ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

Кафедра прикладной информатики и информационных технологий

РАЗРАБОТКА АВТОМАТИЗИРОВАННОГО РАБОЧЕГО МЕСТА МЕНЕДЖЕРА ПО РАБОТЕ С КЛИЕНТАМИ ООО «ВЕРЕСК-М»

Выпускная квалификационная работа

студентки заочной формы обучения
направления подготовки 09.03.03 Прикладная информатика
5 курса группы 07001151
Дерипаско Карины Сергеевны

Научный руководитель:
к.т.н., доцент Гахова Н.Н.

БЕЛГОРОД 2016

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современных условиях для сопровождения организации в любой профессиональной сфере деятельности для обеспечения качественного и оперативного управления основными бизнес-процессами необходимо использовать информационные технологии.

Актуальность выбранной темы выпускной квалификационной работы обоснована следующим:

– в последнее время значительно увеличилось количество обрабатываемой информации;

– ужесточились требования к оформлению документации;

– увеличение объема документов на бумажных носителях значительно усложняет и замедляет время рассмотрения клиентских заявок и оформления договоров.

Объектом исследования является отдел по работе с клиентами ООО «Вереск-М».

Предметом исследования выступает бизнес-процесс сопровождения клиента.

Целью работы является повышение производительности компании.

Поставленная цель может быть достигнута разными методами, в данной работе для достижения цели будет разработано автоматизированное рабочее место менеджера по работе с клиентами для ООО «Вереск-М».

В процессе достижения цели планируется выполнить следующие задачи:

– изучить организационную структуру ООО «Вереск-М»;

– исследовать предметную область деятельности предприятия;

– сформулировать основные требования к разрабатываемому АРМ;

– провести анализ и обосновать выбор программных средств;

– спроектировать АРМ менеджера по работе с клиентами;

– разработать АРМ менеджера по работе с клиентами;

– протестировать АРМ менеджера по работе с клиентами.

Пояснительная записка к выпускной квалификационной работе состоит из введения, трех разделов, заключения, списка использованных источников и приложений.

Во введении обоснована актуальность работы, описаны объект и предмет исследования, сформулирована цель, поставлены задачи, решение которых необходимо для достижения поставленной цели, дано содержательное описание пояснительной записки выпускной квалификационной работы.

В первом разделе рассмотрена технико-экономическая характеристика организации и отдела по работе с клиентами ООО «Вереск-М». Выполнено обоснование необходимости использования ПК, представлена экономическая сущность задачи исследования. Для проведения анализа существующего подхода к организации бизнес-процесса были разработаны диаграммы, построена и проанализирована модель деятельности организации «Как есть».

Во втором разделе описаны проектные решения в рамках технического, информационного, программного и технологического обеспечений. Также представлено обоснование программных средств разработки автоматизированного рабочего места.

В третьем разделе рассмотрен процесс проектирования и разработки АРМ менеджера по работе с клиентами. Построена модель информационной базы данных и дана ее характеристика. Также рассчитана экономическая эффективность внедрения АРМ и калькуляция себестоимости данного проекта.

В заключении приведены основные результаты, полученные в ходе выполнения выпускной квалификационной работы, сделаны выводы.

Список использованных источников включает 46 наименований.

В приложении представлены материалы, характеризующие ход выполнения выпускной квалификационной работы. Также приведены рисунки и программный код разработанного АРМ менеджера по работе с клиентами ООО «Вереск-М».

Работа изложена на 77 страницах основного текста, содержащего 36 рисунков и 7 таблиц.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Технико-экономическая характеристика предметной области

1.1.1 Характеристика предприятия

Строительная индустрия – совокупность постоянно действующих (в основном подрядных, строительно-монтажных) организаций, которая является важнейшим звеном индустриализации строительства.

Одним из элементов отрасли строительства в Белгородской области является строительная организация ООО «Вереск-М», которая расположена по адресу: 308013, Российская Федерация, Белгородская область, г. Белгород, Михайловское шоссе, д. 14.

Общество с ограниченной ответственностью «Вереск-М» (ООО «Вереск-М») обладает полной хозяйственной самостоятельностью, уставной капитал которого является собственностью учредителя. Компания имеет в собственности обособленное имущество, учитываемое на его самостоятельном балансе, может от своего имени приобретать и осуществлять имущественные и личные неимущественные права, исполнять обязанности, быть истцом и ответчиком в суде.

Дата создания компании совпадает с днем официальной регистрации - 29 декабря 1995 года. В своей деятельности ООО «Вереск-М» руководствуется федеральными и местными нормативными документами и законодательными актами, а также Уставом организации, который является единственным документом, регламентирующим структуру данного предприятия, внутреннее устройство, виды деятельности, взаимодействие с другими организациями, лицами, государственными и муниципальными органами, права и обязанности в соответствии с действующим гражданским законодательством и т.д.

Организация тесно сотрудничает с государственными, бюджетными, казенными и хозрасчетными организациями на территории города Белгорода, Белгородской, Воронежской и Курской областях.

ООО «Вереск-М» в большей степени оказывает услуги в следующих областях:

– монтажно-строительные работы;

– косметический и капитальный ремонт помещений;

– реставрация и реконструкция сооружений и зданий.

ООО «Вереск-М» оказывая высококачественные услуги (как строительная организация), регулярно выполняет государственные заказы.

Виды продукции и услуг (по кодам ОКПД), реализуемые организацией:

– работы по подготовке строительного участка (стройплощадки);

– общестроительные работы по возведению зданий учреждений здравоохранения;

– общестроительные работы по пристройке к одноквартирным и двухквартирным домам дополнительных помещений (веранд, туалетов и др.) из неметаллических конструкций, выполняемые по индивидуальным заказам;

– общестроительные работы по ремонту одноквартирных и двухквартирных домов, кроме работ, выполняемых по индивидуальным заказам;

– общестроительные работы по ремонту одноквартирных и двухквартирных домов, выполняемые по индивидуальным заказам;

– общестроительные работы по возведению складских и производственных зданий;

– общестроительные работы по строительству индивидуальных гаражей, выполняемые по индивидуальным заказам.

Основная цель строительной организации ООО «Вереск-М», как коммерческого предприятия, заключается в получении прибыли.

ООО «Вереск-М» – это строительная организация, существующая более 20 лет. За прошедшие годы организацией было успешно реализовано ряд проектов, которые позволили предприятию занять одно из лидирующих положений в своей отрасли и зарекомендовать себя как надежный и долгосрочный партнёр.

Обществом с ограниченной ответственностью «Вереск-М» предоставляется широкий спектр услуг в сфере строительства и ремонта. ООО «Вереск-М», обеспечивает:

– соблюдение тщательной предварительной подготовки к проекту;

– разработку подробных планов;

– компетентную организацию процессов, связанных с ремонтом и отделкой;

– мониторинг над деятельностью, осуществляемой организацией.

Ответственность, внимательность и аккуратность при работе с документацией, к которой относятся оформление документов, составление заказа или заключение договора, характеризуют ООО «Вереск-М» как профессиональную компанию в своей сфере деятельности, которая серьезно подходит к выполнению своих обязательств.

ООО «Вереск-М» учрежден несколькими лицами, уставной капитал которого разделен на доли, определенными учредительными документами (устав и учредительный договор). Учредители ООО «Вереск-М» не отвечают по его обязательствам и несут риск убытков, связанных с деятельностью общества в пределах стоимости внесенными ими вкладов. Выбывающий вправе продать или уступить ее одному, либо нескольким участникам ООО «Вереск-М» или третьему лицу.

Участники пользуются преимущественным правом покупки доли, но, если они не воспользуются этим правом в течении месяца, предусмотренным уставом, доля может быть отчуждена третьему лицу либо приобретена самим обществом.

Основной государственный регистрационный номер организации, присвоенный при регистрации: 1023101660361, дата присвоения ОГРН: 22.11.2002, ИНН: 3123030057, КПП: 312301001. Директором организации является Курской Михаил Анатольевич.

В ООО «Вереск-М» можно выделить следующие структурные единицы:

– директор;

– отдел бухгалтерского учета, возглавляемый главным бухгалтером;

– отдел кадров, возглавляемый руководителем отдела;

– отдел по работе с клиентами;

– производственный отдел, которым руководит главный инженер организации.

Организационная структура управления ООО «Вереск-М» представлена на рисунке 1.1.

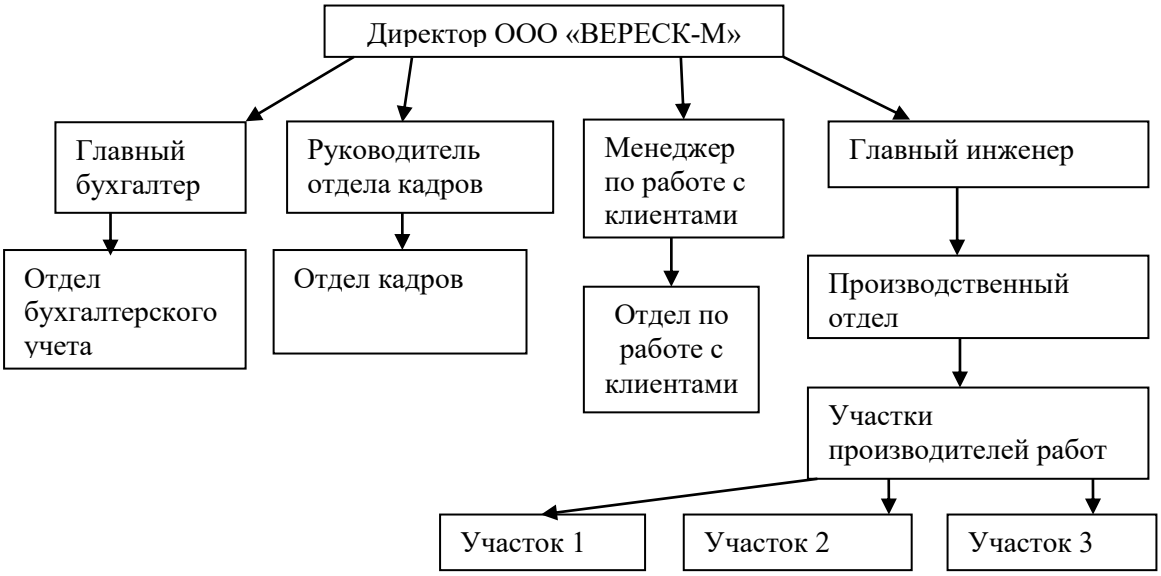


Рисунок 1.1 – Организационная структура управления ООО «Вереск-М»

Должностные обязанности директора и функции перечисленных отделов ООО «Вереск-М» описаны в приложении А

1.1.2 Краткая характеристика отдела по работе с клиентами ООО «Вереск-М»

Специфика деятельности ООО «Вереск-М», которая заключается в том, что при документировании процессов выполнения подготовительных, строительных, монтажных и ремонтных работ задействуются практически все

структурные подразделения организации, при этом отделу по работе с клиентами отводится особое место.

Менеджерами отдела обеспечивается взаимодействие организации с клиентами, выступая в роли персональных консультантов в ходе выполнения каждого заказа. Качество работы менеджера отдела оказывает большое влияние на количество заключенных договоров с заказчиками и на привлечение потенциальных клиентов [16]. Менеджерами по работе с клиентами проводятся действия по заключению договоров и осуществлению окончательных расчетов. Также в их должностные обязанности входит осуществление контроля сроков выполнения строительных работ по договорам, для того, чтобы своевременно реагировать и, по возможности, предотвращать претензии со стороны заказчика. Организация ведения базы данных клиентов, взаимодействие с подразделениями ООО для своевременного получения необходимой информации – все это обеспечивают менеджеры по работе с клиентами для осуществления внешних связей с заказчиками [33].

Менеджер по работе с клиентами принадлежит к категории «специалисты» и непосредственно подчиняется генеральному директору.

Назначение и освобождение от должности менеджера по работе с клиентами осуществляется приказом генерального директора по представлению начальника отдела продаж.

На должность менеджера по работе с клиентами назначается лицо, имеющее высшее образование и опыт работы на аналогичной должности не менее двух лет.

В период отсутствия менеджера по работе с клиентами его функциональные обязанности, ответственность, права переходят к иному должностному лицу, о чем сообщается в приказе по организации.

Менеджер по работе с клиентами знает: правила заключения и выполнения договоров; основы законодательства РФ; порядок предоставления услуг, поставки товаров; правила обслуживания клиентов, установленные в организации; характеристики, номенклатуру товаров; правила оформления

документации; нормы делового общения и этикета; правила противопожарной защиты, охраны труда, техники безопасности; отчетные документы, направляемые непосредственному руководителю.

В отделе для работы используются следующие программные продукты: Microsoft Office Word 2010 и Microsoft Office Excel 2010. Для выполнения заказа клиента оформляются документы: заявление клиента, техническое задание, дефектная ведомость либо рабочий проект, смета на строительство/ремонт/монтаж объекта, договор, акт приемки выполненных работ. Оформление производится в Microsoft Office Word 2010 [9].

В Microsoft Office Excel 2010 ведется работа над базой данных клиентов ООО «Вереск-М». Она должна обладать такими функциями как возможность ввода данных, их обработка и хранение. Всеми необходимыми параметрами обладает табличный процессор Microsoft Office Excel. Используя данную электронную таблицу можно создавать как простые, так и сложные базы данных, хранящие информацию об услугах, товарах, сотрудниках, заказах и т.д

1.2 Экономические аспекты решения задачи

В настоящее время перед все большим количеством компаний и организаций ставятся вопросы, которые необходимо решать для их дальнейшего успешного функционирования и динамичного развития. К таким задачам можно отнести внедрение новых технологий, поиск новых рынков сбыта и заказчиков, активное участие в тендерах.

Использование устаревших методов работы для достижения цели по этим направлениям чаще не дают желаемого результата. Одним из решений рассмотренных проблем может стать активное использование новых информационных технологий (НИТ) и программных решений [27].

Рассмотрев и проанализировав деятельность менеджера по работе с клиентами ООО «Вереск-М» можно отметить особенности и недостатки.

Деятельность менеджера связана не только с большим количеством

информации, передаваемой в другие отделы, но с неоднородностью и неравнозначностью данных[40]. Например, оформление заявки на строительство дома. Менеджер должен отслеживать не только информацию, относящуюся к производству, ремонту или монтажу, и вести документацию по каждому объекту, но и отслеживать дополнительную информацию, связанную с подготовительной частью (в качестве примера, можно отметить отслеживание документов по участкам).

Менеджер ведет документацию всех клиентов, по каждому из которых формируется пакет документов в бумажном виде (включая главный документ – договор), как следствие, быстрый отбор нужных данных по каждому клиенту вызывает трудности и приводит к потерям времени.

Менеджер осуществляет первичную связь с клиентом, далее часть работы передается в зависимости от их вида в другие подразделения, но отследить потоки документов затруднительно. Например, взаимодействие осуществляется передачей документов по выбору участка, строительству, монтажу или ремонту, выполненных в бумажном виде, что весьма неудобно, учитывая дислокацию подразделений, которые территориально разнесены [15].

Для автоматизации деятельности менеджера по работе с клиентами экономическую постановку задачи можно рассмотреть через возможность повышения результативности труда посредством автоматизации рутинной деятельности.

В качестве результата ВКР планируется разработать АРМ менеджера отдела по работе с клиентами, которое должно реализовывать следующее: регистрацию поступающей в отдел документации, оперативное оформление клиентов и их заявок на оказание услуги.

Преимуществом разрабатываемого АМР можно отметить отсутствие некоторого количества документов на бумажных носителях, удобную возможность регистрации договоров с заказчиками, регистрацию клиентов, заказов, отслеживание оплаты, выгрузку готовых шаблонов для оформления договоров и заявок.

Предполагается, что разработанная система уменьшит временные затраты при поиске необходимого договора или заявки по заданной дате, обеспечит хранение большого объема информации с меньшими финансовыми затратами и позволит уменьшить время на оформление клиента и/или заказа [11]. Внедрение АРМ менеджера в отдел по работе с клиентами приведет к:

– снижению трудовых затрат;

– возможности наиболее эффективным образом модифицировать технологию создания и движения документов внутри компании;

– сокращению времени обработки информации на конкретную операцию;

– повышению скорости работы с внутренними документами;

– максимальному сокращению количества документов на бумажных носителях;

– ускорению получения различных отчетов, как периодических, так и по запросу.

1.3 Обоснование необходимости и цели использования вычислительной техники для решения задачи

Проведенный анализ поставленной задачи показал, что традиционными средствами учета ее решение возможно, но неэффективно. К традиционным (базовым) средствам относится обработка документов на бумажных носителях. Кроме того, возникают сложности в процессе хранения информации и проведения поиска по конкретному клиенту или заказу, если вся документация представлена в бумажном виде [30].

Дополнительными возможностями разрабатываемой системы является наличие подсистемы анализа (аналитическая информация), то есть АРМ предоставляет новые функциональные возможности. Использование модуля аналитической информации позволит получить любую информацию о совершенных сделках, провести анализ в разрезе востребованности услуг,

сравнить данные за несколько периодов. В традиционном виде (бумажном) подобные действия занимали очень много времени на просмотр и обработку документации и отвлекали сотрудников от непосредственных должностных обязанностей.

Следующим преимуществом можно назвать обеспечение безопасности хранения всей информации и документации в электронном виде. При традиционном способе потеря бумажного носителя приводила к невосполнимой потери документа. Если информация хранится в электронном виде, то возможно дублирование необходимых документов и информации, что обеспечивает ее надежное хранение.

К преимуществам также можно отнести возможность использования шаблонов, что позволяет, введя шаблон один раз в систему, использовать его многократно, а также извлечение информации из справочников, без вторичного внесения, например, данных клиента. Таким образом, внедрив АРМ менеджера, можно сократить время и увеличить скорость оформления документации.

Предполагается, что в результате внедрения системы в любой момент времени можно будет иметь точные сведения по всей необходимой информации: (количество заявок, их виды, клиенты), а также сократить время на подготовку отчетов и передачу документов за счет электронной формы [38].

При проведении анализа был изучен документооборот отдела по работе с клиентами. Входящими документами являются:

1) документы клиента:

– документ, удостоверяющий его личность;

– оригинал либо копию документа, подтверждающего право владения объектом, на территории которого будут производиться строительные, монтажные или ремонтно-отделочные работы;

– доверенность.

2) прайс-лист услуг;

3) чертежи, эскизы клиента;

4) документ на оплату.

Основные документы, с помощью которых происходит осуществление деятельности менеджера, – это нормативные и методические документы, распоряжения руководства и должностные инструкции.

Вся бумажная документация обрабатывается менеджером. Накопление все большего количества документов на бумажных носителях приводит к проблеме, которая связана с быстротой их обработки.

Рассмотрим деятельность отдела по работе с клиентами ООО «Вереск-М» как структурного подразделения. Так как отдел представлен как функционирующее подразделение в рамках организации, то для построения модели была использована нотация IDEF0. На рисунке 1.2 приведена контекстная диаграмма «Как есть», на рисунке 1.3 представлена декомпозиция контекстной диаграммы.

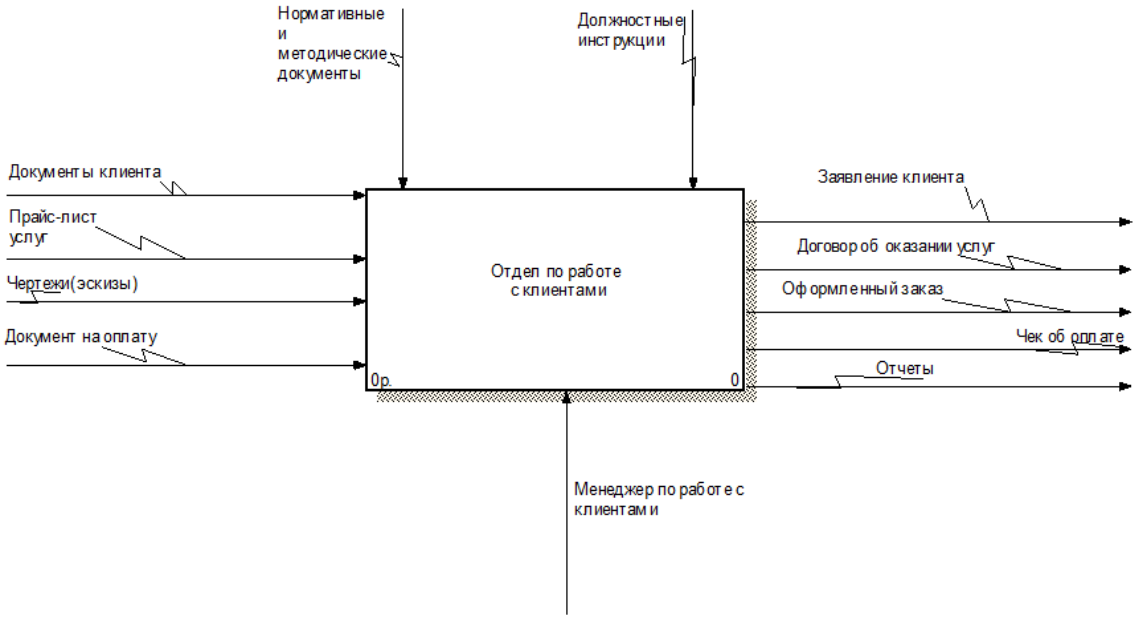


Рисунок 1.2 – Контекстная диаграмма «Отдел по работе с клиентами»

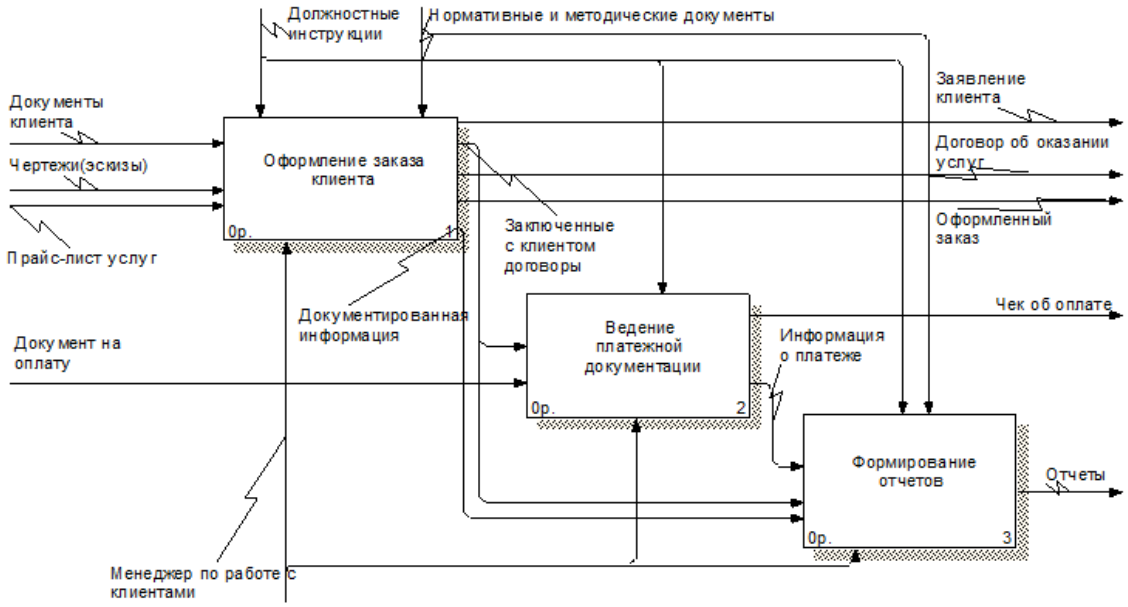


Рисунок 1.3 – Декомпозиция контекстной диаграммы «Отдел по работе с клиентами»

Блок «Оформление заказа клиента» включает деятельность менеджера по принятию заказа от клиента. В процессе выполнения строительных и работ в ООО «Вереск-М» задействованы многие подразделения, в том числе отдел по работе с клиентами. Менеджеры данного отдела являются связующим звеном между предприятием и заказчиком, выступают в роли персональных консультантов клиентов в ходе выполнения заказа. Именно от работы менеджера отдела зависит количество заключенных договоров с заказчиками и привлечение потенциальных клиентов. Менеджеры по работе с клиентами заключают договоры и проводят окончательные расчеты. Также они производят контроль сроков выполнения строительных работ по договорам для своевременного реагирования и предотвращения претензий со стороны заказчика, организуют ведение базы данных договоров (клиентов), взаимодействуют с подразделениями компании для своевременного получения информации, необходимой для осуществления внешней связи с клиентами.

Для выполнения заказа клиента оформляются следующие документы:

– заявление клиента;

– дефектная ведомость либо проект (ремонтно-отделочные работы выполняются на основании дефектной ведомости, строительные – по проекту);

– смета на строительство или ремонт объекта;

– договор;

– акт приемки выполненных работ (в случае поэтапного выполнения заказа – промежуточный акт приемки выполненных работ);

– счет на оплату;

– счет-фактура.

Все эти документы создаются в нескольких подразделениях ООО «Вереск-М»: отделе по работе с клиентами, отделе подготовки строительства, строительно-монтажном отделе, бухгалтерии. В обязанности менеджера отдела по работе с клиентами входит контроль своевременного поступления и согласования необходимых документов в подразделения предприятия, а также информирование заказчика о состоянии выполненных работ по его запросу.

Технология документирования выполнения заказа клиента следующая: клиент приходит в отдел по работе с клиентами, менеджер информирует его о составе услуг, выполняемых компанией, знакомит с прейскурантом цен, сроками выполняемых работ, затем внимательно выслушивает клиента и рассматривает его заказ, при необходимости консультирует клиента по оптимальному составлению заказа. Клиент пишет заявление и приносит с собой, если имеются, чертежи или эскизы.

Клиент при подаче заявления представляет менеджеру документы:

1) документ, удостоверяющий его личность,

2) оригинал или копию документа, подтверждающего право владения объектом, на территории которого будут производиться строительные или ремонтно-отделочные работы (помещение, квартира, земельный участок);

3) доверенность.

Декомпозиция блока «Оформление заказа клиента» представлена на рисунке 1.4.

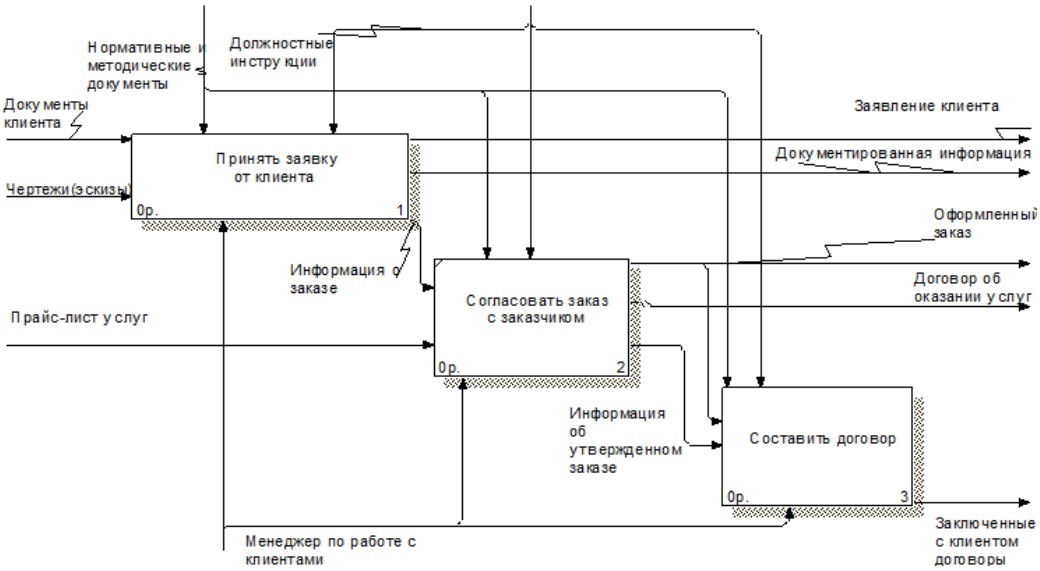


Рисунок 1.4 – Декомпозиция блока «Оформление заказа клиента»

«Ведение платежной документации» предполагает оформление оплаты клиента за услуги, предоставляемые ООО «Вереск-М» (см. рисунок 1.5).

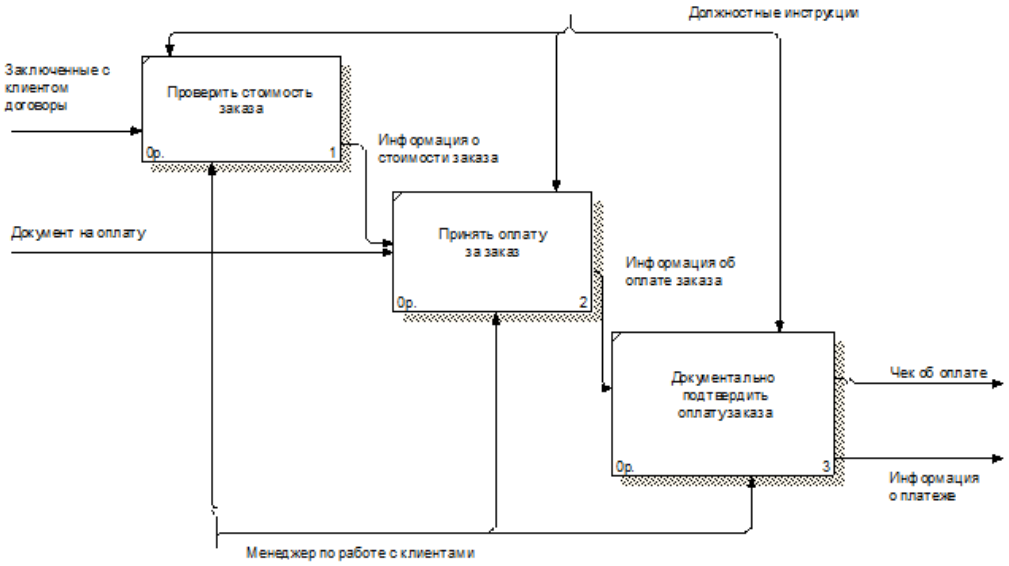


Рисунок 1.5 – Декомпозиция блока «Ведение платежной документации»

Из приведенных на рисунках 1.2-1.5 диаграмм видно, что менеджеру по работе с клиентами приходится вести достаточно объемную документацию, Заявление клиента передается на рассмотрение в отдел подготовки строительства для согласования сроков выполнения работ, объема расходных материалов и стоимости данного заказа.

Главный инженер отдела подготовки строительства определяет

сложность заказа и визирует заявление клиента (проставляется резолюция для составления сметы). Специалисты отдела подготовки строительства составляют смету и передают ее в отдел по работе с клиентами менеджеру, ведущему данный заказ. Подготовленная смета предоставляется клиенту на рассмотрение. Данный документ содержит наименование работ и затрат, сроки выполнения заказа согласовываются сторонами. Менеджер согласовывает смету с заказчиком и информирует его о цене. Если заказчика устраивает цена, то составляется договор. Договор с заказчиком играет главную роль в процессе документирования деятельности отдела по работе с клиентами. Договор на выполнение строительных работ ООО «Вереск-М», заключенный с клиентом, называется договором подряда. В нем содержится информация о предмете договора, правах и обязанностях сторон, заключивших его, ответственность сторон, порядок приемки работ, требования к качеству, порядок расчета по договору. Также в договоре содержится информация о заказчике и исполнителе, их подписи свидетельствуют о согласии с оговоренными в договоре условиями. Договор подряда составляется в двух экземплярах и имеет равную юридическую силу. После составления менеджер подписывает договор у генерального директора и предоставляет на подпись заказчику. Один экземпляр остается в отделе по работе с клиентами, а другой передается заказчику. В настоящее время в действующих нормативных документах не содержится обязательных к исполнению правил об оформлении договора подряда. Российским законодательством форма договора подряда не унифицирована. Структура договора подряда определяется заключающими его сторонами. Но есть ряд правил, соблюдение которых при составлении текста договора позволяет сторонам впоследствии избежать недоразумений и споров, связанных с различиями в понимании тех или иных условий договора. Поэтому очень важно, чтобы договор содержал все необходимые реквизиты, а его условия не противоречили действующему законодательству. Формулировки условий договора должны быть точными, исключающими возможность их двоякого толкования.

1.4 Постановка задачи

1.4.1 Цель и назначение автоматизированного варианта решения задачи

В процессе ознакомления с деятельностью предприятия ООО «Вереск-М» была поставлена задача автоматизации деятельности менеджера по работе с клиентами для повышения эффективности работы ООО «Вереск-М», скорости обработки информации о клиентах, контроля сроков оплаты, сокращения сроков формирования отчетов работы менеджера по работе с клиентами за месяц, актов выполнения работ.

Основные функции, которые должно выполнять АРМ менеджера по работе с клиентами:

– добавление и редактирование информации о клиентах, их контактов;

– оперативное получение информации о клиентах с использованием функции поиска;

– добавление и редактирование данных договоров;

– ведение реестра договоров;

– ведение реестра отправки и возврата актов выполненных работ;

– контроль оплаты заказов и услуг, предоставленных клиентам по договорам;

– выдача и формирование отчета в соответствии с требуемым шаблоном;

– предоставление возможности печати необходимых документов (пакета договора, акта выполненных работ, отчета о проделанной работе менеджера по работе с клиентами за месяц).

Целью автоматизации работы менеджера по работе с клиентами является увеличение производительности компании.

Отличительной особенностью АРМ менеджера по работе с клиентами

является его ориентация на принятие управленческих решений. По сравнению с АРМ экономиста, бухгалтера, маркетолога и т.д., предназначенных для решения регламентированных и формализованных функциональных задач управления, АРМ руководителя должно помочь в принятии управленческих решений с часто нерегламентированных и трудно формализованных заданием.

Автоматизация рабочего места представляет собой организацию места пользователя-специалиста той или иной профессии, оборудование средствами, необходимыми для автоматизации выполнения им определенных функций. Такими средствами, как правило, является ПК, дополняемый по мере необходимости другими вспомогательными электронными устройствами.

С помощью автоматизации деятельности менеджера по работе с клиентами можно будет определить текущее положение ООО «Вереск-М», вести историю взаимоотношений с клиентами, получить аналитическую информацию за необходимый период времени. Все это позволит принимать верные управленческие решения, снизить издержки, уменьшить временные затраты, тем самым увеличить прибыль предприятия.

1.4.2 Общая характеристика организации решения задачи на электронной вычислительной машине

Разрабатываемая АИС должна обеспечивать автоматизированный контроль, а также учет поставок в ООО «Вереск-М». Для этого создаваемая система должна:

– обеспечивать ввод данных, связанных с поставками в компании и осуществлять их обработку;

– создавать отчетные документы;

– иметь систему помощи по программе;

– создаваемые документы должны отвечать отраслевым стандартам, принятым в ООО «Вереск-М».

Система должна быть доступна с различными правами директору

предприятия, начальнику склада, начальнику отдела снабжения и бухгалтерам. Заданные характеристики функционирования должны обеспечиваться при условиях, которые определяются конкретным носителем данных, на котором они хранятся.

