**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Институт информационных технологий и управления в технических системах (полное название института)

кафедра «Информационные системы» (полное название кафедры)

**Пояснительная записка**  к расчетно-графической работе по дисциплине «Управление данными»

|  |  |
| --- | --- |
| на тему | «База данных гостиничного комплекса» |
|  | |
|  | |

|  |  |
| --- | --- |
| Выполнил: студент III курса, группы: | ИС/б-20-2-о |
| Направления подготовки (специальности) | 09.03.02 |

Информационные системы и технологии

(код и наименование направления подготовки (специальности))

|  |  |
| --- | --- |
| профиль (специализация) | Информационные системы |
|  | |
| Белик Георгий Максимович | |

(фамилия, имя, отчество студента)

|  |  |
| --- | --- |
| Руководитель | Гончаренко Дмитрий Геннадиевич, доцент кафедры |

(фамилия, инициалы, степень, звание, должность)

|  |  |  |  |
| --- | --- | --- | --- |
| Защита | « 30 » декабря 2022 г. | Оценка |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Руководитель |  |  |  |

(подпись) (инициалы, фамилия)

20 22 г.

СОДЕРЖАНИЕ

[**1 АНАЛИТИЧЕСКАЯ ЧАСТЬ** 4](#_Toc123311968)

[1.1 Анализ предметной области 4](#_Toc123311969)

[1.2 Постановка задачи 6](#_Toc123311970)

[**2 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ** 7](#_Toc123311971)

[2.1 Построение простой и сложной сетевых моделей 7](#_Toc123311972)

[2.2 Построение диаграммы «сущность-связь» 8](#_Toc123311973)

[2.3 Построение модели, основанной на ключах 9](#_Toc123311974)

[2.4 Построение полной атрибутивной модели 10](#_Toc123311975)

[**3 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ** 12](#_Toc123311976)

[3.1 Выбор аппаратной и программной платформы для реализации БД 12](#_Toc123311977)

[3.2 Реализация базы данных 12](#_Toc123311978)

[3.3 Тестирование базы данных 21](#_Toc123311979)

[**4 РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ** 26](#_Toc123311980)

[4.1. Подсистема клиента 26](#_Toc123311981)

[4.2 Подсистема администратора 28](#_Toc123311982)

[4.4 Тестирование системы 30](#_Toc123311983)

[**ЗАКЛЮЧЕНИЕ** 32](#_Toc123311984)

[**ЛИТЕРАТУРНЫЕ ИСТОЧНИКИ** 33](#_Toc123311985)

[**ПРИЛОЖЕНИЕ А Исходный код программы** 34](#_Toc123311986)

**ВВЕДЕНИЕ**

В рамках настоящей работы ведется разработка мобильного приложения для работы с базой данных на тему «База данных медицинской страховой компании» на основании документа – техническое задание и в рамках организации – Севастопольский государственный университет. Дата выдачи задания: 17.09.2022.

Целью расчетно-графической работы является систематизация, закрепление и углубление знаний в области управления данными на SQL подобных языках и их совершенствование путем применения при разработке простого мобильного приложения.

Для достижения цели на разных этапах курсового проектирования должны быть решены следующие задачи:

* анализ предметной области;
* построение логической модели БД;
* построение физической БД;
* разработка клиентского приложения.

# **1 АНАЛИТИЧЕСКАЯ ЧАСТЬ**

## 1.1 Анализ предметной области

Гостиничный комплекс – юридическое лицо, которое занимается предоставлением услуг о расселении, проживании, отдыхе посетителей. Она может иметь различные апартаменты, перечтены такие как обычный номер, вилла и спа.

Поскольку предметной областью является база данных гостиничного комплекса, основной сущностью, связывающей межу собой все остальные, должна выступать бронь. Она должен содержать такое данные как id клиента, id номера, id персонала, закрепленного за ним, и данные о том, в каких из всех апартаментов гость проживает.

Помимо сущности «Бронь» база данных должна содержать сущности гостей, содержащие их id, ФИО, данные паспорта, даты заселения, выселения. В базу данных следует включить сущность «Номер», который будет содержать основные данные для персонала, такую как состояние (чистота, свободен ли и готов к заселению), его стоимость, а также вместимость.

Поскольку, в сущности, «Бронь» имеются данные о рабочем (ответственной уборщице в случае необходимости), требуется создать отдельную сущность для персонала, содержащую его данные (ID рабочего, к каким апартаментам относится, название должности и ФИО).

Сущность персонала будет зависеть от сущности «Должность», в которой будут храниться данные о названии должность, и о зарплате. Также должность будет включать в себя такую сущность как расписание, чтобы у каждой профессии был свой график работы.

Последней сущностью для создания полноценной базы данных является сущность апартаментов данной гостиницы. Апартаменты должны иметь название гостиницы, название апартаментов, тип, администратора.

## 1.2 Постановка задачи

База данных должны хранить следующие сущности:

* апартаменты;
* персонал;
* должность;
* расписание;
* номер.
* гость.
* бронь.

Сущности «Персонал» и «Расписание» должны содержать поле с названием должности из сущности «Должность», сущность «Апартаменты» должна содержать поле «название гостиницы» из сущности «Гостиница», сущность «Номер» должен содержать поле «название апартаментов» из сущности «Апартаменты», сущность «Гость» должна содержать поле «ID номера» из сущности «Номер», и наконец-то сущность «Бронь» должна содержать поля «ID гостя» из сущности «Гость», «ID номера» из сущности «Номер», «ID персонала» из сущности «Персонал», «ID апартаментов» из сущности «Апартаменты».

База данных будет доступна для клиентов, и администраторов.

Представители группы клиентов должны иметь доступ к таблице существующих броней, чтобы найти свою или же найти желаемый номер и увидеть его доступность на данный момент.

Представителям группы администраторов должны быть доступны все действия с таблицей «Бронь» :просмотр, поиск, удаление данных, их сохранение.

После анализа предметной области и определения с основными сущностями и их атрибутами можно приступать к созданию логической модели БД.

# **2 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ**

## 2.1 Построение простой и сложной сетевых моделей

Была построена сетевая модель базы данных.

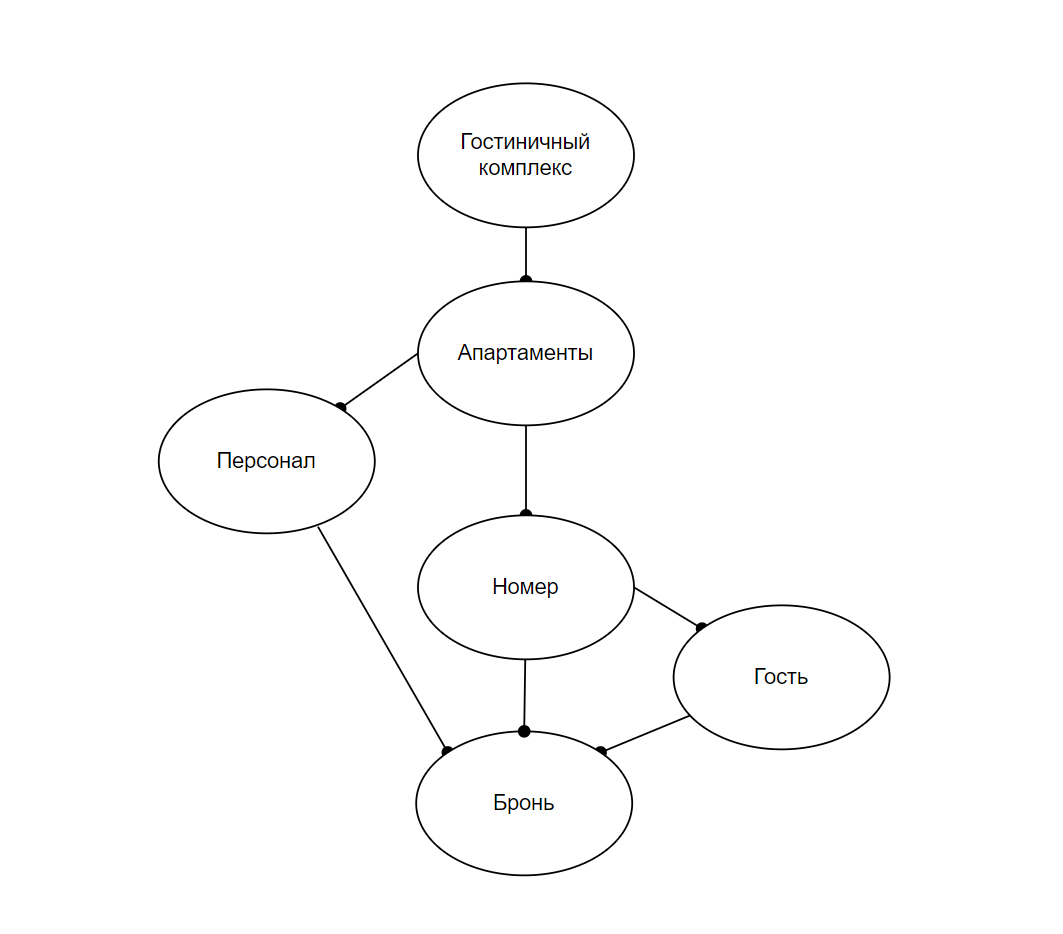


Рисунок 1 – Сетевая модель

После построения сетевой моделей в которых определены связи между сущностями можно переходить к построению диаграммы «сущность-связь»

## 2.2 Построение диаграммы «сущность-связь»

Диаграмма «сущность-связь» определяет более глубокие взаимосвязи между сущностями и содержит их атрибуты.

Из простой сетевой модели была построена диаграмма «сущность-связь». К сущностям были добавлены атрибуты и определены взаимосвязи (Рис. 3).

