**Частное образовательное учреждение высшего образования**

**«Международный Институт Дизайна и Сервиса»**

**(ЧОУВО МИДиС)**

Кафедра математики и информатики

**КУРСОВАЯ РАБОТА**

По дисциплине

«Объектно-ориентированный анализ и программирование на языке С#»

На тему: «Информационная система военного округа»

Направление: 09.03.03 Прикладная Информатика

Выполнил работу:

A.В Фомин

Группа П-226

Проверил работу:

С.С. Чеботарев

Челябинск 2023

**Содержание**

[ВВЕДЕНИЕ 3](#_Toc136355698)

[ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc136355699)

[1.1 Описание предметной области 4](#_Toc136355700)

[1.2 Анализ существующих решений 4](#_Toc136355701)

[ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА БАЗЫ ДАННЫХ 11](#_Toc136355703)

[2.1 Обоснование выбора СУБД 11](#_Toc136355704)

[2.2 Разработка инфологической модели 15](#_Toc136355705)

[2.3 Нормализация базы данных 16](#_Toc136355705)

[2.4 Разработка даталогической модели 18](#_Toc136355705)

[2.5 Инициализация базы данных 19](#_Toc136355705)

[2.6 Заполнение таблиц данными 21](#_Toc136355705)

[2.7 Дополнительные инструменты 23](#_Toc136355705)

[ГЛАВА 3. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ УПРАВЛЕНИЯ БАЗОЙ ДАННЫХ 25](#_Toc136355703)

[3.1 Обоснование выбора среды разработки 25](#_Toc136355704)

[3.2 Проектирование приложения 27](#_Toc136355704)

[3.3 Разработка приложения 29](#_Toc136355705)

[ЗАКЛЮЧЕНИЕ 36](#_Toc136355703)

СПИСОК ИСПОЛЬЗОВАННЫХ РЕСУРСОВ………………………………...37

# **ВВЕДЕНИЕ**

Информационная система военного округа - является сложной и многофункциональной системой, которая включает в себя множество компонентов. Это совокупность технических, программных средств и системы организационных мероприятий, предназначенная для автоматизации информационных процессов в военно-профессиональной деятельности. Целью курсовой работы является приобретение навыков проектирования и разработки базы данных, а также навыков управления данными в базе по средствам программного приложения.

Объектом исследования курсовой работы является база данных.

Предметом исследования является информационная система военного округа.

Основные задачи работы:

1. Исследование предметной области.
2. Разработка модели системы военного округа.
3. Выбор СУБД и среды программирования.
4. Проектирование и создание базы данных.
5. Проектирование и создание приложения для управления базой

данных.

По завершению программный продукт будет использоваться для работы с различной информацией о военном округе в базе данных.

# **ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

## **1.1 Описание предметной области**

Военные части округа расквартированы по различным местам дислокации, причем в одном месте могут располагаться несколько частей.

Каждая воинская часть состоит из рот, роты из взводов, взводы из отделений.

Военный округ представлен офицерским составом (генералы, полковники, подполковники, майоры, капитаны, лейтенанты) и рядовым и сержантским составом (старшины, сержанты, прапорщики, ефрейторы, рядовые).

Каждое из подразделений имеет командира, причем военнослужащие

офицерского состава могут командовать любым из вышеперечисленных

подразделений, а военнослужащие рядового и сержантского состава

только взводом и отделением. Все военнослужащие имеют одну или

несколько воинских специальностей.

Каждой воинской части придана боевая и транспортная техника: БМП,

тягачи, автотранспорт и пр. и вооружение: карабины, автоматическое

оружие, артиллерия, ракетное вооружение и т.д. Инфраструктура военной

части представлена набором сооружений (сооружение №1, сооружение

№2…)

## **1.2 Анализ существующих решений**

Каких-то существующих решений реализации программных продуктов в области военного и оборонного направления, я не нашел.

Поэтому в качестве примеров для рассмотрения, были выбраны продукты не связанные с военной тематикой, но по функционалу приложения они являются схожими.

**1С:Предприятие** — программный продукт компании «1С», предназначенный для автоматизации деятельности на предприятии.

«1С:Предприятие» предназначено для автоматизации любого бизнес-процесса предприятия. Наиболее известны продукты по автоматизации бухгалтерского и управленческого учётов (включая начисление зарплаты и управление кадрами), экономической и организационной деятельности предприятия.

Технологическая платформа «1С:Предприятие» представляет собой программную оболочку над базой данных. Используются базы на основе DBF-файлов в 7.7, собственный формат 1CD с версии 8.0 или СУБД Microsoft SQL Server на любой из этих версий. Кроме того, с версии 8.1 хранение данных возможно в PostgreSQL и IBM DB2, а с версии 8.2 добавилась и Oracle. Платформа имеет свой внутренний язык программирования, обеспечивающий, помимо доступа к данным, возможность взаимодействия с другими программами посредством OLE и DDE, в версиях 7.7, 8.0 и 8.1 — с помощью COM-соединения.

Клиентская часть платформы функционирует в среде Microsoft Windows, а начиная с версии 8.3, также в среде Linux и macOS. Начиная с версии 8.1, серверная часть платформы в клиент-серверном варианте работы «1С:Предприятия» может функционировать на ОС Microsoft Windows и Linux. Существуют специальные версии среды исполнения 1С для ноутбуков и PDA, ПО создания веб-приложений, взаимодействующих с базой данных «1С:Предприятие».

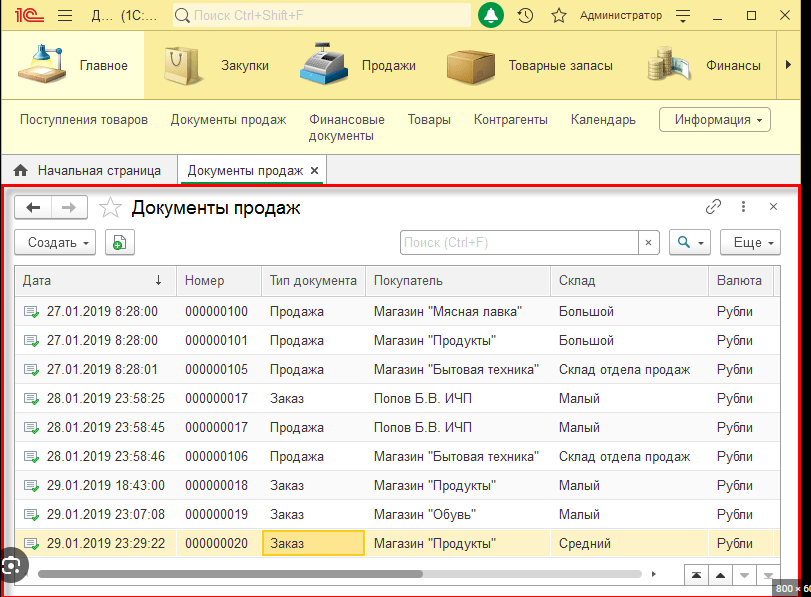


Рисунок 1 – Программа 1С

**«Avaya IP Office»** - это программа, которая предоставляет гибкую и масштабируемую телефонную систему для малых и средних бизнесов. Она предоставляет полный спектр функциональности телефонной системы, включая управление вызовами, маршрутизацию вызовов, голосовую почту, конференцсвязь, мобильную интеграцию и аналитику звонков. IP Office может быть установлен как на сервер, так и в облачной среде, что обеспечивает большую гибкость при выборе архитектуры.

Основными функциями «Avaya IP Office» являются: - Управление звонками: IP Office позволяет управлять звонками и маршрутизацией вызовов в соответствии с заданными правилами. Это включает автоматическое распределение вызовов, управление очередями вызовов и перенаправление вызовов.

- Голосовая почта: IP Office может обрабатывать голосовую почту, позволяя пользователям записывать голосовые сообщения и получать их через телефон или электронную почту.

- Конференцсвязь: IP Office позволяет создавать многопользовательские конференции, которые могут включать до нескольких сотен участников. - Мобильная интеграция: IP Office позволяет пользователям использовать мобильные устройства для совершения звонков и доступа к другим функциям телефонной системы.

- Аналитика звонков: IP Office предоставляет аналитические данные о звонках, которые могут быть использованы для улучшения эффективности работы бизнеса.

