

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)**

Кафедра комплексной информационной безопасности электронно-  
вычислительных систем (КИБЭВС)

### **РАССЕЛЕНИЕ СТУДЕНТОВ В ОБЩЕЖИТИИ**

Пояснительная записка к курсовой работе  
по дисциплине «Безопасность систем баз данных»

Студент гр.718-1  
\_\_\_\_\_ А.С. Зазуля  
«\_\_» \_\_\_\_ 2021

Руководитель  
Доцент кафедры КИБЭВС  
\_\_\_\_\_ К.С. Сарин  
«\_\_» \_\_\_\_ 2021

Томск 2021

## РЕФЕРАТ

Пояснительная записка содержит 81 страниц, 85 иллюстраций, 9 источников, 8 таблиц, 7 приложений.

Ключевые слова: АВТОРИЗАЦИЯ, БД, ТАБЛИЦА, БАЗА ДАННЫХ, АДМИНИСТРАТОР, РАСЕЛИТЕЛЬ, КОМЕНДАНТ, ЗАЯВЛЕНИЯ, СТУДЕНТЫ, КОМНАТА, ИНФОРМАЦИЯ, РАСПОРЯЖЕНИЯ, ПРОГРАММА.

### СУБД «РАССЕЛЕНИЕ СТУДЕНТОВ В ОБЩЕЖИТИИ».

Цель работы – разработка СУБД «РАССЕЛЕНИЕ СТУДЕНТОВ В ОБЩЕЖИТИИ» которая предназначена для автоматизированной работы студенческих общежитий, а также для хранения данных студентов в БД.

Преимуществами данной программы является:

- простой и удобный интерфейс;
- автоматизированные процессы;
- конфиденциальность данных

В результате работы была создана программа, которая корректно выполняет свои функции.

Разработка программы проводилась на основании технического задания.

Разработка программы проводилась на языке программирования C#. Разработка СУБД проводилась в реляционной СУБД «Microsoft SMMS». Пояснительная записка выполнена в текстовом редакторе Microsoft Word 2016.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра комплексной информационной безопасности электронно-  
вычислительных систем (КИБЭВС)

**УТВЕРЖДАЮ**

Заведующий кафедрой  
КИБЭВС, д-р техн. наук,  
профессор

\_\_\_\_\_ А.А. Шелупанов

“ \_\_\_\_ ” 2021 г.

**ЗАДАНИЕ**

на курсовую работу по дисциплине «Безопасность систем баз данных»  
студенту Зазуля Анастасии Сергеевне группы 718-1 факультета безопасности.

1 Тема работы: Создание автоматизированной информационной  
системы с учебно-исследовательской базой данных: Расселение студентов в  
общежитии

2 Исходные данные к работе:

2.1 Реляционная СУБД MS SQL Server

2.2 Данные по предметной области организация базы данных  
«Расселение студентов в общежитии», для хранения данных о студентах,  
заявлений, распоряжений, комнатах, общежитиях.

3 Срок сдачи студентом законченной работы: до 6 июня 2021 г.

4 Содержание курсовой работы:

#### 4.1 Проектирование инфологической модели данных:

- описание и структуризация предметной области (описание бизнес-процессов, диаграммы IDEF0);
- представление модели «Сущность-связь» (ER-модель);
- сценарий пользовательского интерфейса.

#### 4.2 Проектирование даталогической (логической) модели данных:

- проектирование реляционной базы данных на основе принципов нормализации;
- проектирование концептуальной модели данных (использование методологии IDEF1X);
- составление глоссария модели.

#### 4.3 Физическое проектирование БД:

- создание базы данных и ее необходимых элементов;
- описание ограничений на базу данных;
- сопоставление логических и физических имен.

#### 4.4 Написание программы обработки и работы с данными:

- генерация программы меню, реализующей пользовательский интерфейс;
- режим просмотра данных с использованием экранных форм;
- использование режимов редактирования данных;
- процедуры поиска и манипулирования данными (сортировки, фильтры и пр.);
- использование SQL операторов (SQL запросы, операторы определения данных, операторы манипулирования данными);
- обеспечение безопасности данных.

#### 5 Содержание пояснительной записи:

- титульный лист;
- реферат на русском языке;
- задание;

- содержание;
- введение;
- вопросы проектирования БД;
- обоснование выбора программных средств;
- руководство пользователю;
- описание прикладной программы;
- заключение;
- список использованных источников;
- приложения (структуры БД, экранные формы, отчетная форма, листинг программы).

Пояснительная записка должна быть оформлена в соответствии со стандартом ТУССР.

В конверте на обложке приложить диск с БД, исходными текстами программы с соответствующими файлами, исполнительными файлами, пояснительной запиской.

6 Дата выдачи задания:

«17» февраля 2021 г.

Задание согласовано:

Руководитель работы

Сарин К.С., доцент кафедры КИБЭВС

“ \_\_\_\_ ” 2021 г.

\_\_\_\_\_

Задание принято к исполнению

Зазуля А.С., студент группы 718-1

“ \_\_\_\_ ” 2021 г.

\_\_\_\_\_

## Содержание

1 Концептуальное (инфологическое) проектирование предметной области (ПО).....	8
1.1 Неформальное описание ПО с использованием естественного языка. 8	
1.2 Описание бизнес-процессов ПО в методологии функционального моделирования IDEF0.....	10
1.2.1 Описание ПО в методологии функционального моделирования IDEF0 до внедрения автоматизированной системы .....	10
1.2.2 Описание ПО в методологии функционального моделирования IDEF0 после внедрения автоматизированной системы .....	11
1.3 Цели автоматизации ПО.....	12
1.4 Концептуальная информационная модель данных для ПО. ....	13
1.4.1 Основные объекты ПО, информация о которых будет накапливаться в БД. Их характеристики и свойства (атрибуты). ....	13
1.4.2 Определение связей между объектами ПО. ....	14
1.4.3 Графическое представление концептуальной информационной модели данных.....	15
1.5 Политика безопасности по работе с данными .....	15
1.5.1 Типы пользователей.....	15
1.5.2 Ограничения пользователей по работе с объектами ПО. ....	16
2 Проектирование реляционной модели базы данных .....	17
2.1 Логическое (даталогическое) проектирование модели данных.....	17
2.1.1 Определение отношений и связей между отношениями на основе концептуальной информационной модели. Первичные и внешние ключи....	17
2.1.2 Нормализация логической модели данных. ....	18
2.1.3 Графическое представление логической модели данных в методологии IDEF1x. ....	18
. 2.2 Физическое проектирование с учетом выбранной СУБД. ....	19

2.2.1 Определение типов данных атрибутов отношений.....	19
2.2.2 Графическое представление физической модели данных в методологии IDEF1x .....	22
2.3 Представление физической модели данных на языке SQL.....	23
2.4 Представление политики безопасности пользователей на языке SQL	
24	
3 Программный комплекс для работы с СУБД.....	27
3.1 Учетная запись «Администратор» .....	27
3.2 Учетная запись «Раселитель» .....	44
3.3 Учетная запись «Комендант» .....	56
4 Заключение .....	62
5 Список используемых источников.....	64
Приложение А .....	65
Приложение Б .....	68
Приложение В .....	70
Приложение Г .....	72
Приложение Д .....	76
Приложение Е.....	79
Приложение Ё.....	81

# 1 Концептуальное (инфологическое) проектирование предметной области (ПО).

## 1.1 Неформальное описание ПО с использованием естественного языка.

В качестве предметной области выбран «Расселение студентов в общежитии». Процесс заселение будущего студента начинается с обращения абитуриента к реселителю общежития с вопросом о заселении. Студенту выдается договор о спальном месте в общежитие. Обязательными графами являются:

1. ФИО;
2. Документ удостоверяющий личность студента;
3. Факультет и номер группы студента;
4. Дата заполнения и подпись студента.

Далее реселитель общежития обязан рассказать студенту о правила проживания в общежитии. Далее реселитель обязан выделить студенту свободную комнату с приемлемыми условиями проживания.

Мало кто из иногородних студентов, приезжающих учиться в ВУЗ, может себе позволить снять отдельную квартиру.

Многие вынуждены селиться в студенческие общежития. И хотя сами студенты далеко не всегда довольны жилищными условиями, которые им предлагает вуз, а депутаты подвергают сомнению безопасность проживания в существующих общежитиях, такое жилье остается востребованным по сей день.

Я поняла, что удобное расселение в общежитии имеет свои отрицательные моменты. Именно поэтому я выбрала тему курсовой работы «Расселение студентов в общежитии». Неоднократно слышала от наших реселителей, что excel совсем не удобен для заселения студентов. Программа

все время дагает и не может корректно работать, когда это так надо, поэтому я считаю, что моя разработка даст удобства для наших расселителей.

Почему нужно использовать именно мою программу?

1. Удобства в использование. Не нужно миллион лет искать ну или иную фамилию, комнату или номер группы.

2. Понятность. Программа на столько понятная, что даже инструкции не нужно для нее.

3. Красивый дизайн. Можете посмеяться, но мое мнение такого, что должна красота присутствовать в работе, ибо не будет желания даже к ней прикасаться.

## 1.2 Описание бизнес-процессов ПО в методологии функционального моделирования IDEF0.

### 1.2.1 Описание ПО в методологии функционального моделирования IDEF0 до внедрения автоматизированной системы

Для начала было необходимо описать бизнес-процесс «Заселение студента в общежитие» в методологии IDEF0 верхнего уровня (рисунок 2.1) и его декомпозиции (рисунок 2.2)



Рисунок 2.1 – IDEF0 диаграмма 0 уровня

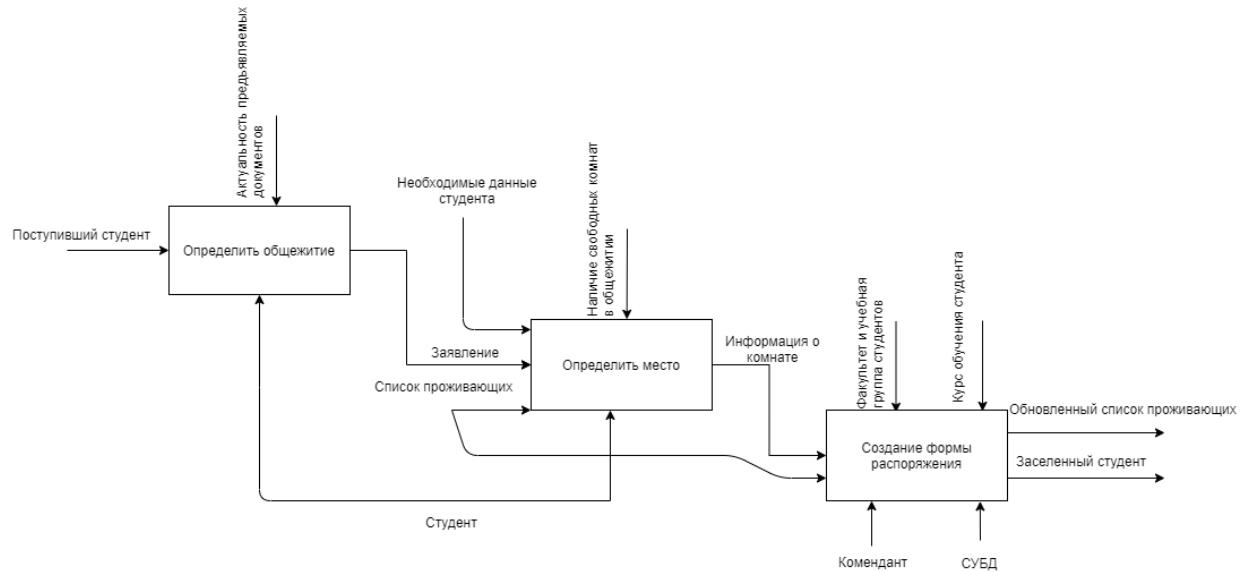
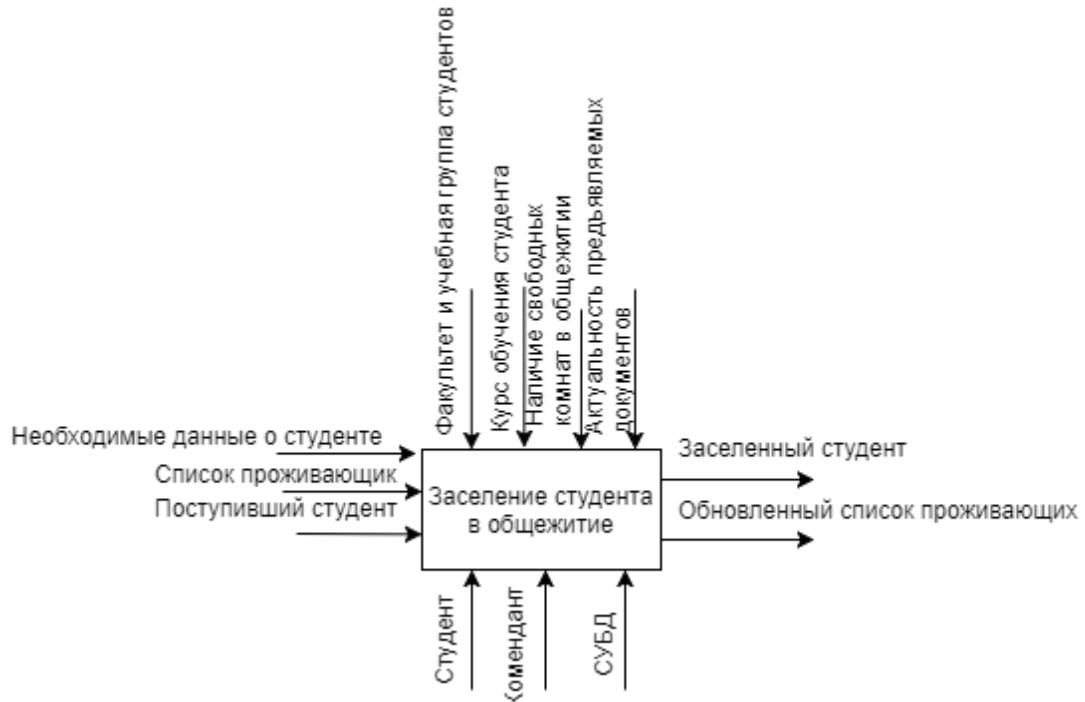


Рисунок 2.2 – IDEF – диаграмма 1 уровня

### 1.2.2 Описание ПО в методологии функционального моделирования IDEF0 после внедрения автоматизированной системы

Рисунок 2.3 – IDEF диаграмма 0 уровня после внедрения  
автоматизированной системы

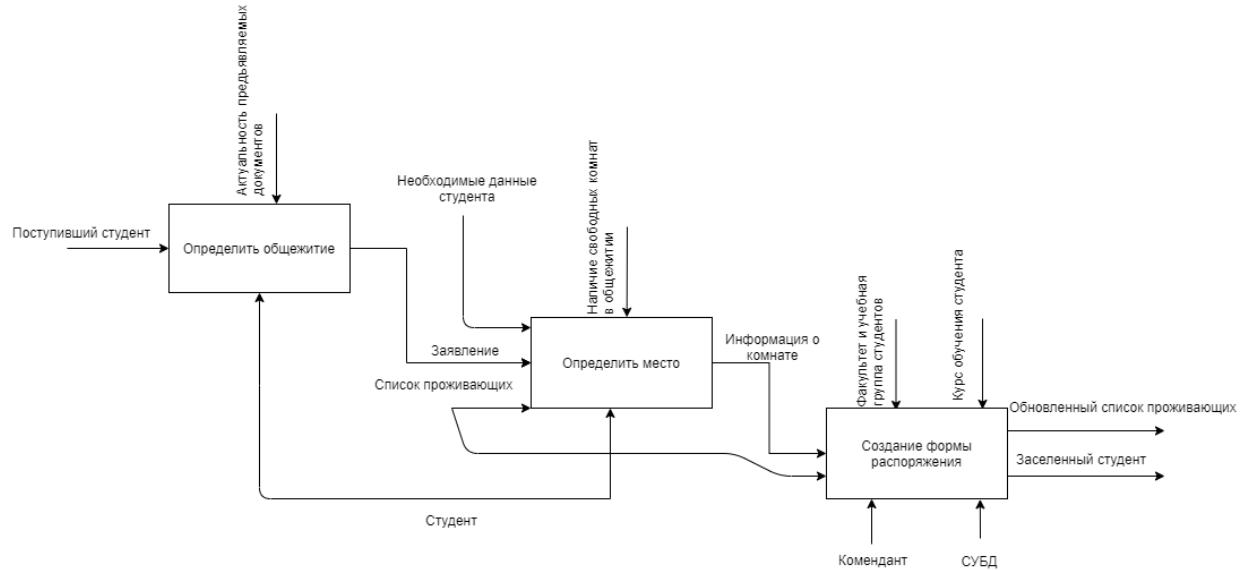


Рисунок 2.4 – IDEF диаграмма 1 уровня после внедрения автоматизированной системы

### 1.3 Цели автоматизации ПО

Целями автоматизации ПО является:

- Хранение всей необходимой информации в виде БД;
- Удобство в пользовании;
- Экономия времени и ресурсов;
- Непрерывность работы;
- Возможность просмотра необходимой информации для коменданта без участия расселителей.

## 1.4 Концептуальная информационная модель данных для ПО.

1.4.1 Основные объекты ПО, информация о которых будет накапливаться в БД. Их характеристики и свойства (атрибуты).

Основные объекты ПО, их характеристики и атрибуты представлены в таблице 1.

Таблица 1 – Основные объекты ПО

Имя сущности	Характеристика	Атрибуты
Студенты	Информация, которая показывает ФИО заселяющего студента	– Id – Фамилия – Имя – Отчество – Номер группы – Курс обучения
Заявление	Информация об заселении студента	– Id – Номер записи – Дата – FK_Студента
Распоряжение	Информация об заселении студента	– Id – FK_Студента – Дата – Дата заселения – Дата выселения
Комната	Информация о комнате куда заселили студента	– Id – FK_Общежития – Кол-во жителей – Этаж – Номер комнаты
Общежитие	В какое общежитие был заселен студент	– Id – Адрес

#### 1.4.2 Определение связей между объектами ПО.

1. Студент – Заявление на заселение. Студент может подать несколько заявления на заселение в общежитие.

Мощность связи: один ко многим (1:M).

2. Студент – Комната. Студенту могут предоставить несколько комнат.

Мощность связи: многое ко многим (M:M).

3. Комната – Распоряжение. Комната включает в себя распоряжение на заселение студента.

Мощность связи: одно к одному (1:1).

4. Общежитие – Комната. В одном общежитии имеются много комнат для заселения студента.

Мощность связи: один ко многим (1:M).

5. Распоряжение – Заявление на заселения. Распоряжение составляется на основе заявления на заселения.

Мощность связи: одно к одному (1:1).

### 1.4.3 Графическое представление концептуальной информационной модели данных.

Концептуальная информационная модель данных (рисунок 2.5).