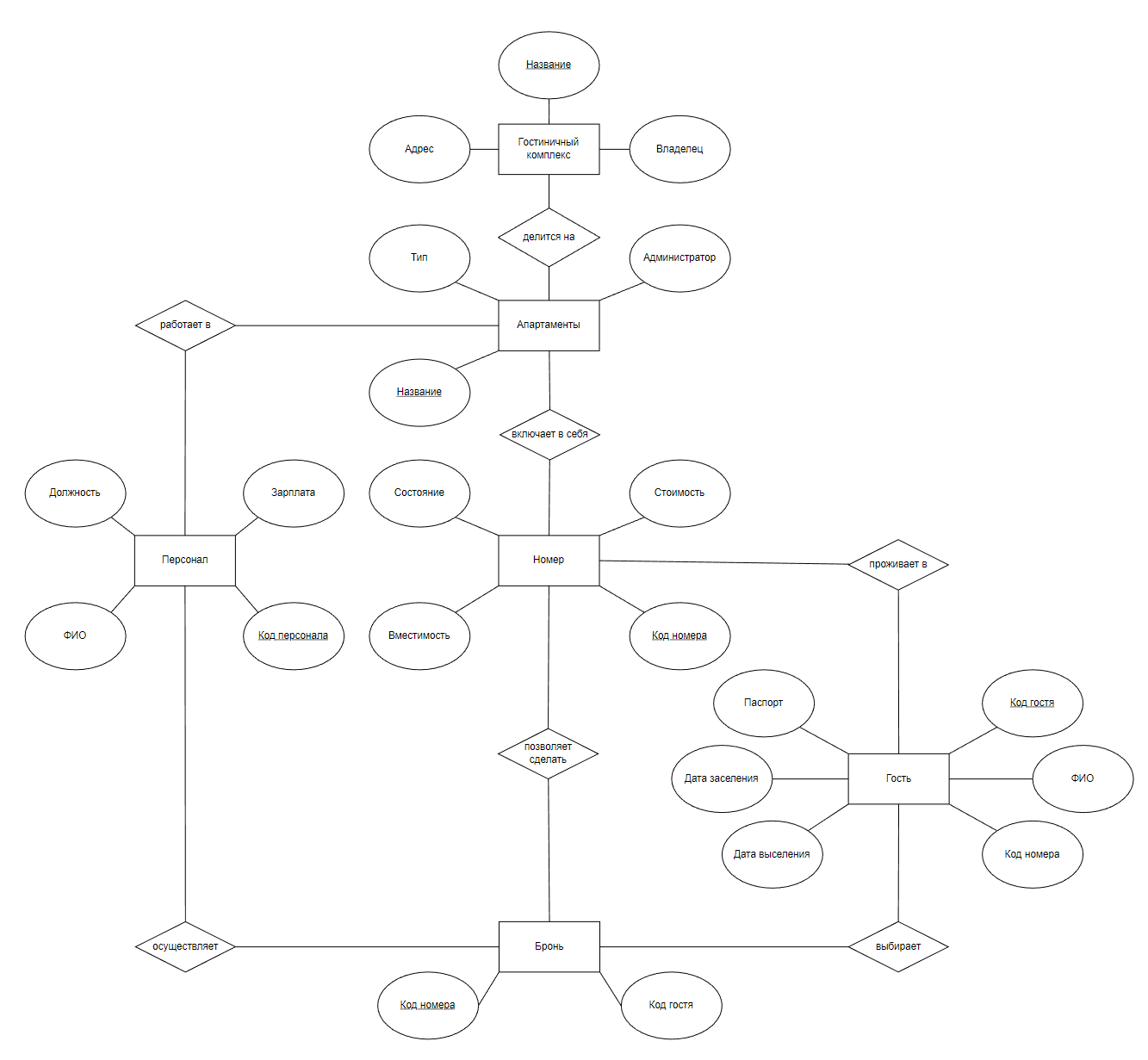


Рисунок 3 – Диаграмма «сущность-связь»

После построения диаграммы «сущность-связь» можно переходить к построению диаграммы построенной на ключах.

## 2.3 **Построение модели, основанной на ключах**

Диаграмма, основанная на ключах представляет сущности в виде прямоугольников, в которых указываются первичные и внешние ключи.

Из диаграммы «сущность-связь» была построена модель, основанная на ключах. Из атрибутов сущности были выбраны первичные ключи и определены внешние ключи (Рис. 4).

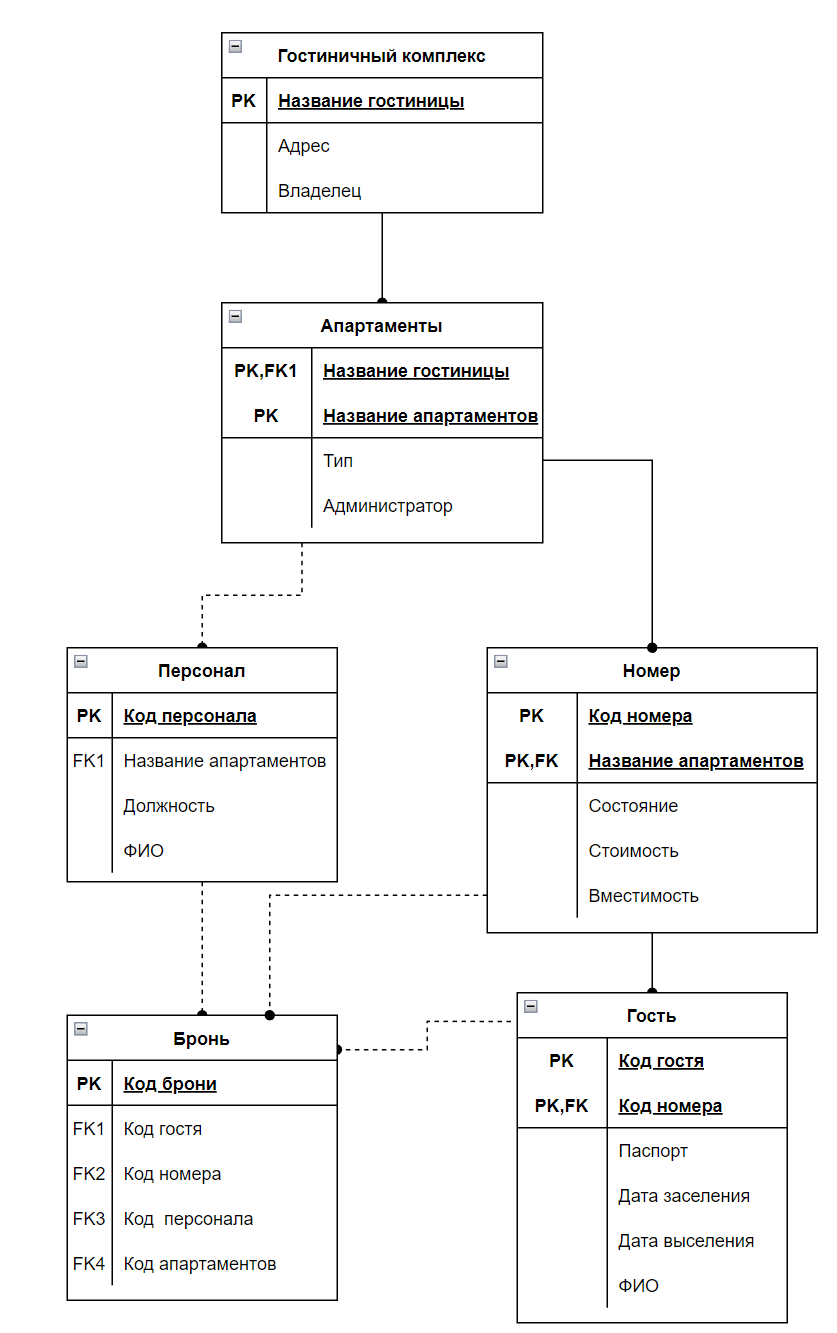


Рисунок 3 – Модель БД на ключах

Из диаграммы, основанной на ключах, можно переходить к последнему этапу построения логической модели БД.

## 2.4 Построение полной атрибутивной модели

Перед построением полной атрибутивной модели необходимо провести нормализацию БД ради устранения лишних атрибутов и взаимосвязей.

1НФ. Все выделенные атрибуты атомарны, следовательно, схема находится в 1НФ.

2НФ. Все неключевые атрибуты зависят полностью от ключей, следовательно, схема находится в 2НФ.

3НФ. Все неключевые атрибуты нетранзитивно зависит от первичного ключа, следовательно, схема находится в 3НФ.

Из нормализации видно что в БД нет лишних взаимосвязей и атрибутов.

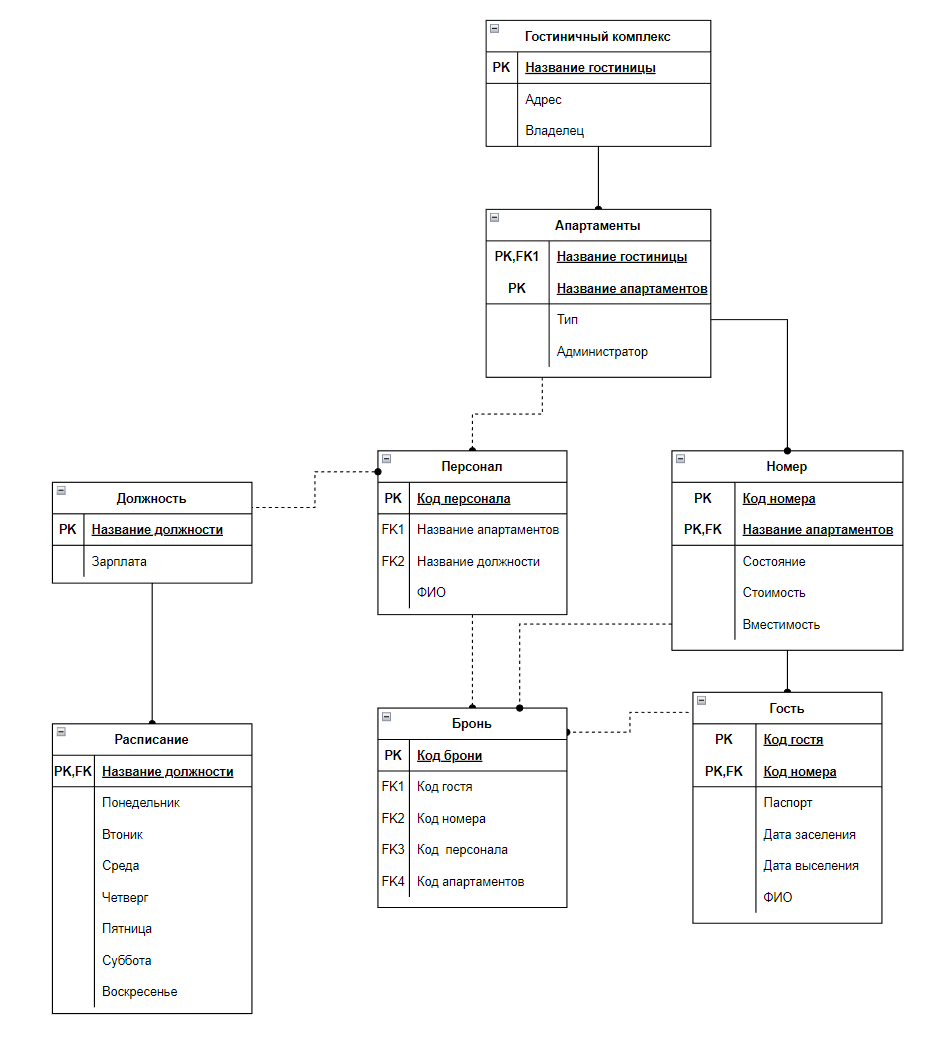


Рисунок 4 – Полная атрибутивная модель

# **3 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ**

## 3.1 Выбор аппаратной и программной платформы для реализации БД

Для реализации физической модели базы данных была выбрана платформа Microsoft SQL Server Management Studio. Microsoft SQL Server Management Studio – это Утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера. Также для построения клиентского приложения было выбрано приложение Microsoft Visual Studio, Windows Forms, использован язык программирования c#.

Для полей названия гостиницы, названия апартаментов, типа апартаментов, администратора, названия персонала, названия должности, графика работы, ФИО работников, названия номеров, состояния номеров, паспортов гостей, дат пребывания, ФИО гостей использовался строковый тип данных varchar().

Остальные поля сущностей базы данных имеют числовой тип данных, поскольку хранят либо количественную меру, либо номер.

