone	Текстовый	10	Номер рабочего телефона
10	Born	Числовой	4	Год рождения
11	Picture	Поле OLE	Авто	Фотография врача
12	Fio	Текстовый	60	ФИО пациента
13	Number	Текстовый	10	Номер карточки пациента
14	Address	Текстовый	80	Адрес пациента
15	District	Текстовый	20	Район города, где проживает
16	PolicyNumber	Текстовый	20	Номер страхового полиса
17	Year	Числовой	4	Год рождения пациента
18	Sign	Логический	1	Работник предприятия (да/нет)
19	Department	Текстовый	40	Отдел, в котором работает
20	TreatyID	Числовой	10	Идент. номер записи на прием
21	DateStart	Дата	Авто	Дата приема
22	TimeStart	Текстовый	10	Время приема
23	Cost	Денежный	15	Стоимость приема
24	ExemptID	Числовой	2	Идентификатор льготы
25	ExemptType	Текстовый	60	Название льготы (инвалид, ветеран)
26	Exempt	Денежный	15	Сумма льготы
27	Summa	Денежный	15	К оплате
28	Comment	Поле Memo	Авто	Примечания (результаты приема)

Вариант 24. Разработать прикладное программное обеспечение деятельности рекламного агентства «Rapid».

В собственности этого агентства находится примерно около сотни рекламных щитов, расположенных по всему городу. Установка их согласована с администрацией города, и все необходимые формальности выполнены. На этих щитах может быть размещена реклама по заказу любой организации города. Срок размещения, стоимость аренды щита и стоимость изготовления самой рекламы – договорные.

Одна организация может арендовать несколько рекламных щитов. Один щит не сдается в аренду нескольким арендаторам, так как является неделимой рекламной единицей.

Договор размещения рекламы может быть продлен по взаимной договоренности сторон.

Таблица 9.24

Набор данных к варианту 24

№	Поле	Тип	Размер	Описание
1	RegNumber	Числовой	10	Регистрационный номер щита
2	Address	Текстовый	60	Адрес расположения щита
3	District	Текстовый	15	Район города
4	Orientation	Текстовый	60	Местоположение (ОДОРА, Депо-2 …)
5	Square	Числовой	Авто	Площадь рекламного щита
6	Size	Текстовый	10	Размеры
7	CustomerID	Числовой	10	ИНН арендатора щита
8	Status	Текстовый	15	Статус арендатора (ТОО, ЗАО, ИЧП)
9	Customer	Текстовый	40	Название арендатора
10	AddressCust	Текстовый	60	Юридический адрес арендатора
11	Chief	Текстовый	40	Руководитель
12	Phone	Текстовый	10	Телефон руководителя
13	Bank	Текстовый	60	Банк арендатора
14	Account	Текстовый	20	Номер счета в банке
15	Tax	Текстовый	15	Налоговая инспекция арендатора
16	Worker	Текстовый	40	Ответственный от арендатора
17	PhoneWorker	Текстовый	10	Телефон ответственного
18	TreatyID	Числовой	5	Номер договора аренды щита
19	DateStart	Дата	Авто	Начало действия договора
20	StopDate	Дата	Авто	Окончание действия
21	SignDate	Дата	Авто	Дата подписания договора
22	Advertisment	Логический	1	Изготовление рекламы (да/нет)
23	Cost	Денежный	15	Стоимость изготовления рекламы
24	Leasing	Денежный	15	Стоимость аренды щита
25	Picture	Поле OLE	Авто	Фотография щита с рекламой
26	Employee	Текстовый	40	Ответственный от агентства
27	Period	Текстовый	15	Оплата (ежемесячная, кварт., годовая)
28	Comment	Поле Memo	Авто	Дополнительные условия

Вариант 25. Разработать прикладное программное обеспечение деятельности ООО «Центр оценки и продажи недвижимости».

Одним из источников прибыли этой организации является покупка и продажа квартир. Центр оценки имеет большой штат специалистов, позволяющий этой организации проводить сделки купли-продажи на высоком профессиональном уровне. Владелец квартиры, желающий ее продать, заключает договор с Центром, в котором указывается сумма, срок продажи и процент отчислений в пользу Центра оценки и продажи недвижимости в случае успешного проведения сделки.

Один клиент может заключить с Центром более одного договора купли-продажи одновременно, если он владеет несколькими квартирами.

Обмен квартир специалисты центра непосредственно не производят. Для этих целей используется вариант купли-продажи.

Таблица 9.25

Набор данных к варианту 25

№	Поле	Тип	Размер	Описание
1	Registr	Числовой	10	Регистрационный номер клиента
2	Address	Текстовый	60	Адрес клиента
3	Name	Текстовый	60	ФИО клиента
4	Phone	Текстовый	10	Телефон для связи с клиентом
5	TreatyID	Числовой	5	Регистрационный номер договора
6	AddressFlat	Текстовый	60	Адрес квартиры
7	District	Текстовый	15	Район города
8	Floors	Числовой	2	Этажей в доме
9	Floor	Числовой	2	Этаж, на котором расположена квартира
10	TypeHouse	Текстовый	20	Тип дома (кирпичный, панельный)
11	TypePlan	Текстовый	20	Тип планировки (хрущевка, новая)
12	TypeToilet	Текстовый	20	Тип санузла (раздельный, совмещенный)
13	SqAll	Числовой	Авто	Общая площадь квартиры
14	SqLife	Числовой	Авто	Жилая площадь квартиры
15	SqKit	Числовой	Авто	Площадь кухни
16	Agent	Текстовый	40	Фамилия агента Центра оценки
17	Privat	Логический	1	Наличие приватизации
18	SignPhone	Логический	1	Наличие телефона в квартире
19	DateStart	Дата	Авто	Начало действия договора
20	StopDate	Дата	Авто	Окончание действия
21	Cost	Денежный	15	Стоимость квартиры
22	Bonus	Денежный	15	Вознаграждение Центра оценки
23	Picture	Поле OLE	Авто	Фотография здания
24	Plan	Поле OLE	Авто	План квартиры
25	Structure	Поле Memo	Авто	Инфраструктура территории
26	Document	Текстовый	40	Документ на право собственности
27	Prolong	Дата	Авто	Продление срока действия договора
28	Comment	Поле Memo	Авто	Дополнительные условия

Вариант 26. Разработать прикладное программное обеспечение деятельности отдела вневедомственной охраны квартир. Этот отдел обеспечивает электронную охрану квартир граждан в одном районе города. Для установки охранной сигнализации требуется наличие квартирного телефона. Один гражданин может заключить договор на охрану нескольких квартир. Из-за ложных срабатываний сигнализации возможно несколько выездов патрульных экипажей по одной квартире. На владельца квартиры, вовремя не отключившего сигнализацию после своего прихода домой, налагается штраф, величина которого оговаривается при заключении договора охраны. Если отдел вневедомственной охраны не уберег имущество владельца квартиры, то он выплачивает пострадавшему заранее оговоренную сумму. От величины этой суммы зависит размер ежемесячной оплаты за охрану квартиры.

