МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Кафедра информатики и математики

КУРСОВАЯ РАБОТА

РАЗРАБОТКА БАЗЫ ДАННЫХ ОТДЕЛА ВНЕВЕДОМСТВЕННОЙ ОХРАНЫ КВАРТИР

Работу выполнил _____ П. С. Баран
(подпись, дата) (инициалы, фамилия)
Филиал ФГБОУ ВО «КубГУ» в г. Новороссийске курс 2 ОФО
Направление 38.03.05 «Бизнес-информатика»

Научный руководитель
Канд.физ.-мат.наук, доцент _____ С.В. Дьяченко
(подпись, дата) (инициалы, фамилия)

Нормоконтролер
Специалист по УМР _____ О.П. Гринчишина
(подпись, дата) (инициалы, фамилия)

Краснодар 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данная курсовая работа направлена на изучение и создание базы данных отдела вневедомственной охраны квартир.

Человек сталкивается с огромным количеством чисел везде – на работе и дома, в магазине и банке и так далее [8]. Для многих вычисления стали обыденностью. Однако для того, чтобы автоматизировать вычисления и представить полученный результат в понятном для обычного пользователя виде, как правило, сейчас все чаще используются электронные таблицы.

Сейчас, базы данных прочно закрепились на своем месте [13]. Они используются для решения огромного количества задач, упрощения повседневной жизни. Наиболее общими задачами для баз данных являются задачи по хранению информации, будь то списки рабочих, цены на продукты или таблица заработной платы предприятия. Также немаловажной задачей баз данных является редактирование, добавление и удаление различных данных, это позволяет оперативно поддерживать базу данных, в актуальном состоянии [16]. База данных служит для накопления, обработки и автоматизации огромного количества различных сведений.

Актуальность данной работы обусловлена тем, что каждой предприятие и организация, стремится уменьшить всевозможные затраты и сократить время необходимое для обработки поступающей информации. Именно базы данных помогают справиться с этим. Ведь именно благодаря им, теперь не нужно изучать огромный объем информации [22]. Вся информация в базе данных четко структурирована и может быть изменена самим пользователем, под его нужды.

Объектом данной работы являются данные, а также способы их организации, структурирования и систематизирования

Предмет исследования – основы создания баз данных.

Цель данной курсовой работы – изучение проектирования баз данных, а также получение практических навыков в создании собственной базы данных.

Методы исследования – изучение и анализ литературы, обобщение данных.

Задачи работы: создать рабочее клиентское приложение, которое будет иметь простой и понятный интерфейс для обычного пользователя, создание различных запросов в этой базе данных

База данных была разработана в Microsoft Access. Клиентское приложение реализовано с помощью Delphi 2007

1 Язык программирования Delphi

1.1 Языки программирования

Рассмотрим языки программирования, которые использовались при разработке клиентского приложения:

1) Delphi это язык программирования, используемый в среде с таким же названием и является сочетанием некоторых важнейших технологий:

– Компилятор, который имеет высокую производительность, в машинный код.

– Простое быстрое и визуальное создание программ из программных прототипов.

– Различные средства для создания и построения баз данных.

Изначально данный язык программирования назывался Object Pascal. Затем начиная со среды Delphi 7.0, в различных официальных документах Borland начала использование названия Delphi для обозначения языка Object Pascal.

При разработке языка программирования Delphi не ставилась задача по обеспечению наиболее максимальной производительности исполняемого кода для того что бы экономить оперативную память. С самого начала своего существования, язык ставил во главу стройность и высокую читабельность, поскольку, одной из задач данного языка являлось обучение программированию. Эта изначально заданная стройность, позволила языку в дальнейшем, при росте аппаратных мощностей, упростила расширение языка новыми функциями и конструкциями.

2) SQL (Structured Query Language - структурированный язык запросов) - это язык, предназначенный для управления базами данных для реляционных баз данных. Как таковой SQL-язык не является Тьюринг-полным языком программирования, но, однако, его стандарт дает возможность создавать для него различные процедурные расширения, которые помогают расширять его

функциональность до полноценного и самостоятельного языка программирования.

1.2 Delphi как среда программирования

Borland Delphi представляет из себя средство создания и разработки программ для Microsoft Windows. Delphi является простым и мощным в использовании инструментом для разработки и создания автономных программ, имеющих графический интерфейс (GUI), или 32-битных консольных приложений (программ, не имеющих графического интерфейса).

В сочетании с Borland Kylix, программисты, работающие на Delphi, могут разрабатывать только лишь из исходного текста приложения для Windows и Linux. Это дает больше возможностей новые, а также увеличивает потенциально возможную отдачу от усилий, которые были вложены в изучение Delphi. В Delphi применяется кроссплатформенная библиотека компонентов CLX и визуальные дизайнеры для создания высокопроизводительных приложений для Windows, которые повторной компиляций можно легко превратить в приложения для Linux.

Delphi является одним из первых языков программирования, который обладает простой и понятной в использовании средой для оперативной разработки программ и приложений. Delphi разрушает барьеры между сложными языками, находящимися на более высоком уровне, и языками, на более низком уровне, которые разговаривают с системой на языке битов и байтов.

При воссоздании графического интерфейса программ Delphi, есть все возможности и преимущества языка программирования Object Pascal, "завернутого" в среду RAD (от английского rapid application development — быстрая разработка приложений). Такие компоненты окна графического пользовательского интерфейса, как формы, списки объектов, кнопки, уже включены в состав приложения Delphi. Все это значительно упрощает работу по созданию программ и приложений. Теперь не нужно писать никакой код, чтобы

добавить их на нашу форму. Нужно просто переносим их на Форму, как в простом графическом редакторе. Также есть возможность внести на Форму элементы управления ActiveX, для создания в кратчайшие сроки специализированных приложений, например, таких, как веб-браузеры. Delphi дает возможность разработчикам дизайна вносить в интерфейс новые элементы и изменять, и кодировать их события одним щелчком мыши.

Delphi представлен в различных конфигурациях и версиях, настроенных на потребности и желания различных предприятий. В Delphi можно писать программы для Windows быстро и легко.

1.3 Интерфейс Delphi

Среда программирования в Delphi состоит из множества составных частей. Из их большого числа можно выделить самые основные:

1) Дизайнер Форм (Form Designer)

2) Окно Редактора Исходного Текста (Editor Window)

3) Палитра Компонентов (Component Palette)

4) Инспектор Объектов (Object Inspector)

5) Справочник (On-line help)

Конечно же это не все, составляющие Delphi. Есть и другие важные составляющие, например, системное меню, линейка инструментов и так далее.

Разберем каждую из этих частей более подробно.

1) Дизайнер форм. Дизайнер Форм в Delphi достаточно интуитивно понятен и прост в применении, что создание визуального интерфейса похоже на детскую игру. Дизайнер Форм изначально состоит из одного пустого окна, которое можно заполнить большим количеством всевозможных объектов, которые можно выбрать на Палитре Компонентов. Пример Дизайнера Форм показан на рисунке 1.

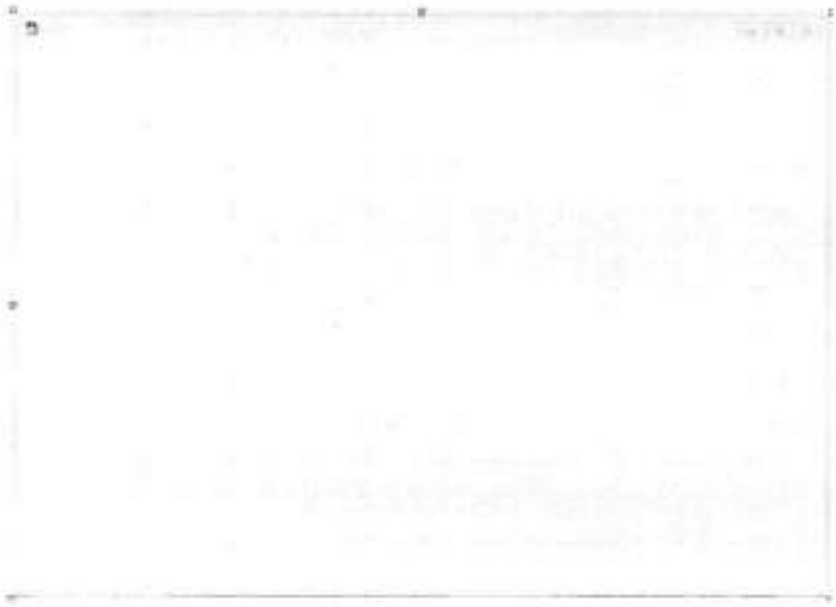

Рисунок 1 – Дизайнер форм

2) Окно редактора исходного текста. Однако несмотря на всю важность Дизайнера Форм, основным местом, где программисты проводят большую часть времени, является Редактор. Логика является движущей силой программы и Редактор является тем самым местом, где программист ее «кодирует». Пример окна редактора исходного текста представлен на рисунке 2.

