

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
Рязанский государственный радиотехнический университет  
имени В.Ф. Уткина

Кафедра ЭВМ  
К защите  
Руководитель работы:

\_\_\_\_\_  
дата, подпись

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
К КУРСОВОЙ РАБОТЕ**

по дисциплине

**«Клиент серверные приложения баз данных»**

Тема:

**«Разработка информационной системы общежития»**

Выполнил студент группы 145

Жупин С.Ю.

\_\_\_\_\_  
дата сдачи на проверку, подпись

Руководитель работы  
ассистент каф. ЭВМ  
Баранова С.Н.

\_\_\_\_\_  
оценка

\_\_\_\_\_  
дата защиты, подпись

Рязань 2024

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	4
1 Выявление задач автоматизации .....	5
1.1 Описание автоматизированной области .....	5
1.2 Обоснование актуальности разработки .....	6
1.3 Постановка задачи .....	6
1.4 Разработка архитектуры ИС .....	7
2 Разработка серверной части ИС .....	9
2.1 Инфологическое проектирование БД .....	9
2.1.1 Требуемая информация .....	9
2.1.2 Выделение сущностей .....	10
2.1.3 Выделение связей .....	11
2.1.4 Построение ER диаграммы (Рисунки 2-11) .....	11
2.2 Дatalogическое проектирование БД .....	15
2.2.1 Формирование предварительных отношений .....	15
2.2.2 Распределение атрибутов по отношениям .....	16
2.2.3 Проверка отношений на БКНФ .....	18
2.3 Разработка объектов поддержания целостности данных .....	22
2.3.1 Разработка правил, умолчаний и типов .....	22
2.3.2 Разработка триггеров .....	25
2.3.3 Разработка процедур .....	25
3 Разработка клиентской части ИС .....	26
3.1 Разработка прототипа интерфейса пользователя .....	26
3.2 Реализация интерфейса пользователя .....	28
4 Тестирование основных функций приложения .....	31
ЗАКЛЮЧЕНИЕ .....	32
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ .....	33
ПРИЛОЖЕНИЕ А: сценарий создания объектов БД .....	34
ПРИЛОЖЕНИЕ Б: сценарий заполнения таблиц БД .....	45

ПРИЛОЖЕНИЕ В:исходный код клиентского приложения .....	47
--	----

## **ВВЕДЕНИЕ**

На сегодняшний день информационные системы (далее ИС) применяются во множестве различных сферах деятельности человека. ИС позволяют автоматизировать и ускорить процессы обработки, передачи и создания различных данных тем самым позволяя оптимизировать различные прикладные процессы. Благодаря цифровому представлению данных можно избавиться от большого количества бумажных архивов, значительно ускорить передачу информации и упростить обработку информации, а также обеспечить предоставление информации в удобной форме.

В данном курсовом проекте будет создана информационная система общежития, которая будет состоять из сервера состоящем из базы данных MS SQL Server и клиентского приложения на платформе Windows. Данная ИС позволит хранить, добавлять и обрабатывать информацию, требуемую для администрирования общежития в удобном виде, а также позволит делать это удаленно.

# **1 Выявление задач автоматизации**

## **1.1 Описание автоматизированной области**

Необходимо спроектировать ИС для студенческого общежития РГРТУ, предоставляющей проживание для студентов как самого РГРТУ, так и других вузов и профессиональных училищ. ИС. ИС разрабатывается для одного общежития, но при необходимости ее можно будет улучшить для администрирования сразу множества общежитий, например, в пределах одного студенческого городка.

Система хранит информацию о проживающих в общежитии студентах с указанием их персональных данных, местом их обучения, а также о тарифе и долгах за проживание.

Система хранит информацию о структуре общежития, такую как список блоков с указанием их типов и назначениях.

Система хранит информацию о всех помещениях общежития с указанием типа комнаты, ее принадлежности к блоку и количеству доступных для заселения мест.

Система хранит информацию о работниках общежития (вахтеры, уборщики, слесари и др.) с указанием их персональных данных и занимаемой должности.

Система хранит информацию о существующих в общежитии должностях. Должность определяет базовый оклад работника.

Ведется отслеживание различного рода инвентаря (мебель, фурнитура и т.д.) и хранение информации о нем, такой как название и описание, стоимость, дата поставки и комната его размещения.

При заселении нового или переселении уже существующего в системе студента администратор общежития вносит соответствующие изменения.

Проживающие в общежитии могут создать заявку с жалобой или предложением, на которую администратором будет назначен ответственный за ее решение из работников общежития.

При устройстве на работу в общежитии администратор вносит нового работника в систему с указанием его персональных данных и должности.

Администратор назначает дежурных по блокам общежития из работников, а также вахтеров на вахты из ранее размеченного расписания вахт.

Администратор составляет расписание вахт в общежитии с указанием даты захода и длительностью смены.

### **1.2 Обоснование актуальности разработки**

При изучении предметной области было замечено что многая информация о проживающих в общежитии хранится в бумажном виде, а также отсутствует доступ о данных студентах из деканатов образовательных организаций, из-за этого возникают ситуации беготни с бумагами. Также хранение информации в бумажном виде сильно ограничивает скорость ее обработки и увеличивает шансы допущения ошибок, которые могут нарушить целостность данных.

Также электронное представление данных значительно оптимизирует работу администратора общежития что позволит повысить производительность труда.

### **1.3 Постановка задачи**

Для данной предметной области основными задачами автоматизации являются:

- Автоматизация операций заселения/переселения проживающих.
- Автоматизация операций с заявками и предложениями.
- Автоматизация инвентаризации.
- Автоматизация назначения работ для работников общежития.
- Удобный и быстрый поиск и отображение информации.
- Удаленный просмотр информации.

#### **1.4 Разработка архитектуры ИС**

Для данной ИС была выбрана двухуровневая клиент-серверная архитектура. Она позволяет значительно сэкономить на серверной части, так как в отличие от распределенных систем здесь требуется только один физический сервер.

Из других преимуществ такая архитектура способна поддерживать целостность данных на обоих уровнях и равномерно распределять нагрузку между клиентом и сервером. Также такая архитектура позволяет организовать централизованную защиту данных, т.к. все данные будут храниться на сервере. Клиентское приложение в свою очередь будет отправлять запросы на сервер и выводить пользователю полученную информацию в форматированном виде. Также с клиента можно отправлять параметризованные запросы с различными ограничениями на добавление/изменение/удаление данных, которые позволят исключить нарушение целостности данных и SQL-инъекции.

Централизованное хранилище данных является не только преимуществом, но и недостатком, так как при нарушении работы сервера все экземпляры приложений-клиентов потеряют доступ к данным, а также при несанкционированном доступе будет утечка сразу всех данных.

Схема архитектуры ИС представлена на рисунке 1. Виртуальный сервер БД физически располагается на устройстве-сервере, также сервер БД представляет часть слоя логики и выполняет большинство операций по обработке данных. Клиентское приложение содержит в себе остальную часть слоя логики который формирует запросы к серверу и слой представления данных полученных от сервера.

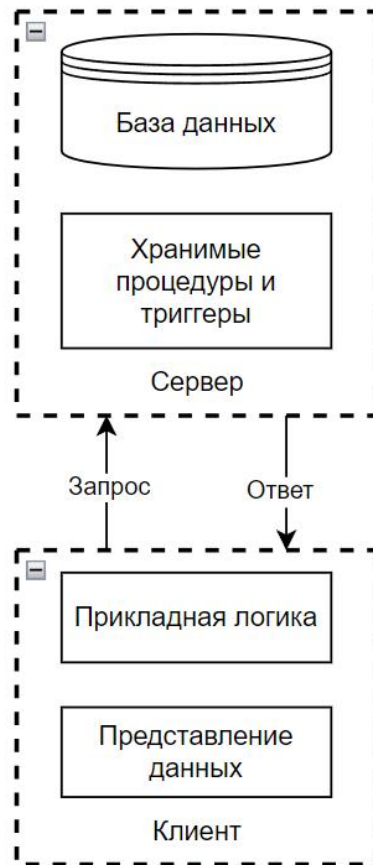


Рисунок 1 – Архитектура ИС



## **2 Разработка серверной части ИС**

### **2.1 Инфологическое проектирование БД**

#### **2.1.1 Требуемая информация**

Исходя из предметной области в БД требуется хранить следующую информацию:

##### **1. Информацию о проживающих:**

- ФИО,
- Серия и номер паспорта,
- Пол (муж/жен),
- Дата рождения,
- Номер телефона для связи,
- Учебное заведение, в котором он обучается,
- Комната проживания,
- Тариф, который зависит от льгот и места обучения.

##### **2. Информацию о комнате:**

- Тип комнаты,
- Номер блока, в котором она находится,
- Количество койко-мест, которое зависит от типа комнаты.