Таблица 9.26

Набор данных к варианту 26

№	Поле	Тип	Размер	Описание
1	Registr	Числовой	10	Регистрационный номер клиента
2	Address	Текстовый	60	Адрес клиента
3	Name	Текстовый	60	ФИО клиента
4	Phone	Текстовый	10	Телефон для связи с клиентом
5	TreatyID	Числовой	5	Регистрационный номер договора
6	AddressFlat	Текстовый	60	Адрес квартиры
7	Key	Логический	1	Наличие кодового замка на подъезде
8	Floors	Числовой	2	Количество этажей в доме
9	Floor	Числовой	2	Этаж, на котором расположена квартира
10	TypeHouse	Текстовый	20	Тип дома (кирпичный, панельный)
11	TypeDoor	Текстовый	20	Тип квартирной двери (мет, дер, две шт.)
12	Balcony	Логический	1	Наличие балкона
13	TypeBalcony	Текстовый	60	Тип балкона (отдельный, совмещенный)
14	Plan	Поле OLE	Авто	План квартиры
15	Cost	Денежный	15	Стоимость ежемесячной оплаты
16	Compensation	Денежный	15	Компенсация при краже имущества
17	DateStart	Дата	Авто	Начало действия договора
18	StopDate	Дата	Авто	Окончание действия
19	ActionID	Числовой	7	Номер выезда на захват
20	PatrolID	Числовой	4	Номер экипажа, выезжавшего на захват
21	Chief	Текстовый	20	Командир экипажа
22	Brand	Текстовый	15	Марка автомобиля
23	DateTime	Дата	Авто	Дата и время выезда
24	False	Логический	1	Вызов ложный (да/нет)
25	Tax	Денежный	15	Величина штрафа за ложный вызов
26	Document	Текстовый	40	Документ, оформленный при задержании
27	Prolong	Дата	Авто	Продление срока действия договора
28	Comment	Поле Memo	Авто	Дополнительные условия

Вариант 27. Разработать прикладное программное обеспечение деятельности отдела приватизации жилья администрации города. В нашем городе на начало 2001 г. приватизировано около 80 000 квартир граждан. Еще далеко не все проживающие в «своих» квартирах стали собственниками своего жилья. Процесс приватизации продолжается и займет еще несколько лет. Главная задача программного комплекса – не допустить приватизации одним человеком более одной квартиры. К сожалению, в отделе приватизации не используется уникальный кадастровый номер здания, поэтому вам придется использовать составной первичный ключ (адрес) для таблицы зданий, квартир и проживающих. Помните, что некоторые из проживающих в квартире могут не участвовать в приватизации.

Таблица 9.27

Набор данных к варианту 27

№	Поле	Тип	Размер	Описание
1	Address	Текстовый	60	Адрес здания
2	District	Текстовый	15	Район города
3	Balance	Текстовый	60	Балансодержатель
4	Year	Числовой	4	Год постройки здания
5	Material	Текстовый	15	Материал стен здания
6	Base	Текстовый	15	Материал фундамента
7	Comment	Поле Memo	Авто	Примечания
8	Wear	Числовой	2	Износ в процентах
9	Flow	Числовой	2	Число этажей в здании
10	Line	Числовой	5	Расстояние от центра города
11	Square	Числовой	10	Площадь квартир
12	Hall	Логический	1	Наличие нежилых помещений
13	Picture	Поле OLE	Авто	Фото здания
14	Flats	Числовой	3	Число квартир в здании
15	Elevator	Логический	1	Наличие лифта
16	Flat	Числовой	4	Номер квартиры
17	Storey	Числовой	2	Номер этажа
18	Rooms	Числовой	1	Количество комнат
19	SquareFlat	Числовой	Авто	Общая площадь квартиры
20	Dwell	Числовой	Авто	Жилая площадь квартиры
21	Branch	Числовой	Авто	Всп. площадь квартиры
22	Balcony	Числовой	Авто	Площадь балкона
23	Height	Числовой	Авто	Высота квартиры
24	Record	Числовой	2	Номер записи о приватизации
25	Document	Текстовый	60	Документ на право приватизации
26	DateDoc	Дата	Авто	Дата документа о приватизации
27	Cost	Денежный	Авто	Инвентаризационная стоимость
27	FioHost	Текстовый	60	Ф.И.О. проживающего
28	Pasport	Поле Memo	Авто	Данные его паспорта
29	Sign	Логический	1	Участие в приватизации (да/нет)
30	Born	Числовой	4	Год рождения
31	Status	Текстовый	20	Статус в семье

Вариант 28. Разработать прикладное программное обеспечение деятельности предприятия «Газкомплект» по учету платы за пользование газом и газовыми приборами. Плата взымается с каждой квартиры в зависимости от количества потребленного газа или от числа проживающих, если счетчик отсутствует. Ответственный квартиросъемщик обязан каждый месяц снимать показания счетчика и производить оплату за потребленный газ через сбербанк. Наряду с отслеживанием платы за газ предприятие производит профилактическое обслуживание газовых приборов. Правила техники безопасности предусматривают осмотр газовой плиты инспектором предприятия раз в квартал. Если обнаружены неполадки в подключении плиты или ее работе, то работник предприятия обязан немедленно устранить их за счет абонента. Оплата оказанных услуг осуществляется на месте по квитанции.