Рисунок 2 – Окно редактора исходного текста

3) Палитра Компонентов. Палитра Компонентов дает возможность программисту выбрать необходимые объекты для размещения их на Дизайнере Форм. Для использования Палитры Компонентов нужно лишь выбрать необходимый элемент для работы и перенести его на Дизайнер Форм или же просто первый раз щелкнуть мышкой на него и потом второй раз - на Дизайнере Форм. Выбранный объект появится на проектируемом окне и его можно изменять и редактировать с помощью мыши. Пример Палитры Компонентов представлен на рисунке 3.

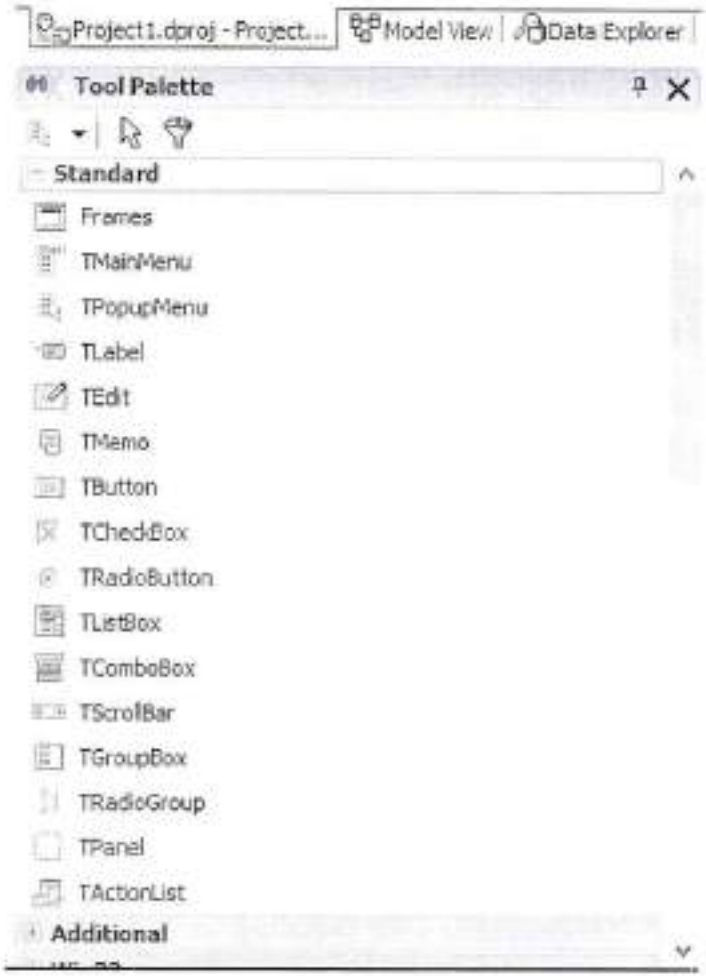

Рисунок 3 – Палитра компонентов

Палитра Компонентов использует постраничную группировку объектов. Внизу Палитры находится множество наборов закладок - Standard, Additional, Dialogs и т.д.

4) Инспектор Объектов. Слева от Дизайнера Форм находится Инспектор Объектов. Информация в нем меняется в зависимости от объекта, который выбирается на форме. Важно понимать, что каждый компонент является настоящим объектом из этого следует то что можно менять его вид и поведение с помощью Инспектора Объектов.

Инспектор Объектов состоит из двух разделов. Каждый из этих разделов возможно применять для определения поведения этого компонента. Первая страница этого раздела - это список свойств, вторая страница — это список событий. Если необходимо редактировать что-либо, связанное с определенным компонентом, то это обычно делается через Инспектора Объектов. Например, нужно изменить размер и имя компонента TLabel изменяя свойства Caption, Left, Top, Height, и Width и так далее. Пример Инспектора Объектов представлен на рисунке 4.

Рисунок 4 – Инспектор Объектов

Вторая страница — это страница событий. Она напрямую связана с Редактором; если дважды щелкнуть мышкой на правую сторону какого-нибудь подпункта, то код, который соответствует данному событию автоматически, запишется в Редактор, в этот момент, Редактор получит фокус, и сразу же появится возможность добавить код обработчика данного события.

5) Справочник. Последняя, наиболее важная часть среды Delphi – это Справочник. Для того, чтобы получить доступ к данному инструменту необходимо выбрать в системном меню пункт Help и затем Contents. Тогда на экране появится необходимый нам справочник. Пример справочника показан на рисунке 5.

Рисунок 5 – Справочник

Справочник контекстно-зависим. Это означает то, что при нажатии клавиши F1 будет дана подсказка, которая соответствует текущему состоянию. Например, если мы, находясь в Инспекторе Объектов, выберем какое-либо свойство и нажмем F1, то мы получим справку о назначении данного свойства.

1.4 База данных в Delphi

Для того что бы подключить к Delphi базу данных изначально нужно ее создать в Microsoft Office Access. После того как наша база данных будет готова можно приступать к подключению.

В среде Delphi имеется большой набор инструментов, который поможет создать клиентское приложение и связать его с любой базой данных. Основные инструменты для баз данных будут представлены ниже [4]:

– TDBGrid – данный компонент предназначен для вывода информации из базы данных, в нашем приложении

– TDataSource – данный компонент предназначен для связи нашей сетки отображения данных, с самой базой данных.

– TADOConnection – данный компонент предназначен для подключения нашей базы данных к приложению.

– TADOQuery – данный компонент предназначен для создания запросов.

– TADOTable – данный компонент вступает в контакт с указанной нами таблицей базы данных.

Что бы связать нашу базу данных с приложением необходимо в компоненте TADOConnection выбрать свойство ConnectionString. После того как необходимый компонент будет выбран, откроется окно, которое показано на рисунке 6.

Рисунок 6 – Свойство ConnectionString

Для того что бы связать нашу базу данных, необходимо указать путь, к нашей базе данных, созданной в Microsoft Office Access. Для этого нам необходимо нажать на клавишу «Build…», после чего откроет окно, представленное на рисунке 7.

Рисунок 7 – Свойства канала передачи данных

В текущем окне необходимо выбрать свойство Microsoft Jet 4.0 OLE DB PROVIDER. После того как необходимое свойство выбрано, нужно установить соединение с заранее созданной базой данных. Это делает в следующем окне, представленном на рисунке 8.

Рисунок 7 – Подключение базы данных

При нажатии на клавишу «Проверить соединение» будет проведена проверка соединения приложения с выбранной базой данных. Если подклю-

чение выполнено верно программа выдаст окно с надписью: «Проверка подключения выполнена».

После переноса всех необходимых компонентов на формы можно приступать к созданию, изменению, редактированию базы данных.

Компонент TADOQuery – аналог компонента TQuery, который используется при работе с BGE [5]. TADOQuery применяется для создания и выполнения SQL запросов. Основное свойство этого компонента – SQL, которое содержит методы выполнения запроса и, непосредственно, сам запрос. TADOQuery точно так же, как и TQuery имеет свойство DataSource, которое, позволяет ему передать параметры запроса от одного компонента другому.

2 Microsoft Access как средство для разработки баз данных

2.1 Базовые понятия, связанные с Microsoft Access

Приложение, разработанное компанией Microsoft, под названием Access относится к приложениям, предназначенным для создания, редактирования и управления данными. Ключевыми особенностями данной программы являются простота в ее освоении и интуитивно понятная рабочая среда [3].

Разница между Access и другими системами управления базой данных (далее СУБД) заключается в том, что Access сохраняет все необходимые данные в одном исполняемом файле, однако он продолжает распределять их по различным таблицам, как и положено реляционной СУБД [1]. К этим данным относится не только информация в таблицах, но и другие объекты базы данных, которые будут описаны ниже.

Немаловажным плюсом является то, что большая часть важных операций в Access может быть выполнена при незначительных усилиях, приложенных человеком. Помогают в этом «Мастера»: их огромное количество значительно экономит время рядовому пользователю, особенно тем, кто не знаком с программированием [20].