Благодаря своей гибкости и масштабируемости, «Avaya IP Office» является популярным выбором для малых и средних бизнесов, которые нуждаются в надежной и эффективной телефонной системе.

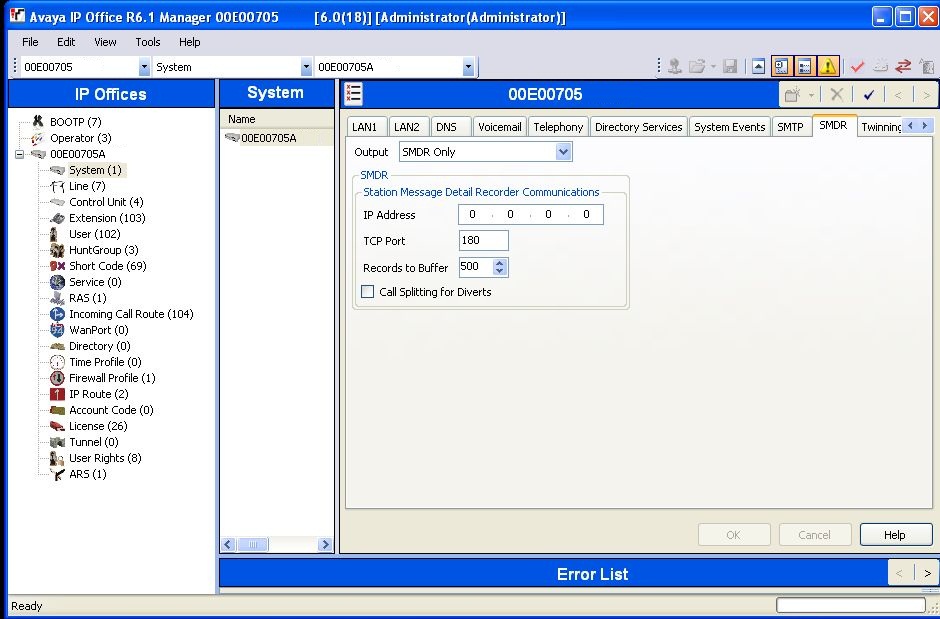


Рисунок 2 – Avaya IP Office

**МойСклад** — коммерческий SaaS продукт для управления торговлей и складского учёта, предназначенный для автоматизации малого и среднего бизнеса. Сервис реализует функции обработки заказов, управления продажами и закупками, складского учёта и контроля финансовых расчетов.

МойСклад стал одним из первых сервисов SaaS, разработанных в России и предназначенных для российского рынка. Первая версия сервиса была запущена осенью 2007 года. В 2008 году компания-разработчик получила финансирование от венчурного фонда Ambient Sound Investments.

В 2011 году инвестором и совладельцем сервиса стала компания 1С. По итогам 2012 года МойСклад вошел в ТОП-10 рейтинга «Крупнейшие поставщики SaaS в России 2012». В 2017 году МойСклад и Сбербанк создали облачный сервис управления торговлей для малого и среднего бизнеса Моя Торговля.

В 2019 году МойСклад занял 5 место в рейтинге «CNews Analytics: Крупнейшие поставщики SaaS в России 2019». В 2020 году МойСклад стал 6-м в рейтинге «Лучшие ИТ-работодатели России 2019: ежегодный рейтинг Хабр Карьеры».

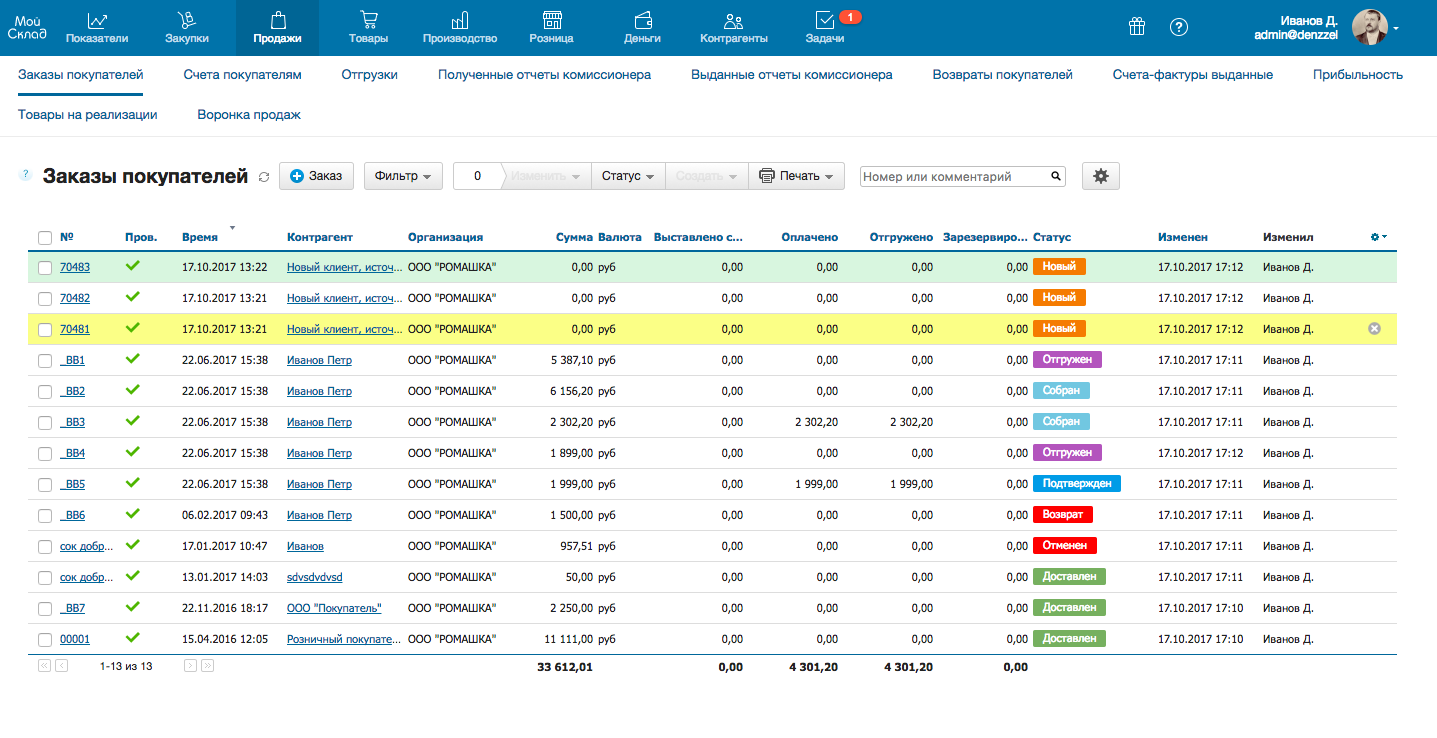


Рисунок 3 – МойСклад

**Zoho Office Suite** — онлайновый офисный пакет, включающий такие приложения, как текстовый процессор, электронные таблицы, презентации, базы данных, заметки, вики, систему управления взаимоотношениями с клиентами (CRM), управление проектами, ведение бюджета и тому подобное.

Название пакета является несколько переработанной распространенной аббревиатурой SOHO (англ. Small office, home office — малый офис, домашний офис). Поскольку пакет построен как веб-служба, все его приложения является платформо-независимыми. Zoho является примером бизнес схемы «Программа как услуга» (SaaS), или облачных вычислений, когда приложения выполняются в большей степени на удаленном сервере или на персональном компьютере пользователя. Работа с пакетом осуществляется через веб-браузер.

При использовании на начальном уровне базовый пакет Zoho предоставляется бесплатно, но при более интенсивном и профессиональном использовании, офисный пакет будет на платной основе. Одним из плюсов использования этой системы является полная автоматизация всех бизнес процессов.