## 3.2 Реализация базы данных

В приложении Microsoft SQL Server Management Studio были созданы и заполнены таблицы базы данных.

create database hotel\_complex;

use hotel\_complex;

CREATE TABLE hotel (

hotel\_name varchar(50) NOT NULL PRIMARY KEY,

address varchar(50) NOT NULL,

owner VARCHAR(100) NOT NULL

);

CREATE TABLE apartments(

hotel\_name varchar(50) NOT NULL,

appart\_name VARCHAR (50) NOT NULL,

style VARCHAR (50) NOT NULL check (style = 'villa' or style = 'number' or style = 'spa'),

administrator VARCHAR (50) NOT NULL,

PRIMARY KEY (appart\_name),

FOREIGN KEY (hotel\_name) REFERENCES hotel (hotel\_name) ON DELETE CASCADE ON UPDATE CASCADE

) ;

CREATE TABLE job(

job\_title VARCHAR(50) NOT NULL PRIMARY KEY,

salary INTEGER NOT NULL

);

CREATE TABLE timetable (

job\_title VARCHAR(50) NOT NULL PRIMARY KEY,

FOREIGN KEY(job\_title) REFERENCES job (job\_title) ON DELETE CASCADE ON UPDATE CASCADE,

monday VARCHAR(100),

tuesday VARCHAR(100),

wednesday VARCHAR(100),

thursday VARCHAR(100),

friday VARCHAR(100),

saturday VARCHAR(100),

sunday VARCHAR(100)

);

CREATE TABLE personnel(

personnel\_id INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

appart\_name VARCHAR (50) NOT NULL,

FOREIGN KEY (appart\_name) REFERENCES apartments (appart\_name) ON DELETE CASCADE ON UPDATE CASCADE,

job\_title VARCHAR(50) NOT NULL,

FOREIGN KEY (job\_title) REFERENCES job (job\_title) ON DELETE CASCADE ON UPDATE CASCADE,

fio VARCHAR(100) NOT NULL

);

CREATE TABLE number(

number\_id INTEGER IDENTITY(1,1) NOT NULL,

appart\_name VARCHAR (50) NOT NULL,

FOREIGN KEY (appart\_name) REFERENCES apartments (appart\_name) ON DELETE CASCADE ON UPDATE CASCADE,

PRIMARY KEY (number\_id),

cleanness VARCHAR (50) NOT NULL,

price INTEGER NOT NULL,

capacity INTEGER NOT NULL

);

CREATE TABLE guest(

guest\_id INTEGER IDENTITY(1,1) NOT NULL,

number\_id INTEGER NOT NULL,

PRIMARY KEY (guest\_id),

FOREIGN KEY (number\_id) REFERENCES number (number\_id) ON DELETE CASCADE ON UPDATE CASCADE,

passport VARCHAR (100) NOT NULL,

comming DATE NOT NULL,

leave DATE NOT NULL,

FIO VARCHAR (100) NOT NULL

);

CREATE TABLE reservation(

id INTEGER IDENTITY(1,1) NOT NULL ,

guest\_id INTEGER NOT NULL ,

number\_id INTEGER NOT NULL ,

personnel\_id INTEGER NOT NULL ,

appart\_name VARCHAR (50) NOT NULL,

PRIMARY KEY(id,guest\_id,number\_id,personnel\_id,appart\_name),

FOREIGN KEY (guest\_id) REFERENCES guest (guest\_id) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (number\_id) REFERENCES number (number\_id) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (personnel\_id) REFERENCES personnel (personnel\_id) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (appart\_name) REFERENCES apartments (appart\_name) ON DELETE NO ACTION ON UPDATE NO ACTION

);

CREATE TABLE register(

id\_user INT IDENTITY(1,1) NOT NULL,

login\_user VARCHAR (50) NOT NULL,

password\_user VARCHAR (50) NOT NULL

);

Далее было осуществлено заполнение таблиц данными:

INSERT INTO hotel VALUES('MRIYA RESORT AND SPA','YALTA','Сбербанк');

INSERT INTO hotel VALUES('Общаги СевГУ','Sevastopol','Нечаев Владимир Дмитриевич');

SELECT \* FROM hotel;

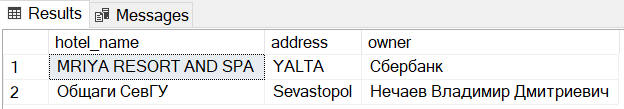


Рисунок 5 – Таблица отелей

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','deluxe-1','villa','Василий');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','deluxe-2','number','Василий');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','deluxe-3','spa','Василий');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','luxe-1','villa','Ирина');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','luxe-2','number','Ирина');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','luxe-3','spa','Ирина');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','family-1','villa','Мария');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','family-2','number','Мария');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','family-3','spa','Мария');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','king-1','villa','Олег');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','king-2','number','Олег');

INSERT INTO apartments VALUES('MRIYA RESORT AND SPA','king-3','spa','Олег');

INSERT INTO apartments VALUES('Общаги СевГУ','№1-101','number','Инга');

INSERT INTO apartments VALUES('Общаги СевГУ','№1-102','number','Инга');

INSERT INTO apartments VALUES('Общаги СевГУ','№2-101','number','Валерий');

SELECT \* FROM apartments;

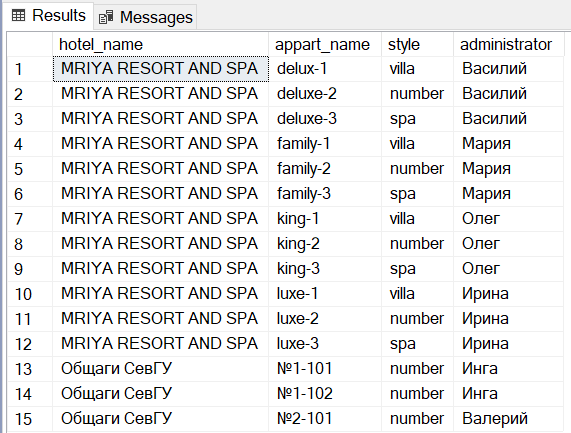


Рисунок 6 – Таблица апартаментов

INSERT INTO job VALUES('hostess',45000);

INSERT INTO job VALUES('security',35000);

INSERT INTO job VALUES('cleaning',30000);

INSERT INTO job VALUES('cook',50000);

INSERT INTO job VALUES('gardener',35000);

INSERT INTO job VALUES('driver',50000);

Select \* from job;

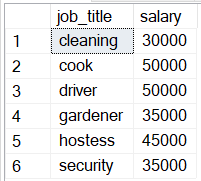


Рисунок 7 – Таблица должностей

INSERT INTO timetable VALUES('hostess','9:00-22:00','9:00-22:00','9:00-22:00','9:00-22:00','9:00-22:00','9:00-23:00','9:00-23:00');

INSERT INTO timetable VALUES('security','all day','all day','all day','all day','all day','all day','all day');

INSERT INTO timetable VALUES('cleaning','12:00-13:00','12:00-13:00','12:00-13:00','12:00-13:00','12:00-13:00','12:00-14:00','12:00-14:00');

INSERT INTO timetable VALUES('cook','7:00-20:00','7:00-20:00','7:00-20:00','7:00-20:00','7:00-20:00','7:00-20:00','7:00-20:00');

INSERT INTO timetable VALUES('gardener','','','19:00-21:00','','','','19:00-21:00');

INSERT INTO timetable VALUES('driver','9:00-22:00','9:00-22:00','9:00-22:00','9:00-22:00','9:00-22:00','9:00-22:00','9:00-22:00');

select \* from timetable;



Рисунок 8 – Таблица расписания

INSERT INTO personnel VALUES('deluxe-1','hostess','Skrinnik Evgeniya Olegovna');

INSERT INTO personnel VALUES('deluxe-1','security','Petrov Ivan Ivanovich');

INSERT INTO personnel VALUES('deluxe-1','cleaning','Velnik Izabel Omarovna');

INSERT INTO personnel VALUES('deluxe-1','cook','Skaldin Idragim Filipovich');

INSERT INTO personnel VALUES('deluxe-1','gardener','Krinizkiy Vladimir Adamovich');

INSERT INTO personnel VALUES('deluxe-1','driver','Holischev Rodion Maksimovich');

INSERT INTO personnel VALUES('luxe-1','hostess','Skrovik Evgeniya Olegovna');

INSERT INTO personnel VALUES('luxe-1','security','Petuhov Ivan Ivanovich');

INSERT INTO personnel VALUES('luxe-1','cleaning','felipov Izabel Omarovna');

INSERT INTO personnel VALUES('luxe-1','cook','Skudin Idragim Filipovich');

INSERT INTO personnel VALUES('luxe-1','gardener','-');

INSERT INTO personnel VALUES('luxe-1','driver','Golischev Rodion Maksimovich');

INSERT INTO personnel VALUES('family-1','hostess','Skripnik Evgeniya Olegovna');

INSERT INTO personnel VALUES('family-1','security','Navalin Ivan Ivanovich');

INSERT INTO personnel VALUES('family-1','cleaning','Coboleva Izabel Omarovna');

INSERT INTO personnel VALUES('family-1','cook','Kodin Idragim Filipovich');

INSERT INTO personnel VALUES('family-1','gardener','Vahromeev Vladimir Adamovich');

INSERT INTO personnel VALUES('family-1','driver','Podlesniy Rodion Maksimovich');

INSERT INTO personnel VALUES('king-1','hostess','Krivaya Evgeniya Olegovna');

INSERT INTO personnel VALUES('king-1','security','Petrus Ivan Ivanovich');

INSERT INTO personnel VALUES('king-1','cleaning','Garan Izabel Omarovna');

INSERT INTO personnel VALUES('king-1','cook','Ananin Idragim Filipovich');

INSERT INTO personnel VALUES('king-1','gardener','Medzitov Vladimir Adamovich');

INSERT INTO personnel VALUES('king-1','driver','Fudin Rodion Maksimovich');

INSERT INTO personnel VALUES('№1-101','hostess','-');

INSERT INTO personnel VALUES('№1-101','security','Saburov Nurlan Imamovich');

INSERT INTO personnel VALUES('№1-101','cleaning','Primak Anastasia Viktorovna');

INSERT INTO personnel VALUES('№1-101','cook','-');

INSERT INTO personnel VALUES('№1-101','gardener','-');

INSERT INTO personnel VALUES('№1-101','driver','-');

select \* from personnel;

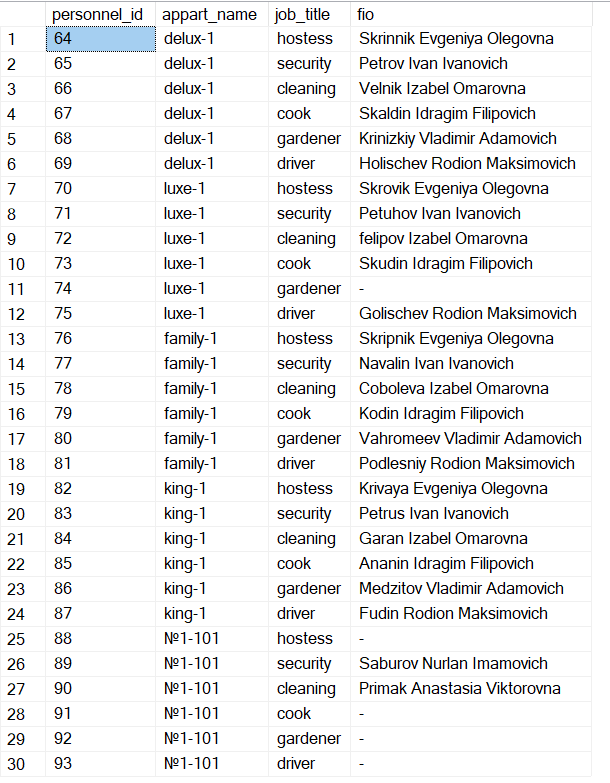


Рисунок 9 – Таблица персонала

INSERT INTO number VALUES('delux-1','yes',25000,2);

INSERT INTO number VALUES('delux-1','yes',27000,2);

INSERT INTO number VALUES('delux-1','no',23200,2);

INSERT INTO number VALUES('delux-1','no',24000,2);