Доступ к базе данных созданной в Access может получить сразу несколько пользователей: либо через файловый сервер, либо с помощью локальной одноранговой сети [2]. Сама сеть отвечает за поддержку аппаратного характера и программными средствами обеспечивает обмен данными между компьютерами. Access контролирует границы доступа различных пользователей, тем самым обеспечивая защищённость данных. Присутствуют определённые ограничения по осуществлению многопользовательской работы с Access по причине того, что данная СУБД не является серверной. Как правило, на файловый сервер загружается файл Access, с присущим ему расширением *.mdb, чтобы другие пользователи могли удалённо получить доступ к данным, используя сеть. Однако обработка данных происходит непосредственно

там, где запущено приложение, то есть на основном клиенте, что резко ограничивает круг пользователей до двадцати человек, а при особенно больших объёмах данных, содержащихся в таблицах – и вовсе до пятнадцати, поскольку нагрузка на сеть значительно возрастает [18].

Если рассматривать Access со стороны поддержания целостности данных, то он соответствует моделям базы данных средней сложности. Отсутствие таких средств как, например, триггеры и хранимые процедуры вынуждает разработчиков перекладывать ответственность за поддержание бизнес логики на клиентское приложение [21].

Если рассматривать Access с точки зрения защищенности, то можно смело сказать, что Access не обладает достаточным уровнем защиты. Здесь присутствует лишь простая защита с применением пароля, который может установить пользователь, однако данный барьер не остановит компетентного в данном вопросе специалиста.

При этом, несмотря на все перечисленные недостатки у Microsoft Access имеется ряд неоспоримых преимуществ по сравнению подобными СУБД.

Одно из немаловажных достоинств Access это доступность. Эти он обязан своей принадлежностью корпорации Microsoft. В свою очередь, Microsoft предоставляет операционную систему и программное обеспечение внушительной части пользователей персональных компьютеров. Полная совместимость с операционной системой Windows, регулярные обновления и поддержка со стороны производителя, возможность импорта и экспорта данных в формат Excel, – другой популярной программы от Microsoft, – а также поддержка множества языков являются слагаемыми популярности Access.

Программой Access может воспользоваться совершенно любой пользователь: люди с совершенно разной профессиональной подготовкой могут справиться с поставленной задачей, благодаря наличию ранее упомянутых «Мастеров», развитой системе справок и доступному для человека с базовым уровнем владения персональным компьютером интерфейсу [9]. Всё это ощу-

тимо ускоряет процесс проектирования и создания базы данных, а также облегчает выборку данных из неё.

Microsoft Access предоставляет возможность обойтись без разработки запросов на языке SQL и не прибегать к языку VBA для программирования модулей или макросов, предлагая взамен самые разные диалоговые средства.

В целом, в Microsoft Access достаточно сильно развиты встроенные средства разработки приложений. Основная масса приложений, которые распространяются среди пользователей, в том или ином объёме содержит код VBA (Visual Basic for Applications). Построение команд SQL в процессе работы программы, использование Windows API, работа с переменными, обработка ошибок – все эти и многие другие стандартные задачи выполняются в Access при помощи VBA.

Язык макрокоманд – одно из средств программирования в Access. Макросы — это программы, созданные на данном языке [15]. Они используются для построения связей между определёнными действиями, которые реализуются при помощи форм, запросов и отчётов. Управление макросами осуществляется путем работы с диалоговыми формами, которая влияет на макросы системными событиями.

Можно сделать вывод о том, что, хотя Access имеет черты свойственные СУБД, однако он выходит за пределы границ ее возможностей. Access дает возможность создавать приложения, которые будут работать с базами данных, но не создает лишних трудностей связанных с освоение и использованием СУБД.

2.2 Основные функции Microsoft Access

В данном разделе можно ознакомится с основными функциями Microsoft Access, дабы прояснить аспекты, которые связаны с его возможностями.

В Access 2016 имеются следующие типы объектов:

– Таблица.

– Запрос.

– Форма.

– Отчет.

– Макрос.

– Модуль.

Пример данных объектов представлен на рисунке 8.

Рисунок 8 – Окно объектов базы данных

Сам по себе Access не может работать с несколькими базами данных одновременно, но одна БД включает в себя большое количество форм, таблиц, отчетов, макросов, запросов и модулей. Все это хранится в определенном файле, имеющем расширение mdb [25].

В объекте базы данных под названием «Таблица» данные располагаются в виде записей и полей (строк и столбцов). Это главный структурный элемент в системе управления реляционной базой данных. Для ускорения досту-

па к данным в каждой таблице задаётся первичный ключ, который задаётся по нажатию на кнопку «Ключевое поле», представленную на рисунке 9.

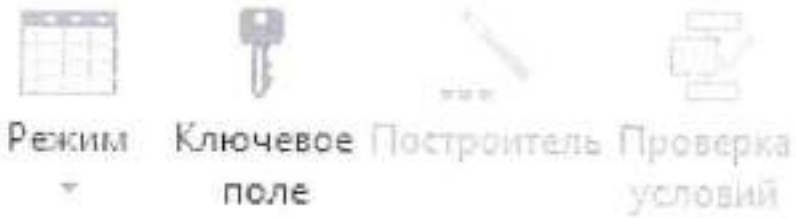

Рисунок 9 – Ключевое поле

Для того, чтобы задать ключевое поле, необходимо для начала создать структуру самой таблицы. Сделать это можно тремя способами:

– Войти в режим конструктора.

– Выбрать один из предложенных шаблонов.

– Ввести данные в пустую таблицу.

Выбор в пользу того или иного способа напрямую связан с целью и уровнем подготовки пользователя, а также с планами по использованию данных. Рассмотрим подробнее каждый из имеющихся вариантов [24]:

1) Режим конструктора. Чтобы воспользоваться данным режимом, нужно лишь перейти во вкладку «Создание» и в группе «Таблицы» выбрать «Конструктор таблиц». В открывшемся окне можно задать имя поля под соответствующей надписью и присвоить ему определённый тип данных. Полный инструментарий режима «Конструктора» представлен на рисунке 10.

Рисунок 10 – Конструктор таблиц

2) Шаблоны. Если пользователь ограничен по времени и слабо владеет персональным компьютером, он может воспользоваться данными заготовками. В Microsoft Access присутствует огромный набор уже заранее подготовленных шаблонов и таблиц с полями, которые готовы к заполнению. Чтобы ими воспользоваться, необходимо перейти в меню «Файл» и выбрать пункт «Создать», после чего откроется выбор всех доступных шаблонов. Это продемонстрировано на рисунке 11.

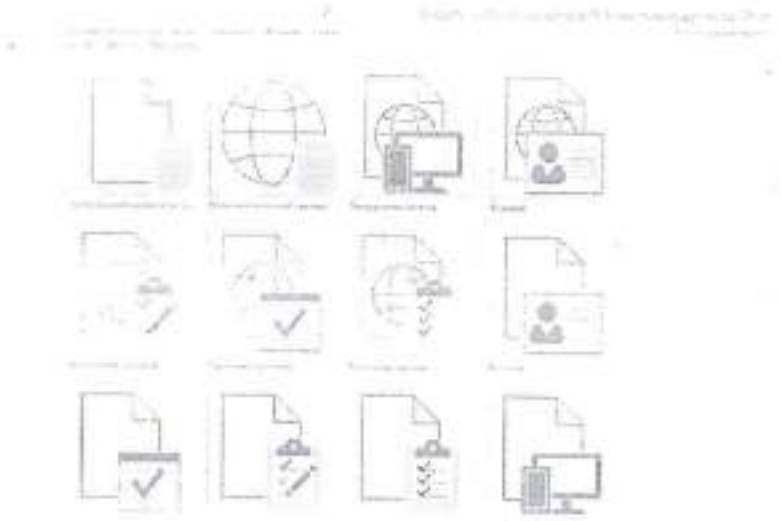

Рисунок 11 – Шаблоны

3) Пустая таблица. Для создания пустой таблицы, нужно лишь во вкладке «Создание» выбрать «Таблица» в соответствующей группе. Всё, что далее требуется от пользователя, это ввести названия полей и перейти к их заполнению, а Access сам определит тип данных в зависимости от содержимого, что наглядно продемонстрировано на рисунке 12.