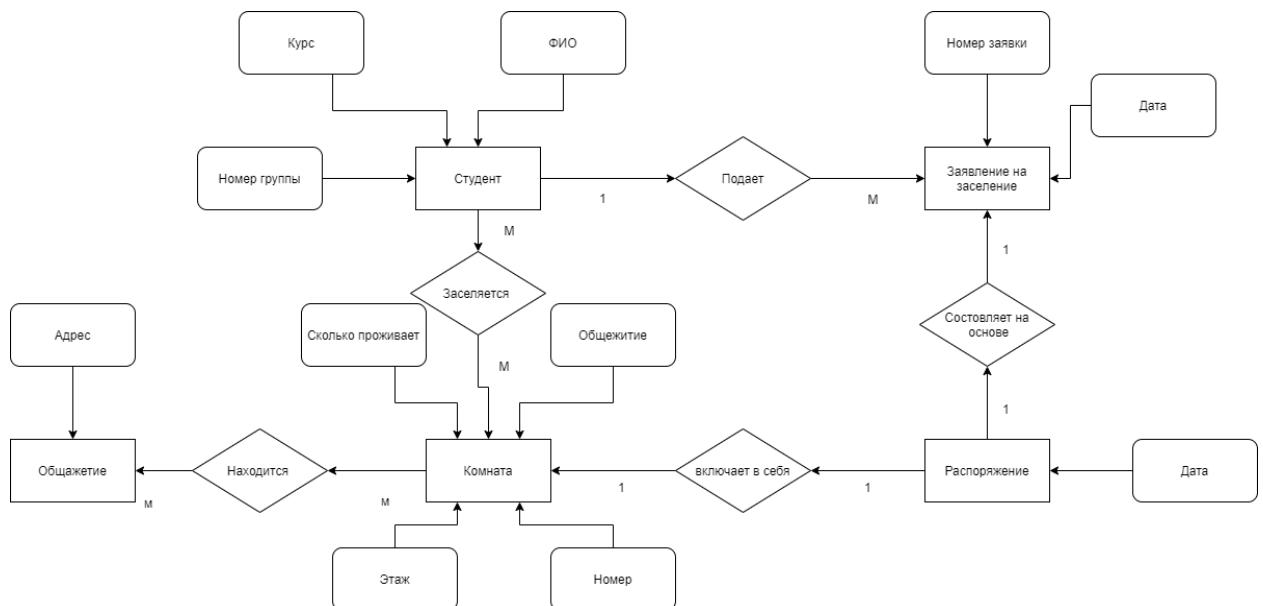


Рисунок 2.5 – Концептуальная информационная модель данных

### 1.5 Политика безопасности по работе с данными

#### 1.5.1 Типы пользователей.

В данной автоматизированной информационной системе присутствует три типа пользователей «Комендант», «Раселитель», «Администратор».

Коллективное использование базы данных требует административного контроля. Необходимо поручить эти обязанности одному или нескольким сотрудникам, которые будут выполнять роль администраторов базы данных. «Администратор» имеет полный доступ к таблицам базы данных.

Администрирование базы данных – это функция управления базой данных. «Администратор» следит за тем, чтобы база данных не вышла из строя, или в противном случае остались резервные копии. Также

«Администратор» отвечает за выработку требований к базе данных, её проектирование, реализацию. Не менее важной функцией администратора БД является поддержка целостности базы данных.

Пользователи, наделенные ролью «Комендант» могут просматривать информацию о студентах (но только основную, таблицу с дополнительной информацией о студенте они просматривать не могут), комнату, могут посмотреть их фото и описание.

«Раселитель» также, как и «Комендант» может просматривать информацию: о студентах (помимо основной, ещё и таблицу с дополнительной информацией).

### 1.5.2 Ограничения пользователей по работе с объектами ПО.

Администратор имеет возможность выбирать (SELECT), добавлять (INSERT), изменять (UPDATE), удалять строки (DELETE) во всех таблицах.

Раселитель имеет возможность выбирать (SELECT), добавлять (INSERT), изменять (UPDATE) строки в некоторых таблицах.

Комендант имеют возможность выбирать (SELECT) и добавлять (INSERT) строки в некоторых таблицах.

Матрица доступа представлена в таблице 2

Таблица 2 – Матрица доступа

Польз-ли Таблица	Администратор	Раселитель	Комендант
Студент	SELECT, INSERT, UPDATE, DELETE	SELECT, INSERT, UPDATE, DELETE	SELECT,
Заявление	SELECT, INSERT, UPDATE, DELETE	SELECT, INSERT, UPDATE, DELETE	-
Распоряжение	SELECT, INSERT, UPDATE, DELETE	SELECT, INSERT, UPDATE, DELETE	SELECT, INSERT

Общежитие	SELECT, INSERT, UPDATE, DELETE	SELECT	SELECT
Комната	SELECT, INSERT, UPDATE, DELETE	SELECT, INSERT, UPDATE, DELETE	SELECT

## 2 Проектирование реляционной модели базы данных

### 2.1 Логическое (даталогическое) проектирование модели данных.

2.1.1 Определение отношений и связей между отношениями на основе концептуальной информационной модели. Первичные и внешние ключи.

В базе данных присутствуют 6 таблиц

Таблица «Студент» предназначена для хранения информации о студентах, хранит в себе «Id», «Фамилия», «Имя», «Отчество», «Номер группы» и «Курс обучения» (администратор/раселитель/комендант)

Таблица «Заявление» предназначена для хранения информации о студентах, которые подали заявление на заселение в общежитие, хранит в себе «Id», «Номер записи», «Дата» «FK\_Студента».

Таблица «Распоряжение» предназначена для хранения информации о студентах, которые подали заявление на заселение в общежитие и рассматривается университетом возможно ли заселение, хранит в себе «Id», «FK\_Студента», «Дата заселения» и «Дата выселения».

Таблица «Комната» предназначена для хранения информации о комнате, куда был заселен студент, хранит в себе «Id», «FK\_Общежития», «Кол-во жителей», «Этаж» и «Номер комнаты».

Таблица «Общежитие» предназначена для хранения информации о общежитии, куда был заселен студент, хранит в себе «Id» и «Адрес».

### 2.1.2 Нормализация логической модели данных.

Нормализация не требуется, логическая модель уже приведена к третьей нормальной форме

### 2.1.3 Графическое представление логической модели данных в методологии IDEF1x.

На рисунке 2.6 представлена логическая модель базы данных, она включает все сущности базы с указанием первичных и внешних ключей. Также продемонстрированы все отношения между таблицами

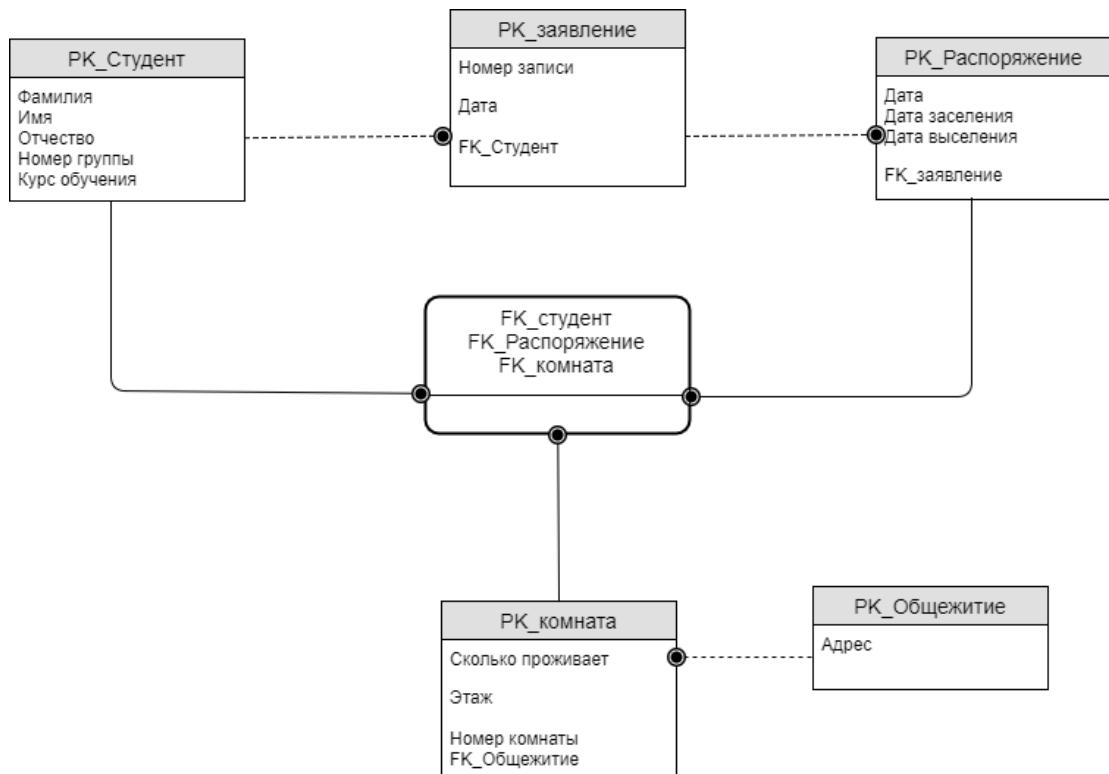


Рисунок 2.6 - Логическая модель БД

. 2.2 Физическое проектирование с учетом выбранной СУБД.

2.2.1 Определение типов данных атрибутов отношений.

Ниже представлено описание таблицы Студент (таблица 3)

Таблица 3 – Описание таблицы Студент

Имя поля	Тип данных поля	Описание	Ограничения
PK_Студент	Числовой	Уникальный идентификатор студента.	Первичный ключ; условие на значение: >0.
Фамилия	Текстовый	Фамилия студента	Размер поля: 30. Обязательное поле.
Имя	Текстовый	Имя студента	Размер поля: 30. Обязательное поле.
Отчество	Текстовый	Отчество студента	Размер поля: 30.
Номер группы	Текстовый	Номер группы студента	Размер поля: 15. Обязательное поле.
Курс обучения	Числовой	Курс обучения в ВУЗе	Обязательное поле.

Ниже представлено описание таблицы «Заявление» (таблица 4).

Таблица 4 – Описание таблицы «Заявление»

Имя поля	Тип данных поля	Описание	Ограничения
Дата	Текстовый	Дата заполнения заявления	Размер поля: 30. Обязательное поле.
PK_заявление	Числовой	Уникальный идентификатор заявления.	Первичный ключ; условие на значение: >0.
FK_студент	Числовой	Сылается на таблицу Студент.	Внешний ключ; условие на значение: >0; обязательное поле.

Ниже представлено описание таблицы Распоряжение (таблица 5)

Таблица 5 – Описание таблицы «Распоряжение»

Имя поля	Тип данных поля	Описание	Ограничения
Дата	Текстовый	Дата заполнения заявления	Размер поля: 32. Обязательное поле.
Дата заселения	Текстовый	Дата заселения в общежитие	Размер поля: 32. Обязательное поле.
Дата выселения	Текстовый	Дата выселения в общежитие	Размер поля: 32. Обязательное поле.
PK_Распоряжение	Числовой	Уникальный идентификатор распоряжения.	Первичный ключ; условие на значение: >0.
FK_заявление	Числовой	Ссылается на таблицу Заявление.	Внешний ключ; условие на значение: >0; обязательное поле.

Ниже представлено описание таблицы «Заселенный\_студент» (таблица 6)

Таблица 6 – Описание таблицы «Заселенный\_студент»

Имя поля	Тип данных поля	Описание	Ограничения
Заселенный_студент	Логический	Проверка данных	Обязательное поле.
FK_студент	Числовой	Ссылается на таблицу Студент	Внешний ключ; условие на значение: >0; обязательное поле.
FK_распоряжение	Числовой	Ссылается на таблицу Распоряжение	Внешний ключ; условие на значение: >0; обязательное поле.

FK_комната	Числовой	Сылается на таблицу Комната.	Внешний ключ; условие на значение: >0; обязательное поле.
------------	----------	------------------------------	---

Ниже представлено описание таблицы «Комната» (таблица 7)

Таблица 7 – Описание таблицы «Комната»

Имя поля	Тип данных поля	Описание	Ограничения
Сколько проживает	Числовой	Кол-во проживающих в комнате	Размер поля: 30. Обязательное поле
Этаж	Числовой	Номер этажа в общежитии	Размер поля: 30. Обязательное поле
Номер комнаты	Числовой	Номер комнаты в общежитии	Размер поля: 30. Обязательное поле
PK_комната	Числовой	Уникальный идентификатор комнаты.	Первичный ключ; условие на значение: >0.
FK_комната	Числовой	Сылается на таблицу Комната.	Внешний ключ; условие на значение: >0; обязательное поле.

Ниже представлено описание таблицы «Общежитие» (таблица 8)

Таблица 8 – Описание таблицы «Общежитие»

Имя поля	Тип данных поля	Описание	Ограничения
PK_Общежитие	Числовой	Уникальный идентификатор комнаты.	Первичный ключ; условие на значение: >0.
Адрес	Текстовый	Адрес общежития.	Размер поля: 30. Обязательное поле.

## 2.2.2 Графическое представление физической модели данных в методологии IDEF1x

На рисунке 2.7 представлена логическая модель базы данных, она включает все сущности базы с указанием первичных и внешних ключей. Также продемонстрированы все отношения между таблицами.

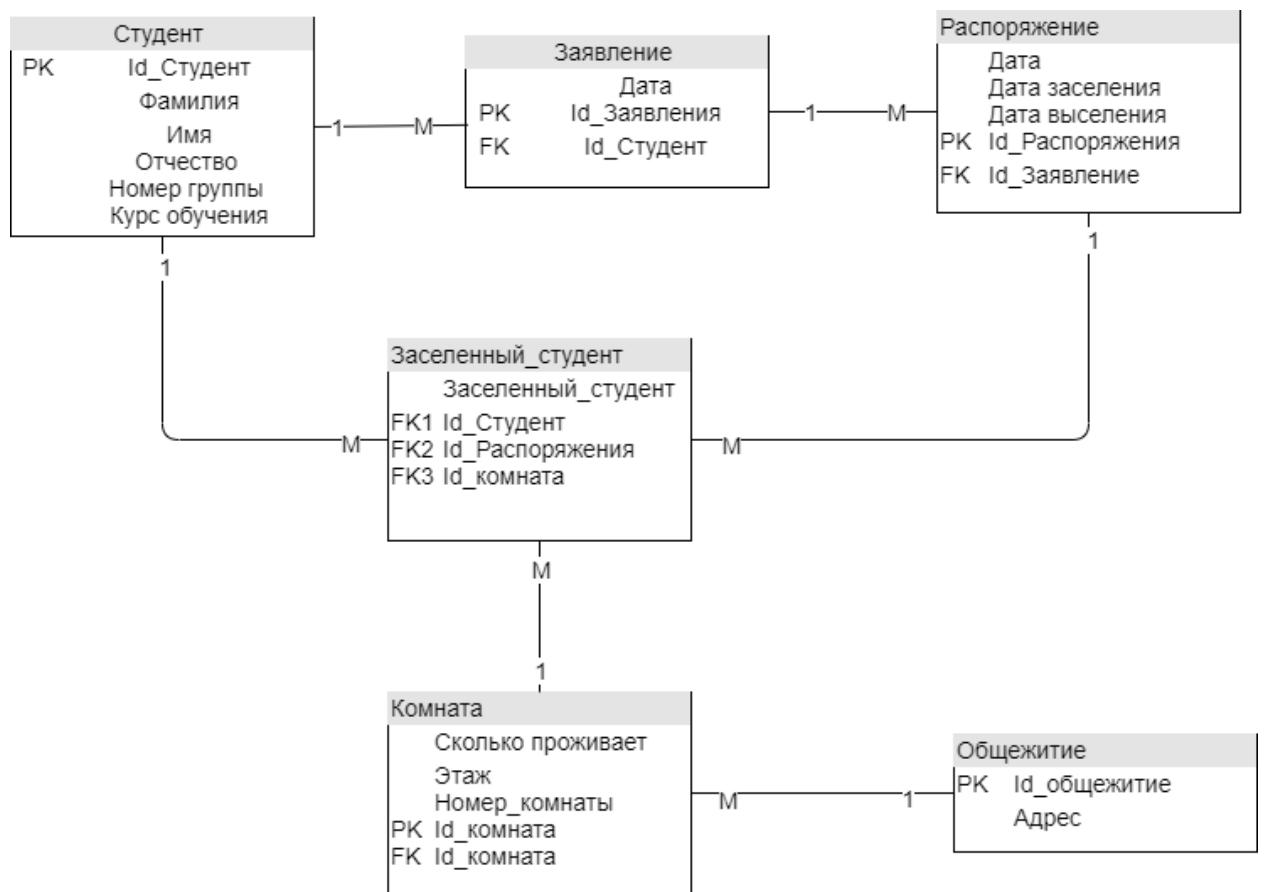


Рисунок 2.7 – Физическая модель

### 2.3 Представление физической модели данных на языке SQL.

Все первичные и вторичные ключи имеют тип данных «INT», так как хранят в себе id.