Таблица 9.28

Набор данных к варианту 28

№	Поле	Тип	Размер	Описание
1	Address	Текстовый	60	Адрес здания
2	District	Текстовый	15	Район города
3	Material	Текстовый	15	Тип стен
4	Floor	Текстовый	15	Тип перекрытий
5	Picture	Поле OLE	Авто	Фото здания
6	Owner	Текстовый	15	Балансодержатель
7	Doorway	Числовой	2	Количество подъездов в доме
8	Flats	Числовой	3	Количество квартир в доме
9	LastName	Текстовый	20	Фамилия квартиросъемщика
10	FirstName	Текстовый	20	Имя квартиросъемщика
11	Patronymic	Текстовый	20	Отчество квартиросъемщика
12	Passport	Текстовый	20	Номер паспорта
13	Flat	Числовой	3	Номер квартиры
14	FlatType	Числовой	1	Вид квартиры
15	People	Числовой	2	Количество проживающих
16	Phone	Текстовый	15	Телефон квартиросъемщика
17	Account	Текстовый	15	Номер абонентской книжки
18	DateCount	Дата	Авто	Дата выдачи книжки
19	Stop	Логический	1	Наличие задвижки на входе в квартиру
20	Number	Текстовый	10	Номер счетчика
21	ViewNumber	Дата	Авто	Дата поверки счетчика
22	MadeIn	Текстовый	15	Страна изготовления счетчика
23	WhenMade	Дата	Авто	Дата изготовления счетчика
24	DateView	Дата	Авто	Дата снятия показаний
25	Result	Числовой	Авто	Показания счетчика
26	PayMonth	Денежный	Авто	Плата за месяц
27	Prophylaxis	Дата	Авто	Дата профилактического осмотра
28	PayDefect	Денежный	Авто	Стоимость исправления дефекта
29	Surname	Текстовый	20	Фамилия ответственного инспектора

Вариант 29. Разработать прикладное программное обеспечение деятельности «Бюро технической инвентаризации» по изготовлению и выдаче технических паспортов на объекты недвижимости. Перед регистрацией сделки с объектом недвижимости собственник объекта должен получить в БТИ на него технический паспорт. Ежедневно в БТИ обращается до 200 человек. Основное назначение программного комплекса – не пропустить ни одного документа. Если технический паспорт не готов в назначенный срок, то БТИ должно выплатить неустойку. Алгоритм изготовления документа следующий. Клиент обращается к инспектору, сдает ему необходимые справки, согласовывает дату выхода техника на обмер, уплачивает аванс. Инспектор передает заявку начальнику отдела. Начальник отдела назначает исполнителя и техника. Техник выполняет обмер объекта. Исполнитель изготавливает документ и передает в отдел выдачи. В назначенный срок клиент забирает готовый документ, доплатив недостающую сумму. Один клиент (физическое или юридическое лицо) может заказать несколько технических паспортов, за изготовление которых оплата может производиться частями.

Таблица 9.29

Набор данных к варианту 29

№	Поле	Тип	Размер	Описание
1	NumberClaim	Числовой	10	Номер заявки на изготовление документа
2	Name	Текстовый	60	ФИО заказчика
3	Phone	Текстовый	10	Телефон для связи с заказчиком
4	Receipt	Логический	1	Физическое или юридическое лицо
5	Bank	Текстовый	60	Банк заказчика
6	Account	Текстовый	20	Номер счета в банке
7	Address	Текстовый	60	Адрес объекта
8	District	Текстовый	15	Район города
9	DateStart	Дата	Авто	Дата приема заявки
10	Document	Текстовый	60	Название документа
11	Speed	Логический	1	Срочное изготовление (да/нет)
12	DateStop	Дата	Авто	Дата выдачи документа
13	Cost	Денежный	15	Стоимость изготовления документа
14	Inspector	Дата	Авто	Дата выхода техника
15	Time	Текстовый	20	Время выхода техника
16	Chief	Текстовый	30	ФИО начальника отдела
17	Worker	Текстовый	30	ФИО исполнителя
18	DateWorker	Дата	Авто	Дата передачи исполнителю
19	Helper	Текстовый	30	ФИО техника
20	Cash	Логический	1	Оплата наличными (да/нет)
21	DateCost	Дата	Авто	Дата оплаты
22	Value	Денежный	15	Оплаченная сумма
23	Finish	Логический	1	Документ выдан (да/нет)
24	Comment	Поле Memo	Авто	Примечания

Вариант 30. Разработать прикладное программное обеспечение деятельности отдела аренды ЗАО «Сириус». После удачной приватизации, когда у руководства этого предприятия оказалась большая часть акций, дела некогда мощного предприятия пошли на спад. Основная часть работников была уволена по сокращению штатов. В настоящее время основной статьей получения прибыли является сдача в аренду другим предприятиям и организациям площадей, которыми владеет «Сириус». В его собственности имеется 12-этажное здание, которое состоит примерно из 300 помещений. Почти все они сдаются в аренду.

Один арендатор может арендовать несколько помещений, причем срок аренды для каждого устанавливается отдельно. Величина арендной платы и ее периодичность устанавливается арендодателем. После окончания срока аренды, договор может быть продлен на прежних или новых условиях. Субаренда площадей запрещена. Закрытые договоры не удаляются из базы данных для отслеживания предыдущих арендаторов.