##### **3. Информацию о блоках:**

- Тип блока, который определяет какие типы проживающих рекомендуется заселять в блок,
- Принадлежность блока к крылу и этажу,
- Информацию о работнике являющийся дежурным по блоку.

##### **4. Информацию об инвентаре**

- Наименование и описание,
- Дата поставки,
- Стоимость,
- Номер комнаты размещения.

##### **5. Информацию о вахтах:**

- Тип вахты,
- Ставка почасового оклада за вахту в зависимости от типа вахты
- Дата и время заступления на вахту,
- Длительность вахты в часах
- Назначенный на вахту работник общежития.

#### **6. Информацию о работниках общежития:**

- ФИО,
- Серия и номер паспорта,
- Телефон для связи,
- Должность,
- Базовый оклад, зависящий от должности.

#### **7. Информацию о заявках –**

- Тема для удобного поиска и сортировки,
- Основной текст,
- Статус заявки (создана, закрыта, в процессе),
- Дата создания,
- Составитель из проживающих,

#### **2.1.2 Выделение сущностей**

Для хранения вышеуказанной информации необходимо выделить следующие сущности:

- Проживающий (Код проживающего)
- Тип проживающего (Название типа)
- Комната (Номер комнаты)
- Тип комнаты (Название типа)
- Блок (Номер блока)
- Тип блока (Название типа)
- Инвентарь (Наименование)
- Работник (Код работника)
- Должность (Название должности)

- Вахта (Номер вахты)
- Тип вахты (Название типа)
- Заявка (Номер заявки)

### 2.1.3 Выделение связей

Проживающий имеет тип проживающего;

Проживающий проживает в комнате;

Заявка составляется проживающим;

Комната имеет тип комнаты;

Комната содержит инвентарь;

Блок содержит комнаты;

Работник дежурит в блоке;

Работник дежурит на вахтах;

Работник занимает должность;

Вахта имеет тип вахты;

### 2.1.4 Построение ER диаграммы (Рисунки 2-11)

#### **Проживающий имеет тип**

- Проживающий имеет только один тип
- Проживающий обязательно имеет тип
- Один тип может иметь множество проживающих
- Тип не обязательно должен иметь проживающих



Рисунок 2 – Проживающий имеет тип

#### **Проживающий проживает в комнате**

- Проживающий живет только в одной комнате
- Проживающий обязательно должен проживать в комнате
- В одной комнате могут жить несколько проживающих
- В комнате не обязательно должны проживать

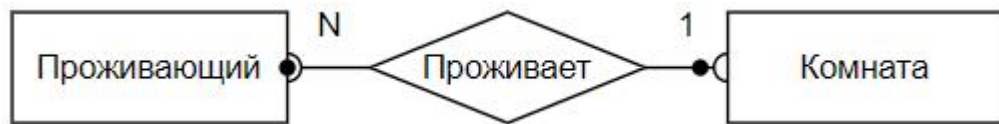


Рисунок 3 – Проживающий проживает в комнате

#### **Заявка составляется проживающим**

- Проживающий может составить множество заявок
- Проживающему не обязательно составлять заявки
- Одну заявку составляет только один проживающий
- Заявка обязательно содержит информацию о составителе

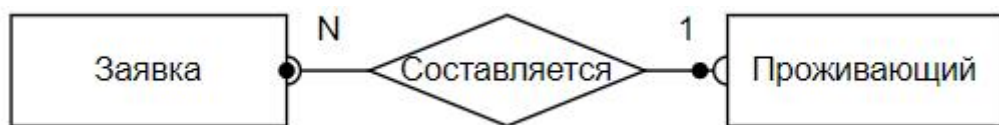


Рисунок 4 – Заявка составляется проживающим

#### **Комната имеет тип**

- Комната обязательно имеет тип
- Комната может иметь только один тип
- Одному типу может соответствовать множество комнат
- Типу не обязательно иметь соответствующую ему комнату



Рисунок 5 – Комната имеет тип

#### **Комната содержит инвентарь**

- В одной комнате может содержаться множество инвентаря
- В комнате может не быть инвентаря
- Инвентарь обязательно имеет комнату, в которой он размещается
- Единица инвентаря размещается только в одной комнате



Рисунок 6 – Комната содержит инвентарь

### **Блок содержит комнаты**

- В одном блоке может быть множество комнат
- Комната обязательно должна иметь блок, в котором она находится
- Комната может находиться только в одном блоке
- Блок необязательно должен иметь комнаты



Рисунок 7 – Блок содержит комнаты

### **Работник дежурит в блоке**

- В одном блоке может дежурить только один работник
- В блоке необязательно должны дежурить
- Работник может дежурить в множестве блоках
- Работнику необязательно дежурить



Рисунок 8 – Работник дежурит в блоке

### **Работник дежурит на вахтах**

- Работнику необязательно дежурить на вахте
- Работник может дежурить на нескольких вахтах
- На одну вахту назначается только один работник
- Вахте обязательно должен быть назначен работник



Рисунок 9 – Работник дежурит на вахтах

### **Работник занимает должность**

- Работник занимает только одну должность
- Работнику обязательно занимать должность
- На одной должности может состоять множество работников

– Должность может никто не занимать

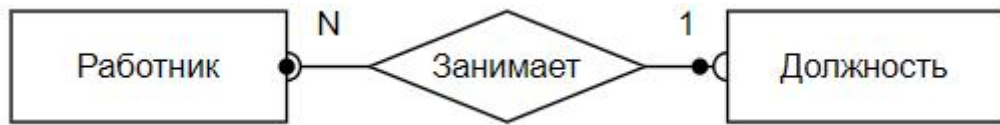


Рисунок 10 – Работник занимает должность

### Вахта имеет тип вахты

- Вахта имеет только один тип вахты
- Вахта обязательно имеет тип
- Одному типу может соответствовать множество вахт
- Тип может не иметь соответствующих ему вахт



Рисунок 11 – Вахта имеет тип вахты

**Полная ER диаграмма** представлена на рисунке 12.

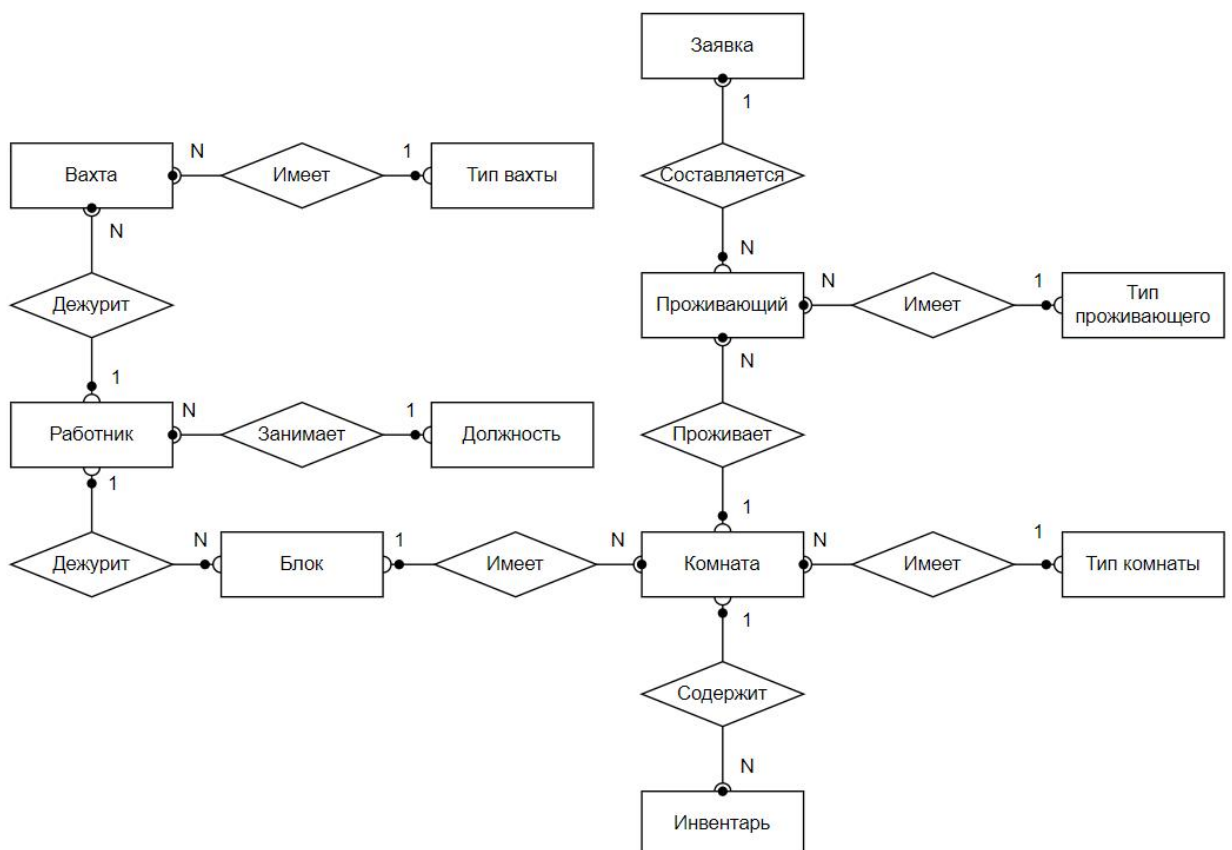


Рисунок 12 – ER диаграмма

## 2.2 Даталогическое проектирование БД

### 2.2.1 Формирование предварительных отношений

**Связь «проживающий имеет тип проживающего»** по правилу 2 формирует 2 отношения:

Проживающий(Проживающий, ТипПроживающего)