Для всех атрибутов показаны их типы и их ограничения (рисунок 2.8)

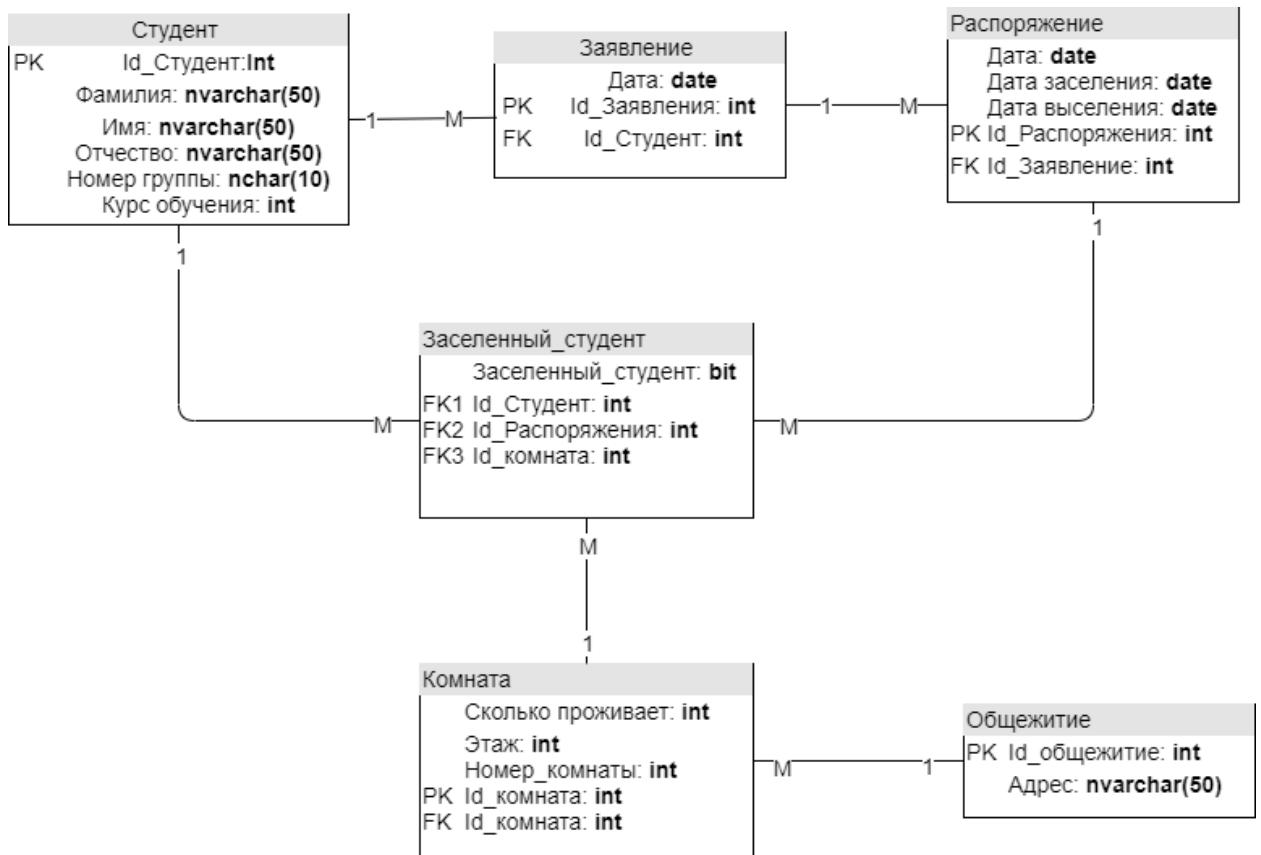


Рисунок 2.8 – Физическая модель БД

Так же была создана диаграмма в Microsoft SQL Server Management Studio (MSSMS) (рисунок 2.9)

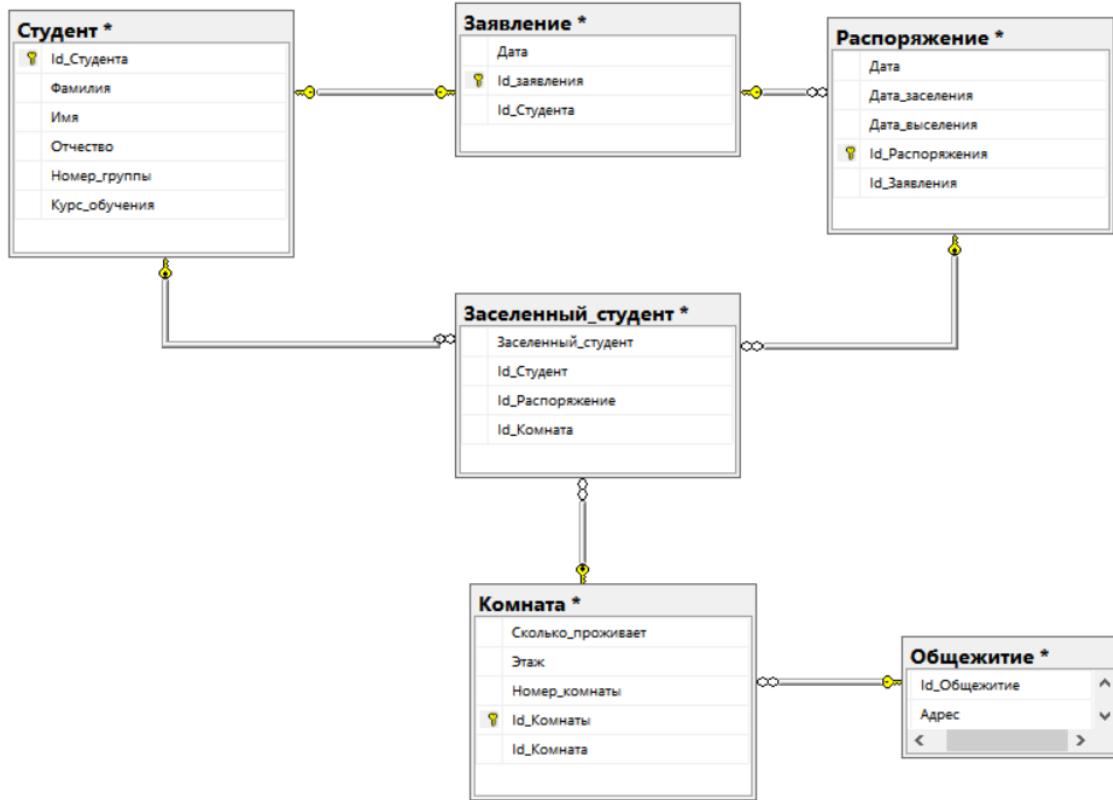


Рисунок 2.9 – Диаграмма БД

## 2.4 Представление политики безопасности пользователей на языке SQL

Для реализации политики безопасности была выбрана реализации на уровне программы. Была создана дополнительная база данных «Users», далее были добавлены столбцы «Логин, Пароль, Роль».

SQL Код:

```

"CREATE TABLE users("id" INTEGER PRIMARY KEY
AUTOINCREMENT,
login TEXT(24),
password TEXT(32),
role TEXT(20));"
  
```

Далее были добавлены пользователи (рисунок 2.10).

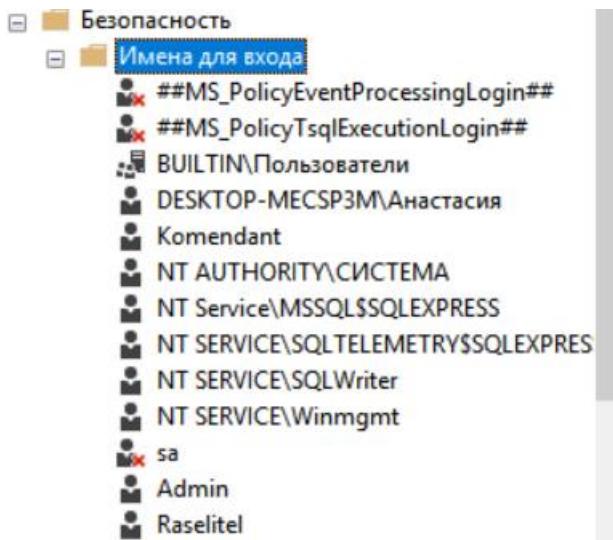


Рисунок 2.10 – Пользователи

Так же на уровне программы был составлен код на языке программирования C# для проверки авторизации и входа в программу по ролям.

Создание ролей:

На рисунке 2.11 представлена политика безопасности для «Admin».

```

CREATE ROLE admin;
GRANT SELECT, INSERT, DELETE, UPDATE ON Студент TO Admin;
GRANT SELECT, INSERT, DELETE, UPDATE ON Распоряжение TO Admin;
GRANT SELECT, INSERT, DELETE, UPDATE ON Общежитие TO Admin;
GRANT SELECT, INSERT, DELETE, UPDATE ON Комната TO Admin;
GRANT SELECT, INSERT, DELETE, UPDATE ON Заявление TO Admin;
  
```

Рисунок 2.11 – Политика безопасности «Admin»

На рисунке 2.12 представлена политика безопасности для «Komendant».

The screenshot shows a SQL query window titled "SQLQuery1.sql - D...3M\Анастасия (52)\*". The code defines a role "Komendant" and grants various permissions on tables "Студент", "Заявление", "Распоряжение", "Общежитие", and "Комната" to this role. It includes both GRANT and DENY statements for SELECT, INSERT, DELETE, and UPDATE operations.

```
CREATE ROLE Komendant;
GRANT SELECT ON Студент TO Komendant;
DENY INSERT, DELETE, UPDATE ON Студент TO Komendant;
DENY SELECT, INSERT, DELETE, UPDATE ON Заявление TO Komendant;
DENY SELECT, INSERT ON Распоряжение TO Komendant;
DENY DELETE, UPDATE ON Распоряжение TO Komendant;
GRANT SELECT ON Общежитие TO Komendant;
DENY INSERT, DELETE, UPDATE ON Общежитие TO Komendant;
GRANT SELECT ON Комната TO Komendant;
DENY INSERT, DELETE, UPDATE ON Комната TO Komendant;
```

Рисунок 2.12 – Политика безопасности «Komendant»

На рисунке 2.13 представлена политика безопасности для «Raselitel».

The screenshot shows a SQL query window titled "SQLQuery1.sql - D...3M\Анастасия (52)\*". The code defines a role "Raselitel" and grants various permissions on tables "Студент", "Распоряжение", "Общежитие", "Заявление", and "Комната" to this role. It includes both GRANT and DENY statements for SELECT, INSERT, DELETE, and UPDATE operations.

```
CREATE ROLE Raselitel;
GRANT SELECT, INSERT, DELETE, UPDATE ON Студент TO Raselitel;
GRANT SELECT, INSERT, DELETE, UPDATE ON Распоряжение TO Raselitel;
GRANT SELECT ON Общежитие TO Raselitel;
DENY INSERT, DELETE, UPDATE ON Общежитие TO Raselitel;
GRANT SELECT, INSERT, DELETE, UPDATE ON Заявление TO Raselitel;
GRANT SELECT, INSERT, DELETE, UPDATE ON Комната TO Raselitel;
```

Рисунок 2.13 – Политика безопасности «Raselitel»

### 3 Программный комплекс для работы с СУБД

#### 3.1 Учетная запись «Администратор»

Для создания программы была выбрана среда программирования Visual Studio 2017. Программа разрабатывалась с помощью Windows Form на языке программирования C#.

При запуске программы появляется окно входа, куда нужно ввести данные администратора (рисунок 3.1).

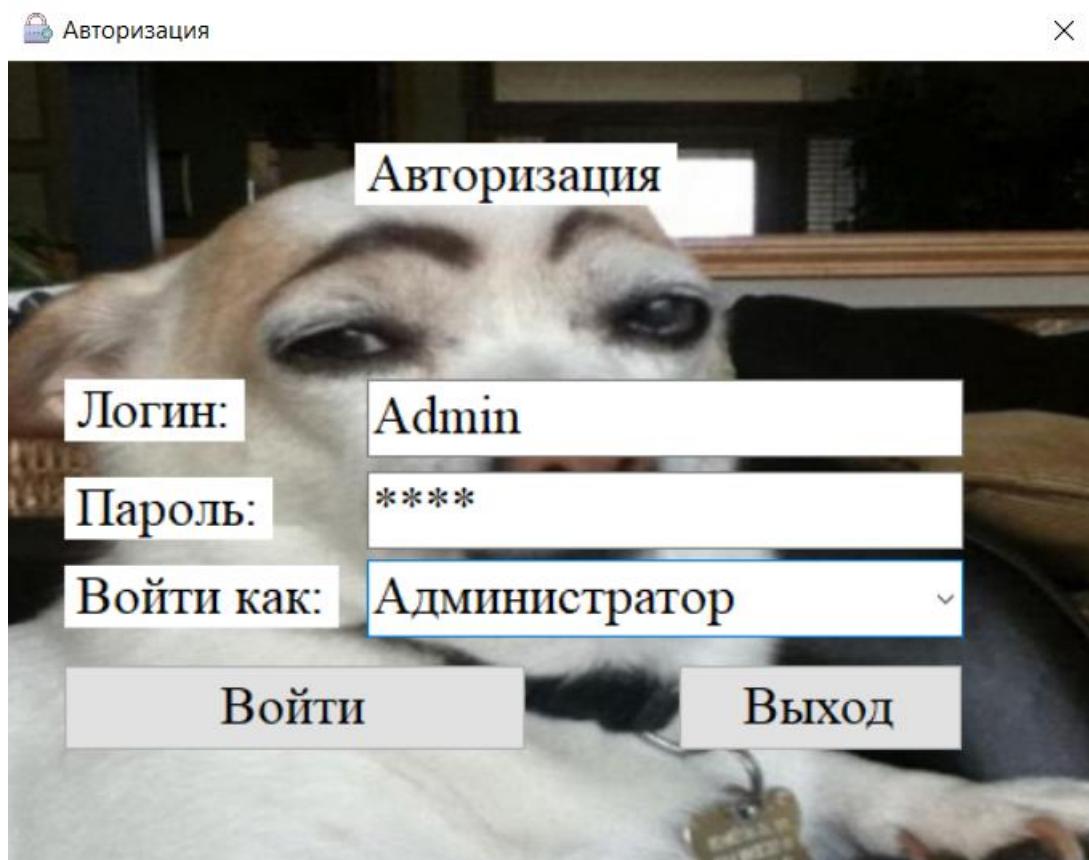


Рисунок 3.1 – Окно входа

Если пользователь ввел не правильный пароль, то появляется окно о неправильно введен логине или пароле (рисунок 3.2).

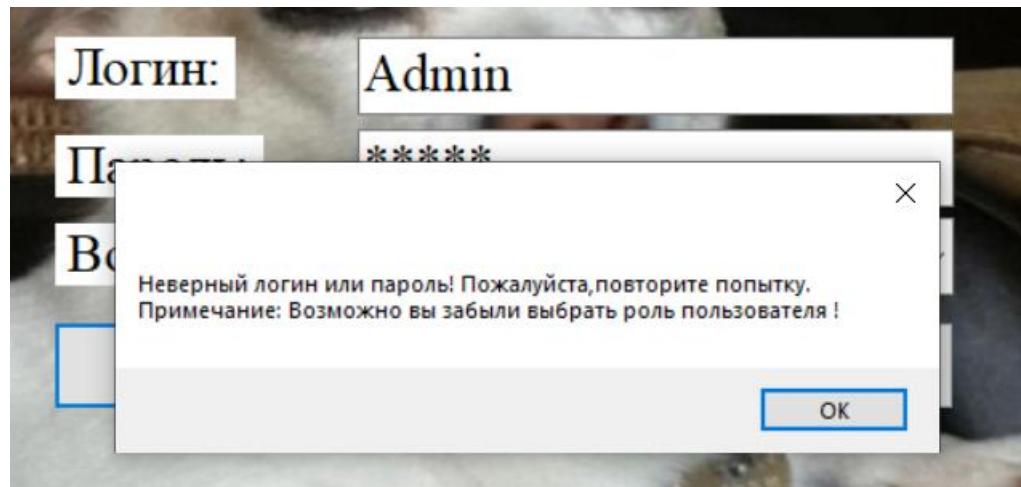


Рисунок 3.2 – Неверный ввод данных

После удачного входа администратор попадает на главное меню программы (рисунок 3.3).

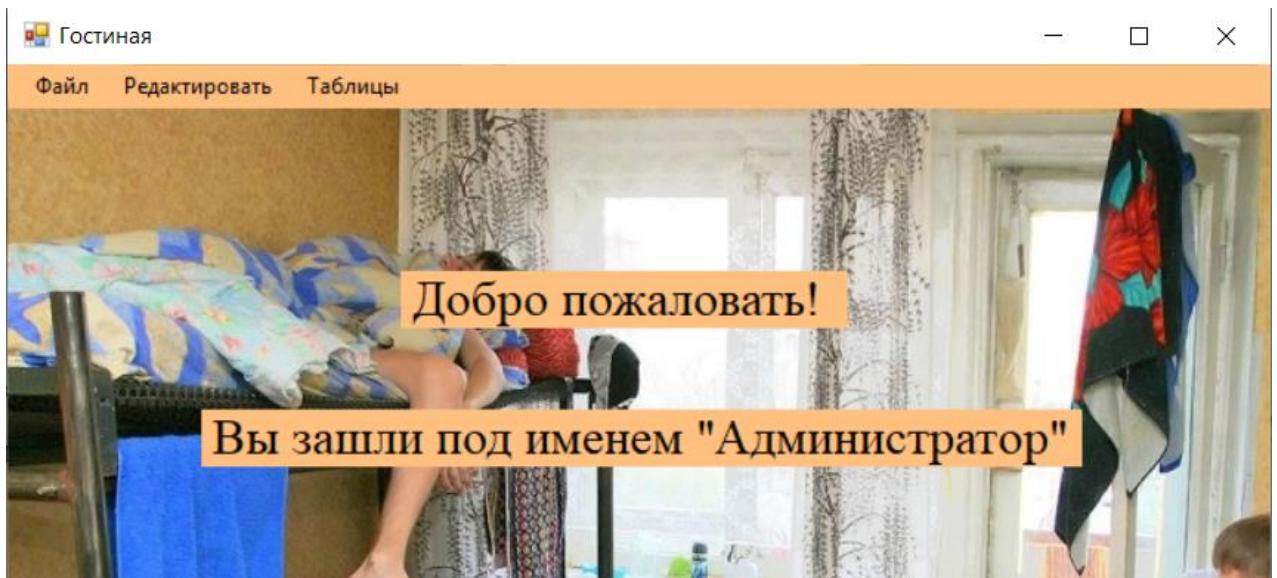


Рисунок 3.3 – Главное меню

Вверху окна находятся навигационное меню (рисунок 3.4), если нажать на кнопку «Файл» (рисунок 3.5), то перед администратором появится раздел «О программе» и «Выйти».

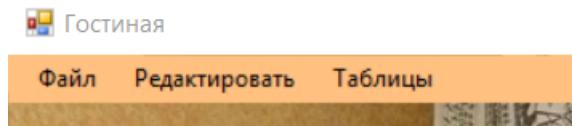


Рисунок 3.4 – Навигационное меню

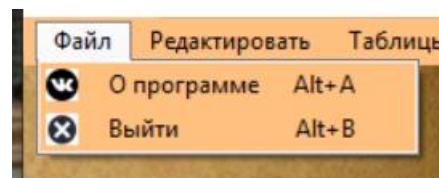


Рисунок 3.5 – Файл

Нажав на раздел «О программе» выйдет диалоговое окно, где будут написаны данные о разработчике (рисунок 3.6).

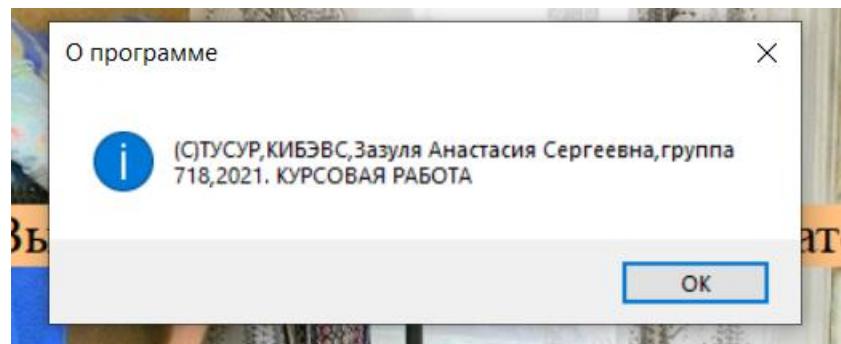


Рисунок 3.6 – Описание программы

Нажав на раздел «Выход» вылезет диалоговое окно, которое просит пользователя подтвердить выход из программы (рисунок 3.7).

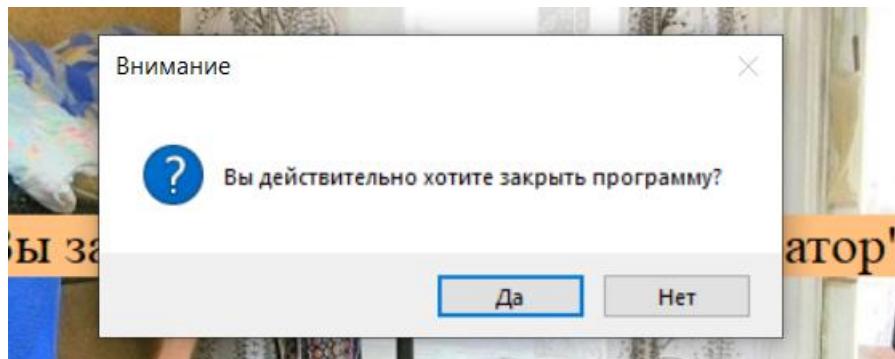


Рисунок 3.7 – Выйти

Если нажать на кнопку «Редактировать», перед администратором появятся 3 раздела «Запросы по данным», «Изменение данных» и «Операторы» (рисунок 3.8).