INSERT INTO number VALUES('delux-1','yes',22500,2);

INSERT INTO number VALUES('delux-1','yes',28700,3);

INSERT INTO number VALUES('deluxe-2','yes',29500,3);

INSERT INTO number VALUES('deluxe-2','yes',31700,3);

INSERT INTO number VALUES('deluxe-3','no',28590,3);

INSERT INTO number VALUES('deluxe-3','yes',27830,3);

INSERT INTO number VALUES('luxe-1','yes',34000,3);

INSERT INTO number VALUES('luxe-1','no',37000,3);

INSERT INTO number VALUES('luxe-2','no info',32500,3);

INSERT INTO number VALUES('luxe-2','yes',38700,4);

INSERT INTO number VALUES('luxe-3','yes',32500,2);

INSERT INTO number VALUES('luxe-3','no info',38700,3);

INSERT INTO number VALUES('family-1','yes',44000,5);

INSERT INTO number VALUES('family-1','no',47000,5);

INSERT INTO number VALUES('family-2','no',42500,4);

INSERT INTO number VALUES('family-2','yes',48700,5);

INSERT INTO number VALUES('family-3','yes',42500,5);

INSERT INTO number VALUES('family-3','yes',48700,5);

INSERT INTO number VALUES('king-1','yes',74000,5);

INSERT INTO number VALUES('king-1','no info',77000,5);

INSERT INTO number VALUES('king-2','yes',72500,4);

INSERT INTO number VALUES('king-2','yes',78700,5);

INSERT INTO number VALUES('king-3','yno',72500,4);

INSERT INTO number VALUES('king-3','yes',78700,5);

INSERT INTO number VALUES('№1-101','no info',550,3);

INSERT INTO number VALUES('№1-102','yes',550,3);

INSERT INTO number VALUES('№2-101','no',600,2);

select \* from number;

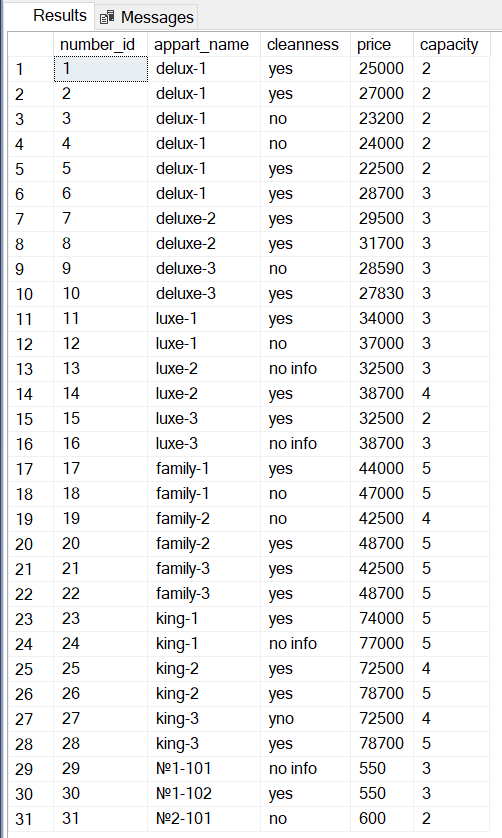


Рисунок 10 – Таблица номеров отеля

INSERT INTO guest VALUES(1,'russia 3715 080845','27-12-2022','07-01-2023','Velitan Anton Mihailovich');

INSERT INTO guest VALUES(7,'russia 2345 280845','24-12-2022','05-01-2023','Saakyan Grachik Vaaghnovich');

INSERT INTO guest VALUES(3,'russia 6254 380845','25-12-2022','03-01-2023','Kovalev Vladislav Andreevich');

INSERT INTO guest VALUES(9,'russia 7356 588045','22-12-2022','09-01-2023','Manuyan David Armenovich');

INSERT INTO guest VALUES(11,'russia 6256 088845','19-12-2022','11-01-2023','Sumulidi Afina Egorona');

INSERT INTO guest VALUES(13,'russia 7256 738568','26-12-2022','30-01-2023','Samohodova Elena Nicolaevna');

INSERT INTO guest VALUES(16,'russia 7456 983478','29-12-2022','23-01-2023','Drozin Foma Romanovich');

INSERT INTO guest VALUES(17,'russia 2525 456654','30-12-2022','07-01-2023','Kobchenko Albina Vitalivna');

INSERT INTO guest VALUES(19,'russia 3868 935853','12-12-2022','06-01-2023','Abramovich Aleksandra Uriebna');

INSERT INTO guest VALUES(22,'russia 3915 039234','30-11-2022','04-01-2023','Goncharenko Dmitriy Genadieviech');

INSERT INTO guest VALUES(23,'russia 3546 693946','23-11-2022','05-01-2023','Kovalenko Sviatoslav Timurovich');

INSERT INTO guest VALUES(26,'russia 3754 925549','15-12-2022','06-01-2023','Karlusov Vadim Urievich');

INSERT INTO guest VALUES(27,'russia 3895 209258','18-12-2022','09-01-2023','Smetanina Tatiana Ivanovna');

INSERT INTO guest VALUES(29,'russia 2492 239480','20-12-2022','11-01-2023','Svibovich Vladimir Vladimirovich');

INSERT INTO guest VALUES(29,'russia 2342 320494','19-12-2022','01-01-2023','Sulianov Anatoliy Sergeevich');

INSERT INTO guest VALUES(30,'russia 2434 053843','30-12-2022','02-01-2023','Garan Aleksandr Andreevich');

INSERT INTO guest VALUES(31,'russia 2394 259382','19-11-2022','01-02-2023','Pravdin Dmitriy Arturovich');

INSERT INTO guest VALUES(31,'russia 3895 348529','28-12-2022','07-01-2023','Belik Georgij Maksimovich');

select \* from guest;

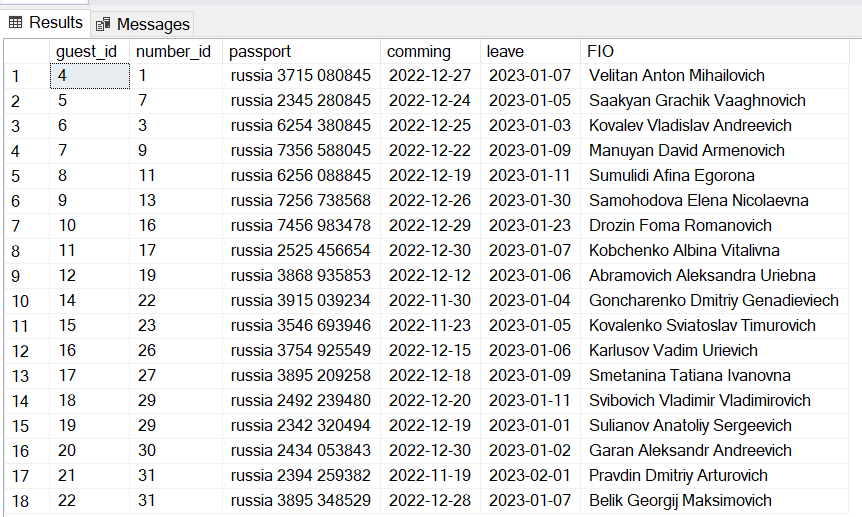


Рисунок 11 – Таблица гостей

INSERT INTO reservation VALUES(4,1,66,'deluxe-1');

INSERT INTO reservation VALUES(5,7,66,'deluxe-2');

INSERT INTO reservation VALUES(6,3,66,'deluxe-1');

INSERT INTO reservation VALUES(7,9,66,'deluxe-3');

INSERT INTO reservation VALUES(8,11,72,'luxe-1');

INSERT INTO reservation VALUES(9,13,72,'luxe-2');

INSERT INTO reservation VALUES(10,16,72,'luxe-3');

INSERT INTO reservation VALUES(11,17,78,'family-1');

INSERT INTO reservation VALUES(12,19,78,'family-2');

INSERT INTO reservation VALUES(14,22,78,'family-3');

INSERT INTO reservation VALUES(15,23,84,'king-1');

INSERT INTO reservation VALUES(16,26,84,'king-2');

INSERT INTO reservation VALUES(17,27,84,'king-3');

INSERT INTO reservation VALUES(18,29,90,'№1-101');

INSERT INTO reservation VALUES(19,29,90,'№1-101');

INSERT INTO reservation VALUES(20,30,90,'№1-102');

INSERT INTO reservation VALUES(21,31,90,'№2-101');

INSERT INTO reservation VALUES(22,31,90,'№2-101');

SELECT \* FROM reservation;

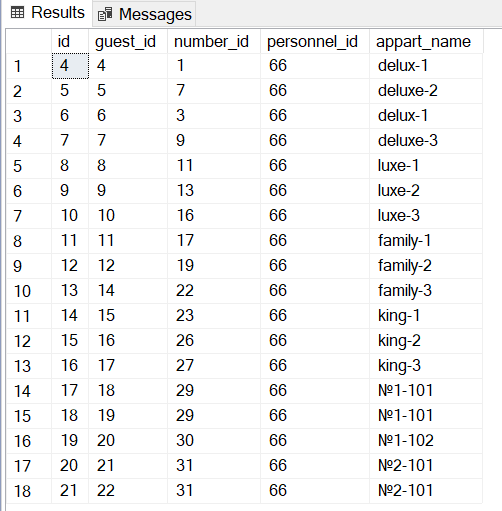


Рисунок 12 – Таблица брони

INSERT INTO register VALUES('admin','admin');

INSERT INTO register VALUES('guest','guest');

SELECT \* FROM register;

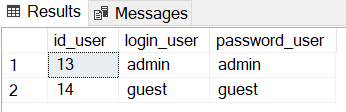


Рисунок 13 – Таблица пользователей

После создания и заполнения БД нужно протестировать ее на целостность и правильную работу запросов.

## 3.3 Тестирование базы данных

Все разграничения прав будут представлены в пользовательском приложении.

Для проверки внешних ключей на изменение данных были измены значения атрибутов.

Запрос изменения названия отеля.

UPDATE hotel

SET hotel\_name = 'Общага СевГУ'

WHERE hotel\_name = 'Общаги СевГУ';

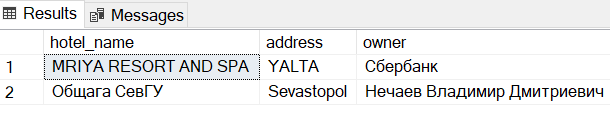


Рисунок 11 – Изменение в таблице отелей

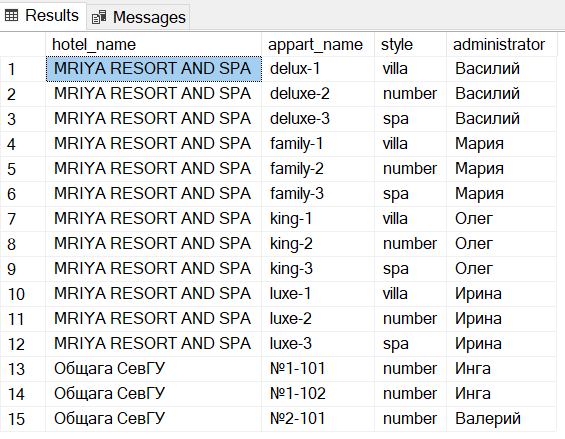


Рисунок 12 – Результат изменений в таблице апартаментов

Аналогично с рабочими и страховками.

Были введены следующие запросы, описанные далее.

Вывести данные о всех должностях.