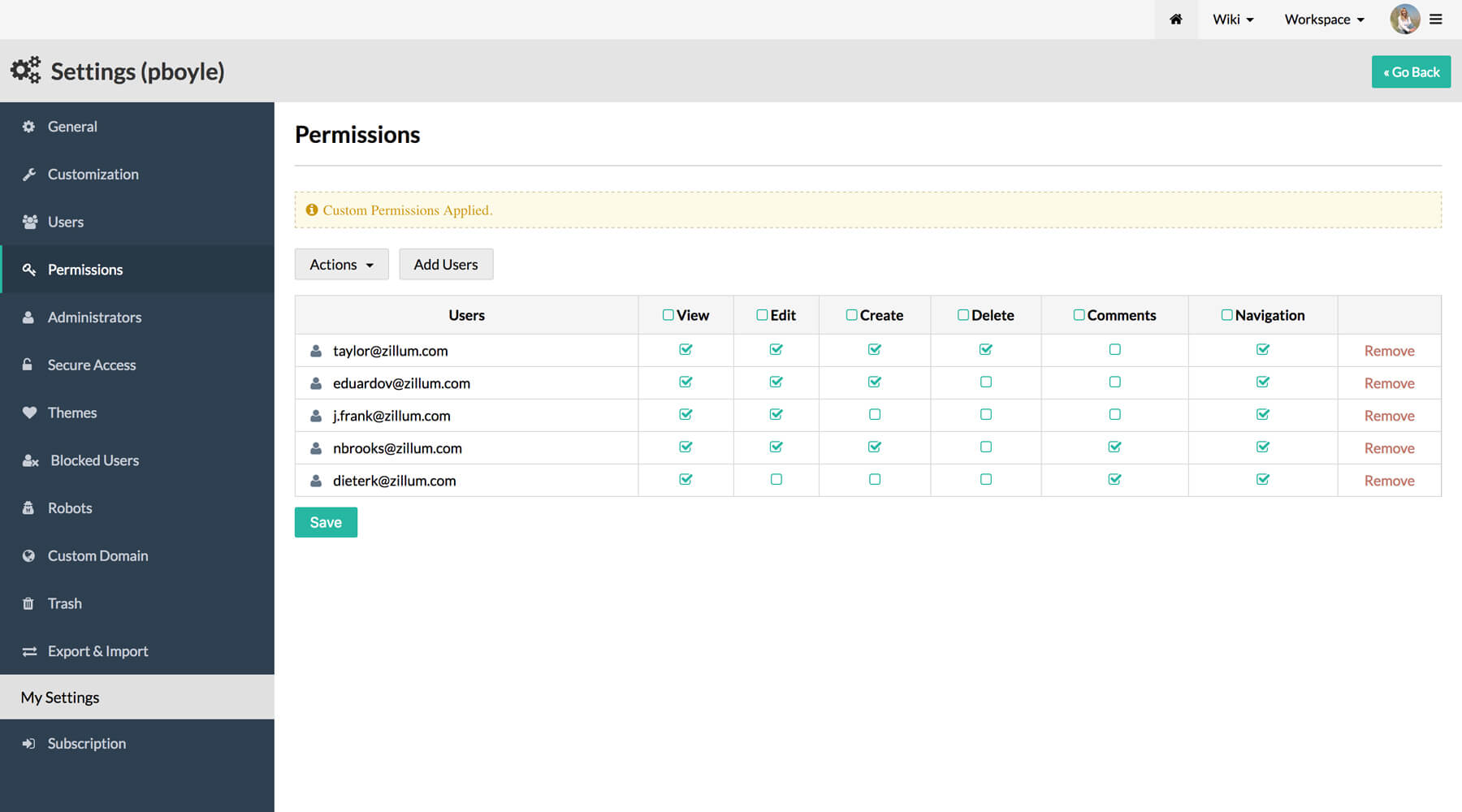


Рисунок 4 – ZOHO

Приложения могут использоваться как для создания нужного содержания через интернет, так и для манипуляций с файлами, созданными в других распространенных офисных пакетах. Zoho может читать и писать файлы в форматах Microsoft Office, OpenOffice.org и других систем.

Программы из пакета могут быть использованы индивидуально, но предлагаются также дополнительные преимущества интеграции для совместной работы. Файлы данных (документы) можно хранить на серверах Zoho, в также в нескольких партнеров Zoho типа box.net или Omnidrive, которые позволяют синхронизировать файлы на локальный компьютер. Хотя некоторые приложения, в частности Zoho CRM или Zoho Projects, имеют продвинутые возможности, доступные за дополнительную оплату, в общем позиция Zoho придерживается обязательства поддерживать бесплатный статус приложений начального уровня. Простая регистрация предоставляет доступ ко всем продуктам Zoho.

Zoho имеет открытый программный интерфейс к приложениям (API) для шести из своих продуктов: Zoho Writer, Zoho Sheet, Zoho Show, Zoho Creator, Zoho Meeting и Zoho Planner.

Это дает возможность сторонним разработчикам создавать приложения, использующие службы Zoho. Сама Zoho создает плагины для интеграции Zoho в Microsoft Word и Excel, а также как плагин браузера, может открывать документы и таблицы без установленных текстового процессора и электронных таблиц на компьютере.

В связи с выходом популярного iPhone, Zoho выпустила мобильную версию Zoho Writer, Sheet и Show, под названием iZoho, доступные на iZoho.com. Zoho также сделала свои приложения доступными на Facebook, после того как социальная сеть открыла свои API.

# **ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА БАЗЫ ДАННЫХ**

## **2.1 Обоснование выбора СУБД**

Система управления базой данных (СУБД) - важнейший компонент информационной системы. Для создания и управления информационной системой СУБД необходима в той же степени, как для разработки программы на алгоритмическом языке необходим транслятор. Основные функции СУБД:

• управление данными во внешней памяти (на дисках);

• управление данными в оперативной памяти;

• журнализация изменений и восстановление базы данных после сбоев; • поддержание языков БД (язык определения данных, язык манипулирования данными).

Систем управления базами данных очень много. Но реально используют перечисленные ниже:

• Реляционные;

• Key-value;

• Документные;

• Графовые;

• Колоночные;

• Time Series;

• Spatial;

• Search engines;

• Объектные;

Для небольшого pet-project может подойти любая встраиваемая или бесплатная СУБД. Коммерческий проект требует от СУБД соблюдения стандартов безопасности, производительности, скорости работы и т. д. Здесь возможностей бесплатных СУБД может не хватить.

Что будет храниться. Некоторые СУБД лучше работают с текстом, другие заточены под медиаконтент. Если знать, что будет храниться в БД, выбор СУБД станет проще. Объём. В технической документации для каждой СУБД указаны лимиты на объём одного файла, таблицы и других объектов. В случае, когда нужно хранить большие массивы данных, заранее проверьте способность системы управления базами данных «переваривать» такие объёмы.

Нагрузка и масштабируемость. Важно заранее предусмотреть возможный рост нагрузки, который неизбежно возникнет при расширении компании. Бесконечно наращивать оперативную память, процессоры и другие ресурсы не получится, поэтому выбирайте СУБД с прицелом на способность переносить текущие и двукратно увеличенные нагрузки, а также масштабироваться в будущем.

Файловая или серверная. База данных может работать по сети и локально. От этого зависит выбор СУБД. Они бывают файловые и серверные.

Отказоустойчивость и безопасность. Для некоторых проектов отказоустойчивость — критично важный параметр. Что будет, если упадёт БД банка, объяснять не нужно. Поэтому ориентируйтесь на реальные потребности проекта в уровне отказоустойчивости СУБД. Аналогично и с безопасностью — сертификаты, шифрование, дополнительные возможности нужны для крупных проектов. Небольшому достаточно и менее «навороченной» СУБД.

Стоимость. Существуют бесплатные опенсорсные решения, которые дают много возможностей, но требуют самостоятельной поддержки. Есть платные СУБД с поддержкой от разработчика. Выбор зависит от бюджета проекта. Далее рассмотрим два типа связки СУБД + приложение для взаимодействия с СУБД и их особенности.

## **СУБД – MySQL + phpMyAdmin**

**MySQL** — свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB.

Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации.

MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertrigoServ.

Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц.

Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.

**phpMyAdmin** — веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер и не только осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд.

## **СУБД - Microsoft SQL Server + MS SSMS**

**Microsoft SQL Server** — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft.

Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

**Microsoft SQL Server Management Studio** (MS SSMS) — утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера. Главным инструментом SQL Server Management Studio является Object Explorer, который позволяет пользователю просматривать, извлекать объекты сервера, а также полностью ими управлять. Также есть SQL Server Management Studio Express для Express версии сервера, которая является бесплатной. Однако в ней нет поддержки ряда компонентов (Analysis Services, Integration Services, Notification Services, Reporting Services) и SQL Server 2005 Mobile Edition. Начиная с версии 16.5.3 пакет SSMS выделен в отдельный обновляемый продукт, доступный для скачивания на сайте Microsoft. Текущая доступная версия SSMS 18.4 (15.0.18206.0) (general availability) поддерживает MS SQL server начиная с версии 2008 по 2019.