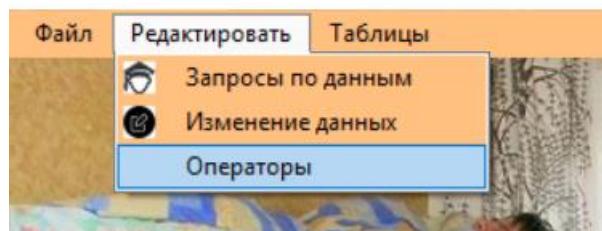


Рисунок 3.8 – «Редактировать»

Перейдя в раздел «Запросы по данным» на экране у администратора появится окно с подразделами «Информация по таблицам» и «Заселенные студенты» (рисунок 3.9).

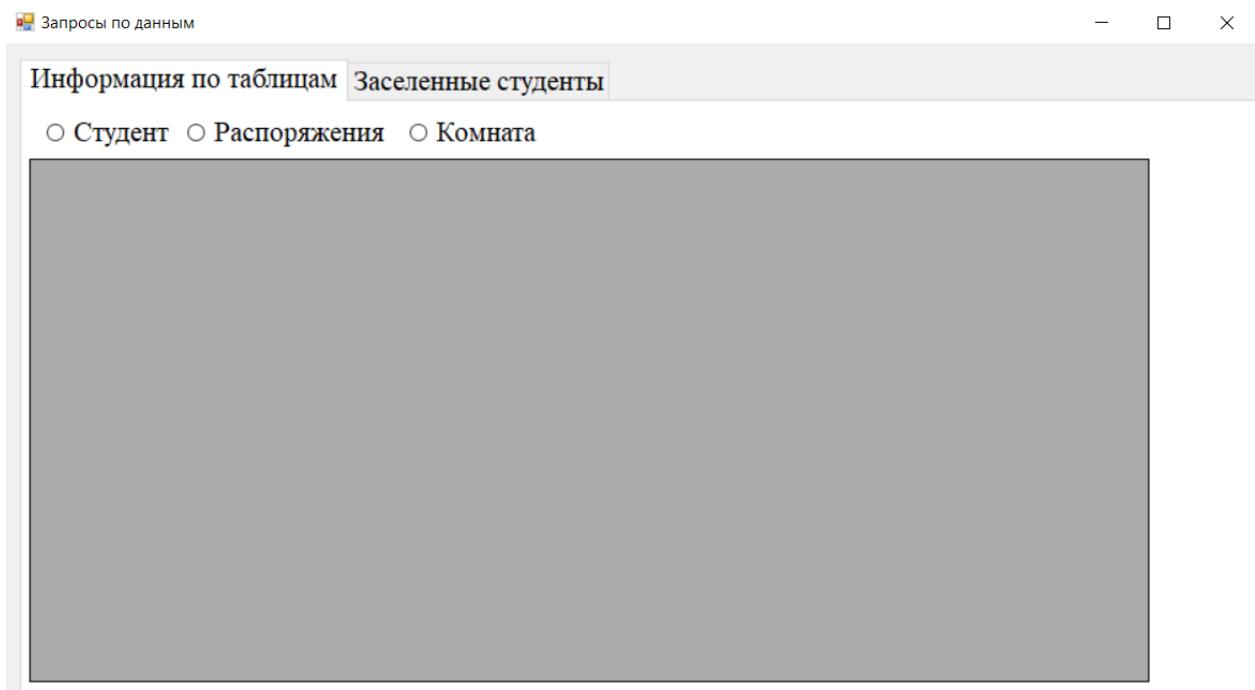


Рисунок 3.9 – «Информация по таблицам»

Нажав на пометку «Студент» перед администратором появится информация о номере студента, ФИО, номер группы и курс обучения (рисунок 3.10).

	Id_Студен	Фамилия	Имя	Отчество	Номер_группы	Курс_обуче
▶	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	3	Смирнов	Алексей		429	2
	4	Кузнецо...	Надежда	Витальевна	728-2	3
	5	Буракша...	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	8	Логинов	Сергей	Николаевич	656	5
*	9	Романова	Татьяна	Станиславовна	327	4

Рисунок 3.10 – Информация о студентах

Нажав на пометку «Распоряжения» перед администратором появится информация о номере распоряжения, номер заявления, дата, когда было подано заявление, дата заселения и дата выселения (рисунок 3.11).

Информация по таблицам Заселенные студенты					
<input type="radio"/> Студент <input checked="" type="radio"/> Распоряжения <input type="radio"/> Комната					
	Id_Распор	Id_Заявлеи	Дата	Дата_заселения	Дата_выселения
▶	1	1	18.05.2021	15.05.2019	31.08.2019
	2	2	23.04.2020	25.08.2020	31.08.2020
	4	4	03.04.2020	20.03.2020	31.08.2020
	5	5	14.03.2020	14.10.2020	31.08.2020
	6	6	23.07.2020	11.05.2020	31.08.2020
	3	3	02.03.2021	28.01.2021	31.08.2021
	7	7	30.12.2020	22.12.2020	31.08.2021
	8	8	02.08.2021	15.01.2021	31.08.2021
	9	9	21.10.2020	07.08.2020	31.08.2021
	10	10	15.06.2021	11.01.2021	31.08.2021
*					

Рисунок 3.11 – Информация о распоряжениях

Нажав на пометку «Комната» перед администратором появится информация о номере комнаты, сколько студентов проживают, этаж, номер комнаты, номер студента адрес общежития и заселен студент на данный момент или нет (рисунок 3.12).

Информация по таблицам Заселенные студенты							
<input type="radio"/> Студент <input type="radio"/> Распоряжения <input checked="" type="radio"/> Комната							
	Id_Комнат	Сколько_1	Этаж	Номер_ко	Id_Студен	Адрес	Заселенный_студен
▶	1	4	6	20	1	Ф.Лытки...	<input checked="" type="checkbox"/>
	2	3	4	38	2	Ф.Лытки...	<input checked="" type="checkbox"/>
	2	3	4	38	4	Ф.Лытки...	<input checked="" type="checkbox"/>
	2	3	4	38	5	Ф.Лытки...	<input checked="" type="checkbox"/>
	3	2	2	2	3	Ф.Лытки...	<input type="checkbox"/>
	3	2	2	2	7	Ф.Лытки...	<input type="checkbox"/>
	5	4	5	10	10	Ф.Лытки...	<input type="checkbox"/>
	6	3	8	40	9	Ф.Лытки...	<input checked="" type="checkbox"/>
	7	2	7	4	6	Ф.Лытки...	<input checked="" type="checkbox"/>
	7	2	7	4	8	Ф.Лытки...	<input checked="" type="checkbox"/>
*							<input type="checkbox"/>

Рисунок 3.12 – Информация о комнатах

Перейдя на подраздел «Заселенный студент» перед администратором появится окно, в котором нужно вписать фамилию студента и посмотреть по базе заселен он или нет (рисунок 3.13).

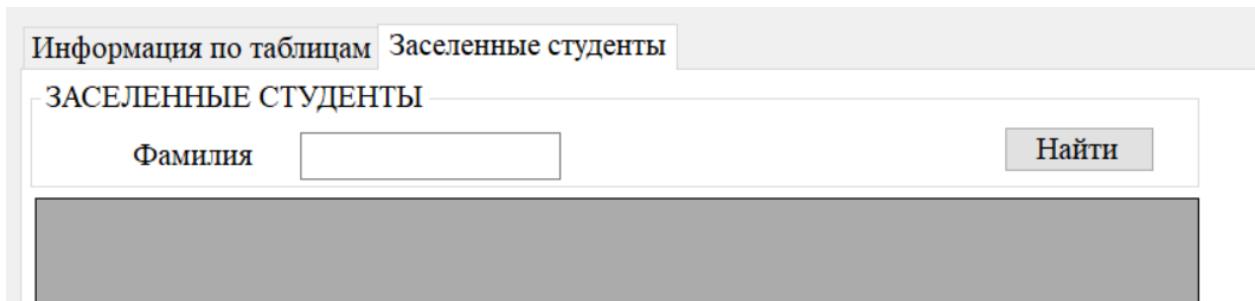


Рисунок 3.13 – «Заселенный студент»

Для примера введем в текстовое окно фамилию студента и нажать кнопу «Найти» (рисунок 3.14 – 3.15).

ЗАСЕЛЕННЫЕ СТУДЕНТЫ						
	Id_Студен	Фамилия	Имя	Отчество	Номер_группы	Заселенный_студент
▶	1	Зазуля	Анастас...		718	<input checked="" type="checkbox"/>
*						<input type="checkbox"/>

Рисунок 3.14 – Результат запроса (1)

ЗАСЕЛЕННЫЕ СТУДЕНТЫ						
	Id_Студен	Фамилия	Имя	Отчество	Номер_группы	Заселенный_студент
▶	9	Романова	Татьяна	Станисл...	327	<input checked="" type="checkbox"/>
*						<input type="checkbox"/>

Рисунок 3.15 – Результат запроса (2)

Перейдя в раздел «Подзапросы по данным» (рисунок 3.16) перед администратором появляется два вида запросы:

1. Коррелированный подзапрос – внутренний запрос для своего выполнения должен получить данные из внешнего запроса. В коррелированном подзапросе внутренний запрос не может быть реализован немедленно: он ссылается на внешний запрос и выполняется поочередно для каждой строки во внешнем запросе (рисунок 3.17 – 3.18);
2. Некоррелированный подзапрос – внутренний запрос выполняется независимо, передавая результаты во внешний запрос (рисунок 3.19 – 3.21).

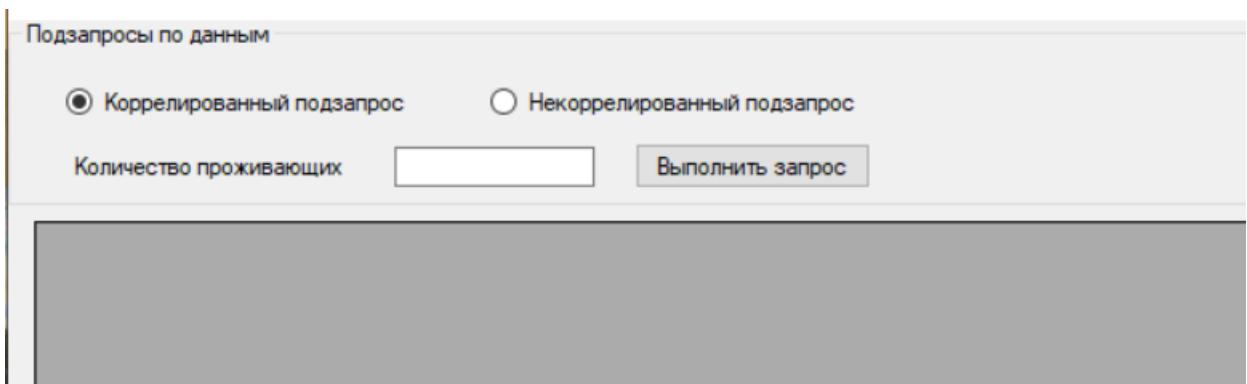


Рисунок 3.16 – «Подзапросы по данным»

	Номер_комнаты	Этаж	Сколько_прожива
▶	20	6	4
	15	9	4
	10	5	4
*			

Рисунок 3.17 – «Коррелированный подзапрос» (1)

Подзапросы по данным

Коррелированный подзапрос     Некоррелированный подзапрос

Количество проживающих

	Номер_комнаты	Этаж	Сколько_прожива
▶	2	2	2
	4	7	2
*			

Рисунок 3.18 – «Коррелированный подзапрос» (2)

Подзапросы по данным

Коррелированный подзапрос     Некоррелированный подзапрос

Количество проживающих

	Номер_комнаты	Этаж	Сколько_прожива
*			

Информация

Нет таких значений!

Рисунок 3.19 – Ошибка при введение некорректных значений при «Некоррелированный подзапрос»

Подзапросы по данным

Коррелированный подзапрос     Некоррелированный подзапрос

Количество проживающих

	Номер_комнаты	Этаж	Сколько_прожива
▶	20	6	4
*			

Рисунок 3.20 – «Некоррелированный подзапрос» (1)

Подзапросы по данным			
<input type="radio"/> Коррелированный подзапрос		<input checked="" type="radio"/> Некоррелированный подзапрос	
Количество проживающих		10	<input type="button" value="Выполнить запрос"/>
	Номер_комнаты	Этаж	Сколько_прожива
▶	10	5	4
*			

Рисунок 3.21 – «Некоррелированный подзапрос» (2)

Перейдя в раздел «Операторы» (рисунок 3.22) перед администратором появится окно, в котором можно добавить (рисунок 3.23), изменить (рисунок 3.24) или удалить данные (рисунок 3.25).

Оператор – это зарезервированное слово, или символ, который используется в SQL выражениях с использованием WHERE для выполнения операции или операций, например, сравнение.

Операторы			
<input checked="" type="radio"/> Добавить данные (ID не вводить при добавлении)	<input type="radio"/> Изменить Данные	<input type="radio"/> Удалить Данные	
ID Студента	<input type="text"/>	<input type="button" value="Выполнить запрос"/>	
Фамилия	<input type="text"/>	Группа студента	<input type="text"/>
Имя	<input type="text"/>	Курс обучения	<input type="text"/>
Отчество	<input type="text"/>		
		<input type="button" value="Показать список"/>	

Рисунок 3.22 – «Операторы»

Операторы

Добавить данные (ID не вводить при добавлении)     Изменить Данные     Удалить Данные

ID Студента	<input type="text"/>	<input type="button" value="Выполнить запрос"/>
Фамилия	<input type="text" value="Иванов"/>	Группа студента <input type="text" value="122"/>
Имя	<input type="text" value="Иван"/>	Курс обучения <input type="text" value="6"/>
Отчество	<input type="text" value="Батькович"/>	

Id_Студент	Фамилия	Имя	Номер_группы	Column1
1	Зазуля	Анастасия	718	2
10	Рыбас	Роман	301-м	2
11	Пупкина	Станиславна	123	2
12	Иванов	Иван	122	6
3	Смирнов	Алексей	429	2
4	Кузнецова	Надежда	728-2	2
5	Буракшаева	Юлия	728-1	2
6	Бирюков	Евгений	740	2
7	Демчук	Алексей	420	2
9	Романова	Татьяна	327	2

Рисунок 3.23 – Добавление данных с помощью оператора

Операторы

Добавить данные (ID не вводить при добавлении)     Изменить Данные     Удалить Данные

ID Студента	<input type="text" value="12"/>	<input type="button" value="Выполнить запрос"/>
Фамилия	<input type="text" value="Иванов"/>	Группа студента <input type="text" value="122"/>
Имя	<input type="text" value="Иваннн"/>	Курс обучения <input type="text" value="6"/>
Отчество	<input type="text" value="Батькович"/>	

Id_Студент	Фамилия	Имя	Номер_группы	Column1
1	Зазуля	Анастасия	718	2
10	Рыбас	Роман	301-м	2
11	Пупкина	Станиславна	123	2
12	Иванов	Иваннн	122	6
3	Смирнов	Алексей	429	2
4	Кузнецова	Надежда	728-2	2
5	Буракшаева	Юлия	728-1	2
6	Бирюков	Евгений	740	2
7	Демчук	Алексей	420	2

Рисунок 3.24 – Изменение данных с помощью оператора

Операторы

Добавить данные (ID не вводить при добавлении)     Изменить Данные     Удалить Данные

ID Студента	12	<input type="button" value="Выполнить запрос"/>	
Фамилия	<input type="text"/>	Группа студента	<input type="text"/>
Имя	<input type="text"/>	Курс обучения	<input type="text"/>
Отчество	<input type="text"/>		

	Id_Студент	Фамилия	Имя	Номер_группы	Column1
▶	1	Зазуля	Анастасия	718	2
	10	Рыбас	Роман	301-м	2
	11	Пупкина	Станиславна	123	2
	3	Смирнов	Алексей	429	2
	4	Кузнецова	Надежда	728-2	2
	5	Буракшаева	Юлия	728-1	2
	6	Бирюков	Евгений	740	2
	7	Демчук	Алексей	420	2

Рисунок 3.25 – Удаление данных с помощью оператора

Перейдя в раздел «Таблицы» перед администратором появляется таблица с информацией о студентах (номер, ФИО, номер группы и курс обучения), для удобства каждый курс обучения был присвоен свой цвет (рисунок 3.26).

Студенты

0 для 0 | ▶ | + | X | 1 | Поиск | Фильтр

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
▶	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	3	Смирнов	Алексей		429	2
	4	Кузнецова	Надежда	Витальевна	728-2	3
	5	Буракшаева	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	8	Логинов	Сергей	Николаевич	656	5
*	9	Романова	Татьяна	Станиславовна	327	4

Рисунок 3.26 – Таблица «Студенты»

В разделе «Студенты» присутствуют такие функции как:

1. «Поиск» – поиск производится на основе введенных данных администратором для быстрого поиска информации (рисунок 3.27).
2. Фильтр – фильтрует информацию (рисунок 3.28).

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	3	Смирнов	Алексей		429	2
	4	Кузнецова	Надежда	Витальевна	728-2	3
▶	5	Буракшаева	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	8	Погинов	Сергей	Николаевич	656	5
	9	Романова	Татьяна	Станиславовна	327	4
*						

Рисунок 3.27 – «Поиск»

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	3	Смирнов	Алексей		429	2
▶	1	Зазуля	Анастасия		718	3
	4	Кузнецова	Надежда	Витальевна	728-2	3
	5	Буракшаева	Юлия	Сергеевна	728-1	3
	9	Романова	Татьяна	Станиславовна	327	4
	8	Погинов	Сергей	Николаевич	656	5
	10	Рыбас	Роман	Владиславович	301-м	6
*						

Рисунок 3.28 – «Фильтр»

Перейдя в раздел «Распоряжение» перед администратором появляется информация о номере распоряжения и заявления, когда было написано заявление, дата заселения и дата выселения студента (рисунок 3.29).