Листинг 1 – Код запроса о должностях.

Select \* from job;

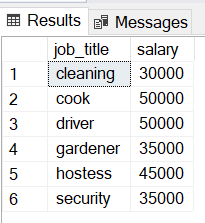


Рисунок 13 – Данные о филиалах

Посчитать количество гостей.

Листинг 2 – Код запроса о клиентах

Select count(guest\_id) from guest

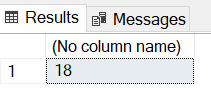


Рисунок 14 – Количество клиентов

Найти максимальную зарплату.

Листинг 3 – Код запроса о зарплате

Select max(salary) from job

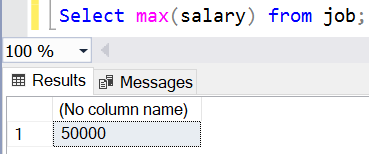


Рисунок 15 – Максимальная зарплата

Определить среднюю стоимость номера.

Листинг 4 – Код запроса о стоимости номера

Select avg(price) from number;

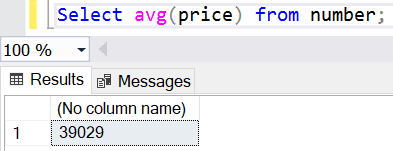


Рисунок 16 – Средняя цена номера

Определить рабочего с минимальной продуктивностью.

Листинг 5 – Код запроса минимума

Select min(salary) from job;

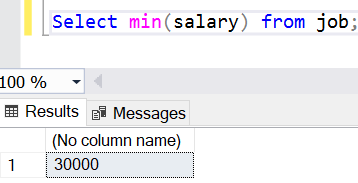


Рисунок 17 – Рабочий с минимальной продуктивностью

Найти все номера в апартаментах delux-1.

Листинг 6 – Код запроса на delux-1

Select \* from number where appart\_name = 'delux-1';

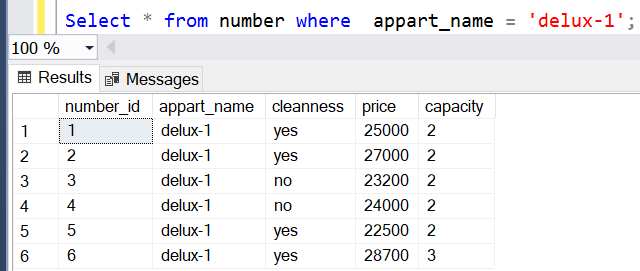


Рисунок 18 – Номера в delux-1

Вывести зарплаты рабочих и области их филиалов.

Листинг 7 – Код запроса отеля, апартаментов и стоимостей номеров

Select Distinct hotel\_name, number.appart\_name, price from number, apartments

where apartments.appart\_name = number.appart\_name;

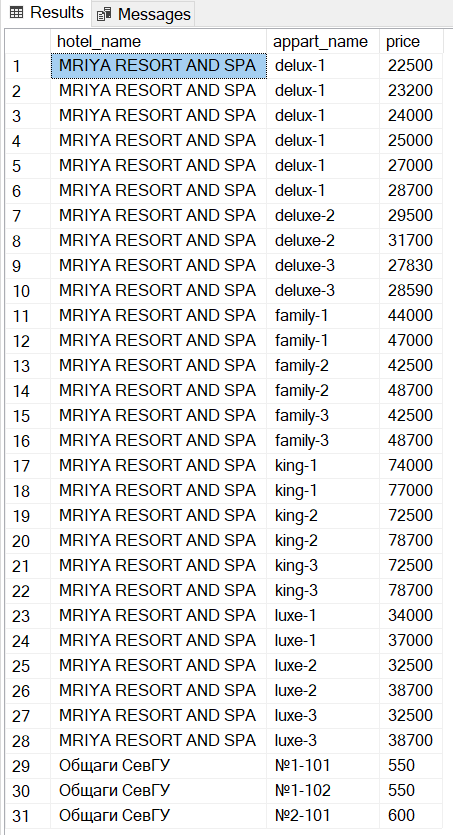


Рисунок 19 – Отели, апартаменты и стоимости номеров

Найти клиентов с именем на букву S.

Листинг 8 – Код запроса на имена клиентов

Select \* from guest where fio like 'S%';

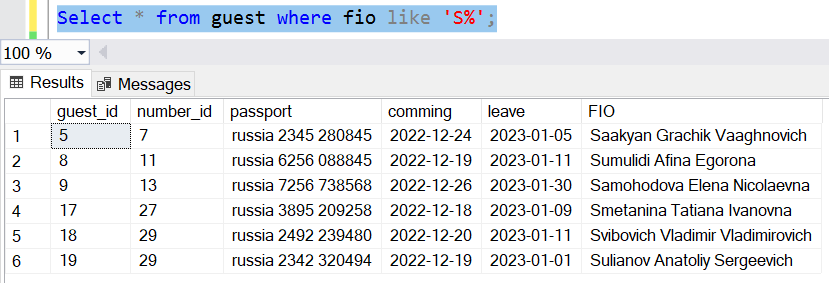


Рисунок 20 – Имена клиентов на букву S

Найти профессии с буквой a в названии.

Листинг 9 – Код запроса профессий с буквой f

Select \* from job where job\_title like '%a%';

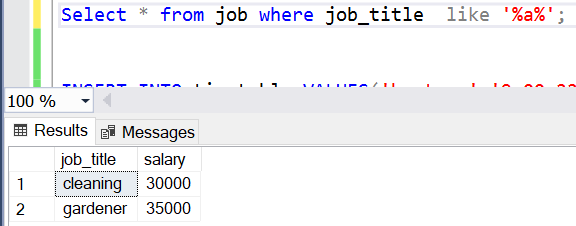


Рисунок 21 – Профессии с буквой a в названии

Найти имя клиента с российским паспортом 3895 348529.

Листинг 10 – Код запроса номеров

Select fio from guest where passport = 'russia 3895 348529';

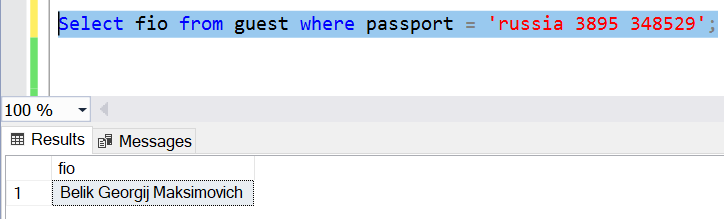


Рисунок 22 – Поиск клиента по номеру паспорта

# **4 РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ**

## 4.1. Подсистема клиента

Для клиентов доступна только таблица брони.

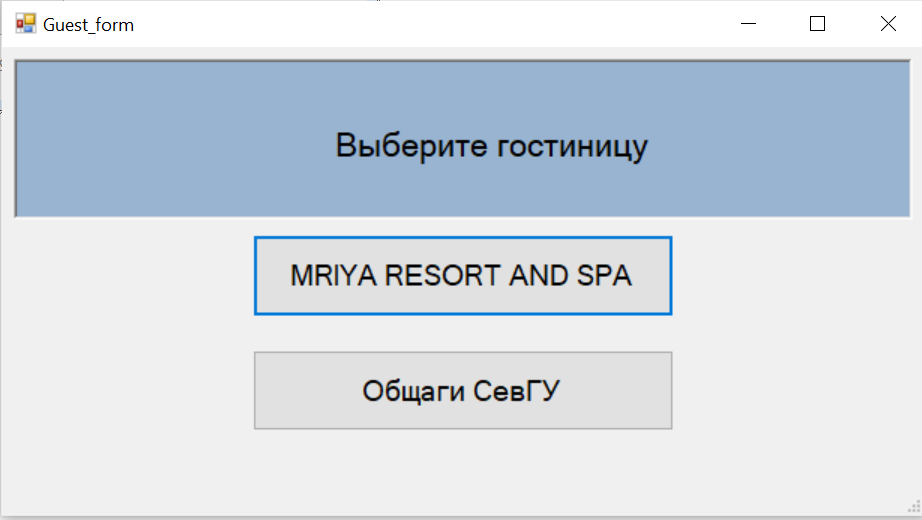


Рисунок 23 – Сообщение c выбором отеля

Пользователь может свободно просматривать доступные таблицы.

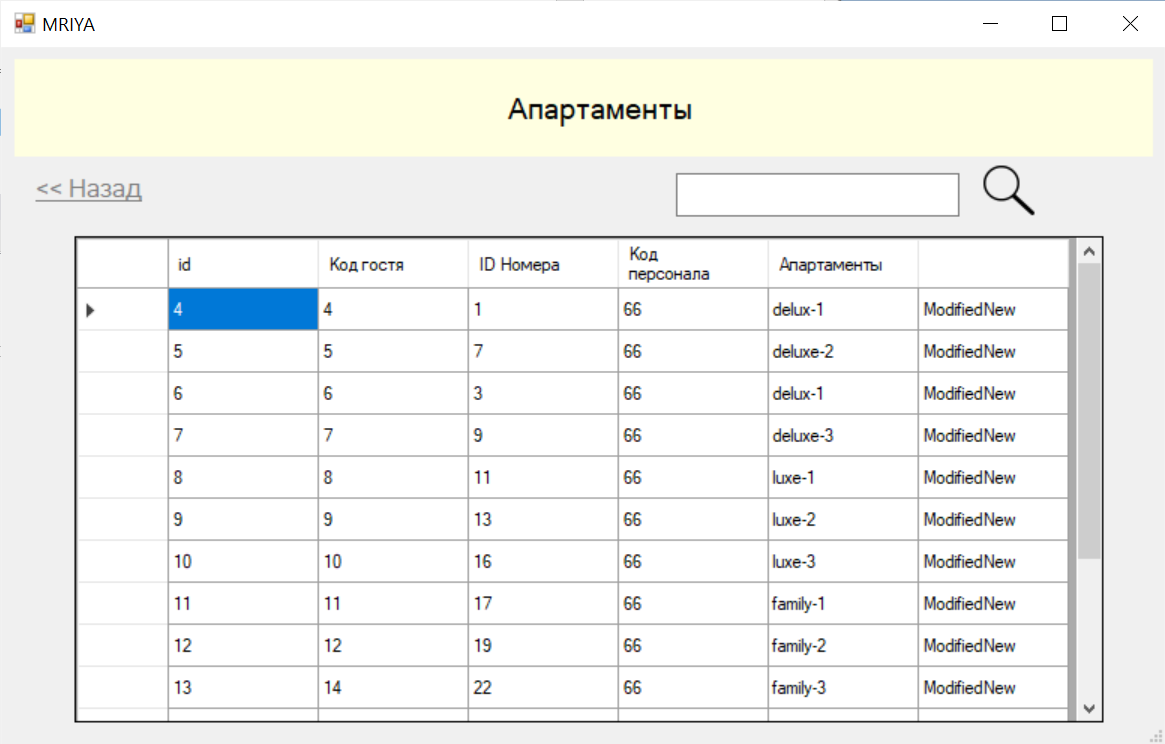


Рисунок 24 – Таблица отеля MRIYA

Пользователь может сортировать таблицу по любому столбцу.

