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 РЕГИСТРАТУРЫ ВЕДОМСТВЕННОЙ ПОЛИКЛИНИКИ "ЭСКУЛАП"

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

Научный руководитель
Кандидат 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

Целью данной работы является более глубокое рассмотрение теории Баз Данных и непосредственное создание базы данных для Поликлиники.

Актуальность данной работы состоит в том, что создание базы данных является одним из самых важных частей предприятия в целом. Она позволяет правильно сформировать необходимую информацию в кратком и понятном виде, сортировать данные для простоты использования информации.

Сама по себе База данных представляет собой совокупность информации, которая характеризуется определенными признаками, позволяющими выбрать отдельные необходимые данные, или же ввести новые, сразу присваивая им конкретные признаки, которые будут использоваться для сортировки тех или иных данных. Так же она является набором сведений, хранящихся некоторым упорядоченным способом. Можно сравнить базу данных со шкафом, в котором хранятся документы. Иными словами, база данных - это хранилище данных. Сами по себе базы данных не представляли бы интереса, если бы не было систем управления базами данных (СУБД).

В данной работе будут рассмотрены такие программы как Microsoft Access, для создания самой базы данных, и Delphi для создания клиентского приложения.

Предмет исследования - базы данных.

Метод исследования - изучение теории, анализ литературы, обобщение собранных данных, практическое применение полученных знаний о базах данных.

Задачами данной курсовой работы являются более глубокое изучение теории баз данных, обобщение способов создания, формирование клиентского приложения для ведомственной регистратуры поликлиники "Эскулап".

1 Работа с Microsoft Access

1.1 Основные понятия и термины

Данная программа является одной из самых популярных систем для создания базы данных. Она удобна и проста в применении, пользуется популярностью не только у профессионалов, но и у новичков, так как не требует каких-то определенных или глубоких знаний.

Для создания базы данных следует знать основные понятия и термины, используемые этой системой, и упоминающиеся в этой работе[8].

Они следующие:

- DBMS - то же самое, что и сама СУБД, т.е система управления базой данных.
- MEMO - тип данных, позволяющий хранить большое количество текстовой информации.
- База Данных - непосредственно сам файл, в котором хранятся все созданные пользователем и необходимые для существования базы объекты. Программы, графические и текстовые данные, отчеты, формы и прочее.
- Объекты базы данных - основные части базы данных.
- Свойства - набор параметров, которые характеризуют определенный объект.
- Конструктор - непосредственный режим, в котором происходит создание таблиц и объектов.
- Таблица - состоит из столбцов (полей)и строк (записей).
- Форма - окно или же область в нем, которое представляет поля с данными, таблицами или элементами управления.
- Схема данных - графическое представление таблиц и связи между ними.

- Ключевое поле - позволяет создать поле, по которому будут определятся дальнейшие данные (например, нумерация или ID). Без него таблица существовать не может.
- Индекс - свойство поля, которое позволяет ускорить поиск и сортировку по базе данных.
- Счетчик - поле, в котором происходит автоматическая нумерация записей, введенных в ту или иную строку (в зависимости в какой строке было применено это свойство)[4].

1.2 Создание таблиц и ввод данных

Создание таблиц и ввод данных является основой для создания самой базы данных. В таблицу вводятся те или иные характеристики, по которым будет описываться объект с определенными свойствами, будь то клиент или, например, какой-либо продукт.

Для создания таблицы мы открываем саму программу Microsoft Access, затем нажимаем на "создание таблицы в режиме конструктора".

База данных любой компании, будь то поликлиника или магазин, основывается на многих вещах: люди, товары и прочее. Для конкретного примера мы возьмем людей. Для создания таблицы мы вводим в поля соответствующие названия полей:

- Номер или же ID
- Фамилия
- Отчество
- Дата рождения
- Номер телефона

Все эти данные являются основными и присущи для любого предприятия.

Далее наша задача выбрать тип полей. С помощью предлагаемого списка выбираем необходимые характеристики для того или иного поля.

На рисунке 1 представлена таблица, на которой мы выбираем определенную характеристику. Номер (ID) представляет собой нумерацию того или иного человека, и в конечном итоге подсчитывает количество введенных сотрудников. Поэтому для подсчета этого поля нам необходим "счетчик". Грубо говоря, каждый номер, присвоенный тому или иному человеку, будет уникальным и свойственным только ему.

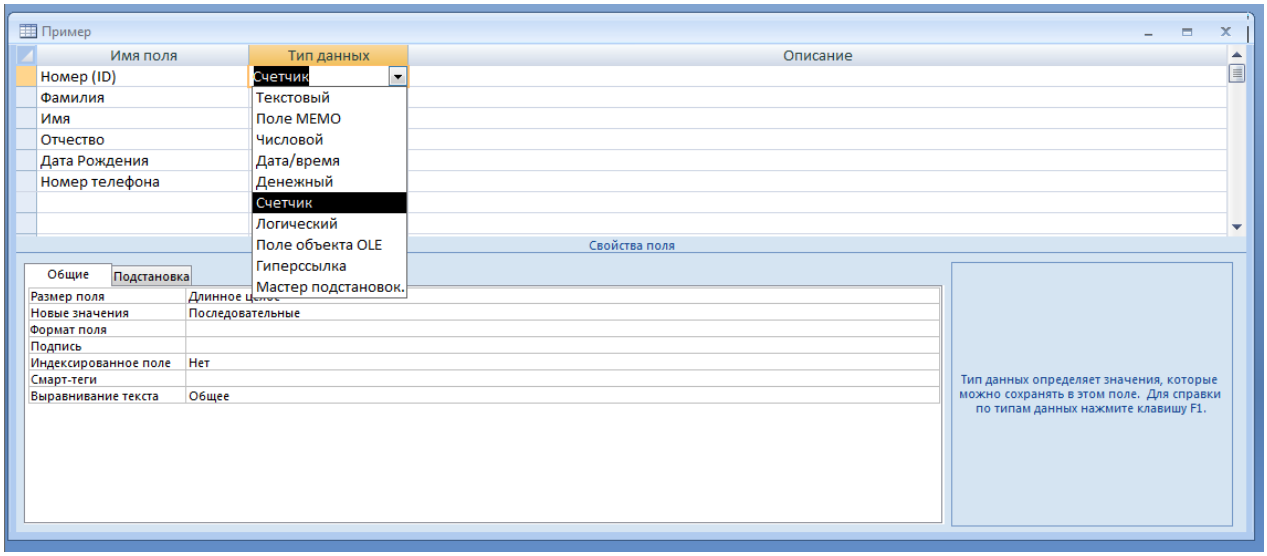


Рисунок 1 - Счетчик

На рисунке номер 2 представлена та же таблица, но теперь наша задача выбрать тип поля для Фамилии, Имени и Отчества. Так все они состоят исключительно из букв, значит тип поля необходим текстовый.

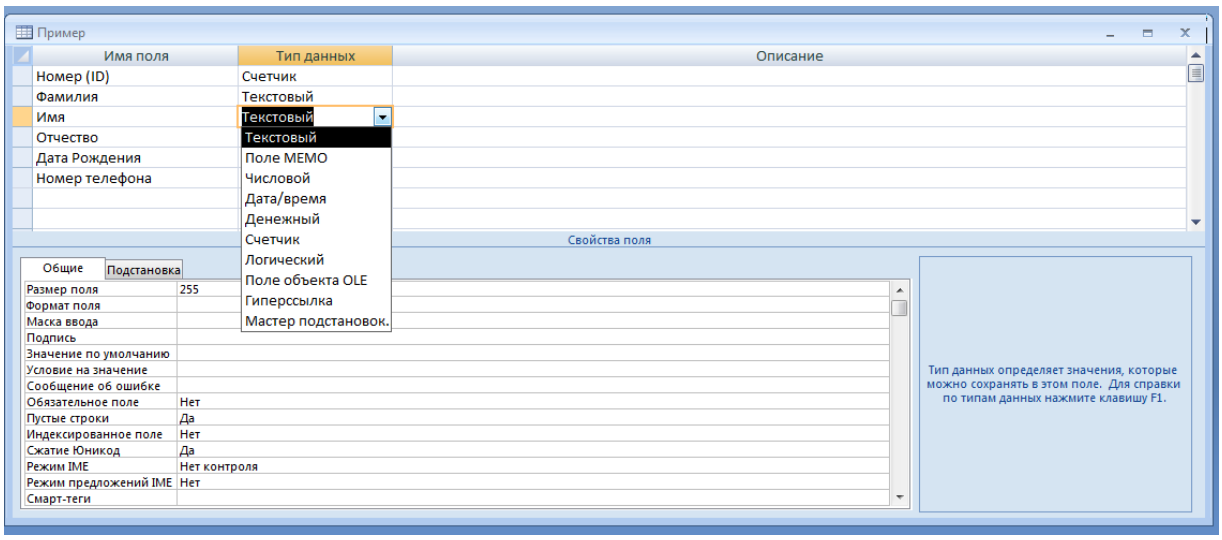


Рисунок 2 - Текстовый тип

На рисунке 3 мы выполняем следующую задачу: выбираем тип поля для Номера телефона. Он состоит из цифр, значит тип поля должен быть числовым.

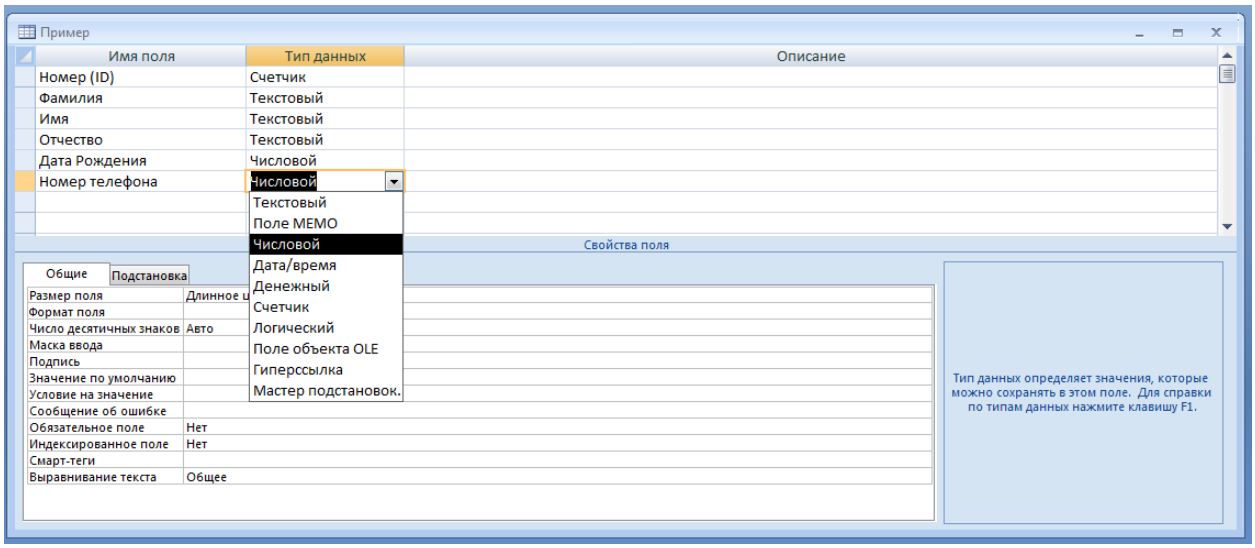


Рисунок 3 - Числовой тип

Затем, как показано на рисунке 4, мы выбираем тип поля для Даты рождения. Он состоит из цифр, но несет в себе определенную дату. Данная

программа позволяет вынести дату как отдельный тип, именно его мы и выбираем: Дата/Время.