	Id_Распоряжение	Id_Заявление	Дата	Дата_заселения	Дата_выселения
▶	1	1	18.05.2021	15.05.2019	31.08.2019
	2	2	23.04.2020	25.08.2020	31.08.2020
	3	3	02.03.2021	28.01.2021	31.08.2021
	4	4	03.04.2020	20.03.2020	31.08.2020
	5	5	14.03.2020	14.10.2020	31.08.2020
	6	6	23.07.2020	11.05.2020	31.08.2020
	7	7	30.12.2020	22.12.2020	31.08.2021
	8	8	02.08.2021	15.01.2021	31.08.2021
	9	9	21.10.2020	07.08.2020	31.08.2021
	10	10	15.06.2021	11.01.2021	31.08.2021

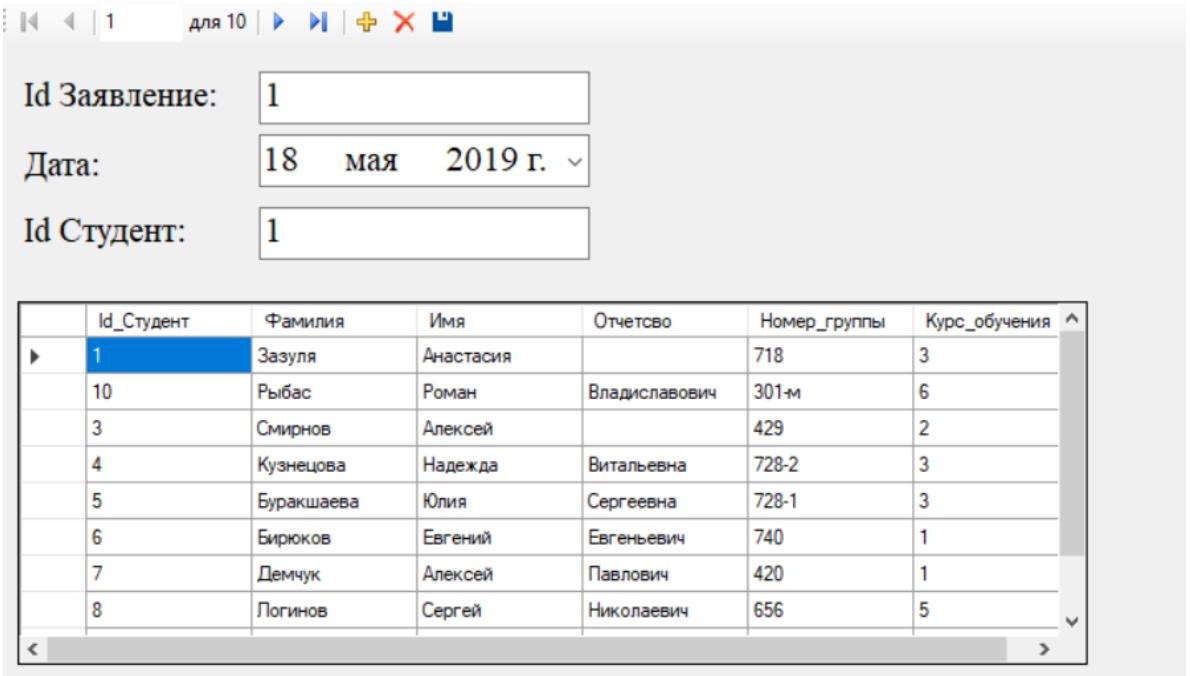
Рисунок 3.29 – Таблица «Распоряжение»

Перейдя в раздел «Комнаты» перед администратором появляется информация о номере комнаты, в каком общежитие находится, этаже и сколько студентов там проживают (рисунок 3.30).

	Id_Комната	Id_Общежитие	Сколько_прожива	Этаж	Номер_комнаты
▶	1	1	4	6	20
	2	2	3	4	38
	3	3	2	2	2
	4	1	4	9	15
	5	2	4	5	10
	6	3	3	8	40
	7	1	2	7	4
*					

Рисунок 3.30 – Таблица «Комнаты»

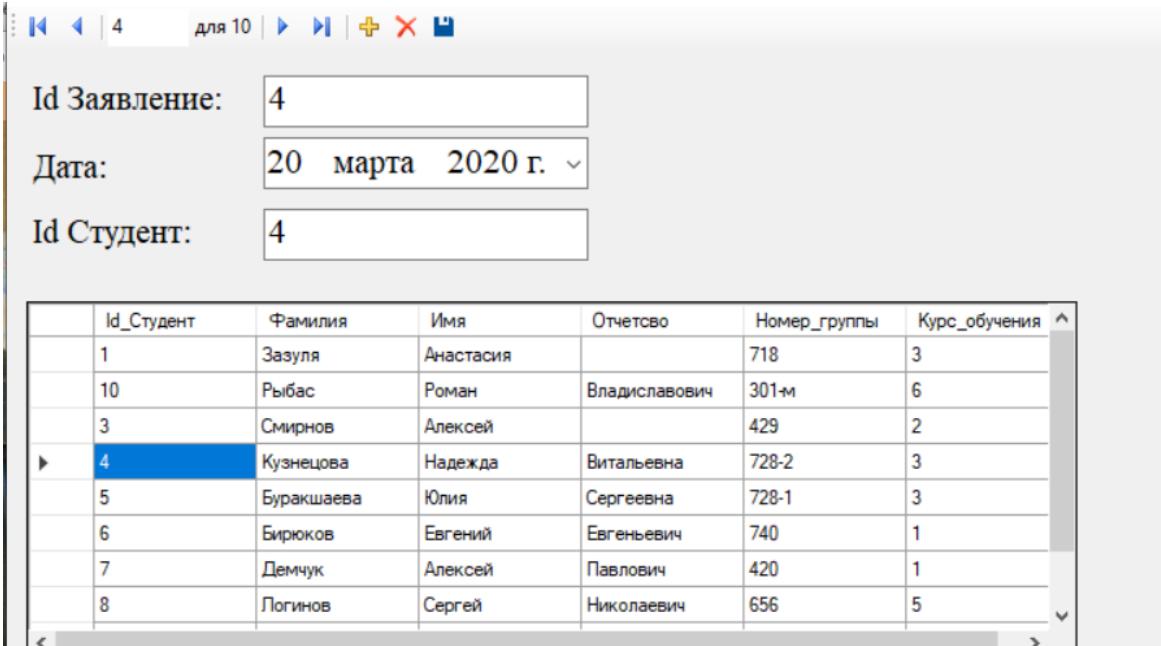
Перейдя в раздел «Заявление» перед администратором появляется информация о номере заявления, когда было написано заявление и информация о студенте в соответствие с номером заявления (рисунок 3.31 – 3.32).



The screenshot shows a software interface for managing requests. At the top, there are navigation buttons (back, forward, search, etc.) and a status bar indicating 'для 10'. Below this, three input fields are displayed: 'Id Заявление:' with value '1', 'Дата:' with value '18 мая 2019 г.', and 'Id Студент:' with value '1'. Below these fields is a table with the following data:

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
▶	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	3	Смирнов	Алексей		429	2
	4	Кузнецова	Надежда	Витальевна	728-2	3
	5	Буракшаева	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	8	Логинов	Сергей	Николаевич	656	5

Рисунок 3.31 – Таблица «Заявление» (1)



The screenshot shows the same software interface for managing requests. The top section remains the same with navigation buttons, a status bar 'для 10', and input fields for 'Id Заявление:' (4), 'Дата:' (20 марта 2020 г.), and 'Id Студент:' (4). Below the input fields is a table with the following data:

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
▶	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	3	Смирнов	Алексей		429	2
	4	Кузнецова	Надежда	Витальевна	728-2	3
	5	Буракшаева	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	8	Логинов	Сергей	Николаевич	656	5

Рисунок 3.32 – Таблица «Заявление» (2)

Перейдя в раздел «Данные общежития» перед администратором появляется информация о номере общежития и адрес (рисунок 3.33), так же можно добавить (рисунок 3.34 – 3.35), сохранить (рисунок 3.36) или удалить данные (рисунок 3.37).

	Id_Общежитие	Адрес
▶	1	Ф.Лыткина 18
	2	Ф.Лыткина 8
	3	Ф.Лыткина 10
*		

Рисунок 3.33 – Таблица «Данные общежития»

	Id_Общежитие	Адрес	Добавить
▶	1	Ф.Лыткина 18	
	2	Ф.Лыткина 8	
	3	Ф.Лыткина 10	
*			

Рисунок 3.34 – Добавление данных в таблицу «Данные общежития»

	Id_Общежитие	Адрес
	1	Ф.Лыткина 18
	2	Ф.Лыткина 8
	3	Ф.Лыткина 10
...	4	Ленина 122
*		

Рисунок 3.35 – Добавленные данные в таблицу «Данные общежития»

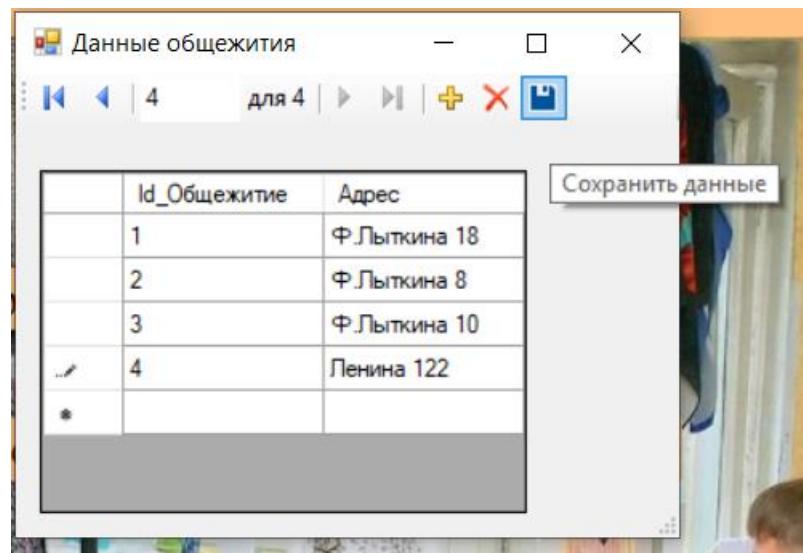


Рисунок 3.36 – Сохранение данных в таблице «Данные общежития»

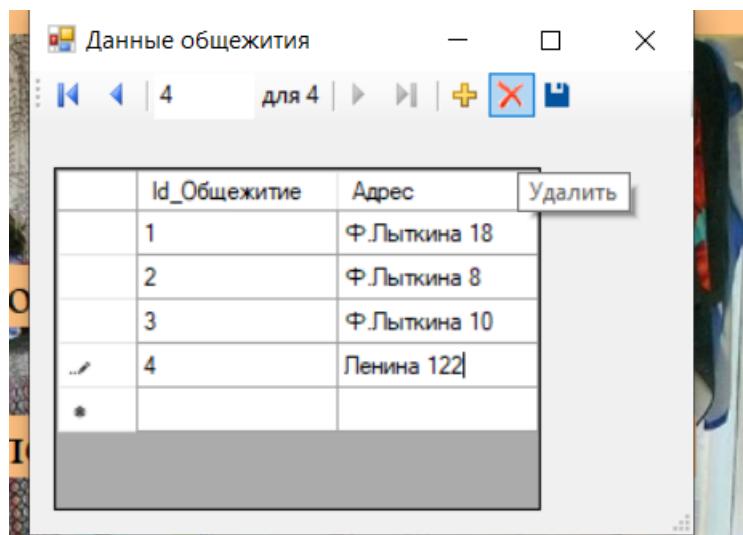


Рисунок 3.37 – Удаление данных в таблице «Данные общежития»

### 3.2 Учетная запись «Раселитель»

При запуске программы появляется окно входа, куда нужно ввести данные Раселителя (рисунок 3.38).

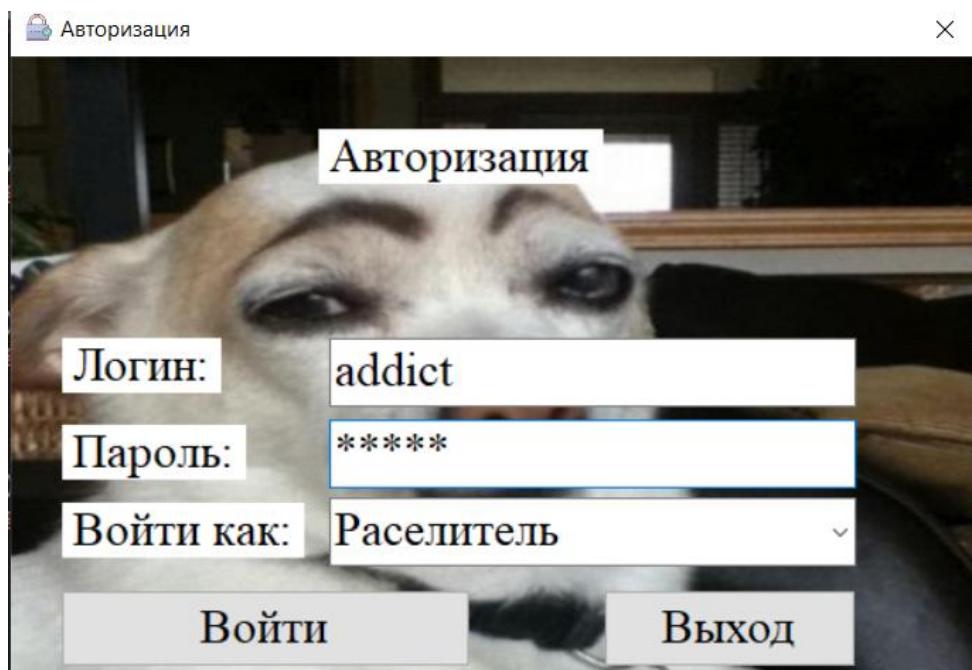


Рисунок 3.38 – Окно входа

После удачного входа пользователь попадает на главное меню программы (рисунок 3.39).

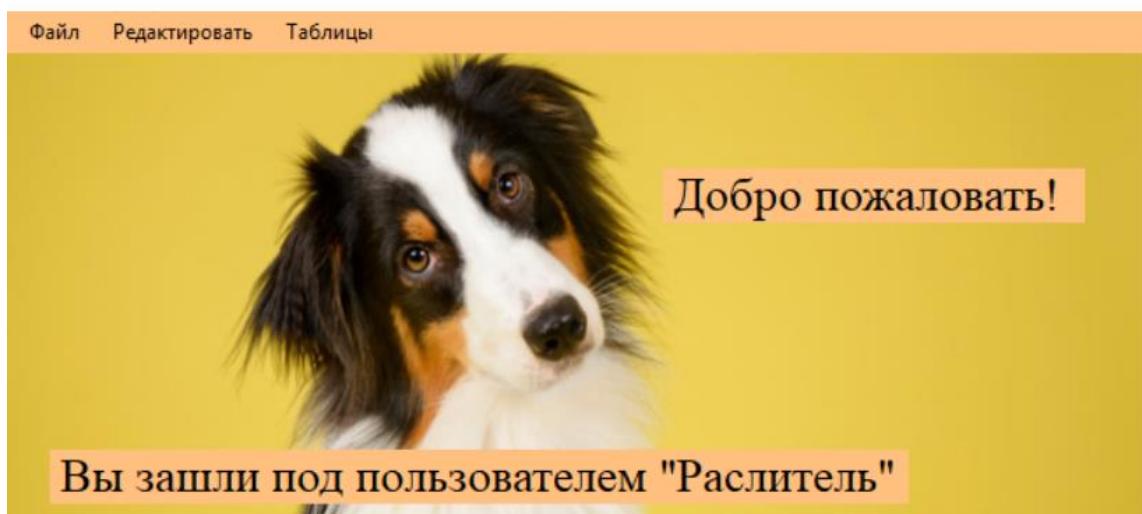


Рисунок 3.39 – Главное меню

Вверху окна находятся навигационное меню, если нажать на кнопку «Файл» (рисунок 3.40), то перед раселителем появится раздел «О программе» и «Выйти»

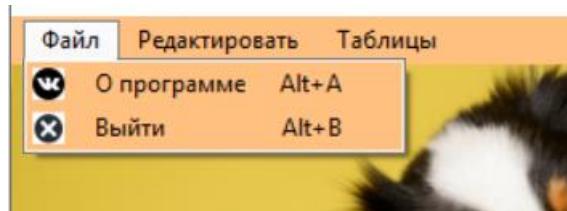


Рисунок 3.40 – Навигационное меню

Нажав на раздел «О программе» выйдет диалоговое окно, где будут написаны данные о разработчике (рисунок 3.41).

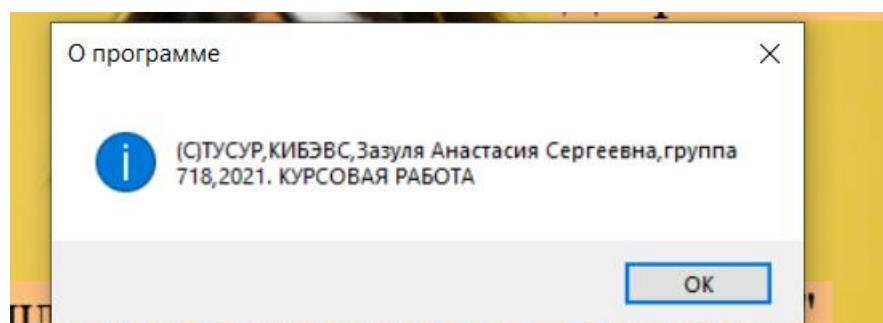


Рисунок 3.41 – Описание программы

Нажав на раздел «Выход» вылезет диалоговое окно, которое просит пользователя подтвердить выход из программы (рисунок 3.42).

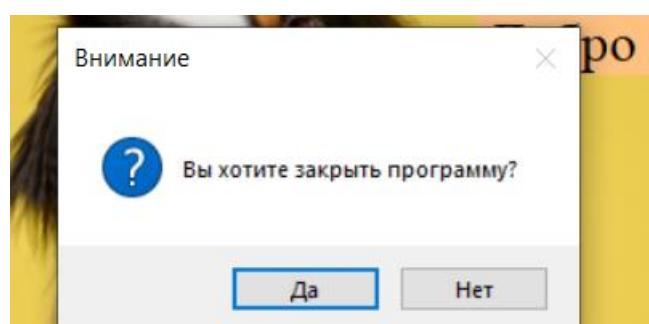


Рисунок 3.42 – Выйти

Если нажать на кнопку «Редактировать», перед раселителем появятся 3 раздела «Запросы по данным», «Изменение данных» и «Операторы» (рисунок 3.43).

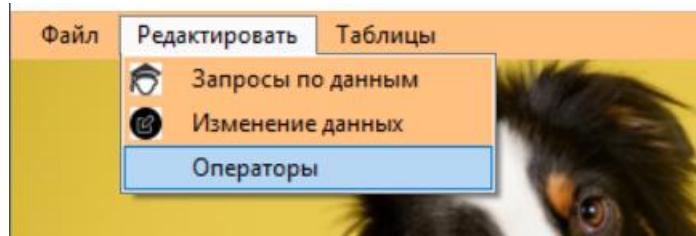


Рисунок 3.43 – «Редактировать»

Перейдя в раздел «Запросы по данным» на экране у раселителя появится окно с подразделами «Информация по таблицам» и «Заселенные студенты».

Нажав на пометку «Студент» перед пользователем появится информация о номере студента, ФИО, номер группы и курс обучения (рисунок 3.44).

	Id_Студен	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
▶	1	Зазуля	Анастас...		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	3	Смирнов	Алексей		429	2
	4	Кузнецо...	Надежда	Витальевна	728-2	3
	5	Буракша...	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	8	Логинов	Сергей	Николаевич	656	5
*	9	Романова	Татьяна	Станиславовна	327	4

Рисунок 3.44 – Информация о студентах

Нажав на пометку «Распоряжения» перед раселителем появится информация о номере распоряжения, номер заявления, дата, когда было подано заявление, дата заселения и дата выселения (рисунок 3.45)

Информация по таблицам   Заселенные студенты					
	Id_Распор	Id_Заявле	Дата	Дата_заселения	Дата_выселения
▶	5	5	14.03.2020	14.10.2020	31.08.2020
	4	4	03.04.2020	20.03.2020	31.08.2020
	2	2	23.04.2020	25.08.2020	31.08.2020
	6	6	23.07.2020	11.05.2020	31.08.2020
	9	9	21.10.2020	07.08.2020	31.08.2021
	7	7	30.12.2020	22.12.2020	31.08.2021
	3	3	02.03.2021	28.01.2021	31.08.2021
	1	1	18.05.2021	15.05.2019	31.08.2019
	10	10	15.06.2021	11.01.2021	31.08.2021
	8	8	02.08.2021	15.01.2021	31.08.2021
*					

Рисунок 3.45 – Информация о распоряжениях

Нажав на пометку «Комната» перед раселителем появится информация о номере комнаты, сколько студентов проживают, этаж, номер комнаты, номер студента адрес общежития и заселен студент на данный момент или нет (рисунок 3.46).