Поскольку, создаваемая база данных в дальнейшем будет использована в разработке приложения для Windows на платформе WPF (Windows Presentation Foundation, была выбрана связка Microsoft SQL Server + MS SSMS, для обеспечения более удобного взаимодействия между используемыми инструментами разработки.

## **2.2 Разработка инфологической модели**

Инфологическая модель – это потоки информации, сущности и связи данной области. В такой модели указываются связи между сущностями данной предметной области. Представлена на (рисунке 1).

Сущность – это любой объект, отличающийся от другого, информацию о котором необходимо сохранить.

Связь – это ассоциирование нескольких сущностей с целью отыскания одних из них по значениям других.

База данных может содержать неограниченное количество сущностей и такое же количество связей между ними, что определяет сложность инфологических моделей.

Атрибут – это характеристика сущности. Это может быть числовой характеристикой, классификацией, идентификацией. Его наименование должно быть уникальным для конкретного типа сущностей и может совпадать с атрибутами других сущностей.

Ключ представляет собой минимальное количество атрибутов, с помощью которого можно отыскать необходимый экземпляр сущности.

Цель инфологического моделирования – обеспечить оптимальные способы сбора и представления информации, хранимой в базе данных.

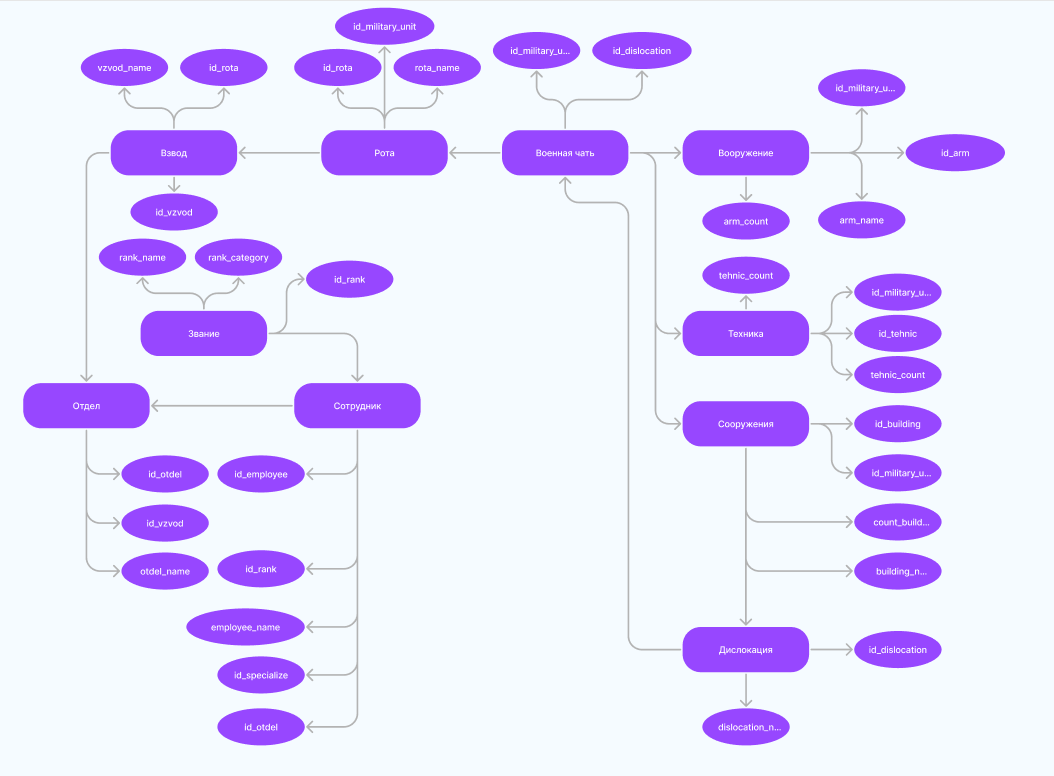


Рисунок 5 – Концептуальная модель

## **2.3 Нормализация базы данных**

Нормализация — это процесс организации данных в базе данных, включающий создание таблиц и установление отношений между ними в соответствии с правилами, которые обеспечивают защиту данных и делают базу данных более гибкой, устраняя избыточность и несогласованные зависимости.

Избыточность данных приводит к непродуктивному расходованию свободного места на диске и затрудняет обслуживание баз данных. Например, если данные, хранящиеся в нескольких местах, потребуется изменить, в них придется внести одни и те же изменения во всех этих местах.

**Первая нормальная форма**

• Устранение повторяющихся групп в отдельных таблицах;

• Создание отдельных таблиц для каждого набора связанных данных;

• Идентификация каждого набора связанных данных с помощью первичного ключа;

**Вторая нормальная форма**

• Создание отдельных таблиц для наборов значений, относящихся к нескольким записям;

• Связь этих таблиц с помощью внешнего ключа;

**Третья нормальная форма**

• Устранение полей, не зависящих от ключа;

Значения, входящие в запись и не являющиеся частью ключа этой записи, не принадлежат таблице (рисунок 2).

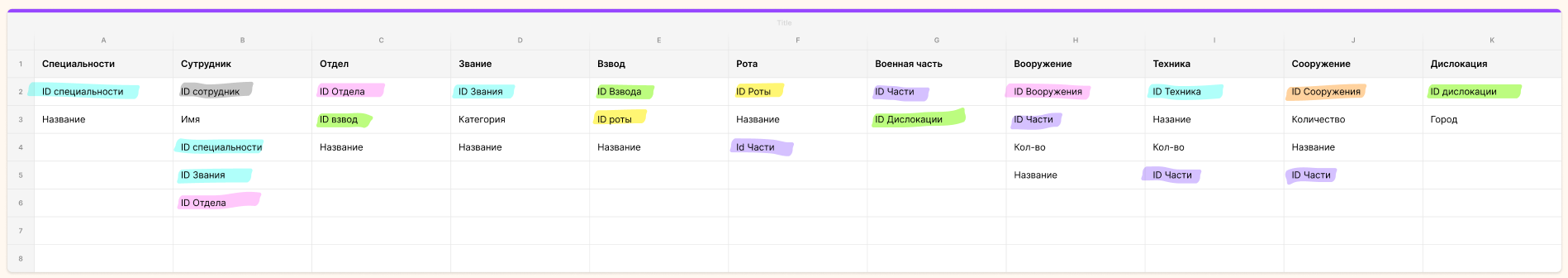


Рисунок 6 – Таблица данных

## **2.4 Разработка даталогической модели базы данных**

Даталогическое проектирование сводится к представлению инфологической модели в терминах выбранной системы управления базами данных (СУБД), т. е. даталогическая модель описывает собственно данные (информацию, которая будет записана в памяти компьютера) и связи между данными. Каждая СУБД опирается на определенную модель данных.