ТипПроживающего(ТипПроживающего)

**Связь «проживающий проживает в комнате»** по правилу 2 формирует 2 отношения:

Проживающий(Проживающий, ТипПроживающего, Комната)

Комната(Комната)

**Связь «Заявка составляется проживающим»** по правилу 2 формирует 2 отношения:

Заявка(Заявка, Проживающий)

Проживающий(Проживающий, ТипПроживающего)

**Связь «Комната имеет тип комнаты»** по правилу 2 формирует 2 отношения:

Комната(Комната, ТипКомнаты)

ТипКомнаты(ТипКомнаты)

**Связь «Комната содержит инвентарь»** по правилу 2 формирует 2 отношения:

Инвентарь(Инвентарь, Комната)

Комната(Комната, ТипКомнаты)

**Связь «Блок содержит комнаты»** по правилу 2 формирует 2 отношения:

Комната(Комната, ТипКомнаты, Блок)

Блок(Блок)

**Связь «Работник дежурит в блоке»** по правилу 5 формирует 3 отношения:

Дежурство(Блок, Работник)

Блок(Блок)

Работник(Работник)

**Связь «Работник дежурит на вахте»** по правилу 2 формирует 2 отношения:

Вахта(Вахта, Работник)

Работник(Работник)

**Связь «Работник занимает должность»** по правилу 2 формирует 2 отношения:

Работник(Работник, Должность)

Должность(Должность)

**Связь «Вахта имеет тип вахты»** по правилу 2 формирует 2 отношения:

Вахта(Вахта, Работник, ТипВахты)

ТипВахты(ТипВахты)

**Полная предварительная диаграмма отношений** представлена на рисунке 14

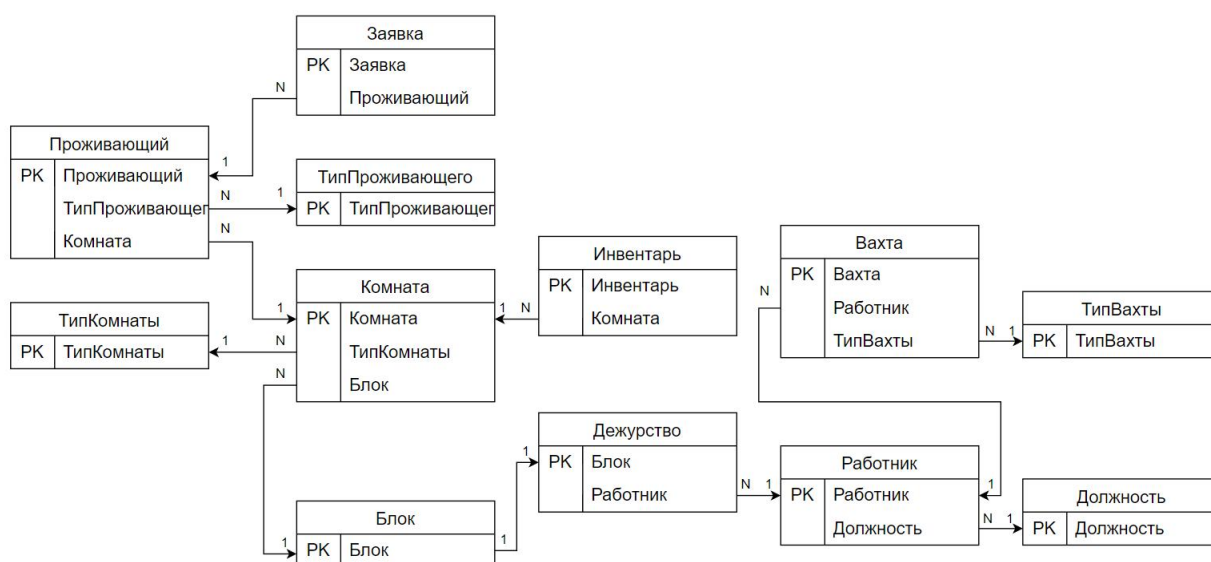


Рисунок 13 – Предварительная диаграмма отношений

## 2.2.2 Распределение атрибутов по отношениям

**Отношение Комната содержит атрибуты:**

– Номер комнаты



- Тип
- Количество мест
- Блок

**Отношение Проживающий содержит атрибуты:**

- Код проживающего (серия и номер паспорта)
- ФИО
- Телефон
- Пол
- Дата рождения
- Тип
- Номер комнаты

**Отношение Работник содержит атрибуты:**

- Кол работника (серия и номер паспорта)
- ФИО
- Телефон
- Должность

**Отношение Блок содержит атрибуты:**

- Код блока
- Этаж
- Крыло
- Тип

**Отношение Инвентарь содержит атрибуты:**

- Код инвентаря
- Название
- Дата поставки
- Стоимость
- Комната

**Отношение Заявки содержит атрибуты:**

- Номер

- Тема
- Текст
- Статус
- Дата составления
- Составитель

**Отношение Вахты содержит атрибуты:**

- Номер
- Тип
- Дата захода
- Длительность
- Вахтер

**Отношение Дежурство содержит атрибуты:**

- Блок
- Дежурный

**Отношение ТипПроживающего содержит атрибуты:**

- Название (место обучения + льготы)
- Тариф

**Отношение ТипКомнаты содержит атрибуты:**

- Название
- Количество мест

**Отношение Должность содержит атрибуты:**

- Название
- Оклад

**Отношение ТипВахты содержит атрибуты:**

- Название
- Ставка

### 2.2.3 Проверка отношений на БКНФ

Все отношения находятся:

- В 1 НФ, так как все атрибуты являются атомарными,

- Во 2НФ поскольку все неключевые элементы функционально полно зависят от первичного ключа,
- В 3НФ так как в нем нет транзитивных зависимостей,
- В БКНФ потому что детерминант функциональных зависимостей является единственным потенциальным ключом (первичным).