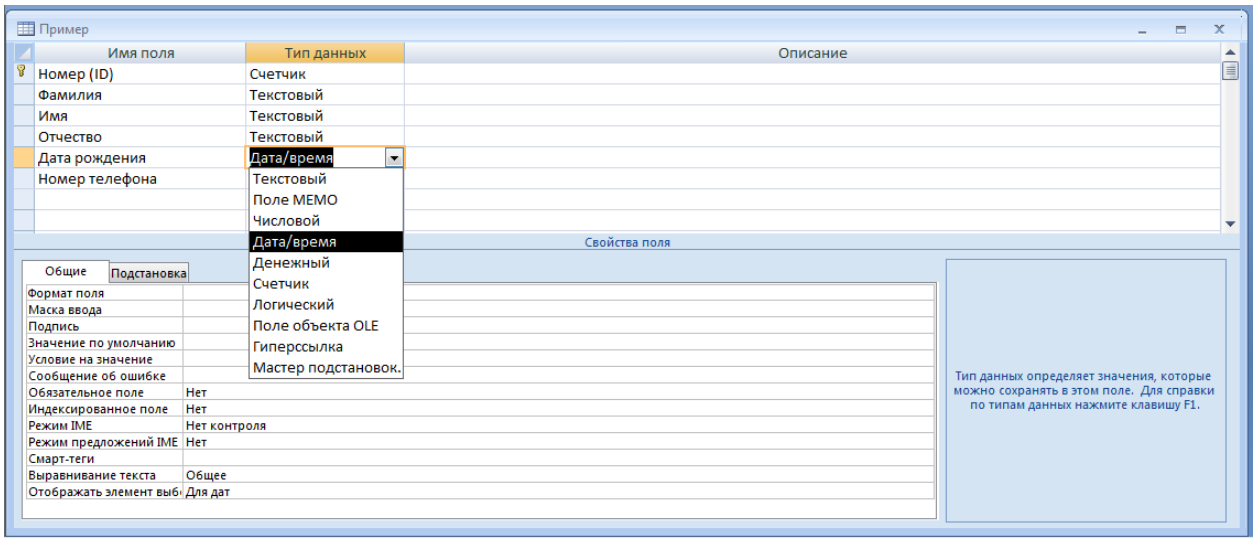


Рисунок 4 - Тип Дата/Время

После всех проделанных действий по названию полей и выбору их типов мы должны выбрать ключевое поле. Нажав на Нумерацию правой кнопкой мыши, выбираем Ключевое поле. Сохраняем таблицу и даем ей имя.

Затем нажав на нее дважды, программа откроет таблицу уже в конечном виде. И мы можем ввести туда необходимые нам данные[7].

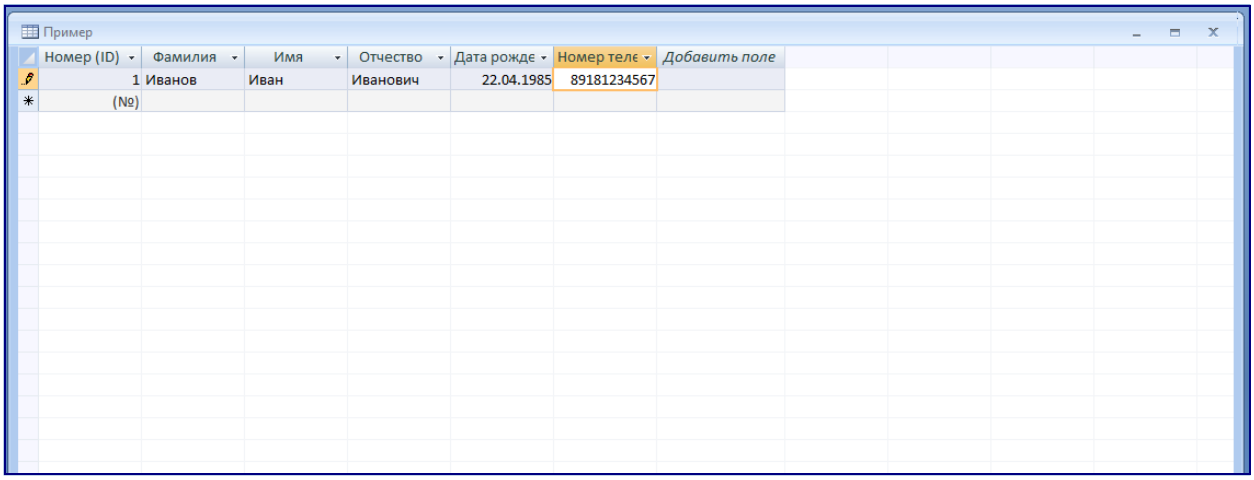


Рисунок 5 - Итог

1.3 Понятие схемы базы данных

Сама по себе любая схема представляет собой связь между элементами той или иной системы, к которой и относится схема.

Схема базы данных включает себя содержание, структура и целостность таблиц. Кратко говоря, она обобщает все таблицы и выводит их связь между друг другом.

Графическая структура базы данных очень похожа на логическую структуру базы данных, которая является адекватным отображением всех связей модели. Она создается, чтобы увидеть как все таблицы, существующие в созданной базе, связаны между собой, и как из одной таблице можно обратиться к другой.

Каждая таблица полученной схемы отображается в краткой форме, на которой выводятся имена полей, т.е те основные поля, которые были включены в таблицу[11].

Связи между таблицами отображаются в виде линий, которые связывают имя поля одной таблицы с именем поля другой. Таким образом получается графическое отображение связи таблиц.

Вся логическая структура базы зависит от количества таблиц и тех ссылок между ними, которые были вложены. Т.е реквизит одной таблицы связан с каким-либо объектом другой таблицы.

Связи между объектами данных осуществляются одинаковыми реквизитами – ключами связи в соответствующих таблицах. При этом ключом связи всегда является уникальный ключ главной таблицы. Ключом связи в подчиненной таблице является либо некоторая часть уникального ключа в ней, либо поле, не входящее в состав первичного ключа.

Создание таких схем в Access предполагается для того, чтобы отображать логическую структуру (составляющие ее части) базы данных. Это позволяет определить многозадачность созданной модели. Так же при наглядном рассмотрении схемы проще усваивать связь между таблицами, чтобы скорректировать ее для большей простоты в использовании[13].

На рисунке 6 представлена схема факультета, на которой отображены связи между таблицами.

Рисунок 6 - Схема данных факультета

Разберем каждую таблицу и ее связи подробнее.

Факультет является главной таблицей, от которой идут дальнейшие связи. Эта таблица связана с курсом и кафедрой, так как они являются ее составляющей частью.

Курс существует как отдельная таблица, связанная с факультетом. В этой таблице основными полями являются Номер курса, Начало сессии и Окончание сессии.

Группа тоже является отдельной таблицей, но связана с предыдущей таблицей, так как является ее составляющей частью. Здесь включены поля характерные для этой таблицы.

Таблица Студент включает в себя поля Номер зачетной книжки и Фамилия Имя и Отчество того или иного студента.

На этой ветви схемы мы видим что каждая таблица связана с предыдущей, и каждая из них, так или иначе, ссылается к предыдущей.

Рассматривая другую ветвь этой же схемы, мы видим 2 таблицы: Кафедра и Дисциплина. Они связаны между собой и непосредственно с самой таблицей Факультет. Здесь аналогичная ситуация, и каждая таблица связана с предыдущей, значит через каждую таблицу проходит связь с главной таблицей Факультета.

Вся эта схема позволяет наглядно рассмотреть свою базу данных в графическом представлении со связями всех ее элементов. Графическое представление позволяет более четко сформировать вид существующих данных, что позволяет скорректировать ее в случае необходимости.

Пользователь может установить необходимое ему количество связей, которые будут наиболее четко отражать структурную составляющую самой базы данных[1].

2 Язык программирования Delphi

2.1 Основные понятия

Delphi - это специальная система визуального объектно-ориентированного программирования. Она является одной из самых распространенных систем, позволяющих на современном уровне создавать как отдельные прикладные программы Windows, так и разветвленные комплексы систем, предназначенные для работы в корпоративных сетях и в Интернете[3].

В данной программе, как и в других системах, существуют свои базовые понятия, позволяющие работать с приложением. Они следующие:

- Класс объектов - совокупность объектов с определенными свойствами и характеристиками.
- Процедура - обработчик какого-либо события (например развертывание объекта при помощи клика мыши).
- Функция - тоже является обработчиком события, но в отличие от процедуры, имеет некоторое вычисляемое значение.
- Форма (Form) - является окном, которое видит пользователь при открытии программы. При запуске программы будет отображаться пустое поле[15].

Как и любое приложение для программирования, данная программа базируется на совокупности каких-либо действий. То есть для выполнения определенного конечного действия нужно выполнить ряд предшествующих ему действий. Так, например, для создания окна, на котором будет отображаться кнопка, при нажатии на которую, будет происходить то или иное действие. Чтобы осуществить эту задачу нужно поместить кнопку на пустое поле и проделать ряд действий описывая ее задачи с помощью заданных свойств кнопки (данная программа имеет ряд свойств каждого

объекта, что значительно упрощает работу с ней) и в конечном итоге будет получена простейшая программа[1].

2.2 Задачи и методы использования

Как и любая другая платформа, эта система имеет свои задачи и цели, для которых она была создана. Опишем их ниже.

- Быстрое создание оконного интерфейса.
- Создание различных приложений вычислительного, графического и мультимедийного характера.
- Создание систем и приложений для работы с различными базами данных, как удаленных так и локальных.
- Создание сложных приложений с характерной для них многозадачностью и разветвленностью компонентов.
- Создание приложений, которые могут быть использованы и другими программами: Microsoft Office, как Word, Excel и др.
- Составление программ различных классов и видов сложности для работы с ними в Интернете.
- Создание программ профессионального характера для установки приложений Windows, учитывая все требования, характеристики и специфику.

Возможности, которые поддерживает данная программа, сводят работу с созданием приложений к простым, наглядным и понятным процедурам, за которыми легко следить[2].

Во время работы с формой и размещением на ней каких-либо объектов Delphi автоматически формирует код программы. Проектирование сводится к наибольшей простоте работы: размещению компонентов, заданию им некоторых определенных свойств и написанию обработки событий.

Все компоненты библиотеки Delphi формируются по классам, т.е все элементы определенного класса характеризуются свойственным только им признаком. В классах описываются различные свойства объекта, события и методы, на которые он может реагировать[5].

Класс содержит несколько полей:

- FileVar — файловая переменная, необходимая для доступа к файлу.
- Delimiter — символ, который служит разделителем элементов.
- Items — массив элементов, полученных разбором последней считанной строки.

Класс также содержит ряд методов (процедур и функций):

- PutItem — помещает элемент в массив Items по индексу Index; если индекс превышает верхнюю границу массива, то размер массива автоматически увеличивается.
- SetActive — открывает или закрывает файл, из которого производится чтение строк[14].
- ParseLine — осуществляет разбор строки: выделяет элементы из строки и помещает их в массив Items; возвращает количество выделенных элементов.
- NextLine — считывает очередную строку из файла и с помощью метода ParseLine осуществляет ее разбор; в случае успешного чтения очередной строки функция возвращает значение True, а иначе значение False (достигнут конец файла).
- GetEndOfFile — возвращает булевское значение, показывающее, достигнут ли конец файла[6].