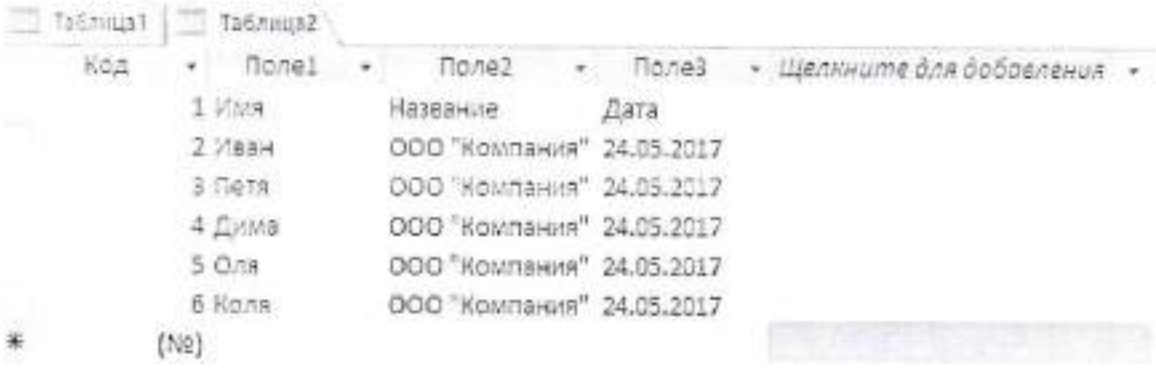

Рисунок 12 –«Пустая таблица»

Говоря о типах данных, необходимо раскрыть суть основных из них:

1) Текстовый. В поле содержится текст с количеством символов до 255.

2) Поле Memo. Предназначен для содержания длинных блоков текста, например, описания чего-либо. В нём может помещаться до 64000 символов.

3) Числовой. Поле, содержащее числовое значение.

4) Дата/ время. В данном поле содержится дата и время, начиная с 100, и заканчивая 9999 годом.

5) Денежный. Поле, в котором обозначаются валюты. Можно записать 15 знаков до и 4 знака после запятой.

6) Счётчик. Производится автоматическая вставка последовательных или случайных чисел по мере добавления новых записей.

7) Логический. Здесь содержатся значения «Да», «Нет», а также прочие поля, содержащие лишь одно истинное из двух возможных значений.

8) Поле объекта OLE. Позволяет хранить данные разных типов, причём одновременно, что устраняет главное ограничение реляционных баз данных – невозможность хранить данные сразу нескольких типов в одном поле.

9) Гиперссылка. В нём может храниться как текст, так и его комбинация с числами, сохранённая как текст. Используется как контейнер для хранения пути к какому-либо объекту [10].

В программе Access изначально заложена возможность добавления новых записей, а также их редактирования, удаления и просмотра. Также можно сортировать данные и видоизменять таблицы. Представить все имеющиеся связи в графическом виде позволяет средство под названием «Схема данных», который можно увидеть на рисунке 13.

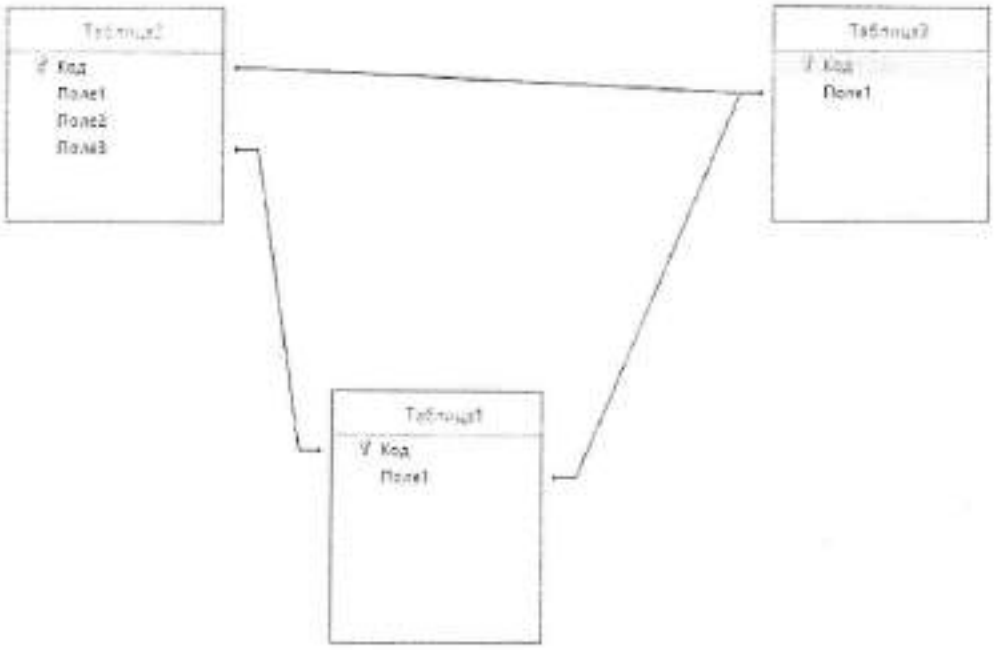

Рисунок 13 – Схема данных

Под запросом понимается объект, в котором находится текст SQL-запроса, который содержит индивидуальное имя в определённой базе данных. Создать запрос, воспользовавшись «Мастером запросов» или «Конструктором запросов» – решение целиком и полностью зависящее от пользователя. Если сделан выбор в пользу первого варианта, то после нажатия на соответствующую иконку пользователю будут предложены несколько типов запросов: простой, перекрёстный, на повторяющиеся записи, а также на записи без подчинённых. И если с первым запросом всё понятно, то оставшимся трём требуется дать определение [12].

Перекрестный запрос. В данном запрос исходные данные обрабатываются статистическим образом, а затем результат выводится в виде таблицы, которая, по сути, являет собой сводную таблицу данных из Excel.

Чтобы избежать вывода одинаковых, повторяющихся записей можно произвести поиск дубликатов, чему и способствует запрос на повторяющиеся записи. Вывести запрос, который покажет, например, всех клиентов, оставшихся без заказа, поможет запрос на записи без подчинённых.

Во всех представленных случаях от пользователя требуется лишь выбрать таблицу и поле для выборки данных в ней. Поскольку интерфейс во всех представленных случаях почти одинаков, в качестве примера показано создание только простого запроса в режиме «Мастер запросов» на рисунке 14.

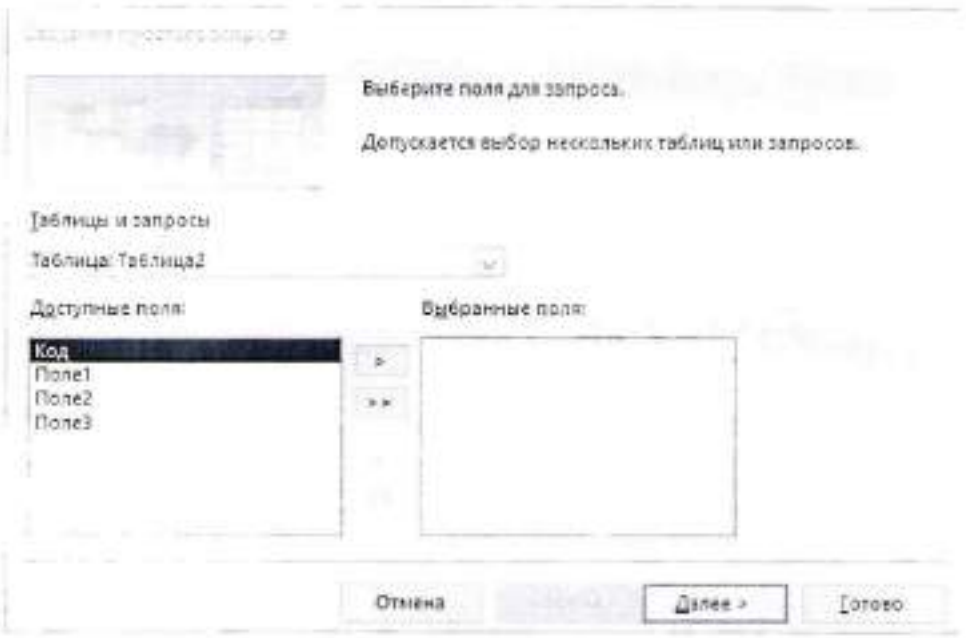

Рисунок 14 –Мастера запросов

«Конструктор запросов» дает пользователю возможность не только выбирать таблицы, связывать их графическим способом и определять поля, по которым будут выбираться данные, но и задавать дополнительные условия и параметры сортировки для каждого из этих полей. Все эти возможности отражены на рисунке 15.