Таблица 9.30

Набор данных к варианту 30

№	Поле	Тип	Размер	Описание
1	CustomerID	Числовой	4	Идентификатор арендатора
2	CustomerType	Логический	1	Тип арендатора (физ./юр. лицо)
3	CustomerFio	Текстовый	60	ФИО арендатора (для физ. лица)
4	CustomerName	Текстовый	60	Название арендатора (для юр. лица)
5	CustomerInn	Текстовый	13	ИНН арендатора (для юр. лица)
6	Chief	Текстовый	40	Руководитель (для юр. лица)
7	Phone	Текстовый	10	Телефон арендатора
8	Address	Текстовый	60	Юридический адрес арендатора
9	Bank	Текстовый	60	Банк арендатора (для юр. лица)
10	District	Текстовый	15	Район заказчика
11	Worker	Текстовый	30	Ответственный от арендатора
12	PhoneWorker	Текстовый	10	Телефон ответственного
13	HallID	Числовой	3	Номер помещения
14	Square	Числовой	Авто	Площадь помещения
15	Size	Текстовый	20	Размеры помещения
16	Floor	Числовой	2	Этаж, на кот. расположено помещение
17	PhoneHall	Логический	1	Телефон в помещении (есть/нет)
18	Decoration	Текстовый	10	Отделка (обычная, улучш., евро)
19	TreatyID	Числовой	5	Номер договора аренды
20	Type	Логический	1	Договор действует/закрыт
21	DateStart	Дата	Авто	Дата начала действия договора
22	DateStop	Дата	Авто	Окончание срока действия
23	Period	Текстовый	20	Периодичность оплаты (ежемес., кварт)
24	Value	Денежный	15	Сумма оплаты
25	Inspector	Текстовый	30	Ответственный от арендодателя
26	Target	Текстовый	30	Цель аренды (офис, киоск, склад)
27	Tax	Денежный	15	Штраф за нарушение условий договора
28	Comment	Поле Memo	Авто	Примечания

Вариант 31. Разработать прикладное программное обеспечение деятельности телефонной компании.

Основное назначение программного комплекса – отслеживание абонентской платы за телефоны. Клиентами компании могут быть как физические лица, так и организации. Расчет с организациями ведется в безналичной форме через банк. Физические лица вносят плату через кассу компании.

Клиент телефонной компании может иметь несколько телефонных номеров. Дополнительная плата за подключенный параллельно аппарат не взимается. Если телефон у абонента не работает более суток, то плата за пользование телефоном уменьшается.

Междугородние и международные звонки оплачиваются отдельно по заранее установленным расценкам.

Таблица 9.31

Набор данных к варианту 31

№	Поле	Тип	Размер	Описание
1	CustomerID	Числовой	4	Идентификатор клиента компании
2	CustomerType	Логический	1	Тип клиента (физ./юр. лицо)
3	CustomerFio	Текстовый	60	ФИО клиента (для физ. лица)
4	CustomerName	Текстовый	60	Название клиента (для юр. лица)
5	CustomerInn	Текстовый	13	ИНН клиента (для юр. лица)
6	Chief	Текстовый	40	Руководитель (для юр. лица)
7	Phone	Текстовый	10	Телефон для связи (для юр. лица)
8	Address	Текстовый	60	Юридический адрес клиента
9	Bank	Текстовый	60	Банк клиента (для юр. лица)
10	Account	Текстовый	20	Номер счета в банке
11	PhoneNumber	Текстовый	10	Номер телефона
12	PhoneAddress	Текстовый	60	Адрес, где он установлен
13	Value	Денежный	15	Ежемесячная плата за телефон
14	ExemptType	Текстовый	20	Тип льготы
15	Exempt	Денежный	15	Величина льготы
16	DateClaim	Дата	Авто	Дата заявки о поломке телефона
17	NumberClaim	Числовой	5	Номер заявки о поломке
18	Inspector	Текстовый	15	Фамилия принявшего заявку
19	DateRepair	Дата	Авто	Дата восстановления связи
20	Compensation	Денежный	15	Вычеты из арендной платы
21	DateRing	Дата	Авто	Дата внешнего звонка
22	RingType	Логический	1	Междугородний/Международный
23	Number	Текстовый	10	Вызываемый номер
24	County	Текстовый	15	Страна
25	Town	Текстовый	15	Город
26	Value	Числовой	3	Количество минут
27	Summa	Денежный	15	Стоимость звонка
28	Comment	Поле Memo	Авто	Примечания

Вариант 32. Разработать прикладное программное обеспечение деятельности мелкооптового книжного магазина. Менеджер магазина, изучив спрос на книжную продукцию в городе, принимает решение о закупке партии книг в том или ином издательстве. Некоторые, пользующиеся повышенным спросом книги, могут быть закуплены у посредников. Часть продукции заказывается через Internet. Покупателем в мелкооптовом магазине может быть любой человек или организация, при условии, что величина покупки превысит одну тысячу рублей. Расчет с организациями производится через банк. Расчет с физическими лицами – наличными. Покупателю выписывается счет-фактура, которая имеет уникальный номер и содержит список книг с указанием их стоимости. После уплаты указанной суммы покупатель получает товар на складе.

Таблица 9.32

Набор данных к варианту 32

№	Поле	Тип	Размер	Описание
1	Provider	Текстовый	40	Поставщик книг
2	INN	Текстовый	10	ИНН поставщика книг
3	Address	Текстовый	60	Юридический адрес поставщика
4	Bank	Текстовый	60	Банк поставщика книг
5	Account	Текстовый	20	Номер счета в банке
6	Sign	Логический	1	Признак посредника
7	Film	Текстовый	20	Название книги
8	Author	Текстовый	60	Авторы
9	Comment	Поле Memo	Авто	Краткое содержание книги
10	Pages	Числовой	3	Количество страниц
11	Company	Текстовый	40	Издательство
12	Year	Числовой	4	Год издания
13	Cost	Денежный	15	Стоимость приобретения
14	Cdrom	Логический	1	Наличие компакт диска к книге
15	Customer	Текстовый	20	Название покупателя
16	CustomerSign	Логический	1	Признак покупателя (юр./физ.)
17	INNcustomer	Текстовый	10	ИНН покупателя
18	AddressCust	Текстовый	60	Юридический адрес покупателя
19	Chief	Текстовый	60	Директор
20	BankCustomer	Текстовый	60	Банк покупателя
21	Phone	Текстовый	10	Телефон для связи
22	District	Текстовый	15	Район города
23	AccountCust	Текстовый	20	Номер счета покупателя в банке
24	CountNumber	Числовой	4	Номер счет-фактуры
25	DateStart	Дата	Авто	Дата выписки счет-фактуры
26	Value	Денежный	15	Сумма к уплате
27	Worker	Текстовый	60	Выдавший счет-фактуру
28	Tax	Денежный	15	Величина налога с продаж
29	Plus	Поле Memo	Авто	Примечания

Вариант 33. Разработать прикладное программное обеспечение деятельности ОАО «Автовокзал».