Схемы функциональных зависимостей показаны на рисунках 14-21

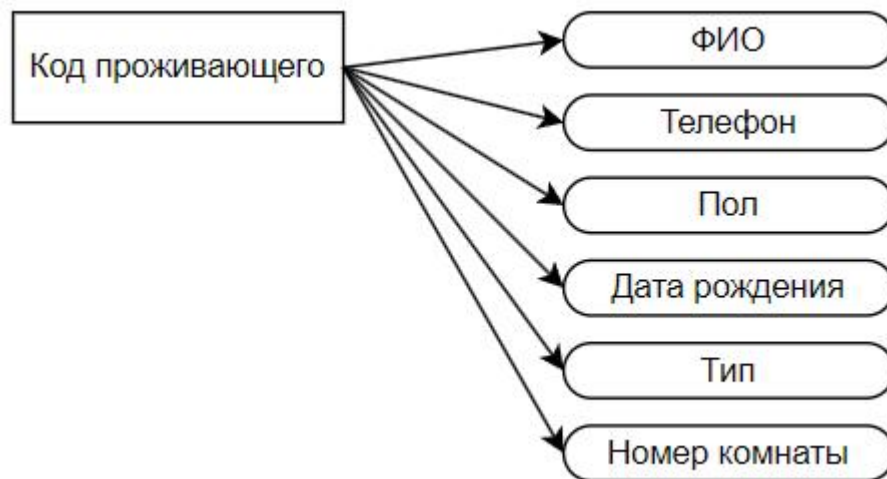


Рисунок 14 – зависимости отношения Проживающий



Рисунок 15 – зависимости отношения Комната

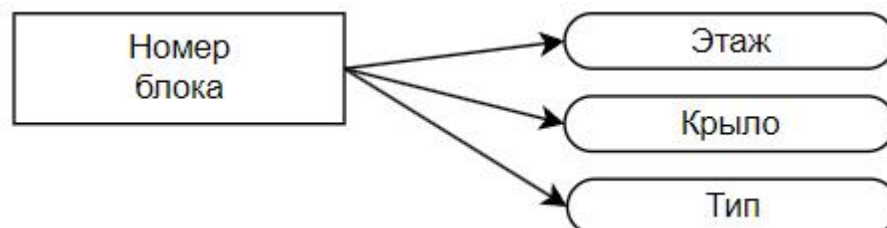


Рисунок 16 – зависимости отношения Блок

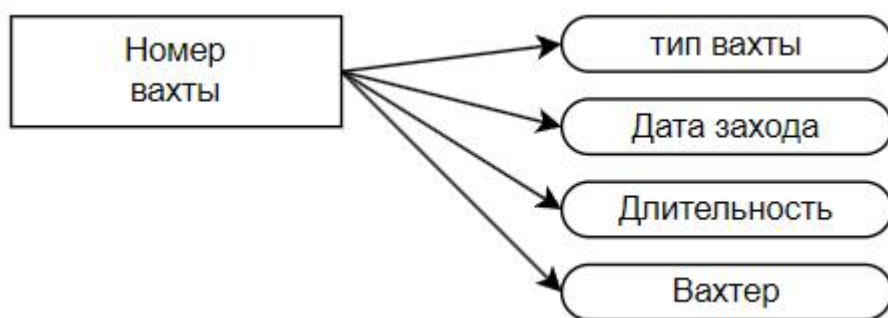


Рисунок 17 – зависимости отношения Вахта

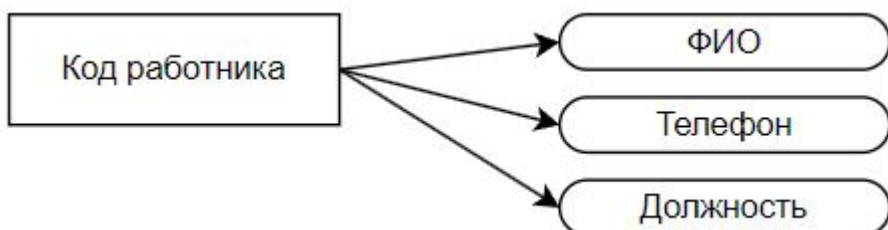


Рисунок 18 – зависимости отношения Работник

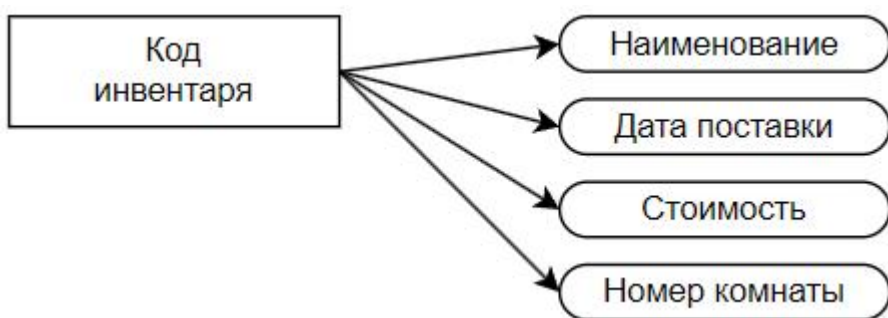


Рисунок 19 – зависимости отношения Инвентарь

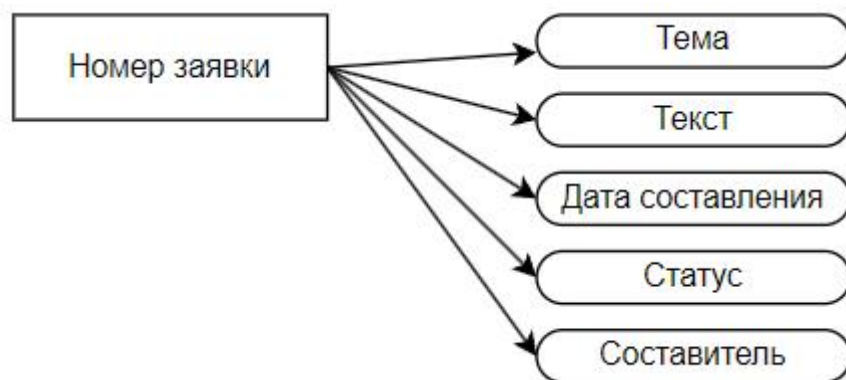


Рисунок 20 – зависимости отношения Заявка



## **2.3 Разработка объектов поддержания целостности данных**

Сценарии создания объектов БД представлены в ПРИЛОЖЕНИИ А.

### **2.3.1 Разработка правил, умолчаний и типов**

#### **Таблица «Проживающие»**

Атрибут «Код» (номер паспорта) является первичным ключем, не может принимать NULL значения, значения находятся в диапазоне 1000000000 – 9999999999.

Атрибут «ФИО» не может принимать NULL значения, значения должны соответствовать шаблону [А-Я]% [А-Я]% [А-Я]%.

Атрибут «Телефон» не может принимать NULL значения, значения должны соответствовать шаблону +7([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9] или 0([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9].

Атрибут «Пол» не может принимать NULL значения, значения могут быть только «муж» или «жен».

Атрибут «Дата рождения» не может принимать NULL значения.

Атрибут «Тип» не может принимать NULL значения.

Атрибут «Комната» не может принимать NULL значения.

#### **Таблица «Комнаты»**

Атрибут «Номер» является первичным ключем, не может принимать NULL значения, не может быть отрицательным.

Атрибут «Тип» не может принимать NULL значения.

Атрибут «Блок» не может принимать NULL значения.

#### **Таблица «Заявки»**

Атрибут «Номер» является первичным ключем, не может принимать NULL значения.

Атрибут «Тема» не может принимать NULL значения, значение по умолчанию «Без темы».

Атрибут Статус не может принимать NULL значения, список допустимых значений: «Открыта», «Выполняется», «Отклонена» или «Выполнена», значение по умолчанию «Открыта».

Атрибут «ДатаСоставления» не может принимать NULL значения, по умолчанию устанавливается текущая дата и время.

Атрибут Текст является необязательным.

Атрибут Составитель не может принимать NULL значения.

#### **Таблица «Вахты»**

Атрибут «Номер» является первичным ключем, не может принимать NULL значения.

Атрибут «Тип» не может принимать NULL значения.

Атрибут «ДатаНачала» не может принимать NULL значения, по умолчанию устанавливается текущая дата и время.

Атрибут «Длительность» не может принимать NULL значения и значения меньше нуля.

Атрибут «Вахтер» не может принимать NULL значения.

#### **Таблица «Инвентарь»**

Атрибут «Код» является первичным ключем, не может принимать NULL значения.

Атрибут «Наименование» не может принимать NULL значения.

Атрибут «ДатаПоставки» не может принимать NULL значения, по умолчанию устанавливается текущая дата и время.

Атрибут «Стоимость» не может принимать NULL значения и отрицательные значения.

Атрибут «Комната» не может принимать NULL значения.

#### **Таблица «Блоки»**

Атрибут «Номер» является первичным ключем, не может принимать NULL значения.

Атрибут «Этаж» не может принимать NULL значения.

Атрибут «Крыло» не может принимать NULL значения.

Атрибут «Тип» не может принимать NULL значения, список допустимых значений: «Мужской», «Женский», «Общий» «Преподавательский», «Семейный» или «Служебный» .

#### **Таблица «Работники»**

Атрибут «Код» (номер паспорта) является первичным ключом, не может принимать NULL значения, значения находятся в диапазоне 1000000000 – 9999999999.

Атрибут «ФИО» не может принимать NULL значения, значения должны соответствовать шаблону [А-Я]% [А-Я]% [А-Я]%.

Атрибут «Телефон» не может принимать NULL значения, значения должны соответствовать шаблону +7([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9] или 0([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9].

Атрибут «Должность» не может принимать NULL значения.

#### **Таблица «Дежурство»**

Атрибут «Блок» является первичным ключом, не может принимать NULL значения.

Атрибут «Дежурный» не должен принимать NULL значения.

#### **Таблица «Типы Проживающих»**

Атрибут «Название» является первичным ключом, не может принимать NULL значения.

Атрибут «Тариф» не должен принимать NULL значения, не может быть отрицательным.

#### **Таблица «Типы Комнат»**

Атрибут «Название» является первичным ключом, не может принимать NULL значения.

Атрибут «КойкоМеста» не может принимать NULL значения, не может принимать отрицательные значения.

#### **Таблица «Типы Вахт»**



Атрибут «Название» является первичным ключом, не может принимать NULL значения.

Атрибут «Ставка» не может принимать NULL значения, не может принимать отрицательные значения.

#### **Таблица «Должности»**

Атрибут «Название» является первичным ключом, не может принимать NULL значения.

Атрибут «Оклад» не может принимать NULL значения, не может принимать отрицательные значения.

#### 2.3.2 Разработка триггеров

##### **Для таблицы «Блоки»**

Проверяет на уникальность множество (Этаж, Крыло).

##### **Для таблицы «заявки»**

Если статус заявки «Выполнена» или «Отклонена», не дает изменять значения, так как эти заявки уже логически завершены.

##### **Для таблицы «Вахты»**

Дает назначить из работников только вахтера.