В качестве организации архитектуры аппаратной платформы решения задачи была выбрана архитектура клиент-сервер.

Технология клиент-сервер – способ взаимодействия программных компонентов, образующих единую систему. Как видно из самого названия, существует некий клиентский процесс, требующий определенных ресурсов, а также серверный процесс, который эти ресурсы предоставляет. Совсем необязательно, чтобы они находились на одном компьютере. Обычно принято размещать сервер на одном узле локальной сети, а клиентов – на других узлах.

В контексте базы данных клиент управляет пользовательским интерфейсом и логикой приложения, действуя как рабочая станция, на которой выполняются приложения баз данных. Клиент принимает от пользователя запрос, проверяет синтаксис и генерирует запрос к базе данных на языке SQL или другом языке базы данных, соответствующем логике приложения. Затем передает сообщение серверу, ожидает поступления ответа и форматирует полученные данные для представления их пользователю. Сервер принимает и обрабатывает запросы к базе данных, после чего отправляет полученные результаты обратно клиенту. Такая обработка включает проверку полномочий клиента, обеспечение требований целостности, а также выполнение запроса и обновление данных. Помимо этого, поддерживается управление параллельностью и восстановлением.

Архитектура клиент-сервер обладает рядом преимуществ:

– обеспечивается более широкий доступ к существующим базам данных;

– повышается общая производительность системы: поскольку клиенты и сервер находятся на разных компьютерах, их процессоры способны выполнять приложения параллельно. Настройка производительности

компьютера с сервером упрощается, если на нем выполняется только работа с базой данных;

– снижается стоимость аппаратного обеспечения (достаточно мощный компьютер с большим устройством хранения нужен только серверу для хранения и управления базой данных);

– сокращаются коммуникационные расходы. Приложения выполняют часть операций на клиентских компьютерах и посылают через сеть только запросы к БД, что позволяет значительно сократить объем пересылаемых по сети данных;

– повышается уровень непротиворечивости данных. Сервер может самостоятельно управлять проверкой целостности данных, поскольку лишь на нем определяются и проверяются все ограничения. При этом каждому приложению не придется выполнять собственную проверку;

– архитектура клиент-сервер естественно отображается на архитектуру открытых систем.

1.5 Анализ существующих разработок и обоснование выбора технологии проектирования

На рынке представлено несколько программных продуктов, которые справляются с задачей автоматизации деятельности менеджера по работе с клиентами. Ниже приведено описание таких систем.

CRM решения (от англ. customer relationship management) компании ASoft представляют собой ряд готовых CRM систем, которые разработаны для различных отраслей и размеров бизнеса и призваны решить следующие задачи:

– повысить уровень продаж;

– улучшить управление процессами и обслуживание клиентов;

– оптимизировать маркетинговую деятельность компании.

CRM системы, разработанные компанией, помогут накапливать и систематизировать данные о клиентах, партнерах и поставщиках, а также

эффективно использовать их для принятия решений. Вследствие получим не только хранилище постоянно обновляющихся сведений, но и появится система управления процессами, обеспечивающая возможность оперативно взаимодействовать с клиентами, учитывать все пожелания, а также контролировать работу сотрудников.

ASoft CRM – это автоматизированная система управления взаимоотношениями с контрагентами, затрагивающая три основных направления деятельности: маркетинг, продажи и сервис. Для различных задач наших клиентов были разработаны соответствующие CRM решения.

ASoft iCRM - приложение для работы с данными CRM программы через iPhone в online и offline режимах.

При разработке CRM решений компания ASoft руководствуется своими стандартными принципами разработки систем управления. Система должна быть: простой, функциональной и не содержать скрытой стоимости.

Рабочее окно программы ASoft CRM представлено на рисунке 1.6.

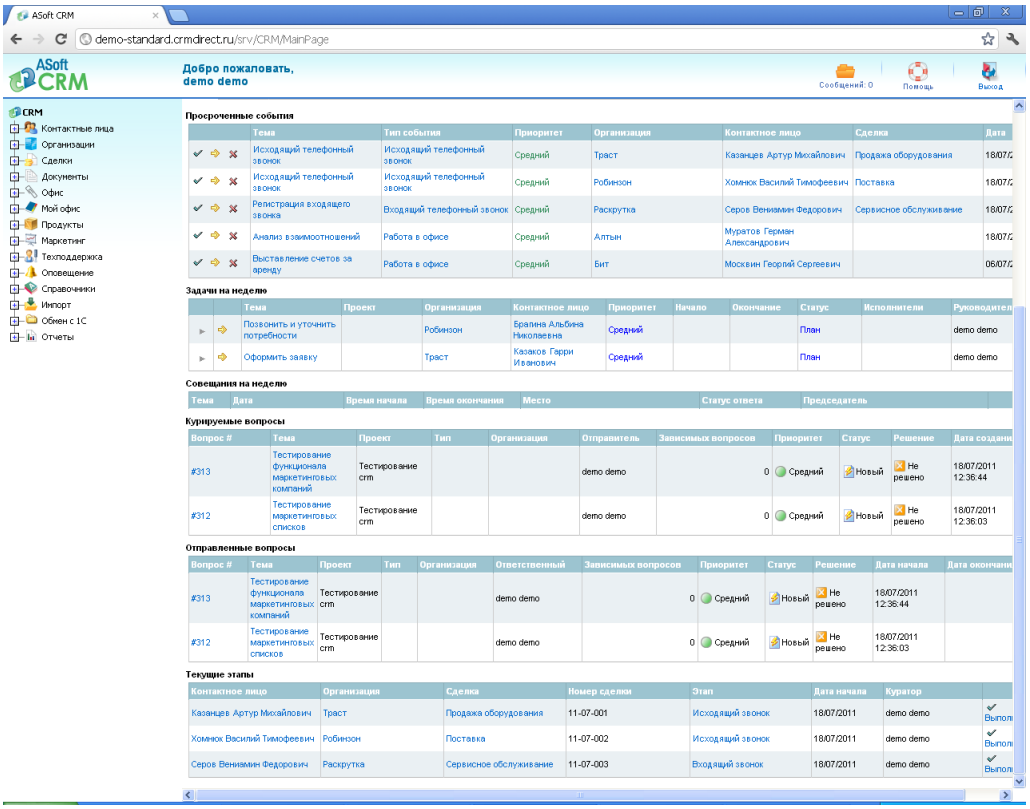


Рисунок 1.6 – Рабочее окно программы ASoft CRM

Управление продажами с помощью ASoft CRM позволяет сократить цикл продаж, сделать объемы сбыта прогнозируемыми, свести к минимуму организационные проблемы, не потерять потенциальных и, что не менее важно, удержать существующих клиентов.

ASoft CRM дает возможность вести полную историю взаимодействия со всеми контрагентами:

– реквизиты, контактная информация, координаты контрагентов;

– сделки, заключенные договора, этапы заключения сделок, платежная история;

– взаимодействие в области маркетинга;

– встречи, телефонные переговоры, обмен электронной корреспонденцией и другие события;

– активность компаний-конкурентов;

– документооборот по сделкам с контрагентами.

Рабочее окно программы «Учет клиентов» представлено на рисунке 1.7.

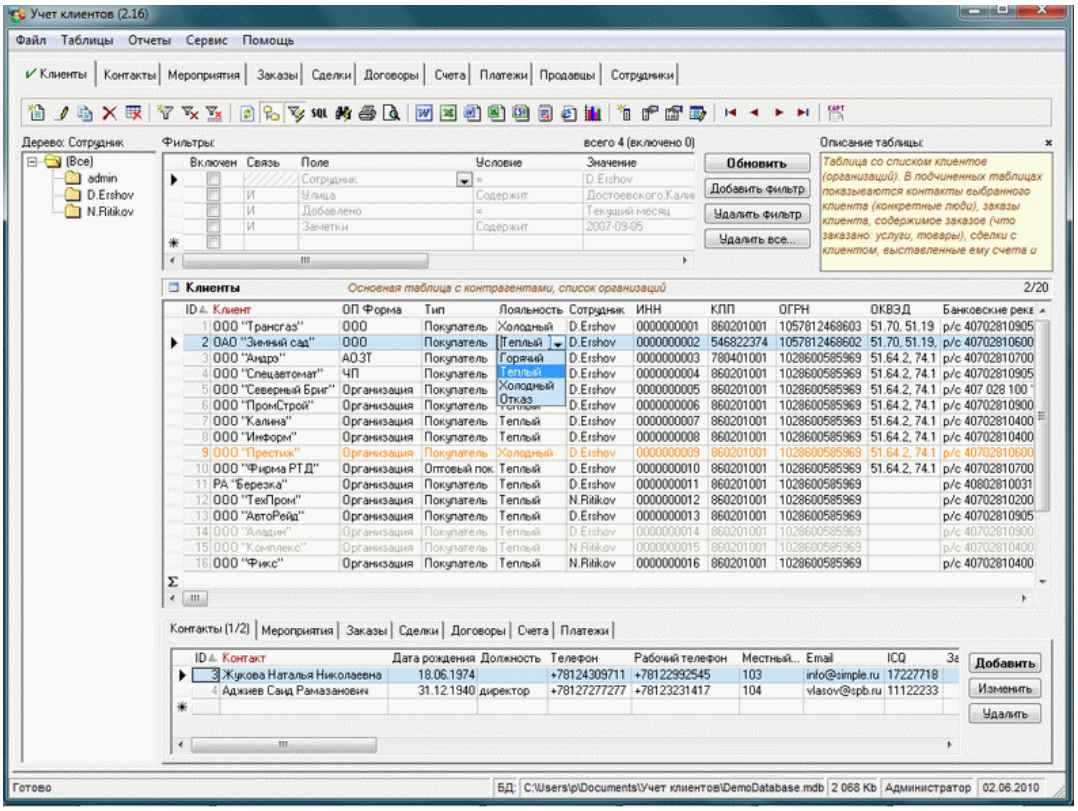


Рисунок 1.7 – Рабочее окно программы «Учет клиентов»

CRM программа позволяет наладить эффективное взаимодействие с партнерами по бизнесу:

1) каждое входящее событие, касающееся контрагента, будет зарегистрировано и обработано, ответ на него будет находиться на контроле у соответствующего руководителя;

2) использование встроенных инструментов, которые оптимизируют управление продажами, позволяет планировать и контролировать деятельность персонала и ставит сделки на поток;

3) взаимодействие с партнерами по бизнесу и клиентами будет осуществляться согласно заранее определенным бизнес-процессам.

1С:Управление торговлей и взаимоотношениями с клиентами (CRM) 2.0. – программа позволяет регистрировать любые взаимодействия с клиентами и поддерживает оформление всех основных первичных документов торгового учета.

С помощью расширенного функционала «1С:Управление торговлей и взаимоотношениями с клиентами (CRM)» учитывается вся информация в процессе работы с клиентом:

– отдельные факты взаимодействия с клиентами (встречи, звонки, e-mail и др.);

– последовательность и регламент действий по продаже, сервисному обслуживанию или разбору жалобы (бизнес-процессы);

– совершение хозяйственных операций;

– предпродажное и послепродажное взаимодействие (в т.ч. сервисное);

– маркетинговые воздействия (реклама, анкетирование и опросы).

Основные направления развития в новой редакции конфигурации «1С:Управление торговлей и взаимоотношениями с клиентами (CRM)»:

– развитие управленческой функциональности;

– повышение масштабируемости системы;

– использование новых возможностей версии 8.2 платформы 1С:Предприятие».

Итак, в первом разделе выпускной квалификационной работы была обоснована актуальность разработки АРМ, определена цель, поставлены задачи. Была рассмотрена технико-экономическая характеристика предметной области ООО «Вереск-М, обоснована необходимость использования ПК, представлена экономическая постановка задачи исследования. Были проанализированы существующие программные продукты, позволяющие вести учет клиентов, заказов, договоров.

Для проведения анализа существующего подхода к организации бизнес-процесса, связанного с взаимодействием с клиентами организации, были разработаны диаграммы деятельности ООО «Вереск-М» (было выбрано CASE-средство) AllFusion Process Modeler с использованием нотации IDEF0. Построена и проанализирована модель деятельности организации «Как есть».

2 ОБОСНОВАНИЕ ПРОЕКТНЫХ РЕШЕНИЙ

2.1 Обоснование проектных решений по техническому обеспечению

Техническое обеспечение – комплекс технических средств, предназначенных для работы информационной системы, а также соответствующая документация на эти средства и технологические процессы.

Комплекс технических средств составляют:

– компьютеры любых моделей;

– устройства сбора, накопления, обработки, передачи и вывода информации;

– устройства передачи данных и линий связи;

– оргтехника и устройства автоматического съема информации;

– эксплуатационные материалы и др.

Компьютерная техника предназначена, в основном, для реализации комплексных технологий обработки и хранения информации и является базой интеграции всех современных технических средств обеспечения управления информационными ресурсами.

Документацией оформляются предварительный выбор технических средств, технологический процесс обработки данных, технологическое оснащение.

Документация делится на:

– общесистемную, включающую государственные и отраслевые стандарты по техническому обеспечению;

– специализированную, содержащую комплекс методик по всем этапам разработки технического обеспечения;

– нормативно-справочную, используемую при выполнении расчетов по техническому обеспечению.

Характеристики персонального компьютера, на котором работает менеджер по работе с клиентами, представлены в таблице 2.1.

Таблица 2.1 – Техническое обеспечение отдела по работе с клиентами

Наименование	Характеристика
Монитор Samsung	тип - жк-монитор, диагональ – 17, разрешение – 1280x1024 (5:4); тип жк-матрицы – TFTTN, яркость кд/м2, максимальное количество цветов – 16,2, блок питания встроенный.
Системный блок	тип процессора: Intel Core TMi3, рабочая частота: 3.3 ГГц, объем оперативной памяти: 4096 Мб, объем накопителя: 500 Гб, оптический привод: DWD-RW, видеокарта: NVIDIA GeForce GT 520X, установленнаяОС:Windows 7.
Клавиатура Oklick 720G	настольный компьютер, интерфейс подключения: USB, цвет: черный, количество клавиш:104, размеры (ШхВхГ):470x30x195 мм.
МышьOklick 105 M	проводная мышь, интерфейс USB, светодиодная, 3 клавиши, разрешение сенсора мыши 800 dpi.
Принтер Epson L800	принтер, A4, печать пьезоэлектрическая струйная цветная, 6-цветная, 37 стр/мин ч/б, 38 стр/мин цветн., 5760x1440 dpi, подача: 120 лист., вывод: 50 лист., USB, печать фотографий, печать на CD/DVD.
Маршрутизатор беспроводной D-link DIR-632	Wi-Fi-точка доступа (роутер), стандарт Wi-Fi: 802.11n, макс. скорость: 300 Мбит/с, коммутатор 8xLAN, поддержка VPN, скорость портов 100 Мбит/сек, принт-сервер: USB.
Коммутатор неуправляемый D-link DES-1008A/E1A	коммутатор (switch), 8 портов Ethernet 10/100 Мбит/сек.
Точка доступа Wi-fiZyXEL KeeneticUltra	гигабитная Wi-Fi точка доступа, 802.11n, MIMO, 750 Мбит/с, маршрутизатор, коммутатор 4xLAN, принт-сервер.

2.2 Обоснование проектных решений по информационному обеспечению

Информационное обеспечение управления – это связь информации с системами управления предприятием и управленческим процессом в целом. Оно может рассматриваться не только в целом, охватывая все функции управления, но и по отдельным функциональным управленческим работам, например, прогнозированию и планированию, учету и анализу. Это дает

возможность оттенить специфические моменты, присущие информационному обеспечению функционального управления, раскрыв в то же самое время его общие свойства, что позволяет направить исследования вглубь.

Информационное обеспечение чаще всего определяют через ряд составляющих: совокупность справочных данных, классификаторов информации (справочно-нормативное информационное обеспечение); унифицированных систем документации; специально организованных массивов информации.

Информационное обеспечение подразделяют на внемашинное и внутримашинное (см. рисунок 2.1). К внемашинному информационному обеспечению относят: оперативную документацию, содержащую сведения о состоянии управляемого объекта и среды, нормативно-справочные документы, включающие систематизированную проектно-сметную, техническую, технологическую, организационную и производственную документацию, а также архивную информацию; систему классификации и кодирования информации; инструкции по организации ввода, хранения, внесения изменений в нормативно-справочную документацию, в том числе и в массивы данных о среде.

Внутримашинное информационное обеспечение включает в себя информационную базу на машинных носителях и систему программ ее организации, накопления, ввода и доступа к данным. Источником формирования внутримашинного информационного обеспечения служит внемашинная информационная база.

Основные требования к информационному обеспечению АСУ формулируются на основе данных предпроектного обследования строительной организации. В них обязательно должна быть отмечена необходимость обеспечения адекватности информационной базы предметной области и однозначного трактования модели. Информационная база также должна содержать данные о предметной области, достаточные для программной реализации информационного обеспечения.

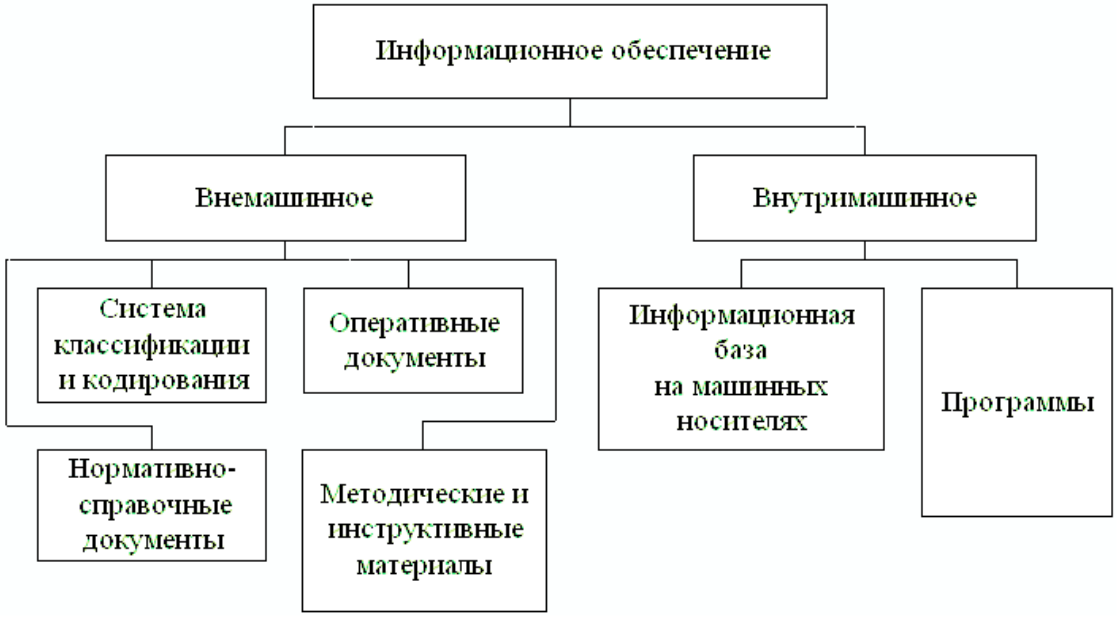


Рисунок 2.1 – Структура информационного обеспечения АСУ

Общероссийский классификатор видов экономической деятельности (сокращ. ОКВЭД) – документ, входящий в состав общероссийских классификаторов технико-экономической и социальной информации. В ООО «Вереск-М» в ОКВЭД используется код группировок 45 «Строительство». Эта группировка включает: новое строительство, реконструкцию, капитальный и текущий ремонт зданий и сооружений, включая индивидуальное строительство и ремонт по заказам населения.

Примеры внутримашинного обеспечения рассмотрены в следующем разделе.

2.3 Обоснование проектных решений по программному обеспечению

Под программным обеспечением следует понимать совокупность программ, обеспечивающих функционирование вычислительной системы (системное программное обеспечение), а также программ, предназначенных для решения конкретных задач пользователя (прикладное программное обеспечение). В таблице 2.2 рассмотрено программное обеспечение рабочего

места менеджера по работе с клиентами ООО «Вереск-М». Программное обеспечение делится на общее и программное обеспечение.

Таблица 2.2 – Общее и прикладное программное обеспечение

Общее программное обеспечение	Специальное программное обеспечение
1. Операционная система Windows 7 2. Драйвера 3. Антивирусная программа Kaspersky Small Office Security. 4. Текстовый редактор Microsoft Word 2010 5. Табличный процессор Microsoft Excel 2010 6. Коммуникационные системы: Google Chrome 7. Архиватор WinRar	1. СУБД Firebird 2.Утилита IBExpert 3. C++ Builder 6 4. IBM Rational Rose 5. AllFusion Process Modeler 7 6. Справочные правовые системы: Гарант, Консультант Плюс

Прикладное программное обеспечение предназначено для того, чтобы обеспечить применение вычислительной техники в различных сферах деятельности человека.

Система управления базой данных (СУДБ), под управлением которой будет работать база данных FireBird 1.5.0. Firebird (FirebirdSQL) – компактная, кроссплатформенная, свободная система управления базами данных, работающая на Linux, Microsoft Windows и разнообразных Unix платформах.

Firebird – это свободная система управления базами данных (СУБД). Это проект, независимый, с коммерческой точки зрения. Основана Firebird на исходном коде свободной версии СУБД Interbase 6.0, изданной компанией Borland 25 июля 2000 года. Работает Firebird на Microsoft Windows, Linux и всевозможных Unix платформах.

Многоверсионная архитектура, являющаяся несомненным преимуществом Firebird, обеспечивает параллельную обработку аналитических и оперативных запросов (дело в том, что читающие и пишущие пользователи не блокируют друг друга). Плюсами Firebird являются компактность, высокая

эффективность и мощная языковая поддержка для хранимых триггеров и процедур. Кроме того, отмечается, что Firebird легко поддерживает довольно-таки большие базы данных. Среди минусов чаще всего называют отсутствие кеша итогов запросов, полнотекстовых индексов.

Firebird поддерживает большинство международных наборов символов с огромным количеством вариантов сортировки.

Система объектно-ориентированного программирования C++ Builder предназначена для операционных систем Windows. Интегрированная среда C++ Builder обеспечивает скорость визуальной разработки, продуктивность повторно используемых компонент в сочетании с мощью языковых средств C++, усовершенствованными инструментами и разномасштабными средствами доступа к базам данных.

Профессиональные средства языка C++ интегрированы в визуальную среду разработки. C++Builder предоставляет быстродействующий компилятор с языка Borland C++, эффективный инкрементальный загрузчик и гибкие средства отладки как на уровне исходных инструкций, так и на уровне ассемблерных команд – в расчете удовлетворить высокие требования программистов-профессионалов [26].

2.4 Обоснование проектных решений по технологическому обеспечению

Техническое обеспечение представляет собой комплекс технических средств (технические средства сбора, регистрации, передачи, обработки, отображения информации, оргтехника и др.), обеспечивающих работу АИС. Центральное место среди технических средств занимает ПК. Структурными элементами технического обеспечения наряду с техническими средствами является также методические и руководящие материалы, техническая документация и обслуживающий эти технические средства персонал.

В отделе по работе с клиентами менеджер сталкивается с трудностями в

хранение и поиске необходимой информации о клиентах и их заказах, которые представлены в бумажном виде [25].

Анализ поставленной задачи приводит к выводу, что ее решение традиционными средствами учета с использованием бумажных документов возможно, но неэффективно. Для ООО «Вереск М» количество таких операций может достигать такого количества, что ручное их оформление становится просто невозможным. Использование вычислительной техники дает не только удобство и быстроту поиска информации и оформления документов, но и поднимает эффективность работы на принципиально новый уровень, предоставляя функции, ранее недоступные.

Можно выделить следующие основные преимущества использования автоматизированной информационной системы для решения задач учета:

– повышение удобства поиска и отбора данных из справочников хранения статистической информации и журналов выполненных операций;

– повышение скорости поиска и отбора информации;

– обеспечение безопасности хранения информации;

– обеспечение многопользовательской работы.

При разработке АРМ менеджера по работе с клиентами будет использована система объектно-ориентированного программирования Borland C++Builder 6. Это мощная и надежная среда быстрой разработки высокоэффективных web-служб и приложений для электронного бизнеса. Поддержка платформы BizSnap позволяет создавать web-службы и средства коммуникации согласно новейшим стандартам SOAP/XML для интеграции приложений B2B (business-to-business). Платформа разработки компонентных web-приложений WebSnap повышает скорость и эффективность создания законченных решений. Технология DataSnap объединяет бизнес-процессы предприятия и его деловых партнеров с помощью мощного ПО промежуточного уровня (middleware) для доступа к данным. [45].

Платформа BizSnap Web Services позволяет реализовать преимущества SOAP/XML-web-служб и коммуникаций для организации совместной работы

корпоративных бизнес-приложений, и соответствующих систем поставщиков и заказчиков. C++Builder предлагает всевозможные средства для работы с XML-документами и обеспечивает создание систем, которые могут напрямую взаимодействовать с приложениями деловых партнеров, развернутыми на платформах .Net и BizTalk (Microsoft) и Sun ONE (Sun Microsystems).

C++Builder включает обширный набор средств, которые повышают производительность труда программистов и сокращают продолжительность цикла разработки. Многофункциональная интегрированная среда разработки C++Builder включает компилятор, удовлетворяющий стандарта ANSI/ISO, встроенный дизайнер форм, богатый набор средств для работы с компонентами, инструмент Object Inspector, менеджер проектов и отладчик.

Объекты данных классов помогут организовать технологическое обеспечение разрабатываемого АРМ, устранить недостатки в работе менеджера посредством внедрения новых подходов и технологий [10].

2.5 Обоснование программных средств

Для разработки АРМ менеджера по работе с клиентами будут использованы следующие программные средства:

– система управления базами данных Firebird, обеспечивающая управление созданием и использованием баз данных;

– IBExpert —оболочка, предназначенная для разработки и администрирования баз данных InterBase и Firebird, а также для выбора и изменения данных, хранящихся в базах;

– C++ Builder – средство быстрой разработки приложений, позволяющее создавать приложения на языке C++, используя при этом среду разработки и библиотеку компонентов Delphi;

– AllFusion Process Modeler 7 - мощный программный продукт, с которым можно моделировать, анализировать, описывать и в последствие оптимизировать бизнес-процессы.

Описание преимуществ в использовании таких программных средств, как Firebird, IBExpert, C++ Builder было описано в разделе 2.3.

llFusion Process Modeler 7 (ранее BPwin) - инструмент для моделирования, анализа, документирования и оптимизации бизнес-процессов. AllFusion Process Modeler 7 можно использовать для графического представления бизнес-процессов. Графически представленная схема выполнения работ, обмена информацией, документооборота визуализирует модель бизнес-процесса. llFusion Process Modeler 7 (BPwin) помогает четко документировать важные аспекты любых бизнес-процессов: действия, которые необходимо предпринять, способы их осуществления и контроля, требующиеся для этого ресурсы, а также визуализировать получаемые от этих действий результаты. AllFusion Process Modeler 7 повышает бизнес-эффективность ИТ-решений, позволяя аналитикам и проектировщикам моделей соотносить корпоративные инициативы и задачи с бизнес-требованиями и процессами информационной архитектуры и проектирования приложений. Таким образом, формируется целостная картина деятельности предприятия: от потоков работ в небольших подразделениях до сложных организационных функций [36].

Дизайн должен быть реализован с помощью диаграмм потоков данных (DFD), поскольку они способны моделировать систему как единый комплекс мер, которые позволяют управлять данными в «хранилище» внутри и вне границ моделируемой системы.

Всего DFD использует четыре важных элемента: процессы, стрелки, внешние ссылки, хранилища данных. На одной диаграмме может присутствовать несколько копий одного и того же хранилища данных [35].

Итак, во втором разделе выпускной квалификационной работы был описаны проектные решения в рамках технического, информационного, программного и технологического обеспечений, были определены основные требования к аппаратному и программному обеспечению для разрабатываемого АРМ. Также было представлено обоснование программных средств разработки автоматизированного рабочего места.

3 ПРОЕКТНАЯ ЧАСТЬ

3.1 Информационное обеспечение задачи автоматизированного рабочего места

Информационное обеспечение АРМ предусматривает организацию его информационной базы, регламентирует информационные связи и предполагает состав и содержание всей системы информационного отображения.

Автоматизированное рабочее место (АРМ) можно определить, как совокупность информационно-программно-технических ресурсов, обеспечивающую конечному пользователю обработку данных и автоматизацию управленческих функций в конкретной предметной области. Создание АРМ предполагает, что основные операции по накоплению, хранению и переработке информации возлагаются на вычислительную технику, а экономист выполняет часть ручных операций и операций, требующих творческого подхода при подготовке управленческих решений.

Персональная техника применяется пользователем для контроля производственно- хозяйственной деятельности, изменения, значений отдельных параметров в ходе решения задачи, а также ввода исходных данных в АИС для решения текущих задач и анализа функций управления. Анализируя сущность АРМ, специалисты определяют их чаще всего как профессионально-ориентированные малые вычислительные системы, расположенные непосредственно на рабочих местах специалистов и предназначенные для автоматизации их работ. Для каждого объекта управления нужно предусмотреть автоматизированные рабочие места, соответствующие их функциональному назначению.

Эффективность АРМ следует рассматривать как интегральный показатель уровня реализации приведенных выше принципов, отнесенного к затратам по созданию и эксплуатации системы.

Возможности создаваемых АРМ в значительной степени зависят от

технико- эксплуатационных характеристик ПК, на которых они базируются. В связи с этим на стадии проектирования АРМ четко формулируются требования к базовым параметрам технических средств обработки и выдачи информации, набору комплектующих модулей, сетевым интерфейсам, эргономическим параметрам устройств и т.д.

Обязательным условием функционирования АРМ является техническое обеспечение. Это обоснованно выбранный комплекс технических средств для их оснащения. Средства обработки информации – вычислительные машины разных мощностей и типов – составляют основу технического обеспечения вычислительных сетей. Характерной особенностью практического использования технических средств в организационно- экономическом управлении в настоящее время является переход к децентрализованной и сетевой обработке на базе ППК. Если ППК используется в качестве АРМ небольшой локальной сети, на котором централизованно хранится вся информация, необходимая для работы, объем обрабатываемой информации невелик. Скорость работы при этом определяется не быстродействием компьютера, а скоростью диалога оператора и машины. Отсюда вытекает, что в данном случае вполне приемлема ППК с небольшим быстродействием и минимальным объемом ОЗУ. В другом случае, если компьютер предназначается для регулярной подготовки объемных документов и использует для этого большие массивы информации, необходима установка мощных машин с большим объемом внешней и внутренней памяти. Информационное наполнение АРМ при определении круга пользователей и выяснении сущности решаемых ими задач осуществляет информационное обеспечение АРМ.

В сфере организационного управления пользователи могут быть условно разделены на три категории:

- руководители;
- персонал руководителей;
- обслуживающий персонал.

Разрабатываемые АРМ для разных категорий пользователей отличаются видами представления данных. Руководителям требуются как внутренние, так и

внешние данные для реализации цели управления или принятия решения. Применение АРМ не должно нарушать привычный пользователю ритм работы. АРМ концентрируют внимание пользователя на логической структуре решаемых задач, а не на характеристике реализующей их программной системы.

Эффективными в АРМ являются многофункциональные интегрированные пакеты, реализующие несколько функций переработки информации, например, табличную, графическую, управление базами данных, текстовую обработку в рамках одной программной среды. Интегрированные пакеты удобны для пользователей. Они имеют единый интерфейс, не требуют стыковки входящих в них программных средств, обладают достаточно высокой скоростью решения задач.

Лингвистическое обеспечение АРМ включает языки общения с пользователем, языки запросов, информационно-поисковые языки, языки-посредники в сетях. Языковые средства АРМ обеспечивают однозначное смысловое соответствие действий Пользователя аппаратной части в виде ППК. Одновременно языки АРМ должны быть пользовательско-ориентированными, в том числе профессионально-ориентированными. Основу языков АРМ составляют заранее определяемые термины, описания способов установления новых терминов, списки правил, на основе которых пользователь может строить формальные конструкции, соответствующие его информационной потребности.

Организационное обеспечение АРМ включает комплекс документов, регламентирующих деятельность специалистов при использовании компьютера или терминала другого вида на рабочем месте и определяющих функции и задачи каждого специалиста.

Специалистом выполняются на АРМ следующие операции:

– ввод информации с документов при помощи клавиатуры (с визуальным контролем по экрану дисплея);

– ввод данных в ППК с магнитных носителей с других АРМ;

– прием данных в виде сообщений по каналам связи с других АРМ в условиях функционирования локальных вычислительных сетей;

– редактирование данных и манипулирование ими;

– накопление и хранение данных;

– поиск, обновление и защита данных;

– вывод на экран, печать, магнитный носитель результатной информации, а также различных справочных и инструктивных сообщений пользователю;

– формирование и передача данных на другие АРМ в виде файлов на магнитных носителях или по каналам связи в вычислительных сетях;

– получение оперативных справок по запросам.

3.1.1 Структурно-функциональный анализ программного продукта «КАК ДОЛЖНО БЫТЬ»

На рисунках 3.1-3.2 представлены диаграммы, выполненные в нотации DFD, которые описывают работу автоматизированного рабочего места менеджера по работе с клиентами для ООО «Вереск-М» «Как должно быть».

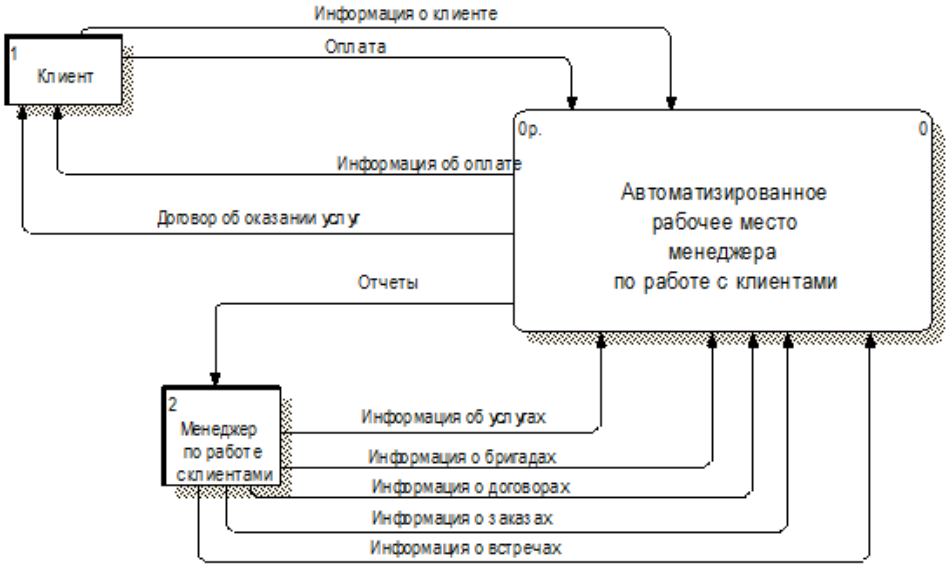


Рисунок 3.1 – Контекстная модель «Автоматизированного рабочего места менеджера по работе с клиентами»

На рисунке 3.2 показано, что для работы с клиентами менеджеру необходимо внести все предоставленные данные в различные модули, после чего они будут переданы в базу данных. При заполнении всех необходимых данных с помощью запросов можно будет формировать отчеты из БД.