Информация по таблицам   Заселенные студенты						
	Id_Комнат	Сколько_1	Этаж	Номер_ко	Id_Студен	Адрес
▶	3	2	2	2	3	Ф.Лыткин...
	3	2	2	2	7	Ф.Лыткин...
	7	2	7	4	6	Ф.Лыткин...
	7	2	7	4	8	Ф.Лыткин...
	5	4	5	10	10	Ф.Лыткин...
	1	4	6	20	1	Ф.Лыткин...
	2	3	4	38	2	Ф.Лыткин...
	2	3	4	38	4	Ф.Лыткин...
	2	3	4	38	5	Ф.Лыткин...
	6	3	8	40	9	Ф.Лыткин...
*						

Рисунок 3.46 – Информация о комнатах

Перейдя на подраздел «Заселенный студент» перед администратором появится окно, в котором нужно вписать фамилию студента и нажать кнопу базе заселен он или нет.

Для примера введем в текстовое окно фамилию студента и нажать кнопу «Найти» (рисунок 3.47 – 3.48).

	Id_Студен	Фамилия	Имя	Отчество	Номер_гр	Заселенны
▶	3	Смирнов	Алексей		429	<input checked="" type="checkbox"/>
*						<input type="checkbox"/>

Рисунок 3.47 – Результат запроса (1)

	Id_Студен	Фамилия	Имя	Отчество	Номер_гр	Заселенны
*						<input type="checkbox"/>

Рисунок 3.48 – Результат запроса с выводом ошибки

Перейдя в раздел «Подзапросы по данным» перед раселителем появляется два вида запросы:

1. Коррелированный подзапрос – внутренний запрос для своего выполнения должен получить данные из внешнего запроса. В коррелированном подзапросе внутренний запрос не может быть реализован немедленно: он ссылается на внешний запрос и выполняется поочередно для каждой строки во внешнем запросе (рисунок 3.49);

2. Некоррелированный подзапрос – внутренний запрос выполняется независимо, передавая результаты во внешний запрос (рисунок 3.50).

	Номер_комнаты	Этаж	Сколько_прожива
▶	38	4	3
	40	8	3
*			

Рисунок 3.49 – «Коррелированный подзапрос»

	Номер_комнаты	Этаж	Сколько_прожива
▶	20	6	4
*			

Рисунок 3.50 – «Некоррелированный подзапрос»

Перейдя в раздел «Операторы» (рисунок 3.51) перед администратором появится окно, в котором можно добавить (рисунок 3.52), изменить (рисунок 3.53) или удалить данные (рисунок 3.54).

Оператор – это зарезервированное слово, или символ, который используется в SQL выражениях с использованием WHERE для выполнения операции или операций, например, сравнение.

Операторы

Добавить данные (ID не вводить при добавлении)     Изменить Данные     Удалить Данные

ID Студента	<input type="text"/>	<input type="button" value="Выполнить запрос"/>	
Фамилия	<input type="text" value="Пушкин"/>	Группа студента	<input type="text" value="123"/>
Имя	<input type="text" value="Станислав"/>	Курс обучения	<input type="text" value="4"/>
Отчество	<input type="text"/>	<input type="button" value="Показать список"/>	

Рисунок 3.51 – «Операторы»

Операторы

Добавить данные (ID не вводить при добавлении)     Изменить Данные     Удалить Данные

ID Студента	<input type="text"/>	<input type="button" value="Выполнить запрос"/>	
Фамилия	<input type="text" value="Пушкин"/>	Группа студента	<input type="text" value="123"/>
Имя	<input type="text" value="Станислав"/>	Курс обучения	<input type="text" value="4"/>
Отчество	<input type="text"/>	<input type="button" value="Показать список"/>	

	Id_Студент	Фамилия	Имя	Номер_группы	Column1
1	Зазуля	Анастасия	718	2	
10	Рыбас	Роман	301-м	2	
3	Смирнов	Алексей	429	2	
4	Кузнецова	Надежда	728-2	2	
5	Буракшаева	Юлия	728-1	2	
6	Бирюков	Евгений	740	2	
7	Демчук	Алексей	420	2	
9	Романова	Татьяна	327	2	
11	Пушкин	Станислав	123	4	

Рисунок 3.52 – Добавление данных с помощью оператора

Операторы

Добавить данные (ID не вводить при добавлении)       Изменить Данные       Удалить Данные

ID Студента	11	Выполнить запрос	
Фамилия	Пупкина	Группа студента	123
Имя	Станиславна	Курс обучения	1
Отчество		Показать список	

Id_Студент	Фамилия	Имя	Номер_группы	Column1
1	Зазуля	Анастасия	718	2
10	Рыбас	Роман	301-м	2
3	Смирнов	Алексей	429	2
4	Кузнецова	Надежда	728-2	2
5	Буракшаева	Юлия	728-1	2
6	Бирюков	Евгений	740	2
7	Демчук	Алексей	420	2
9	Романова	Татьяна	327	2
11	Пупкина	Станиславна	123	1

Рисунок 3.53 – Изменение данных с помощью оператора

Операторы

Добавить данные (ID не вводить при добавлении)       Изменить Данные       Удалить Данные

ID Студента	8	Выполнить запрос	
Фамилия		Группа студента	
Имя		Курс обучения	
Отчество		Показать список	

Id_Студент	Фамилия	Имя	Номер_группы	Column1
1	Зазуля	Анастасия	718	2
10	Рыбас	Роман	301-м	2
3	Смирнов	Алексей	429	2
4	Кузнецова	Надежда	728-2	2
5	Буракшаева	Юлия	728-1	2
6	Бирюков	Евгений	740	2
7	Демчук	Алексей	420	2
9	Романова	Татьяна	327	2
11	Пупкина	Станиславна	123	1

Рисунок 3.54 – Удаление данных с помощью оператора

Перейдя в раздел «Таблицы» (рисунок 3.55) перед раселителем появляется таблица с информацией о студентах (номер, ФИО, номер группы и курс обучения), для удобства каждый курс обучения был присвоен свой цвет (рисунок 3.56).

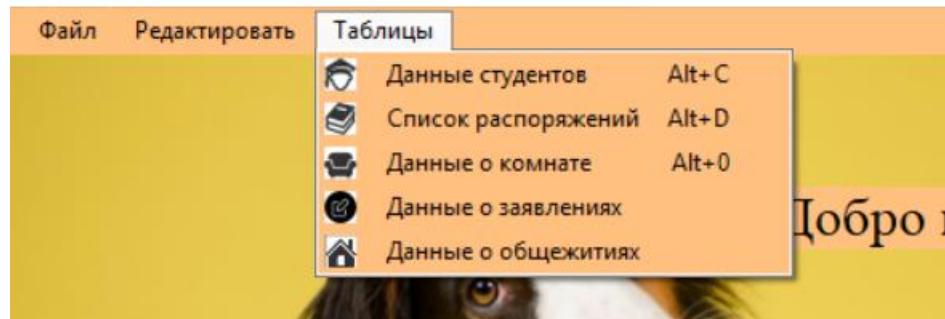


Рисунок 3.55 – Раздел «Таблицы»

Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
1	Зазуля	Анастасия		718	3
10	Рыбас	Роман	Владиславович	301-м	6
3	Смирнов	Алексей		429	2
4	Кузнецова	Надежда	Витальевна	728-2	3
5	Буракшаева	Юлия	Сергеевна	728-1	3
6	Бирюков	Евгений	Евгеньевич	740	1
7	Демчук	Алексей	Павлович	420	1
9	Романова	Татьяна	Станиславовна	327	4
11	Пупкина	Станиславна		123	1

Рисунок 3.56 – Таблица «Студенты»

В разделе «Студенты» присутствуют такие функции как:

1. «Поиск» – поиск производится на основе вводимых данных администратором для быстрого поиска информации (рисунок 3.57).
2. Фильтр – фильтрует информацию.

Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
1	Зазуля	Анастасия		718	3
10	Рыбас	Роман	Владиславович	301-м	6
3	Смирнов	Алексей		429	2
4	Кузнецова	Надежда	Витальевна	728-2	3
5	Буракшаева	Юлия	Сергеевна	728-1	3
6	Бирюков	Евгений	Евгеньевич	740	1
7	Демчук	Алексей	Павлович	420	1
9	Романова	Татьяна	Станиславовна	327	4
► 11	Пупкина	Станиславна		123	1

Рисунок 3.57 – «Поиск»

Перейдя в раздел «Распоряжение» перед раселителем появляется информация о номере распоряжения и заявления, когда было написано заявление, дата заселения и дата выселения студента (рисунок 3.58).

	Id_Распоряжение	Id_Заявление	Дата	Дата_заселения	Дата_выселения
►	1	1	18.05.2021	15.05.2019	31.08.2019
	2	2	23.04.2020	25.08.2020	31.08.2020
	3	3	02.03.2021	28.01.2021	31.08.2021
	4	4	03.04.2020	20.03.2020	31.08.2020
	5	5	14.03.2020	14.10.2020	31.08.2020
	6	6	23.07.2020	11.05.2020	31.08.2020
	7	7	30.12.2020	22.12.2020	31.08.2021
	8	8	02.08.2021	15.01.2021	31.08.2021
	9	9	21.10.2020	07.08.2020	31.08.2021
	10	10	15.06.2021	11.01.2021	31.08.2021
*					

Рисунок 3.58 – Таблица «Распоряжение»

Перейдя в раздел «Комнаты» перед раселителем появляется информация о номере комнаты, в каком общежитие находится, этаже и сколько студентов там проживают (рисунок 3.59).

The screenshot shows a table titled 'Комнаты' (Rooms) with the following columns: Id\_Комната (Room ID), Id\_Общежитие (Building ID), Сколько\_прожива (Number of residents), Этаж (Floor), and Номер\_комнаты (Room number). The table contains 8 rows of data. Row 4 is highlighted with a blue background.

	Id_Комната	Id_Общежитие	Сколько_прожива	Этаж	Номер_комнаты
1	1	4	6	20	
2	2	3	4	38	
3	3	2	2	2	
▶ 4	1	4	9	15	
5	2	4	5	10	
6	3	3	8	40	
7	1	2	7	4	

Рисунок 3.59 – Таблица «Комнаты»

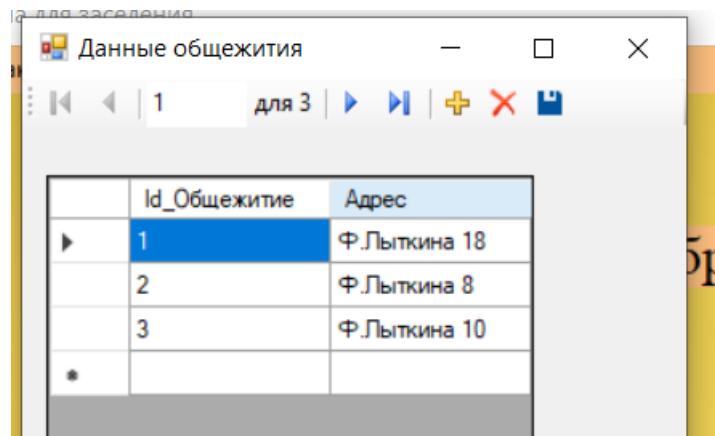
Перейдя в раздел «Заявление» перед администратором появляется информация о номере заявления, когда было написано заявление и информация о студенте в соответствие с номером заявления (рисунок 3.60).

The screenshot shows a window titled 'Заявления' (Applications) with three input fields: Id Заявление (Application ID) set to 3, Дата (Date) set to 28 января 2021 г., and Id Студент (Student ID) set to 3. Below these fields is a table titled 'Студенты' (Students) with the following columns: Id\_Студент (Student ID), Фамилия (Surname), Имя (Name), Отчество (Middle name), Номер\_группы (Group number), and Курс\_обучения (Course). The table contains 8 rows of data. Row 3 is highlighted with a blue background.

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
1	Зазуля	Анастасия			718	3
10	Рыбас	Роман	Владиславович	301-м	6	
11	Пупкина	Станиславна			123	1
▶ 3	Смирнов	Алексей			429	2
4	Кузнецова	Надежда	Витальевна	728-2	3	
5	Буракшаева	Юлия	Сергеевна	728-1	3	
6	Бирюков	Евгений	Евгеньевич	740	1	
7	Демчук	Алексей	Павлович	420	1	

Рисунок 3.60 – Таблица «Заявление»

Перейдя в раздел «Данные общежития» перед администратором появляется информация о номере общежития и адрес (рисунок 3.61), так же можно добавить, сохранить или удалить данные.



The screenshot shows a Windows application window titled "Данные общежития". The window contains a table with three columns: "Id\_Общежитие" (ID of the dormitory) and "Адрес" (Address). The table has four rows, indexed 1 to 3. Row 1 is selected, showing the address "Ф.Лыткина 18". Row 2 shows "Ф.Лыткина 8" and Row 3 shows "Ф.Лыткина 10". A fourth row is present but empty. Above the table, there is a toolbar with icons for navigating between records and performing operations like adding (+), deleting (-), and saving (checkmark).

	Id_Общежитие	Адрес
▶	1	Ф.Лыткина 18
	2	Ф.Лыткина 8
	3	Ф.Лыткина 10
*		

Рисунок 3.61 – Таблица «Данные общежития»

### 3.3 Учетная запись «Комендант»

При запуске программы появляется окно входа, куда нужно ввести данные коменданта (рисунок 3.62). Нужно заметить, что у пользователя «Комендант» ограниченный допуск к информации (смотреть пункт 1.5.2).

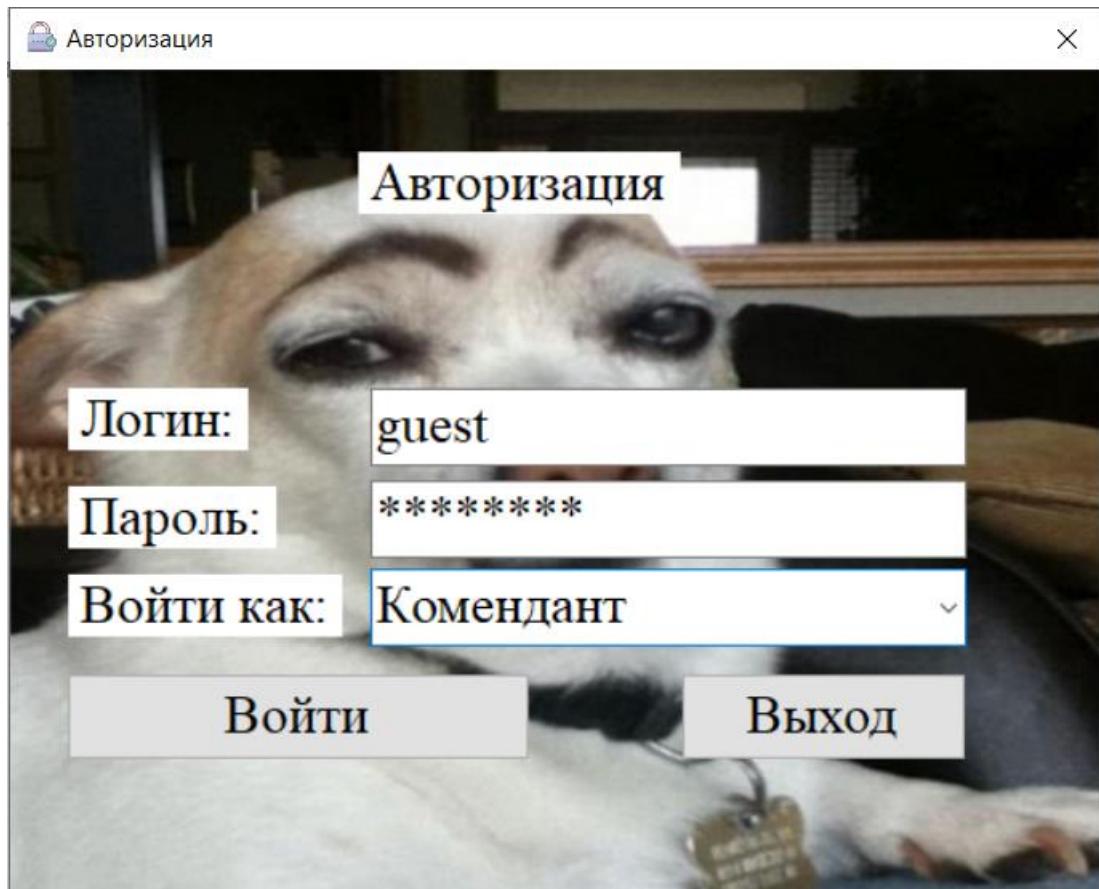


Рисунок 3.62 – Окно входа

После удачного входа комендант попадает на главное меню программы (рисунок 3.63).

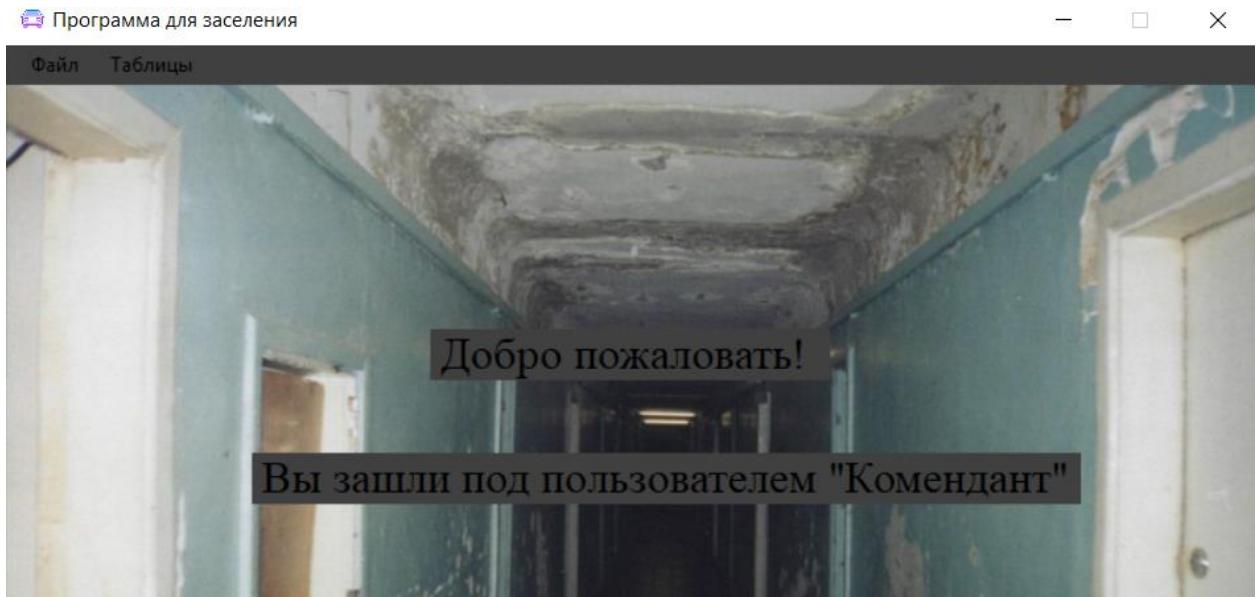


Рисунок 3.63 – Главное меню

Вверху окна находятся навигационное меню, если нажать на кнопку «Файл» (рисунок 3.64), то перед комендантом появится раздел «О программе» и «Выйти».