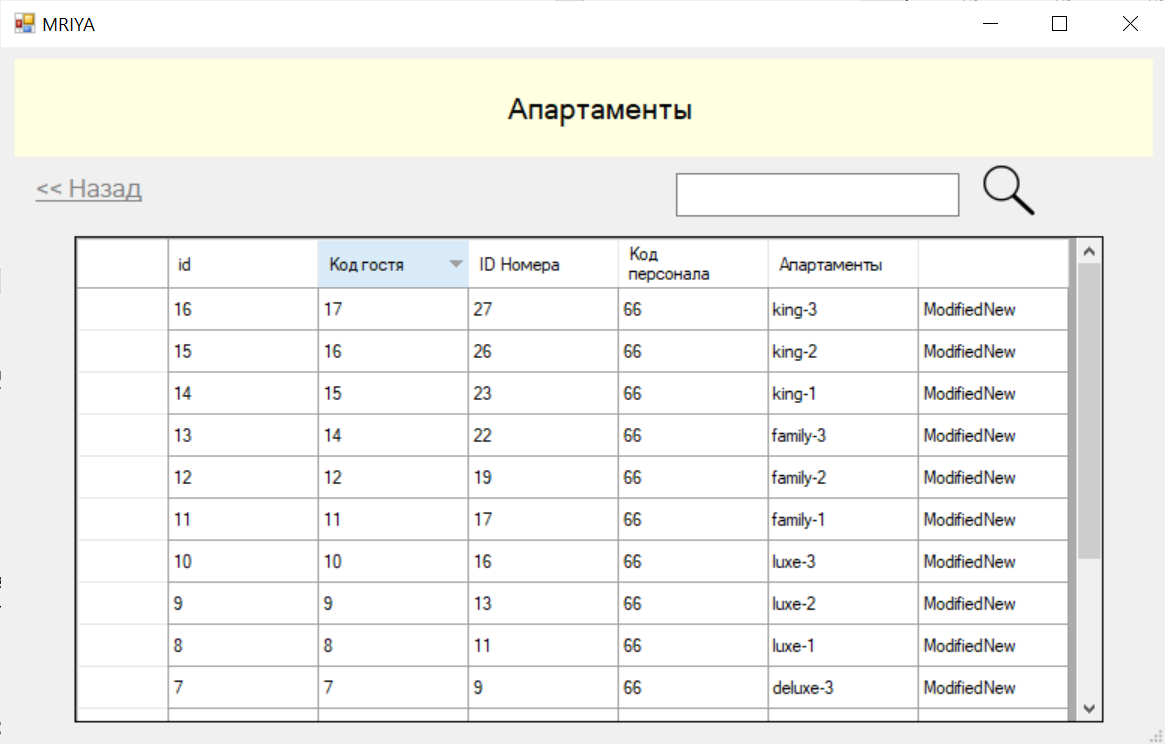


Рисунок 25 – Таблица отеля MRIYA с сортировкой

Пользователь может осуществлять поиск по любому атрибуту, например найти брони по семейным номерам. Также можно вернутся к выбору отелей, нажав кнопку «Назад».

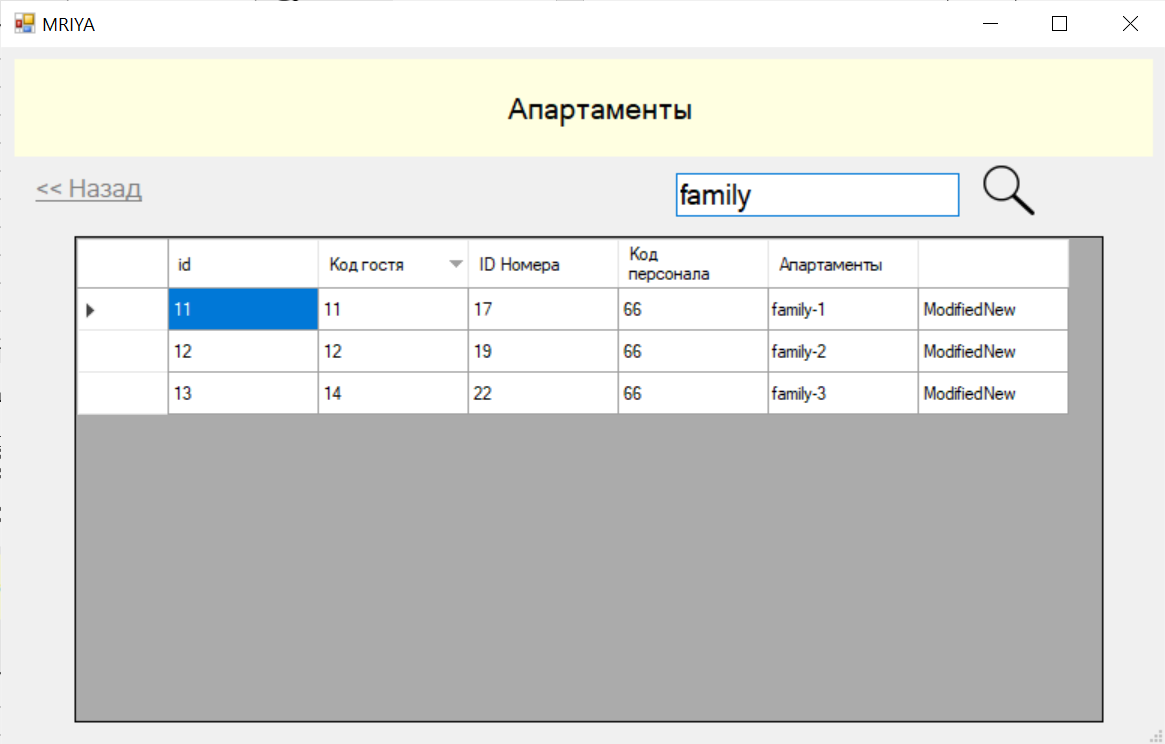


Рисунок 26 – Таблица отеля MRIYA с поиском

## 4.2 Подсистема администратора

Администратору доступны добавление записей в таблицу, просмотра, сортировка, поиск, удалений записей.