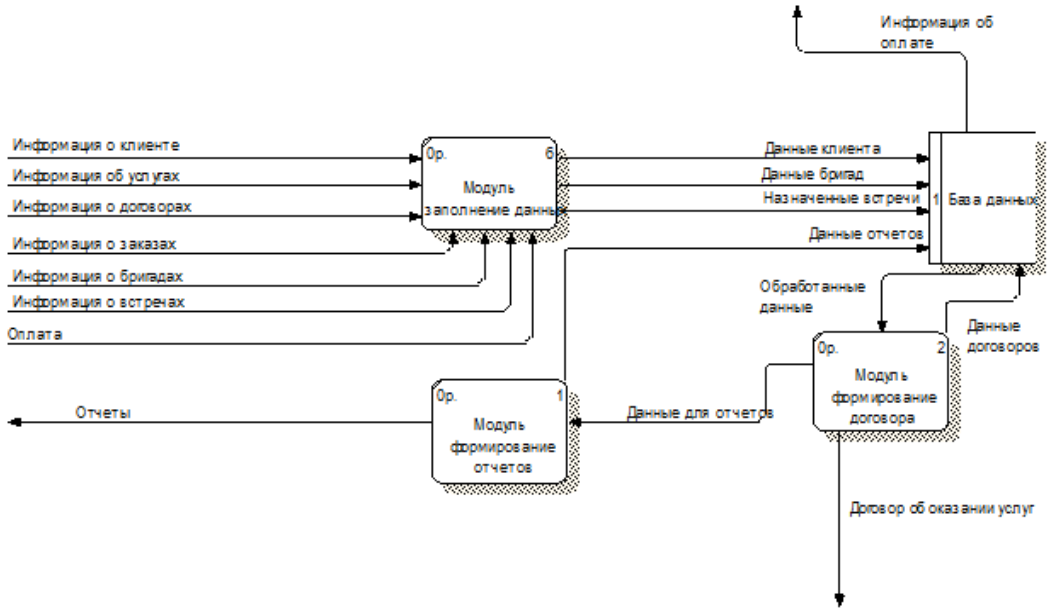


Рисунок 3.2 – Модульная организация системы

На рисунке 3.3 приведена диаграмма прецедентов для понимания того, какие действия и на основании чего выполняют клиенты, менеджер по работе с клиентами и бригады.

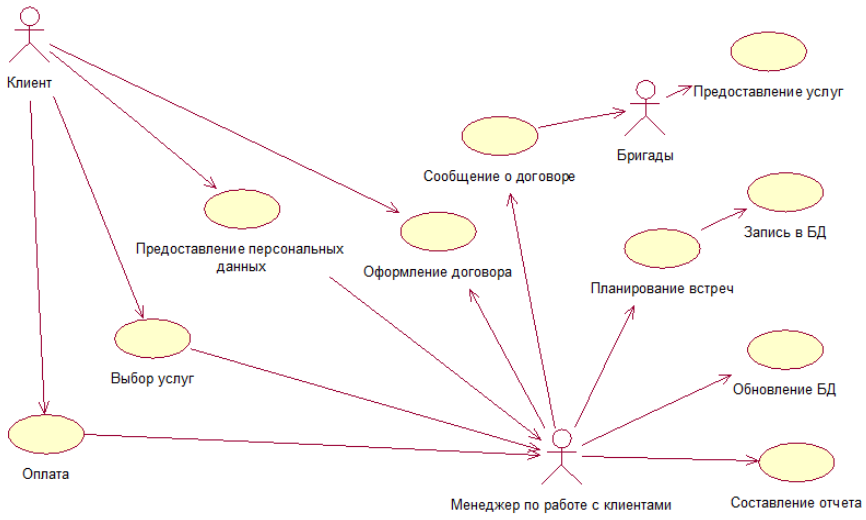


Рисунок 3.3 – Диаграмма прецедентов «Деятельность организации»

3.1.2 Информационная модель и ее описание

Информационная модель – модель объекта, представленная в виде информации, описывающей существенные для данного рассмотрения параметры и переменные величины объекта, связи между ними, входы и выходы объекта и позволяющая путём подачи на модель информации об изменениях входных величин моделировать возможные состояния объекта. Информационные модели нельзя потрогать или увидеть, они не имеют материального воплощения, потому что строятся только на информации. Информационная модель – совокупность информации, характеризующая существенные свойства и состояния объекта, процесса, явления, а также взаимосвязь с внешним миром. [14].

Менеджер по работе с ключевыми клиентами полностью планирует отношения с каждым из ключевых клиентов фирмы. Специалист этой квалификации занимается управлением проектами, координированием, стратегическим планированием, управлением взаимоотношениями, переговорами, планированием потенциала.

На этапе анализа необходимо подробное исследование как будущих функциональных возможностей разрабатываемой системы, так и информации, необходимой для их выполнения. Поэтому особое внимание было уделено как полноте информации, так и поиску противоречивой, дублирующей или неиспользуемой информации. Каждая сущность имеет неограниченное количество атрибутов, но, проанализировав требования к системе и осуществив детализацию хранилищ данных, будущую модель можно представить в виде связанных между собой отношениями сущностей. Проведя анализ предметной области путем изучения вышеперечисленной информации, были выявлены следующие внешние сущности:

- заказчик, лицо, желающее получить строительные услуги;
- подрядчик, лицо предоставляющее строительные услуги;

– поставщик материалов, лицо, обязующееся поставлять материалы на строительство.

– договор, документ, регламентирующий отношения между подрядчиком, заказчиком и поставщиком материалов.

– смета, документ, определяющий стоимость работ по договору.

Накопителями данных являются:

– информация о заказчиках;

– информация о подрядчиках;

– информация о поставщиках материалов;

– информация о договорах;

– информация о проектно-сметной документации

Информационная система разбита на четыре логические подсистемы:

– система формирования и редактирования исходных данных;

– система запросов;

– система формирование отчетов;

– система анализа данных.

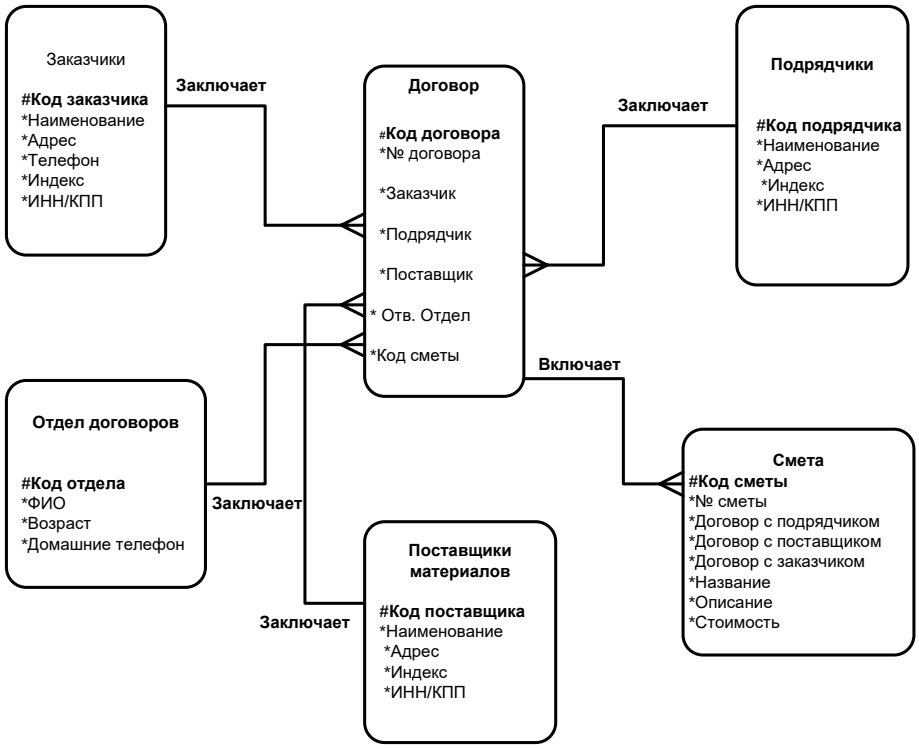


Рисунок 3.4 - Концептуальная модель

На основе проведения анализа предметной области, концептуальной модели, информационных потоков предприятия можно построить схему потоков данных, где определены основные потоки:

1,4,5 – формирование справочника заказчики;

4,2,6 – формирование справочника поставщики;

3,4,7 – формирование справочника подрядчики;

4,11,9 – формирование данных по договору с заказчиками

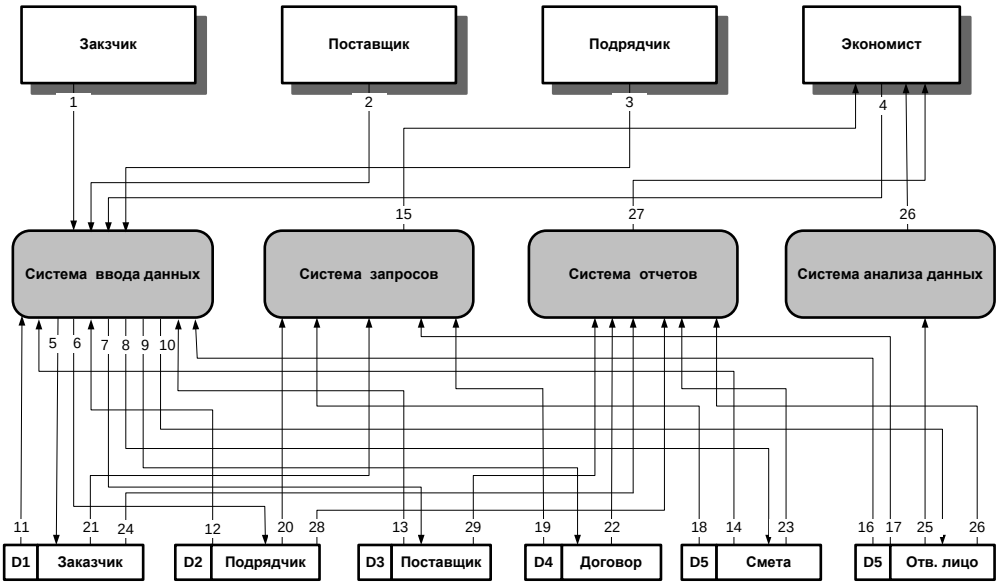


Рисунок 3.5- Схема потоков данных

12,4,9 – формирование данных по договору с подрядчиками

4,13,9 – формирование данных по договору с поставщиками

4,10 – формирование справочника ответственное лицо

4,11,12,9 – формирование учетных данных по договору с заказчиками

4,12,14,9 - формирование учетных данных по договору с подрядчиками

4,14,13,9 - формирование учетных данных по договору с поставщиками

17,15 – формирование запроса по менеджерам

18,19,20,15 – формирование запроса по суммам договоров подрядчиков

21,19,18,15 – формирование запроса по заказчикам, договорам, сметам;

22,23,24,15 – формирование отчета по заказчикам, договорам, сметем;

25,26 – аналитический отчет

27,28 - отчет по подрядчикам

29,27 - отчет по поставщикам материалов

26,27 - отчет по отелу договоров

Полученная модель данных графически представлена инфологической моделью рис 3.6.

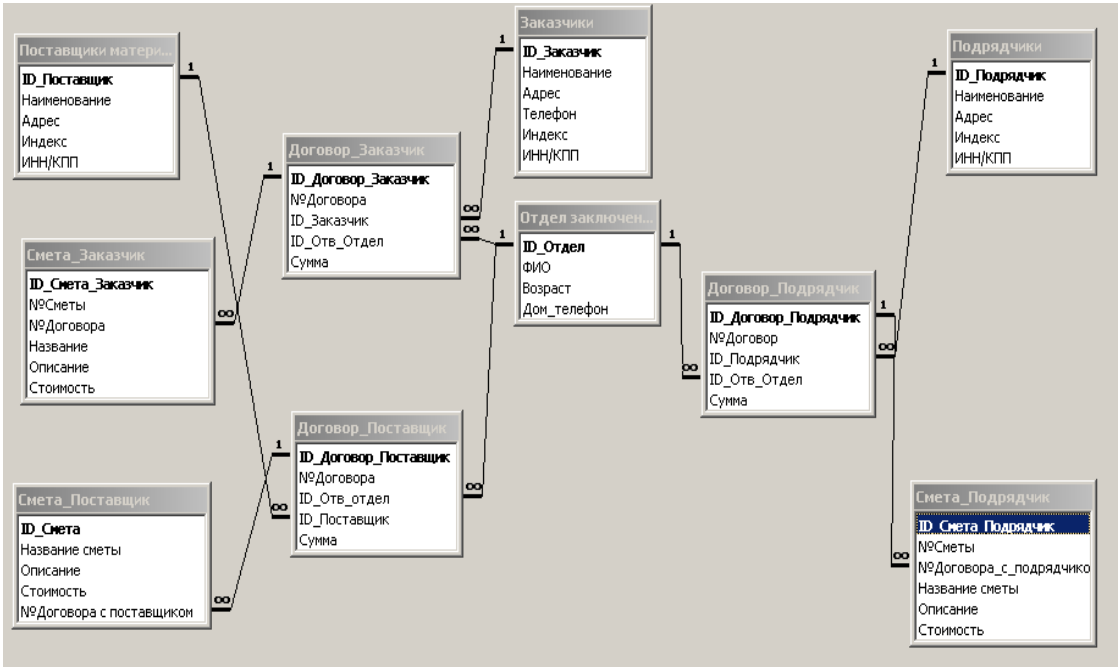


Рисунок 3.6 – Инфологическая модель данных

Датологическая модель (см. рисунок 3.6) строится на основе разработанной инфологической модели и наиболее приемлемой для дальнейшей разработки является реляционная модель данных, где вся информация хранится в виде связанных таблиц, что дает возможность более эффективно хранить информацию, сохраняя целостность данных. Между таблицами устанавливаются отношения «один-ко-многим». Это самый распространенный вид отношений в реляционной базе данных.

На рисунке 3.7 показана модель взаимодействия менеджера по работе с клиентами с формой автоматизированного рабочего места, которая непосредственно связана с базой данных.

Рисунок 3.7 – Модель взаимодействия менеджера с формой АРМ

Информационные модели отражают различные типы систем объектов, в которых реализуются различные структуры взаимодействия и взаимосвязи между элементами системы. Для отражения систем с различными структурами используются различные типы информационных моделей: табличные, иерархические и сетевые. Одним из наиболее часто используемых типов информационных моделей является прямоугольная таблица, которая состоит из столбцов и строк.

3.1.3 Характеристика базы данных

Рассмотрим характеристику инфологической модели базы данных.

Процесс, в ходе которого решается, какой вид будет у вновь создаваемой базы данных, называется проектированием базы данных (database design). Работа по проектированию базы данных включает выбор:

- таблиц, которые будут входить в базу данных;
- столбцов, принадлежащих каждой таблице;
- взаимосвязей между таблицами и столбцами.

Часто при обсуждении вопросов проектирования реляционных баз данных почти все внимание уделяется применению правил нормализации. В ходе нормализации обеспечивается защита целостности данных путем устранения дублирования данных. В результате таблица, которая первоначально казалась «имеющей смысл», разбивается на две или более

связанных таблиц, которые могут быть «собраны вместе» с помощью операции объединения. Этот процесс называется декомпозицией без потерь (non-loss decomposition) и просто означает разделение таблицы на несколько меньших таблиц без потери информации. Нормализация наиболее полезна для проверки созданной вами структуры. Можно проанализировать свои решения о том, какие столбцы должны быть включены в ту или иную таблицу с точки зрения правил нормализации, убедившись при этом, что не сделали каких-то фатальных ошибок. Понимание основ процесса нормализации также может помочь в процессе проектирования базы данных, но оно не является универсальным рецептом при построении базы с нуля. Итак, как определить, какие столбцы должны располагаться в начале таблицы. Общего правила на этот счет не существует. Однако здесь вам может оказать существенную помощь моделирование зависимостей — анализ сущности данных (в терминах объектов или вещей) и зависимостей между ними (один-к-одному, один-ко-многим, многие-ко-многим).

На практике проектирование базы данных требует хорошего понимания моделируемой предметной области, а также знаний в области моделирования зависимостей и нормализации. Проектирование базы данных обычно является итеративным процессом, в ходе которого шаг за шагом достигается требуемый результат, а иногда и пересматривается несколько шагов, переделывая предыдущую работу с учетом появившихся новых потребностей [31].

Для разработки, была реализована база данных, логическая модель которой представлена на рисунке 3.8.

Главной таблицей является «Клиент». К данной таблице привязаны дополнительные таблицы, с помощью связей.

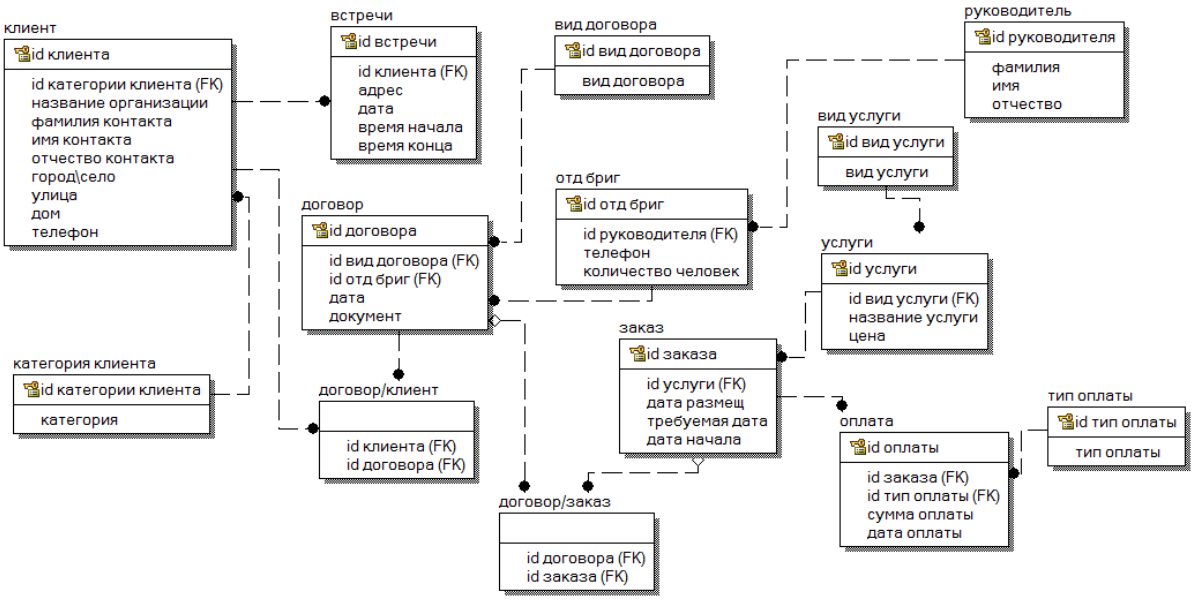


Рисунок 3.8 – Логическая модель базы данных

Основная цель инфологического моделирования - обеспечение для человека наиболее естественных способов представления и сбора информации, хранящейся в создаваемой базе данных.

В связи с этим, инфологическую модель данных строят по аналогии с естественным языком. Основными элементами инфологических моделей являются сущности, связи между ними и их свойства (атрибуты).

Описание, создаваемое по инфологической модели данных, называют даталогической моделью данных. Даталогическая модель отражает логические связи между элементами данных вне зависимости от их содержания и среды хранения. Пользователям выделяются подмножества этой логической модели, называемые внешними моделями, отражающие их представления о предметной области. Внешняя модель соответствует представлениям, которые пользователи получают на основе логической модели, в то время как концептуальные требования отражают представления, которые пользователи первоначально желали иметь и которые легли в основу разработки инфологической модели. Даталогическая модель отображается в физическую память, такую, как диск, лента или какой-либо другой носитель информации. Даталогическая модель в основном используется прикладными программистами для реализации

требований, которые выдвинули конечные пользователи, отражённых в инфологической концептуальной модели.

Конечным результатом даталогического проектирования является описание структуры БД на языке описания данных конкретных СУБД.

Для физического проектирования была реализована даталогическая модель базы данных, приведенная на рисунке 3.9.

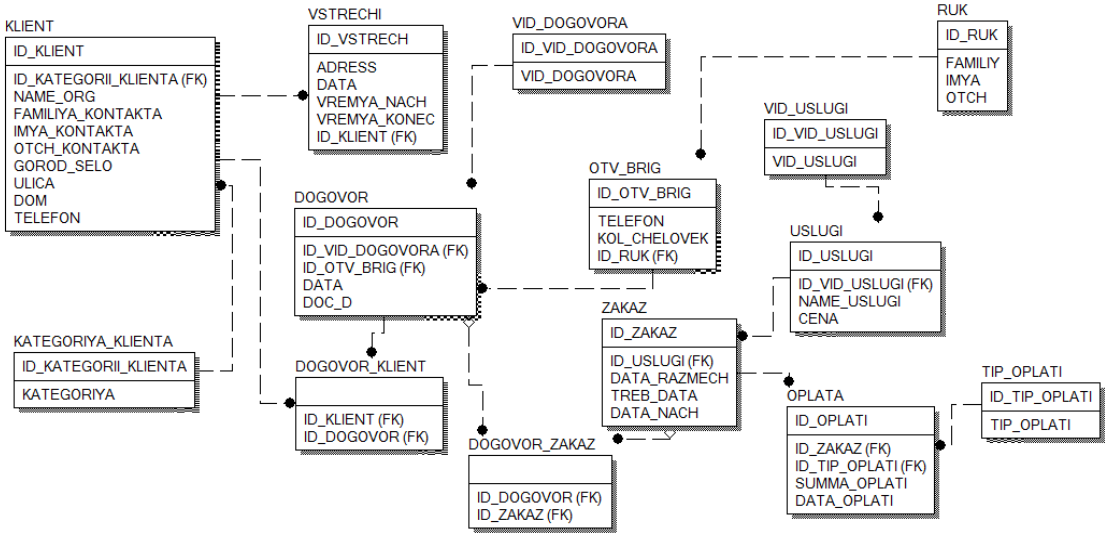


Рисунок 3.9 – Даталогическая модель базы данных

Даталогическая модель еще называется концептуальной моделью БД. В соответствии с выбранной СУБД выполняется структурирование данных. На данном этапе проектировщик создает структуру данных и организует связь между объектами.

3.1.4 Разработка базы данных

Для разработки хранилища данных в данной выпускной квалификационной работе используется СУБД FireBird. Для создания и управления базой данных используется утилита IBExpert [13]. Данная утилита очень удобна в использовании и позволяет выполнять любые манипуляции с базой данных. Форма данной утилиты представлена на рисунке 3.10.

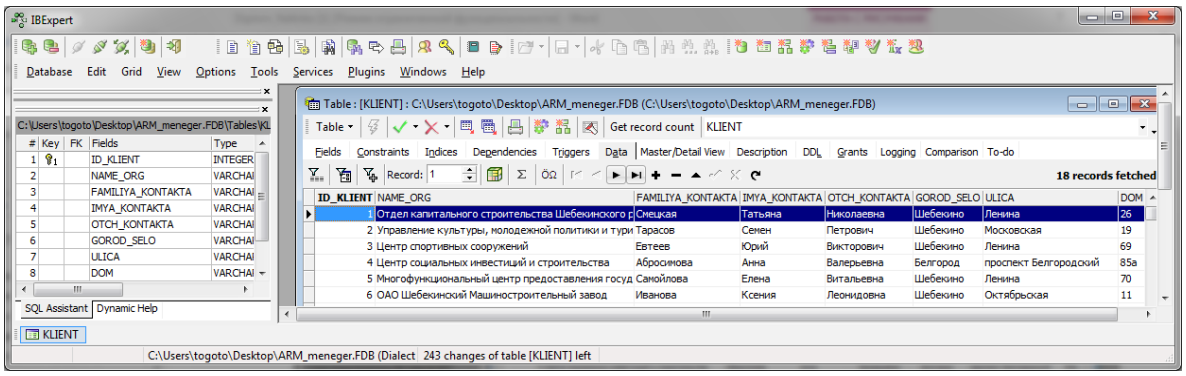


Рисунок 3.10 – Форма утилиты IBExpert

Разработка базы данных начинается с создания таблиц. В рассматриваемой утилите есть удобный мастер создания таблиц, который в последующем генерирует sql код таблицы [7]. Например, для создания таблицы клиентов, мастер сгенерировал следующий код:

```
create table klient (
id_klient integer not null,
name_org varchar(90),
familiya_kontakta varchar(30),
imya_kontakta varchar(30),
otch_kontakta varchar(30),
gorod_selo varchar(30),
ulica varchar(30),
dom varchar(10),
telefon varchar(30),
id_kategorii_klienta  integer
);
```

Важной частью базы данных являются хранимые процедуры. Для каждой таблицы реализована хранимая процедура на добавление. Рассмотрим реализацию на примере добавления новой записи в таблицу «Клиент». Код процедуры для добавления представлен ниже:

```
CREATE OR ALTER PROCEDURE ADD_KLIENT (
id_klient integer not null,
id_kategorii_klienta integer not null,
name_org varchar(90),
familiya_kontakta varchar(30) not null,
imya_kontakta varchar(30) not null,
otch_kontakta varchar(30) not null,
gorod_selo varchar(30) not null,
```

```
ulica varchar(30) not null,
dom varchar(10) not null,
telefon varchar(30) not null)
as
begin
insert into klient (klient.id_klient, klient.id_kategorii_klienta,
klient.name_org, klient.familiya_kontakta,
klient.imya_kontakta, klient.otch_kontakta, klient.gorod_selo, klient.ulica,
klient.dom, klient.telefon)
values (:id_klient, :id_kategorii_klienta, :name_org, :familiya_kontakta,
:imya_kontakta, :otch_kontakta,
:gorod_selo, :ulica, :dom, :telefon );
end
```

Входными параметрами рассматриваемой процедуры являются значения полей новой записи таблицы. Подобные процедуры реализованы для всех таблиц.

Ниже приведен код для изменения и удаления данных в таблице.

```
CREATE OR ALTER PROCEDURE UPDATE_KLIENT (
id_klient integer not null,
id_kategorii_klienta integer not null,
name_org varchar(90),
familiya_kontakta varchar(30) not null,
imya_kontakta varchar(30) not null,
otch_kontakta varchar(30) not null,
gorod_selo varchar(30) not null,
ulica varchar(30) not null,
dom varchar(10) not null,
telefon varchar(30) not null)
as
begin
update KLIENT set

klient.id_kategorii_klienta=:id_kategorii_klienta,
klient.name_org=:name_org,
klient.familiya_kontakta=:familiya_kontakta,
klient.imya_kontakta=:imya_kontakta,
klient.otch_kontakta=:otch_kontakta,
klient.gorod_selo=:gorod_selo,
klient.ulica=:ulica,
klient.dom=:dom,
klient.telefon=:telefon
```

```
where klient.id_klient=:id_klient;
suspend;
end

CREATE OR ALTER PROCEDURE DEL_KLIENT (
id_klient integer not null)
as
begin
delete from KLIENT
where klient.id_klient=:id_klient;
end
```

Просмотры (представления) позволяют возвращать наборы данных, удовлетворяющие нужды конкретных пользователей или групп. После создания просмотра с ним можно обращаться точно так же, как и с обычной таблицей.

Для того, чтобы в дальнейшем создавать отчеты, необходимо было создать представления, связывая между собой несколько таблиц в базе. На рисунке 3.11 приведен результат создания представления «Информация о клиенте» из таблиц «Клиенты» и «Категория клиента». Ниже приведен код, для создания данного представления.

```
create view klient_info(
id_klient,
kategoriya,
name_org,
familiya_kontakta,
imya_kontakta,
otch_kontakta,
gorod_selo,
ulica,
dom,
telefon
)
as
select id_klient, kategoriya, name_org, familiya_kontakta, imya_kontakta,
otch_kontakta, GOROD_SELO, ULICA, DOM, TELEFON
from klient, kategoriya_klienta
where klient.id_kategorii_klienta=kategoriya_klienta.id_kategorii_klienta
;
```

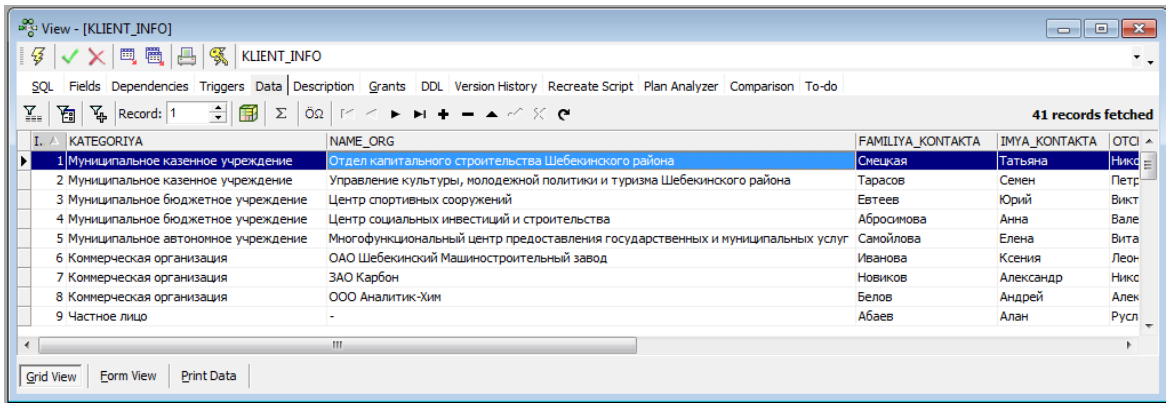


Рисунок 3.11 – Представление «Информация о клиенте»

Просмотр может строиться на базе одной или нескольких таблиц, или даже на основе других просмотров.

3.2 Программное обеспечение автоматизированного рабочего места

3.2.1 Общие положения

Для разрабатываемого автоматизированного рабочего места существуют функции управления и обработки данных. При этом можно выделить и детализировать два подмножества функций: реализующих служебные функции и реализующих основные функции (ввода первичной информации, обработки, ведения справочников, ответов на запросы и др.).

При наличии различных вариантов ответа на ввод функции пользователем в последующих шагах происходит детализация, или уточнение действий, например, какая информация должна вводится, в каком виде или на какое устройство желательно осуществить вывод и т.д. [5]. На рисунке 3.12 показано дерево функций автоматизированного рабочего места менеджера по работе с клиентами в строительной организации ООО «Вереск-М».

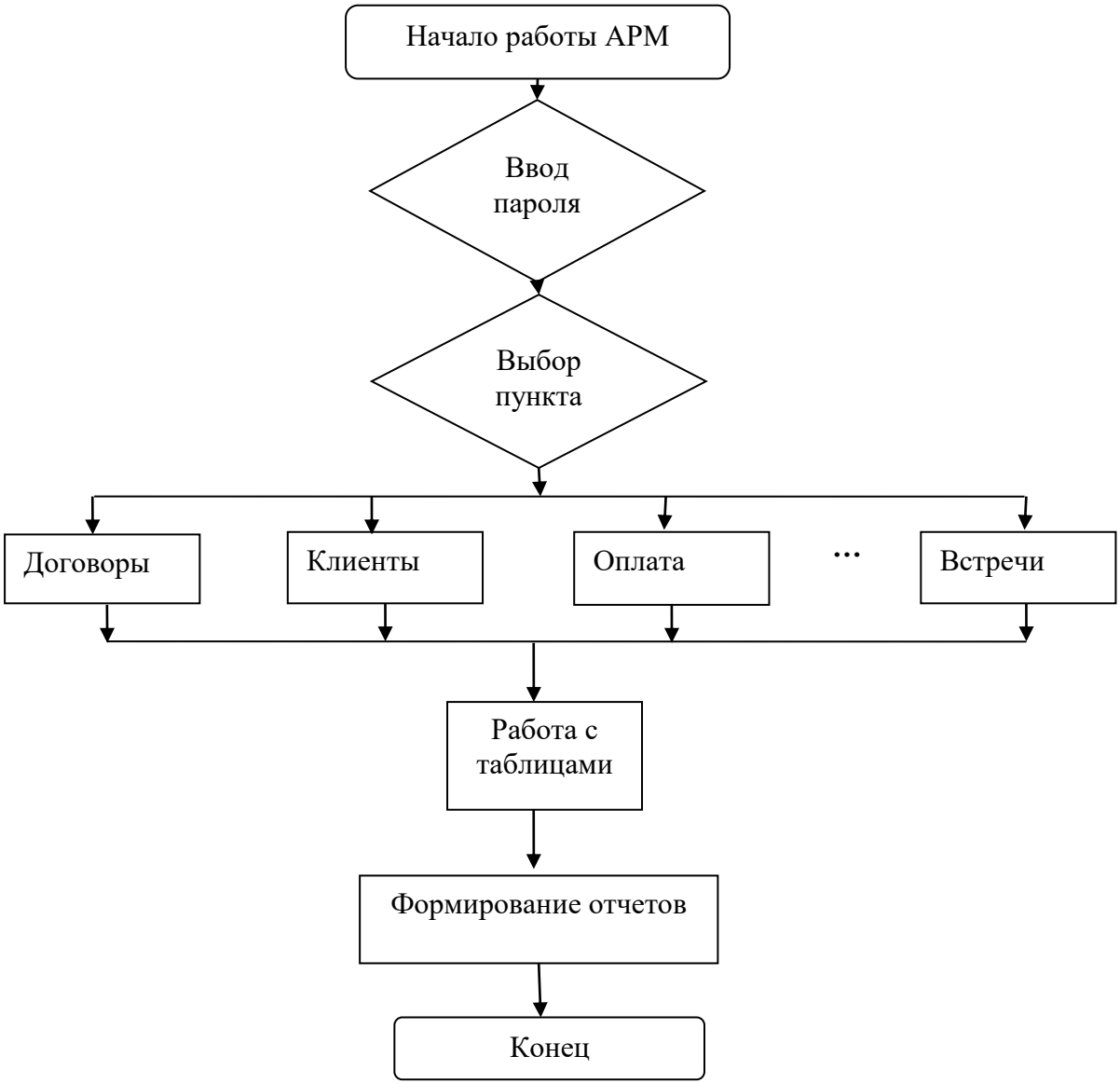


Рисунок 3.12 – Дерево функций

В любом автоматизированном рабочем месте должна быть предусмотрена возможность обмена сообщениями между пользователем и ПК. Такая функция реализуется в виде сценария диалога, поскольку именно диалог является наиболее привычной формой общения для человека. На рисунке 3.13 представлен сценарий диалога автоматизированного рабочего места менеджера по работе с клиентами ООО «Вереск-М» в режиме «Справочники», который означает выбор определенной таблицы системы.