Это открытое акционерное общество занимается междугородними пассажирскими перевозками по Дальневосточному региону. В его собственности находится несколько десятков автобусов различной вместимости.

Штат водителей превышает количество автобусов. Средний уровень сменности для машины – 2.5. Водитель не может работать более одной смены в сутки. Билеты на рейсы продаются только в здании автовокзала. Возможна предварительная продажа. Маршрут автобуса может пролегать через несколько населенных пунктов. В этом случае пассажир может купить билет до любого промежуточного пункта. Освободившимся местом после выхода пассажира распоряжается водитель. Полученную выручку он сдает в кассу предприятия после прибытия с маршрута. На линии работает контроль. Если в автобусе будет обнаружен пассажир без билета, то на водителя налагается штраф.

Таблица 9.33

Набор данных к варианту 33

№	Поле	Тип	Размер	Описание
1	LastName	Текстовый	20	Фамилия водителя
2	FirstName	Текстовый	20	Имя водителя
3	Patronymic	Текстовый	20	Отчество водителя
4	Experience	Числовой	2	Стаж работы
5	Year	Числовой	4	Год рождения
6	Category	Текстовый	20	Категория водителя (D,E)
7	Class	Текстовый	20	Классность водителя (1, 2, 3)
8	DriverID	Числовой	4	Идентификационный номер водителя
9	BusNumber	Числовой	4	Идентификационный номер автобуса
10	Brand	Текстовый	15	Марка автобуса
11	Picture	Поле OLE	Авто	Фотография автобуса
12	Model	Текстовый	15	Модель автобуса
13	Capacity	Числовой	2	Количество мест в автобусе
14	YearBus	Числовой	4	Год выпуска автобуса
15	YearRepair	Числовой	4	Год капитального ремонта
16	Distance	Числовой	6	Пробег автобуса, км
17	RouteID	Числовой	3	Идентификатор маршрута
18	PointStart	Текстовый	20	Начальный пункт
19	PointStop	Текстовый	20	Конечный пункт
20	DateStart	Дата	Авто	Дата отправления
21	TimeStart	Текстовый	10	Время отправления
22	TimeAll	Текстовый	10	Время в пути до конечного пункта
23	PlaceNumber	Числовой	2	Номер места
24	PlaceSign	Логический	1	Билет продан на автовокзале (да/нет)
25	SumDriver	Денежный	15	Выручка на маршруте
26	SumTax	Денежный	15	Штраф на водителя
27	Comment	Поле Memo	Авто	Промежуточные пункты маршрута

Вариант 34. Разработать прикладное программное обеспечение деятельности агентства знакомств.

Агентство занимается организацией знакомств одиноких мужчин и женщин. Возможен один из двух вариантов: человек либо регистрируется в агентство, оставляет информацию о себе, чтобы любой мог ознакомиться с его кандидатурой, либо знакомится с базой уже зарегистрированных и выбирает подходящую кандидатуру. Регистрация и подбор кандидатуры – платные. Тот, кто делает выбор по базе, платит за каждый выбранный вариант. После того, как выбор сделан, агентство согласовывает дату и время встречи с каждой из сторон, формирует и передает приглашения для знакомства обеим сторонам. Во избежание недоразумений первая встреча происходит в агентстве.

Клиенты, желающие быть исключенными из базы, переносятся в архив.

Таблица 9.34

Набор данных к варианту 34

№	Поле	Тип	Размер	Описание
1	CandidateID	Числовой	5	Регистрационный номер кандидата
2	Name	Текстовый	40	ФИО кандидата
3	Gender	Числовой	1	Пол кандидата
4	Age	Числовой	2	Возраст кандидата
5	Myself	Поле Memo	Авто	Информация кандидата о себе
6	Demand	Поле Memo	Авто	Требования кандидата к избраннику
7	Phone	Текстовый	15	Телефон кандидата
8	Picture	Поле OLE	Авто	Фотография кандидата
9	Registrar	Текстовый	15	Фамилия регистрирующего
10	RegDate	Дата	Авто	Дата регистрации кандидата
11	ClientID	Числовой	5	Регистрационный номер клиента
12	ClientName	Текстовый	40	ФИО клиента
13	ClientGender	Числовой	1	Пол клиента
14	ClientPhone	Текстовый	15	Телефон клиента
15	Date	Дата	Авто	Дата регистрации клиента
16	ClientMyself	Текстовый	200	Информация клиента о себе
17	ClientMore	Текстовый	50	Дополнительная информация
18	ClientAge	Числовой	2	Возраст клиента
19	ReceiptID	Числовой	5	Номер квитанции об оплате
20	PayDate	Дата	Авто	Дата оплаты
21	Seller	Текстовый	15	Фамилия кассира
22	Cash	Логический	1	Оплата наличными (да/нет)
23	Sum	Денежный	5	Сумма
24	MeetDate	Дата	Авто	Согласованная дата встречи
25	MeetTime	Время	Авто	Время встречи
26	InvitationID	Числовой	5	Номер приглашения
27	DeleteDate	Дата	Авто	Дата перевода в архив
28	Deleter	Текстовый	15	Фамилия удалившего в архив
29	Reason	Текстовый	30	Причина переноса в архив

Вариант 35. Разработать прикладное программное обеспечение деятельности ломбарда.

Человек обращается в ломбард в том случае, если ему срочно нужны деньги. Например, недостает небольшой суммы для покупки квартиры, а подходящая квартира как раз продается, и на неё уже есть и другие покупатели. Тогда человек может пойти в ломбард и заложить вещи на необходимую сумму.

В ломбарде с клиентом заключается договор. В нем оговариваются следующие условия: до какого срока выкуп вещи возможен без процентов, с какого времени будет взыматься процент, по истечении какого срока выкуп вещи невозможен, и она поступает в собственность ломбарда. Невыкупленные вещи ломбард выставляет на продажу.