2.3 Объектно-ориентированное программирование

Само понятие объектно-ориентированного программирования (ООП) представляет собой такой способ программирования, в котором все

программы представляются в виде системы (совокупности) объектов, каждый из которых имеет определенно заданное свойство и выполняет своб функцию.

Важными частями ООП являются следующие детали:

- Объектно-ориентированное программирование использует в качестве логических элементов объекты, а не алгоритмы.
- Каждый из этих элементов является составляющей частью определенного класса.
- Все классы в своей совокупности представляют собой иерархию наследования.

В объектно-ориентированном программировании существуют следующие основные понятия:

- Абстракция данных
- Инкапсуляция
- Наследование
- Полиморфизм подтипов
- Класс
- Объект

Абстракция данных представляет собой выделение каких-то определенных данных из всех имеющихся. Очень часто это понятие просто называют "абстракция", в этом понятии так же подразумевается набор определенных характеристик, которые доступны остальной программе[9].

Инкапсуляция данных является свойством программы, которое позволяет ей объединять данные, методы, классы, с которыми работает программа.

Наследование позволяет программе описать новые данные с помощью характеристик других данных. Т.е описать входящие данные на основе уже существующих и функциональных признаков. "Родоначальник" наследования

является родителем, базовым началом или суперклассом. Новый класс, описанный с помощью данных своего родителя, называют потомком.

Полиморфизм подтипов позволяет обрабатывать данные различных типов без информации о разновидности.

Объектом является тот элемент, копия которого была использована для создания той или иной необходимой программы.

Класс является универсальным типом данных, в которых объединены все комплексы данных, состоящих из похожих набора полей[12].

Он содержит несколько полей:

- FileVar —переменная, которая нужна для доступа к файлу.
- Delimiter — символ, используемый для разделения элементов.
- Items — данные, которые были получены разбором последней существующей строки.

Класс также содержит ряд процедур и функций:

- PutItem — помещает элемент в массив Items по индексу Index. Если индекс превышает верхнюю границу массива, то размер массива автоматически увеличивается[13].
- SetActive — открывает или закрывает файл, из которого производится чтение строк.
- ParseLine — осуществляет разбор строки, выделяет элементы из строки и помещает их в массив Items; возвращает количество выделенных элементов в изначальное состояние.
- NextLine — считывает очередную строку из файла и с помощью метода ParseLine осуществляет ее разбор. В случае успешного чтения очередной строки функция возвращает значение True, а иначе — значение False (что говорит о достижении конца файла).
- GetEndOfFile — значение, которое показывает достигнут ли конец файла[10].

3 Разработка базы данных Поликлиника

3.1 Создание базы данных в Microsoft Access

Для создания базы данных нам нужно понять на чем именно базируется работа поликлиники. Т.е нам необходимо знать какие именно таблицы нам нужны. Прежде всего нам необходима база пациентов, в которой будут указываться ключевые характеристики пациентов. Так же необходима база врачей с их квалификациями и фамилиями. Необходима база диагнозов, их симптомов и способов лечения. И одним из важных моментов является журнал посещения для отслеживания посещений пациентов и врачей, у которых они наблюдаются.

Большим количеством информации будет обладать база пациентов, поэтому с нее и начнем.

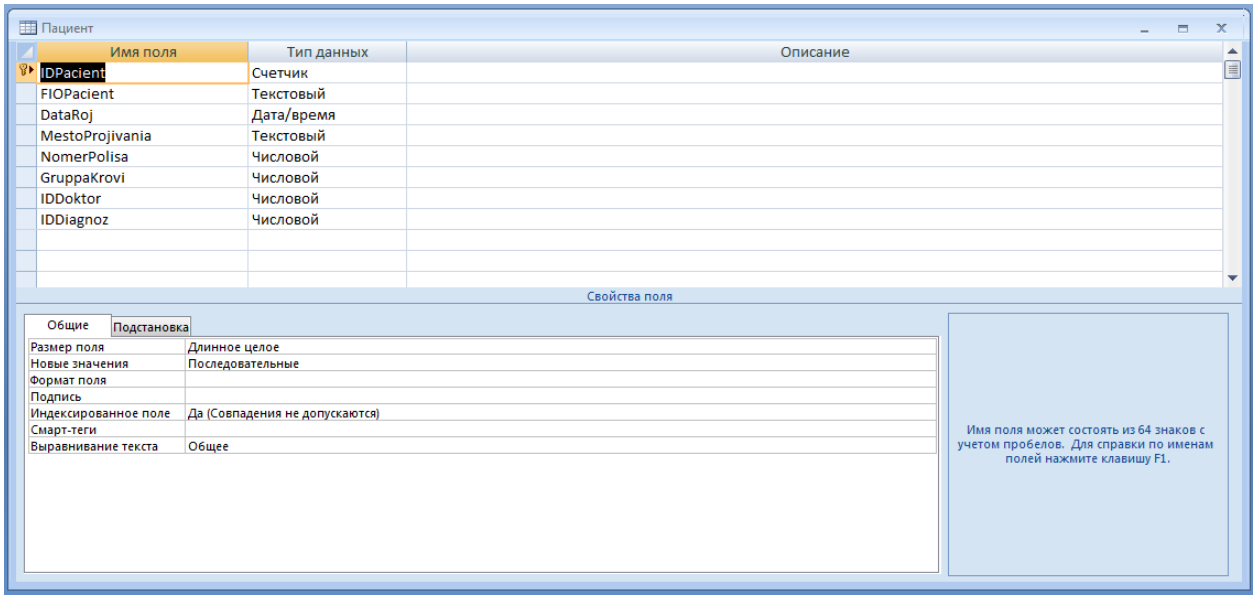


Рисунок 7 - Конструктор таблицы Пациентов

Для создания данных таблицы нам необходимо знать о пациенте определенные данные:

- Фамилию Имя Отчество

- Дату рождения

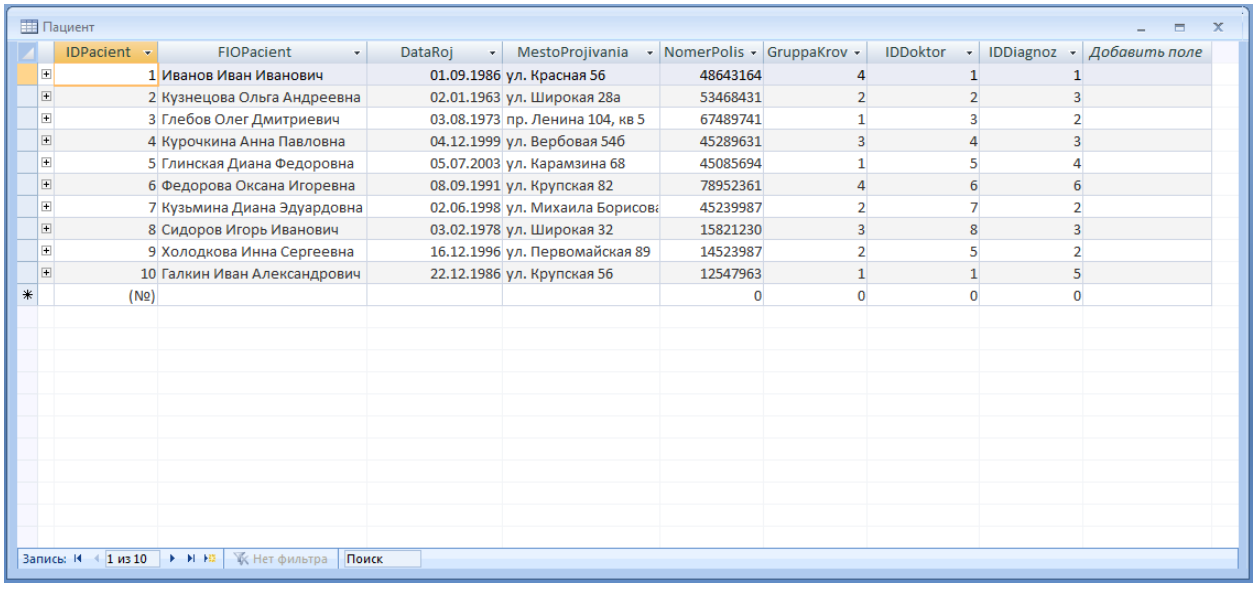
- Место проживания

- Номер полиса

- Группу крови

Чтобы связывать таблицы между собой мы создаем поле IDPacient, чтобы в дальнейшем пользоваться им как связующим элементом между таблицами. Такие же понятия будут создаваться и в других таблицах, поэтому выделяем отдельные поля для Доктора (IDDoktor) и Диагноза (IDDiagnoz), тип которых является числовым.

Выделяем ключевым полем IDPacient, открываем таблицу и заполняем ее:


Пациент

IDPacient	FIOPacient	DataRoj	MestoProjivania	NomerPolis	GruppaKrov	IDDoktor	IDDiagnoz	Добавить поле
1	Иванов Иван Иванович	01.09.1986	ул. Красная 56	48643164	4	1	1	
2	Кузнецова Ольга Андреевна	02.01.1963	ул. Широкая 28а	53468431	2	2	3	
3	Глебов Олег Дмитриевич	03.08.1973	пр. Ленина 104, кв 5	67489741	1	3	2	
4	Курочкина Анна Павловна	04.12.1999	ул. Вербовая 54б	45289631	3	4	3	
5	Глинская Диана Федоровна	05.07.2003	ул. Карамзина 68	45085694	1	5	4	
6	Федорова Оксана Игоревна	08.09.1991	ул. Крупская 82	78952361	4	6	6	
7	Кузьмина Диана Эдуардовна	02.06.1998	ул. Михаила Борисов	45239987	2	7	2	
8	Сидоров Игорь Иванович	03.02.1978	ул. Широкая 32	15821230	3	8	3	
9	Холодкова Инна Сергеевна	16.12.1996	ул. Первомайская 89	14523987	2	5	2	
10	Галкин Иван Александрович	22.12.1986	ул. Крупская 56	12547963	1	1	5	
(№)				0	0	0	0	

Запись: 1 из 10 | Нет фильтра | Поиск

Рисунок 8 - Итоговая таблица Пациент

Теперь переходим к таблице Докторов. Для создания таблицы нам необходимо знать Фамилию Имя и Отчество доктора и его квалификацию.