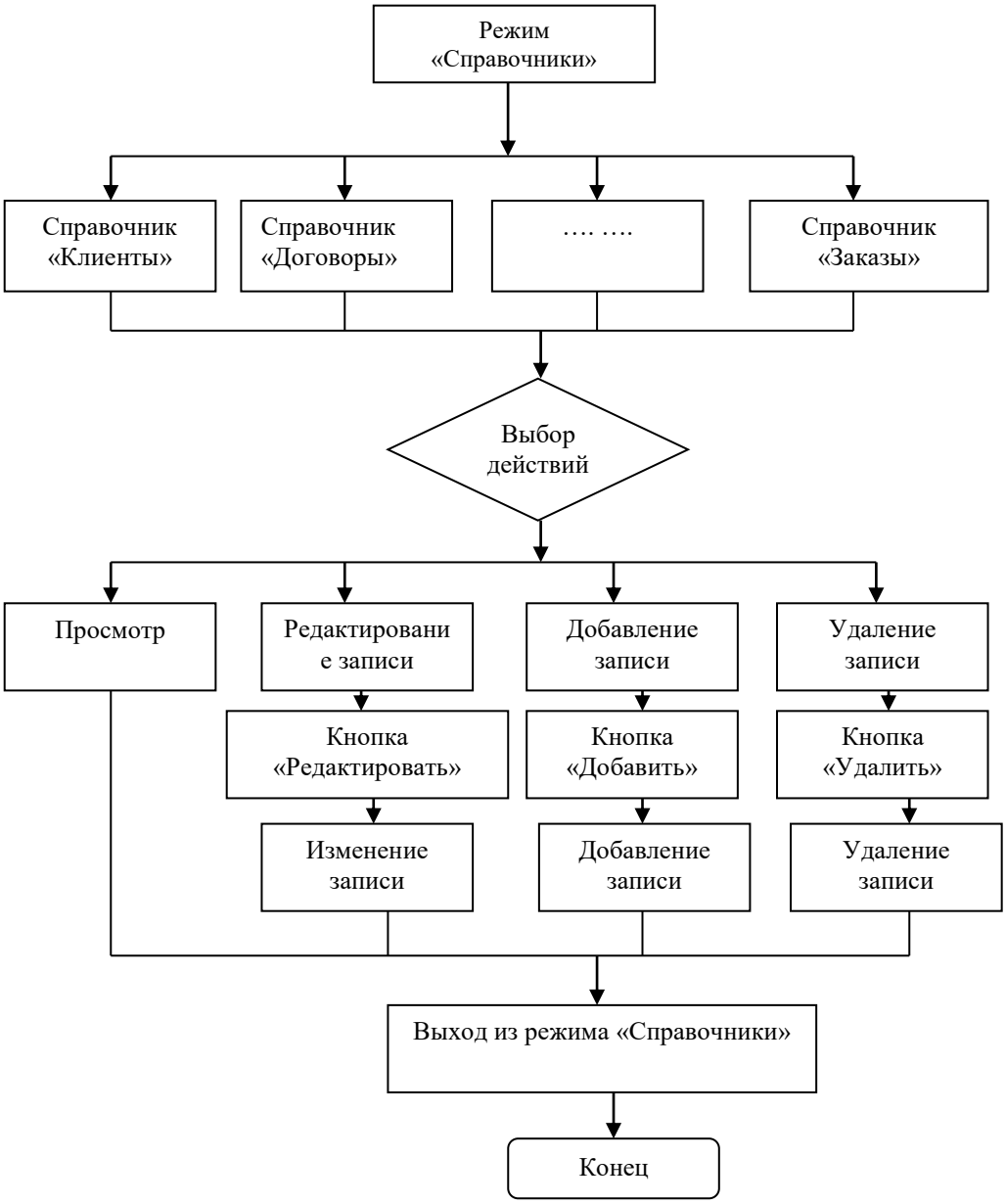


Рисунок 3.13 – Сценарий диалога «Справочники»

При выборе определенного типа справочника специалисту предоставляются набор манипулирования данными. Сценарий диалога позволяет описать процесс взаимодействия пользователя с приложением на уровне решаемой им прикладной задачи [11]. Однако для программной реализации интерфейса такое описание носит слишком общий характер. На этапе реализации необходимо перейти на уровень описания соответствующих процессов с помощью используемых инструментальных средств разработки приложения.

3.2.2 Описание программных модулей

После того, как разработана база данных, необходимо реализовать клиентское приложение, которое позволит управлять и работать с данными. Для разработки приложения будет использован программный инструмент C++Builder, который содержит все необходимое для разработки [26]. На рисунке 3.14 представлен общий вид проекта.

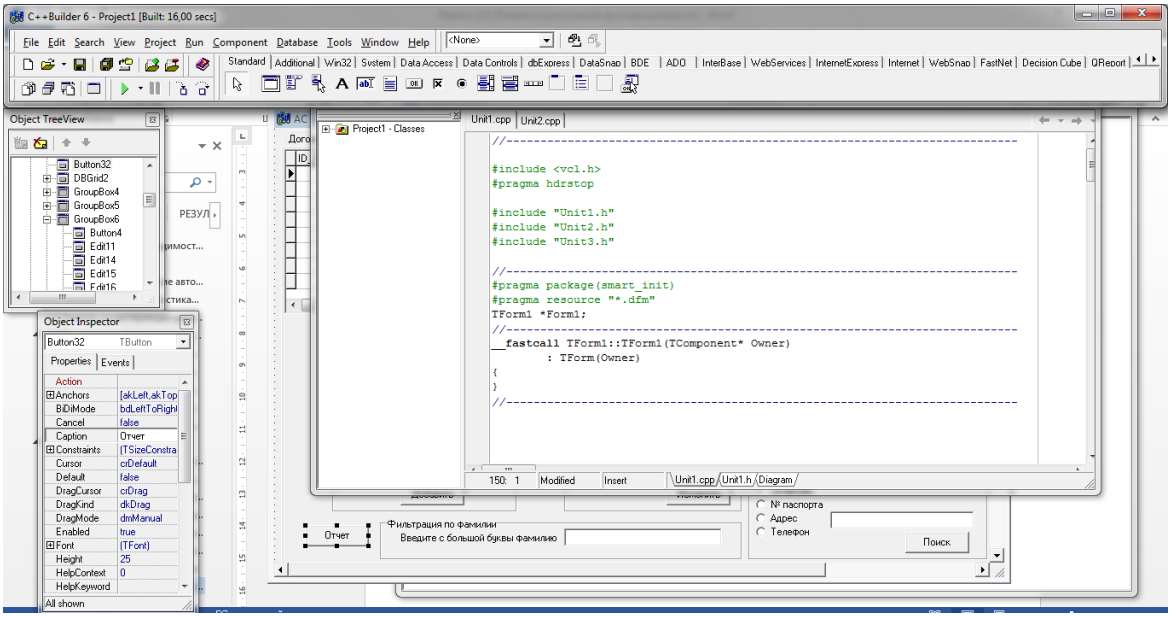


Рисунок 3.14 – Проект по созданию приложения

Компоненты InterBase eXpress (IBX) предназначены для работы с сервером Firebird или InterBase, используя InterBase API. Используя данные компоненты, можно получать данные, вносить в них изменения, управлять транзакциями, получать сведения о базе данных, отслеживать состояние процессов выполнения запросов [18].

Начальным этапом при создании приложения является создание нового проекта. Затем необходимо создать DataModule (по умолчанию DataModule2), на котором будем размещать все невидимые компоненты для доступа и работы с БД, они показаны на рисунке 3.15.

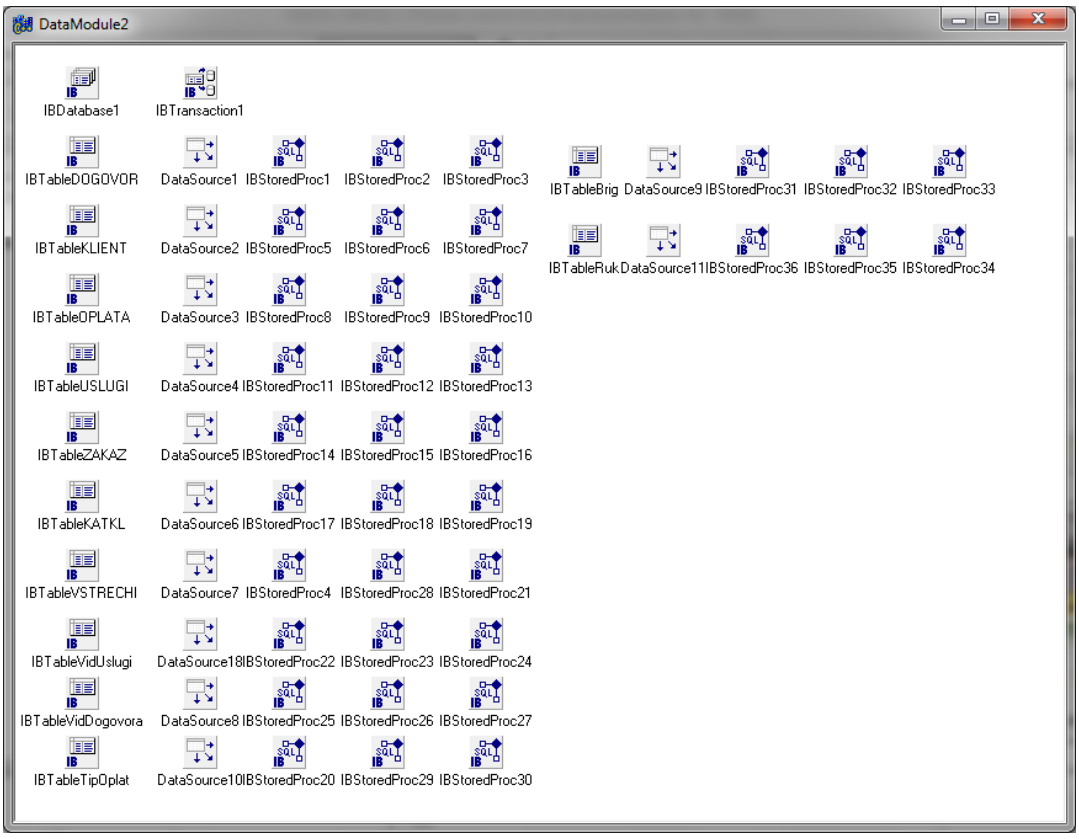


Рисунок 3.15 – Невизуальные компоненты разработанного приложения

Необходимо настроить компонент IBDatabase1, как показано на рисунке 3.16, для соединения с БД.

Каждая организация должна следить за конфиденциальностью данных, предоставляемых клиентами. Поэтому необходимо создать форму авторизации, доступ к которой будет иметь только менеджер по работе с клиентами [19]. На рисунке 3.17 показана разработанная форма авторизации.

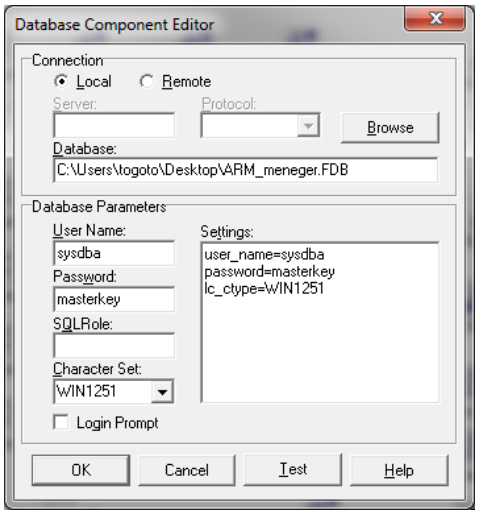


Рисунок 3.16 – Настройка компонента IBDataBase1

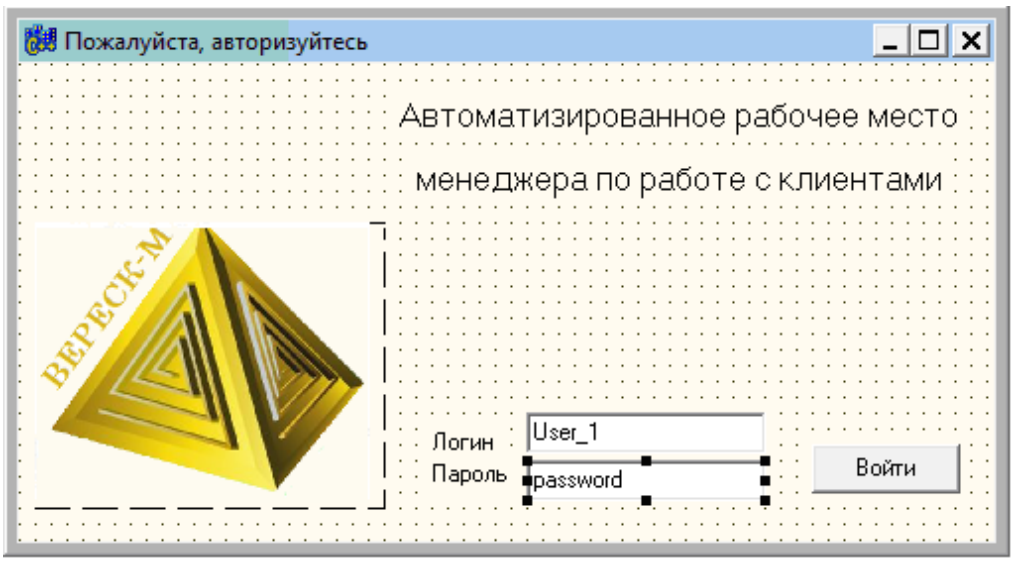


Рисунок 3.17 – Окно авторизации

Ниже приведена часть кода, с помощью которого производится авторизация.

```
void __fastcall TMyAuthForm::Button1Click(TObject *Sender)
{
std::string name = std::string(this->Edit1->Text.c_str());
std::string password = std::string(this->Edit2->Text.c_str());
this->user = User::getUserByName(name);
if (this->user == NULL) {
ShowMessage("Неверные данные");
return;
}
if (!this->user->verifyPassword(this->Edit2->Text.c_str())) {
ShowMessage("Неверные данные");
return;
}
this->result = this->RESULT_OK;
this->Close();
}
```

Клиентское приложение состоит из нескольких форм, первой является форма авторизации, затем форма основного приложения и формы с отчетами. На основной форме приложения расположено несколько вкладок, поэтому целесообразно описать одну из них (остальные были созданы аналогичным образом и приведены в Приложении Б).

Вкладка приложения «Клиенты», которая представлена на рисунке 3.18, имеет стандартный интерфейс, на котором в верхней части расположен объект отображения хранимых данных о клиентах, а в нижней части находятся элементы управления.

Элементы управления представляют собой добавление, изменение и удаление записей таблицы [14]. Также можно осуществлять сортировку, фильтрацию и поиск данных о клиентах по выбранным параметрам.

На данной форме расположена таблица с данными о категориях клиентов, чтобы менеджеру было удобно работать с данными.

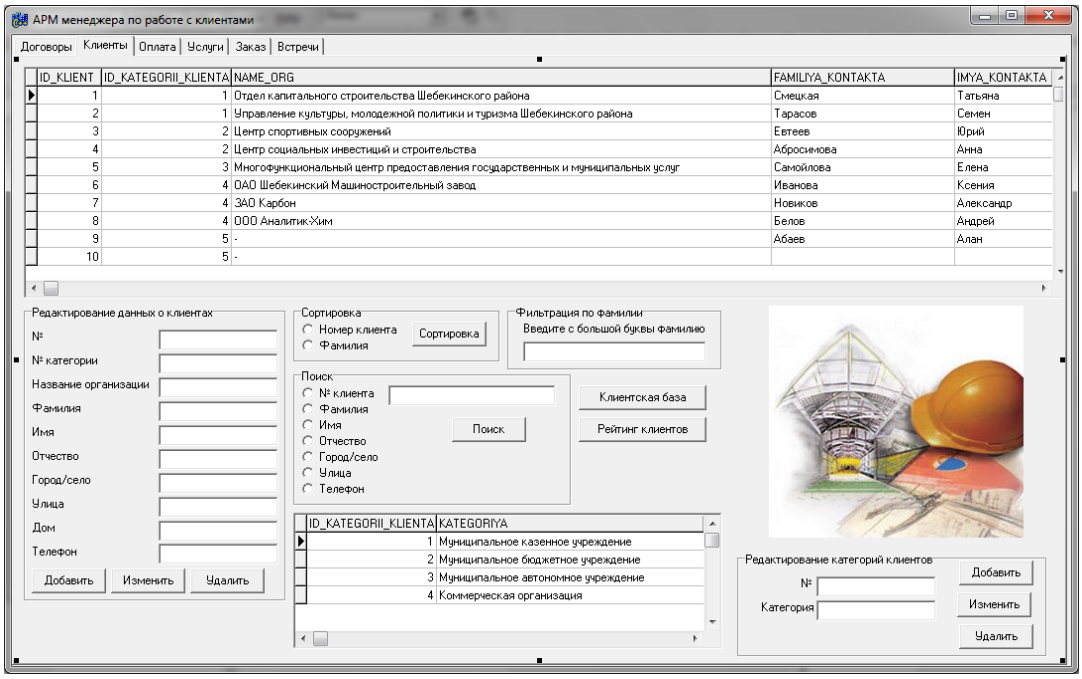


Рисунок 3.18 – Форма системы «Клиенты»

Для того, чтобы осуществлять процедуры для добавления, изменения и удаления записей таблицы необходимо использовать программный код, приведенный ниже. Полный код всех процедур приведен в Приложении В.

```
void __fastcall TForm1::Button4Click(TObject *Sender)
{
DataModule2->IBTableKLIENT->Active=false;
DataModule2->IBStoredProc5->ParamByName("ID_KLIENT")->AsString=Form1->Edit14-
>Text;
DataModule2->IBStoredProc5->ParamByName("ID_KATEGORII_KLIENTA")->AsString=Form1-
```

```
>Edit45->Text;
DataModule2->IBStoredProc5->ParamByName("NAME_ORG")->AsString=Form1->Edit44-
>Text;
DataModule2->IBStoredProc5->ParamByName("FAMILIYA_KONTAKTA")->AsString=Form1-
>Edit15->Text;
DataModule2->IBStoredProc5->ParamByName("IMYA_KONTAKTA")->AsString=Form1-
>Edit16->Text;
DataModule2->IBStoredProc5->ParamByName("OTCH_KONTAKTA")->AsString=Form1-
>Edit49->Text;
DataModule2->IBStoredProc5->ParamByName("GOROD_SELO")->AsString=Form1->Edit50-
>Text;
DataModule2->IBStoredProc5->ParamByName("ULICA")->AsString=Form1->Edit50->Text;
DataModule2->IBStoredProc5->ParamByName("DOM")->AsString=Form1->Edit8->Text;
DataModule2->IBStoredProc5->ParamByName("TELEFON")->AsString=Form1->Edit65-
>Text;
DataModule2->IBStoredProc5->Prepare();
DataModule2->IBStoredProc5->ExecProc();
DataModule2->IBTableKLIENT->Active=true;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button5Click(TObject *Sender)
{
DataModule2->IBTableKLIENT->Active=false;
DataModule2->IBStoredProc6->ParamByName("ID_KLIENT")->AsString=Form1->Edit14-
>Text;
DataModule2->IBStoredProc6->ParamByName("ID_KATEGORII_KLIENTA")->AsString=Form1-
>Edit45->Text;
DataModule2->IBStoredProc6->ParamByName("NAME_ORG")->AsString=Form1->Edit44-
>Text;
DataModule2->IBStoredProc6->ParamByName("FAMILIYA_KONTAKTA")->AsString=Form1-
>Edit15->Text;
DataModule2->IBStoredProc6->ParamByName("IMYA_KONTAKTA")->AsString=Form1-
>Edit16->Text;
DataModule2->IBStoredProc6->ParamByName("OTCH_KONTAKTA")->AsString=Form1-
>Edit49->Text;
DataModule2->IBStoredProc6->ParamByName("GOROD_SELO")->AsString=Form1->Edit50-
>Text;
DataModule2->IBStoredProc6->ParamByName("ULICA")->AsString=Form1->Edit50->Text;
DataModule2->IBStoredProc6->ParamByName("DOM")->AsString=Form1->Edit8->Text;
DataModule2->IBStoredProc6->ParamByName("TELEFON")->AsString=Form1->Edit65-
>Text;
DataModule2->IBStoredProc6->Prepare();
DataModule2->IBStoredProc6->ExecProc();
```

```
DataModule2->IBTableKLIENT->Active=true;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button6Click(TObject *Sender)
{
DataModule2->IBTableKLIENT->Active=false;
DataModule2->IBStoredProc7->ParamByName("ID_KLIENT")->AsString=Form1->Edit14-
>Text;
DataModule2->IBStoredProc7->Prepare();
DataModule2->IBStoredProc7->ExecProc();
DataModule2->IBTableKLIENT->Active=true;
}
//---------------------------------------------------------------------------
```

Для перехода на новую форму, на которой будет расположен один из отчетов о клиентах, для кнопки «Клиентская база» необходимо написать следующий код:

```
Form3->Visible=false;
Form3->QuickRep1->Preview();
Form3->Visible=true;
```

На новой форме размещаются все необходимые компоненты, как показано на рисунке 3.19.

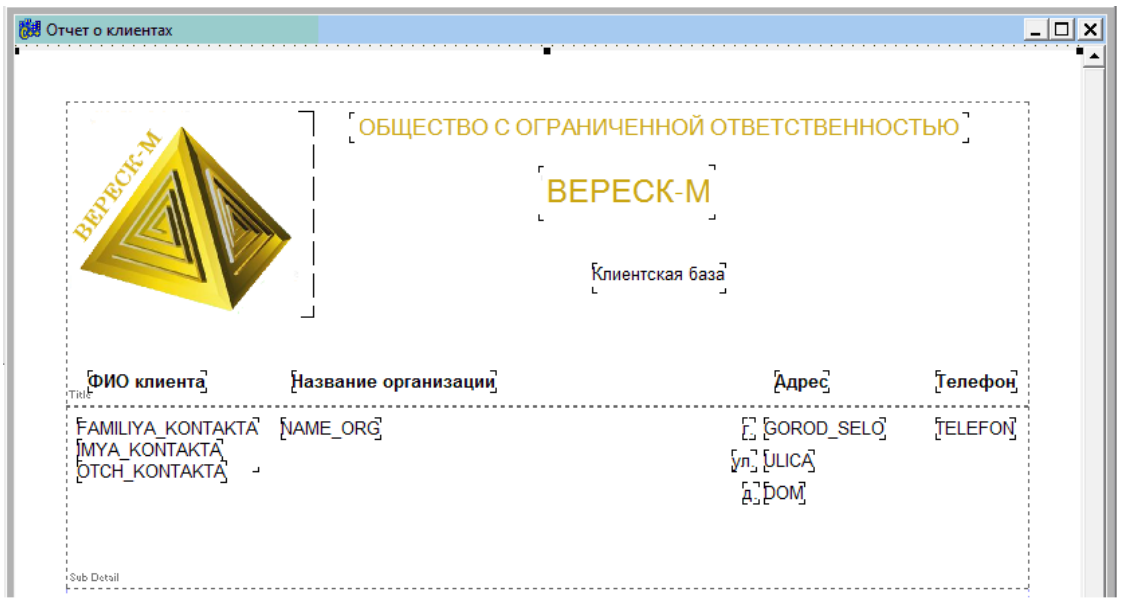
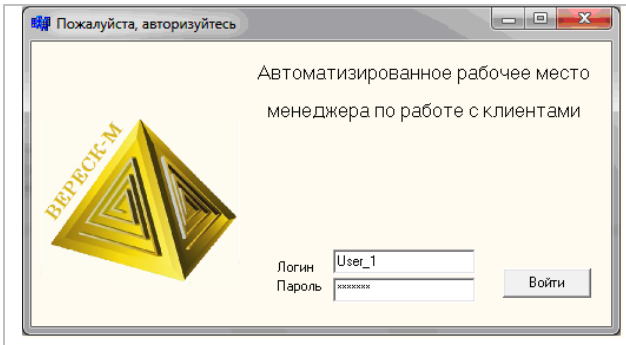


Рисунок 3.19 – Форма отчета «Отчет о клиентах»

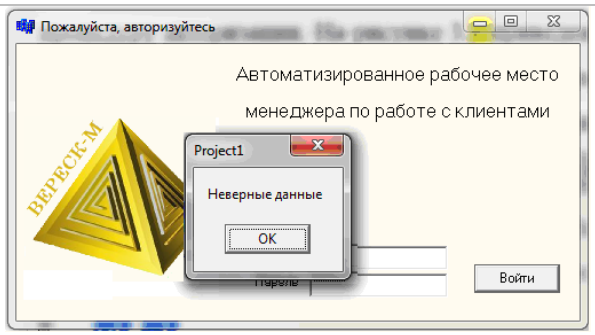
Отчёты позволяют менеджерам оперативно получать наглядную информацию по клиентам.

3.3 Описание контрольного примера реализации проекта

При запуске разработанной системы менеджеру необходимо пройти процедуру авторизации. На рисунке 3.17 приведена форма авторизации. Если менеджер ввел все данные правильно, то открывается основное приложение (см. рисунок 3.20а), иначе выводится сообщение об ошибке (см. рисунок 3.20б).



а – ввод данных



б – сообщение об ошибке

Рисунок 3.20 – Форма авторизации

После успешной авторизации менеджеру предлагается перейти на одну из вкладок, для заполнения данных. На примере вкладки «Клиенты» рассмотрим результат разработанного приложения [45]. Здесь существует возможность редактирования записей из базы данных о клиентах и их категориях. Работа с данной формой представлена на рисунке 3.21.

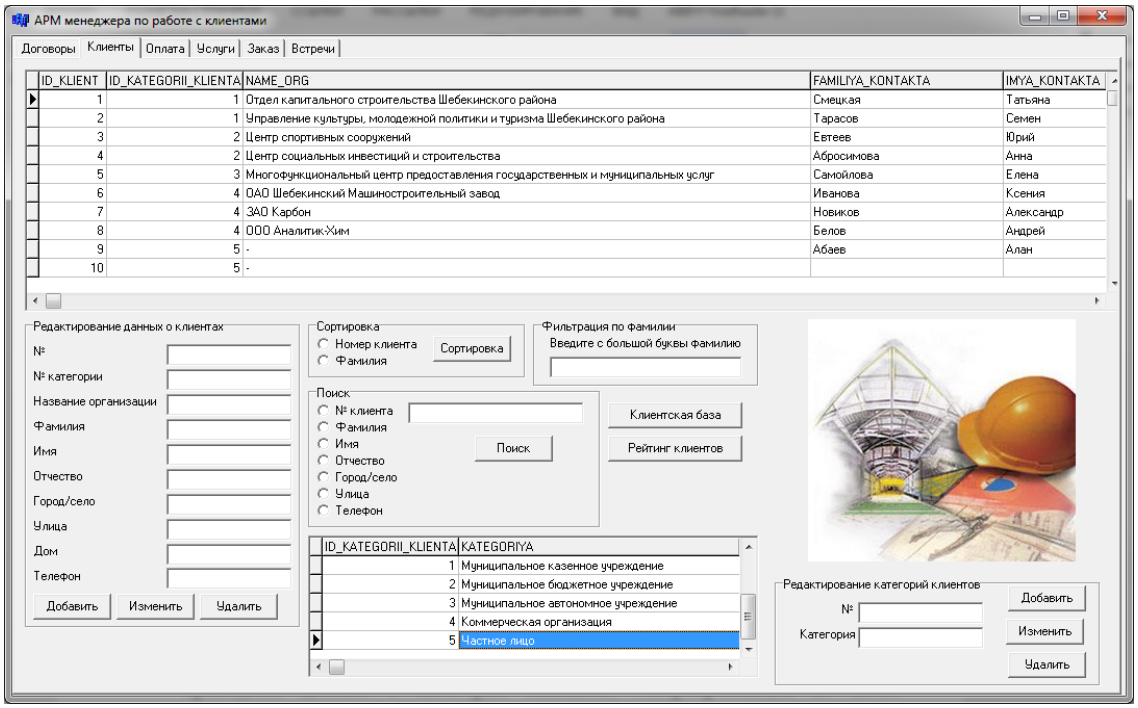


Рисунок 3.21 – Форма системы «Клиенты»

Каждый менеджер ежедневно сталкивается с обработкой большого объема данных, поэтому необходимо, чтобы автоматизированное рабочее место имело функции сортировки и фильтрации данных. По умолчанию сортировка данных выполняется по номеру клиента. На рисунке 3.22 приведены данные о клиентов, отсортированные по полю «Фамилия».

Помимо этого, менеджеру приходится осуществлять поиск информации в базе данных [26]. На рисунке 3.23 результаты поиска данных о клиенте, поиск производился по полю «Фамилия» и параметру «Б» (поиск осуществляется после нажатия кнопки «Поиск»).

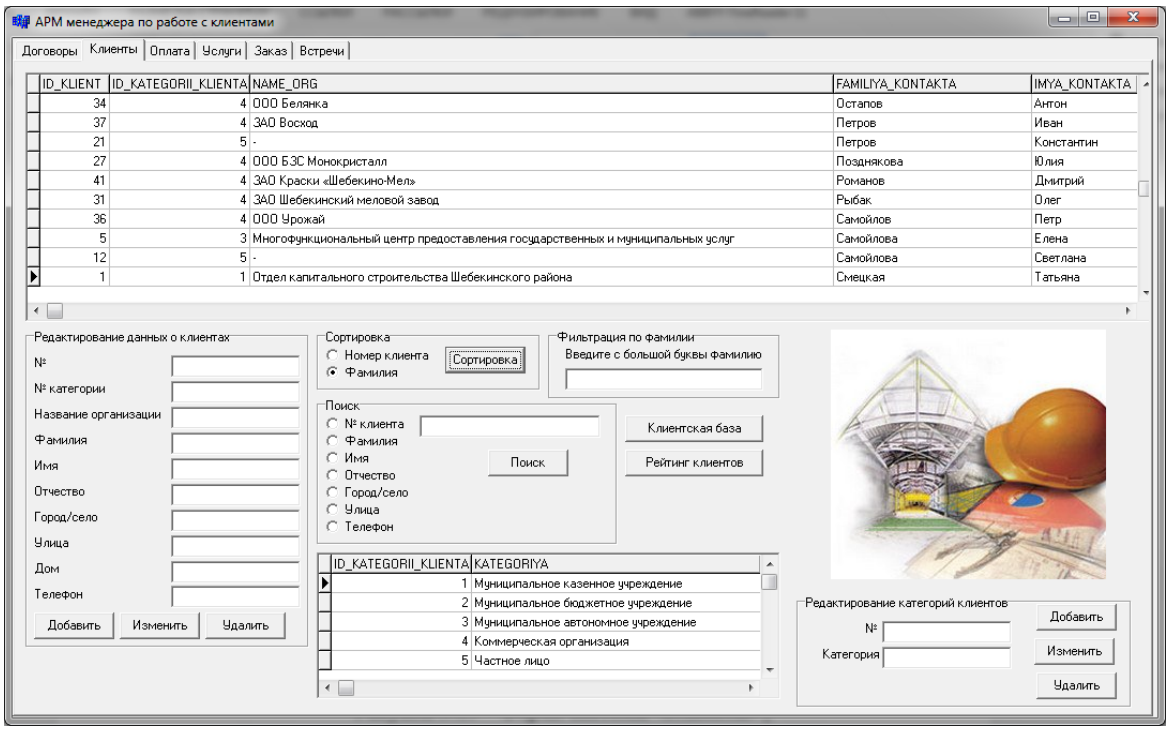


Рисунок 3.22 – Результат сортировки данных

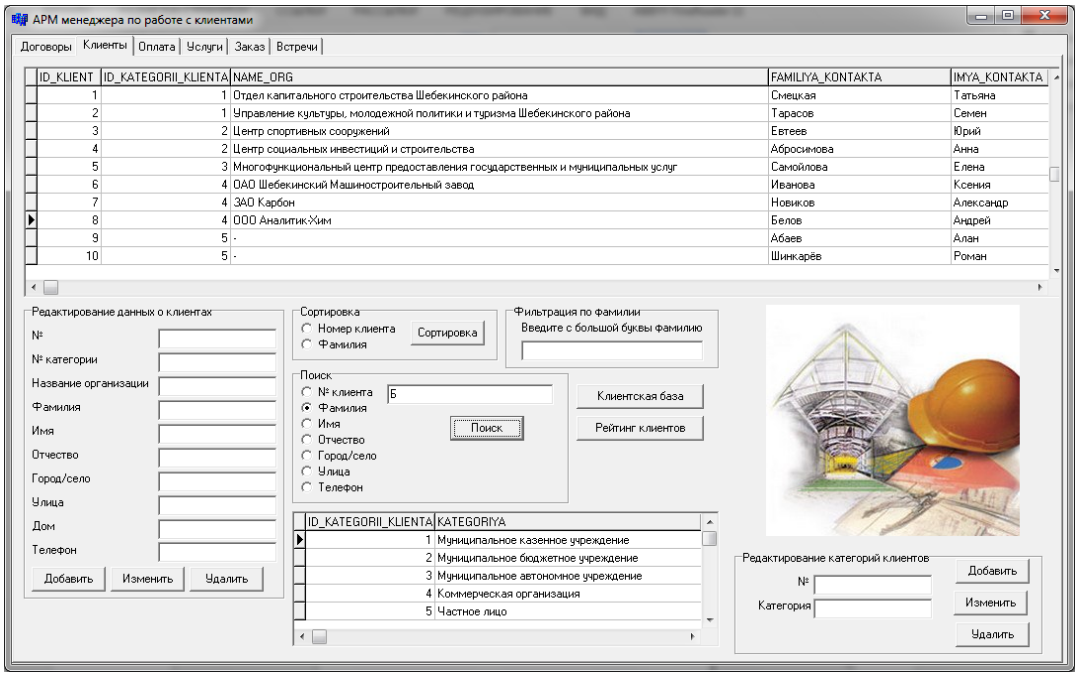


Рисунок 3.23 – Результат поиска данных

Аналогично происходят операции удаления, изменения и добавления данных в базу данных. Добавим в БД категорию клиента «Общество с ограниченной ответственностью». Результат добавления на рисунке 3.24.