Таблица 9.35

Набор данных к варианту 35

№	Поле	Тип	Размер	Описание
1	ClientID	Числовой	4	Регистрационный номер клиента
2	Name	Текстовый	40	ФИО клиента
3	Date	Дата	Авто	Дата обращения в ломбард
4	Address	Текстовый	60	Адрес клиента
5	District	Текстовый	15	Район проживания
6	Phone	Текстовый	15	Телефон клиента
7	Passport	Текстовый	20	Номер паспорта
8	PassportDate	Дата	Авто	Дата выдачи паспорта
9	Region	Текстовый	40	Кем выдан паспорт
10	Agreement	Числовой	5	Номер договора
11	StartDate	Дата	Авто	Дата приема вещи
12	PercentDate	Дата	Авто	Дата, с которой за выкуп берется пени
13	StopDate	Дата	Авто	Дата, с которой выкуп уже невозможен
14	Registrar	Текстовый	15	Фамилия приемщика
15	ThingID	Числовой	4	Регистрационный номер вещи
16	Category	Текстовый	15	Категория вещи
17	Thing	Текстовый	30	Название вещи
18	Count	Числовой	2	Количество принятых вещей
19	Defects	Логический	1	Наличие дефектов (да/нет)
20	Cost	Денежный	10	Оценочная стоимость вещи
21	Sum	Денежный	10	Сумма, полученная клиентом
22	Comment	Поле Memo	Авто	Примечания
23	Back	Дата	Авто	Дата выкупа вещи
24	Tax	Денежный	10	Пени за несвоевременный выкуп
25	BackSum	Денежный	10	Сумма, заплаченная за выкуп
26	Seller	Текстовый	15	Фамилия вернувшего вещь
27	FreeThing	Числовой	4	Номер вещи, поступившей в продажу
28	Price	Денежный	10	Цена вещи, поступившей в продажу

Вариант 36. Разработать прикладное программное обеспечение деятельности гостиницы.

В любой уважающей себя гостинице существует большое количество возможных вариантов заселения гостей: все номера различаются по категориям (суперлюкс, люкс и т. д.), по количеству комнат в номере, количеству мест в каждом номере, а также по обустройству комнат – учитывается, например, наличие телевизора, холодильника, телефона.

В обязанности администратора гостиницы входит подбор наиболее подходящего для гостя варианта проживания, регистрация гостей, прием платы за проживание, оформление квитанций, выписка отъезжающих. Учитывается также возможность отъезда гостя раньше указанного при регистрации срока, при этом производится перерасчет. Существует также услуга бронирования номера.

Таблица 9.36

Набор данных к варианту 36

№	Поле	Тип	Размер	Описание
1	GuestID	Числовой	4	Регистрационный номер гостя
2	Name	Текстовый	40	ФИО гостя
3	Date	Дата	Авто	Дата регистрации
4	Address	Текстовый	60	Адрес гостя
5	Town	Текстовый	20	Город, из которого приехал гость
6	Aim	Текстовый	30	Цель приезда
7	Passport	Текстовый	20	Номер паспорта
8	PassportDate	Дата	Авто	Дата выдачи паспорта
9	Region	Текстовый	40	Кем выдан паспорт
10	Work	Текстовый	20	Место работы или учебы
11	Year	Числовой	4	Год рождения гостя
12	Money	Денежный	10	Плата за выбранный номер
13	Cash	Логический	1	Оплата наличными (да/нет)
14	Receipt	Числовой	4	Номер квитанции
15	End	Дата	Авто	Дата отъезда
16	Comment	Поле Memo	Авто	Примечания
17	Registrar	Текстовый	15	Фамилия администратора
18	Picture	Поле OLE	Авто	Фотография номера
19	Number	Числовой	4	№ номера
20	Rooms	Числовой	2	Количество комнат в номере
21	Storey	Числовой	2	Номер этажа
22	TV	Логический	1	Телевизор (есть/нет)
23	Fridge	Логический	1	Холодильник (есть/нет)
24	Bed	Числовой	2	Количество мест в номере
25	Type	Числовой	1	Категория номера
26	Balcony	Логический	1	Балкон (есть/нет)
27	Reservation	Числовой	4	Забронированный номер
28	ReservName	Текстовый	40	ФИО забронировавшего номер
29	Come	Дата	Авто	Дата приезда
30	Leave	Дата	Авто	Дата предполагаемого отъезда

Вариант 37. Разработать прикладное программное обеспечение института селекции растений.

Данный институт занимается сбором, выведением и продажей различных сортов семян. В его ассортименте можно найти семена практически всех возможных видов растений: от помидоров до редких цветов. Только что выведенные сорта заносятся в отдельный список для дальнейшего тестирования. Каждый сорт семян имеет свои характеристики, такие как урожайность, морозоустойчивость, адаптация к местным условиям, сроки созревания (раннеспелый, среднеспелый, поздний) и т. п. Покупатель может выбрать сорт, отвечающий тем или иным характеристикам. Компания занимается как оптовыми, так и розничными продажами. Оптовые покупатели заносятся в базу главным образом для того, чтобы информировать их о поступлении новых или отсутствовавших в определенный момент в продаже сортов.

Таблица 9.37

Набор данных к варианту 37

№	Поле	Тип	Размер	Описание
1	SortID	Числовой	10	Уникальный номер сорта
2	Name	Текстовый	40	Название сорта
3	Year	Числовой	4	Год, в котором выведен сорт
4	Adaptation	Логический	1	Адаптация к местным условиям (да/нет)
5	Frost	Логический	1	Морозоустойчивый (да/нет)
6	Description	Текстовый	200	Описание характеристик сорта
7	Technology	Поле Memo	Авто	Способ посадки
8	Picture	Поле OLE	Авто	Фотография представителя сорта
9	Part	Числовой	10	Номер партии
10	End	Дата	Авто	Годен до
11	Yes	Логический	1	Одобрен инспекцией (да/нет)
12	Batch	Текстовый	20	Вид упаковки
13	Amount	Числовой	4	Количество семян в упаковке
14	BatchTime	Дата	Авто	Дата расфасовки
15	Weight	Числовой	4	Вес семян в упаковке
16	Period	Числовой	1	Срок созревания
17	NewSort	Числовой	5	Номер нового сорта для тестирования
18	Date	Дата	Авто	Дата выведения нового сорта
19	Comment	Поле Memo	Авто	Примечания
20	ClientID	Числовой	5	Номер оптового покупателя
21	Company	Текстовый	25	Название фирмы-покупателя
22	BuyDate	Дата	Авто	Дата покупки
23	Address	Текстовый	60	Адрес фирмы-покупателя
24	Phone	Текстовый	15	Телефон покупателя
25	Sum	Денежный	10	Цена покупки
26	Cash	Логический	1	Оплата наличными (да/нет)
27	Seller	Текстовый	15	Фамилия продавца

Вариант 38. Разработать прикладное программное обеспечение деятельности приемной комиссии университета.