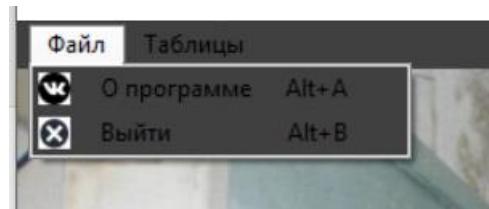


Рисунок 3.64 – Навигационное меню

Нажав на раздел «О программе» выйдет диалоговое окно, где будут написаны данные о разработчике (рисунок 3.65)

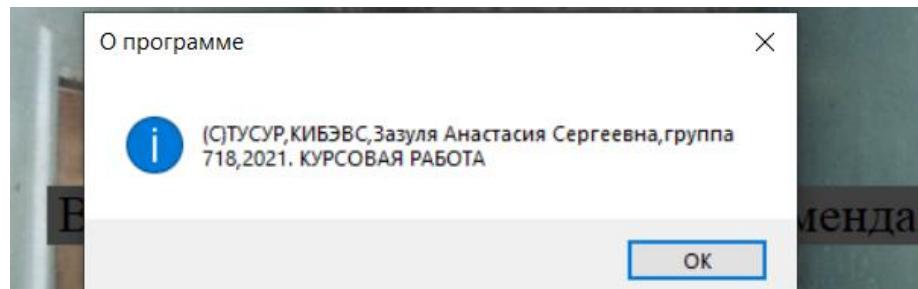


Рисунок 3.65 – Описание программы

Нажав на раздел «Выход» вылезет диалоговое окно, которое просит пользователя подтвердить выход из программы (рисунок 3.66).

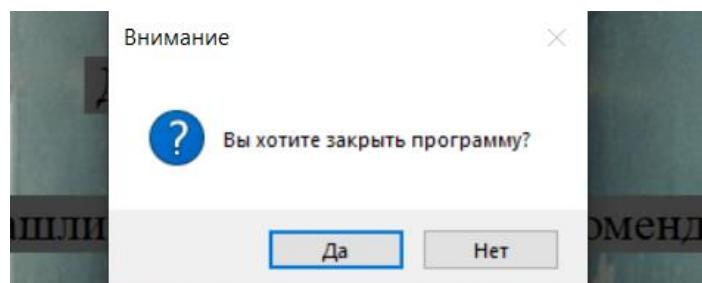


Рисунок 3.66 – Выйти

Перейдя в раздел «Таблицы» (рисунок 3.67) перед комендантом появляется таблица с информацией о студентах (номер, ФИО, номер группы и курс обучения), для удобства каждый курс обучения был присвоен свой цвет (рисунок 3.68).

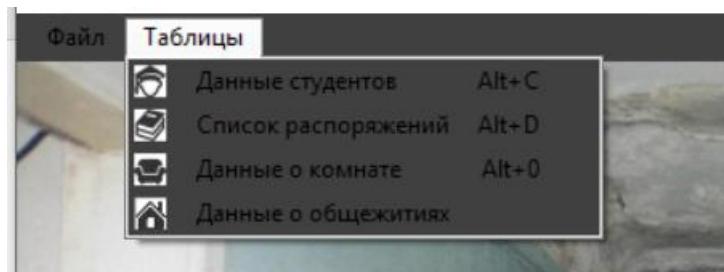


Рисунок 3.67 – Раздел «Таблицы»

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
▶	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	11	Пупкина	Станиславна		123	1
	3	Смирнов	Алексей		429	2
	4	Кузнецова	Надежда	Витальевна	728-2	3
	5	Буракшаева	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
	7	Демчук	Алексей	Павлович	420	1
	9	Романова	Татьяна	Станиславовна	327	4
*						

Рисунок 3.68 – Таблица «Студенты»

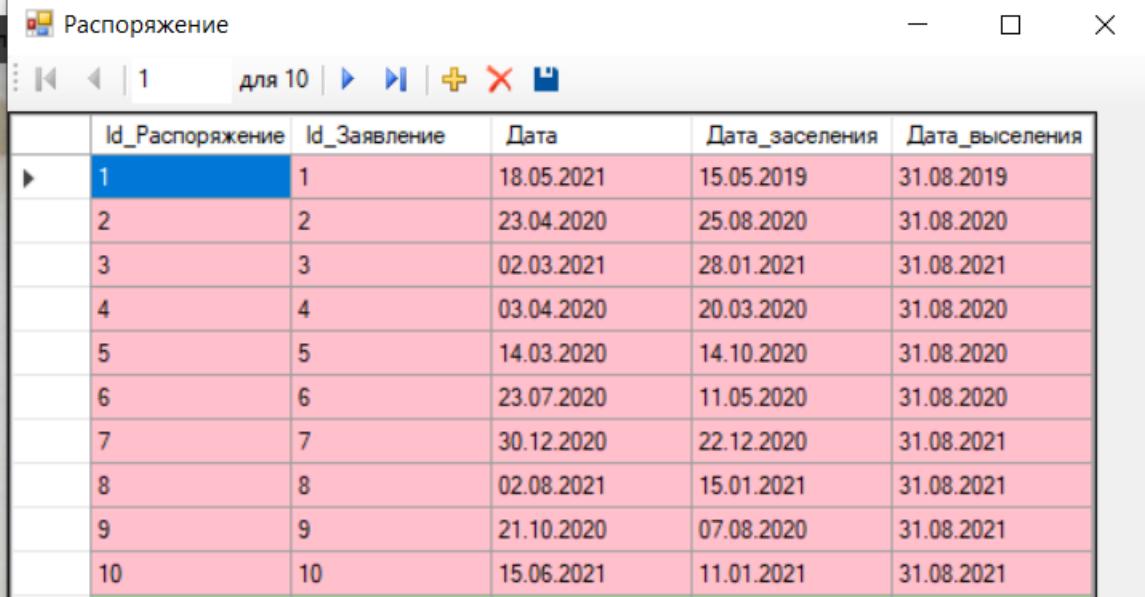
В разделе «Студенты» присутствуют такие функции как:

1. «Поиск» – поиск производится на основе введенных данных администратором для быстрого поиска информации (рисунок 3.69).
2. Фильтр – фильтрует информацию.

	Id_Студент	Фамилия	Имя	Отчество	Номер_группы	Курс_обучения
▶	1	Зазуля	Анастасия		718	3
	10	Рыбас	Роман	Владиславович	301-м	6
	11	Пупкина	Станиславна		123	1
	3	Смирнов	Алексей		429	2
	4	Кузнецова	Надежда	Витальевна	728-2	3
	5	Буракшаева	Юлия	Сергеевна	728-1	3
	6	Бирюков	Евгений	Евгеньевич	740	1
▶	7	Демчук	Алексей	Павлович	420	1
	9	Романова	Татьяна	Станиславовна	327	4
*						

Рисунок 3.69 – «Поиск»

Перейдя в раздел «Распоряжение» перед администратором появляется информация о номере распоряжения и заявления, когда было написано заявление, дата заселения и дата выселения студента (рисунок 3.70).

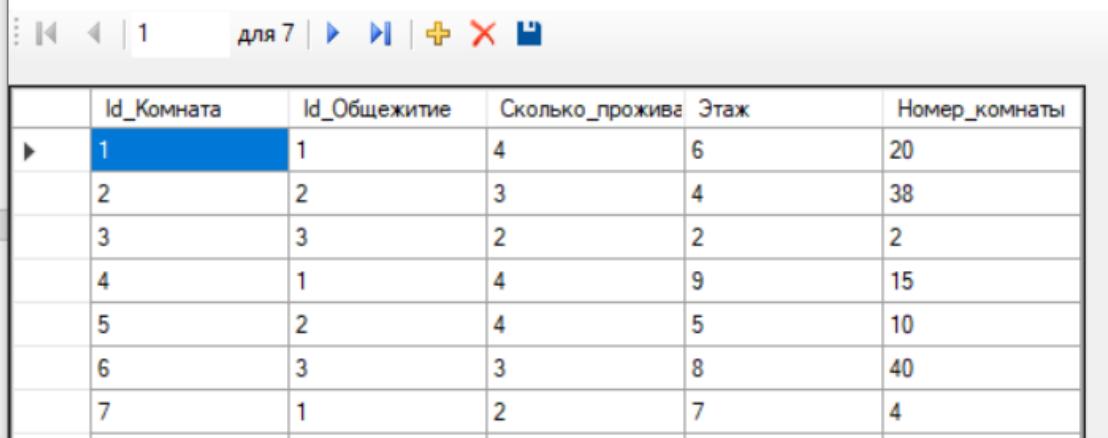


The screenshot shows a Windows application window titled "Распоряжение". The window contains a table with the following columns: Id\_Распоряжение, Id\_Заявление, Дата, Дата\_заселения, and Дата\_выселения. The table has 10 rows, each representing a record from 1 to 10. The first row (Id\_Распоряжение 1) is highlighted in blue, indicating it is selected.

	Id_Распоряжение	Id_Заявление	Дата	Дата_заселения	Дата_выселения
▶	1	1	18.05.2021	15.05.2019	31.08.2019
	2	2	23.04.2020	25.08.2020	31.08.2020
	3	3	02.03.2021	28.01.2021	31.08.2021
	4	4	03.04.2020	20.03.2020	31.08.2020
	5	5	14.03.2020	14.10.2020	31.08.2020
	6	6	23.07.2020	11.05.2020	31.08.2020
	7	7	30.12.2020	22.12.2020	31.08.2021
	8	8	02.08.2021	15.01.2021	31.08.2021
	9	9	21.10.2020	07.08.2020	31.08.2021
	10	10	15.06.2021	11.01.2021	31.08.2021

Рисунок 3.70 – Таблица «Распоряжение»

Перейдя в раздел «Комнаты» перед комендантом появляется информация о номере комнаты, в каком общежитие находится, этаже и сколько студентов там проживают (рисунок 3.71).

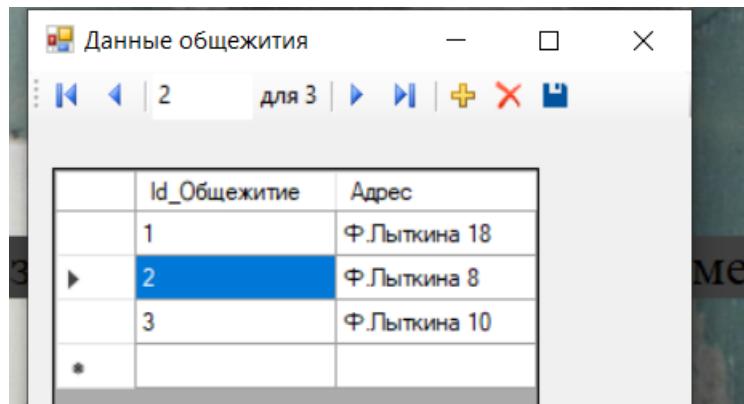


The screenshot shows a Windows application window titled "Комнаты". The window contains a table with the following columns: Id\_Комната, Id\_Общежитие, Сколько\_прожива, Этаж, and Номер\_комнаты. The table has 7 rows, each representing a room record from 1 to 7. The first row (Id\_Комната 1) is highlighted in blue, indicating it is selected.

	Id_Комната	Id_Общежитие	Сколько_прожива	Этаж	Номер_комнаты
▶	1	1	4	6	20
	2	2	3	4	38
	3	3	2	2	2
	4	1	4	9	15
	5	2	4	5	10
	6	3	3	8	40
	7	1	2	7	4

Рисунок 3.71 – Таблица «Комнаты»

Перейдя в раздел «Данные общежития» перед комендантом появляется информация о номере общежития и адрес (рисунок 3.72).



The screenshot shows a Windows application window titled "Данные общежития". The window has a standard title bar with minimize, maximize, and close buttons. Below the title bar is a toolbar with several icons: back, forward, a number "2", a text field containing "для 3", another number "3", a plus sign, a minus sign, a red X, and a blue square. The main area of the window contains a table with three columns and four rows. The columns are labeled "Id\_Общежитие" and "Адрес". The rows contain the following data:

	Id_Общежитие	Адрес
	1	Ф.Лыткина 18
▶	2	Ф.Лыткина 8
	3	Ф.Лыткина 10
*		

Рисунок 3.72 – Таблица «Данные общежития»

#### 4 Заключение

В ходе выполнения курсовой работы была создана база данных «Заселение в общежитие».

Выполнены все требуемые условия, так же реализована база данных SQL Server для авторизации пользователя. Так же был использован метод SHA-1 для хеширования паролей. Программа работает корректно, без сбоев.

В итоге программа содержит 10 форм.

«Авторизация» – проверка паролей в базе данных. Если данные верны, то программа, ссылаясь на роль переводит в нужную форму/

«Меню» – интерфейс программы.

«Запросы по таблицам»:

1. «Информация по таблицам» – данном разделе можно узнать информацию о студентах, распоряжениях и комнатах.

2. «Заселенные студенты» - данном разделе можно ввести фамилию студента и узнать, есть он в списках заселенных студентов или нет.

«Запросы по данным»:

1. «Коррелированный подзапрос» - это оператор SELECT, вложенный в другой оператор, и ссылающийся на один или несколько столбцов внешнего запроса.

2. «Некоррелированный подзапрос» - данный подзапрос независим от контекста.

«Операторы» - данном разделе можно добавить данные студента, изменить данные или удалить.

«Студенты» - данном разделе можно посмотреть ФИО, номер группы и курс обучения, которые будут подсвечиваться разными цветами для удобства пользователя.

«Распоряжения» - данном разделе можно посмотреть номер распоряжения, номер заявления, дату, дату заселения и дату выселения.

«Комнаты» - данном разделе можно посмотреть на каком этаже находится комната, номер комнаты и в каком общежитие находится комната.

«Заявление» - данном разделе можно посмотреть номер заявления, дату, когда было написано заявление и какой студент поддал заявление на заселение.

«Данные общежития» - данном разделе можно посмотреть адреса общежитий.

Была освоена компетенция ОПК-4: способность понимать значение информации в развитии современного общества, применять информационные технологии для поиска и обработки информации.

Была освоена компетенция ПК-3: способность администрировать подсистемы информационной безопасности объекта защиты.

Получены навыки оформления отчета по курсовой работе, согласно ГОСТ ОС ТУСУР 01-2013.

## 5 Список используемых источников

1. Образовательный стандарт вуза ОС ТУСУР 01-2013. [Электронный ресурс]. – Режим доступа: [https://storage.tusur.ru/files/40668/rules\\_tech\\_012013.pdf](https://storage.tusur.ru/files/40668/rules_tech_012013.pdf) (дата обращения: 20.02.2021)
2. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила применения. [Электронный ресурс]. – Режим доступа: <http://cert.obninsk.ru/gost/282/282.html> (дата обращения: 20.02.2021)
3. Нормализация базы данных. [Электронный ресурс]. – Режим доступа: <https://sdo.tusur.ru/mod/book/view.php?id=49450> (дата обращения: 1.03.2021)
4. Учебно-методические указания для создания приложения и работы с базой данных [Электронный курс] – Режим доступа: [https://sdo.tusur.ru/pluginfile.php/273806/mod\\_resource/content](https://sdo.tusur.ru/pluginfile.php/273806/mod_resource/content) (дата обращения: 9.03.2021)
5. Руководство по языку C# [Электронный ресурс]. – Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/csharp/> (дата обращения: 9.03.2021)
6. Руководство по Microsoft SSMS [Электронный ресурс]. – Режим доступа: <https://docs.microsoft.com/ru-ru/sql/ssms/tutorials/tutorial-sql-servermanagement-studio?view=sql-server-2014> (дата обращения: 30.03.2021)
7. Sha-1 – Tutorial [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/SHA-1> (дата обращения: 4.04.2021)
8. Запросы SQL [Электронный ресурс]. – Режим доступа: <https://tproger.ru/translations/sql-recap/> (дата обращения: 14.05.2021)
9. Сброс Автоинкремента [Электронный ресурс]. – Режим доступа: <https://zalinux.ru/?p=2582> (дата обращения: 21.05.2021)

Приложение А  
(Справочное)

Листинг формы авторизации

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SQLite;
using System.IO;
using System.Security.Cryptography;

namespace KursavBSBD
{
    public partial class FormAuth : Form
    {
        public string password, password_hash, textbox_hash, textbox_role;

        private void button2_Click(object sender, EventArgs e)
        {
            Close();
        }

        private void FormAuth_FormClosing(object sender, FormClosingEventArgs e)
        {
            e.Cancel = MessageBox.Show("Вы хотите закрыть программу?", "Внимание", MessageBoxButtons.YesNo, MessageBoxIcon.Question) != DialogResult.Yes;
        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            textBox1.MaxLength = 24;
        }

        private void textBox2_TextChanged(object sender, EventArgs e)
        {
            textBox2.MaxLength = 32;
        }

        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            comboBox1.MaxLength = 20;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Class2 hash = new Class2();
            password_hash = hash.Hash(password);
            textbox_hash = hash.Hash(textBox2.Text);
            textbox_role = comboBox1.Text;
            SQLiteConnection conn = new SQLiteConnection("Data Source=account.db; Version = 3;");
            SQLiteDataAdapter sda = new SQLiteDataAdapter("Select Count (*) From users where Login = '" + textBox1.Text + "' and Password = '" + textbox_hash + "' and Role = '" + comboBox1.Text + "'", conn);
        }
    }
}

```

```

DataTable dt = new DataTable();
sda.Fill(dt);
if (dt.Rows[0][0].ToString() == "1")
{
    if (textbox_role == "Администратор")
    {
        this.Hide();
        var FormMain = new FormMain();
        FormMain.Closed += (s, args) => this.Close();
        FormMain.Show();
    }
    else if (textbox_role == "Раселитель")
    {
        this.Hide();
        var FormMain = new FormMain1();
        FormMain.Closed += (s, args) => this.Close();
        FormMain.Show();
    }
    else if (textbox_role == "Комендант")
    {
        this.Hide();
        var FormMain = new FormMain2();
        FormMain.Closed += (s, args) => this.Close();
        FormMain.Show();
    }
}
else
{
    MessageBox.Show("Неверный логин или пароль! Пожалуйста, повторите попытку." +
                    " Примечание: Возможно вы забыли выбрать роль пользователя !");
    textBox1.Text = "";
    textBox1.Clear();
    textBox2.Text = "";
    textBox2.Clear();
    comboBox1.Text = " ";
}
}

private void FormAuth_Load(object sender, EventArgs e)
{
    SQLiteConnection conn = new SQLiteConnection("Data Source=account.db; Version = 3;");
    try
    {
        conn.Open();
    }
    catch (SQLiteException ex)
    {
        MessageBox.Show((ex.Message));
    }
    if (conn.State == ConnectionState.Open)
    {
        SQLiteCommand cmd = conn.CreateCommand();
        string sql_command = "DROP TABLE IF EXISTS users;" +
            "CREATE TABLE users(" +
            "id INTEGER PRIMARY KEY AUTOINCREMENT," +
            "login TEXT(24), " +
            "password TEXT(32)," +
            "role TEXT(20));";
        cmd.CommandText = sql_command;
        try
        {

```