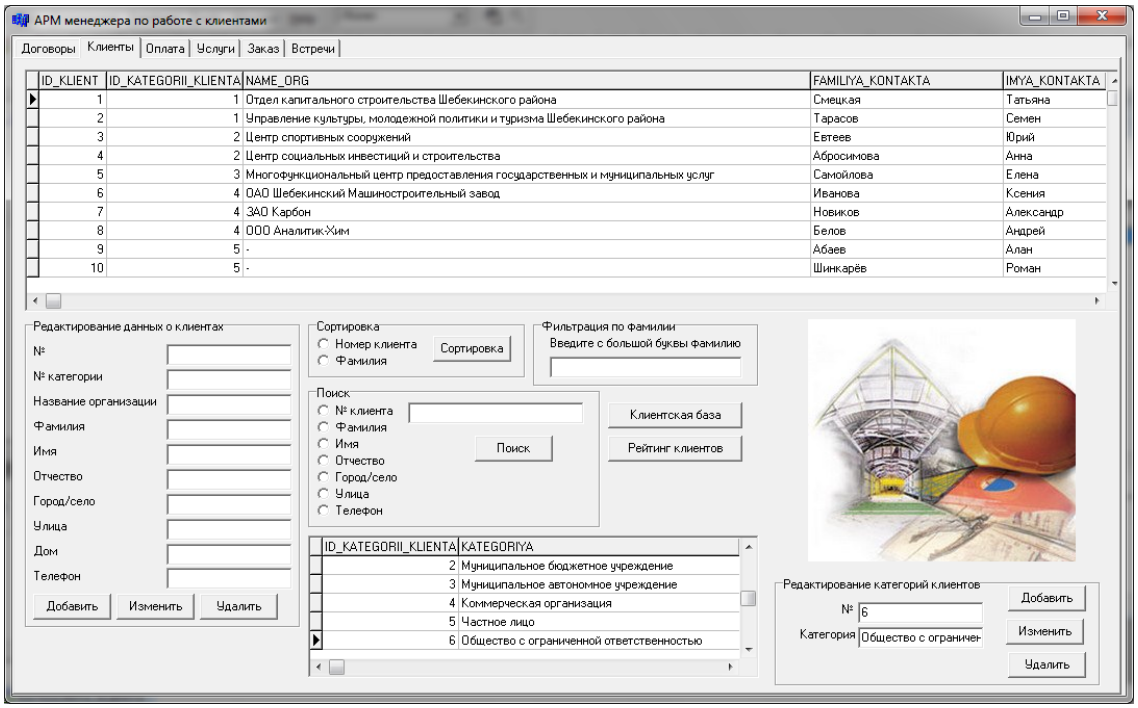


Рисунок 3.24 – Результат добавления данных

Помимо добавления данные можно изменять, поэтому заменим категорию клиента «Общество с ограниченной ответственностью» на «Открытое акционерное общество» [10]. Для этого необходимо в поле «№» ввести номер категории клиента, а в поле «Категория» ввести новые данные и нажать кнопку «Изменить». Результат изменения данных представлен на рисунке 3.25.

На рисунке 3.26 приведен результат удаления данных. Для этого необходимо в поле «№» ввести номер категории клиента для удаления и нажать кнопку «Удалить» [8].

На рисунках 3.27-3.28 приведены результаты выполнения отчетов «Отчет о клиентах» и «Рейтинг клиентов». На рисунках видно, что менеджер сможет выполнять такие операции с отчетами как сохранение и печать.

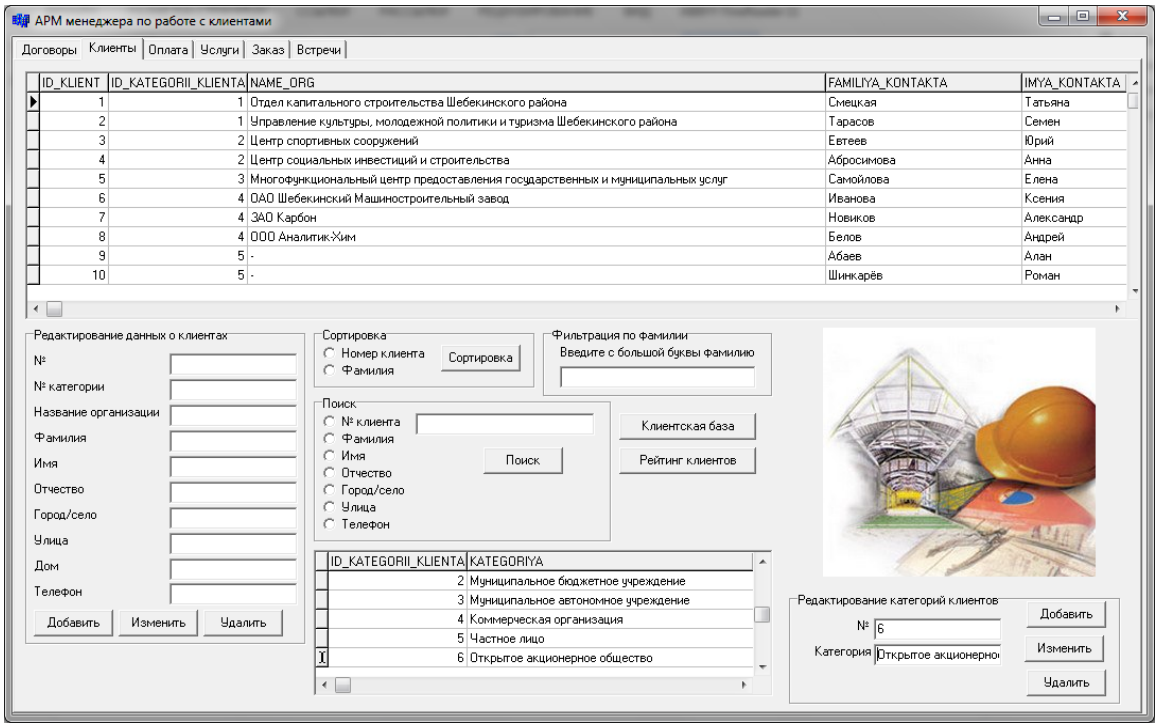


Рисунок 3.25 – Результат изменения данных

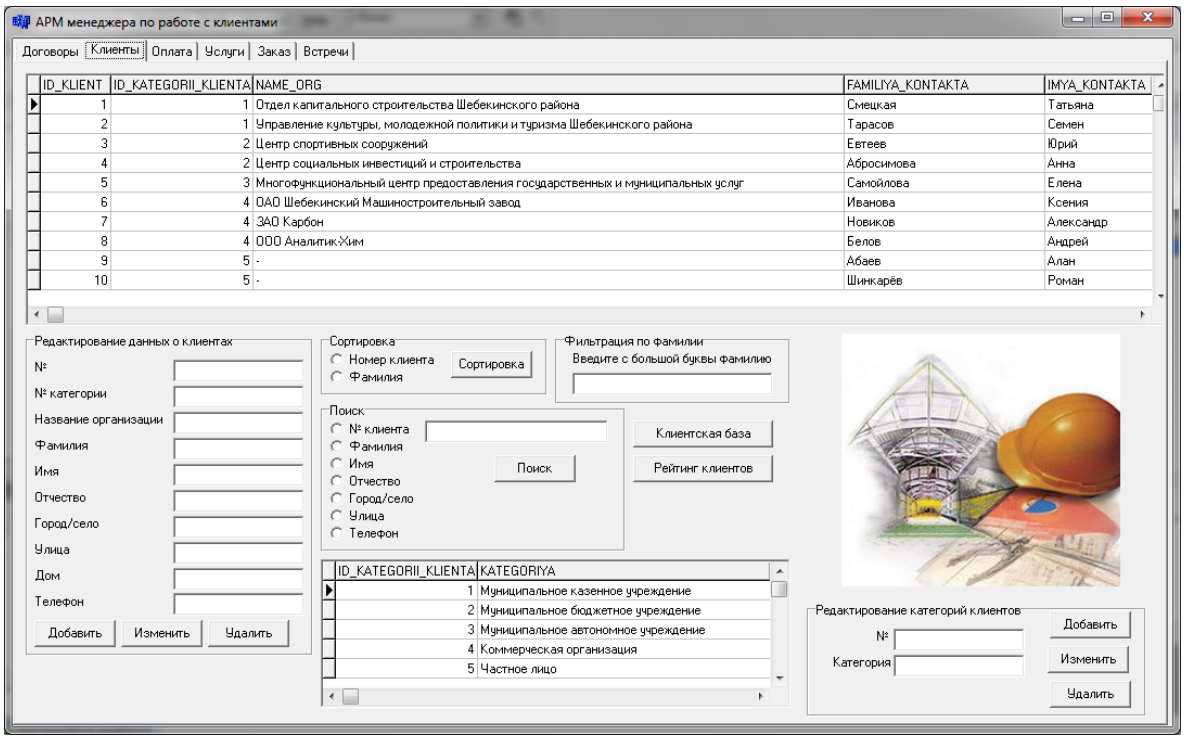


Рисунок 3.26 – Результат удаления данных

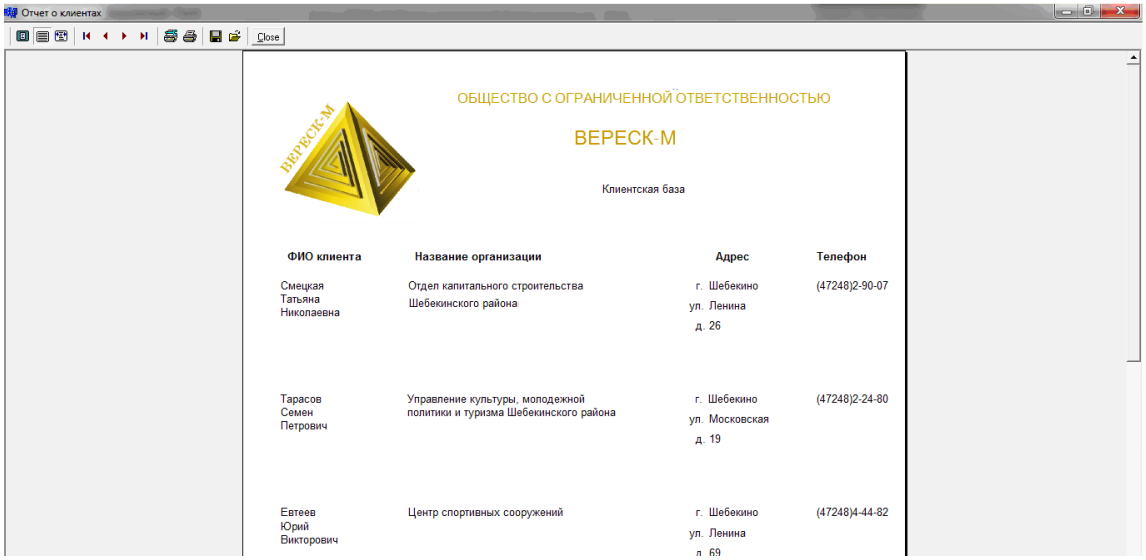


Рисунок 3.27 – Результат выполнения «Отчет о клиентах»

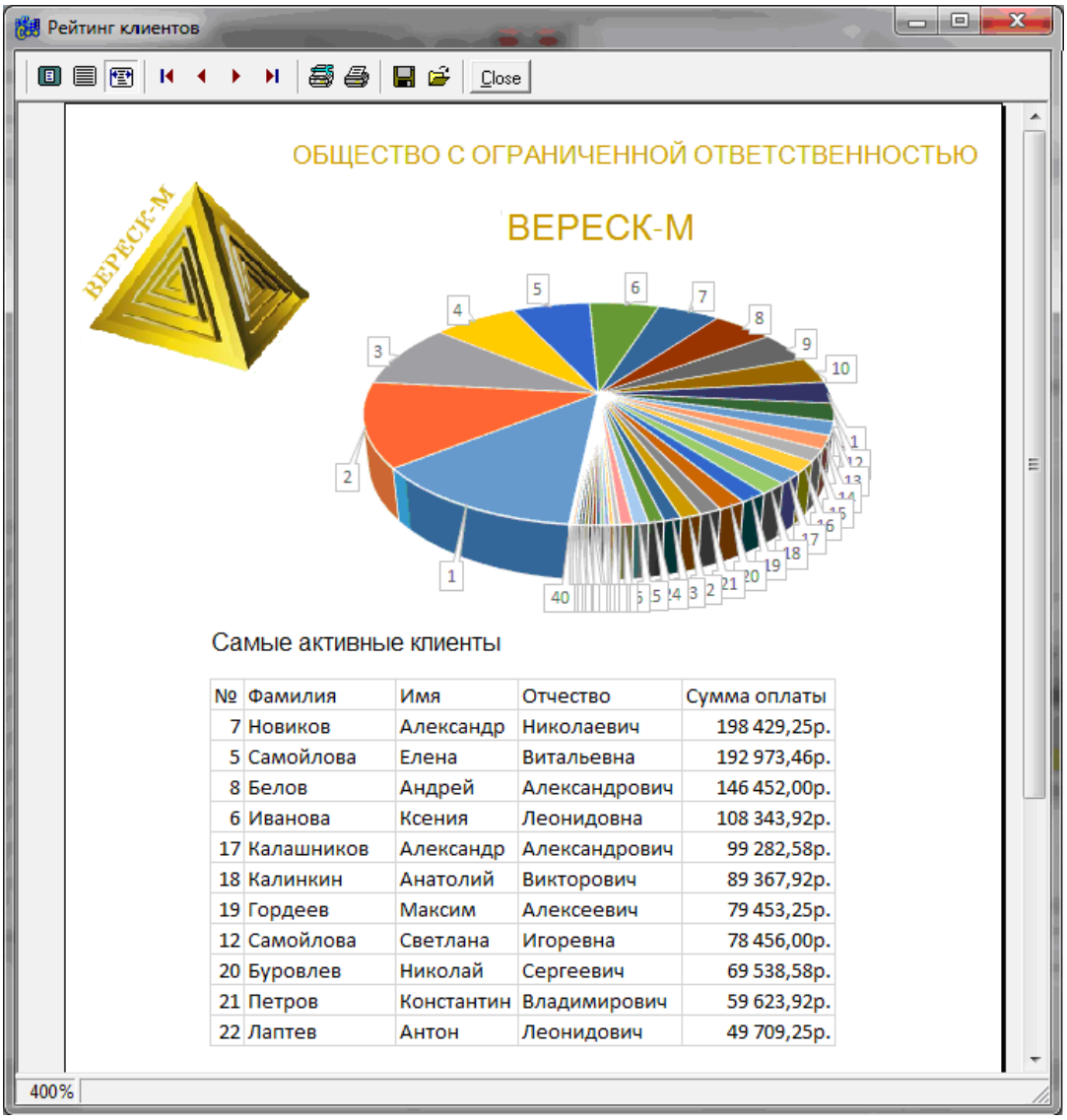


Рисунок 3.28 – Результат выполнения «Рейтинг клиентов»

3.4 Организационно-экономическая часть

3.4.1 Оценка эффективности функционирования АРМ

Разработанное АРМ создает принципиально новые условия для более эффективной организации работы с клиентами и документами. Сравнительная характеристика эффективности введения автоматизированной системы представлена в таблице 3.27.

Таблица 3.1 – Сравнительная характеристика эффективности введения автоматизированной системы

Неавтоматизированная технология	**Автоматизированная технология**
ручной прием документов, поступивших по электронным каналам (факс, телеграф, электронная почта).	автоматизированный прием документов, поступивших по электронным каналам.
ручная печать карточек, подготовка сводок, справок, отчетов, реестров	автоматическая печать карточек, стандартных сводок, справок и реестров, формирование произвольных отчетов
ручное перемещение проекта документа в процессе согласования и утверждения	автоматизированная пересылка проекта между согласующими и утверждающими должностными лицами по заранее заданному маршруту
размножение документов и карточек, транспортировка и передача их исполнителям	автоматическая рассылка необходимого числа электронных версий документов и карточек по компьютерной сети.
ручная регистрация факта передачи документов и резолюций должностному лицу, которая часто не выполняется ввиду халатности сотрудников	автоматическая регистрация фактов пересылки и получения документов и резолюций по компьютерной сети
многократная регистрация одних и тех же документов в журналах и архивах различных структурных подразделений	однократная первичная регистрация документа в любом подразделении и последующее автоматическое отслеживание движения и исполнения документа в электронных картотеках подразделений
ручной поиск документов и сбор сведений об их исполнении в разрозненных архивах различных подразделений	автоматизированный сквозной поиск по архиву с учетом прав доступа
почтовая и курьерская отправка документов и резолюций в филиалы и другие учреждения с ручным составлением соответствующих реестров	автоматизация функций оформления рассылки, отправка документов и резолюций в филиалы и другие учреждения посредством электронной почты с защитой их средствами криптографии

В таблице 3.2 приведены расчеты временных затрат по базовой технологии и прогнозируемые временные затраты по предлагаемому варианту.

Таблица 3.2 – Расчет временных затрат до и после внедрения АРМ

Действие сотрудника делопроизводства	Время, затрачиваемое при базовой технологии/ мин.	Время, затрачиваемое по предлагаемому варианту/ мин.
Заключение договора с новым клиентом	30–40	15–20
Перезаключение договора	30 - 40	5-7
Поиск документов клиента	20 - 50	5
Подготовка сопроводительных документов	30	5
Оформление платежных документов	30	10
Оформление пакета документов	20	10
Передача пакета документов	20-40	5
Итого, минут:	180-300	62

Таким образом, при существующей системе на совершение указанных в таблице действий необходимо было потратить от 3 до 5 часов. При внедрении предлагаемой системы время сокращается до 1 часа, то есть в несколько раз.

Ускорение бизнес-процессов – наиболее очевидная выгода от внедрения АРМ. При бумажном документообороте почта сотрудникам обычно разносится один - два раза в день, а в зависимости от регламента сопровождения документ может проходить инстанции от трех дней до месяца и более. Контроль задержки исполнения по отдельным документам невозможен, а комплексный контроль исполнительской дисциплины отдельного сотрудника осложняется непрозрачностью его деятельности. При внедрении АРМ документы передаются в другой отдел сразу же после оформления. То есть существенная экономия времени [18].

3.4.2 Расчет трудоемкости разработки АРМ

Для определения трудоемкости разработки приведен перечень всех основных этапов и видов работ, которые должны быть выполнены.

Форма разделения работ по этапам с указанием трудоемкости их выполнения приведена в таблице 3.3.

Таблица 3.3 – Трудоемкость этапов работ

Этап разработки АРМ	Вид работы на данном этапе	Трудоемкость разработки АРМ, ч.
Техническое задание	постановка задачи ; сбор материалов и анализ существующих разработок; определение требований к системе.	5 20 10
Эскизный проект	функционал, строение и дизайн системы.	20
Технический проект	выбор инструментальных средств; определение требований к аппаратному обеспечению.	10 10
Рабочий проект	разметка таблиц структуры БД; программирование; тестирование и отладка.	20 220 30
ИТОГО трудоемкость выполнения проекта	345	

Поскольку количество часов активной работы по разработке программного продукта равно 345 ч, а в сутки на разработку выделялось шесть часов, следовательно, срок выполнения проекта равен 58 суток.

3.4.3 Калькуляция себестоимости научно-технической продукции

ВКР предполагает использование разработанного АРМ, поэтому нужно определить стоимость создания и внедрения данного проекта.

Необходимо рассчитать:

1) стоимость затрат на материалы (представлены в таблице 3.4);

Таблица 3.4 – Стоимость материальных затрат

Наименование материальных затрат	Ед. изм.	Кол-во	Цена	Сумма
CD-RW	шт.	1	98	98
Бумага форматная белая, А4, пачка 500 листов	шт.	1	230	230
Картридж для принтера RICON SP 100	шт.	1	2779	2779
Итого:				3107

Было приобретено всего материалов на сумму 3107 рублей.

2) социальные отчисления;

Разработка выполнялась инженером-программистом в течение 58 рабочих дней при шестичасовом рабочем дне.

Месячный фонд времени работы инженера-программиста 132 часа, среднемесячная заработная плата 16000 руб.

З(ср.дн.) – 16000/22=727 руб/день.

Основная заработная плата разработчика составила:

З(осн) – 58*727 руб/день=42166 руб.

Дополнительная заработная плата составляет 20%:

З(доп)=0,2*З(осн)=0.2*42166= 8433 руб.

Затраты на оплату труда с учетом поясного коэффициента (25%):

ЗТР=1,25*(Зосн +Здоп)=1,25*(42166+8433)= 63248 руб.

3) отчисления на социальные нужды составляют:

– отчисления в Пенсионный фонд (28% от затрат на оплату труда):

0,28*42166=11806 руб.;

– отчисления в Фонд Социального страхования (4% от затрат на оплату труда):

0,04*42166=1686 руб.;

– отчисления в Федеральный Фонд обязательного медицинского страхования (0,2% от затрат на оплату труда):

0,002*42166=84 руб.;

– отчисления в Территориальный Фонд обязательного медицинского страхования (3,4% от затрат на оплату труда):

0,034*42166=1433 руб.

В итоге единый социальный налог составляет 15009 руб.

Страховой взнос на обязательное социальное страхование от несчастных случаев на производстве и профессиональных заболеваний (0,2% от затрат на оплату труда):

0,002*42166=84 руб.

В итоге отчисления на социальные нужды составляют 15093 руб.

4) Затраты на электроэнергию составляют: 58 дней по 6-часовой работе при мощности компьютера 0.45 кВт/ч стоимостью 3.53 руб за 1 КВт:

58*6*0.45*3.53=552.80 руб.

5) Амортизационные отчисления по эксплуатации технических средств составляет 20% от первоначальной стоимости (19000) при сроке эксплуатации 5 лет, т.е. сутки:

19000*0.2=3800, 3800/365=10.41 руб.

Амортизационные отчисления по эксплуатации технических средств при работе инженера-программиста составляют: 58*10.41=603.78 руб.

Капитальное вложение на проектирование, разработку и внедрение разработанного продукта представлено в таблице 3.31.

Таблица 3.31 – Капитальное вложение на разработку и внедрение АРМ

Наименование статей затрат	Сумма
Материалы	3107,00
Затраты на оплату труда инженера-программиста, занимавшегося созданием научно-технической продукции	42166,00
Отчисления на социальные нужды	15093,00
Затраты на расход электроэнергии	552,80
Амортизационные отчисления	603,78
Итого:	61522,58
Всего себестоимость	61522,58

Итак, в третьем разделе выпускной квалификационной работы был рассмотрен процесс проектирования и разработки АРМ менеджера по работе с клиентами. Используя CASE-средство AllFusion ERwin Data Modele была построена модель информационной базы данных и дана ее характеристика. Также рассчитана экономическая эффективность внедрения АРМ, временные затраты менеджером до внедрения АРМ и после. Приведена калькуляция себестоимости научно-технической продукции, которая наглядно показала, что внедрение разработанного АРМ эффективно и менее затратно.

ЗАКЛЮЧЕНИЕ

Написание выпускной квалификационной работы проводилось по данным строительной организации ООО «Вереск-М».

В результате выполнения ВКР была достигнута цель – повышение производительности компании.

Была приведена характеристика предприятия и отдела по работе с клиентами, обоснованы проектные решения по техническому, информационному, программному и технологическому обеспечению, смоделирована база данных, спроектировано и разработано автоматизированное рабочее место менеджера по работе с клиентами.

При разработке и проектировании АРМ менеджера по работе с клиентами был осуществлен анализ современных программных продуктов, которые позволяют создавать диалоговые системы. Подходящей программой оказалась Borland C++ Builder 6. Была выбрана методология проектирования AllFusion Process Modeler 7. При помощи этого средства были смоделированы диаграммы потоков данных в организации.

Также была проведена оценка целесообразности разработки АРМ менеджера по работе с клиентами с экономической точки зрения, приведена калькуляция себестоимости разработанного АРМ.

Для выполнения поставленной цели были выполнены следующие задачи:

1) проведен анализ деятельности предприятия;
2) сформулированы основные требования к разрабатываемому АРМ;
3) выполнен анализ и обоснован выбора инструментальных средств;
4) спроектировано и разработано автоматизированное рабочее место менеджера по работе с клиентами;
5) выполнено тестирование разработанного АРМ менеджера по работе с клиентами.

Таким образом, разработанное автоматизированное рабочее место

менеджера по работе с клиентами было протестировано с реальными данными и показало, что использование полученной разработки позволит ускорить и увеличить качество работы, выполняемой менеджером по работе с клиентами.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 7.32. Отчет о научно-исследовательской работе. Структура и правила оформления, 2001. – 22 с.
2. ГОСТ 2.105. Общие требования к текстовым документам, 1995г. – 2 с.
3. ГОСТ Р 7.0.5. Библиографическая ссылка. Общие требования и правила составления, 2008. – 16 с.
4. ГОСТ 7.1. Библиографическая запись. Библиографическое описание. Общие требования и правила составления, 2003. – 54 с.
5. ГОСТ 7.80. Библиографическая запись. Заголовок. Общие требования и правила составления, 2000. - 11 с.
6. ГОСТ 7.82. Библиографическая запись. Библиографическое описание электронных ресурсов. Общие требования и правила составления, 2001. – 27 с.
7. Алан Р. Саймон. Стратегические технологии баз данных: менеджмент на 2000 год / Пер. с англ и предисл. М.Р. Когаловского. – М.: Финансы и статистика, 1999.
8. Александреску, А. Современное проектирование на C++ [Текст]/ А. Александреску. – М.: Вильямс, 2015. – 336 с.
9. Алешин, Л.И. Информационные технологии [Текст]: учебное пособие / Л.И. Алешин. – М.: Маркет ДС, 2011. – 384 с.
10. Архангельский, А. Я. «C++ Builder 6 [Текст]: справочное пособие. Книга 1. Язык C++. – М.: Бином, 2002. – 544 с.
11. Балдин, К.В. Информационные технологии в менеджменте [Текст]: учеб. для студ. учреждений высш. проф. образования / К.В. Балдин. – М.: ИЦ Академия, 2012. – 288 с.
12. Баронов, В.В. Автоматизация управления предприятием [Текст] / В.В. Баронов, Г.Н.Калянов, – М.: Инфра-М, 2000.
13. Бондарь А. InterBase и Firebird: [Текст] практическое руководство

для умных пользователей и начинающих разработчиков / А. Бондарь. Издательство: БХВ-Петербург. 2012. – 592 с.

14. Вендров, А.М. Проектирование программного обеспечения экономических информационных систем [Текст]: учебник. – М.: Финансы и статистика, 2000.

15. Гаврилов, М.В. Информатика и информационные технологии [Текст]: учебник для бакалавров / М.В. Гаврилов, В.А. Климов; Рецензент Л.В. Кальянов, Н.М. Рыскин. – М.: Юрайт, 2013. – 378 с.

16. Гвоздева, В.А. Информатика, автоматизированные информационные технологии и системы [Текст]: учебник / В.А. Гвоздева. – М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. – 544 с.

17. Герб Саттер. Новые сложные задачи на C++ [Текст] / Пер. с англ. И. Красиков. – М.: Вильямс, 2015. – 272 с.

18. Герб Саттер. Решение сложных задач на C++ [Текст] / Пер. с англ. И. Красиков. – М.: Вильямс, 2015. – 400 с.

19. Герберт Шилдт. C++. Базовый курс [Текст] / Пер. с англ. Н. Ручко. – М.: Вильямс, 2014. – 624 с.

20. Годин, В.В. Информационное обеспечение управленческой деятельности [Текст] / В.В. Годин, И.К. Корнеев. – Мск. Изд. «Высшая школа», 2001.

21. Гришин, В.Н. Информационные технологии в профессиональной деятельности [Текст]: учебник / В.Н. Гришин, Е.Е. Панфилова. – М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. – 416 с.

22. Давыдова, Т.Ю. Информационные системы и технологии. Экономика. Управление. Бизнес [Текст]: учебное пособие / Т.Ю. Давыдова, С.И. Шелобаев, Ю.Н. Арсеньев. – Юнити-Дана 2012 г. – 447 с

23. Дейт К. Дж. Введение в системы баз данных. – 6-е изд. – М., СПб., Киев, Изд. дом Вильяме, 2000.

24. Джесси Рассел. SWOT-анализ [Текст] / Джесси Рассел. М.: Книга по Требованию, 2012. –42 с.

25. Иванова, Г.С. Технология программирования [Текст]: учебник для вузов. – 2-е изд., стереотип. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2003. – 320 с: ил. (Сер.Информатика в техническом университете.)

26. Ивасенко, А.Г. Информационные технологии в экономике и управлении [Текст]: учебное пособие / А.Г. Ивасенко, А.Ю. Гридасов, В.А. Павленко. - М.: КноРус, 2013. – 158 с.

27. Крэг Ларман. Применение UML 2.0 и шаблонов проектирования. Введение в объектно-ориентированный анализ, проектирование и итеративную разработку [Текст] / Пер. с англ. А. Шелестов. – М.: Вильямс, 2013. – 736 с.

28. Леоненков, А. Самоучитель UML 2 [Текст] / А. Леоненков. Издательство: БХВ-Петербург. 2011. – 576 с.

29. Липаев, В. В. Системное проектирование сложных программных средств для информационных систем [Текст]. – М.: Синтег, 1999.

30. Максимов Е.М. Базы данных в системах управления производственными процессами [Текст] / Е.М. Максимов, Н.Н. Бахтадзе. Издательство МГОУ 2011 г., – 160 с.

31. Максимов, Н.В. Современные информационные технологии [Текст]: учебное пособие / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. – М.: Форум, 2013. – 512 с.

32. Мельников, А.В. Информационные системы в экономике [Текст]: учебное пособие / А.В. Мельников, С.В. Бухарин. – ВГУИТ 2012 г. – 103 с.

33. Мельников, В.П. Информационные технологии [Текст]: учебник для студентов высших учебных заведений / В.П. Мельников. – М.: ИЦ Академия, 2009. – 432 с.

34. Мишенин, А.И. Теория экономических информационных систем [Текст]: учеб. пособие. — М.: Финансы и статистика, 2012.

35. Назарова, О.Б. Разработка реляционных баз данных с использованием CASE-средства All Fusion Data Modeler [Текст]: учеб. метод. пособие / О.Б. Назарова, О.Е. Масленникова. — 2-е изд., стер. — М.: ФЛИНТА, 2013. — 74 с.

36. Наумов А.Н. Системы управления базами данных и знаний [Текст] / А.Н. Наумов, А.М. Вендеров. – М.: Финансы и статистика, 2001г. – 348 с.

37. Петров В.Н. Информационные системы М – 688л. Изд. Питер, 2002.

38. Семенов М.И. Автоматизированные информационные технологии в экономике [Текст]: учебник / М.И. Семенов, И.Т. Трубилин, В.М. Лойко, Т.П. Барановская; Под общей ред. И.Т. Трубилина. — М.: Финансы и статистика, 2001.

39. Синаторов, С.В. Информационные технологии. [Текст]: чебное пособие / С.В. Синаторов. - М.: Альфа-М, НИЦ ИНФРА-М, 2013. – 336 с.

40. Стефан Рэнди Дэвис. C++ для чайников [Текст] / Пер. с англ. И. Красиков. – М.: Вильямс, 2014. – 336 с.

41. Ульман Дж. Основы систем баз данных [Текст] . – М.: Финансы и статистика, 2003г. – 320 с.

42. Эрих Гамма, Ричард Хелм, Ральф Джонсон, Джон Влиссидес. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] / Пер. с англ. А. Слинкин. – Спб.: Питер, 2015. –368 с.

43. Програмирование по-русски на C++.[Электронный ресурс]. – Режим доступа: http://codingrus.ru/

44. Региональный центр новых информационных технологий [электронный ресурс] / Режим доступа: http://rrc.karelia.ru/site/Resources/iias/

45. Системы управления базами данных и экспертные системы. [Электронный ресурс]. – Режим доступа: http://www.lessons-tva.info/edu/e-inf2/m2t4_2.html

46. Экономическая эффективность. [Электронный ресурс]. – Режим доступа: http://dic.academic.ru/dic.nsf/econ_dict/16622

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Должностные обязанности директора и функции отделов ООО «Вереск-М»

В должностные обязанности директора ООО «Вереск-М» входит: планирование и руководство деятельностью организацией; координация взаимодействия структурных подразделений организации; распределение обязанностей и определение степени ответственности работников организации; контроль над соблюдением стандартов качества работы; представление интересов организации в суде, арбитраже, органах государственной власти и управления.

Бухгалтерский учет осуществляется отделом бухгалтерии. Отдел выполняет следующие функции: осуществление предварительного контроля за своевременным и правильным оформлением документов; контроль за правильным и экономным расходованием средств; начисление и выплата заработной платы рабочим и служащим; участие в проведении инвентаризации денежных средств; составление и представление в ПФР и налоговые органы, соответствующей бухгалтерской и статистической отчетности в установленные сроки [21].

Отдел кадров является самостоятельным структурным подразделением предприятия. К основным функциям отдела кадров на предприятии относятся: подбор персонала; анализ текучести кадров; подготовка штатного расписания предприятия; оформление личных дел сотрудников; комплекс операций с трудовыми книжками; ведение учета отпусков, составление графиков и оформление отпусков; организация аттестаций сотрудников, составление планов карьерного движения персонала; подготовка планов повышения квалификации трудящихся [35].

Отдел по работе с клиентами — группа менеджеров, которые являются связующим звеном между клиентом и ООО «Вереск-М» [12]. В обязанности любого менеджера по работе с клиентами входит: планирование, организация и

контроль работы отдела - обслуживание и развитие отношений с существующими клиентами организации, участие в тендерах и привлечение новых клиентов, обработка входящих запросов, решение конфликтных ситуаций, подготовка и сдача отчетов, документооборот [20].

Производственный отдел - осуществляет реализацию всех мероприятий, связанных с производством. Он занимается организацией и выполнением различного рода подготовительных, строительных и монтажных работ. Также отделом осуществляется контроль над обеспечением производства технической документацией, оборудованием, инструментом, материалами, комплектующими изделиями, транспортом, погрузочно-разгрузочными средствами, а также за своевременное обеспечение строек проектно-сметной документацией, соблюдением строительных норм и правил, сдачей объектов в эксплуатацию в установленные сроки [40].