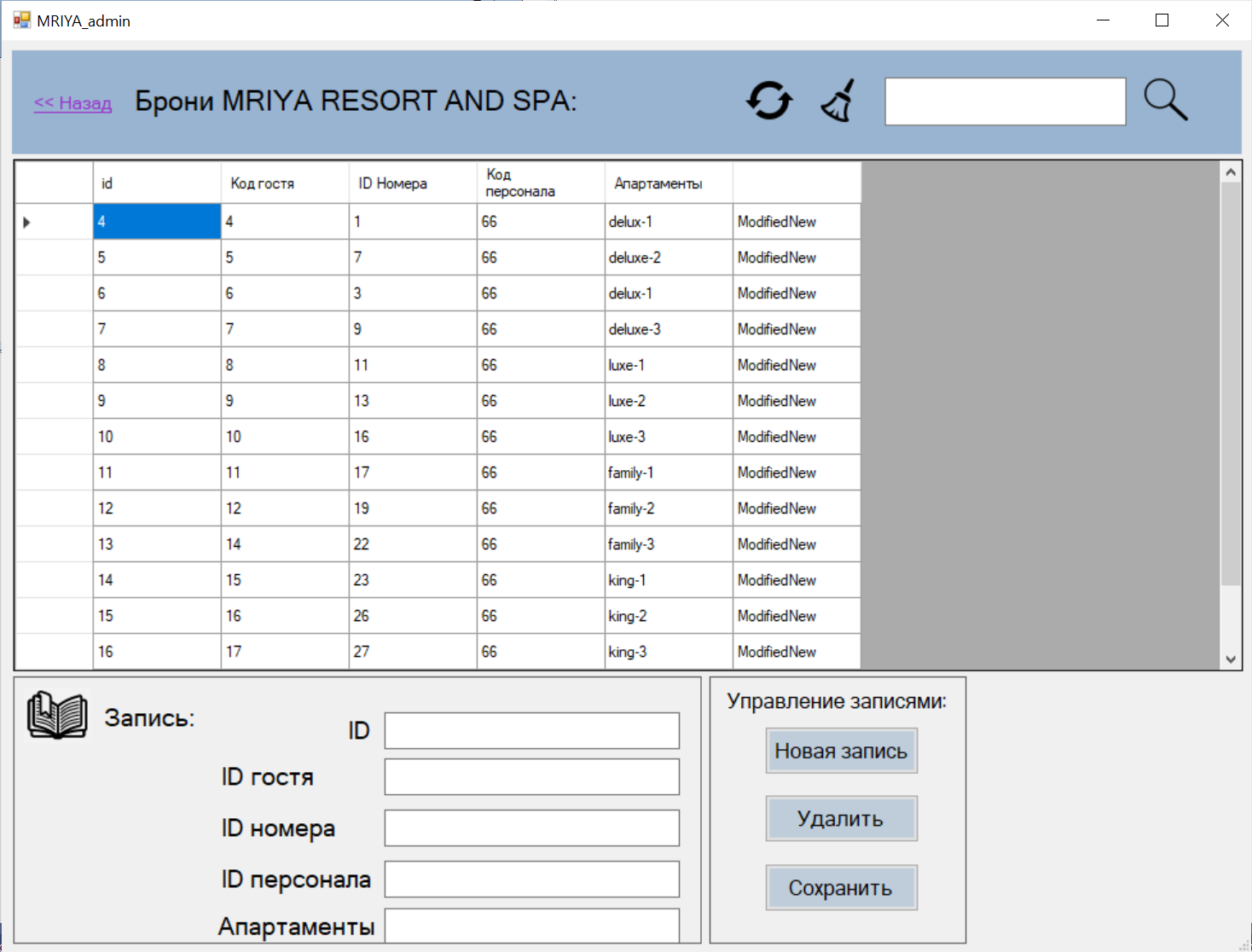


Рисунок 27 – Окно администратора отеля MRIYA

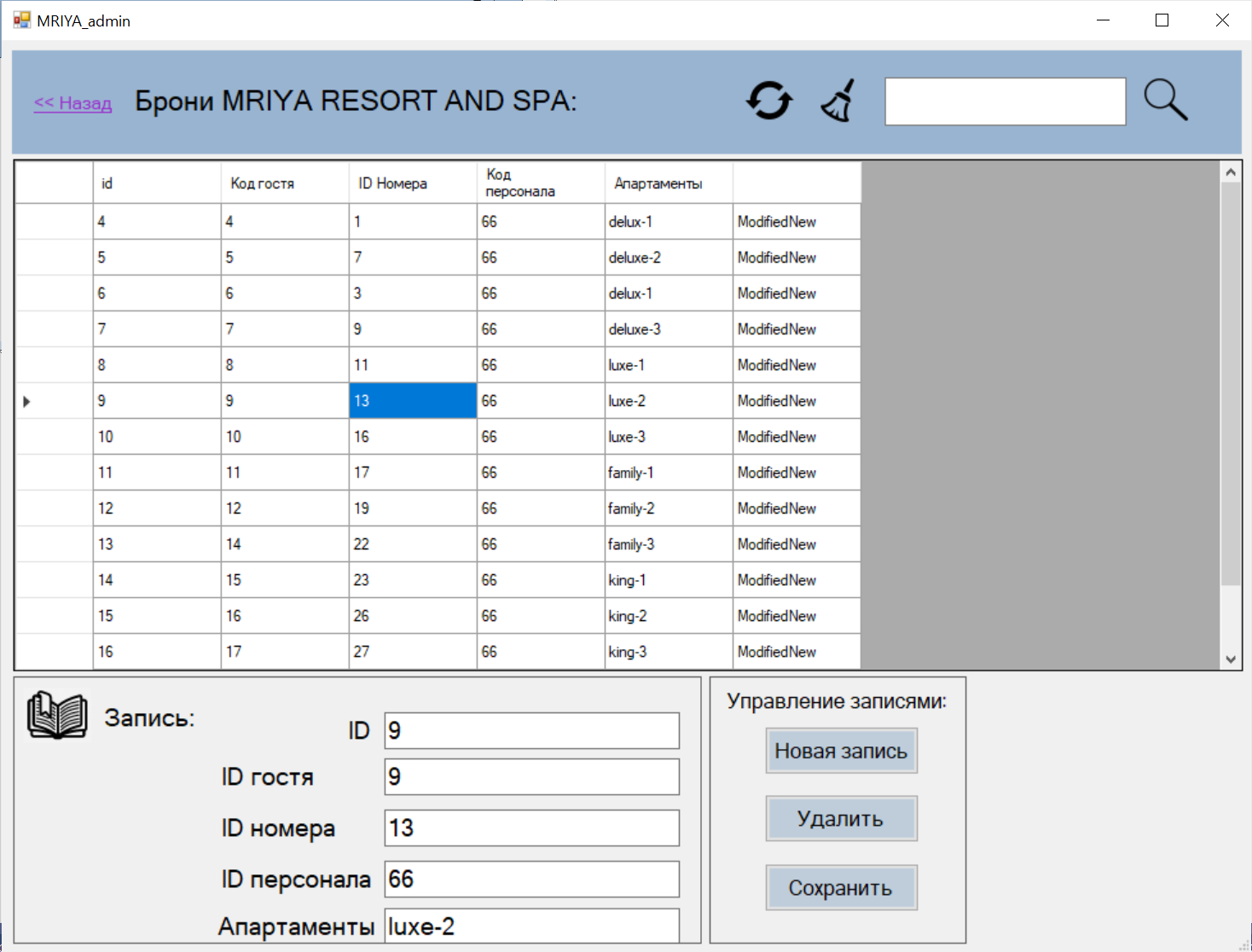


Рисунок 28 – Занесение данных в ячейки при нажатии

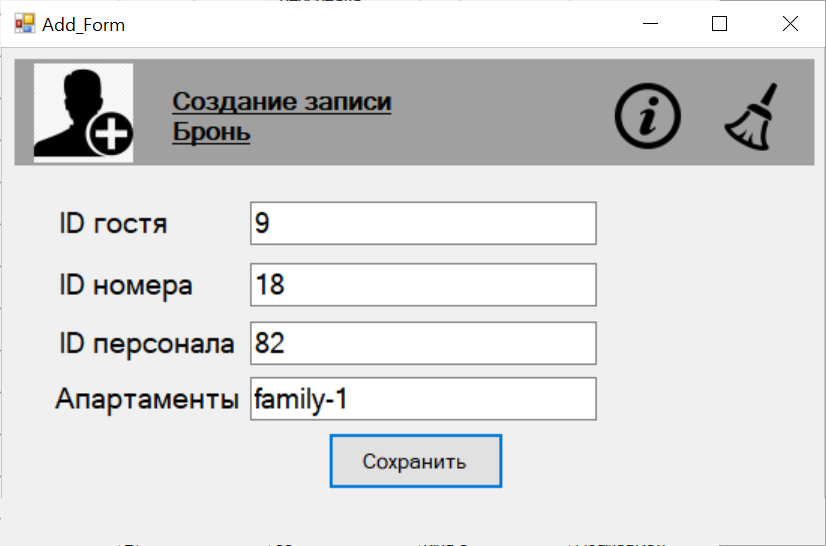
****

Рисунок 29 – Создание новой записи

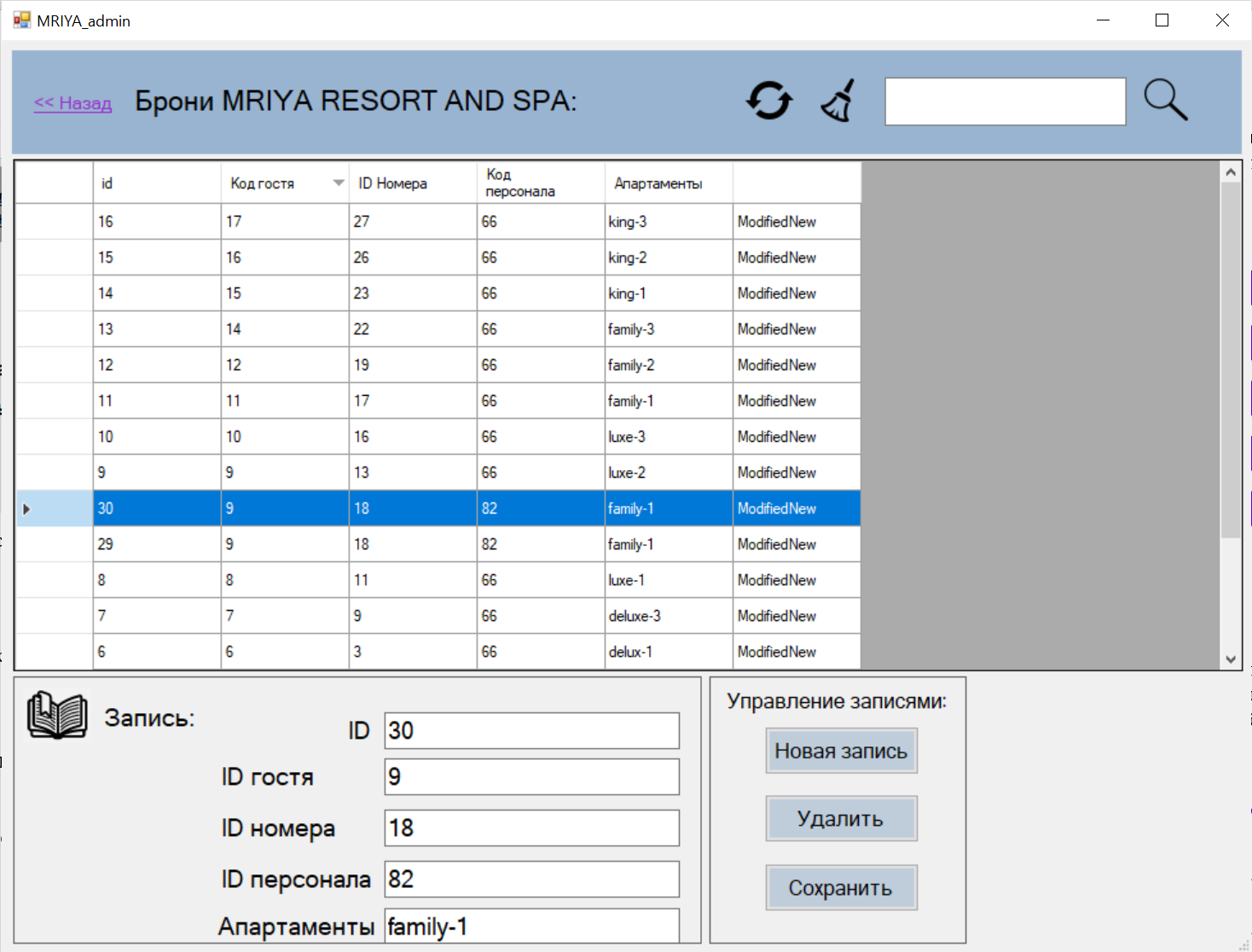


Рисунок 30 – Подтверждение существования новой записи

Таким образом администратору доступен весь функционал приложения по манипуляции над БД.

## 4.4 Тестирование системы

Клиентское приложение запускается с экрана авторизации.

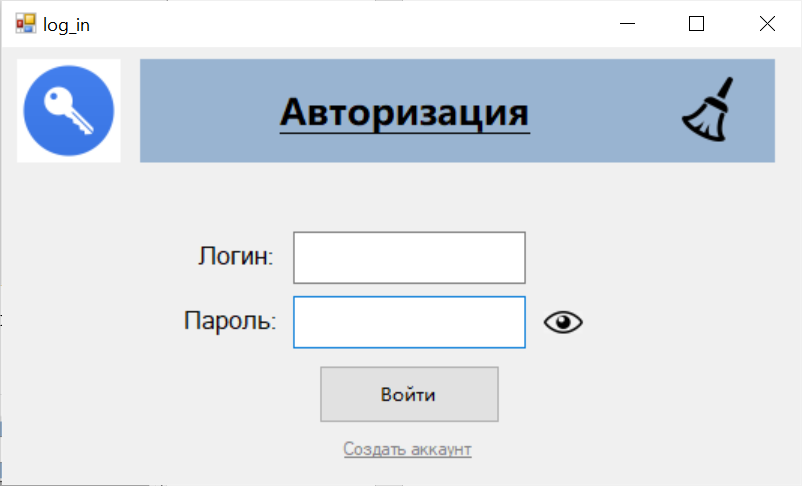


Рисунок 31 ­ Экран авторизации

В случае отсутствия аккаунта можно нажать на «Создать аккаунт» и перейти в окно регистрации. Доступные логины только guest или admin. Для того чтобы зарегистрироваться как администратор необходимо знак секретный код(1234).

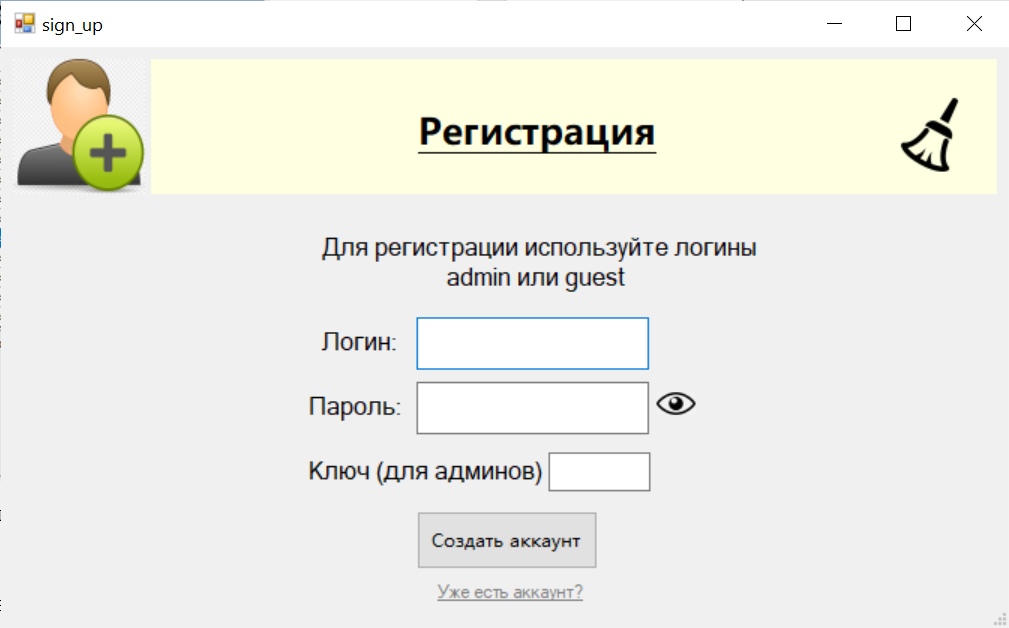


Рисунок 32 ­ Экран регистрации

Для доступа к полному функционалу приложения была выбрана роль администратора. На следующем экране происходит выбор таблицы.