Рисунок 15 –Конструктора запросов

Форма – особый контейнер для таких компонентов, связанных с интерфейсом, как поле для ввода данных, кнопки, поле для отображения данных из таблиц и прочее. В зависимости от разрабатываемого приложения, разработчик волен расположить на форме компоненты, позволяющие просматривать, вводить, исправлять и группировать данные. Пример работы с формой представлен на рисунке 16.

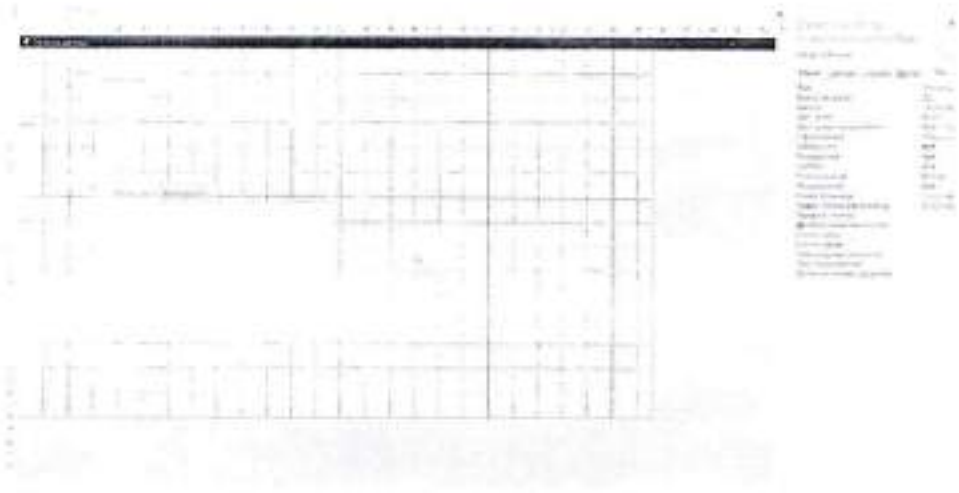

Рисунок 16 – Конструктор форм

Отчёт – объект, который применяется в случае, когда требуется произвести выборку из базы данных в структурированном виде, то есть оформить как таблицу или список, а затем создать из этого документ, который можно было бы распечатать или включить в документ другого приложения.

Страницы – инструмент для публикации данных в интернете или же в локальной сети. Создание страницы аналогично созданию формы, отличаются только используемые компоненты. Основное отличие при работе заключается в редактировании данных в базе, помимо возможности их просматривать. После сохранения страницы в качестве объекта базы данных возможен её экспорт в виде файла формата HTML, что позволяет получить доступ к данным, используя интернет-браузер.

Макрос – объект, который представляет собой упорядоченный набор макрокоманд с целью автоматизировать те действия, которые выполняются очень часто при работе с базой. Макрокоманду можно выбрать в предложенном списке, в то время как параметры задаёт разработчик.

Модуль – контейнер программного кода, который написан на языке VBA. Редактор Visual Basic используется для просмотра и редактирования кода, который находится в наборе модулей.

3 Разработка клиентского приложения

3.1 Разработка базы данных в приложении Microsoft Access

Чтобы создать базу данных для охранного предприятия, для начала нужно определиться с той информацией, которая будет отслеживаться в нашей БД. Так как это коммерческая организация, то для предприятия, немаловажной будет информация об имеющихся договорах. Это будет самая объемная таблица, так как в ней будет собрана вся информация о клиентах и их заказах. Далее нам необходима информация о сотрудниках, работающих в компании. В данной таблице будет собрана вся информация о них. С таблицей сотрудников будут связаны таблицы Районов, за которым закреплен каждый сотрудник и таблица Бригад. Затем нам необходима таблица с информацией о клиентах, дабы иметь необходимую информацию о клиентах, заключающих договоры. Так же стоит отметить таблицу, в которой будет собрана информация о ценах, за каждую из услуг, предоставляемых охранным предприятием.

Как уже было сказано раннее, основной таблицей для коммерческого предприятия может являться таблица договоров именно поэтому она обладает наибольшим количеством информации и именно поэтому имеет смысл начать описание с нее [6].

1) В таблице под названием «Договор» будет собрана ключевая информация из других таблиц. Пример данной таблице будет представлен на рисунке 17.

Рисунок 17 – Договор

Почти каждый столбец данной таблицы, является связующим звеном с другими таблицами этой базы данных. Например, столбец «IDKlienta» указывает на связь с таблицей «Клиенты». Этот столбец в дальнейшем даст нам информацию о каждом из клиентов, которые заключали договора. Более детально, связь каждого столбца с другими таблицами будет показана в теме, где будет рассматриваться связь.

Так же стоит отметить что для определенных полей используются определенные типы данных. Например, для столбца, в котором храниться информация о дате заключения договора, используется специальный тип данных – «Дата и время». Для столбца, в котором храниться информация о сумме заказа используется специальный тип данных – «Денежный» и так далее. Использование различных типов данных, помогает простому пользователю, без лишних усилий создавать различные поля, для различных целей.

2) Следующая таблица не является такой объемной, однако ее важность крайне высока. Это таблица под названием «Клиенты». Пример данной таблицы представлен на рисунке 18.

Рисунок 18 – Клиенты

В данной таблице виден пофамильный список всех клиентов компании, а также их номера телефонов [7]. Ключевым полем для связи с другими таблицами является поле под названием «IdKlienta».

3) Следующая таблица, которая напрямую связана с главной таблицей договоров является таблица цен на услуги, предоставляемые предприятием. Называется данная таблица «Прайс». Пример таблицы представлен на рисунке 19.

Рисунок 19 – Прайс

Хотя таблица не обладает большим количеством информации, она также является очень важной, потому что с помощью этой таблицы, предприятие может вести учет предоставляемых услуг и подсчитывать их конкретную стоимость. Ключевым полем для связи с другими таблицами является поле под названием «IDPrice».

4) Далее идет еще одна немаловажная таблица, в которой содержится информация о сотрудниках, работающих в компании. Пример этой таблицы представлен на рисунке 20.

Рисунок 20 – Сотрудники

Помимо основной информации о каждом из сотрудников, можно видеть два столбца, которые являются связующими звеньями с другими таблицами. Столбец «IDdoljnost» даст нам информацию о должности каждого сотрудника, а столбец «IDBrigada» даст нам информацию о бригаде, в которой находится каждый из сотрудников. Два этих столбца значительно ускорят процесс сортировки информации и упростят ее поиск.

5) Оставшиеся таблицы будут собраны в одном подпункте, из-за своего малого объема. Обе эти таблицы служат для дополнения информации о работе сотрудников. Пример первой таблицы под названием «Бригада» будет представлен на рисунке 21.

Рисунок 21 – Бригада

В этой таблице представлена информация о каждой из четырех бригад. За каждой бригадой запределен определенный сотрудник. В свою очередь каждая бригада закреплена за определенным районом. Эту связь можно отслеживать благодаря отдельному столбцу «IDRayona». Сама таблица Районов представлена на рисунке 22.

Рисунок 22 – Район

Смысл данной таблицы заключается в том, чтобы дополнить информацию по каждой бригаде.

3.2 Связи между таблицами

В реляционной базе данных связи дают возможность избегать переизбытка данных. Например, при создании БД, в которой будет содержаться информация о книгах, появится таблица с названием «Книги». В этой таблице будет находиться полная информация о каждой книге, такая как название, дата публикации и издатель. Кроме всего этого, есть и другие дополнительные сведения об издателе, которые может потребоваться сохранить, такие как его

адрес, почтовый индекс и номер телефона. И если хранить эту информацию в таблице с книгами, то, например, телефонный номер издателя, будет повторятся в каждой опубликованной им книге. Для того чтобы не потерять синхронизацию, нужно обеспечить целостность данных между этими двумя таблицами. Связь с обеспечением сохранения целостности позволит следить за тем, чтобы данные в разных таблицах не смешивались и соответствовали данным в другой таблице.

Чтобы создать связь между таблицами нужно закрыть все открытые таблицы и перейти во вкладку «Схема данных». Затем вынести все необходимые таблицы на рабочее поле и установить необходимые связи. На рисунке 23 показаны все связи, которые имеются в данной базе данных.