```
        cmd.ExecuteNonQuery();
    }
    catch (SQLiteException ex)
    {
        MessageBox.Show((ex.Message));
    }
    Class2 hash = new Class2();
    password = "1";
    password_hash = hash.Hash(password);
    string str = "insert into users(login,password,role) values ('1','" +
password_hash + "','Администратор')";
    cmd.CommandText = str;
    cmd.ExecuteNonQuery();
    password = "12345";
    password_hash = hash.Hash(password);
    string std = "insert into users(login,password,role) values ('addict','" +
+ password_hash + "','Раселитель')";
    cmd.CommandText = std;
    cmd.ExecuteNonQuery();
    password = "12345678";
    password_hash = hash.Hash(password);
    string stt = "insert into users(login,password,role) values ('guest','" +
password_hash + "','Комендант')";
    cmd.CommandText = stt;
    cmd.ExecuteNonQuery();
}
conn.Dispose();
}

public FormAuth()
{
    InitializeComponent();
}
}
```

Приложение Б  
(Справочное)

Листинг формы «Запросы по данным»

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace KursavBSBD
{
    public partial class ZaprosiOdanih : Form
    {
        public ZaprosiOdanih()
        {
            InitializeComponent();
        }

        private static ZaprosiOdanih i;
        public static ZaprosiOdanih inf
        {
            get
            {
                if (i == null || i.IsDisposed) i = new ZaprosiOdanih();
                return i;
            }
        }

        //объявляем метод, на вход подаем строку запроса, а возвращаем объект DataTable
        DataTable FillDataGridView(string sqlSelect)
        {
            //Создаем объект connection класса SqlConnection для соединения с Бд
            SqlConnection connection = new
            SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
            //Создаем объект command для SQL команды
            SqlCommand command = connection.CreateCommand();
            //Заносим текст SQL запроса через параметр sqlSelect
            command.CommandText = sqlSelect;
            //Создаем объект adapter класса SqlDataAdapter
            SqlDataAdapter adapter = new SqlDataAdapter();
            //Задаем адаптеру нужную команду, в данном случае команду Select
            adapter.SelectCommand = command;
            //Создаем объект table для последующего отображения результата запроса
            DataTable table = new DataTable();
            //заполним набор данных результатом запроса
            adapter.Fill(table);
            return table;
        }

        private void radioButton6_CheckedChanged(object sender, EventArgs e)
        {
            dataGridViewSelect.DataSource = FillDataGridView("SELECT * FROM Студент ");
        }
    }
}

```

```
private void radioButton5_CheckedChanged(object sender, EventArgs e)
{
    dataGridViewSelect.DataSource = FillDataGridView("SELECT * FROM
Распоряжение");
}

private void radioButton4_CheckedChanged_1(object sender, EventArgs e)
{
    dataGridViewSelect.DataSource = FillDataGridView("SELECT Комната.Id_Комната,
Комната.Сколько_проживает, Комната.Этаж, Комната.Номер_комнаты,
Заселенный_студент.Id_Студент, Общежитие.Адрес, Заселенный_студент.Заселенный_студент FROM
Заселенный_студент INNER JOIN Комната ON Заселенный_студент.Id_Комната =
Комната.Id_Комната INNER JOIN Общежитие ON Комната.Id_Общежитие =
Общежитие.Id_Общежитие");
}

private void buttonF_select_Click(object sender, EventArgs e)
{
    if (String.IsNullOrEmpty(textBoxWorker.Text))
    {
        MessageBox.Show("Обязательно укажите ID необходимого студента.\nДопустим
ввод первых символов.", "Внимание", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
        return;
    }
    string sqlSelect = @"SELECT Студент.Id_Студент, Студент.Фамилия, Студент.Имя,
Студент.Отчество, Студент.Номер_группы, Заселенный_студент.Заселенный_студент FROM
Студент, Заселенный_студент WHERE Студент.Id_Студент = Заселенный_студент.Id_Студент AND
Студент.Фамилия LIKE @Фамилия";
    /*SELECT PK_Студент.PK_Студент, PK_Студент.Фамилия, PK_Студент.Имя,
PK_Студент.Отчество, PK_Студент.Номер_группы, Заселенный_студент.Заселенный_студент FROM
PK_Студент, Заселенный_студент WHERE (PK_Студент.PK_Студент LIKE @ID);*/
    /*SELECT * FROM PK_Студент WHERE PK_Студент.PK_Студент LIKE @ID*/
    SqlConnection connection = new
SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
    SqlCommand command = connection.CreateCommand();
    command.CommandText = sqlSelect;
    command.Parameters.Add("@Фамилия", SqlDbType.NChar);
    command.Parameters["@Фамилия"].Value = textBoxWorker.Text + "%";
    SqlDataAdapter adapter = new SqlDataAdapter();
    adapter.SelectCommand = command;
    DataTable table = new DataTable();
    adapter.Fill(table);
    dataGridViewSelect2.DataSource = table;
    if (table.Rows.Count == 0) MessageBox.Show("Нет значений!",
"Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
}
```

Приложение В  
(Справочное)

Листинг формы «Подзапросы по данным»

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace KursavBSBD
{
    public partial class Ismenenie : Form
    {
        public Ismenenie()
        {
            InitializeComponent();
        }

        DataTable FillDataGridView(string sqlSelect)
        {
            SqlConnection connection = new
SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
            SqlCommand command = connection.CreateCommand();
            command.CommandText = sqlSelect;
            SqlDataAdapter adapter = new SqlDataAdapter();
            adapter.SelectCommand = command;
            DataTable table = new DataTable();
            adapter.Fill(table);
            return table;
        }

        private void groupBoxSubquery_Enter(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(textBoxText.Text))
            {
                MessageBox.Show("Введите необходимые данные",
                "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }
            string sqlSelect = "";
            if (radioButtonCorrelated.Checked)
                sqlSelect = @"SELECT Комната.Номер_комнаты, Комната.Этаж,
Комната.Сколько_ проживает FROM Комната WHERE Сколько_ проживает IN (SELECT
Сколько_ проживает FROM Комната WHERE Сколько_ проживает LIKE @ID) ";
            else
                if (radioButtonNoCorrelated.Checked)

```

```
sqlSelect = @"SELECT Комната.Номер_комнаты, Комната.Этаж,
Комната.Сколько_ проживает FROM Комната WHERE Сколько_ проживает > (SELECT
AVG(Сколько_ проживает) FROM Комната) AND Комната.Номер_комнаты LIKE @ID ";
    else
    {
        MessageBox.Show("Не выбран вид подзапроса", "Ошибка",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    SqlConnection connection = new
SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
    SqlCommand command = connection.CreateCommand();
    command.CommandText = sqlSelect;
    command.Parameters.Add("@ID", SqlDbType.NChar);
    command.Parameters["@ID"].Value = textBoxText.Text + "%";
    SqlDataAdapter adapter = new SqlDataAdapter();
    adapter.SelectCommand = command;
    DataTable table = new DataTable();
    adapter.Fill(table);
    dataGridView1.DataSource = table;
    if (table.Rows.Count == 0) MessageBox.Show("Нет таких значений!",
    "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void Ismenenie_Load(object sender, EventArgs e)
{
}

}

}
```

## Приложение Г

(Справочное)

## Листинг формы «Операторы»

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace KursavBSBD
{
    public partial class Operators : Form
    {
        public Operators()
        {
            InitializeComponent();
        }

        DataTable FillDataGridView(string sqlSelect)
        {
            SqlConnection connection = new
SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
            SqlCommand command = connection.CreateCommand();
            command.CommandText = sqlSelect;
            SqlDataAdapter adapter = new SqlDataAdapter();
            adapter.SelectCommand = command;
            DataTable table = new DataTable();
            adapter.Fill(table);
            return table;
        }

        private void buttonExecuteDML_Click(object sender, EventArgs e)
        {
            if (radioButtonAdd.Checked)
                AddStudent();
            else
                if (radioButtonUpdate.Checked)
                    UpdateStudent();
            else
                if (radioButtonDelete.Checked)
                    DeleteStudent();
            else
                MessageBox.Show("Вы не выбрали действие", "Внимание",
                MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }

        void AddStudent()
        {
            if (String.IsNullOrEmpty(textBoxFDML.Text) ||
(String.IsNullOrEmpty(textBoxNDML.Text) || (String.IsNullOrEmpty(textBoxODML.Text) ||
(String.IsNullOrEmpty(textBoxGroupDML.Text) ||
(String.IsNullOrEmpty(textBoxCourseDML.Text))))))
            {

```

```

        MessageBox.Show("Обязательно введите его ФИО, номер группы, курс и
комнату", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    int passnum = 0;
    if (!int.TryParse(textBoxCourseDML.Text, out passnum))
    {
        MessageBox.Show("Некоректные данные!", "Внимание",
        MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    string sqlInsert = @"INSERT INTO PK_Студент (Id_Студент,Фамилия, Имя,
Отчество, Номер_группы, Курс_обучения) VALUES (@ID, @Фамилия, @Имя, @Отчество,
@Номер_группы, @Курс_обучения)";
    /*@Общежитие, @Комната);*/
    SqlConnection connection = new
    SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
    connection.Open();
    SqlCommand command = connection.CreateCommand();
    command.CommandText = sqlInsert;
    command.Parameters.AddWithValue("@Фамилия", textBoxFDML.Text);
    command.Parameters.AddWithValue("@Имя", textBoxNDML.Text);
    command.Parameters.Add("@Отчество", SqlDbType.NVarChar).Value =
    textBoxODML.Text;
    command.Parameters.AddWithValue("@Номер_группы", textBoxGroupDML.Text);
    command.Parameters.AddWithValue("@Курс_обучения", textBoxCourseDML.Text);
    /*command.Parameters.AddWithValue("@Общежитие", textBoxHostelDML.Text);
    command.Parameters.AddWithValue("@Комната", textBoxRoomNumDML.Text);*/

    connection.Close();
    buttonSelectStudent_Click(this, EventArgs.Empty);
}

private void buttonSelectStudent_Click(object sender, EventArgs e)
{
    string sqlSelect = @"SELECT Студент.Id_Студент, Студент.Фамилия, Студент.Имя,
Студент.Номер_группы, '2' FROM Студент ";
    /*SELECT * FROM PK_Студент WHERE PK_Студент.PK_Студент LIKE @ID*/
    SqlConnection connection = new
    SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
    SqlCommand command = connection.CreateCommand();
    command.CommandText = sqlSelect;
    /*command.Parameters.Add("@ID", SqlDbType.NChar);
    command.Parameters["@ID"].Value = textBoxWorker.Text + "%";*/
    SqlDataAdapter adapter = new SqlDataAdapter();
    adapter.SelectCommand = command;
    DataTable table = new DataTable();
    adapter.Fill(table);
    dataGridView1.DataSource = table;
    if (table.Rows.Count == 0) MessageBox.Show("Нет значений!",
    "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
void UpdateStudent()
{
    if (String.IsNullOrEmpty(textBoxID.Text))
    {
        MessageBox.Show("Обязательно укажите ID студента, данные которого
подлежать корректировке", "Внимание", MessageBoxButtons.OK,
        MessageBoxIcon.Warning);
        return;
    }
    int id;
}

```

```

    if (!int.TryParse(textBoxID.Text, out id))
    {
        MessageBox.Show("Некоректное значение ID студента!", "Внимание",
    MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    string sqlUpdate = "update Студент set Фамилия= @Фамилия, Имя = @Имя,
Отчество = @Отчество, Номер_группы = @Номер_группы, Курс_обучения = @Курс_обучения WHERE
Id_Студент = @id ";
    /*UPDATE PK_Студент SET Фамилия= @Фамилия, Имя = @Имя, Отчество = @Отчество,
Номер_группы = @Номер_группы, Курс_обучения = @Курс_обучения WHERE PK_Студент = @id */;
    SqlConnection connection = new
    SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
    connection.Open();
    SqlCommand command = connection.CreateCommand();
    string sqlValues = "";
    if (!String.IsNullOrEmpty(textBoxFDML.Text))
        sqlValues += "Фамилия=@Фамилия";

    if (!String.IsNullOrEmpty(textBoxNDML.Text))
        sqlValues += "Имя=@Имя";

    if (!String.IsNullOrEmpty(textBoxODML.Text))
        sqlValues += "Отчество=@Отчество";

    if (!String.IsNullOrEmpty(textBoxGroupDML.Text))
        sqlValues += "Номер_группы=@Номер_группы";
    }
    void DeleteStudent()
    {
        if (String.IsNullOrEmpty(textBoxID.Text))
        {
            MessageBox.Show("Обязательно укажите ID записи, которую необходимо
удалить", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Warning);
            return;
        }
        int id;
        if (!int.TryParse(textBoxID.Text, out id))
        {
            MessageBox.Show("Некоректное значение кода блюда!", "Внимание",
    MessageBoxButtons.OK, MessageBoxIcon.Warning);
            return;
        }
        string sqlDelete = @"DELETE FROM Студент WHERE Id_Студент = @id";
        SqlConnection connection = new
        SqlConnection(Properties.Settings.Default.ne_nadoConnectionString);
        connection.Open();
        SqlCommand command = connection.CreateCommand();
        command.CommandText = sqlDelete;
        command.Parameters.AddWithValue("@id", textBoxID.Text);
        try
        {
            command.ExecuteNonQuery();
        }
        catch (Exception err)
        {
            MessageBox.Show(err.Message, "Ошибка удаления");
        }
        connection.Close();
        buttonSelectStudent_Click(this, EventArgs.Empty);
    }
}

```

```
    }

    private void groupBoxDML_Enter(object sender, EventArgs e)
    {
    }

}
```

Приложение Д  
(Справочное)

Листинг формы Группа

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace KursavBSBD.Студент
{
    public partial class DNStudent : Form
    {
        public DNStudent()
        {
            InitializeComponent();
        }

        private void студентBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            try
            {
                this.Validate();
                this.студентBindingSource.EndEdit();
                this.tableAdapterManager.UpdateAll(this.ne_nadoDataSet);
            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Ошибка при заполнении данных",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

        private void DNStudent_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу
            "ne_nadoDataSet.Студент". При необходимости она может быть перемещена или удалена.
            this.студентTableAdapter.Fill(this.ne_nadoDataSet.Студент);
        }

        // одиночка
        private static DNStudent f;
        public static DNStudent fd
        {
            get
            {
                if (f == null || f.IsDisposed) f = new DNStudent();
                return f;
            }
        }

        private void toolStripLabel1_Click(object sender, EventArgs e)
    }
}

```

```

{
}

private void студентDataGridView_CellFormatting(object sender,
DataGridViewCellFormattingEventArgs e)
{
    if
((студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value == null)
||

(студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value.ToString(
) == "1")) e.CellStyle.BackColor = Color.Green;
    else
    {
        if
(студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value.ToString(
) == "2")
            e.CellStyle.BackColor = Color.SkyBlue;
        if
(студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value.ToString(
) == "3")
            e.CellStyle.BackColor = Color.Yellow;
        if
(студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value.ToString(
) == "4")
            e.CellStyle.BackColor = Color.Pink;
        if
(студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value.ToString(
) == "5")
            e.CellStyle.BackColor = Color.Blue;
        if
(студентDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn6"].Value.ToString(
) == "6")
            e.CellStyle.BackColor = Color.Aqua;
    }
}

private void toolStripButton1_Click(object sender, EventArgs e)
{
}

private void ученикBindingNavigator_RefreshItems(object sender, EventArgs e)
{
}

private void toolStripButtonFind_Click(object sender, EventArgs e)
{
    if (toolStripTextBoxFind.Text == "")
    {
        MessageBox.Show("Вы ничего не задали", "Внимание",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        return;
    }
    int indexPos;
    try
{

```

```
        indexPos =
    студентBindingSource.Find(GetSelectedFieldName(),toolStripTextBoxFind.Text);
    }
    catch (Exception err)
    {
        MessageBox.Show("Ошибка поиска \n" + err.Message);
        return;
    }
    if (indexPos > -1)
        студентBindingSource.Position = indexPos;
    else
    {
        MessageBox.Show("Таких студентов нет", "Внимание",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        студентBindingSource.Position = 0;
    }

}

private void checkBoxFind_CheckedChanged(object sender, EventArgs e)
{
    if (checkBoxFind.Checked)
    {
        if (toolStripTextBoxFind.Text == "")
            MessageBox.Show("Вы ничего не задали", "Внимание",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
        else
            try
            {
                студентBindingSource.Filter =
                    GetSelectedFieldName() + "=" + toolStripTextBoxFind.Text + "";
            }
            catch (Exception err)
            {
                MessageBox.Show("Ошибка фильтрации \n" +
                    err.Message);
            }
    }
    else
        студентBindingSource.Filter = "";
    if (студентBindingSource.Count == 0)
    {
        MessageBox.Show("Нет таких");
        студентBindingSource.Filter = "";
        checkBoxFind.Checked = false;
    }
}

private string GetSelectedFieldName()
{
    throw new NotImplementedException();
}
```

Приложение Е  
(Справочное)

Листинг формы «Распоряжение»

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace KursavBSBD.Распоряжение
{
    public partial class DNRaspore : Form
    {
        public DNRaspore()
        {
            InitializeComponent();
        }

        private void распоряжениеBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.распоряжениеBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.ne_nadoDataSet);

        }

        private void DNRaspore_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу
            "ne_nadoDataSet.Распоряжение". При необходимости она может быть перемещена или удалена.
            this.распоряжениеTableAdapter.Fill(this.ne_nadoDataSet.Распоряжение);

        }

        private void распоряжениеDataGridView_CellFormatting(object sender,
DataGridViewCellFormattingEventArgs e)
        {
            if
((распоряжениеDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn3"].Value ==
null) ||
(распоряжениеDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn3"].Value.ToString() ==
"1")) eCellStyle.BackColor = Color.LightGreen;
            else
{
            if
(распоряжениеDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn3"].Value.ToString() ==
"2")
                eCellStyle.BackColor = Color.SkyBlue;
            if

```

```
(распоряжениеDataGridView.Rows[e.RowIndex].Cells["dataGridViewTextBoxColumn3"].Value.ToString() == "4")
    e.CellStyle.BackColor = Color.Yellow;
else
    e.CellStyle.BackColor = Color.Pink;
}
}

private void распоряжениеBindingNavigator_RefreshItems(object sender, EventArgs
e)
{
}
}
}
```

Приложение Ё  
(Справочное)  
Хеширование «SHA1»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography;

namespace KursavBSBD
{
    class Class2
    {
        public string Hash(string input)
        {
            using (SHA1Managed Password = new SHA1Managed())
            {
                var hash = Password.ComputeHash(Encoding.UTF8.GetBytes(input));
                var sb = new StringBuilder(hash.Length * 2);
                foreach (byte b in hash)
                {
                    sb.Append(b.ToString("x2"));
                }
                return sb.ToString();
            }
        }
    }
}
```