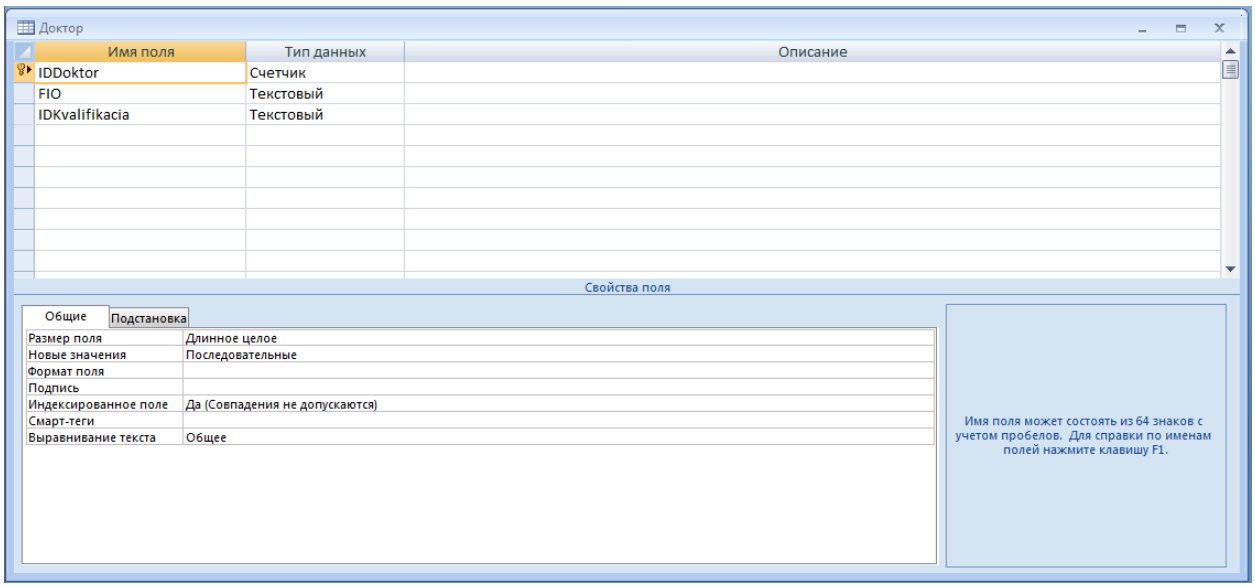


Рисунок 9 - Конструктор таблицы Докторов

В таблице докторов нам необходима информация о квалификации, но квалификаций может быть множество, они могут повторяться и вводить одну и ту же квалификацию не практично, поэтому создадим поле IDKvalifikacia и таблицу квалификаций отдельно, куда потом будет ссылаться таблица докторов.

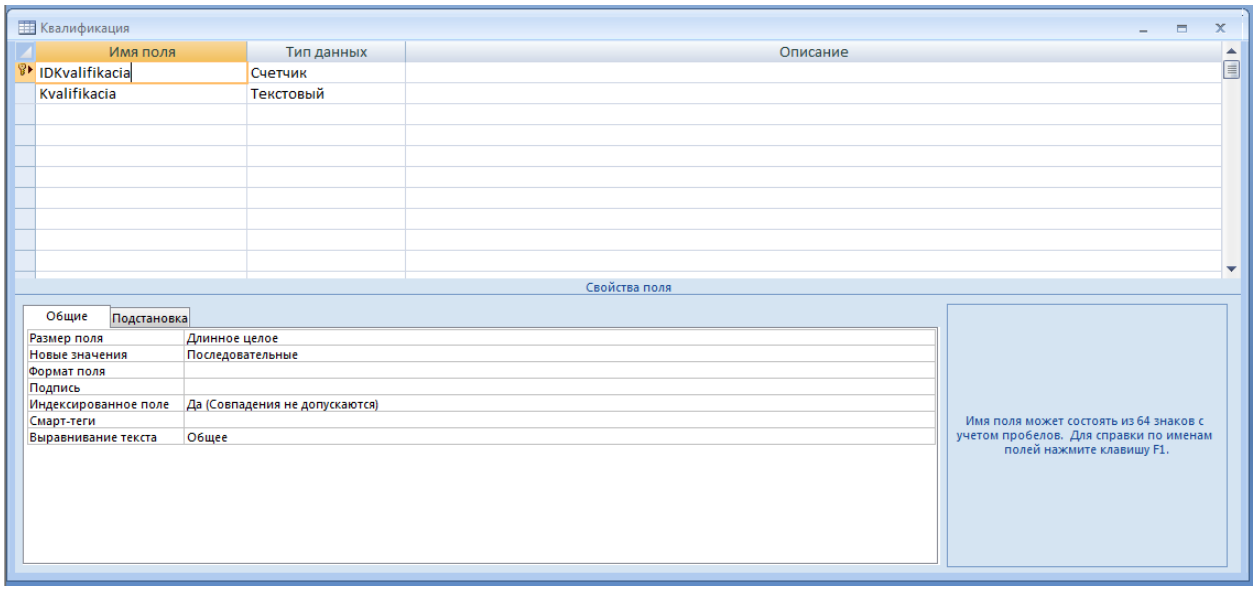


Рисунок 10 - Конструктор таблицы Квалификаций

Затем открываем таблицу Доктор и заполняем ее.

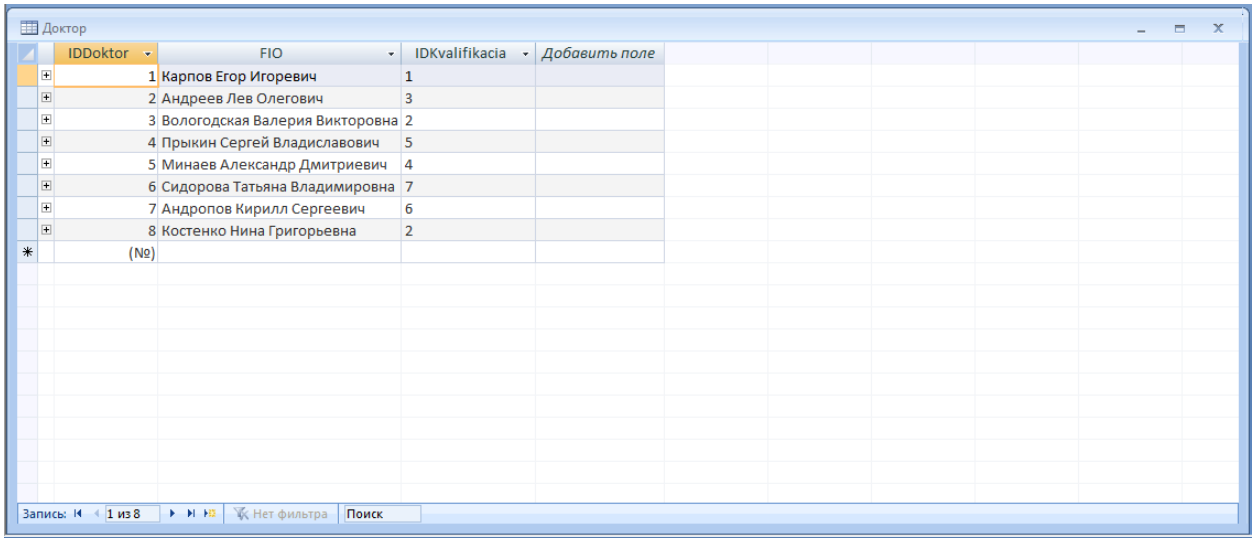


Рисунок 11 - Итог таблица Доктор

Открываем таблицу Квалификации и тоже заполняем ее:

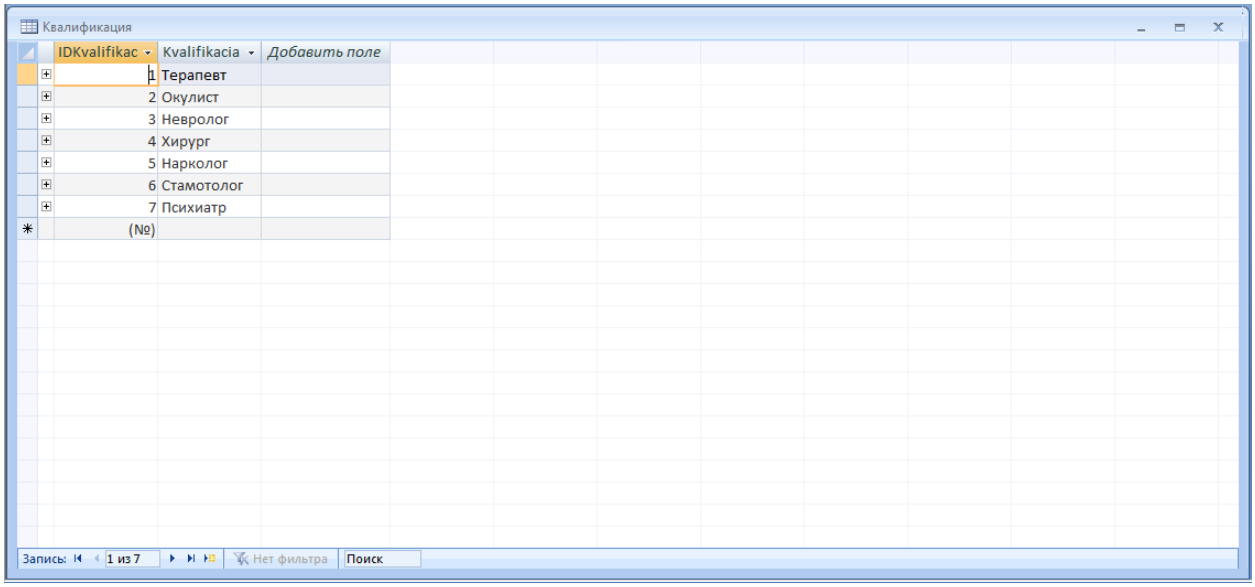


Рисунок 12 - Итог таблица Квалификация

Затем приступаем к созданию таблицы Диагноз. Для осуществления ее нам необходима следующая информация: название диагноза, симптомы и лечение.

Создаем таблицу в режиме конструктора:

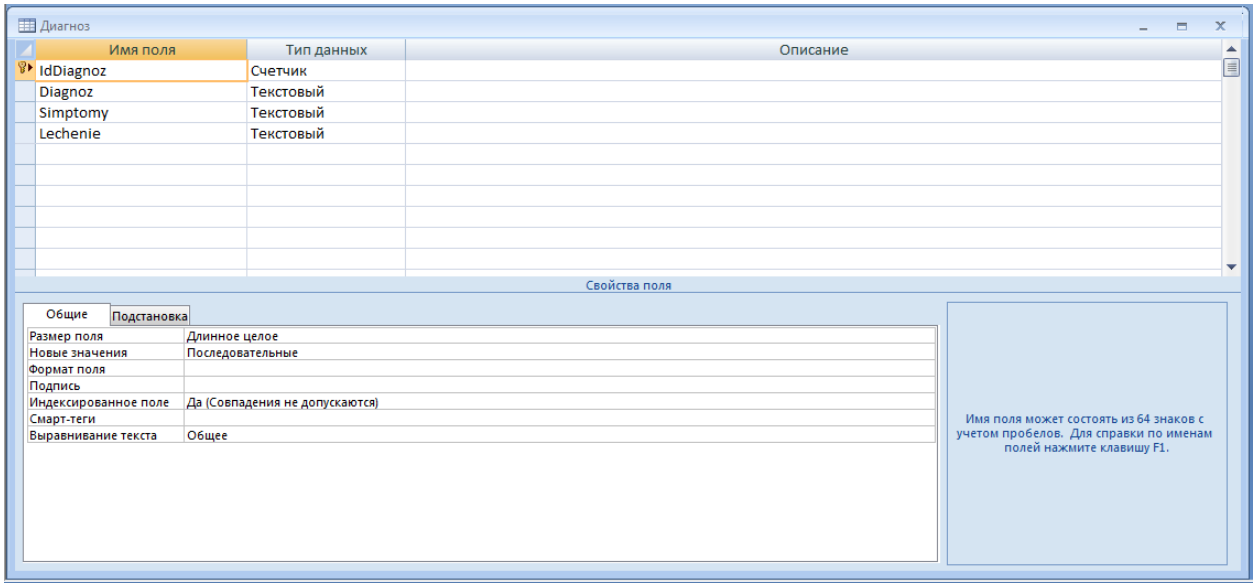


Рисунок 13 - Конструктор таблицы Диагноза

Для названий диагнозов, симптомов и лечения нам необходим текстовый тип полей.