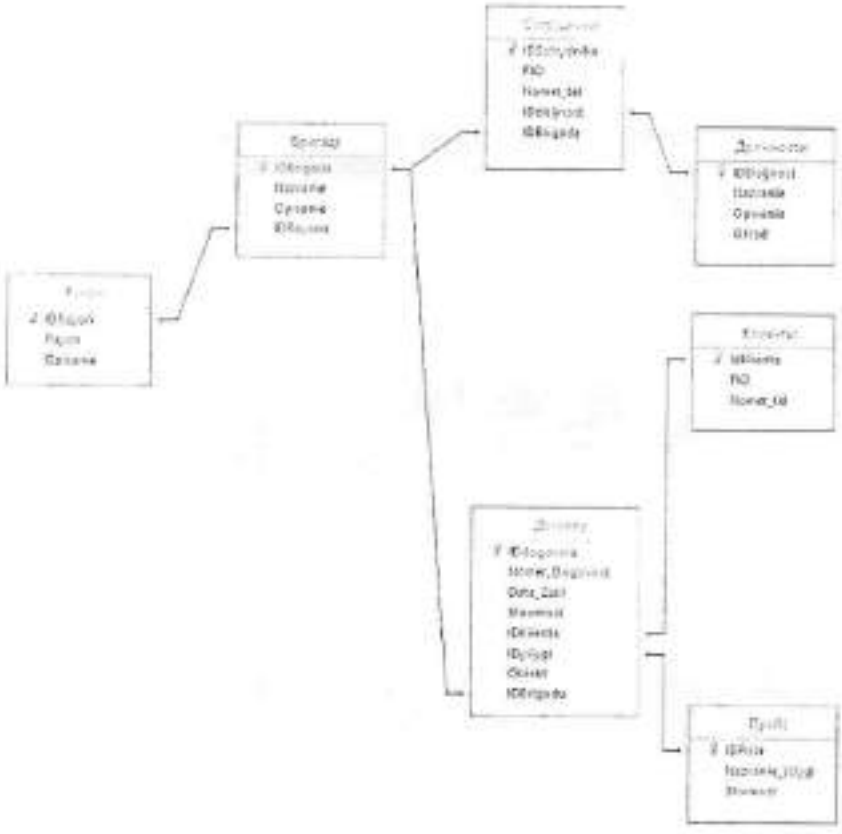

Рисунок 23 – Связи

Благодаря данному рисунку, можно наглядно увидеть все связи.

3.3 Создание клиентского приложения

Клиентское приложение будет создаваться в среде программирования Delphi. Данная среда была выбрана по ряду причины: простота освоения, большое количество готовых инструментов, множество функций необходимых для создания рабочей базы данных.

После того как база данных будет создана в Microsoft Access ее можно переносить в Delphi. Процедура переноса была описана в предыдущих главах и проиллюстрирована соответствующими рисунками.

Для создания клиентского приложения нам понадобится форма под названием «DataModule». На данной форме можно разместить необходимые инструменты, которые понадобятся нам при создании приложения. Пример этой формы представлен на рисунке 24.

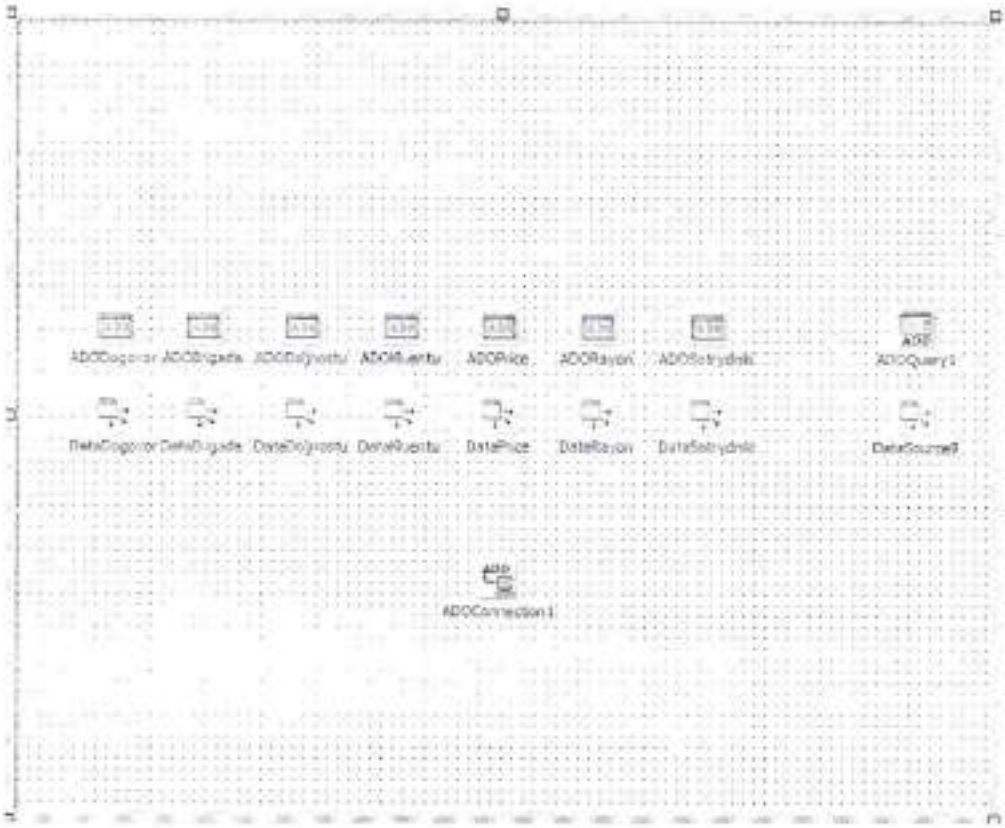

Рисунок 24 – Форма DataModule

Самым первым и одним из главных инструментов на данной форме является «ADOConnection» [23]. Благодаря этому инструменту будет устанавливаться связь между базой данных, которая была создана в Microsoft Access и между Delphi. Затем нам необходимо вынести инструменты «ADOConection» и «DataSource». Эти инструменты нужны для того что бы вывести каждую таблицу отдельно, на новой форме. Чтобы выводить таблицы из Access в Delphi, на новую форму необходимо добавить такой инструмент как «TDBGrid». Если необходимо, то на форму, так же добавляется инструмент «TDBNavigator» – он служит для редактирования, удаления и добавления новых записей в таблицу. Инструмент под названием ADOQuery необходим для создания запросов. Более подробно запросы будут рассмотрены позже.

После того как все необходимые инструменты будут добавлены на форму «DataModule» можно приступать к созданию таблиц на формах при помощи «TDBGrid», «TDBNavigator» и других. Помимо «TDBGrid» и «TDBNavigator» на главной форме будет располагаться инструмент «MainMenu». Он необходим для создания интерфейса для клиентского приложения. Пример интерфейса, который был создан с помощью данного инструмента, представлен на рисунке 25.

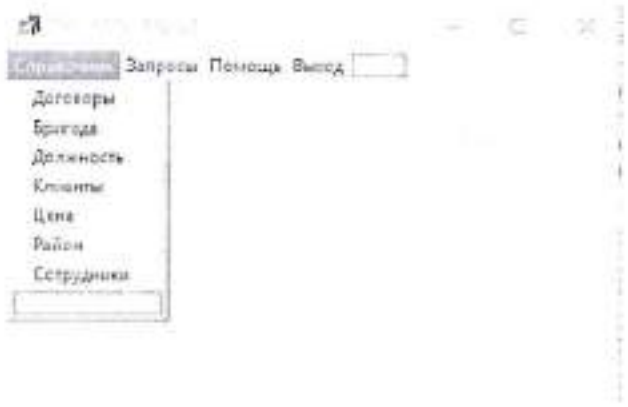

Рисунок 24 – MainMenu

Вкладка «Помощь» в интерфейсе, содержит необходимую информацию для пользователя, если он вдруг столкнется с непредвиденной ситуацией.

Вкладка «Выход» завершит работу клиентского приложения [19].

Когда работы с основной таблицей закончены, можно приступать к созданию других, второстепенных таблиц. Пример одной из форм будет показан на рисунке 25.