ПРИЛОЖЕНИЕ Б

Иллюстрации вкладок, разработанного АРМ менеджера по работе с клиентами

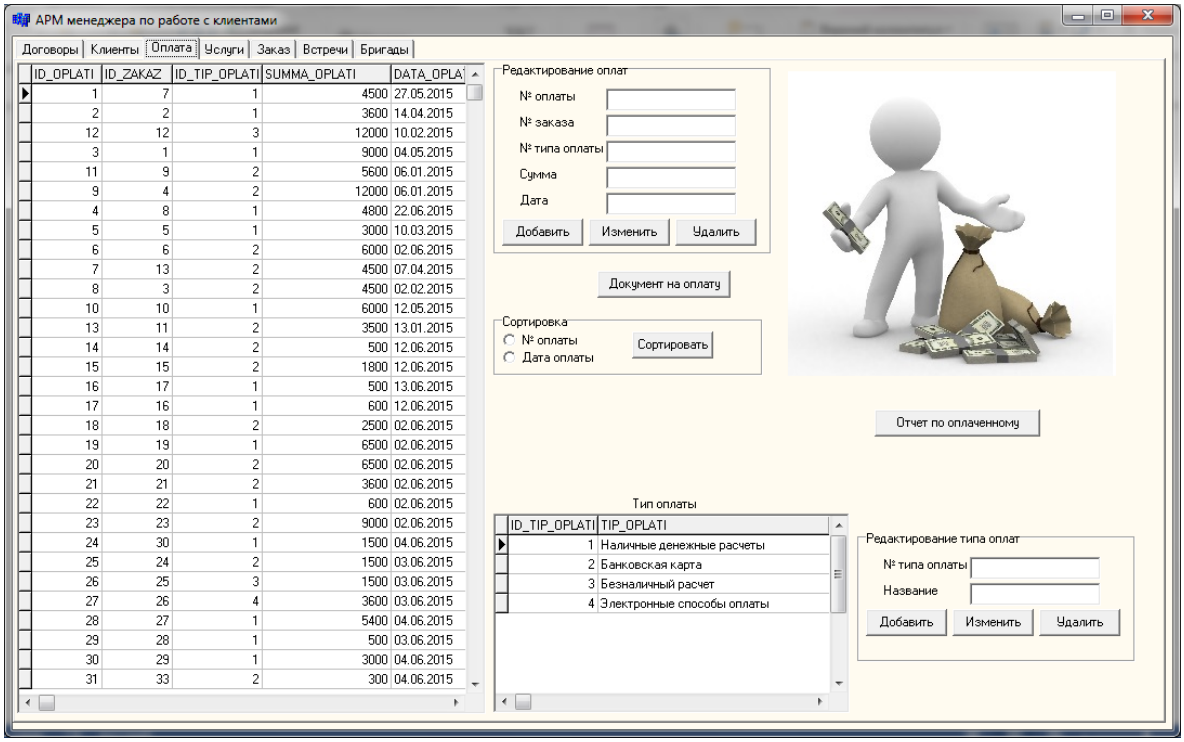


Рисунок Б.1 – Вкладка АРМ «Оплата»

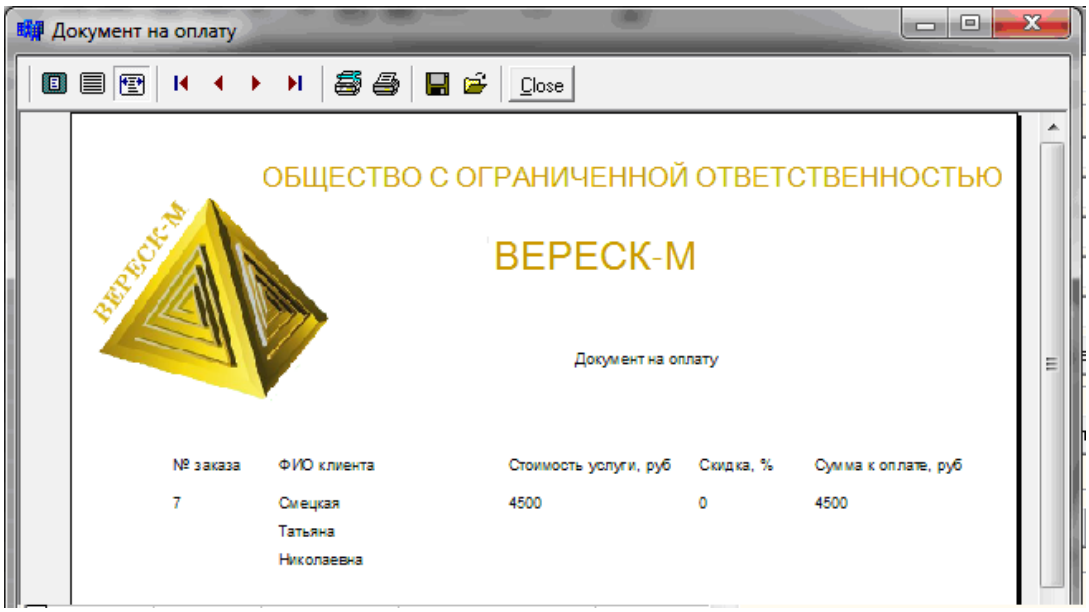


Рисунок Б.2 – Печатная форма документа на оплату

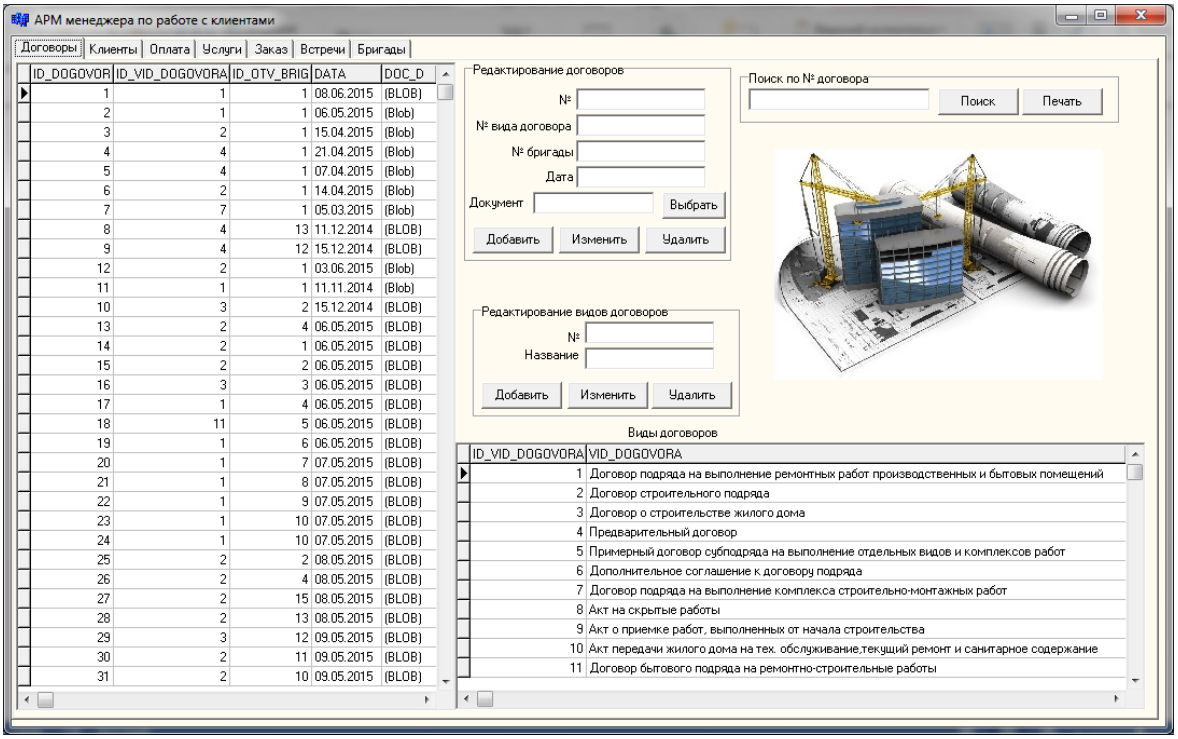


Рисунок Б.3 – Вкладка АРМ «Договоры»

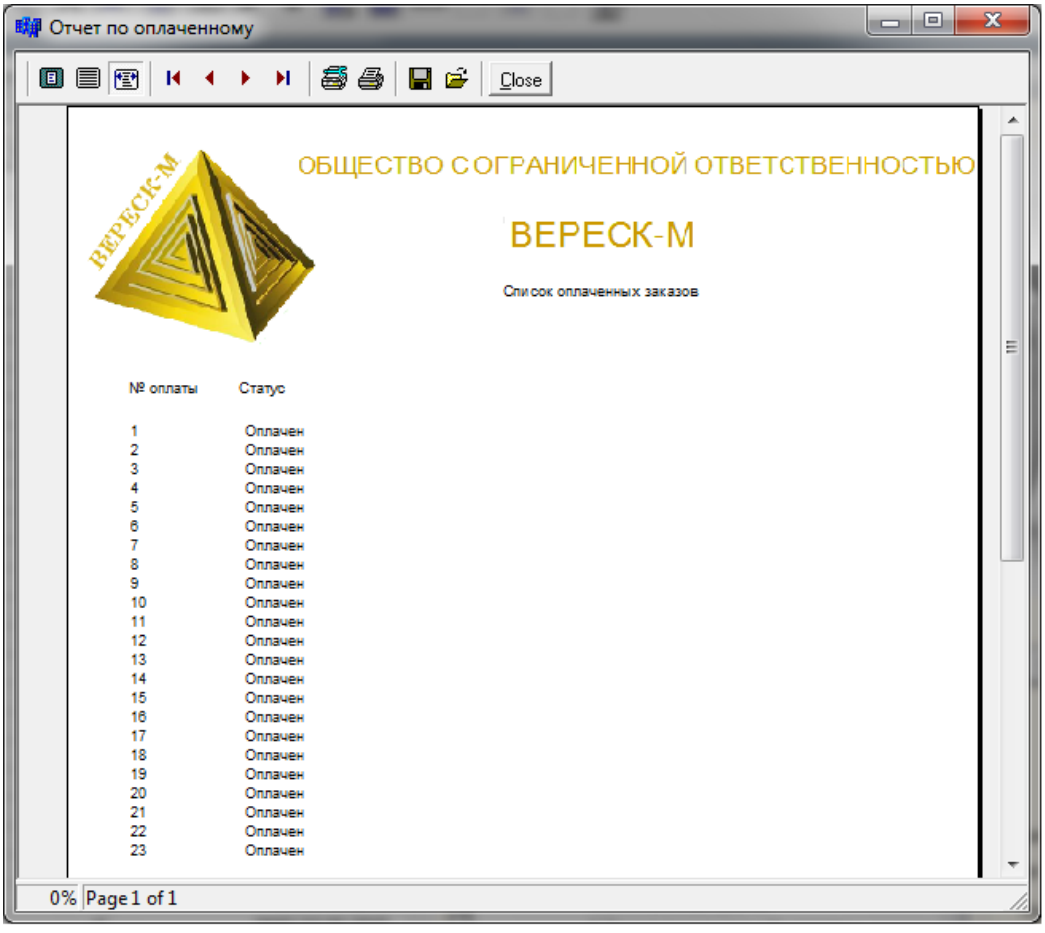


Рисунок Б.4 – Печатная форма отчета по оплаченным услугам

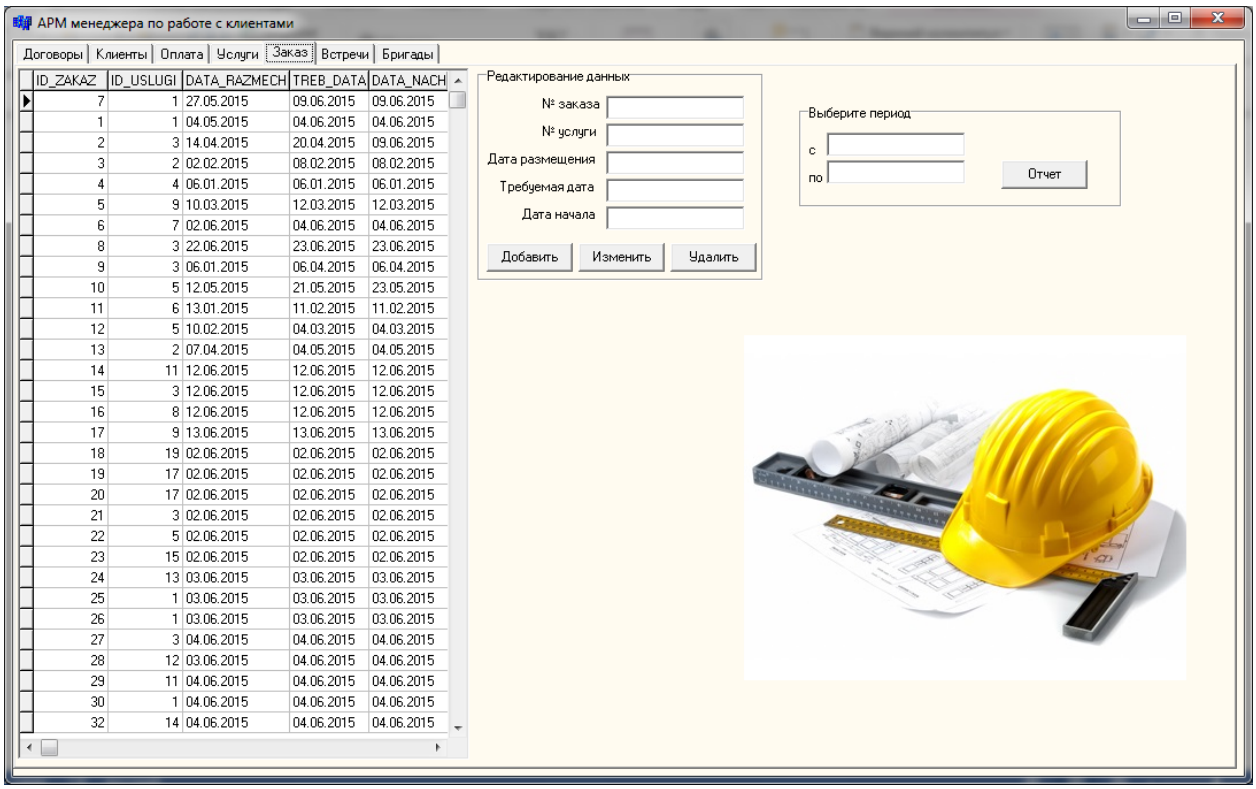


Рисунок Б.5 – Вкладка АРМ «Заказ»

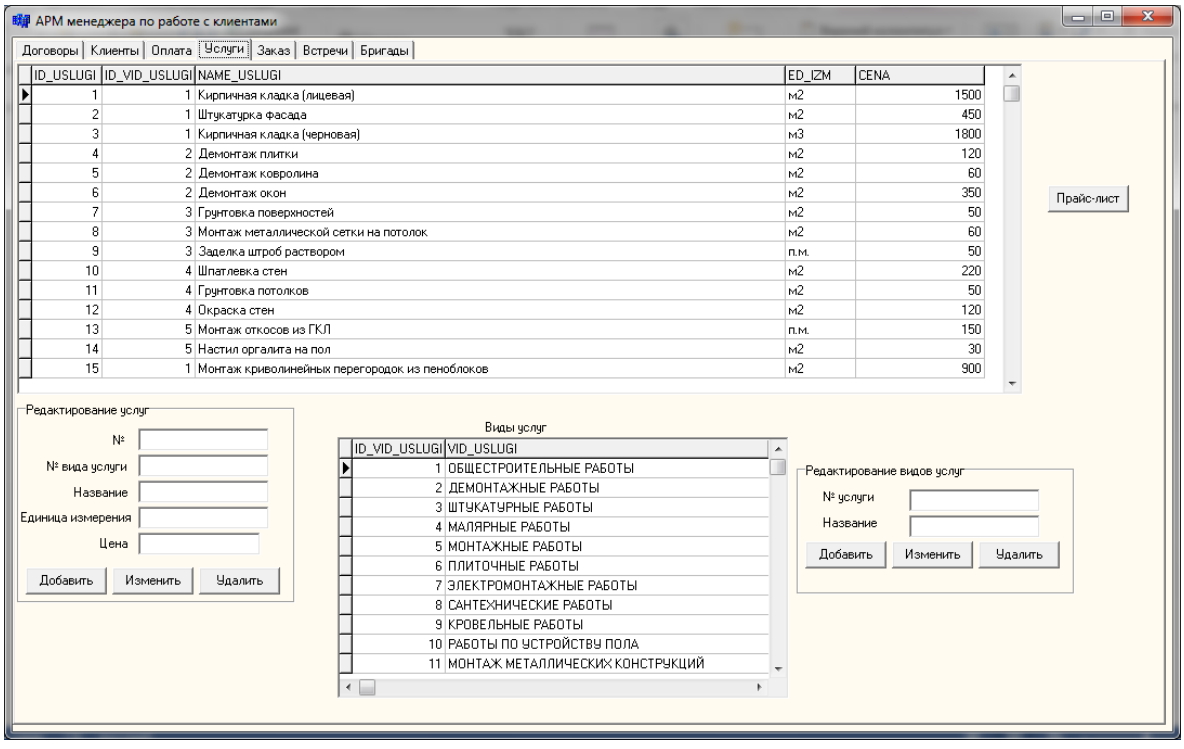


Рисунок Б.6 – Вкладка АРМ «Услуги»

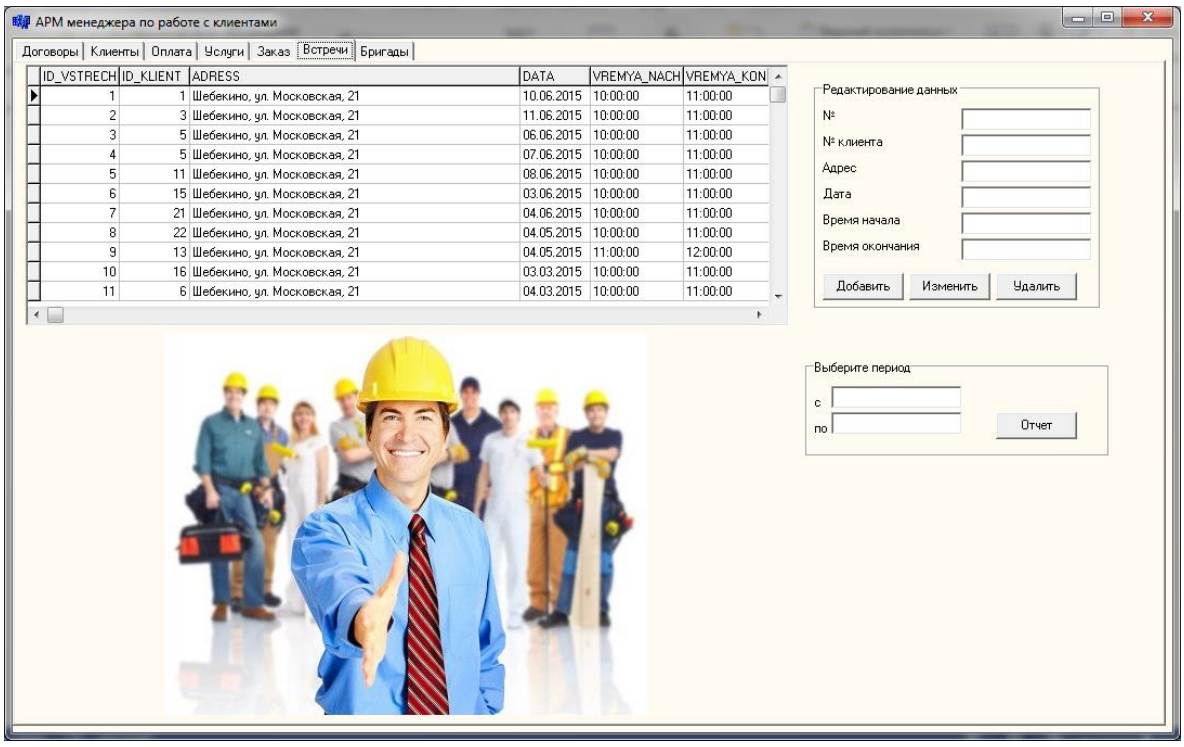


Рисунок Б.7 – Вкладка АРМ «Встречи»

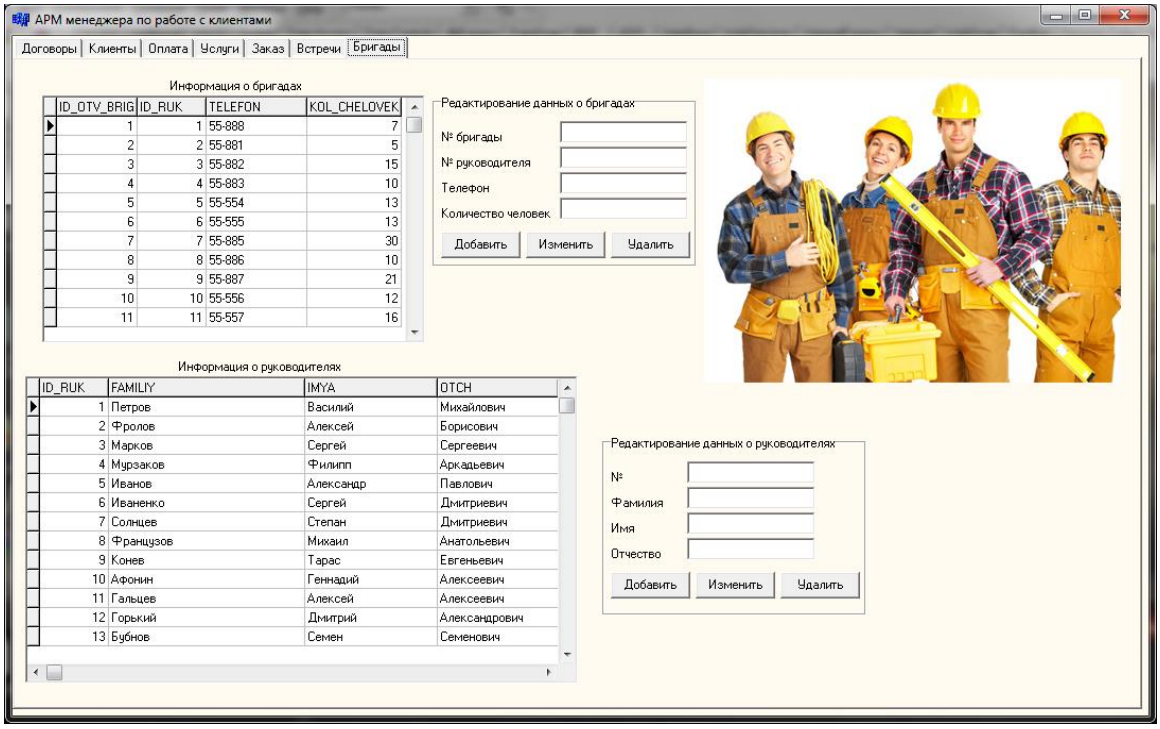


Рисунок Б.8 – Вкладка АРМ «Бригады»

ПРИЛОЖЕНИЕ В

Листинг кода, разработанного АРМ менеджера по работе с клиентами

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"
#include "Unit3.h"
#include "Unit4.h"
#include "Unit6.h"
#include "Unit7.h"
#include "Unit8.h"
#include "Unit9.h"
#include "Unit10.h"
#include "Unit11.h"
#include "Unit12.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button7Click(TObject *Sender)
{
  Form7->Visible=false;
  Form7->QuickRep1->Preview();
   Form7->Visible=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button4Click(TObject *Sender)
{
DataModule2->IBTableKLIENT->Active=false;

DataModule2->IBStoredProc5->ParamByName("ID_KLIENT")->AsString=Form1->Edit14-
>Text;
DataModule2->IBStoredProc5->ParamByName("ID_KATEGORII_KLIENTA")->AsString=Form1-
>Edit45->Text;
DataModule2->IBStoredProc5->ParamByName("NAME_ORG")->AsString=Form1->Edit44-
>Text;
DataModule2->IBStoredProc5->ParamByName("FAMILIYA_KONTAKTA")->AsString=Form1-
>Edit15->Text;
DataModule2->IBStoredProc5->ParamByName("IMYA_KONTAKTA")->AsString=Form1-
>Edit16->Text;
DataModule2->IBStoredProc5->ParamByName("OTCH_KONTAKTA")->AsString=Form1-
>Edit49->Text;
DataModule2->IBStoredProc5->ParamByName("GOROD_SELO")->AsString=Form1->Edit50-
>Text;
DataModule2->IBStoredProc5->ParamByName("ULICA")->AsString=Form1->Edit50->Text;
```

```
DataModule2->IBStoredProc5->ParamByName("DOM")->AsString=Form1->Edit8->Text;
DataModule2->IBStoredProc5->ParamByName("TELEFON")->AsString=Form1->Edit65-
>Text;

DataModule2->IBStoredProc5->Prepare();
DataModule2->IBStoredProc5->ExecProc();
DataModule2->IBTableKLIENT->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button5Click(TObject *Sender)
{
DataModule2->IBTableKLIENT->Active=false;

DataModule2->IBStoredProc6->ParamByName("ID_KLIENT")->AsString=Form1->Edit14-
>Text;
DataModule2->IBStoredProc6->ParamByName("ID_KATEGORII_KLIENTA")->AsString=Form1-
>Edit45->Text;
DataModule2->IBStoredProc6->ParamByName("NAME_ORG")->AsString=Form1->Edit44-
>Text;
DataModule2->IBStoredProc6->ParamByName("FAMILIYA_KONTAKTA")->AsString=Form1-
>Edit15->Text;
DataModule2->IBStoredProc6->ParamByName("IMYA_KONTAKTA")->AsString=Form1-
>Edit16->Text;
DataModule2->IBStoredProc6->ParamByName("OTCH_KONTAKTA")->AsString=Form1-
>Edit49->Text;
DataModule2->IBStoredProc6->ParamByName("GOROD_SELO")->AsString=Form1->Edit50-
>Text;
DataModule2->IBStoredProc6->ParamByName("ULICA")->AsString=Form1->Edit50->Text;
DataModule2->IBStoredProc6->ParamByName("DOM")->AsString=Form1->Edit8->Text;
DataModule2->IBStoredProc6->ParamByName("TELEFON")->AsString=Form1->Edit65-
>Text;

DataModule2->IBStoredProc6->Prepare();
DataModule2->IBStoredProc6->ExecProc();
DataModule2->IBTableKLIENT->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button6Click(TObject *Sender)
{
DataModule2->IBTableKLIENT->Active=false;
DataModule2->IBStoredProc7->ParamByName("ID_KLIENT")->AsString=Form1->Edit14-
>Text;
DataModule2->IBStoredProc7->Prepare();
DataModule2->IBStoredProc7->ExecProc();
DataModule2->IBTableKLIENT->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button9Click(TObject *Sender)
{
 if (RadioButton1->Checked==true)
        {DataModule2->IBTableKLIENT->IndexFieldNames="ID_KLIENT";}

        else if (RadioButton2->Checked==true)
        {DataModule2->IBTableKLIENT->IndexFieldNames="FAMILIYA_KONTAKTA";}

}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button10Click(TObject *Sender)
{
if (RadioButton4->Checked)
```

```
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("ID_KLIENT", Edit13->Text, LO<<loPartialKey<<
loCaseInsensitive);
}
else if (RadioButton5->Checked)
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("FAMILIYA_KONTAKTA",            Edit13->Text,
LO<<loPartialKey<< loCaseInsensitive);
}
else if (RadioButton34->Checked)
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("IMYA_KONTAKTA",                Edit13->Text,
LO<<loPartialKey<< loCaseInsensitive);
}
else if (RadioButton35->Checked)
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("OTCHESTVO", Edit13->Text, LO<<loPartialKey<<
loCaseInsensitive);
}
else if (RadioButton36->Checked)
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("NOMER_PASPORTA",               Edit13->Text,
LO<<loPartialKey<< loCaseInsensitive);
}
else if (RadioButton37->Checked)
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("ADRESS",   Edit13->Text,   LO<<loPartialKey<<
loCaseInsensitive);
}
else if (RadioButton38->Checked)
{
TLocateOptions LO;
DataModule2->IBTableKLIENT->Locate("TELEFON",  Edit13->Text,  LO<<loPartialKey<<
loCaseInsensitive);
}

}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button32Click(TObject *Sender)
{
  Form3->Visible=false;
  Form3->QuickRep1->Preview();
   Form3->Visible=true;
}
//---------------------------------------------------------------------------


void __fastcall TForm1::Button19Click(TObject *Sender)
{
  Form4->Visible=false;
  Form4->QuickRep1->Preview();
   Form4->Visible=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button8Click(TObject *Sender)
{
```

```
DataModule2->IBTableOPLATA->Active=false;

DataModule2->IBStoredProc8->ParamByName("ID_OPLATI")->AsString=Form1->Edit21-
>Text;
DataModule2->IBStoredProc8->ParamByName("ID_ZAKAZ")->AsString=Form1->Edit22-
>Text;
DataModule2->IBStoredProc8->ParamByName("ID_TIP_OPLATI")->AsString=Form1-
>Edit23->Text;
DataModule2->IBStoredProc8->ParamByName("SUMMA_OPLATI")->AsString=Form1->Edit24-
>Text;
DataModule2->IBStoredProc8->ParamByName("DATA_OPLATI")->AsString=Form1->Edit25-
>Text;

DataModule2->IBStoredProc8->Prepare();
DataModule2->IBStoredProc8->ExecProc();
DataModule2->IBTableOPLATA->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button11Click(TObject *Sender)
{
DataModule2->IBTableOPLATA->Active=false;

DataModule2->IBStoredProc9->ParamByName("ID_OPLATI")->AsString=Form1->Edit21-
>Text;
DataModule2->IBStoredProc9->ParamByName("ID_ZAKAZ")->AsString=Form1->Edit22-
>Text;
DataModule2->IBStoredProc9->ParamByName("ID_TIP_OPLATI")->AsString=Form1-
>Edit23->Text;
DataModule2->IBStoredProc9->ParamByName("SUMMA_OPLATI")->AsString=Form1->Edit24-
>Text;
DataModule2->IBStoredProc9->ParamByName("DATA_OPLATI")->AsString=Form1->Edit25-
>Text;

DataModule2->IBStoredProc9->Prepare();
DataModule2->IBStoredProc9->ExecProc();
DataModule2->IBTableOPLATA->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button12Click(TObject *Sender)
{
DataModule2->IBTableOPLATA->Active=false;

DataModule2->IBStoredProc10->ParamByName("ID_OPLATI")->AsString=Form1->Edit21-
>Text;


DataModule2->IBStoredProc10->Prepare();
DataModule2->IBStoredProc10->ExecProc();
DataModule2->IBTableOPLATA->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button22Click(TObject *Sender)
{
DataModule2->IBTableTipOplat->Active=false;

DataModule2->IBStoredProc20->ParamByName("ID_TIP_OPLATI")->AsString=Form1-
>Edit19->Text;
DataModule2->IBStoredProc20->ParamByName("TIP_OPLATI")->AsString=Form1->Edit20-
>Text;

DataModule2->IBStoredProc20->Prepare();
```

```
DataModule2->IBStoredProc20->ExecProc();
DataModule2->IBTableTipOplat->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button34Click(TObject *Sender)
{
DataModule2->IBTableTipOplat->Active=false;

DataModule2->IBStoredProc29->ParamByName("ID_TIP_OPLATI")->AsString=Form1-
>Edit19->Text;
DataModule2->IBStoredProc29->ParamByName("TIP_OPLATI")->AsString=Form1->Edit20-
>Text;

DataModule2->IBStoredProc29->Prepare();
DataModule2->IBStoredProc29->ExecProc();
DataModule2->IBTableTipOplat->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button37Click(TObject *Sender)
{
DataModule2->IBTableTipOplat->Active=false;

DataModule2->IBStoredProc30->ParamByName("ID_TIP_OPLATI")->AsString=Form1-
>Edit19->Text;

DataModule2->IBStoredProc30->Prepare();
DataModule2->IBStoredProc30->ExecProc();
DataModule2->IBTableTipOplat->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button13Click(TObject *Sender)
{
DataModule2->IBTableVidUslugi->Active=false;

DataModule2->IBStoredProc22->ParamByName("ID_VID_USLUGI")->AsString=Form1-
>Edit17->Text;
DataModule2->IBStoredProc22->ParamByName("VID_USLUGI")->AsString=Form1->Edit18-
>Text;

DataModule2->IBStoredProc22->Prepare();
DataModule2->IBStoredProc22->ExecProc();
DataModule2->IBTableVidUslugi->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button15Click(TObject *Sender)
{
DataModule2->IBTableVidUslugi->Active=false;

DataModule2->IBStoredProc23->ParamByName("ID_VID_USLUGI")->AsString=Form1-
>Edit17->Text;
DataModule2->IBStoredProc23->ParamByName("VID_USLUGI")->AsString=Form1->Edit18-
>Text;

DataModule2->IBStoredProc23->Prepare();
DataModule2->IBStoredProc23->ExecProc();
DataModule2->IBTableVidUslugi->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button23Click(TObject *Sender)
```

```
{
DataModule2->IBTableVidUslugi->Active=false;

DataModule2->IBStoredProc24->ParamByName("ID_VID_USLUGI")->AsString=Form1-
>Edit17->Text;

DataModule2->IBStoredProc24->Prepare();
DataModule2->IBStoredProc24->ExecProc();
DataModule2->IBTableVidUslugi->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button14Click(TObject *Sender)
{
DataModule2->IBTableUSLUGI->Active=false;

DataModule2->IBStoredProc11->ParamByName("ID_USLUGI")->AsString=Form1->Edit26-
>Text;
DataModule2->IBStoredProc11->ParamByName("ID_VID_USLUGI")->AsString=Form1-
>Edit27->Text;
DataModule2->IBStoredProc11->ParamByName("NAME_USLUGI")->AsString=Form1->Edit28-
>Text;
DataModule2->IBStoredProc11->ParamByName("ED_IZM")->AsString=Form1->Edit29-
>Text;
DataModule2->IBStoredProc11->ParamByName("CENA")->AsString=Form1->Edit30->Text;
DataModule2->IBStoredProc11->Prepare();
DataModule2->IBStoredProc11->ExecProc();
DataModule2->IBTableUSLUGI->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button29Click(TObject *Sender)
{
DataModule2->IBTableUSLUGI->Active=false;

DataModule2->IBStoredProc12->ParamByName("ID_USLUGI")->AsString=Form1->Edit26-
>Text;
DataModule2->IBStoredProc12->ParamByName("ID_VID_USLUGI")->AsString=Form1-
>Edit27->Text;
DataModule2->IBStoredProc12->ParamByName("NAME_USLUGI")->AsString=Form1->Edit28-
>Text;
DataModule2->IBStoredProc12->ParamByName("ED_IZM")->AsString=Form1->Edit29-
>Text;
DataModule2->IBStoredProc12->ParamByName("CENA")->AsString=Form1->Edit30->Text;
DataModule2->IBStoredProc12->Prepare();
DataModule2->IBStoredProc12->ExecProc();
DataModule2->IBTableUSLUGI->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button39Click(TObject *Sender)
{
 DataModule2->IBTableUSLUGI->Active=false;

DataModule2->IBStoredProc13->ParamByName("ID_USLUGI")->AsString=Form1->Edit26-
>Text;

DataModule2->IBStoredProc13->Prepare();
DataModule2->IBStoredProc13->ExecProc();
DataModule2->IBTableUSLUGI->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button16Click(TObject *Sender)
```

```
{
DataModule2->IBTableZAKAZ->Active=false;

DataModule2->IBStoredProc14->ParamByName("ID_ZAKAZ")->AsString=Form1->Edit35-
>Text;
DataModule2->IBStoredProc14->ParamByName("ID_USLUGI")->AsString=Form1->Edit36-
>Text;
DataModule2->IBStoredProc14->ParamByName("DATA_RAZMECH")->AsString=Form1-
>Edit37->Text;
DataModule2->IBStoredProc14->ParamByName("TREB_DATA")->AsString=Form1->Edit38-
>Text;
DataModule2->IBStoredProc14->ParamByName("DATA_NACH")->AsString=Form1->Edit39-
>Text;
DataModule2->IBStoredProc14->Prepare();
DataModule2->IBStoredProc14->ExecProc();
DataModule2->IBTableZAKAZ->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button17Click(TObject *Sender)
{
DataModule2->IBTableZAKAZ->Active=false;