##### **Для таблицы «Дежурство**

Дает назначить только работника с должностью «Уборщик»

##### **Для таблицы «Комнаты»**

Триггер для каскадного удаления инвентаря и проживающих а также их заявок.

#### 2.3.3 Разработка процедур

Процедуры добавления/обновления/удаления вахт;

Процедуры добавления/обновления/удаления инвентаря;

Процедуры добавления/изменения статуса инвентаря.

## 3 Разработка клиентской части ИС

### 3.1 Разработка прототипа интерфейса пользователя

Клиентская часть разрабатывается для администратора общежития и содержит пользовательские интерфейсы для операций над вахтами, инвентарем и заявками. После запуска приложения в него необходимо войти с помощью пароля администратора (Рисунок 23)(пароль «linkedList»). После успешного входа пользователю будут предоставлены 3 основных вкладки для управления данными (Рисунки 24 -26).

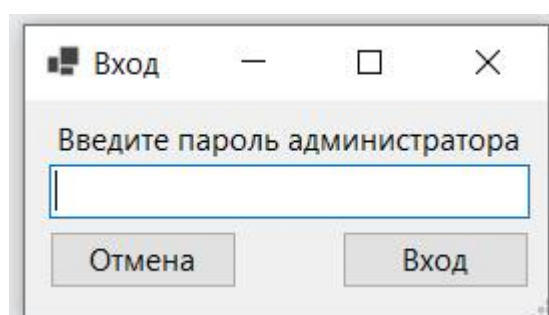


Рисунок 23 – Диалог входа

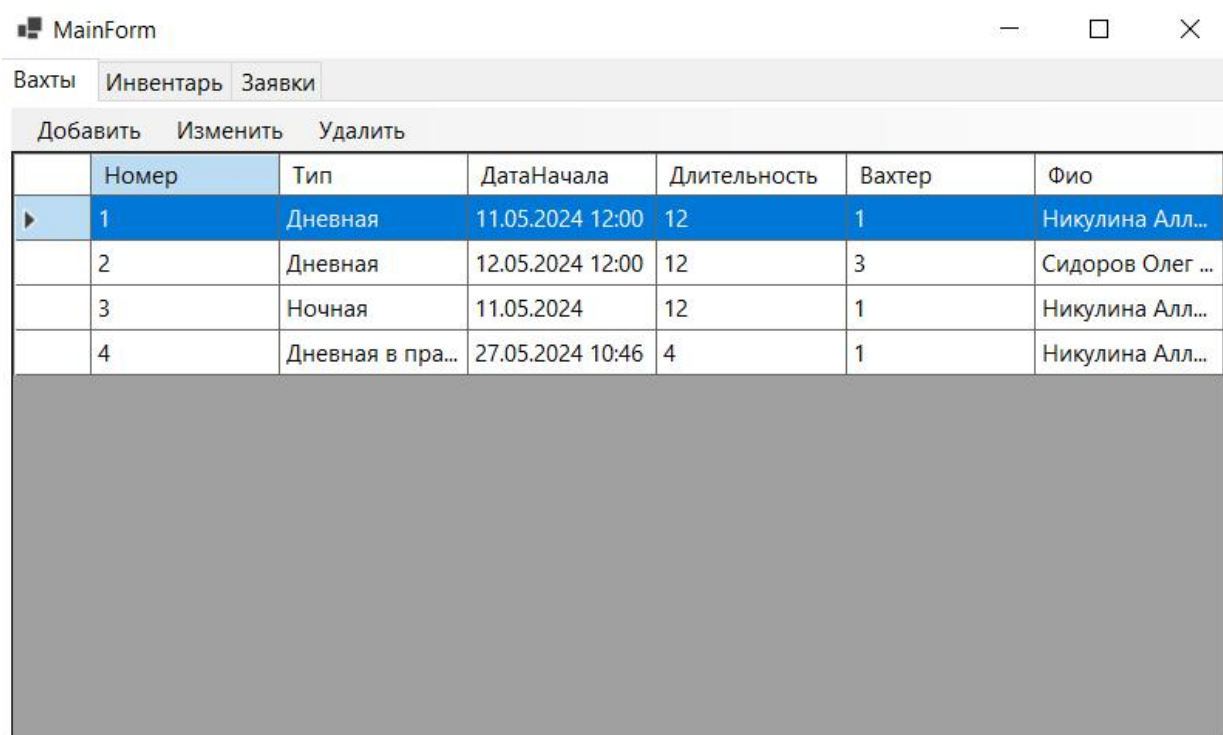


Рисунок 24 – Раздел «Вахты»

MainForm

Вахты Инвентарь Заявки

Добавить Обновить Удалить Вычислить общую стоимость

	Код	Название	ДатаПоставки	Стоимость	Комната	Блок	Всего
▶	2	Стул "Комф...	23.04.2022	900	476	47	4
	3	Стол "Учени...	23.04.2022	2200	476	47	2
	4	Шкаф 2200х...	23.04.2022	12000	476	47	2
	5	Стул "Комф...	23.04.2022	900	477	47	4
	6	Стул "Комф...	23.04.2022	900	477	47	4
	7	Стол "Учени...	23.04.2022	2200	477	47	2
	8	Шкаф 2200х...	23.04.2022	12000	477	47	2
	9	Холодильн...	01.09.1970	120	476	47	1
	10	Стул "Комф...	27.05.2024	950	421	42	4
	11	Пистолет "Ч...	22.02.2022	65000	476	47	1

Рисунок 25 – Раздел «Инвентарь»

MainForm

Вахты Инвентарь Заявки

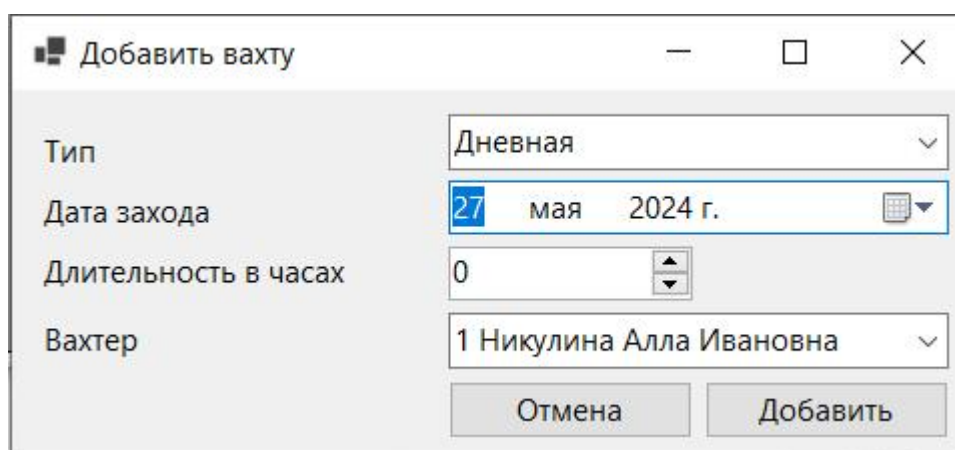
Добавить Обновить статус

	Номер	Тема	Статус	ДатаСоздания	Текст	Составитель	Код
▶	1	Без темы	Выполнена	27.05.2024	Текст	Сивер Илья...	2
	3	Здоровье	Отклонена	01.01.1900	Требую 8ми...	Сивер Илья...	2
	4	Ремонт	Отклонена	27.05.2024	Сломалась ...	Сивер Илья...	2
	5	Финансы	Открыта	27.05.2024	Повысте сти...	Сивер Илья...	2

Рисунок 26 – Раздел «Заявки»

### 3.2 Реализация интерфейса пользователя

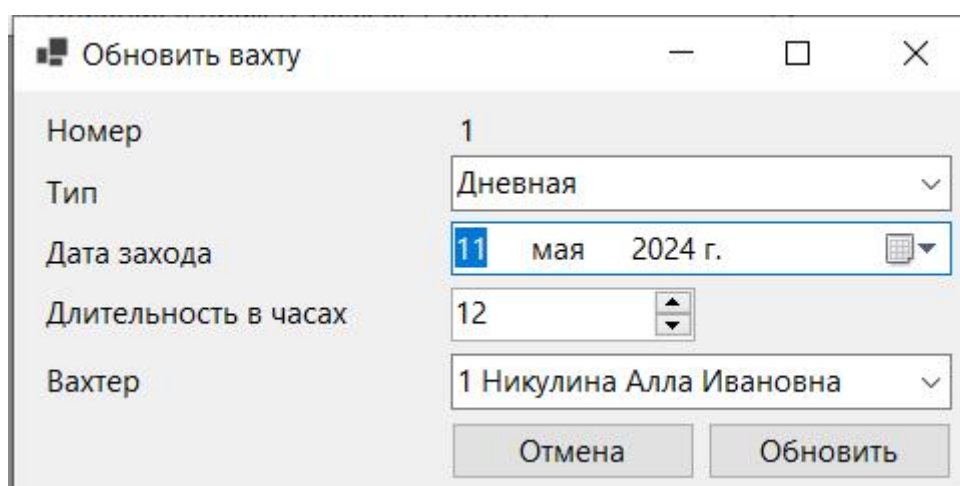
Раздел «Вахты» позволяет вызвать 3 диалоговых окна для операций добавления, изменения и удаления вахт (Рисунки 27-29).