Открываем таблицу и заполняем ее:

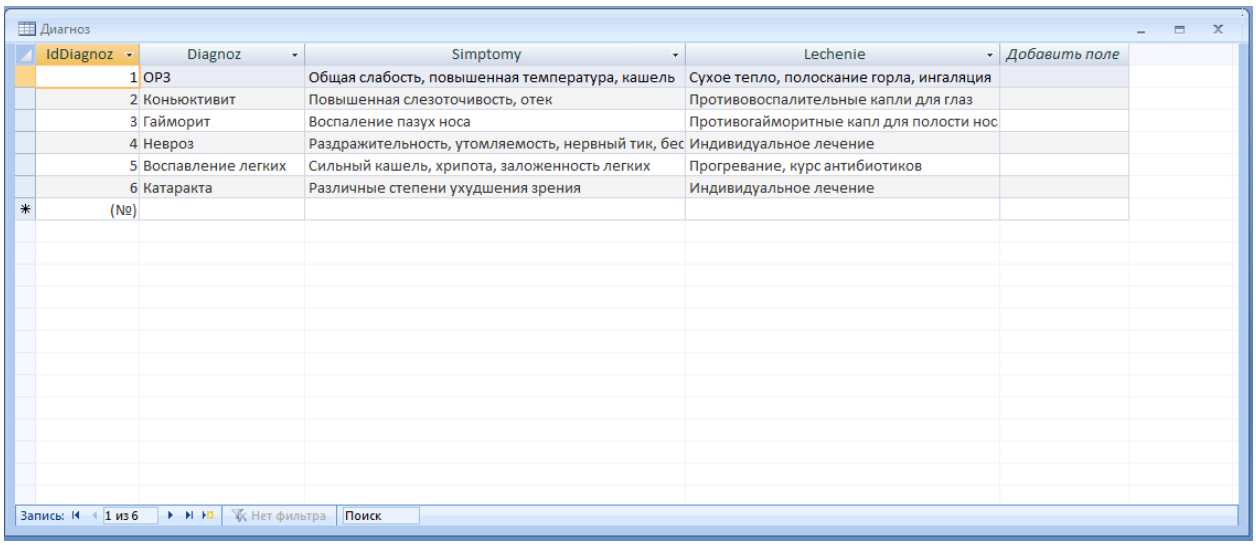


Рисунок 14 - Итог таблицы Диагноза

Теперь необходимо создать таблицу, где будут отображаться все посещения. Для нее нам необходимо знать следующее:

- Номер посещения
- Дату посещения
- Фамилию Имя и Отчество пациента
- Фамилию Имя и Отчество наблюдающего доктора
- Детали посещения (где будут описываться жалобы на самочувствие и прочее)

Создаем таблицу в режиме конструктора и выбираем соответствующие типы полей:

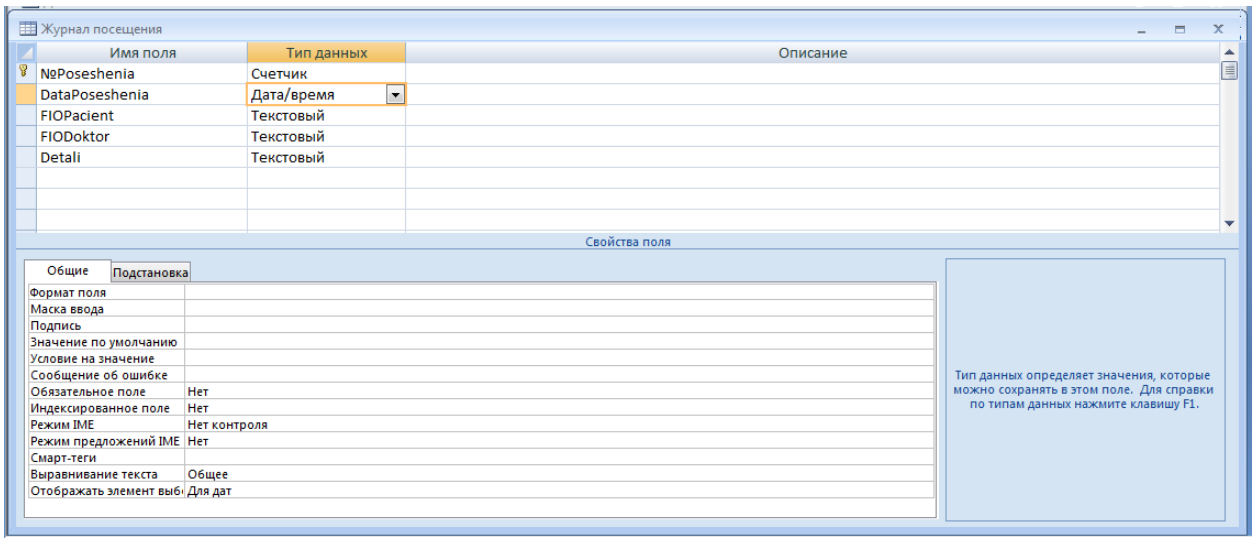


Рисунок 15 - Конструктор таблицы Журнал посещения

После создания всех полей выбираем ключевым полем номер посещения, сохраняем таблицу, открываем и заполняем ее:

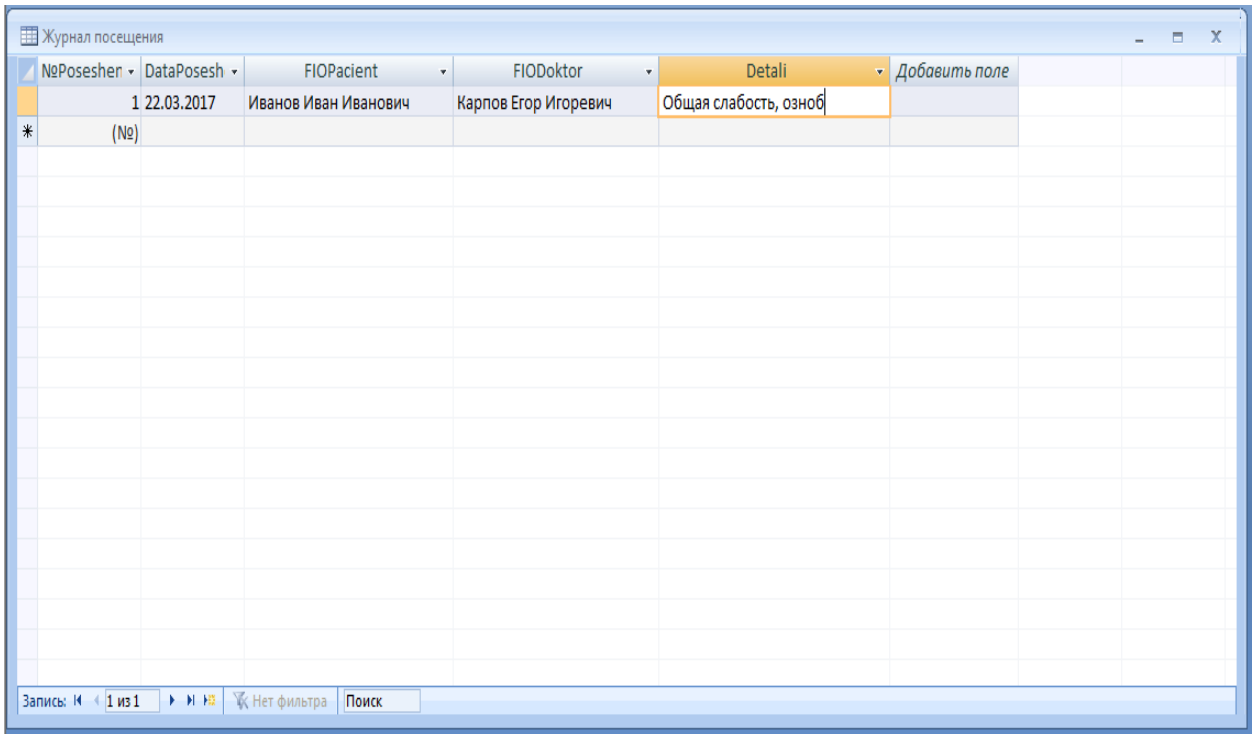


Рисунок 16 - Итог таблицы Журнал посещения

3.2 Создание схемы взаимосвязей

Как уже было оговорено ранее, схемы взаимосвязей позволяют отобразить как именно таблицы связаны между собой в графической форме.

Для создания такой схемы нам необходимо отобразить все созданные таблицы и соединить отдельные элементы одних таблиц с соответствующими элементами других таблиц.

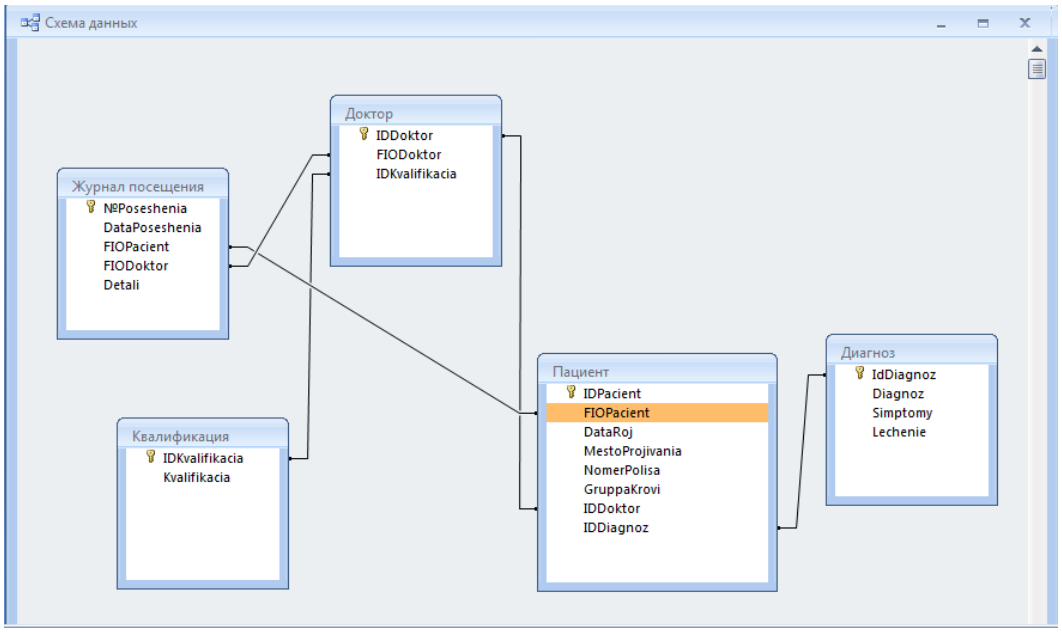


Рисунок 17 - Схема взаимосвязей

На данной схеме видно как, например, связаны таблицы Доктор и Квалификация. В таблице докторов есть поле IDKvalifikacia, которая связывает ее с таблицей Квалификация по аналогичному полю.

3.3 Создание основных форм в Delphi

Для создания основных форм в Делфи требуется сначала соединить базу данных с программой. Для этого нужно использовать Data Modul.

Мы используем ADOConnecting1 для того, чтобы затем соединить все таблицы и запросы между собой. У нас пять таблиц, следовательно необходимо взять пять объектов ADOTabel, затем 5 элементов DATASours, чтобы привязывать формы к таблицам. Так же необходим ADOQuery для создания дальнейших запросов. На рисунке ниже представлен вид Data Modul необходимый для нашей базы.