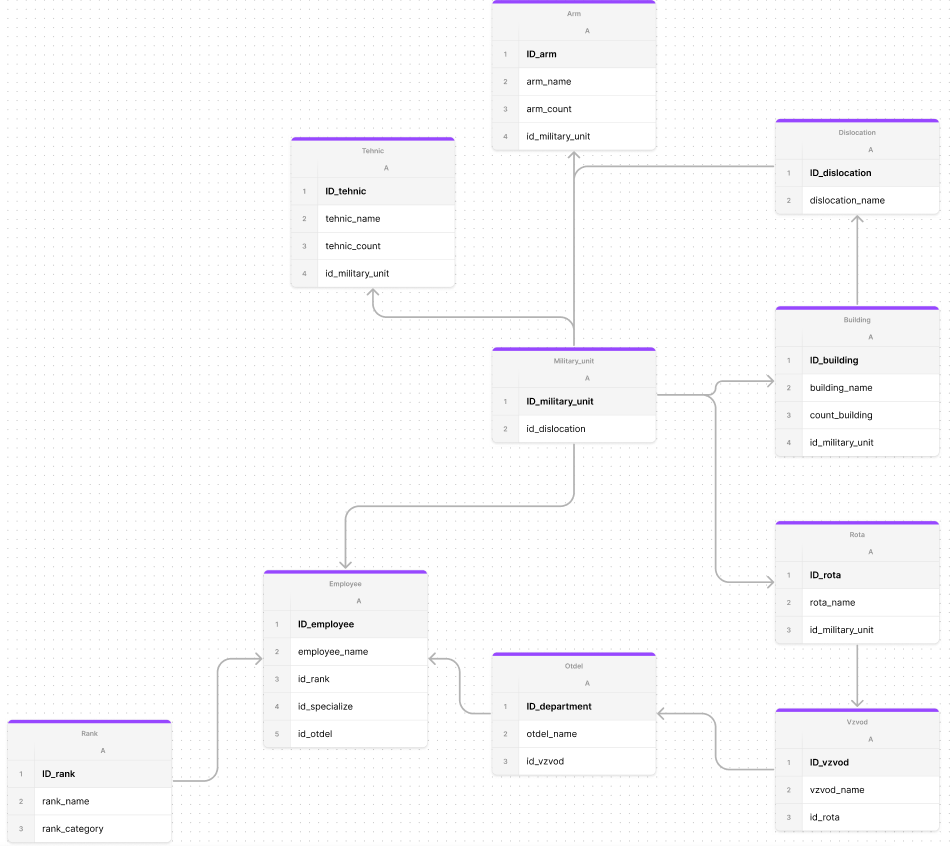


Рисунок 7 – Даталогическая модель

## **2.5 Инициализация базы данных**

Таблицы:

*Звание*

CREATE TABLE [rank] (

[id\_rank] INT CONSTRAINT [PK\_rank\_id] PRIMARY KEY IDENTITY(1, 1),

[rank\_name] NVARCHAR(100) NOT NULL,

[rank\_category] NVARCHAR(100) NOT NULL,

);

*Дислокация*

CREATE TABLE [dislocation] (

[id\_dislocation] INT CONSTRAINT [PK\_dislocation\_id] PRIMARY KEY

IDENTITY(1, 1),

[dislocation\_name] NVARCHAR(100) NOT NULL,

);

*Сооружения*

CREATE TABLE [building] (

[id\_building] INT CONSTRAINT [PK\_building\_id] PRIMARY KEY

IDENTITY(1, 1),

[building\_name] NVARCHAR(100) NOT NULL,

[count\_building] INT,

[id\_military\_unit] INT CONSTRAINT [FK\_military\_unit\_building\_id]

FOREIGN KEY REFERENCES [military\_unit]([id\_military\_unit]),

);

*Военная часть*

CREATE TABLE [military\_unit] (

[id\_military\_unit] INT CONSTRAINT [PK\_military\_unit\_id] PRIMARY

KEY IDENTITY(1, 1),

[id\_dislocation] INT CONSTRAINT [FK\_dislocation\_id] FOREIGN KEY

REFERENCES [dislocation]([id\_dislocation])

);

*Техника*

CREATE TABLE [tehnic] (

[id\_tehnic] INT CONSTRAINT [PK\_tehnic\_id] PRIMARY KEY IDENTITY(1, 1),

[tehnic\_name] NVARCHAR(100) NOT NULL,

[tehnic\_count] INT NOT NULL,

[id\_military\_unit] INT CONSTRAINT [FK\_military\_unit\_id] FOREIGN

KEY REFERENCES [military\_unit]([id\_military\_unit]),

);

*Вооружение*

CREATE TABLE [arm] (

[id\_arm] INT CONSTRAINT [PK\_arm\_id] PRIMARY KEY IDENTITY(1, 1),

[arm\_name] NVARCHAR(100) NOT NULL,

[arm\_count] INT NOT NULL,

[id\_military\_unit] INT CONSTRAINT [FK\_military\_unit\_arm\_id]

FOREIGN KEY REFERENCES [military\_unit]([id\_military\_unit]),

);

*Рота*

CREATE TABLE [rota] (

[id\_rota] INT CONSTRAINT [PK\_rota\_id] PRIMARY KEY IDENTITY(1, 1),

[rota\_name] NVARCHAR(100) NOT NULL,

[id\_military\_unit] INT CONSTRAINT [FK\_military\_unit\_rota\_id]

FOREIGN KEY REFERENCES [military\_unit]([id\_military\_unit]),

);

*Взвод*

CREATE TABLE [vzvod] (

[id\_vzvod] INT CONSTRAINT [PK\_vzvod\_id] PRIMARY KEY IDENTITY(1, 1),

[vzvod\_name] NVARCHAR(100) NOT NULL,

[id\_rota] INT CONSTRAINT [FK\_rota\_vzvod\_id] FOREIGN KEY

REFERENCES [rota]([id\_rota]),

);

*Отдел*

CREATE TABLE [otdel] (

[id\_otdel] INT CONSTRAINT [PK\_otdel\_id] PRIMARY KEY IDENTITY(1, 1),

[otdel\_name] NVARCHAR(100) NOT NULL,

[id\_vzvod] INT CONSTRAINT [FK\_vzvod\_otdel\_id] FOREIGN KEY

REFERENCES [vzvod]([id\_vzvod]),

);

*Сотрудник*

CREATE TABLE [employee] (

[id\_employee] INT CONSTRAINT [PK\_employee\_id] PRIMARY KEY IDENTITY(1, 1),

[employee\_name] NVARCHAR(100) NOT NULL,

[id\_rank] INT CONSTRAINT [FK\_rank\_id] FOREIGN KEY REFERENCES [rank]([id\_rank]),

[id\_specialize] INT CONSTRAINT [FK\_specialize\_employee\_id] FOREIGN KEY REFERENCES [specialize]([id\_specialize]),

[id\_otdel] INT CONSTRAINT [FK\_otdel\_employee\_id] FOREIGN KEY REFERENCES [otdel]([id\_otdel]),

);

*Специализация*

CREATE TABLE [specialize] (

[id\_specialize] INT CONSTRAINT [PK\_specialize\_id] PRIMARY KEY IDENTITY(1, 1),

[specialize\_name] NVARCHAR(100) NOT NULL,

);

## **2.6 Заполнение таблиц данными**

Таблицы:

*Звание*

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Рядовой', N'Сержанский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Ефрейтор', N'Сержанский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Прапорщик', N'Сержанский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Сержант', N'Сержанский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Старшин', N'Сержанский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Лейтенант', N'Офицерский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Капитан', N'Офицерский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Майор', N'Офицерский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Подполковник', N'Офицерский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Полковник', N'Офицерский состав');

INSERT INTO [rank] ([rank\_name], [rank\_category]) VALUES(N'Генералы', N'Офицерский состав');

*Дислокация*

INSERT INTO [dislocation] ([dislocation\_name])

VALUES(N'Москва');

INSERT INTO [dislocation] ([dislocation\_name]) VALUES(N'Челябинск');

*Сооружения*

INSERT INTO [building] ([building\_name], [count\_building], [id\_military\_unit]) VALUES(N'Штаб', 1, 2);

INSERT INTO [building] ([building\_name], [count\_building], [id\_military\_unit]) VALUES(N'казарма', 2, 2);

*Военная часть*

INSERT INTO [military\_unit] ([id\_dislocation]) VALUES(2);

INSERT INTO [military\_unit] ([id\_dislocation]) VALUES(1);

*Техника*

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'Тягач', 5, 1);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'БМП', 10, 1);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'Танк', 2, 1);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'Тягач', 15, 2);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'БМП', 40, 2);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'Танк', 11, 2);

*Вооружение*

INSERT INTO [arm] ([arm\_name], [arm\_count], [id\_military\_unit]) VALUES(N'Карабин', 200, 2);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'Артиллерия', 70, 2);

INSERT INTO [tehnic] ([tehnic\_name], [tehnic\_count], [id\_military\_unit]) VALUES(N'Ракетное вооружение', 30, 2);

*Рота*

INSERT INTO [rota] ([rota\_name], [id\_military\_unit]) VALUES(N'Мотострелковая', 2);

INSERT INTO [rota] ([rota\_name], [id\_military\_unit]) VALUES(N'Десантно-штурмовая', 2);

INSERT INTO [rota] ([rota\_name], [id\_military\_unit]) VALUES(N'Танковая', 2);