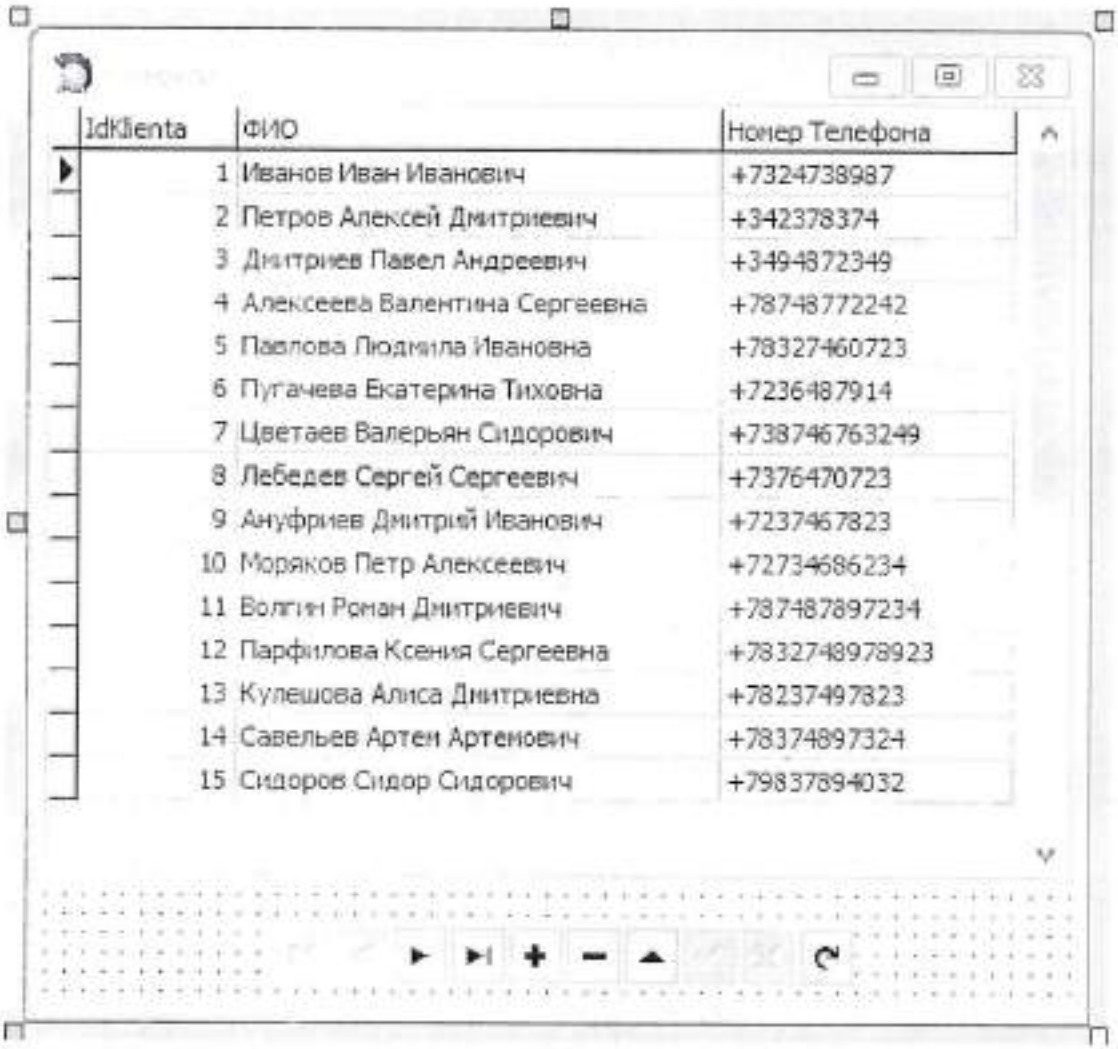

Рисунок 25 – Пример формы

Все эти таблицы имеют одинаковый вид: на форме располагается два инструмента «TDBGrid» и «TDBNavigator», поэтому не имеет смысла показывать каждую форму отдельно.

Редактирование каждой формы производится вручную и подгоняется под определенный размер каждой таблицы.

3.4 Создание запросов в клиентском приложении

Запросы являются важной частью базы данных [17]. Они позволяют ускорить обработку информации. При помощи различных запросов, пользователь затратит намного меньше времени для поиска и обработки информации. Так же при помощи запросов можно обновлять, удалять и добавлять новые записи в таблицах.

Запросы представляют собой команды, которые обращаются к базе данных и сообщают ей, что необходимо отобразить определенную информацию из той или иной таблицы [14]. В данной работе все запросы были написаны при помощи языка SQL [11].

Далее будут подробно разобраны все запросы, которые используются в данной работе:

1) Запрос с применением «ComboBox» для вывода информации по договорам. Результат запроса представлен на рисунке 26.

Рисунок 26 – Запрос 1

Описание запроса:

```
procedure TForm9.ComboBox1Change(Sender: TObject);
var str1:string;
begin
dbgrid1.Visible:=true;
datamodule2.ADOQuery1.close;
datamodule2.ADOQuery1.sql.clear;
str1:=combobox1.Items.Strings[combobox1.ItemIndex];
datamodule2.ADOQuery1.sql.Add ('Select Data_Zakl, Obiekt from Договор
where Договор.Nomer_Dogovora=:p2');
datamodule2.adoquery1.Parameters.ParamValues['p2']:=str1;
datamodule2.adoquery1.Open;
```

Код для динамического заполнения ComboBox:

```
procedure TForm9.FormShow(Sender: TObject);
var nf:string;
begin
datamodule2.ADODogovor.First;
combobox1.Clear;
datamodule2.adoprice.MasterSource:=nil;
while not datamodule2.ADODogovor.Eof do  begin
nf:=datamodule2.ADODogovor.FieldValues['Nomer_Dogovora'];
combobox1.Items.Add(nf);
datamodule2.ADODogovor.next;
combobox1.ItemIndex:=0;
dbgrid1.Visible:=false;
```

Данный запрос позволит пользователю выбрать номер договора и проверить его дату заключения, и объект на котором работает предприятие по данному договору.

2) Запрос с применением «ComboBox» для вывода информации по видам услуг. Результат запроса представлен на рисунке 27

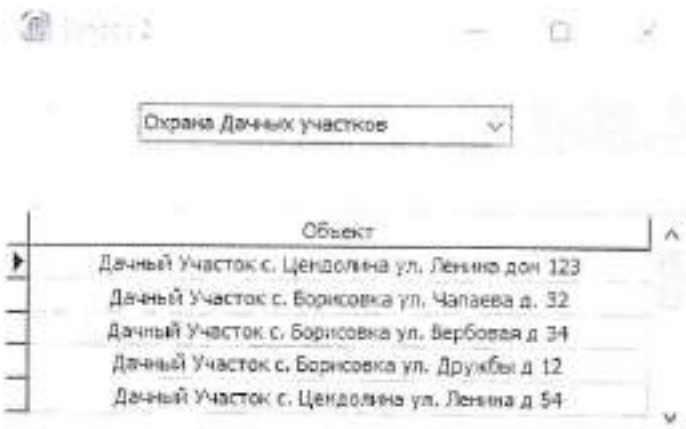

Рисунок 27 – Запрос 2

Описание запроса:

```
procedure TForm10.ComboBox1Change(Sender: TObject);
var str1:string;
begin
dbgrid1.Visible:=true;
datamodule2.ADOQuery1.close;
datamodule2.ADOQuery1.sql.clear;
str1:=combobox1.Items.Strings[combobox1.ItemIndex];
datamodule2.ADOQuery1.sql.Add ('Select Прайс.Nazvanie_Yslygi,Договор.Obiekt from Прайс, Договор where Прайс.IDPrice=Договор.IDyslygi and Прайс.Nazvanie_Yslygi=:p2');
datamodule2.adoquery1.Parameters.ParamValues['p2']:=str1;
datamodule2.adoquery1.Open;
```

Код для динамического заполнения ComboBox:

```
procedure TForm10.FormShow(Sender: TObject);
var nf:string;
begin
datamodule2.ADOPrice.First;
combobox1.Clear;
datamodule2.adoprice.MasterSource:=nil;
```

```
while not datamodule2.ADOPrice.Eof do  begin
nf:=datamodule2.ADOPrice.FieldValues['Nazvanie_Yslygi'];
combobox1.Items.Add(nf);
datamodule2.ADOPrice.next;
combobox1.ItemIndex:=0;
combobox1.itemindex:=0;
dbgrid1.Visible:=false;
```

Данный запрос позволит пользователю выбрать один из трех видов услуг, предоставляемых организацией и увидеть список объектов и адресов на которых используется одна из услуг, выбранных пользователем.

3) Запрос с использованием команд «SUM», «MIN», «MAX» и «AVG». Результат запроса показан на рисунке 28.

Рисунок 28 – Запрос 3

Описание запроса:

```
procedure TForm1.N12Click(Sender: TObject);
begin
Datamodule2.adoquery1.close;
datamodule2.adoquery1.sql.clear;
datamodule2.adoquery1.sql.Add('SELECT SUM (Stoumost) as
Сумма,MIN(Stoumost)as Минимальная,MAX(Stoumost)as
Максимальная,AVG(Stoumost)as Средняя FROM Договор');
```

Datamodule2.adoquery1.open;

С помощью данного запроса, пользователь сможет отслеживать все информацию, связанную со счетами по договорам.