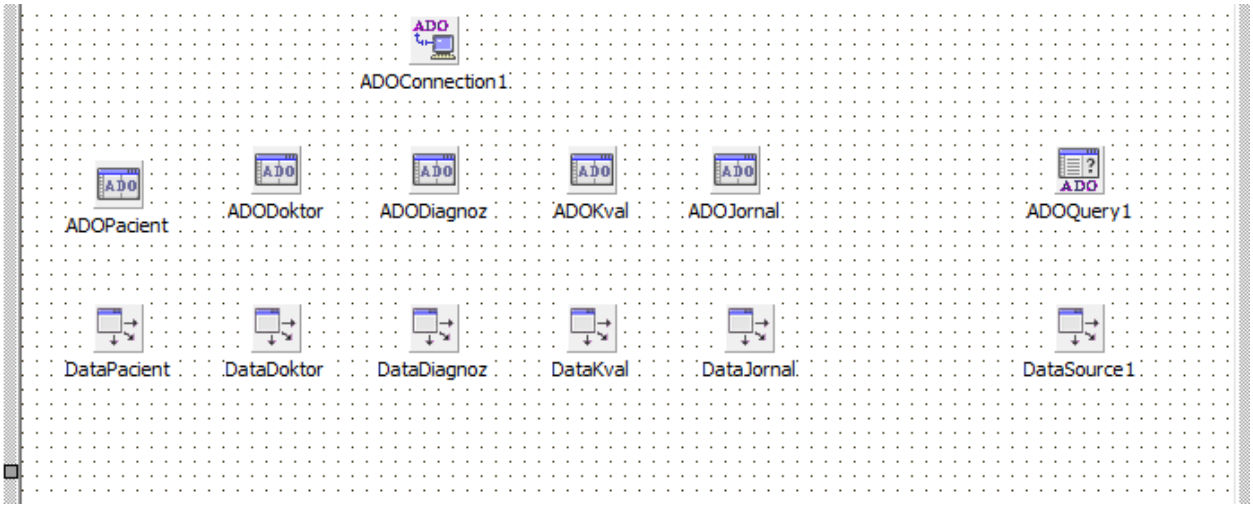


Рисунок 18 - Data Modul

В ADOConnecting мы указываем путь к нашей базе данных, затем подключаем через свойство "Connecting" все таблицы к исходному соединению. Аналогичным образом подключаем DATASours к исходным таблицам. Переименовываем все элементы для удобства.

Переходим к созданию основных форм. Первой, и начальной формой будет таблица с информацией о пациентах, для этого нам необходим объект TBGreed, который отображает данные взятые с той или иной таблицы. Подключаем его к необходимой таблице с помощью свойства DATASours. В свойствах TBGreed мы переименовываем поля для удобства пользования, скрываем поля ID и получаем следующее:

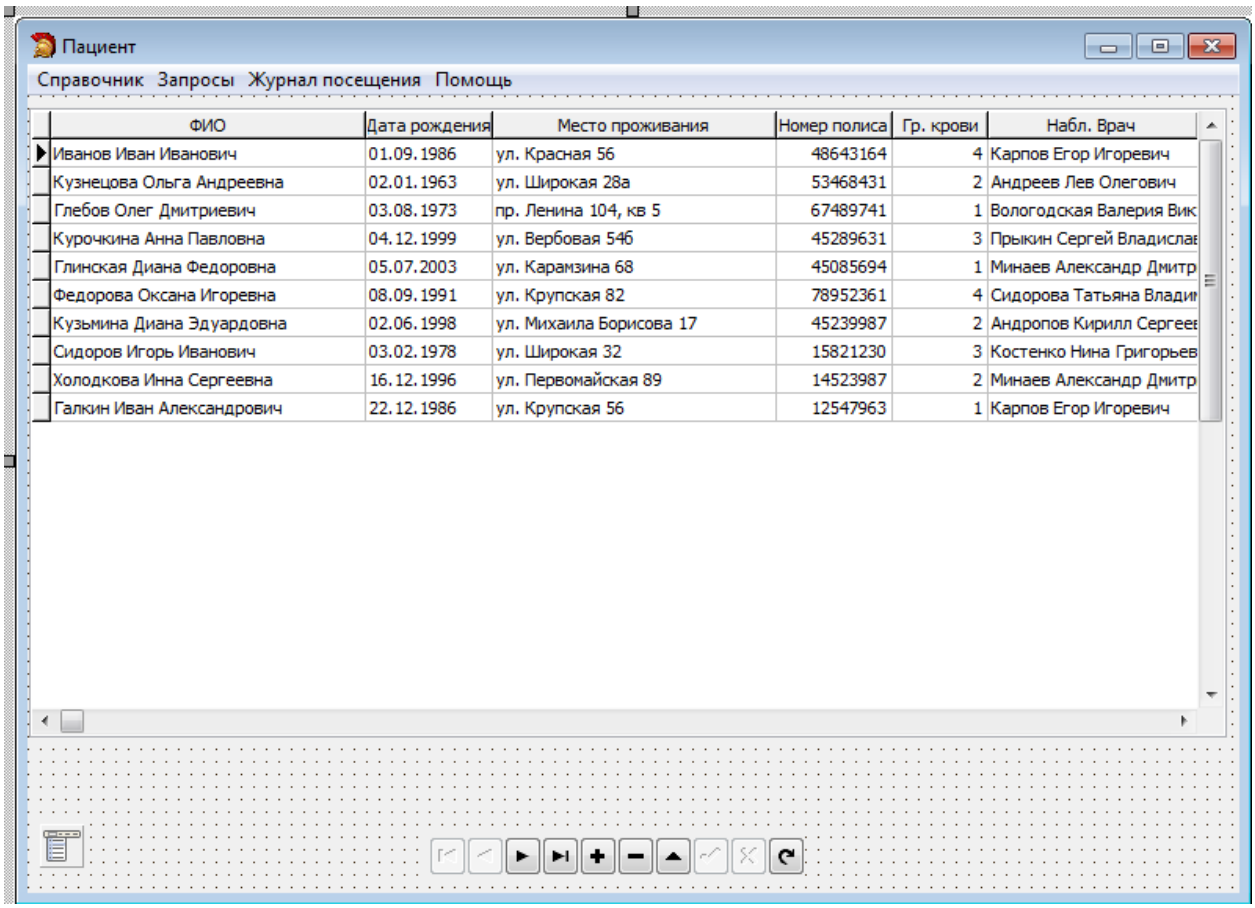


Рисунок 19 - Форма Пациент

Проделываем аналогичный действия с таблицей Докторов и получаем следующую форму:

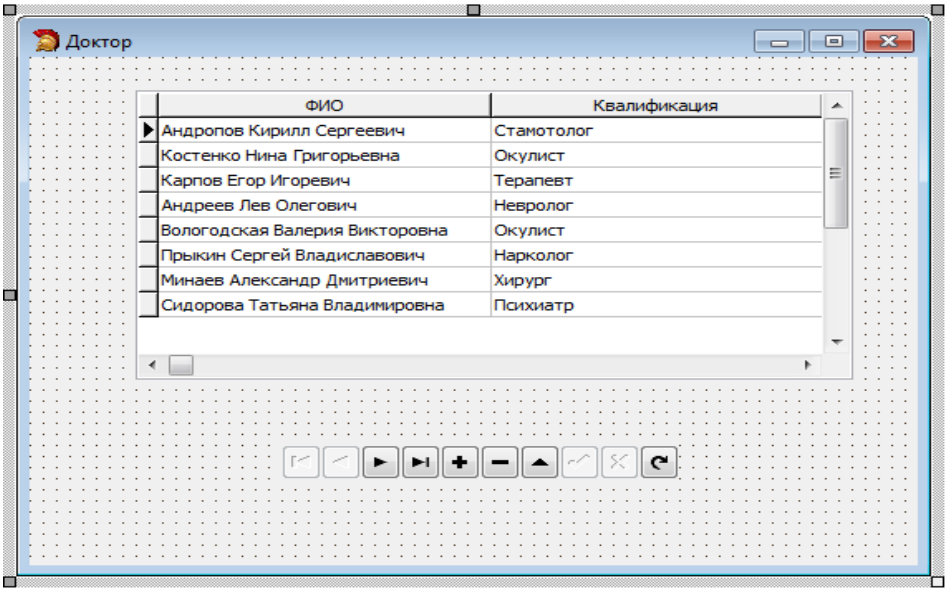


Рисунок 20 - Форма Доктор

Аналогичные действия применяем к Журналу посещения, Диагнозам и Квалификациям.

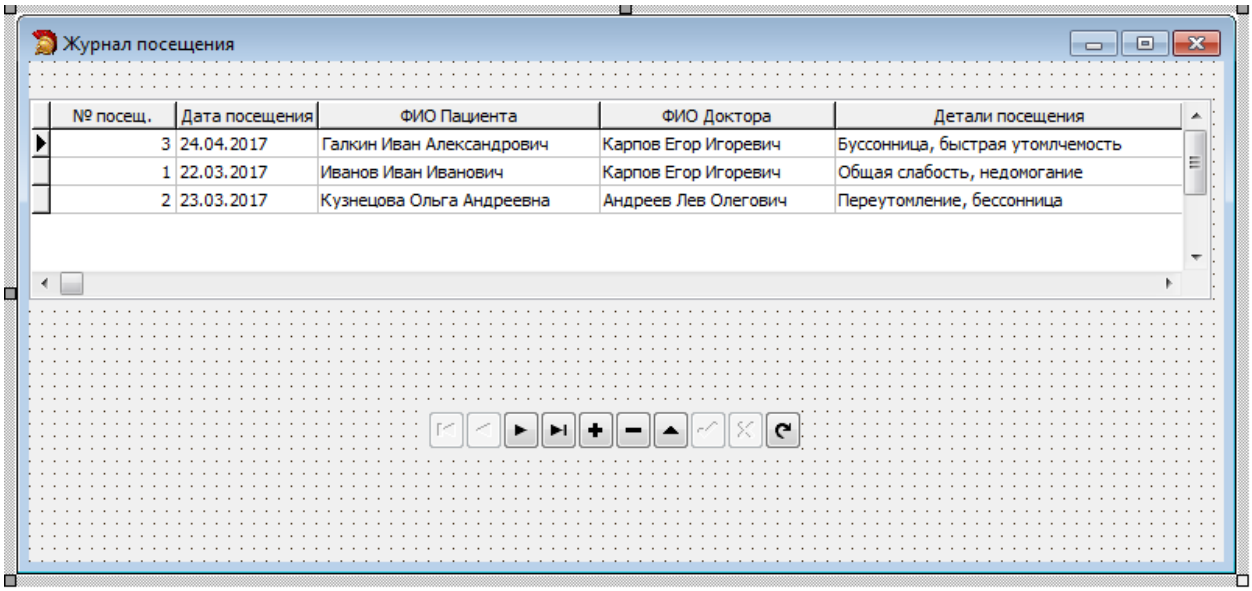


Рисунок 21 - Форма Журнал посещения

На рисунке 21 изображена уже оформленная форма для Журнала посещаемости с исправленными полями и с исправленной размерностью полей.