The screenshot shows a dialog box titled "Добавить вахту" (Add Shift). It contains the following fields and controls:

- Тип** (Type): A dropdown menu with "Дневная" (Day) selected.
- Дата захода** (Start Date): A date picker showing "27 мая 2024 г." (May 27, 2024).
- Длительность в часах** (Duration in hours): A spinner box with the value "0".
- Вахтер** (Shift Worker): A dropdown menu with "1 Никулина Алла Ивановна" (1 Nikulina Alla Ivanovna) selected.
- Buttons: "Отмена" (Cancel) and "Добавить" (Add).

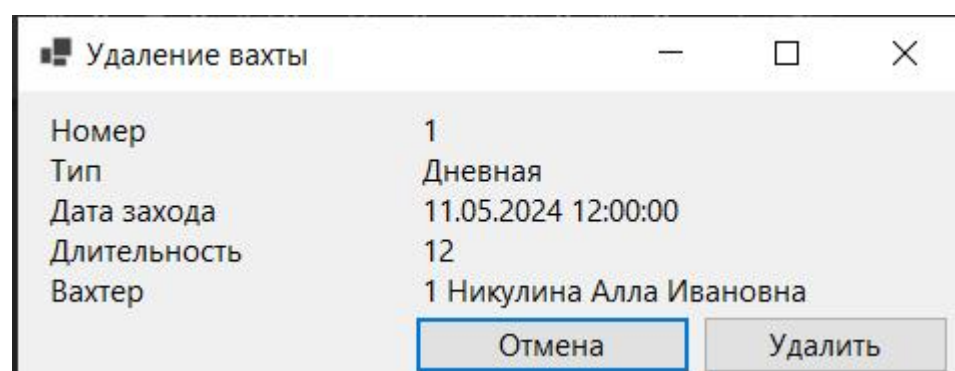
Рисунок 27 – Диалог добавления вахты



The screenshot shows a dialog box titled "Обновить вахту" (Update Shift). It contains the following fields and controls:

- Номер** (Number): A text field with the value "1".
- Тип** (Type): A dropdown menu with "Дневная" (Day) selected.
- Дата захода** (Start Date): A date picker showing "11 мая 2024 г." (May 11, 2024).
- Длительность в часах** (Duration in hours): A spinner box with the value "12".
- Вахтер** (Shift Worker): A dropdown menu with "1 Никулина Алла Ивановна" (1 Nikulina Alla Ivanovna) selected.
- Buttons: "Отмена" (Cancel) and "Обновить" (Update).

Рисунок 28 – Диалог обновления вахты



The screenshot shows a dialog box titled "Удаление вахты" (Delete Shift). It contains the following fields and controls:

- Номер** (Number): A text field with the value "1".
- Тип** (Type): A text field with the value "Дневная" (Day).
- Дата захода** (Start Date): A text field with the value "11.05.2024 12:00:00".
- Длительность** (Duration): A text field with the value "12".
- Вахтер** (Shift Worker): A text field with the value "1 Никулина Алла Ивановна" (1 Nikulina Alla Ivanovna).
- Buttons: "Отмена" (Cancel) and "Удалить" (Delete).

Рисунок 29 – Диалог удаления вахты

Раздел «Инвентарь» также позволяет вызвать 3 диалоговых окна для операций добавления, изменения и удаления записей об инвентаре, а также

имеется возможность вывести общую стоимость всего инвентаря(Рисунки 30-33).

Добавить инвентарь

Наименование	<input type="text"/>
Дата поставки	27 мая 2024 г.
Закупочная стоимость	0
Номер комнаты	421

Отмена Добавить

Рисунок 30 – Диалог добавления инвентаря

Обновить инвентарь

Номер	2
Наименование	Стул "Комфорт"
Дата поставки	23 апреля 2022 г.
Закупочная стоимость	900
Номер комнаты	476

Отмена Обновить

Рисунок 31 – Диалог обновления инвентаря

inventRemove

Номер	2
Наименование	Стул "Комфорт"
Дата поставки	23.04.2022 0:00:00
Закупочная стоимость	900
Комната	476

Отмена Удалить

Рисунок 32 – Диалог удаления инвентаря

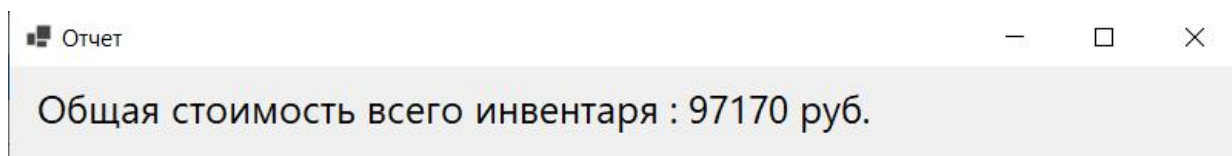


Рисунок 33 – Диалог отчета

Раздел «Заявки» позволяет вызвать 2 диалоговых окна для операций добавления новой заявки и изменения статуса заявки (Рисунки 34-35)

A screenshot of a dialog box titled "Добавить заявку" (Add request). The dialog has a light gray header bar with the title and window controls. The main area contains a text input field labeled "Тема" (Topic), a large text area labeled "Основной текст" (Main text), and a dropdown menu labeled "Составитель" (Author) with the value "2" selected. At the bottom right, there are two buttons: "Отмена" (Cancel) and "Добавить" (Add).

Рисунок 34 – Диалог добавления заявки

A screenshot of a dialog box titled "Изменить статус" (Change status). The dialog has a light gray header bar with the title and window controls. The main area displays the request details: "3" (ID), "Здоровье" (Health), "Статус:" (Status) with a dropdown menu showing "Отклонена" (Rejected), "Дата составления: 01.01.1900 0:00:00" (Creation date), and "Текст" (Text) with a text area containing "Требую 8ми часовой сом" (I need 8 hours of sleep). At the bottom, there is a label "Составитель: Сивер Илья Иванович лена" (Author: Siver Ilya Ivanovich lena) and a "Подтвердить" (Confirm) button.

Рисунок 35 – Диалог обновления статуса заявки

## **4 Тестирование основных функций приложения**

### **В ПРОЦЕССЕ**

## **ЗАКЛЮЧЕНИЕ**

### **В ПРОЦЕССЕ**



**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**  
В ПРОЦЕССЕ

## ПРИЛОЖЕНИЕ А: сценарий создания объектов БД

```
--Пользовательский тип для типа блока--
CREATE TYPE blockType FROM VARCHAR(127)
GO
CREATE RULE blockTypeRule AS
@type = 'Мужской' OR
@type = 'Женский' OR
@type = 'Общий' OR
@type = 'Преподавательский' OR
@type = 'Семейный' OR
@type = 'Служебный'
GO
EXEC sp_bindRule blockTypeRule, blockType
GO

--Пользовательский тип для статуса заявок--
CREATE TYPE statusType FROM VARCHAR(127)
GO
CREATE RULE statusRule AS
@status = 'Открыта' OR
@status = 'Выполняется' OR
@status = 'Отклонена' OR
@status = 'Выполнена'
GO
EXEC sp_bindRule statusRule, statusType
GO
CREATE DEFAULT issueStatusDefault AS
'Открыта'
GO
EXEC sp_bindefault issueStatusDefault, statusType
GO

use master
go
use CourseWorkSQL
go
BEGIN TRANSACTION addTables
    CREATE TABLE ТипыПроживающих (
        Название varchar(127) PRIMARY KEY NOT NULL,
        Тариф int NOT NULL
    )
GO
    CREATE TABLE ТипыКомнат (
        Название varchar(127) PRIMARY KEY NOT NULL,
        КойкоМеста int NOT NULL
    )
GO
    CREATE TABLE Должности (
        Название varchar(127) PRIMARY KEY NOT NULL,
        Оклад int NOT NULL
    )
```

```

)
GO
CREATE TABLE ТипыВахт (
    Название varchar(127) PRIMARY KEY NOT NULL,
    Ставка int NOT NULL
)
GO
CREATE TABLE Работники (
    Код bigint PRIMARY KEY NOT NULL,
    ФИО varchar(127) NOT NULL,
    Телефон varchar(127) NOT NULL,
    Должность varchar(127) NOT NULL,
    FOREIGN KEY (Должность) REFERENCES Должности
)
GO
CREATE TABLE Блоки (
    Номер int PRIMARY KEY NOT NULL,
    Этаж int NOT NULL,
    Крыло int NOT NULL,
    Тип blockType NOT NULL
)
GO
CREATE TABLE Комнаты (
    Номер int PRIMARY KEY NOT NULL,
    Тип varchar(127) NOT NULL,
    Блок int NOT NULL,
    FOREIGN KEY (Тип) REFERENCES ТипыКомнат,
    FOREIGN KEY (Блок) REFERENCES Блоки
)
GO
CREATE TABLE Проживающие (
    Код bigint PRIMARY KEY NOT NULL,
    ФИО varchar(127) NOT NULL,
    Телефон varchar(127) NOT NULL,
    Пол varchar(3) NOT NULL,
    ДатаРождения date NOT NULL,
    Тип varchar(127) NOT NULL,
    Комната int NOT NULL,
    FOREIGN KEY (Тип) REFERENCES ТипыПроживающих,
    FOREIGN KEY (Комната) REFERENCES Комнаты
)
GO
CREATE TABLE Заявки (
    Номер int PRIMARY KEY NOT NULL IDENTITY(1,1),
    Тема varchar(127) NOT NULL,
    Статус statusType NOT NULL,
    ДатаСоздания date NOT NULL default GETDATE(),
    Текст varchar(MAX),
    Составитель bigint NOT NULL,
    FOREIGN KEY (Составитель) REFERENCES Проживающие,
)