DataModule2->IBStoredProc15->ParamByName("ID_ZAKAZ")->AsString=Form1->Edit35-
>Text;
DataModule2->IBStoredProc15->ParamByName("ID_USLUGI")->AsString=Form1->Edit36-
>Text;
DataModule2->IBStoredProc15->ParamByName("DATA_RAZMECH")->AsString=Form1-
>Edit37->Text;
DataModule2->IBStoredProc15->ParamByName("TREB_DATA")->AsString=Form1->Edit38-
>Text;
DataModule2->IBStoredProc15->ParamByName("DATA_NACH")->AsString=Form1->Edit39-
>Text;
DataModule2->IBStoredProc15->Prepare();
DataModule2->IBStoredProc15->ExecProc();
DataModule2->IBTableZAKAZ->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button18Click(TObject *Sender)
{
DataModule2->IBTableZAKAZ->Active=false;

DataModule2->IBStoredProc16->ParamByName("ID_ZAKAZ")->AsString=Form1->Edit35-
>Text;

DataModule2->IBStoredProc16->Prepare();
DataModule2->IBStoredProc16->ExecProc();
DataModule2->IBTableZAKAZ->Active=true;
}
//---------------------------------------------------------------------------


void __fastcall TForm1::Button28Click(TObject *Sender)
{
DataModule2->IBTableVSTRECHI->Active=false;

DataModule2->IBStoredProc4->ParamByName("ID_VSTRECH")->AsString=Form1->Edit31-
>Text;
DataModule2->IBStoredProc4->ParamByName("ID_KLIENT")->AsString=Form1->Edit52-
>Text;
DataModule2->IBStoredProc4->ParamByName("ADRESS")->AsString=Form1->Edit51->Text;
DataModule2->IBStoredProc4->ParamByName("DATA")->AsString=Form1->Edit34->Text;
DataModule2->IBStoredProc4->ParamByName("VREMYA_NACH")->AsString=Form1->Edit46-
```

```
>Text;
DataModule2->IBStoredProc4->ParamByName("VREMYA_KONEC")->AsString=Form1->Edit47-
>Text;

DataModule2->IBStoredProc4->Prepare();
DataModule2->IBStoredProc4->ExecProc();
DataModule2->IBTableVSTRECHI->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button40Click(TObject *Sender)
{
DataModule2->IBTableVSTRECHI->Active=false;

DataModule2->IBStoredProc28->ParamByName("ID_VSTRECH")->AsString=Form1->Edit31-
>Text;
DataModule2->IBStoredProc28->ParamByName("ID_KLIENT")->AsString=Form1->Edit52-
>Text;
DataModule2->IBStoredProc28->ParamByName("ADRESS")->AsString=Form1->Edit51-
>Text;
DataModule2->IBStoredProc28->ParamByName("DATA")->AsString=Form1->Edit34->Text;
DataModule2->IBStoredProc28->ParamByName("VREMYA_NACH")->AsString=Form1->Edit46-
>Text;
DataModule2->IBStoredProc28->ParamByName("VREMYA_KONEC")->AsString=Form1-
>Edit47->Text;

DataModule2->IBStoredProc28->Prepare();
DataModule2->IBStoredProc28->ExecProc();
DataModule2->IBTableVSTRECHI->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button41Click(TObject *Sender)
{
DataModule2->IBTableVSTRECHI->Active=false;

DataModule2->IBStoredProc21->ParamByName("ID_VSTRECH")->AsString=Form1->Edit31-
>Text;

DataModule2->IBStoredProc21->Prepare();
DataModule2->IBStoredProc21->ExecProc();
DataModule2->IBTableVSTRECHI->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button24Click(TObject *Sender)
{
DataModule2->IBTableVidDogovora->Active=false;

DataModule2->IBStoredProc25->ParamByName("ID_VID_DOGOVORA")->AsString=Form1-
>Edit6->Text;
DataModule2->IBStoredProc25->ParamByName("VID_DOGOVORA")->AsString=Form1->Edit7-
>Text;

DataModule2->IBStoredProc25->Prepare();
DataModule2->IBStoredProc25->ExecProc();
DataModule2->IBTableVidDogovora->Active=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button35Click(TObject *Sender)
{
DataModule2->IBTableVidDogovora->Active=false;
```

```
DataModule2->IBStoredProc26->ParamByName("ID_VID_DOGOVORA")->AsString=Form1-
>Edit6->Text;
DataModule2->IBStoredProc26->ParamByName("VID_DOGOVORA")->AsString=Form1->Edit7-
>Text;

DataModule2->IBStoredProc26->Prepare();
DataModule2->IBStoredProc26->ExecProc();
DataModule2->IBTableVidDogovora->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button36Click(TObject *Sender)
      {
      DataModule2->IBTableVidDogovora->Active=false;

      DataModule2->IBStoredProc27->ParamByName("ID_VID_DOGOVORA")-
>AsString=Form1->Edit6->Text;

      DataModule2->IBStoredProc27->Prepare();
      DataModule2->IBStoredProc27->ExecProc();
      DataModule2->IBTableVidDogovora->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button42Click(TObject *Sender)
      {
      DataModule2->IBTableBrig->Active=false;

      DataModule2->IBStoredProc31->ParamByName("ID_OTV_BRIG")->AsString=Form1-
>Edit42->Text;
      DataModule2->IBStoredProc31->ParamByName("ID_RUK")->AsString=Form1-
>Edit43->Text;
      DataModule2->IBStoredProc31->ParamByName("TELEFON")->AsString=Form1-
>Edit48->Text;
      DataModule2->IBStoredProc31->ParamByName("KOL_CHELOVEK")->AsString=Form1-
>Edit53->Text;

      DataModule2->IBStoredProc31->Prepare();
      DataModule2->IBStoredProc31->ExecProc();
      DataModule2->IBTableBrig->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button44Click(TObject *Sender)
      {
      DataModule2->IBTableBrig->Active=false;

      DataModule2->IBStoredProc32->ParamByName("ID_OTV_BRIG")->AsString=Form1-
>Edit42->Text;
      DataModule2->IBStoredProc32->ParamByName("ID_RUK")->AsString=Form1-
>Edit43->Text;
      DataModule2->IBStoredProc32->ParamByName("TELEFON")->AsString=Form1-
>Edit48->Text;
      DataModule2->IBStoredProc32->ParamByName("KOL_CHELOVEK")->AsString=Form1-
>Edit53->Text;

      DataModule2->IBStoredProc32->Prepare();
      DataModule2->IBStoredProc32->ExecProc();
      DataModule2->IBTableBrig->Active=true;
      }
      //---------------------------------------------------------------------
```

```
---

      void __fastcall TForm1::Button45Click(TObject *Sender)
      {
      DataModule2->IBTableBrig->Active=false;

      DataModule2->IBStoredProc33->ParamByName("ID_OTV_BRIG")->AsString=Form1-
>Edit42->Text;

      DataModule2->IBStoredProc33->Prepare();
      DataModule2->IBStoredProc33->ExecProc();
      DataModule2->IBTableBrig->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button46Click(TObject *Sender)
      {
      DataModule2->IBTableRuk->Active=false;

      DataModule2->IBStoredProc36->ParamByName("ID_RUK")->AsString=Form1-
>Edit54->Text;
      DataModule2->IBStoredProc36->ParamByName("FAMILIY")->AsString=Form1-
>Edit55->Text;
      DataModule2->IBStoredProc36->ParamByName("IMYA")->AsString=Form1->Edit56-
>Text;
      DataModule2->IBStoredProc36->ParamByName("OTCH")->AsString=Form1->Edit57-
>Text;

      DataModule2->IBStoredProc36->Prepare();
      DataModule2->IBStoredProc36->ExecProc();
      DataModule2->IBTableRuk->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button47Click(TObject *Sender)
      {
       DataModule2->IBTableRuk->Active=false;

      DataModule2->IBStoredProc35->ParamByName("ID_RUK")->AsString=Form1-
>Edit54->Text;
      DataModule2->IBStoredProc35->ParamByName("FAMILIY")->AsString=Form1-
>Edit55->Text;
      DataModule2->IBStoredProc35->ParamByName("IMYA")->AsString=Form1->Edit56-
>Text;
      DataModule2->IBStoredProc35->ParamByName("OTCH")->AsString=Form1->Edit57-
>Text;

      DataModule2->IBStoredProc35->Prepare();
      DataModule2->IBStoredProc35->ExecProc();
      DataModule2->IBTableRuk->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button48Click(TObject *Sender)
      {
        DataModule2->IBTableRuk->Active=false;

      DataModule2->IBStoredProc34->ParamByName("ID_RUK")->AsString=Form1-
>Edit54->Text;

      DataModule2->IBStoredProc34->Prepare();
```

```
      DataModule2->IBStoredProc34->ExecProc();
      DataModule2->IBTableRuk->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button1Click(TObject *Sender)
      {
       DataModule2->IBTableDOGOVOR->Active=false;

      DataModule2->IBStoredProc1->ParamByName("ID_DOGOVOR")->AsString=Form1-
>Edit1->Text;
      DataModule2->IBStoredProc1->ParamByName("ID_VID_DOGOVORA")-
>AsString=Form1->Edit2->Text;
      DataModule2->IBStoredProc1->ParamByName("ID_OTV_BRIG")->AsString=Form1-
>Edit3->Text;
      DataModule2->IBStoredProc1->ParamByName("DATA")->AsString=Form1->Edit4-
>Text;
      DataModule2->IBStoredProc1->ParamByName("DOC_D")->AsString=Form1->Edit5-
>Text;

      DataModule2->IBStoredProc1->Prepare();
      DataModule2->IBStoredProc1->ExecProc();
      DataModule2->IBTableDOGOVOR->Active=true;
      }
      //---------------------------------------------------------------------
---


      void __fastcall TForm1::Button2Click(TObject *Sender)
      {
       DataModule2->IBTableDOGOVOR->Active=false;

      DataModule2->IBStoredProc2->ParamByName("ID_DOGOVOR")->AsString=Form1-
>Edit1->Text;
      DataModule2->IBStoredProc2->ParamByName("ID_VID_DOGOVORA")-
>AsString=Form1->Edit2->Text;
      DataModule2->IBStoredProc2->ParamByName("ID_OTV_BRIG")->AsString=Form1-
>Edit3->Text;
      DataModule2->IBStoredProc2->ParamByName("DATA")->AsString=Form1->Edit4-
>Text;
      DataModule2->IBStoredProc2->ParamByName("DOC_D")->AsString=Form1->Edit5-
>Text;

      DataModule2->IBStoredProc1->Prepare();
      DataModule2->IBStoredProc1->ExecProc();
      DataModule2->IBTableDOGOVOR->Active=true;
      }
      //---------------------------------------------------------------------
---

      void __fastcall TForm1::Button3Click(TObject *Sender)
      {
       DataModule2->IBTableDOGOVOR->Active=false;

      DataModule2->IBStoredProc3->ParamByName("ID_DOGOVOR")->AsString=Form1-
>Edit1->Text;

      DataModule2->IBStoredProc3->Prepare();
      DataModule2->IBStoredProc3->ExecProc();
      DataModule2->IBTableDOGOVOR->Active=true;
      }
      //---------------------------------------------------------------------
---
```

```
      void __fastcall TForm1::Button49Click(TObject *Sender)
      {
       if (RadioButton3->Checked==true)
              {DataModule2->IBTableOPLATA->IndexFieldNames="ID_OPLATI";}

              else if (RadioButton6->Checked==true)
              {DataModule2->IBTableOPLATA->IndexFieldNames="DATA_OPLATI";}
      }
      //---------------------------------------------------------------------------
---

      void __fastcall TForm1::Button43Click(TObject *Sender)
      {
        Form6->Visible=false;
        Form6->QuickRep1->Preview();
         Form6->Visible=true;
      }
      //---------------------------------------------------------------------------
---

      void __fastcall TForm1::Button38Click(TObject *Sender)
      {
        Form9->Visible=false;
        Form9->QuickRep1->Preview();
         Form9->Visible=true;
      }
      //---------------------------------------------------------------------------
---


      void __fastcall TForm1::Button30Click(TObject *Sender)
      {
        Form12->Visible=false;
        Form12->QuickRep1->Preview();
         Form12->Visible=true;
      }
      //---------------------------------------------------------------------------
---

      void __fastcall TForm1::Buton5Click(TObject *Sender)
      {
        Form10->Visible=false;
        Form10->QuickRep1->Preview();
         Form10->Visible=true;
      }
      //---------------------------------------------------------------------------
---

      void __fastcall TForm1::Button26Click(TObject *Sender)
      {
        Form11->Visible=false;
        Form11->QuickRep1->Preview();
         Form11->Visible=true;
      }
      //---------------------------------------------------------------------------
---
```

ПРИЛОЖЕНИЕ В

Листинг SQL-кода базы данных, разработанного АРМ менеджера по работе с клиентами

```
/******************************************************************************/
/***                                                    Stored  Procedures  ***/
/******************************************************************************/

CREATE PROCEDURE ADD_D_K (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_KLIENT INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_D_Z (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_ZAKAZ INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_DOGOVOR (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_VID_DOGOVORA INTEGER,
    ID_OTV_BRIG INTEGER,
    DATA DATE,
    DOC_D BLOB SUB_TYPE 0 SEGMENT SIZE 80)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    KATEGORIYA VARCHAR(50) NOT NULL)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_KLIENT (
    ID_KLIENT INTEGER NOT NULL,
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    NAME_ORG VARCHAR(90),
    FAMILIYA_KONTAKTA VARCHAR(30) NOT NULL,
    IMYA_KONTAKTA VARCHAR(30) NOT NULL,
    OTCH_KONTAKTA VARCHAR(30) NOT NULL,
    GOROD_SELO VARCHAR(30) NOT NULL,
    ULICA VARCHAR(30) NOT NULL,
    DOM VARCHAR(10) NOT NULL,
    TELEFON VARCHAR(30) NOT NULL)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_OPLATA (
```

```
    ID_OPLATI INTEGER NOT NULL,
    ID_ZAKAZ INTEGER,
    ID_TIP_OPLATI INTEGER,
    SUMMA_OPLATI DECIMAL(15,2),
    DATA_OPLATI DATE)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_OTV_BRIG (
    ID_OTV_BRIG INTEGER NOT NULL,
    ID_RUK INTEGER,
    TELEFON VARCHAR(15),
    KOL_CHELOVEK INTEGER)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_RUK (
    ID_RUK INTEGER NOT NULL,
    FAMILIY VARCHAR(30),
    IMYA VARCHAR(20),
    OTCH VARCHAR(20))
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_TIP_OPLATI (
    ID_TIP_OPLATI INTEGER NOT NULL,
    TIP_OPLATI VARCHAR(50) NOT NULL)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_USLUGI (
    ID_USLUGI INTEGER NOT NULL,
    ID_VID_USLUGI INTEGER,
    NAME_USLUGI VARCHAR(90),
    ED_IZM VARCHAR(10),
    CENA DECIMAL(15,2))
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_VID_DOGOVORA (
    ID_VID_DOGOVORA INTEGER NOT NULL,
    VID_DOGOVORA VARCHAR(90) NOT NULL)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_VID_USL (
    ID_VID_USLUGI INTEGER NOT NULL,
    VID_USLUGI VARCHAR(50) NOT NULL)
AS
BEGIN
  EXIT;
END^
```

```
CREATE PROCEDURE ADD_VSTRECH (
    ID_VSTRECH INTEGER NOT NULL,
    ID_KLIENT INTEGER,
    ADRESS VARCHAR(50),
    DATA DATE,
    VREMYA_NACH TIME,
    VREMYA_KONEC TIME)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE ADD_ZAKAZ (
    ID_ZAKAZ INTEGER NOT NULL,
    ID_USLUGI INTEGER,
    DATA_RAZMECH DATE,
    TREB_DATA DATE,
    DATA_NACH DATE)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE DEL_D_K (
    ID_DOGOVOR INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_D_Z (
    ID_DOGOVOR INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_DOGOVOR (
    ID_DOGOVOR INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_KLIENT (
    ID_KLIENT INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^
```

```
CREATE PROCEDURE DEL_OPLATI (
    ID_OPLATI INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_OTD_BRIG (
    ID_OTV_BRIG INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_RUK (
    ID_RUK INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_TIP_OPLATI (
    ID_TIP_OPLATI INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_USLUGI (
    ID_USLUGI INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_VID_DOGOVORA (
    ID_VID_DOGOVORA INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_VID_USLUGI (
    ID_VID_USLUGI INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE DEL_VSTRECHI (
    ID_VSTRECH INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^
```

```
CREATE PROCEDURE DEL_ZAKAZ (
    ID_ZAKAZ INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPD_DOGOVOR (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_VID_DOGOVORA INTEGER,
    ID_OTV_BRIG INTEGER,
    DATA DATE,
    DOC_D BLOB SUB_TYPE 0 SEGMENT SIZE 80)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_D_K (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_KLIENT INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_D_Z (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_ZAKAZ INTEGER NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    KATEGORIYA VARCHAR(50) NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_KLIENT (
    ID_KLIENT INTEGER NOT NULL,
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    NAME_ORG VARCHAR(90),
    FAMILIYA_KONTAKTA VARCHAR(30) NOT NULL,
    IMYA_KONTAKTA VARCHAR(30) NOT NULL,
    OTCH_KONTAKTA VARCHAR(30) NOT NULL,
    GOROD_SELO VARCHAR(30) NOT NULL,
    ULICA VARCHAR(30) NOT NULL,
    DOM VARCHAR(10) NOT NULL,
    TELEFON VARCHAR(30) NOT NULL)
AS
BEGIN
  EXIT;
END^
```

```
CREATE PROCEDURE UPDATE_OPLATA (
    ID_OPLATI INTEGER NOT NULL,
    ID_ZAKAZ INTEGER,
    ID_TIP_OPLATI INTEGER,
    SUMMA_OPLATI DECIMAL(15,2),
    DATA_OPLATI DATE)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_OTV_BRIG (
    ID_OTV_BRIG INTEGER NOT NULL,
    ID_RUK INTEGER,
    TELEFON VARCHAR(15),
    KOL_CHELOVEK INTEGER)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_RUK (
    ID_RUK INTEGER NOT NULL,
    FAMILIY VARCHAR(30) NOT NULL,
    IMYA VARCHAR(20) NOT NULL,
    OTCH VARCHAR(20) NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_TIP_OPLATI (
    ID_TIP_OPLATI INTEGER NOT NULL,
    TIP_OPLATI VARCHAR(50) NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_USLUGI (
    ID_USLUGI INTEGER NOT NULL,
    ID_VID_USLUGI INTEGER,
    NAME_USLUGI VARCHAR(90),
    ED_IZM VARCHAR(10),
    CENA DECIMAL(15,2))
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_VID_DOGOVORA (
    ID_VID_DOGOVORA INTEGER NOT NULL,
    VID_DOGOVORA VARCHAR(90) NOT NULL)
AS
BEGIN
  EXIT;
END^
```

```
CREATE PROCEDURE UPDATE_VID_USL (
    ID_VID_USLUGI INTEGER NOT NULL,
    VID_USLUGI VARCHAR(50) NOT NULL)
AS
BEGIN
  EXIT;
END^


CREATE PROCEDURE UPDATE_VSTRECH (
    ID_VSTRECH INTEGER NOT NULL,
    ID_KLIENT INTEGER,
    ADRESS VARCHAR(50),
    DATA DATE,
    VREMYA_NACH TIME,
    VREMYA_KONEC TIME)
AS
BEGIN
  EXIT;
END^

CREATE PROCEDURE UPDATE_ZAKAZ (
    ID_ZAKAZ INTEGER NOT NULL,
    ID_USLUGI INTEGER,
    DATA_RAZMECH DATE,
    TREB_DATA DATE,
    DATA_NACH DATE)
AS
BEGIN
  EXIT;
END^

SET TERM ; ^
/******************************************************************************/
/***                                                                      Tables ***/
/******************************************************************************/



CREATE TABLE DOGOVOR (
    ID_DOGOVOR       INTEGER NOT NULL,
    ID_VID_DOGOVORA  INTEGER,
    ID_OTV_BRIG      INTEGER,
    DATA             DATE,
    DOC_D            BLOB SUB_TYPE 0 SEGMENT SIZE 80
);

CREATE TABLE DOGOVOR_KLIENT (
    ID_DOGOVOR  INTEGER NOT NULL,
    ID_KLIENT   INTEGER NOT NULL
);

CREATE TABLE DOGOVOR_ZAKAZ (
    ID_DOGOVOR  INTEGER NOT NULL,
    ID_ZAKAZ    INTEGER NOT NULL
);

CREATE TABLE IBE$REPORTS (
    IBE$REPORT_ID         INTEGER NOT NULL,
    IBE$REPORT_PARENT_ID  INTEGER NOT NULL,
    IBE$REPORT_NAME       VARCHAR(100) NOT NULL,
```

```
    IBE$REPORT_SOURCE     BLOB SUB_TYPE 0 SEGMENT SIZE 100,
    IBE$REPORT_RIGHTS     BLOB SUB_TYPE 0 SEGMENT SIZE 100,
    IBE$REPORT_IS_REPORT  SMALLINT DEFAULT 0 NOT NULL
);

CREATE TABLE KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA  INTEGER NOT NULL,
    KATEGORIYA            VARCHAR(50)
);

CREATE TABLE KLIENT (
    ID_KLIENT             INTEGER NOT NULL,
    ID_KATEGORII_KLIENTA  INTEGER,
    NAME_ORG              VARCHAR(90),
    FAMILIYA_KONTAKTA     VARCHAR(30),
    IMYA_KONTAKTA         VARCHAR(30),
    OTCH_KONTAKTA         VARCHAR(30),
    GOROD_SELO            VARCHAR(30),
    ULICA                 VARCHAR(30),
    DOM                   VARCHAR(10),
    TELEFON               VARCHAR(30)
);

CREATE TABLE OPLATA (
    ID_OPLATI      INTEGER NOT NULL,
    ID_ZAKAZ       INTEGER,
    ID_TIP_OPLATI  INTEGER,
    SUMMA_OPLATI   DECIMAL(15,2),
    DATA_OPLATI    DATE
);

CREATE TABLE OTV_BRIG (
    ID_OTV_BRIG   INTEGER NOT NULL,
    ID_RUK        INTEGER,
    TELEFON       VARCHAR(15),
    KOL_CHELOVEK  INTEGER
);

CREATE TABLE RUK (
    ID_RUK   INTEGER NOT NULL,
    FAMILIY  VARCHAR(30),
    IMYA     VARCHAR(20),
    OTCH     VARCHAR(20)
);

CREATE TABLE TIP_OPLATI (
    ID_TIP_OPLATI  INTEGER NOT NULL,
    TIP_OPLATI     VARCHAR(40)
);

CREATE TABLE USLUGI (
    ID_USLUGI      INTEGER NOT NULL,
    ID_VID_USLUGI  INTEGER,
    NAME_USLUGI    VARCHAR(90),
    ED_IZM         VARCHAR(10),
    CENA           DECIMAL(15,2)
);

CREATE TABLE VID_DOGOVORA (
    ID_VID_DOGOVORA  INTEGER NOT NULL,
    VID_DOGOVORA     VARCHAR(90)
);

CREATE TABLE VID_USLUGI (
```

```
    ID_VID_USLUGI  INTEGER NOT NULL,
    VID_USLUGI     VARCHAR(50)
);

CREATE TABLE VSTRECHI (
    ID_VSTRECH    INTEGER NOT NULL,
    ID_KLIENT     INTEGER,
    ADRESS        VARCHAR(50),
    DATA          DATE,
    VREMYA_NACH   TIME,
    VREMYA_KONEC  TIME
);

CREATE TABLE ZAKAZ (
    ID_ZAKAZ      INTEGER NOT NULL,
    ID_USLUGI     INTEGER,
    DATA_RAZMECH  DATE,
    TREB_DATA     DATE,
    DATA_NACH     DATE
);

/*************************************************************************
*****/
/***                                                                  Views
***/
/*************************************************************************
*****/

/* View: KLIENT_INFO */
CREATE VIEW KLIENT_INFO(
    ID_KLIENT,
    KATEGORIYA,
    NAME_ORG,
    FAMILIYA_KONTAKTA,
    IMYA_KONTAKTA,
    OTCH_KONTAKTA,
    GOROD_SELO,
    ULICA,
    DOM,
    TELEFON)
AS
select id_klient, kategoriya, name_org, familiya_kontakta, imya_kontakta,
otch_kontakta, GOROD_SELO, ULICA, DOM, TELEFON
from klient, kategoriya_klienta
where klient.id_kategorii_klienta=kategoriya_klienta.id_kategorii_klienta
;

/* View: NEW_VIEW */
CREATE VIEW NEW_VIEW(
    ID_KLIENT)
AS
select id_klient from klient
where id_klient like '1'
;

INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (1, 'Петров',
'Василий', 'Михайлович');
INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (2, 'Фролов',
'Алексей ', 'Борисович');
INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (3, 'Марков',
'Сергей', 'Сергеевич');
INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (4, 'Мурзаков',
'Филипп', 'Аркадьевич');
INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (5, 'Иванов',
```

```
'Александр', 'Павлович');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (6, 'Иваненко',
'Сергей ', 'Дмитриевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (7, 'Солнцев',
'Степан', 'Дмитриевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (8, 'Французов',
'Михаил', 'Анатольевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (9, 'Конев', 'Тарас',
'Евгеньевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (10, 'Афонин',
'Геннадий', 'Алексеевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (11, 'Гальцев',
'Алексей', 'Алексеевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (12, 'Горький',
'Дмитрий', 'Александрович');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (13, 'Бубнов',
'Семен', 'Семенович');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (14, 'Тонкий',
'Василий', 'Аркадьевич');
      INSERT INTO RUK (ID_RUK, FAMILIY, IMYA, OTCH) VALUES (15, 'Вялых',
'Матвей', 'Борисович');

      COMMIT WORK;

      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(1, 1, '55-888', 7);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(2, 2, '55-881', 5);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(3, 3, '55-882', 15);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(4, 4, '55-883', 10);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(5, 5, '55-554', 13);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(6, 6, '55-555', 13);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(7, 7, '55-885', 30);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(8, 8, '55-886', 10);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(9, 9, '55-887', 21);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(10, 10, '55-556', 12);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(11, 11, '55-557', 16);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(12, 12, '55-558', 9);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(13, 13, '55-559', 9);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(14, 14, '55-664', 8);
      INSERT INTO OTV_BRIG (ID_OTV_BRIG, ID_RUK, TELEFON, KOL_CHELOVEK) VALUES
(15, 15, '55-668', 8);

      COMMIT WORK;

      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (1,
'Договор подряда на выполнение ремонтных работ производственных и бытовых
помещений');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (2,
'Договор строительного подряда');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (3,
'Договор о строительстве жилого дома');
```

```
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (4,
'Предварительный договор');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (5,
'Примерный договор субподряда на выполнение отдельных видов и комплексов
работ');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (6,
'Дополнительное соглашение к договору подряда');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (7,
'Договор подряда на выполнение комплекса строительно-монтажных работ');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (8, 'Акт
на скрытые работы');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (9, 'Акт о
приемке работ, выполненных от начала строительства');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (10, 'Акт
передачи жилого дома на тех. обслуживание,текущий ремонт и санитарное
содержание');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (11,
'Договор бытового подряда на ремонтно-строительные работы');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (12,
'Договор подряда на выполнение отделочных работ по ремонту квартиры');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (13,
'Договор подряда на капитальное строительство');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (14,
'Договор подряда на строительно - монтажные работы');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (15,
'Особые условия к договору подряда на капитальное строительство');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (16,
'Препроводительное письмо к договору подряда');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (17,
'Договор генерального подряда');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (18,
'Договор аренды транспортных средств');
      INSERT INTO VID_DOGOVORA (ID_VID_DOGOVORA, VID_DOGOVORA) VALUES (19, 'Акт
приема/передачи транспортного средства, возвращаемого арендателем');

      COMMIT WORK;

      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (1, 1, 1, '2015-06-08');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (2, 1, 1, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (3, 2, 1, '2015-04-15');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (4, 4, 1, '2015-04-21');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (5, 4, 1, '2015-04-07');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (6, 2, 1, '2015-04-14');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (7, 7, 1, '2015-03-05');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (8, 4, 13, '2014-12-11');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (9, 4, 12, '2014-12-15');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (12, 2, 1, '2015-06-03');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (11, 1, 1, '2014-11-11');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (10, 3, 2, '2014-12-15');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (13, 2, 4, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
```

```
VALUES (14, 2, 1, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (15, 2, 2, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (16, 3, 3, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (17, 1, 4, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (18, 11, 5, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (19, 1, 6, '2015-05-06');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (20, 1, 7, '2015-05-07');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (21, 1, 8, '2015-05-07');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (22, 1, 9, '2015-05-07');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (23, 1, 10, '2015-05-07');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (24, 1, 10, '2015-05-07');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (25, 2, 2, '2015-05-08');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (26, 2, 4, '2015-05-08');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (27, 2, 15, '2015-05-08');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (28, 2, 13, '2015-05-08');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (29, 3, 12, '2015-05-09');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (30, 2, 11, '2015-05-09');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (31, 2, 10, '2015-05-09');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (32, 2, 6, '2015-05-09');
      INSERT INTO DOGOVOR (ID_DOGOVOR, ID_VID_DOGOVORA, ID_OTV_BRIG, DATA)
VALUES (33, 2, 7, '2015-05-09');

      COMMIT WORK;

      INSERT INTO KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA, KATEGORIYA) VALUES
(1, 'Муниципальное казенное учреждение');
      INSERT INTO KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA, KATEGORIYA) VALUES
(2, 'Муниципальное бюджетное учреждение');
      INSERT INTO KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA, KATEGORIYA) VALUES
(3, 'Муниципальное автономное учреждение');
      INSERT INTO KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA, KATEGORIYA) VALUES
(4, 'Коммерческая организация');
      INSERT INTO KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA, KATEGORIYA) VALUES
(5, 'Частное лицо');
      INSERT INTO KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA, KATEGORIYA) VALUES
(6, 'Общество с ограниченной ответственностью');

      COMMIT WORK;

      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (1, 1, 'Отдел капитального строительства Шебекинского района',
'Смецкая', 'Татьяна', 'Николаевна', 'Шебекино', 'Ленина', '26', '(47248)2-90-
07');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
```

```
TELEFON) VALUES (2, 1, 'Управление культуры, молодежной политики и туризма
Шебекинского района', 'Тарасов', 'Семен', 'Петрович', 'Шебекино', 'Московская',
'19', '(47248)2-24-80');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (3, 2, 'Центр спортивных сооружений', 'Евтеев', 'Юрий',
'Викторович', 'Шебекино', 'Ленина', '69', '(47248)4-44-82');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (4, 2, 'Центр социальных инвестиций и строительства',
'Абросимова', 'Анна', 'Валерьевна', 'Белгород', 'проспект Белгородский', '85а',
'(4722)32-11-49');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (5, 3, 'Многофункциональный центр предоставления государственных
и муниципальных услуг', 'Самойлова', 'Елена', 'Витальевна', 'Шебекино',
'Ленина', '70', '(47248)2-33-92');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (6, 4, 'ОАО Шебекинский Машиностроительный завод', 'Иванова',
'Ксения', 'Леонидовна', 'Шебекино', 'Октябрьская', '11', '(47248)3-32-55');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (7, 4, 'ЗАО Карбон', 'Новиков', 'Александр', 'Николаевич',
'Шебекино', 'Генерала Шумилова', '16', '(47248)2-02-57');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (8, 4, 'ООО Аналитик-Хим', 'Белов', 'Андрей', 'Александрович',
'Шебекино', 'Ржевское шоссе', '16', '(47248) 3-11-95');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (9, 5, '-', 'Абаев', 'Алан', 'Русланович', 'Нежеголь', 'Ленина',
'122', '8951-766-22-52');
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (10, 5, '-', 'Шинкарёв', 'Роман', NULL, NULL, NULL, NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (11, 5, '-', 'Копытова', 'Юлия', NULL, NULL, NULL, NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (12, 5, '-', 'Самойлова', 'Светлана', NULL, NULL, NULL, NULL,
NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (13, 5, '-', 'Белова', 'Татьяна', NULL, NULL, NULL, NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (14, 5, '-', 'Лебедев', 'Владимир', NULL, NULL, NULL, NULL,
NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (15, 5, '-', 'Выродов', 'Андрей', NULL, NULL, NULL, NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (16, 5, '-', 'Смецкой', 'Алексей', NULL, NULL, NULL, NULL,
NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (17, 5, '-', 'Калашников', 'Александр', NULL, NULL, NULL, NULL,
NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (18, 5, '-', 'Калинкин', 'Анатолий', NULL, NULL, NULL, NULL,
```

```
NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (19, 5, '-', 'Гордеев', 'Максим', NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (20, 5, '-', 'Буровлев', 'Николай', NULL, NULL, NULL, NULL,
NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (21, 5, '-', 'Петров', 'Константин', NULL, NULL, NULL, NULL,
NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (22, 5, '-', 'Лаптев', 'Антон', NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (23, 5, '-', 'Юдин', 'Сергей', NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (24, 5, '-', 'Тарасова', 'Галина', NULL, NULL, NULL, NULL,
NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (25, 5, '-', 'Лазарева', 'Юлия', NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (26, 4, 'ООО Белогорье', 'Лазарева', 'Светлана', NULL, NULL,
NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (27, 4, 'ООО БЗС Монокристалл', 'Позднякова', 'Юлия', NULL,
NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (28, 4, 'ООО  НПМП Связьэнергосервис', 'Новиков', 'Егор', NULL,
NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (30, 4, 'ЗАО Шебекинский маслозавод', 'Новгородцева', 'Тамара',
NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (31, 4, 'ЗАО Шебекинский меловой завод', 'Рыбак', 'Олег', NULL,
NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (32, 4, 'ООО Шебекинские корма', 'Высоцкая', 'Любовь', NULL,
NULL, NULL, NULL, NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (33, 4, 'ООО Селена', 'Карасева', 'Наталия', NULL, NULL, NULL,
NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (34, 4, 'ООО Белянка', 'Остапов', 'Антон', NULL, NULL, NULL,
NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (35, 4, 'ООО Бел Трейд', 'Городов', 'Сергей', NULL, NULL, NULL,
NULL, NULL);
      INSERT   INTO   KLIENT   (ID_KLIENT,   ID_KATEGORII_KLIENTA,   NAME_ORG,
FAMILIYA_KONTAKTA,  IMYA_KONTAKTA,  OTCH_KONTAKTA,  GOROD_SELO,  ULICA,  DOM,
TELEFON) VALUES (36, 4, 'ООО Урожай', 'Самойлов', 'Петр', NULL, NULL, NULL,
```