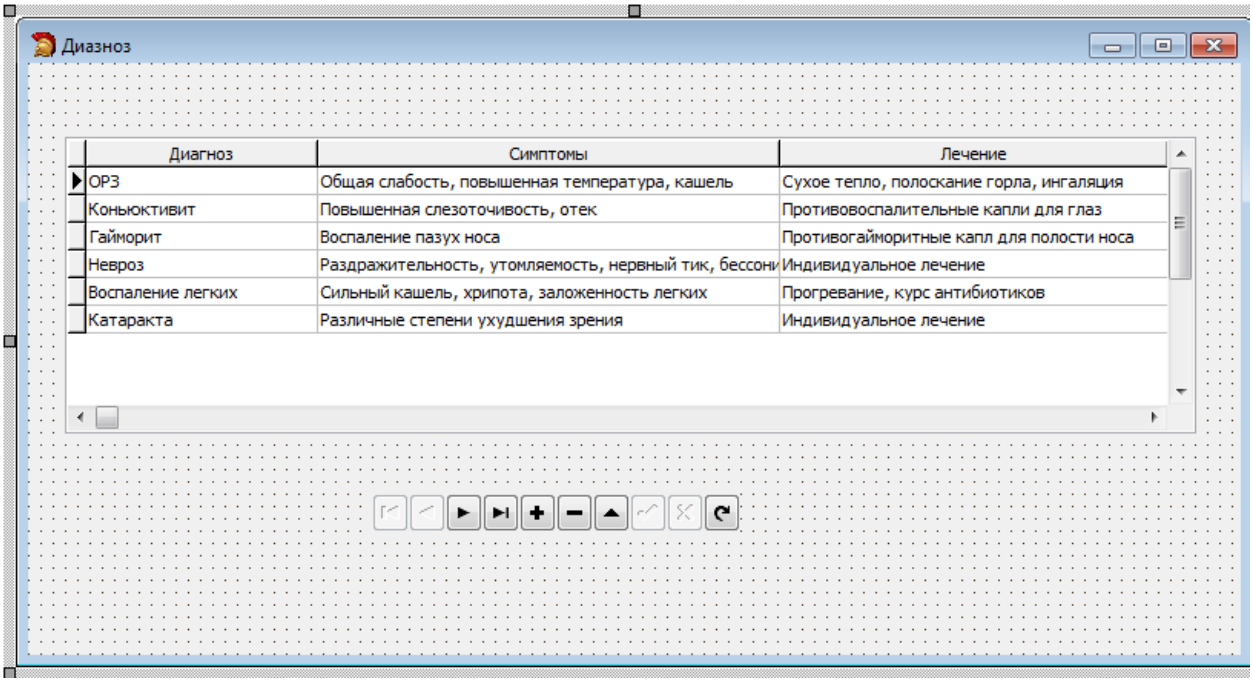


Рисунок 22 - Форма Диагноз

Форма Диагноз показывает название определенного диагноза, его симптомов и предполагаемого лечения. Все поля представлены в удобном для восприятия виде.

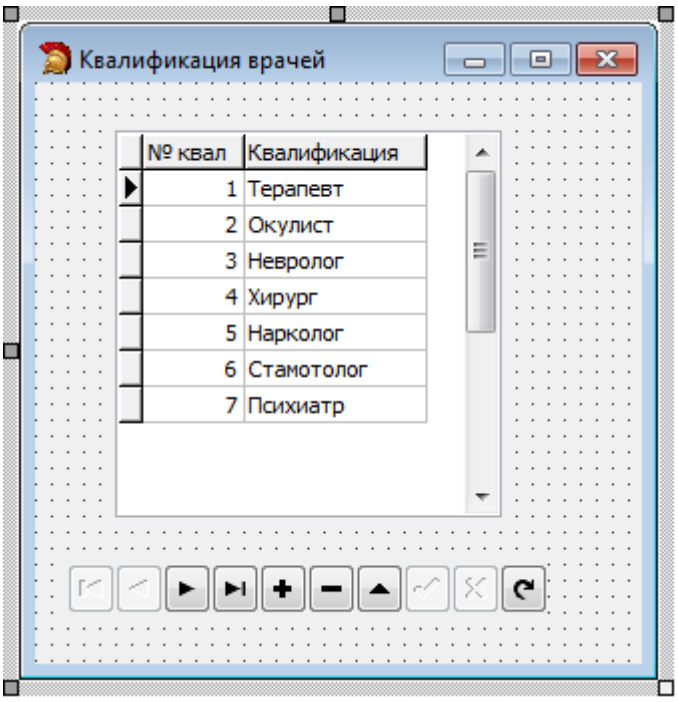


Рисунок 23 - Форма Квалификация

Форма с квалификацией позволяет увидеть разнообразие квалификаций или же профессий всех докторов, работающих в данной поликлинике.

3.4 Создание Запросов

Для удобства пользования программой нам необходимы запросы, где будут сортироваться данные с определенными характеристиками.

Первый запрос будет отображать фамилии людей с определенной группой крови. Для его осуществления создаем форму и располагаем на ней TBGreed, связывая его через DATASours, и Combobox.

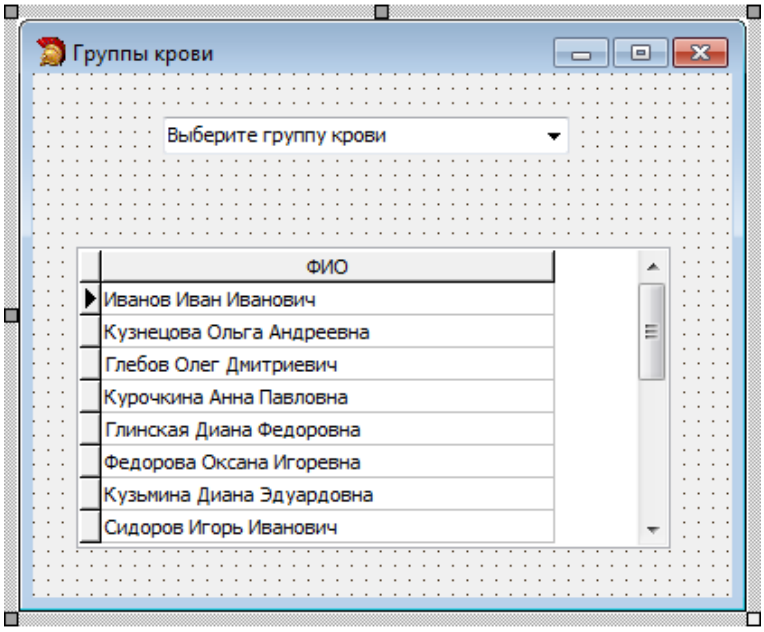


Рисунок 24 - Запрос Группы крови

Сортировка происходит по следующему коду:

```
-procedure TForm7.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 FIOPacient from пациент where   пациент.GruppaKrovi=:p1');
DataModule2.ADOQuery1.Parameters.ParamValues['P1']:=str1;
DataModule2.ADOQuery1.Open;
end;
```

Второй запрос сортирует людей согласно их диагнозам. Создаем форму аналогичную прошлому запросу и проделываем те же действия со свойствами.

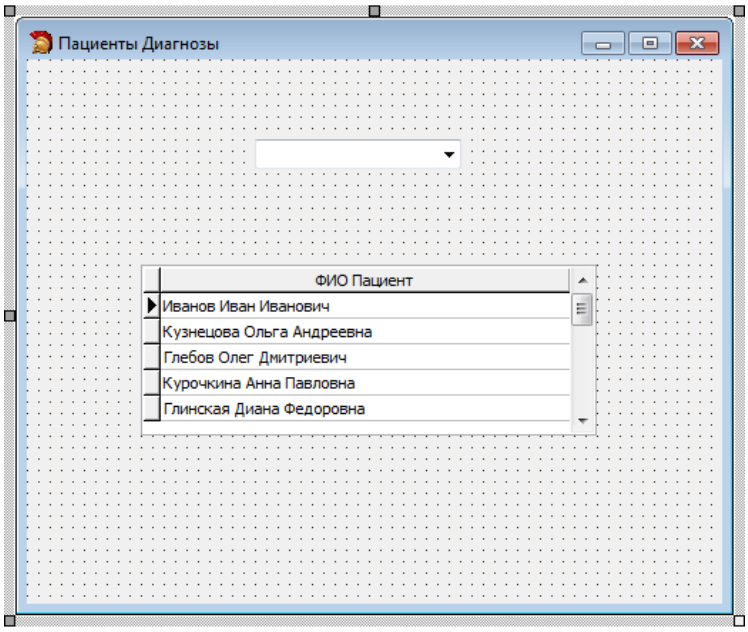


Рисунок 25 - Запрос Диагноз Пациент

Код запроса:

```
- procedure TForm10.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 пациент.FIOPacient, диагноз.diagnoz from пациент, диагноз
where пациент.IdDiagnoz=Диагноз.IdDiagnoz and
Диагноз.diagnoz=:p2');
DataModule2.ADOQuery1.Parameters.ParamValues['P2']:=str1;
DataModule2.ADOQuery1.Open;
end;
```

Третий запрос будет отображать фамилии докторов с определенной квалификацией:

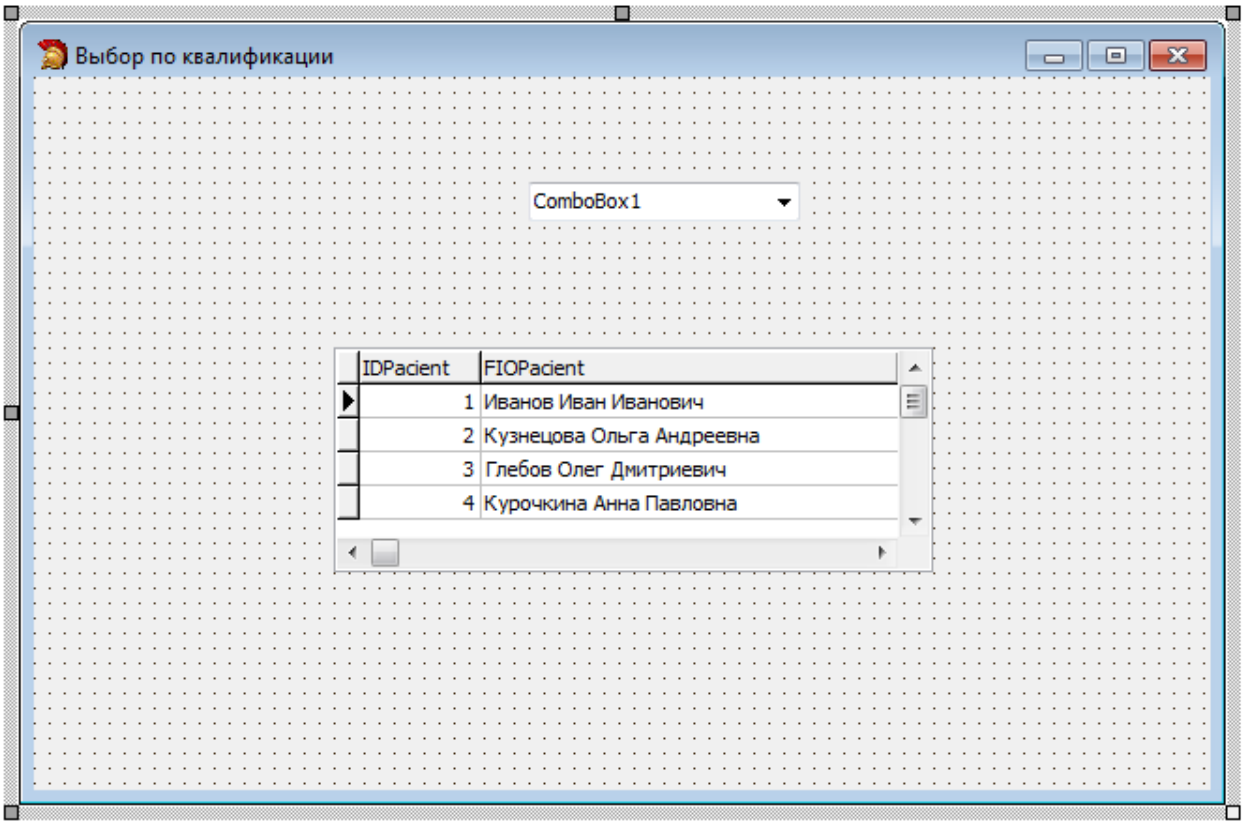


Рисунок 26 - Запрос Выбор по квалификации

Код запроса:

```
- procedure TForm14.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 Доктор.FIO,
Квалификация.Kvalifikacia from Доктор, Квалификация where
Доктор.IDKvalifikacia=Квалификация.IDKvalifikacia and
Квалификация.Kvalifikacia=:p1 ');
```

```
datamodule2.adoquery1.Parameters.ParamValues['p1']:=str1;
datamodule2.adoquery1.Open;
form14.dbgrid1.Columns[0].Width:=180;
form14.dbgrid1.Columns[0].title.alignment:=tacenter;
form14.dbgrid1.Columns[0].alignment:=tacenter;
form14.dbgrid1.Columns[0].title.caption:='ФИО';
form14.dbgrid1.Columns[1].Width:=0;
form14.dbgrid1.Columns[1].alignment:=tacenter;
form14.dbgrid1.Columns[1].title.alignment:=tacenter;
form14.dbgrid1.Columns[1].title.caption:='Квалификация';
end;
```

Последний запрос будет отображать наблюдаемых пациентов у выбранного врача.

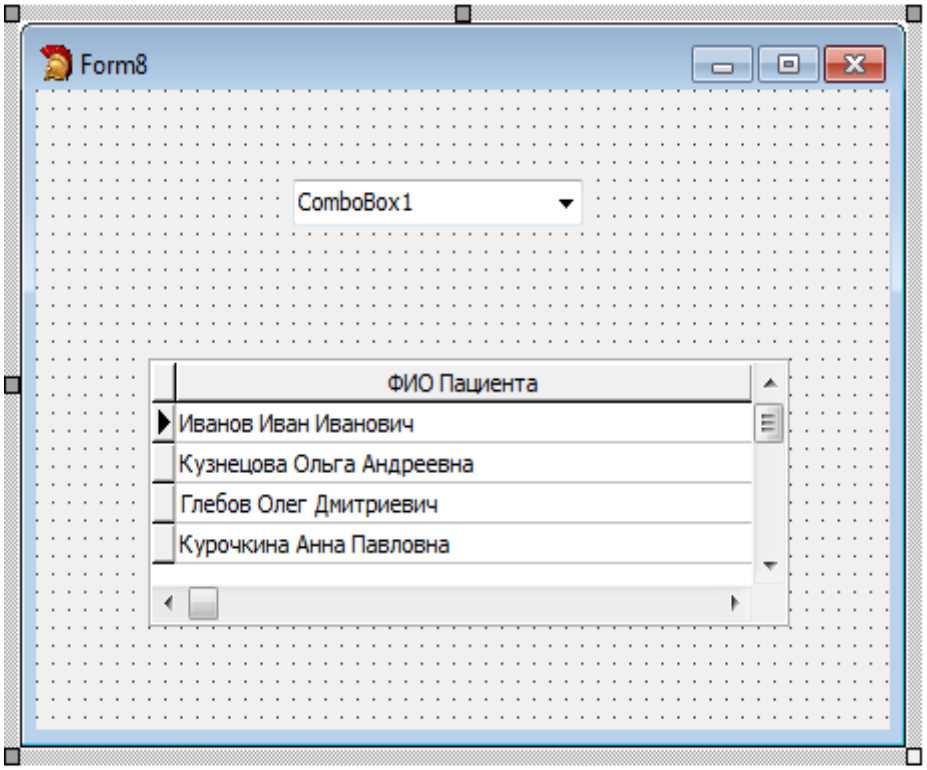


Рисунок 27 - Наблюдающий врач

Код запроса:

```
- procedure TForm8.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 пациент.FIOPacient, доктор.fio from пациент, доктор where
пациент.IdDoktor=доктор.IdDoktor and  Доктор.fio=:p2');
DataModule2.ADOQuery1.Parameters.ParamValues['P2']:=str1;
DataModule2.ADOQuery1.Open;
end;
```

С помощью объекта MainMenu мы выбираем где будут находится формы и запросы, пишем названия и через свойство OnClick описываем каждую форму с помощью свойства ShowModal.

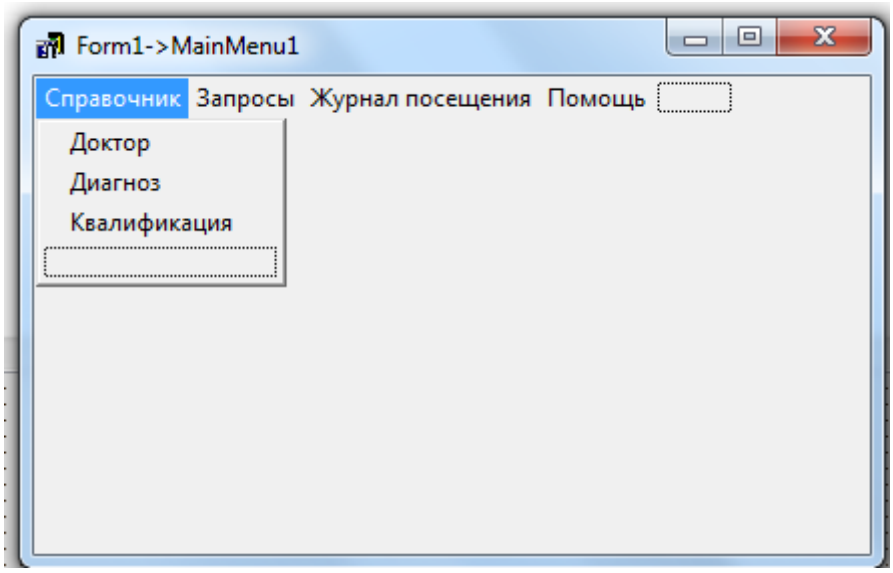


Рисунок 28 - Меню

ЗАКЛЮЧЕНИЕ

В данной курсовой работе была создана база данных для ведомственной регистратуры "Эскулап". Так же было создано клиентское приложение для удобства пользования и быстрого доступа ко всем имеющимся данным.

В процессе разработки базы данных мною был изучен материал для работы с Microsoft Access, а так же использована программа для создания приложения - Delphi. Все запросы были написаны на языке SQL. Все поставленные задачи были выполнены в необходимой последовательности.

Клиентское приложение включает в себя все основные таблицы в упрощенной для восприятия форме, так же имеются запросы с сортировкой необходимых характеристик.

В процессе работы были осуществлены следующие задачи:

- Собраны данные, касающиеся создания приложения и работы с базами данных.
- Созданы основные формы для удобной работы с программой.
- Создан удобный интерфейс.

Создание базы данных, на данный момент развития, является одним из основных критериев существования любого предприятия. Она позволяет отслеживать всю необходимую информацию, вносить новые данные, сортировать их по определенным критериям и т.д.

На сегодняшний день использование баз данных (БД) и информационных систем становится неотъемлемой частью функционирования любых организаций и предприятий. В связи с этим большую актуальность приобретает освоение принципов построения и эффективного применения соответствующих технологий и программных продуктов в виде систем управления базами данных (СУБД).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абаев, Х. С. Работа с базами данных / Х.С. Абаев. - М, 2014. - 121 с.

2. Аникеев, С. В. Разработка приложений баз данных в Delphi: самоучитель / С.В. Аникеев, А.В. Маркин. - М, 2014. - 45 с.

3. Акопян, А.А. Технологии электронного офиса: учебное пособие / А.А. Акопян, Е.В. Воронова. - М, 2016. - 21 с.

4. Баженова, И. Ю. Основы проектирования приложений баз данных / И.Ю. Баженова. - М, 2016. - 238 с.

5. Беккер, Г.Д. Введение в реляционные базы данных / Г.Д. Беккер. - М, 2016. - 19 с.

6. Бекяшев, О.В. Работа с базами данных / О.В. Бекашев, Т.А. Коваленко. - М, 2016. - 150 с.

7. Вааль, В.Т. Технология ведения баз данных : учебное пособие / В.Т. Вааль, Е.А. Контарёв. - М, 2015. - 108 с.

8. Вдовина, И.Ю. Основы проектирования приложений баз данных / И.Ю. Вдовина. - М, 2016. - 238 с.

9. Виноградов, В.Ю. Введение в программирование на Delphi : курс / В.Ю. Виноградов. - М, 2015. - 260 с.

10. Дементьев, А.М. Программирование в среде DELPHI : учебное пособие / А.М. Дементьев, В.А. Епанешников. - М, 2015. - 288 с.

11. Дмитриев, Т.С. Базы данных: модели, разработка, реализация : учебное пособие / Т.С. Дмитриев. - М, 2016. - 241 с.

12. Жабин, А.М. Программирование на языке SQL : учебное пособие / А.М. Жабин, В.А. Курочин. - М, 2015. - 334 с.

13. Жуковский, В.Ю. Программирование баз данных в Delphi : курс / В.Ю. Жуковский. - М, 2016. - 382 с.

14. Журавлева, С.А. Базы данных : учебное пособие / С.А.Журавлева. - М, 2015. - 298 с.

15. Карпова, Т.С. Базы данных: модели, разработка, реализация : учебное пособие / Т.С. Карпова. - М, 2016. - 241 с.

16. Кеплер, В.Т. Технология ведения баз данных : учебное пособие / В.Т. Кеплер, Е.А. Кунин. - М, 2015. - 368 с.

17. Кольцов, И.В. Азбука Delphi: программирование с нуля : учебное пособие / И.В. Кольцов. - М, 2015. - 112 с.

18. Озерцев, И.А. Базы данных. Язык SQL : учебное пособие / И.А. Озерцев. - М, 2014. - 82 с.

19. Остроумов, М.А. Базы данных: проектирование и создание программного приложения в СУБД MS Access : практикум / М.А. Остроумов. - М, 2014. - 56 с.

20. Ощепков, В.М. Проектирование информационных систем и баз данных : учебное пособие / В.М. Ощепков. - М, 2014. - 100 с.

21. Сальников, А.М. DELPHI: проектирование СУБД / А.М. Сальников, В.А. Епанешников. - М, 2016. - 449 с.

22. Сливицкий, П.А. Информатика. Управление базами данных: Лабораторный практикум / П.А. Сливицкий. - М, 2014. - 38 с.

23. Тимереев, И.А. Универсальные объектно-ориентированные базы данных / И.А. Тимереев. - М, 2014. - 226 с.

24. Чернышов, Р.И. Системы управления базами данных / Р.И. Чернышов. - М, 2015. - 162 с.

25. Яковлев, А.А. Технологии программирования : учебно-практическое пособие / А.А. Яковлев. - М, 2015. - 192 с.