```

```

GO
CREATE TABLE Инвентарь (
    Код int PRIMARY KEY NOT NULL IDENTITY(1,1),
    Название varchar(127) NOT NULL,
    ДатаПоставки date NOT NULL default GETDATE(),
    Стоимость int NOT NULL,
    Комната int NOT NULL,
    FOREIGN KEY (Комната) REFERENCES Комнаты,
)
GO
CREATE TABLE Вахты (
    Номер int PRIMARY KEY NOT NULL IDENTITY(1,1),
    Тип varchar(127) NOT NULL,
    FOREIGN KEY (Тип) REFERENCES ТипыВахт,
    ДатаНачала date NOT NULL default GETDATE(),
    Длительность int NOT NULL,
    Вахтер bigint,
    FOREIGN KEY (Вахтер) REFERENCES Работники
)
GO
CREATE TABLE Дежурство(
    Блок int PRIMARY KEY NOT NULL,
    Дежурный bigint NOT NULL,
    FOREIGN KEY(Блок) REFERENCES Блоки,
    FOREIGN KEY(Дежурный) REFERENCES Работники
)
GO
COMMIT TRANSACTION addTables

```

```

--Представление для вывода заявок в приложении--
CREATE VIEW issuesView AS
SELECT З.Номер,З.Тема,З.Статус,З.ДатаСоздания,З.Текст,п.ФИО as
Составитель, З.Составитель AS Код
FROM Заявки З JOIN Проживающие П ON З.Составитель = П.Код
GO

```

```

--Представление для вывода инвентаря в приложении--
CREATE VIEW inventView AS
SELECT И.*, К.Блок as Блок, Всего =
    (SELECT count(*)
     FROM Инвентарь В
     WHERE В.Название = И.Название)
FROM Инвентарь И JOIN Комнаты К ON Комната = К.Номер
GO

```

```

--Представление для вывода вахт в приложении--
CREATE VIEW vahtsView AS
SELECT В.*, Р.ФИО AS Фио
FROM Вахты В JOIN Работники Р ON В.Вахтер = Р.Код
GO

```

```

--Правило для номера телефона--

```

```

CREATE RULE telRule AS
(@tel LIKE '+7([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]') OR
(@tel LIKE '8([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]')
GO
EXEC sp_bindrule telRule, 'Проживающие.Телефон'
GO
EXEC sp_bindrule telRule, 'Работники.Телефон'
GO

--Правило для ФИО--
CREATE RULE fioRule AS
@fio LIKE '[А-Я]% [А-Я]% [А-Я]%'
GO
EXEC sp_bindrule fioRule, 'Проживающие.ФИО'
GO
EXEC sp_bindrule fioRule, 'Работники.ФИО'
GO

--Правило для значений которые должны быть больше нуля--
CREATE RULE notNegativeRule AS
@value >= 0
GO
EXEC sp_bindRule notNegativeRule, 'Должности.Оклад'
GO
EXEC sp_bindRule notNegativeRule, 'ТипыВахт.Ставка'
GO
EXEC sp_bindRule notNegativeRule, 'Вахты.Длительность'
GO
EXEC sp_bindRule notNegativeRule, 'ТипыПроживающих.Тариф'
GO
EXEC sp_bindRule notNegativeRule, 'ТипыКомнат.КойкоМеста'
GO
EXEC sp_bindRule notNegativeRule, 'Инвентарь.Стоимость'
GO

--Правило для номера паспорта--
CREATE RULE passportRule AS
@value >= 1000000000 AND @value <= 9999999999
GO
EXEC sp_bindRule passportRule, 'Работники.Код'
GO
EXEC sp_bindRule passportRule, 'Проживающие.Код'
GO

--Триггер каскадного удаления проживающих в удаляемой комнате,
их заявок и инвентаря--
CREATE TRIGGER cascadeDeleteRoom ON Комнаты
INSTEAD OF DELETE AS
BEGIN

```

```

DECLARE @room int

DECLARE roomCur CURSOR FOR
    SELECT deleted.Номер
    FROM deleted

OPEN roomCur
    FETCH NEXT FROM roomCur INTO @room
    WHILE @@FETCH_STATUS = 0
    BEGIN

        DECLARE @student int

        DECLARE studentCur CURSOR FOR
            SELECT Проживающие.Код
            FROM Проживающие
            WHERE Проживающие.Комната = @room

        OPEN studentCur
            FETCH NEXT FROM studentCur INTO @student
            WHILE @@FETCH_STATUS = 0
            BEGIN
                DELETE FROM Заявки
                WHERE Заявки.Составитель = @student
                FETCH NEXT FROM studentCur INTO
@student
            END
        CLOSE studentCur
        DEALLOCATE studentCur

        DELETE FROM Проживающие
        WHERE Проживающие.Комната = @room

        DELETE FROM Инвентарь
        WHERE Инвентарь.Комната = @room

        DELETE FROM Комнаты
        WHERE Комнаты.Номер = @room

        FETCH NEXT FROM roomCur INTO @room
    END
    CLOSE roomCur
    DEALLOCATE roomCur
END
GO

--Триггер для проверки на уникальность значений этажа и крыла
добавляемых блоков--
CREATE TRIGGER insertBlock ON Блоки
INSTEAD OF INSERT AS
    BEGIN

```

```

        DECLARE @block int
        DECLARE @level int
        DECLARE @wing int
        DECLARE @type varchar(127)
        DECLARE blockCur CURSOR FOR
            SELECT inserted.Номер, inserted.Этаж, inserted.Крыло,
inserted.Тип
            FROM inserted
        OPEN blockCur
        FETCH NEXT FROM blockCur INTO @block, @level, @wing,
@type
        WHILE @@FETCH_STATUS = 0
        BEGIN
            IF NOT EXISTS(SELECT * FROM Блоки WHERE
(Блоки.Номер = @block) OR (Блоки.Этаж = @level and Блоки.Крыло =
@wing))
                BEGIN
                    INSERT INTO Блоки VALUES
(@block,@level,@wing,@type)
                END
            FETCH NEXT FROM blockCur INTO @block, @level,
@wing, @type
        END
        CLOSE blockCur
        DEALLOCATE blockCur
    END
GO

```

--Триггер на изменение статуса заявки--

```

CREATE TRIGGER updateIssue ON Заявки
INSTEAD OF UPDATE AS
BEGIN
    DECLARE @issue int
    DECLARE @status varchar(127)

    DECLARE issueCur CURSOR FOR
        SELECT inserted.Номер, inserted.Статус
        FROM inserted

    OPEN issueCur
    FETCH NEXT FROM issueCur INTO @issue, @status
    WHILE @@FETCH_STATUS = 0
    BEGIN
        IF EXISTS(SELECT * FROM Заявки WHERE Заявки.Номер
= @issue and (Заявки.Статус <> 'Отклонена' OR Заявки.Статус <>
'Выполнена'))
            BEGIN
                UPDATE Заявки
                SET Заявки.Статус = @status
                WHERE Заявки.Номер = @issue
            END
        FETCH NEXT FROM issueCur INTO @issue, @status
    END