Каждый год университет зачисляет новых абитуриентов для возможного их поступления в университет после сдачи вступительных экзаменов. На бюджетную основу могут быть зачислены: абитуриенты, получившие на школьном экзамене высокий балл ЕГЭ и успешно прошедшие собеседование; абитуриенты, набравшие необходимый для бесплатного поступления балл на университетских экзаменах, а также абитуриенты, имеющие направление от какого-либо государственного предприятия. Все остальные могут поступить в университет на платной основе, набрав необходимое установленное университетом число баллов на вступительных экзаменах.

Таблица 9.38

Набор данных к варианту 38

№	Поле	Тип	Размер	Описание
1	PersonID	Числовой	5	Регистрационный номер абитуриента
2	Name	Текстовый	40	ФИО абитуриента
3	Date	Дата	Авто	Дата регистрации
4	Picture	Поле OLE	Авто	Фотография абитуриента
5	Address	Текстовый	60	Домашний адрес абитуриента
6	Phone	Текстовый	15	Телефон абитуриента
7	Birth	Дата	Авто	Дата рождения абитуриента
8	School	Текстовый	20	Название оконченной школы
9	Money	Логический	1	Возможность оплаты обучения (да/нет)
10	Passport	Текстовый	20	Номер паспорта
11	PassportDate	Дата	Авто	Дата выдачи паспорта
12	Region	Текстовый	40	Кем выдан паспорт
13	Attestat	Текстовый	20	Номер аттестата
14	Middle	Числовой	Авто	Средний балл аттестата
15	Faculty	Текстовый	40	Название выбранного факультета
16	Speciality	Текстовый	40	Название выбранной специальности
17	Registrar	Текстовый	15	Фамилия принявшего документы
18	Talk	Логический	1	Рекомендован для собеседования
19	Result	Числовой	Авто	Набрано баллов при поступлении
20	Contract	Логический	1	Договор с предприятием (да/нет)
21	ContractID	Числовой	10	Номер договора
22	ContractAbout	Поле Memo	Авто	Условия договора
23	ContAddress	Текстовый	60	Адрес предприятия
24	ContPhone	Текстовый	15	Телефон предприятия
25	Payment	Логический	1	Платит предприятие (да/нет)
26	EGE	Текстовый	20	Номер аттестата ЕГЭ
27	Lesson	Текстовый	20	Предмет, по которому сдан ЕГЭ
28	Score	Числовой	Авто	Набрано баллов по ЕГЭ
29	YesID	Числовой	5	Номер после зачисления
30	Type	Числовой	1	Основа, на которой зачислен

Вариант 39. Разработать прикладное программное обеспечение деятельности кассы авиакомпании. Касса авиакомпании занимается продажей билетов на предстоящие рейсы. В билете указывается номер и название рейса, а также все остальные необходимые для пассажира данные: дата и время вылета, прибытия, номер места и класс (бизнес, экономический). Цена билета зависит от рейса, лайнера, класса, а также от времени покупки билета – иногда авиакомпании делают скидки купившим билет более чем за месяц или на “горящие рейсы” – все зависит от желания компании. Билеты продаются только совершеннолетним гражданам при предъявлении паспорта. У авиакомпании обычно имеется несколько касс, расположенных в разных концах города, поэтому обязательно необходимо учитывать номер кассы, в которой был продан билет, во избежание недоразумений при сдаче или обмене билета.

Таблица 9.39

Набор данных к варианту 39

№	Поле	Тип	Размер	Описание
1	Passage	Числовой	5	Номер рейса
2	Title	Текстовый	40	Название рейса
3	Date	Дата	Авто	Дата вылета
4	Time	Время	Авто	Время вылета
5	Arrival	Дата	Авто	Дата прибытия
6	ArrivalTime	Время	Авто	Время прибытия
7	Seats	Логический	1	Промежуточные посадки (есть/нет)
8	Places	Текстовый	50	Места промежуточных посадок
9	Passenger	Числовой	5	Регистрационный номер пассажира
10	Name	Текстовый	40	ФИО пассажира
11	Date	Дата	Авто	Дата покупки билета
12	Passport	Текстовый	20	Номер паспорта
13	PassportDate	Дата	Авто	Дата выдачи паспорта
14	Region	Текстовый	40	Кем выдан паспорт
15	Till	Числовой	5	Номер билетной кассы
16	Ticket	Числовой	10	Номер билета
17	Chair	Числовой	3	Номер места
18	Class	Числовой	1	Класс
19	Price	Денежный	6	Цена билета
20	Registrar	Текстовый	15	Фамилия регистратора
21	Airliner	Числовой	10	Номер лайнера
22	AirlinerName	Текстовый	15	Название лайнера
23	Year	Числовой	4	Год создания
24	Picture	Поле OLE	Авто	Фотография лайнера
25	Amount	Числовой	4	Количество совершенных рейсов
26	Repair	Дата	Авто	Дата последнего техосмотра
27	Crew	Числовой	4	Номер экипажа
28	Pilot	Текстовый	15	Фамилия пилота
29	CrewNames	Поле Memo	Авто	Фамилии остальных членов экипажа

Вариант 40. Разработать прикладное программное обеспечение деятельности предприятия по учету платы за потребленную электроэнергию. Плата взымается с каждой квартиры в зависимости от количества потребленной энергии или от числа проживающих, если счетчик отсутствует. Существует несколько методик начисления абонентской платы. Плата зависит от вида счетчика (однофазный, трехфазный), от типа счетчика (возможность учета дневного и ночного тарифов), а также от вида квартиры (коммунальная, отдельная).