INSERT INTO [rota] ([rota\_name], [id\_military\_unit]) VALUES(N'Разведывательная', 2);

INSERT INTO [rota] ([rota\_name], [id\_military\_unit]) VALUES(N'Ремонтная', 2);

INSERT INTO [rota] ([rota\_name], [id\_military\_unit]) VALUES(N'Связь', 2);

*Взвод*

INSERT INTO [vzvod] ([vzvod\_name], [id\_rota]) VALUES(N'Противотанковый', 12);

*Отдел*

INSERT INTO [otdel] ([otdel\_name], [id\_vzvod]) VALUES(N'Связь', 2);

INSERT INTO [otdel] ([otdel\_name], [id\_vzvod]) VALUES(N'Обеспечение', 2);

*Сотрудник*

INSERT INTO [employee] ([employee\_name], [id\_rank], [id\_specialize], [id\_otdel]) VALUES(N'Горбачев Роман Андреевич', 2, 1, 1);

*Специализация*

INSERT INTO [specialize] ([specialize\_name]) VALUES(N'Врач');

INSERT INTO [specialize] ([specialize\_name]) VALUES(N'Инженер');

INSERT INTO [specialize] ([specialize\_name]) VALUES(N'Летчик');

INSERT INTO [specialize] ([specialize\_name]) VALUES(N'Солдат');

INSERT INTO [specialize] ([specialize\_name]) VALUES(N'Радист');

INSERT INTO [specialize] ([specialize\_name]) VALUES(N'Картограф');

**2.7 Дополнительные инструменты**

**Figma** — онлайн-сервис для разработки интерфейсов и прототипирования с возможностью организации совместной работы в режиме реального времени.

Сервис доступен по подписке, предусмотрен бесплатный тарифный план для одного пользователя. Имеются офлайн-версии для Windows, macOS. Реализована интеграция с корпоративным мессенджером Slack и инструментом прототипирования Framer.

Используется как для создания упрощённых прототипов интерфейсов, так и для детальной проработки дизайна интерфейсов мобильных приложений, веб-сайтов, корпоративных порталов.

# **ГЛАВА 3. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ УПРАВЛЕНИЯ БАЗОЙ ДАННЫХ**

## **3.1 Обоснование выбора среды разработки**

**Microsoft Visual Studio** — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментов.

Данные продукты позволяют разрабатывать как консольные приложения, так и игры и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, WPF, UWP а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, .NET Core, .NET, MAUI, Xbox, Windows Phone .NET Compact Framework и Silverlight.

После покупки компании Xamarin корпорацией Microsoft появилась возможность разработки IOS и Android программ.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня.

Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных.

Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

Учитывая все вышесказанное, я выбрал данный инструмент для реализации приложения.

В качестве технологии реализации была выбрана платформа “WPF” в cвязке с пакетом “Entity framework”

**Windows Presentation Foundation (WPF)** — аналог WinForms, система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML. WPF предустановлена в Windows Vista (.NET Framework 3.0), Windows 7 (.NET Framework 3.5 SP1), Windows 8 (.NET Framework 4.0 и 4.5), Windows 8.1 (.NET Framework 4.5.1) и Windows 10 (.NET Framework 4.7). С помощью WPF можно создавать широкий спектр как автономных, так и запускаемых в браузере приложений.

В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление.

Графической технологией, лежащей в основе WPF, является DirectX, в отличие от Windows Forms, где используется GDI/GDI+. Производительность WPF выше, чем у GDI+ за счёт использования аппаратного ускорения графики через DirectX. Также существует урезанная версия CLR, называющаяся WPF/E, она же известна как Silverlight.

**ADO.NET Entity Framework (EF)** — объектно-ориентированная технология доступа к данным, является object-relational mapping (ORM) решением для .NET Framework от Microsoft. Предоставляет возможность взаимодействия с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL. Для облегчения построения web-решений используется как ADO.NET Data Services (Astoria), так и связка из Windows Communication Foundation и Windows Presentation Foundation, позволяющая строить многоуровневые приложения, реализуя один из шаблонов проектирования MVC, MVP или MVVM.

## **3.2 Проектирование приложения**

Для проектирования структуры и дизайна интерфейса выбрал приложение – “Figma”, которое указывал ранее.

Было отрисовано два основных экрана приложения, для представления архитектуры приложения.

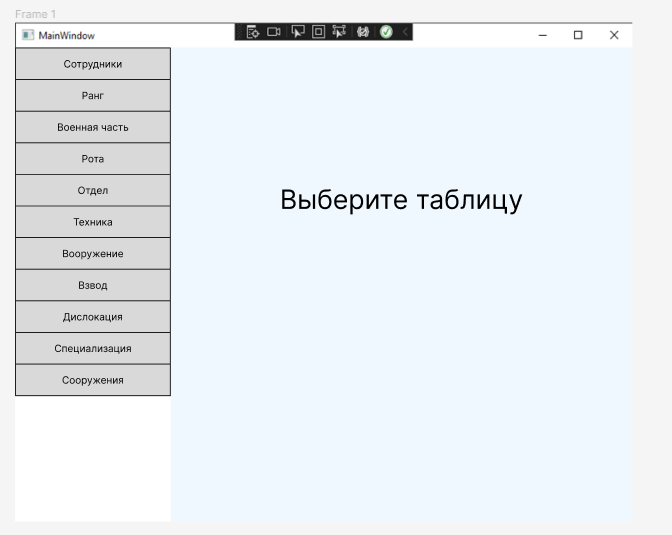


Рисунок 8 – Главный экран приложения

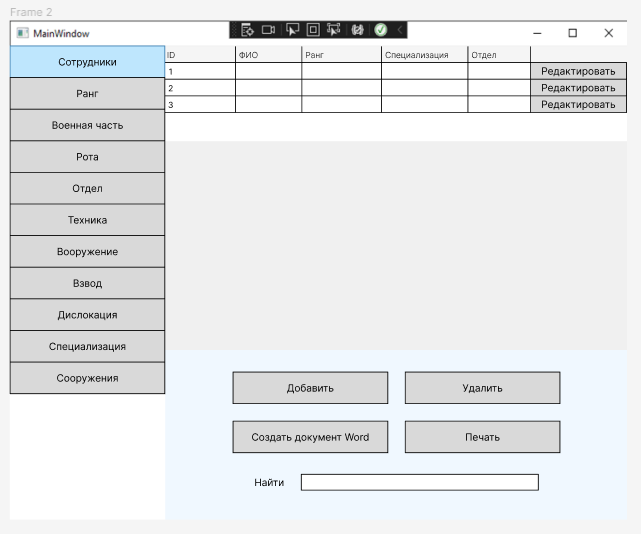


Рисунок 9 – Экран выбранной таблицы

Также были отрисованы состояния кнопок при взаимодействии пользователя с ними.

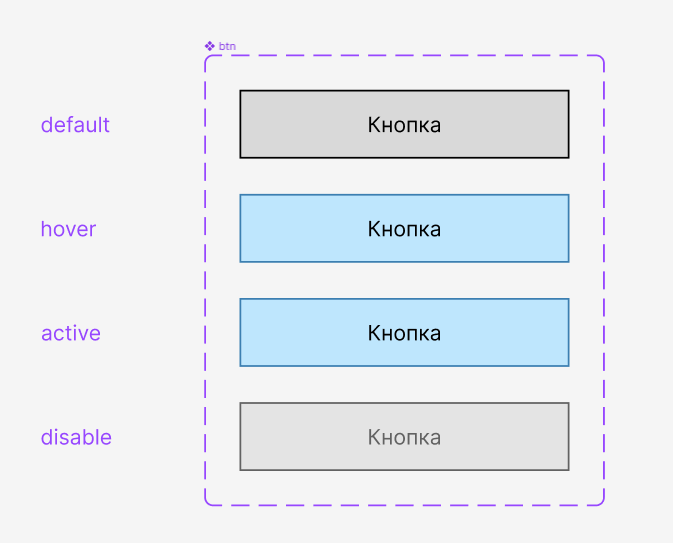


Рисунок 10 – Состояния кнопок

## **3.3 Разработка приложения**

Для реализации подключения базы данных к приложению, был создан класс, который содержит в себе статический метод получения данных из базы данных GetContext().