4) Запрос с применением «ComboBox» для вывода информации о бригадах, задействованных на объектах. Результат запроса представлен на рисунке 29.

Рисунок 29 – Запрос 4

Описание запроса:

```
procedure TForm12.ComboBox1Change(Sender: TObject);
var str1:string;
begin
datamodule2.ADOQuery1.close;
datamodule2.ADOQuery1.sql.clear;
str1:=combobox1.Items.Strings[combobox1.ItemIndex];
datamodule2.ADOQuery1.sql.Add ('Select Бригада.Nazvanie,Договор.Obiekt from Бригада, Договор where Бригада.IDBrigada=Договор.IDBrigadu and Бригада.Nazvanie=:p2');
datamodule2.adoquery1.Parameters.ParamValues['p2']:=str1;
datamodule2.adoquery1.Open;
```

Код для динамического заполнения ComboBox:

```
procedure TForm12.FormShow(Sender: TObject);
```

```
var nf:string;
begin
datamodule2.ADOBrigada.First;  // указатель на первую строку
combobox1.Clear;
datamodule2.adoBrigada.MasterSource:=nil;  // Разрываем связь
while not datamodule2.ADOBrigada.Eof do  begin
nf:=datamodule2.ADOBrigada.FieldValues['Nazvanie'];
combobox1.Items.Add(nf);
datamodule2.ADOBrigada.next;
combobox1.ItemIndex:=0;
```

При помощи данного запроса пользователь сможет отслеживать занятость каждой бригады по отдельности. Благодаря этому запросу пользователь быстро сможет определить какая из бригад менее загружена, а какая наоборот перегружена.

5) Запрос для вывода полного списка сотрудников, их должностей и бригады, в которой они закреплены. Результат этого запроса представлен на рисунке 30.

ФИО	Должность	Бригада
Сергеев Дмитрий Алексеевич	Начальник бригады	Первая Бригада
Алексеев Сергей Владимирович	Охранник	Первая Бригада
Иванов Петр Алексеевич	Водитель	Первая Бригада
Семенов Семен Семенович	Начальник бригады	Вторая Бригада
Петров Петр Петрович	Охранник	Вторая Бригада
Иванчик Дмитрий Павлович	Водитель	Вторая Бригада
Сергеев Сергей Сергеевич	Начальник бригады	Третья Бригада
Иванов Павел Павлович	Охранник	Третья Бригада
Павлов Павел Иванович	Водитель	Третья Бригада
Баран Павел Сергеевич	Начальник бригады	Четвертая Бригада
Голубов Евгений Игоревич	Охранник	Четвертая Бригада

Рисунок 30 – Запрос 5

Описание запроса:

```
procedure TForm1.N14Click(Sender: TObject);
```

```
begin
datamodule2.ADOQuery1.close;
datamodule2.ADOQuery1.sql.clear;
datamodule2.ADOQuery1.sql.Add ('Select Сотрудники.FIO,
Должности.Nazvanie, Бригада.Nazvanie from Сотрудники, Должности, Брига-
да where Сотрудники.IDdoljnost=Должности.IDDoljnost and
Сотрудники.IDBrigada=Бригада.IDBrigada');
datamodule2.adoquery1.Open;
```

С помощью этого запроса пользователь в кратчайшие сроки сможет узнать всю необходимую информацию, связанную со своими сотрудниками.

ЗАКЛЮЧЕНИЕ

Результатом данной курсовой работы, является создание рабочего клиентского приложения, с простым и доступным интерфейсом для обычного пользователя. Также был проведен анализ поставленной задачи и последовательное её решение. Для разработки данной базы был использован Delphi 2007.

В процессе разработки клиентского приложения:

1) Было создано рабочее клиентское приложение.

2) Был создан простой и интуитивно понятный интерфейс.

3) Была изучена информационная база, необходимая для создания БД.

4) Были созданы необходимы запросы, которые подходят именно к нашей базе данных.

5) Все таблицы были заполнены необходимыми данными.

Сейчас почти ни одно предприятие не может обойтись без баз данных. В процессе выполнения курсовой работы, были изучены основы создания баз данных, основы создания запросов для этих баз данных, а также основные системы для управления ими. Базы данных жизненно необходимы предприятиям, так как каждое предприятие стремиться максимизировать свою прибыли и быть конкурентоспособным на рынке.

Интеграция различных информационных систем во все сферы жизни возрастает ежедневно. Актуальной становится разработка подобных баз данных . При это разработчик должен учитывать то, что даже наиболее простые базы данных могут быть подвержены избыточности, но при этом не стоит забывать, что нельзя увлекаться деление баз данных на множество составных таблиц. Современные среды программирования имеют средства, которые позволяют разработать интуитивно понятные приложения, что является несомненным плюсом.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абросимова, М. А. Базы данных: проектирование и создание программного приложения в СУБД MS Access: практикум / М.А. Абросимова. – М, 2014. – 54 с.

2. Абросимова, М. А. Базы данных: манипулирование данными на языке SQL в СУБД MS Access: практикум / М.А. Абросимова. – М, 2013. – 32 с.

3. Абросимова, М. А. Базы данных: работа с формами в СУБД MS Access 2007: практикум / М.А. Абросимова. – М, 2013. – 13 с.

4. Аникеев, С. В. Разработка приложений баз данных в Delphi: самоучитель / С. В. Аникеев. – М, 2013. – 125 с.

5. Ачкасов, В. Ю. Программирование баз данных в Delphi: учебное пособие / В.Ю. Ачкасов. – М, 2013. – 93 с.

6. Баженова, И. Ю. Основы проектирования приложений баз данных / И.Ю. Баженова. – М, 2016. – 19 с.

7. Васюков О. Г. Управление данными: учебно-методическое пособие / О.Г. Васюков. – М, 2014. – 101 с.

8. Гладких, Т. В. Технологии электронного офиса: учебное пособие / Т.В. Гладких, Е.В. Воронова. – М, 2014. – 46 с.

9. Громов, Ю. Ю. Управление данными: учебник / Ю.Ю. Громов. – М, 2015. – 11 с.

10. Грошев, А. С. Информатика: учебник / А.С. Грошев. – М, 2015. – 245 с.

11. Дьяков, И. А. Базы данных. Язык SQL: учебное пособие / И.А. Дьяков. – М, 2013. – 14 с.

12. Жданов, С.А. Информационные системы: учебник / С.А. Жданов. – М, 2015. – 80 с.

13. Кузнецов, С.Д. Введение в реляционные базы данных / С.Д. Кузнецов. – М, 2016. – 145 с.

14. Королев, В. Т. Технология ведения баз данных: учебное пособие / В.Т. Королев, Е.А. Контарёв, А.М. Ченых. – М, 2015. – 56 с.

15. Литовка, Ю. В. Основы проектирования баз данных в САПР: учебное пособие / Ю.В. Литовка. – М, 2013. – 36 с.

16. Маркин, А. В. Разработка отчетов в информационных системах: учебное пособие / А.В. Маркин. – М, 2013. – 213 с.

17. Медведкова, И. Е. Базы данных / Е.И. Медведкова, Ю.В. Бугаев, С.В, Чикунов. – М, 2014. – 59 с.

18. Сирант, О. В. Работа с базами данных / О.В. Сирант, Т.А. Коваленко. – М, 2016. – 61 с.

19. Смирнов, А. А. Технологии программирования: учебное пособие / А.А Смирнов. – М, 2013. – 154 с.

20. Стасышин, В. М. Проектирование информационных систем и баз данных: учебное пособие / В.М. Стасышин. – М, 2013. – 86 с.

21. Сысоев, Э. В. Особенности построения баз данных: учебное пособие / Э.В. Сысоев, А.В. Селезнев. – М, 2013. – 21 с.

22. Токарева, М. А. Введение в современные информационные технологии: учебное пособие / М.А. Токарева. – М, 2013. – 178 с.

23. Хвощев, С. В. Программирование в среде Delphi задач навигации и картографирования: учебное пособие / С.В. Хвощев. – М, 2016. – 36 с.

24. Чурбанова О. В. Базы данных и знаний. Проектирование баз данных в Microsoft Access: учебно-методическое пособие / О.В. Чурбанова. – М, 2016. – 123 с.

25. Щелоков, С. А. Разработка и создание баз данных средствами СУБД Access и SQL Server: учебное пособие / С.А. Щелоков. – М, 2014. – 38 с.