Ответственный квартиросъемщик обязан каждый месяц снимать показания счетчика и производить оплату за потребленную электроэнергию через сбербанк. Второй экземпляр квитанции он обязан хранить у себя и предъявлять инспектору по первому требованию.

Таблица 9.40

Набор данных к варианту 40

№	Поле	Тип	Размер	Описание
1	Address	Текстовый	60	Адрес здания
2	District	Текстовый	15	Район города
3	Material	Текстовый	15	Тип стен
4	Floor	Текстовый	15	Тип перекрытий
5	Picture	Поле OLE	Авто	Фото здания
6	Owner	Текстовый	15	Балансодержатель
7	Doorway	Числовой	2	Количество подъездов в доме
8	Flats	Числовой	3	Количество квартир в доме
9	LastName	Текстовый	20	Фамилия квартиросъемщика
10	FirstName	Текстовый	20	Имя квартиросъемщика
11	Patronymic	Текстовый	20	Отчество квартиросъемщика
12	Passport	Текстовый	20	Номер паспорта
13	Flat	Числовой	3	Номер квартиры
14	FlatType	Числовой	1	Вид квартиры
15	People	Числовой	2	Количество проживающих
16	EStove	Логический	1	Наличие электроплиты (есть/нет)
17	Number	Текстовый	10	Номер счетчика
18	Kind	Числовой	1	Вид счетчика
19	Type	Числовой	1	Тип счетчика
20	Factor	Числовой	Авто	Коэффициент фазности
21	Comment	Поле Memo	Авто	Примечания
22	Tariff	Числовой	Авто	Плата за чел. в месяц без счетчика
23	Day	Числовой	Авто	Плата по дневному тарифу
24	Night	Числовой	Авто	Плата по ночному тарифу
25	24hour	Числовой	Авто	Плата по круглосуточному тарифу
26	BeginMonth	Числовой	Авто	Показания счетчика в начале месяца
27	EndMonth	Числовой	Авто	Показания счетчика в конце месяца
28	Privilege	Денежный	Авто	Льгота в месяц на квартиру в рублях
29	Surname	Текстовый	20	Фамилия ответственного инспектора

ОГЛАВЛЕНИЕ

Учебное издание

Гурвиц Геннадий Александрович

**РАЗРАБОТКА РЕАЛЬНОГО ПРИЛОЖЕНИЯ
С ИСПОЛЬЗОВАНИЕМ MICROSOFT VISUAL FOXPRO 9**

Учебное пособие

Редактор *А.А. Иванова*
Технические редакторы *С.С. Заикина*

План 2007 г.
Сдано в набор 28.09.2006 г. Подписано в печать 03.11.2006 г.
Формат 60×84 1/16. Бумага тип. № 2. Гарнитура «Arial». Печать RISO.
Усл. изд. л. 8,4. Усл. печ. л. 11,6. Зак. 295. Тираж 300 экз.

Издательство ДВГУПС
680021, г. Хабаровск, ул. Серышева, 47.

Гурвиц Геннадий Александрович
Россия, Хабаровск. Тел. 8-4212-35-91-33
E-mail: gurvits@mail.ru

После окончания в 1978 году Хабаровского института инженеров железнодорожного транспорта (ныне Университет) приглашен руководством института на преподавательскую работу, которая продолжается и по сей день. Последние годы – в должности доцента кафедры вычислительной техники и прикладной математики (с 1998 года – кафедра "Информационные технологии и системы").

С 1982 по 1985 год – аспирантура в Московском институте инженеров транспорта (МИИТ). В 1985 году защитил диссертацию на соискание ученой степени кандидата технических наук. С 1979 по 2006 год опубликовал около 20 научных статей в различных изданиях. Область интересов: технологии программирования, операционные системы, базы данных. Кроме чтения курсов по всем основным языкам программирования и операционным системам написал более 25 серьезных программных продуктов. Наиболее крупные из них:

В 1979-1981 годах для строительных организаций Дальневосточного региона "Программы расчета строительных ферм на ЭВМ серии ЕС". Язык PL/1. ОС ЕС ЭВМ. В 1982-1985 годах для Всесоюзного научно-исследовательского института транспортного строительства программный комплекс по расчету нелинейно-деформируемых пластинчатых систем. Языки ALGOL и PL/1.

В 1986-1989 годах для Химико-технической лаборатории Дальневосточной железной дороги "Программный комплекс для статистической обработки результатов испытаний". Язык Turbo Pascal 5.5. Операционная система MS DOS 3.30.

В 1995-1996 годах "Прикладное программное обеспечение деятельности комитета по управлению муниципальным имуществом и приватизированными объектами администрации города Хабаровска". FoxPro 2.0.

В 1997 году "Прикладное программное обеспечение деятельности Регистрационной палаты г.Хабаровска". FoxPro 2.6.

В 1998 году для Управления технической инвентаризации г. Хабаровска программные комплексы "Недвижимость" и "Прием заявок на изготовление документов". Microsoft Visual FoxPro 3.0.

В 1999 году для информационно-аналитического центра налоговой полиции программный комплекс "Владельцы". Microsoft Visual FoxPro 5.0.

В 2000 году для краевого бюро технической инвентаризации программный комплекс по учету недвижимости в Хабаровском крае. Microsoft SQL Server 7.0 (Сервер). Microsoft Visual FoxPro 6.0 (клиент).

В 2001 году для Центра оценки и продажи недвижимости администрации г.Хабаровска программный комплекс "ЦО и ПН". Microsoft Visual FoxPro 6.0.

В 2000-2003 годах для Департамента муниципальной собственности администрации города Хабаровска несколько программных комплексов: "Земля", "Банк данных города", "Департамент", "Реклама", "Приватизация жилья" и др. Microsoft SQL Server 7.0 (Сервер). Microsoft Visual FoxPro 6.0 (клиент) + Microsoft Visual Basic 6 for Applications + Microsoft Excel 2003 и Microsoft Word 2003.

В 2004 году для Хабаровского предприятия электрических сетей Военно-Морского флота комплекс "Учет электроэнергии". MS SQL Server 2000 (Сервер). MS Visual FoxPro 8.0 (клиент).