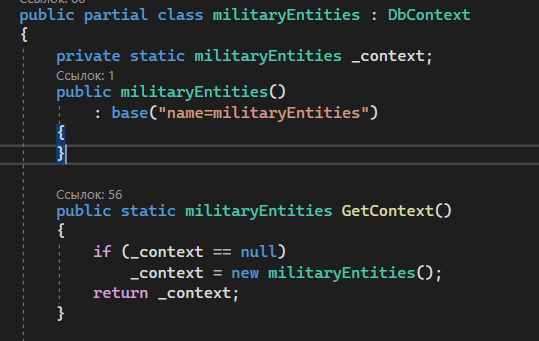


Рисунок 11 – Класс militaryEntities

Также были созданы классы под каждую таблицу.

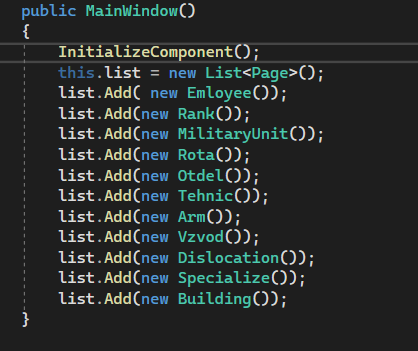


Рисунок 12 – Список табличных классов

Добавление новых данных в базу реализовано по средствам открытия нового WPF окна с полями для ввода данных.

Создаем новый экземпляр класса, в котором будет содержаться информация о текущем объекте. Связываем поля для ввода с полями в таблице.

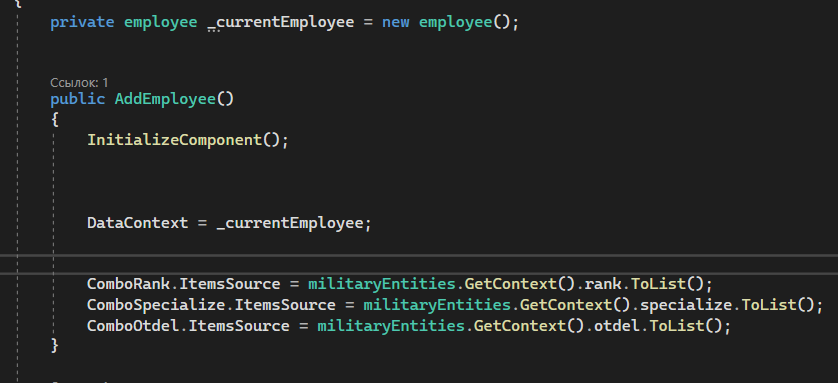


Рисунок 13 – Получение данных о новом объекте

Также реализуем проверку на отсутствие пустых полей при отправке данных.

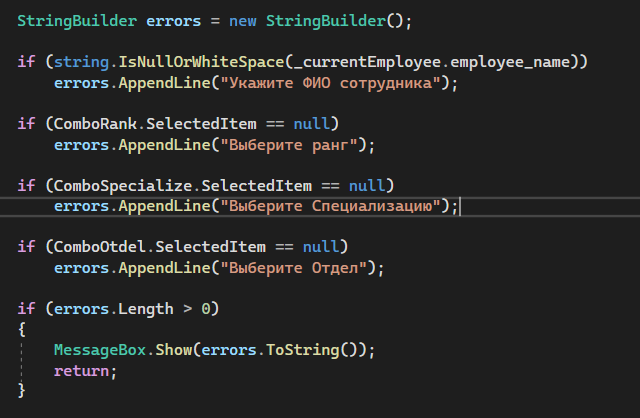


Рисунок 14 – Валидация полей

Создаем кнопку сохранения и реализуем событие нажатия на эту кнопку. Заносим данные о текущем объекте в общую таблицу с данными и сохраняем изменения с помощью специального метода из библиотеки Entity. Все эти действия оборачиваем в конструкцию try-catch, для обработки возможных ошибок, поскольку данные операции с базой данных являются потенциально опасными.

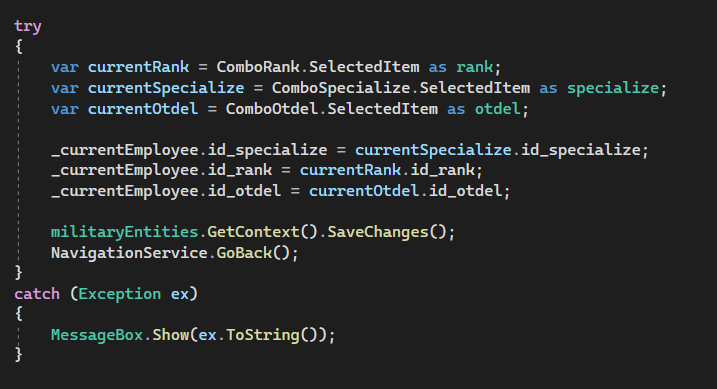


Рисунок 15 – Сохранение данных о новом объекте

Удаление данных из базы происходит путем выделения строки с удаляемым объектом и нажатия на кнопку “Удалить”.

При нажатии на кнопку срабатывает событие, в котором получаем выбранный объект. После вызываем диалоговое окно с вопросом об удалении данных из таблицы, если пользователь соглашается, то вызывается метод удаления данных, после этого сохраняем все изменения и заново отрисовываем нашу таблицу. Все эти операции также оборачиваем в конструкцию “try-catch”.

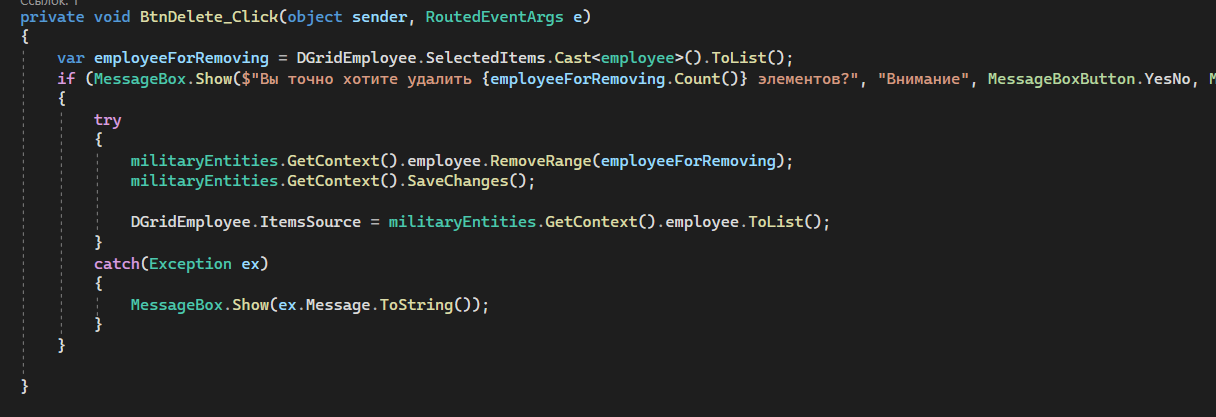


Рисунок 16 – Удаление объекта из таблицы

Реализация функции поиска представлена на рисунке 16.

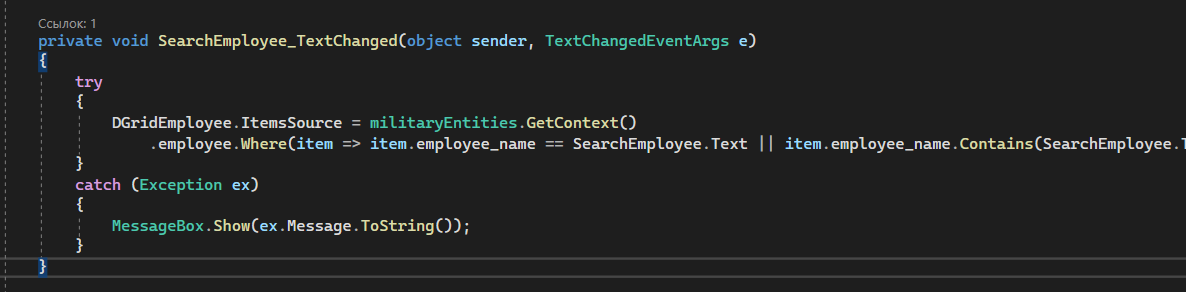


Рисунок 17 – Функция поиска

Также были реализованы функции генерации отчета word и вывода таблицы на печать.