```
NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (37, 4, 'ЗАО Восход', 'Петров', 'Иван', NULL, NULL, NULL, NULL,
NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (38, 4, 'ЗАО Нива', 'Иванов', 'Егор', NULL, NULL, NULL, NULL,
NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (39, 4, 'ООО Победа', 'Калашникова', 'Юлия', NULL, NULL, NULL,
NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (40, 4, 'ЗАО Завод Премиксов 1', 'Буров', 'Михаил', NULL, NULL,
NULL, NULL, NULL);
      INSERT INTO KLIENT (ID_KLIENT, ID_KATEGORII_KLIENTA, NAME_ORG,
FAMILIYA_KONTAKTA, IMYA_KONTAKTA, OTCH_KONTAKTA, GOROD_SELO, ULICA, DOM,
TELEFON) VALUES (41, 4, 'ЗАО Краски «Шебекино-Мел»', 'Романов', 'Дмитрий', NULL,
NULL, NULL, NULL, NULL);

      COMMIT WORK;

      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (1, 1);
      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (2, 2);
      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (3, 5);
      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (4, 8);
      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (5, 12);
      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (6, 6);
      INSERT INTO DOGOVOR_KLIENT (ID_DOGOVOR, ID_KLIENT) VALUES (12, 4);

      COMMIT WORK;

      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (1, 1, 'Кирпичная кладка (лицевая)', 'м2', 1500);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (2, 1, 'Штукатурка фасада', 'м2', 450);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (3, 1, 'Кирпичная кладка (черновая)', 'м3', 1800);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (4, 2, 'Демонтаж плитки', 'м2', 120);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (5, 2, 'Демонтаж ковролина', 'м2', 60);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (6, 2, 'Демонтаж окон', 'м2', 350);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (7, 3, 'Грунтовка поверхностей', 'м2', 50);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (8, 3, 'Монтаж металлической сетки на потолок', 'м2', 60);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (9, 3, 'Заделка штроб раствором', 'п.м.', 50);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (10, 4, 'Шпатлевка стен', 'м2', 220);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (11, 4, 'Грунтовка потолков', 'м2', 50);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (12, 4, 'Окраска стен', 'м2', 120);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (13, 5, 'Монтаж откосов из ГКЛ', 'п.м.', 150);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (14, 5, 'Настил оргалита на пол', 'м2', 30);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (15, 1, 'Монтаж криволинейных перегородок из пеноблоков', 'м2', 900);
```

```
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (16, 1, 'Монтаж перегородок из гипсовых пазогребневых блоков', 'м2',
650);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (17, 1, 'Монтаж перегородок из пеноблоков', 'м2', 650);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (18, 1, 'Монтаж перегородок из стеклоблоков', 'м2', 800);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (19, 1, 'Монтаж гидроизоляции рулонной в 2 слоя', 'м2', 250);
      INSERT INTO USLUGI (ID_USLUGI, ID_VID_USLUGI, NAME_USLUGI, ED_IZM, CENA)
VALUES (20, 2, 'Демонтаж стен из ГКЛ', 'м2', 150);

      COMMIT WORK;

      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (7, 1, '2015-05-27', '2015-06-09', '2015-06-09');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (1, 1, '2015-05-04', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (2, 3, '2015-04-14', '2015-04-20', '2015-06-09');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (3, 2, '2015-02-02', '2015-02-08', '2015-02-08');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (4, 4, '2015-01-06', '2015-01-06', '2015-01-06');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (5, 9, '2015-03-10', '2015-03-12', '2015-03-12');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (6, 7, '2015-06-02', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (8, 3, '2015-06-22', '2015-06-23', '2015-06-23');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (9, 3, '2015-01-06', '2015-04-06', '2015-04-06');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (10, 5, '2015-05-12', '2015-05-21', '2015-05-23');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (11, 6, '2015-01-13', '2015-02-11', '2015-02-11');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (12, 5, '2015-02-10', '2015-03-04', '2015-03-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (13, 2, '2015-04-07', '2015-05-04', '2015-05-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (14, 11, '2015-06-12', '2015-06-12', '2015-06-12');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (15, 3, '2015-06-12', '2015-06-12', '2015-06-12');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (16, 8, '2015-06-12', '2015-06-12', '2015-06-12');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (17, 9, '2015-06-13', '2015-06-13', '2015-06-13');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (18, 19, '2015-06-02', '2015-06-02', '2015-06-02');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (19, 17, '2015-06-02', '2015-06-02', '2015-06-02');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (20, 17, '2015-06-02', '2015-06-02', '2015-06-02');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (21, 3, '2015-06-02', '2015-06-02', '2015-06-02');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (22, 5, '2015-06-02', '2015-06-02', '2015-06-02');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (23, 15, '2015-06-02', '2015-06-02', '2015-06-02');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (24, 13, '2015-06-03', '2015-06-03', '2015-06-03');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (25, 1, '2015-06-03', '2015-06-03', '2015-06-03');
```

```
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (26, 1, '2015-06-03', '2015-06-03', '2015-06-03');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (27, 3, '2015-06-04', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (28, 12, '2015-06-03', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (29, 11, '2015-06-04', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (30, 1, '2015-06-04', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (32, 14, '2015-06-04', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (33, 8, '2015-06-04', '2015-06-04', '2015-06-04');
      INSERT INTO ZAKAZ (ID_ZAKAZ, ID_USLUGI, DATA_RAZMECH, TREB_DATA,
DATA_NACH) VALUES (31, 12, '2015-06-04', '2015-06-04', '2015-06-04');

      COMMIT WORK;

      INSERT INTO DOGOVOR_ZAKAZ (ID_DOGOVOR, ID_ZAKAZ) VALUES (1, 7);
      INSERT INTO DOGOVOR_ZAKAZ (ID_DOGOVOR, ID_ZAKAZ) VALUES (2, 1);
      INSERT INTO DOGOVOR_ZAKAZ (ID_DOGOVOR, ID_ZAKAZ) VALUES (4, 2);
      INSERT INTO DOGOVOR_ZAKAZ (ID_DOGOVOR, ID_ZAKAZ) VALUES (5, 7);
      INSERT INTO DOGOVOR_ZAKAZ (ID_DOGOVOR, ID_ZAKAZ) VALUES (12, 12);

      COMMIT WORK;

      INSERT INTO TIP_OPLATI (ID_TIP_OPLATI, TIP_OPLATI) VALUES (1, 'Наличные
денежные расчеты');
      INSERT INTO TIP_OPLATI (ID_TIP_OPLATI, TIP_OPLATI) VALUES (2, 'Банковская
карта');
      INSERT INTO TIP_OPLATI (ID_TIP_OPLATI, TIP_OPLATI) VALUES (3, 'Безналичный
расчет');
      INSERT INTO TIP_OPLATI (ID_TIP_OPLATI, TIP_OPLATI) VALUES (4, 'Электронные
способы оплаты');

      COMMIT WORK;

      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (1, 7, 1, 4500, '2015-05-27');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (2, 2, 1, 3600, '2015-04-14');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (12, 12, 3, 12000, '2015-02-10');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (3, 1, 1, 9000, '2015-05-04');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (11, 9, 2, 5600, '2015-01-06');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (9, 4, 2, 12000, '2015-01-06');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (4, 8, 1, 4800, '2015-06-22');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (5, 5, 1, 3000, '2015-03-10');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (6, 6, 2, 6000, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (7, 13, 2, 4500, '2015-04-07');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (8, 3, 2, 4500, '2015-02-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (10, 10, 1, 6000, '2015-05-12');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (13, 11, 2, 3500, '2015-01-13');
```

```
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (14, 14, 2, 500, '2015-06-12');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (15, 15, 2, 1800, '2015-06-12');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (16, 17, 1, 500, '2015-06-13');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (17, 16, 1, 600, '2015-06-12');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (18, 18, 2, 2500, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (19, 19, 1, 6500, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (20, 20, 2, 6500, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (21, 21, 2, 3600, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (22, 22, 1, 600, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (23, 23, 2, 9000, '2015-06-02');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (24, 30, 1, 1500, '2015-06-04');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (25, 24, 2, 1500, '2015-06-03');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (26, 25, 3, 1500, '2015-06-03');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (27, 26, 4, 3600, '2015-06-03');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (28, 27, 1, 5400, '2015-06-04');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (29, 28, 1, 500, '2015-06-03');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (30, 29, 1, 3000, '2015-06-04');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (31, 33, 2, 300, '2015-06-04');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (32, 32, 1, 30000, '2015-06-04');
      INSERT INTO OPLATA (ID_OPLATI, ID_ZAKAZ, ID_TIP_OPLATI, SUMMA_OPLATI,
DATA_OPLATI) VALUES (33, 31, 2, 12000, '2015-06-04');

      COMMIT WORK;

      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (1,
'ОБЩЕСТРОИТЕЛЬНЫЕ РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (2, 'ДЕМОНТАЖНЫЕ
РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (3, 'ШТУКАТУРНЫЕ
РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (4, 'МАЛЯРНЫЕ
РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (5, 'МОНТАЖНЫЕ
РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (6, 'ПЛИТОЧНЫЕ
РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (7,
'ЭЛЕКТРОМОНТАЖНЫЕ РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (8,
'САНТЕХНИЧЕСКИЕ РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (9, 'КРОВЕЛЬНЫЕ
РАБОТЫ');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (10, 'РАБОТЫ ПО
УСТРОЙСТВУ ПОЛА');
      INSERT INTO VID_USLUGI (ID_VID_USLUGI, VID_USLUGI) VALUES (11, 'МОНТАЖ
```

```
МЕТАЛЛИЧЕСКИХ КОНСТРУКЦИЙ');

      COMMIT WORK;

      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (1, 1, 'Шебекино, ул. Московская, 21', '2015-06-10',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (2, 3, 'Шебекино, ул. Московская, 21', '2015-06-11',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (3, 5, 'Шебекино, ул. Московская, 21', '2015-06-06',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (4, 5, 'Шебекино, ул. Московская, 21', '2015-06-07',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (5, 11, 'Шебекино, ул. Московская, 21', '2015-06-08',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (6, 15, 'Шебекино, ул. Московская, 21', '2015-06-03',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (7, 21, 'Шебекино, ул. Московская, 21', '2015-06-04',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (8, 22, 'Шебекино, ул. Московская, 21', '2015-05-04',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (9, 13, 'Шебекино, ул. Московская, 21', '2015-05-04',
'11:00:00.000', '12:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (10, 16, 'Шебекино, ул. Московская, 21', '2015-03-03',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (11, 6, 'Шебекино, ул. Московская, 21', '2015-03-04',
'10:00:00.000', '11:00:00.000');
      INSERT INTO VSTRECHI (ID_VSTRECH, ID_KLIENT, ADRESS, DATA, VREMYA_NACH,
VREMYA_KONEC) VALUES (12, 7, 'Шебекино, ул. Московская, 21', '2015-03-04',
'11:00:00.000', '12:00:00.000');

      COMMIT WORK;



      /******************************************************************
*****/
      /***                                                    Primary  Keys
***/
      /******************************************************************
*****/

      ALTER TABLE DOGOVOR ADD CONSTRAINT PK_DOGOVOR PRIMARY KEY (ID_DOGOVOR);
      ALTER TABLE IBE$REPORTS ADD PRIMARY KEY (IBE$REPORT_ID);
      ALTER TABLE KATEGORIYA_KLIENTA ADD CONSTRAINT PK_KATEGORIYA_KLIENTA_1
PRIMARY KEY (ID_KATEGORII_KLIENTA);
      ALTER TABLE KLIENT ADD CONSTRAINT PK_KLIENT PRIMARY KEY (ID_KLIENT);
      ALTER TABLE OPLATA ADD CONSTRAINT PK_OPLATA PRIMARY KEY (ID_OPLATI);
      ALTER TABLE OTV_BRIG ADD CONSTRAINT PK_OTV_BRIG PRIMARY KEY (ID_OTV_BRIG);
      ALTER TABLE RUK ADD CONSTRAINT PK_RUK PRIMARY KEY (ID_RUK);
      ALTER TABLE TIP_OPLATI ADD CONSTRAINT PK_TIP_OPLATI PRIMARY KEY
(ID_TIP_OPLATI);
      ALTER TABLE USLUGI ADD CONSTRAINT PK_USLUGI PRIMARY KEY (ID_USLUGI);
```

```
      ALTER  TABLE  VID_DOGOVORA  ADD  CONSTRAINT  PK_VID_DOGOVORA  PRIMARY  KEY
(ID_VID_DOGOVORA);
      ALTER  TABLE  VID_USLUGI  ADD  CONSTRAINT  PK_VID_USLUGI  PRIMARY  KEY
(ID_VID_USLUGI);
      ALTER TABLE VSTRECHI ADD CONSTRAINT PK_VSTRECHI PRIMARY KEY (ID_VSTRECH);
      ALTER TABLE ZAKAZ ADD CONSTRAINT PK_ZAKAZ PRIMARY KEY (ID_ZAKAZ);


      /*************************************************************************
*****/
      /***                                                          Foreign  Keys
***/
      /*************************************************************************
*****/

      ALTER  TABLE  DOGOVOR  ADD  CONSTRAINT  FK_DOGOVOR_1  FOREIGN  KEY
(ID_VID_DOGOVORA) REFERENCES VID_DOGOVORA (ID_VID_DOGOVORA);
      ALTER TABLE DOGOVOR ADD CONSTRAINT FK_DOGOVOR_2 FOREIGN KEY (ID_OTV_BRIG)
REFERENCES OTV_BRIG (ID_OTV_BRIG);
      ALTER TABLE DOGOVOR_KLIENT ADD CONSTRAINT FK_DOGOVOR_KLIENT_1 FOREIGN KEY
(ID_DOGOVOR) REFERENCES DOGOVOR (ID_DOGOVOR);
      ALTER TABLE DOGOVOR_KLIENT ADD CONSTRAINT FK_DOGOVOR_KLIENT_2 FOREIGN KEY
(ID_KLIENT) REFERENCES KLIENT (ID_KLIENT);
      ALTER  TABLE  DOGOVOR_ZAKAZ  ADD  CONSTRAINT  FK_DOGOVOR_ZAKAZ_1  FOREIGN  KEY
(ID_ZAKAZ) REFERENCES ZAKAZ (ID_ZAKAZ);
      ALTER  TABLE  DOGOVOR_ZAKAZ  ADD  CONSTRAINT  FK_DOGOVOR_ZAKAZ_2  FOREIGN  KEY
(ID_DOGOVOR) REFERENCES DOGOVOR (ID_DOGOVOR);
      ALTER   TABLE   KLIENT   ADD   CONSTRAINT   FK_KLIENT_1   FOREIGN   KEY
(ID_KATEGORII_KLIENTA) REFERENCES KATEGORIYA_KLIENTA (ID_KATEGORII_KLIENTA);
      ALTER  TABLE  OPLATA  ADD  CONSTRAINT  FK_OPLATA_1  FOREIGN  KEY  (ID_ZAKAZ)
REFERENCES ZAKAZ (ID_ZAKAZ);
      ALTER TABLE OPLATA ADD CONSTRAINT FK_OPLATA_2 FOREIGN KEY (ID_TIP_OPLATI)
REFERENCES TIP_OPLATI (ID_TIP_OPLATI);
      ALTER  TABLE  OTV_BRIG  ADD  CONSTRAINT  FK_OTV_BRIG_1  FOREIGN  KEY  (ID_RUK)
REFERENCES RUK (ID_RUK);
      ALTER TABLE USLUGI ADD CONSTRAINT FK_USLUGI_1 FOREIGN KEY (ID_VID_USLUGI)
REFERENCES VID_USLUGI (ID_VID_USLUGI);
      ALTER TABLE VSTRECHI ADD CONSTRAINT FK_VSTRECHI_1 FOREIGN KEY (ID_KLIENT)
REFERENCES KLIENT (ID_KLIENT);
      ALTER  TABLE  ZAKAZ  ADD  CONSTRAINT  FK_ZAKAZ_1  FOREIGN  KEY  (ID_USLUGI)
REFERENCES USLUGI (ID_USLUGI);


      /*************************************************************************
*****/
      /***                                                        Stored  Procedures
***/
      /*************************************************************************
*****/


      SET TERM ^ ;

      ALTER PROCEDURE ADD_D_K (
          ID_DOGOVOR INTEGER NOT NULL,
          ID_KLIENT INTEGER NOT NULL)
      AS
      BEGIN
       insert       into       dogovor_klient       (dogovor_klient.id_dogovor,
dogovor_klient.id_klient)
        values (:id_dogovor, :id_klient);
      END^
```

```
ALTER PROCEDURE ADD_D_Z (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_ZAKAZ INTEGER NOT NULL)
AS
BEGIN
 insert into dogovor_zakaz (dogovor_zakaz.id_dogovor,
dogovor_zakaz.id_zakaz)
  values (:id_dogovor, :id_zakaz);
END^


ALTER PROCEDURE ADD_DOGOVOR (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_VID_DOGOVORA INTEGER,
    ID_OTV_BRIG INTEGER,
    DATA DATE,
    DOC_D BLOB SUB_TYPE 0 SEGMENT SIZE 80)
AS
BEGIN
 insert into DOGOVOR (DOGOVOR.id_dogovor, DOGOVOR.id_vid_dogovora,
DOGOVOR.id_otv_brig, DOGOVOR.data, DOGOVOR.doc_d)
  values (:ID_DOGOVOR, :ID_VID_DOGOVORA, :ID_OTV_BRIG, :DATA, :DOC_D);
END^


ALTER PROCEDURE ADD_KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    KATEGORIYA VARCHAR(50) NOT NULL)
AS
BEGIN
 insert into kategoriya_klienta (kategoriya_klienta.id_kategorii_klienta,
kategoriya_klienta.kategoriya)
  values (:id_kategorii_klienta, :kategoriya );
END^


ALTER PROCEDURE ADD_KLIENT (
    ID_KLIENT INTEGER NOT NULL,
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    NAME_ORG VARCHAR(90),
    FAMILIYA_KONTAKTA VARCHAR(30) NOT NULL,
    IMYA_KONTAKTA VARCHAR(30) NOT NULL,
    OTCH_KONTAKTA VARCHAR(30) NOT NULL,
    GOROD_SELO VARCHAR(30) NOT NULL,
    ULICA VARCHAR(30) NOT NULL,
    DOM VARCHAR(10) NOT NULL,
    TELEFON VARCHAR(30) NOT NULL)
AS
begin
 insert into klient (klient.id_klient, klient.id_kategorii_klienta,
klient.name_org, klient.familiya_kontakta,
  klient.imya_kontakta, klient.otch_kontakta, klient.gorod_selo,
klient.ulica, klient.dom, klient.telefon)
  values (:id_klient, :id_kategorii_klienta, :name_org,
:familiya_kontakta, :imya_kontakta, :otch_kontakta,
  :gorod_selo, :ulica, :dom, :telefon );
end^


ALTER PROCEDURE ADD_OPLATA (
    ID_OPLATI INTEGER NOT NULL,
    ID_ZAKAZ INTEGER,
    ID_TIP_OPLATI INTEGER,
    SUMMA_OPLATI DECIMAL(15,2),
```

```
          DATA_OPLATI DATE)
      AS
      begin
       insert      into      oplata      (oplata.id_oplati,      oplata.id_zakaz,
oplata.id_tip_oplati, oplata.summa_oplati, oplata.data_oplati)
        values    (:ID_OPLATI,    :ID_ZAKAZ,    :ID_TIP_OPLATI,    :SUMMA_OPLATI,
:DATA_OPLATI);
      end^


      ALTER PROCEDURE ADD_OTV_BRIG (
          ID_OTV_BRIG INTEGER NOT NULL,
          ID_RUK INTEGER,
          TELEFON VARCHAR(15),
          KOL_CHELOVEK INTEGER)
      AS
      begin
       insert    into    OTV_BRIG    (OTV_BRIG.ID_OTV_BRIG,    OTV_BRIG.ID_RUK,
OTV_BRIG.TELEFON, OTV_BRIG.KOL_CHELOVEK)
        values (:ID_OTV_BRIG, :ID_RUK, :TELEFON, :KOL_CHELOVEK);
      end^


      ALTER PROCEDURE ADD_RUK (
          ID_RUK INTEGER NOT NULL,
          FAMILIY VARCHAR(30),
          IMYA VARCHAR(20),
          OTCH VARCHAR(20))
      AS
      begin
       insert into ruk (ruk.id_ruk, ruk.familiy, ruk.imya, ruk.otch)
        values (:id_ruk, :FAMILIY, :IMYA, :OTCH);
      end^


      ALTER PROCEDURE ADD_TIP_OPLATI (
          ID_TIP_OPLATI INTEGER NOT NULL,
          TIP_OPLATI VARCHAR(50) NOT NULL)
      AS
      begin
       insert into tip_oplati (tip_oplati.id_tip_oplati, tip_oplati.tip_oplati)
        values (:ID_tip_oplati, :tip_oplati);
      end^


      ALTER PROCEDURE ADD_USLUGI (
          ID_USLUGI INTEGER NOT NULL,
          ID_VID_USLUGI INTEGER,
          NAME_USLUGI VARCHAR(90),
          ED_IZM VARCHAR(10),
          CENA DECIMAL(15,2))
      AS
      begin
       insert    into    uslugi    (uslugi.id_uslugi,    uslugi.id_vid_uslugi,
uslugi.name_uslugi, uslugi.ed_izm, uslugi.cena)
        values (:id_uslugi, :id_vid_uslugi, :name_uslugi, :ed_izm, :cena);
      end^


      ALTER PROCEDURE ADD_VID_DOGOVORA (
          ID_VID_DOGOVORA INTEGER NOT NULL,
          VID_DOGOVORA VARCHAR(90) NOT NULL)
      AS
      begin
```

```
       insert          into          vid_dogovora(vid_dogovora.id_vid_dogovora,
vid_dogovora.vid_dogovora)
        values (:ID_vid_dogovora, :vid_dogovora);
      end^


      ALTER PROCEDURE ADD_VID_USL (
          ID_VID_USLUGI INTEGER NOT NULL,
          VID_USLUGI VARCHAR(50) NOT NULL)
      AS
      begin
       insert into VID_USLUGI (VID_USLUGI.ID_VID_USLUGI, VID_USLUGI.VID_USLUGI)
        values (:ID_VID_USLUGI, :VID_USLUGI);
      end^


      ALTER PROCEDURE ADD_VSTRECH (
          ID_VSTRECH INTEGER NOT NULL,
          ID_KLIENT INTEGER,
          ADRESS VARCHAR(50),
          DATA DATE,
          VREMYA_NACH TIME,
          VREMYA_KONEC TIME)
      AS
      begin
       insert     into     vstrechi(vstrechi.id_vstrech,     vstrechi.id_klient,
vstrechi.adress, vstrechi.data, vstrechi.vremya_nach, vstrechi.vremya_konec)
        values   (:ID_VSTRECH,   :ID_KLIENT,   :ADRESS,   :DATA,   :VREMYA_NACH,
:VREMYA_KONEC);
      end^


      ALTER PROCEDURE ADD_ZAKAZ (
          ID_ZAKAZ INTEGER NOT NULL,
          ID_USLUGI INTEGER,
          DATA_RAZMECH DATE,
          TREB_DATA DATE,
          DATA_NACH DATE)
      AS
      begin
       insert into ZAKAZ (ZAKAZ.ID_ZAKAZ, ZAKAZ.ID_USLUGI, ZAKAZ.DATA_RAZMECH,
ZAKAZ.TREB_DATA, ZAKAZ.DATA_NACH)
        values (:ID_ZAKAZ, :ID_USLUGI, :DATA_RAZMECH, :TREB_DATA, :DATA_NACH);
      end^


      ALTER PROCEDURE DEL_D_K (
          ID_DOGOVOR INTEGER NOT NULL)
      AS
      BEGIN
       delete from dogovor_klient
      where dogovor_klient.id_dogovor=:id_dogovor;
      END^


      ALTER PROCEDURE DEL_D_Z (
          ID_DOGOVOR INTEGER NOT NULL)
      AS
      BEGIN
       delete from dogovor_zakaz
      where dogovor_zakaz.id_dogovor=:id_dogovor;
      END^
```

```
ALTER PROCEDURE DEL_DOGOVOR (
    ID_DOGOVOR INTEGER NOT NULL)
AS
BEGIN
 delete from DOGOVOR
where DOGOVOR.ID_DOGOVOR=:ID_DOGOVOR;
END^


ALTER PROCEDURE DEL_KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA INTEGER NOT NULL)
AS
BEGIN
delete from KATEGORIYA_KLIENTA
where KATEGORIYA_KLIENTA.ID_KATEGORII_KLIENTA=:ID_KATEGORII_KLIENTA;
END^


ALTER PROCEDURE DEL_KLIENT (
    ID_KLIENT INTEGER NOT NULL)
AS
begin
delete from KLIENT
where klient.id_klient=:id_klient;
end^


ALTER PROCEDURE DEL_OPLATI (
    ID_OPLATI INTEGER NOT NULL)
AS
BEGIN
 delete from oplata
where oplata.id_oplati =:id_oplati;
END^


ALTER PROCEDURE DEL_OTD_BRIG (
    ID_OTV_BRIG INTEGER NOT NULL)
AS
BEGIN
 delete from otv_brig
where otv_brig.id_otv_brig =:id_otv_brig;
END^


ALTER PROCEDURE DEL_RUK (
    ID_RUK INTEGER NOT NULL)
AS
BEGIN
 delete from ruk
where ruk.id_ruk =:id_ruk;
END^


ALTER PROCEDURE DEL_TIP_OPLATI (
    ID_TIP_OPLATI INTEGER NOT NULL)
AS
BEGIN
 delete from tip_oplati
where TIP_OPLATI.id_tip_oplati =:id_tip_oplati;
END^


ALTER PROCEDURE DEL_USLUGI (
```

```
    ID_USLUGI INTEGER NOT NULL)
AS
BEGIN
 delete from uslugi
where uslugi.id_uslugi =:id_uslugi;
END^


ALTER PROCEDURE DEL_VID_DOGOVORA (
    ID_VID_DOGOVORA INTEGER NOT NULL)
AS
BEGIN
 delete from vid_dogovora
where vid_dogovora.id_vid_dogovora =:id_vid_dogovora;
END^


ALTER PROCEDURE DEL_VID_USLUGI (
    ID_VID_USLUGI INTEGER NOT NULL)
AS
BEGIN
 delete from vid_uslugi
where vid_uslugi.id_vid_uslugi =:id_vid_uslugi;
END^


ALTER PROCEDURE DEL_VSTRECHI (
    ID_VSTRECH INTEGER NOT NULL)
AS
BEGIN
 delete from vstrechi
where vstrechi.id_vstrech =:id_vstrech;
END^


ALTER PROCEDURE DEL_ZAKAZ (
    ID_ZAKAZ INTEGER NOT NULL)
AS
BEGIN
 delete from zakaz
where zakaz.id_zakaz =:id_zakaz;
END^

ALTER PROCEDURE UPD_DOGOVOR (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_VID_DOGOVORA INTEGER,
    ID_OTV_BRIG INTEGER,
    DATA DATE,
    DOC_D BLOB SUB_TYPE 0 SEGMENT SIZE 80)
AS
BEGIN
   update DOGOVOR set
    DOGOVOR.ID_VID_DOGOVORA=:ID_VID_DOGOVORA,
   DOGOVOR.ID_OTV_BRIG=:ID_OTV_BRIG,
   DOGOVOR.DATA=:DATA,
   DOGOVOR.DOC_D=:DOC_D
 where DOGOVOR.ID_DOGOVOR=:ID_DOGOVOR;
suspend;
END^


ALTER PROCEDURE UPDATE_D_K (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_KLIENT INTEGER NOT NULL)
```

```
AS
begin
  update dogovor_klient set
  dogovor_klient.id_klient=:id_klient
  where dogovor_klient.id_dogovor=:id_dogovor;
suspend;
end^


ALTER PROCEDURE UPDATE_D_Z (
    ID_DOGOVOR INTEGER NOT NULL,
    ID_ZAKAZ INTEGER NOT NULL)
AS
begin
  update dogovor_zakaz set
  dogovor_zakaz.id_zakaz=:id_zakaz
  where dogovor_zakaz.id_dogovor=:id_dogovor;
suspend;
end^


ALTER PROCEDURE UPDATE_KATEGORIYA_KLIENTA (
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    KATEGORIYA VARCHAR(50) NOT NULL)
AS
BEGIN
  update KATEGORIYA_KLIENTA set
    KATEGORIYA_KLIENTA.KATEGORIYA=:KATEGORIYA
 where KATEGORIYA_KLIENTA.id_kategorii_klienta=:id_kategorii_klienta;
suspend;
END^


ALTER PROCEDURE UPDATE_KLIENT (
    ID_KLIENT INTEGER NOT NULL,
    ID_KATEGORII_KLIENTA INTEGER NOT NULL,
    NAME_ORG VARCHAR(90),
    FAMILIYA_KONTAKTA VARCHAR(30) NOT NULL,
    IMYA_KONTAKTA VARCHAR(30) NOT NULL,
    OTCH_KONTAKTA VARCHAR(30) NOT NULL,
    GOROD_SELO VARCHAR(30) NOT NULL,
    ULICA VARCHAR(30) NOT NULL,
    DOM VARCHAR(10) NOT NULL,
    TELEFON VARCHAR(30) NOT NULL)
AS
begin
  update KLIENT set

  klient.id_kategorii_klienta=:id_kategorii_klienta,
  klient.name_org=:name_org,
  klient.familiya_kontakta=:familiya_kontakta,
  klient.imya_kontakta=:imya_kontakta,
  klient.otch_kontakta=:otch_kontakta,
  klient.gorod_selo=:gorod_selo,
  klient.ulica=:ulica,
  klient.dom=:dom,
  klient.telefon=:telefon
  where klient.id_klient=:id_klient;
suspend;
end^


ALTER PROCEDURE UPDATE_OPLATA (
    ID_OPLATI INTEGER NOT NULL,
```

```
    ID_ZAKAZ INTEGER,
    ID_TIP_OPLATI INTEGER,
    SUMMA_OPLATI DECIMAL(15,2),
    DATA_OPLATI DATE)
AS
begin
  update oplata set

  oplata.ID_ZAKAZ=:ID_ZAKAZ,
  oplata.ID_TIP_OPLATI=:ID_TIP_OPLATI,
  oplata.SUMMA_OPLATI=:SUMMA_OPLATI,
  oplata.DATA_OPLATI=:DATA_OPLATI
    where oplata.id_oplati=:id_oplati;
suspend;
end^


ALTER PROCEDURE UPDATE_OTV_BRIG (
    ID_OTV_BRIG INTEGER NOT NULL,
    ID_RUK INTEGER,
    TELEFON VARCHAR(15),
    KOL_CHELOVEK INTEGER)
AS
begin
  update OTV_BRIG set

  OTV_BRIG.ID_RUK=:ID_RUK,
  OTV_BRIG.TELEFON=:TELEFON,
  OTV_BRIG.KOL_CHELOVEK=:KOL_CHELOVEK
    where OTV_BRIG.ID_OTV_BRIG=:ID_OTV_BRIG;
suspend;
end^


ALTER PROCEDURE UPDATE_RUK (
    ID_RUK INTEGER NOT NULL,
    FAMILIY VARCHAR(30) NOT NULL,
    IMYA VARCHAR(20) NOT NULL,
    OTCH VARCHAR(20) NOT NULL)
AS
begin
  update ruk set
  ruk.FAMILIY=:FAMILIY,
  ruk.IMYA=:IMYA,
  ruk.OTCH=:OTCH
  where ruk.id_ruk=:id_ruk;
suspend;
end^


ALTER PROCEDURE UPDATE_TIP_OPLATI (
    ID_TIP_OPLATI INTEGER NOT NULL,
    TIP_OPLATI VARCHAR(50) NOT NULL)
AS
begin
  update tip_oplati set
  tip_oplati.tip_oplati=:tip_oplati
  where tip_oplati.id_tip_oplati=:id_tip_oplati;
suspend;
end^


ALTER PROCEDURE UPDATE_USLUGI (
    ID_USLUGI INTEGER NOT NULL,
```

```
    ID_VID_USLUGI INTEGER,
    NAME_USLUGI VARCHAR(90),
    ED_IZM VARCHAR(10),
    CENA DECIMAL(15,2))
AS
begin
  update uslugi set

  uslugi.ID_VID_USLUGI=:ID_VID_USLUGI,
  uslugi.NAME_USLUGI=:NAME_USLUGI,
  uslugi.ED_IZM=:ED_IZM,
  uslugi.CENA=:CENA
    where uslugi.id_uslugi =:id_uslugi;
suspend;
end^


ALTER PROCEDURE UPDATE_VID_DOGOVORA (
    ID_VID_DOGOVORA INTEGER NOT NULL,
    VID_DOGOVORA VARCHAR(90) NOT NULL)
AS
begin
  update vid_dogovora set
  vid_dogovora.vid_dogovora=:vid_dogovora
  where vid_dogovora.id_vid_dogovora=:id_vid_dogovora;
suspend;
end^


ALTER PROCEDURE UPDATE_VID_USL (
    ID_VID_USLUGI INTEGER NOT NULL,
    VID_USLUGI VARCHAR(50) NOT NULL)
AS
begin
  update VID_USLUGI set
  VID_USLUGI.VID_USLUGI=:VID_USLUGI
  where VID_USLUGI.ID_VID_USLUGI=:ID_VID_USLUGI;
suspend;
end^


ALTER PROCEDURE UPDATE_VSTRECH (
    ID_VSTRECH INTEGER NOT NULL,
    ID_KLIENT INTEGER,
    ADRESS VARCHAR(50),
    DATA DATE,
    VREMYA_NACH TIME,
    VREMYA_KONEC TIME)
AS
begin
  update vstrechi set

  vstrechi.ID_KLIENT=:ID_KLIENT,
  vstrechi.ADRESS=:ADRESS,
  vstrechi.DATA=:DATA,
  vstrechi.VREMYA_NACH=:VREMYA_NACH,
  vstrechi.VREMYA_KONEC=:VREMYA_KONEC
    where vstrechi.ID_VSTRECH=:ID_VSTRECH;
suspend;
end^


ALTER PROCEDURE UPDATE_ZAKAZ (
    ID_ZAKAZ INTEGER NOT NULL,
```

```
    ID_USLUGI INTEGER,
    DATA_RAZMECH DATE,
    TREB_DATA DATE,
    DATA_NACH DATE)
AS
begin
  update ZAKAZ set

  ZAKAZ.ID_USLUGI=:ID_USLUGI,
  ZAKAZ.DATA_RAZMECH=:DATA_RAZMECH,
  ZAKAZ.TREB_DATA=:TREB_DATA,
  ZAKAZ.DATA_NACH=:DATA_NACH
    where ZAKAZ.ID_ZAKAZ=:ID_ZAKAZ;
suspend;
end^
```

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

« » июня 2016 г.

Студент_____ Дерипаско К.С.