```

```

        FETCH NEXT FROM issueCur INTO @issue, @status
    END
    CLOSE issueCur
    DEALLOCATE issueCur
END
GO

--Триггер на добавление вахты, в которой может дежурить только
работник с должностью вахтера--
ALTER TRIGGER addVaht On Вахты
INSTEAD OF INSERT AS
BEGIN
    DECLARE @vaht int
    DECLARE @type varchar(127)
    DECLARE @date datetime
    DECLARE @duration int
    DECLARE @worker int

    DECLARE vahtCur CURSOR FOR
        SELECT *
        FROM inserted

    OPEN vahtCur
        FETCH NEXT FROM vahtCur INTO @vaht,@type,
@date,@duration, @worker
        WHILE @@FETCH_STATUS = 0
        BEGIN
            IF EXISTS(SELECT * FROM Работники WHERE
Работники.Код = @worker AND Работники.Должность = 'Вахтер')
            BEGIN
                INSERT INTO Вахты VALUES
                    (@type, @date,@duration, @worker)
            END
            FETCH NEXT FROM vahtCur INTO @vaht,@type,
@date,@duration, @worker
        END
        CLOSE vahtCur
        DEALLOCATE vahtCur
END
Go

--Триггер на добавление дежурного в блок--
CREATE TRIGGER addDuty ON Дежурство
INSTEAD OF INSERT AS
BEGIN
    DECLARE @worker int
    DECLARE @block int
    DECLARE dutyCur CURSOR FOR
        SELECT inserted.Блок, inserted.Дежурный
        FROM inserted

```



```

OPEN vahtCur
  FETCH NEXT FROM vahtCur INTO @block, @worker
  WHILE @@FETCH_STATUS = 0
  BEGIN
    IF EXISTS(SELECT * FROM Работники WHERE
Работники.Код = @worker AND Работники.Должность = 'Уборщик')
    BEGIN
      INSERT INTO Дежурство VALUES
        (@block, @worker)
    END
    FETCH NEXT FROM vahtCur INTO @block, @worker
  END
CLOSE vahtCur
DEALLOCATE vahtCur
END
GO

```

```

--Процедура добавления новой вахты--
CREATE PROC insertVaht (@type varchar(127), @date datetime,
@duration int, @worker int)
AS
IF EXISTS(SELECT * FROM Работники WHERE Работники.Код = @worker
AND Работники.Должность = 'Вахтер')
BEGIN
  INSERT INTO Вахты (Тип, ДатаНачала, Длительность, Вахтер)
VALUES
  (@type, @date, @duration, @worker)
  RETURN 0
END
ELSE
BEGIN
  RETURN 1
END
GO

```

```

--Процедура обновления вахты--
CREATE PROC updateVaht (@num int, @type varchar(127), @date
datetime, @duration int, @worker int)
AS
IF EXISTS(SELECT * FROM Работники WHERE Работники.Код = @worker
and Должность = 'Вахтер')
BEGIN
  UPDATE Вахты
  SET Тип = @type, ДатаНачала = @date, Длительность =
@duration, Вахтер = @worker
  WHERE Вахты.Номер = @num
  RETURN 0
END
ELSE
BEGIN
  RETURN 1

```

```

END
GO

--Процедура Добавления заявки--
ALTER PROC insertIssue (@title varchar(127), @text varchar(max),
@author int)
AS
IF EXISTS(SELECT * FROM Проживающие WHERE Проживающие.Код =
@author)
BEGIN
    INSERT INTO Заявки (Заявки.Тема, Заявки.Текст,
Заявки.Составитель) VALUES
    (@title, @text, @author )
    RETURN 0
END
ELSE
BEGIN
    RETURN 1
END
GO

--Процедура обновления статуса заявки--
CREATE PROC updateIssue (@num int, @status statusType)
AS
IF EXISTS(SELECT * FROM Заявки WHERE Заявки.Номер = @num and
(Заявки.Статус = 'Открыта' OR Заявки.Статус = 'Выполняется'))
BEGIN
    UPDATE Заявки
    SET Заявки.Статус = @status
    WHERE Заявки.Номер = @num
    RETURN 0
END
ELSE
BEGIN
    RETURN 1
END
Go

--Процедура добавления нового инвентаря--
CREATE PROC insertInvent (@name varchar(127), @date datetime,
@cost int, @room int)
AS
IF EXISTS(SELECT * FROM Комнаты WHERE Комнаты.Номер = @room)
BEGIN
    INSERT INTO Инвентарь VALUES
    (@name , @date , @cost , @room )
    RETURN 0
END
ELSE
BEGIN
    RETURN 1

```

```

END
GO

--Процедура удаления инвентаря--
CREATE PROC deleteInvent (@code int)
AS
IF EXISTS(SELECT * FROM Инвентарь WHERE Инвентарь.Код = @code)
BEGIN
    DELETE Инвентарь
    WHERE Инвентарь.Код = @code
    RETURN 0
END
ELSE
BEGIN
    RETURN 1
END
GO

--Процедура обновления инвентаря--
CREATE PROC updateInvent (@code int, @name varchar(127), @date
datetime, @cost int, @room int)
AS
IF EXISTS(SELECT * FROM Инвентарь WHERE Инвентарь.Код = @code)
BEGIN
    UPDATE Инвентарь
    SET Название = @name, ДатаПоставки = @date, Стоимость =
@cost, Комната = @room
    WHERE Инвентарь.Код = @code
    RETURN 0
END
ELSE
BEGIN
    RETURN 1
END
GO

--Процедура удаления вахты--
CREATE PROC deleteVaht (@num int)
AS
IF EXISTS(SELECT * FROM Вахты WHERE Вахты.Номер = @num)
BEGIN
    DELETE Вахты
    WHERE Вахты.Номер = @num
    RETURN 0
END
ELSE
BEGIN
    RETURN 1
END
GO

```

```
CREATE PROC computeAllCost (@cost int output)
AS
SELECT @cost = sum(Стоимость)
FROM Инвентарь
GO
```

## ПРИЛОЖЕНИЕ Б: сценарий заполнения таблиц БД

```
use master
GO
use CourseWorkSQL
GO
BEGIN TRAN addRows
INSERT INTO ТипыКомнат VALUES
('Двухместная', 2),
('Трехместная', 3),
('Служебная', 0)
Go
INSERT INTO Блоки VALUES
(47, 4, 7, 'Мужской'),
(42, 4, 2, 'Женский')
Go
INSERT INTO Комнаты VALUES
(475, 'Трехместная', 47),
(476, 'Двухместная', 47),
(477, 'Двухместная', 47),
(478, 'Трехместная', 47),
(421, 'Трехместная', 42),
(422, 'Двухместная', 42),
(423, 'Двухместная', 42),
(424, 'Трехместная', 42)
Go
INSERT INTO Инвентарь VALUES
('Стул "Комфорт"', '23.04.2022', 900, 476),
('Стул "Комфорт"', '23.04.2022', 900, 476),
('Стол "Ученический"', '23.04.2022', 2200, 476),
('Шкаф 2200x2000x400', '23.04.2022', 12000, 476),
('Стул "Комфорт"', '23.04.2022', 900, 477),
('Стул "Комфорт"', '23.04.2022', 900, 477),
('Стол "Ученический"', '23.04.2022', 2200, 477),
('Шкаф 2200x2000x400', '23.04.2022', 12000, 477),
('Холодильник "Полюс"', '01.09.1970', 120, 476)
Go
INSERT INTO ТипыПроживающих VALUES
('РГРТУ Очная', 1000),
('РГРТУ Заочная', 1000),
('РГРТУ Очная Льготная', 0)
Go
INSERT INTO Проживающие VALUES
(2342345676, 'Сивер Илья Иванович', '8(923)345-54-23', 'муж', '25.11.2004', 'РГРТУ Очная', 475)
Go
INSERT INTO Заявки VALUES
('Ремонт', 'Открыта', '29.05.2024', 'Сломалась ручка двери в 476 комнате', 2342345676),
('Предложение', 'Открыта', '22.05.2024', 'Как насчет того чтобы ставить в комнаты сабы на 150 ватт?', 2342345676)
```

Go

INSERT INTO Должности VALUES

('Вахтер', 19000),  
('Уборщик', 17900),  
('Слесарь', 32000),  
('Кладовщик', 26000)

GO

INSERT INTO Работники VALUES

(2334555424, 'Чин Чен Чан', '8(333)222-11-00', 'Вахтер'),  
(1455523423, 'Никулина Алла Ивановна', '8(955)452-53-22', 'Вахтер'),  
(2134134123, 'Борисов Иван Ильич', '8(921)432-77-53', 'Слесарь'),  
(3989898784, 'Сидоров Олег Абдулович', '8(677)666-99-69', 'Уборщик')

GO

INSERT INTO ТипыВахт VALUES

('Ночная', 2),  
('Дневная', 1),  
('Ночная в праздник', 3),  
('Дневная в праздник', 2)

Go

INSERT INTO Вахты VALUES

('Дневная', '11.05.2024 12:00', 12, 2334555424),  
('Дневная', '12.05.2024 12:00', 12, 2334555424),  
('Ночная', '11.05.2024 00:00', 12, 1455523423)

Go

COMMIT TRAN addRows

**ПРИЛОЖЕНИЕ В:исходный код клиентского приложения**

<https://github.com/KyKyPy3HuK/ClientServerEF/tree/main>

также здесь есть все скрипты создания объектов БД, заполнения бд и отчет