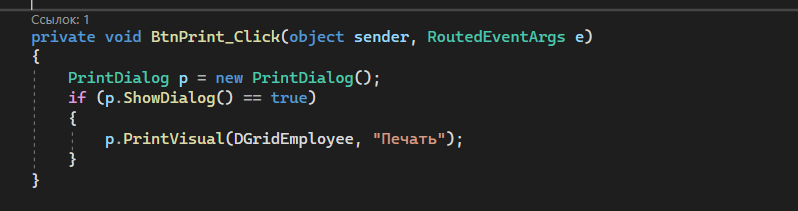


Рисунок 18 – Функция печати

Генерация отчета реализуется с помощью специальной кнопки, которая обращается к созданному классу “Word” (рисунок 18), который добавляет информацию в документ, создавая новый параграф.

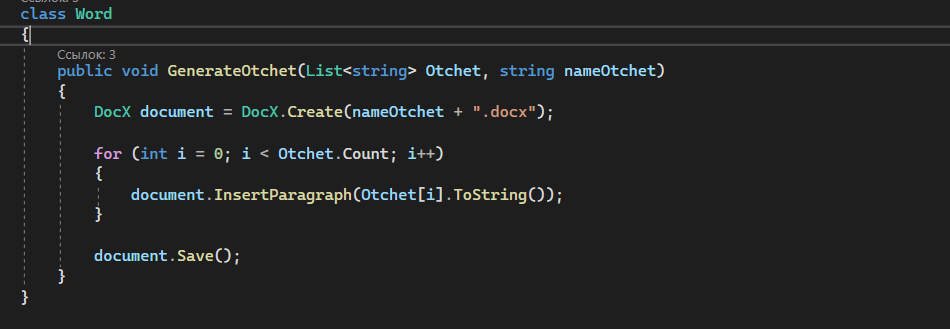


Рисунок 19 – Класс “Word”

Приложение представляет из себя удобный модульный интерфейс, который состоит из так называемых WPF страниц, которые изменяют свое отображение на экране в зависимости от действий пользователя.

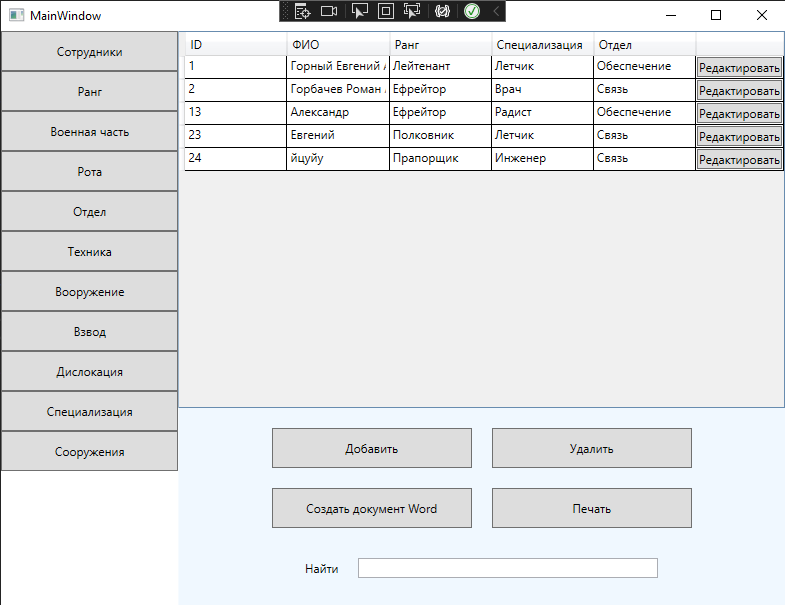


Рисунок 20 – Приложение(общ. вид)

Данные подгружаются на страницы и импортируются в таблицы из базы данных, которая подключается к приложению с помощью специального кода.

Благодаря возможностям Entity Framework, реализованы такие функции управления базой, как – добавление, удаление, редактирование всех данных, через специальные кнопки и другие элементы управления.

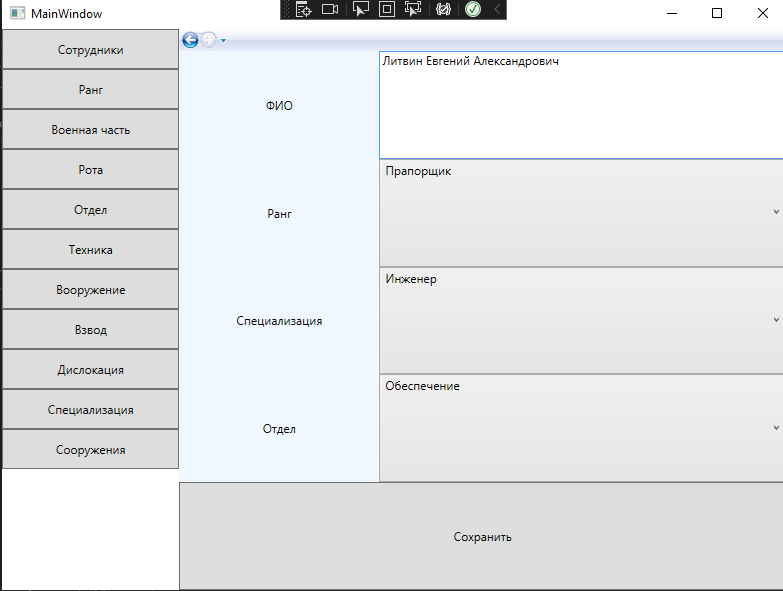


Рисунок 21 – Приложение(Добавление данных в таблицу)

В качестве дополнительных функций была реализована возможность вывода каждой таблицы данных на печать и создание word – отчета, через специализированную библиотеку Xceed – сервис для работы с документами word.

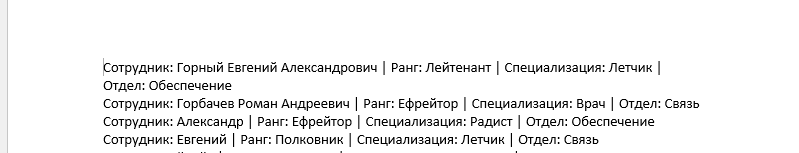


Рисунок 22 – Приложение(Генерация отчета)

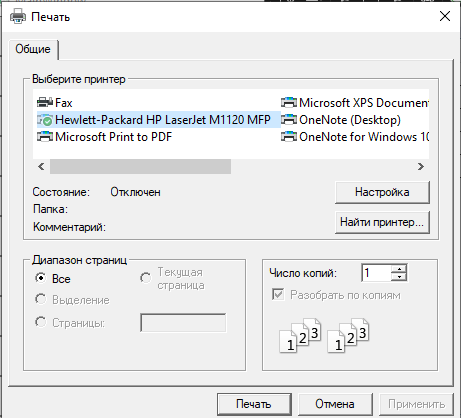


Рисунок 23 – Приложение(Вывод таблиц на печать)

Помимо этого, была реализована функция поиска данных по конкретным таблицам, для более удобного взаимодействия с данными.

Напоследок, реализовал небольшую валидацию данных, для того чтобы избежать отправку пустых форм.

# **ЗАКЛЮЧЕНИЕ**

Рассмотрим результаты проведенной работы, целью которого являлось создание базы данных и приложения для реализации управления этой базой.

В перечень выполненных работ попадает:

• Анализ действующих решений;

• Анализ возможных решений для данной задачи;

• Проектирование и создание реляционной базы данных;

• Проектирование и создание приложения для управления базой данных;

Все эти работы в комплексе помогли научиться создавать правильные с точки зрения логики построения структуры базы данных, разобраться в многообразии инструментов проектирования и программирования баз данных. Освоить для себя новые инструменты для взаимодействия с данными.

На данном этапе развития функционал приложения логически завершён.

# **СПИСОК ИСПОЛЬЗУЕМЫХ РЕСУРСОВ**

1. «Введение в системы баз данных» К. Дж. Дейт.
2. «MySQL по максимуму» Бэрон Шварц, Вадим Ткаченко, Петр Зайцев.
3. WPF: Windows Presentation Foundation в .NET 4.5. Мэтью Макдональд.
4. Основы Windows Presentation Foundation. Крис Андерсон.