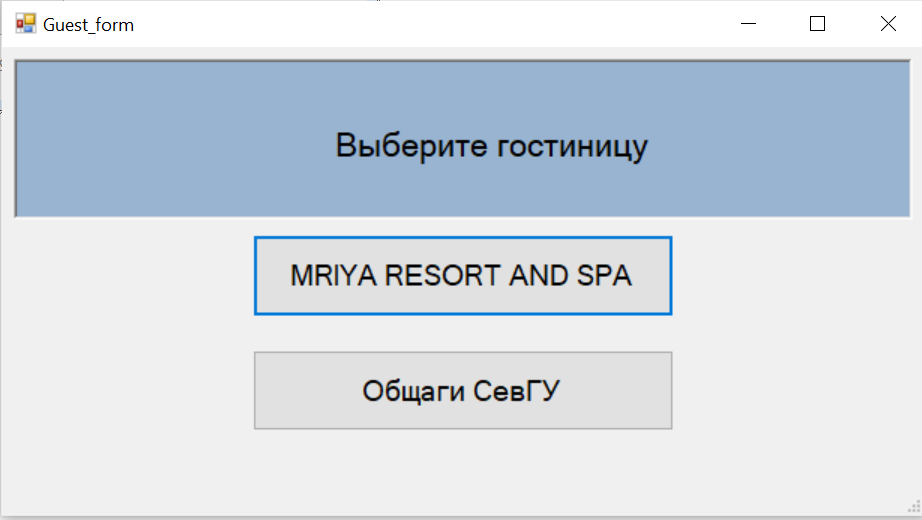


Рисунок 33 – Выбор таблицы

# **ЗАКЛЮЧЕНИЕ**

В соответствии с вариантом задания разработано клиентское приложение с применением базы данных Microsoft SQL Server Management Studio. В приложении реализованы пользовательский интерфейс, разграничение прав доступа, возможность добавления, удаления, просмотра, поиска записей и создание необходимых запросов. Для удобства пользования программой предусмотрены сообщения об ошибках при попытке ввести неверные данные при регистрации или авторизации, кнопки очистки текст боксов. В ходе выполнения расчетно-графической работы были закреплены и применены на практике знания полученные в ходе изучения дисциплины управления данными. Разработанная программа соответствует поставленной задаче

Таким образом, была достигнута цель расчетно-графической работы – систематизированы, закреплены и углублены знания управления данными при создании БД и написании программ с их использованием. Результаты проектирования рекомендуется использовать для изучения языка, повторения или улучшения собственных способностей в программировании. В качестве полноценного клиентского приложения можно использовать только в случае доработки.

# **ЛИТЕРАТУРНЫЕ ИСТОЧНИКИ**

1) ГОСТ Р ИСО/МЭК ТО 10032-2007. Эталонная модель управления данными [Текст]. - Введ. 27.12.2007. - М.: Стандартинформ, 2009. - 40 с.

2) Баженова И.Ю. Разработка распределенных приложений баз данных: Курс лекций. - М.: МГУ им. М.В. Ломоносова, 2006. - 203 с.

3) Баканов В.М. Введение в язык SQL запросов к базам данных: Учебное пособие. - М.: МГАПИ, 2002. - 61 с.: ил.

4) Бураков П.В., Петров В.Ю. Введение в системы баз данных: Учебное пособие. - СПб: СПбГУ ИТМО, 2010. - 128 с.

5) Гудов А.М., Шмакова Л.Е. Введение в язык структурированных запросов SQL: Учебное пособие. - Кемерово, Кемеровский госуниверситет, 2001. - 118 с.

6) Зиборов В.В. MS Visual C++ 2010 в среде .NET. Библиотека программиста. - Спб.: Питер, 2012. - 320 с.: ил.

7) Википедия - свободная энциклопедия [Электронный ресурс]. - URL: https://ru.wikipedia.org/wiki/SQL, свободный доступ.

8) Базы данных URL: http://www.site-do.ru/db/db1.php, свободный доступ.

9) Donald Bales. Beginning Oracle PL/SQL. Second edition. - New York: Apress, 2015. -

10) Шварц Б., Зайцев П., Ткаченко В., Заводны Дж., Ленц А., Бэллинг Д. MySQL. Оптимизация производительности, 2-е издание. - Пер. с англ. - Спб.: Символ-Плюс, 2010. - 832 с.

# **ПРИЛОЖЕНИЕ А Исходный код программы**

Add\_Form

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;

using System.Diagnostics;

using System.Runtime.ConstrainedExecution;

namespace test\_Database

{

public partial class Add\_Form : Form

{

string[] appart\_names = { "delux-1", "deluxe-2", "deluxe-3","luxe-1", "luxe-2", "luxe-3", "family-1", "family-2", "family-3", "king-1", "king-2", "king-3","№1-101", "№1-102","№2-101"};

Database database = new Database();

public Add\_Form()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void button\_save\_Click(object sender, EventArgs e)

{

database.openConnection();

bool right = false;

int guest\_id=-1;

int number\_id=-1;

int personnel\_id=-1;

var appart\_name = textBox\_appart\_name.Text;

if (int.TryParse(textBox\_number\_id.Text, out number\_id) && int.TryParse(textBox\_guest\_id.Text, out guest\_id) && int.TryParse(textBox\_personnel\_id.Text, out personnel\_id))

{

if(guest\_id>=4 && guest\_id<=22 && number\_id>=1 && number\_id<=31 && personnel\_id>=64 && personnel\_id<=93)

{

for (int i = 0; i<appart\_names.Length; i++)

if(appart\_name==appart\_names[i])

right = true;

}

}

if (right)

{

var addQuery = $"insert into reservation(guest\_id,number\_id,personnel\_id,appart\_name) values ({guest\_id},{number\_id},{personnel\_id},'{appart\_name}')";

var command = new SqlCommand(addQuery, database.getConnection());

command.ExecuteNonQuery();

MessageBox.Show("Запись успешно создана!", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Запись не создана!\nВведите существующие данные", "", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

database.closeConnection();

}

}

}

Admin\_from

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 test\_Database

{

public partial class Admin\_form : Form

{

Database database = new Database();

public Admin\_form()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void button\_mriya\_admin\_Click(object sender, EventArgs e)

{

MRIYA\_admin frm\_mriya\_admin = new MRIYA\_admin();

this.Hide();

frm\_mriya\_admin.ShowDialog();

}

private void button\_sevsu\_admin\_Click(object sender, EventArgs e)

{

Sevsu\_admin frm\_sevsu\_admin = new Sevsu\_admin();

this.Hide();

frm\_sevsu\_admin.ShowDialog();

}

}

}

Log\_in

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;

using System.Diagnostics;

using System.Runtime.ConstrainedExecution;

namespace test\_Database

{

public partial class log\_in : Form

{

Database database = new Database();

public log\_in()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void log\_in\_Load(object sender, EventArgs e)

{

// textBox\_password.PasswordChar = '\*';

pictureBox3.Visible = false;

textBox\_login.MaxLength = 50;

textBox\_password.MaxLength = 50;

}

private void btnEnter\_Click(object sender, EventArgs e)

{

var loginUser = textBox\_login.Text;

var passUser = textBox\_password.Text;

SqlDataAdapter adapter = new SqlDataAdapter();

DataTable table = new DataTable();

string querystring = $"select id\_user, login\_user, password\_user from register where login\_user = '{loginUser}' and password\_user = '{passUser}'";

SqlCommand command = new SqlCommand(querystring, database.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count == 1)

{

MessageBox.Show("Вы успешно вошли!", "Успешно", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Hide();

if(loginUser == "admin")

{

Admin\_form frm\_admin = new Admin\_form();

frm\_admin.ShowDialog();

}

else

{

Guest\_form frm\_guest = new Guest\_form();

frm\_guest.ShowDialog();

}

this.Show();

}

else

MessageBox.Show("Такого аккаунта не существует!", "Аккаунта не существует!", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

sign\_up frm\_sign = new sign\_up();

this.Hide();

frm\_sign.ShowDialog();

}

private void btnClear\_Click(object sender, EventArgs e)

{

textBox\_login.Text = "";

textBox\_password.Text = "";

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

textBox\_password.UseSystemPasswordChar = false;

pictureBox3.Visible = false;

pictureBox4.Visible = true;

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

textBox\_password.UseSystemPasswordChar = true;

pictureBox3.Visible = true;

pictureBox4.Visible = false;

}

private void label1\_Click(object sender, EventArgs e)

{

}

}

}

Sign\_up

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 test\_Database

{

public partial class sign\_up : Form

{

Database dataBase = new Database();

public sign\_up()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void sign\_up\_Load(object sender, EventArgs e)

{

// textBox\_password2.PasswordChar = '\*';

pictureBox3.Visible = false;

textBox\_login2.MaxLength = 50;

textBox\_password2.MaxLength = 50;

}

private void btnCreate\_Click(object sender, EventArgs e)

{

if(checkUser() )

{

return;

}

var login = textBox\_login2.Text;

var password = textBox\_password2.Text;

var kod = textBox\_kod.Text;

if(login == "" || password == "")

{

MessageBox.Show("Поля не должны быть пустыми!");

return;

}

if (login == "admin" && kod == "1234"|| login == "guest")

{

string querystring = $"insert into register(login\_user,password\_user) values('{login}','{password}')";

SqlCommand command = new SqlCommand(querystring, dataBase.getConnection());

dataBase.openConnection();

if(command.ExecuteNonQuery() == 1 )

{

MessageBox.Show("Аккаунт успешно создан!","Успех!");

log\_in frm\_login = new log\_in();

frm\_login.ShowDialog();

}

else

{

MessageBox.Show("Аккаунт не создан!");

}

dataBase.closeConnection();

}

else

{

if (login == "admin")

{

MessageBox.Show("Введите Код Админа\n или используйте логин guest ");

}

else

{

MessageBox.Show("Допустимые логины: admin или guest");

}

}

}

private Boolean checkUser()

{

var loginUser = textBox\_login2.Text;

var passUser = textBox\_password2.Text;

SqlDataAdapter adapter = new SqlDataAdapter();

DataTable table = new DataTable();

string querystring = $"select id\_user, login\_user,password\_user from register where login\_user = '{loginUser}' and password\_user = '{passUser}'";

SqlCommand command = new SqlCommand(querystring, dataBase.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

if(table.Rows.Count > 0 )

{

MessageBox.Show("Пользователь уже существует!");

return true;

}

else

{

return false;

}

}

private void btn\_Clear2\_Click(object sender, EventArgs e)

{

textBox\_login2.Text = "";

textBox\_password2.Text = "";

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

textBox\_password2.UseSystemPasswordChar = false;

pictureBox3.Visible = false;

pictureBox4.Visible = true;

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

textBox\_password2.UseSystemPasswordChar = true;

pictureBox3.Visible = true;

pictureBox4.Visible = false;

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

log\_in frm\_log = new log\_in();

this.Hide();

frm\_log.ShowDialog();

}

}

}

Guest\_form

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 test\_Database

{

public partial class Guest\_form : Form

{

public Guest\_form()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void button\_mriya\_Click(object sender, EventArgs e)

{

MRIYA frm\_mriya = new MRIYA();

this.Hide();

frm\_mriya.ShowDialog();

}

private void button\_sevsu\_Click(object sender, EventArgs e)

{

Sevsu frm\_sevsu = new Sevsu();

this.Hide();

frm\_sevsu.ShowDialog();

}

private void Guest\_form\_Load(object sender, EventArgs e)

{

}

}

}

MRIYA

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace test\_Database

{

public partial class MRIYA : Form

{

Database database = new Database();

public MRIYA()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void link\_back\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Guest\_form frm\_guest = new Guest\_form();

this.Hide();

frm\_guest.ShowDialog();

}

private void CreateColumns()

{

dataGridView1.Columns.Add("id", "id");

dataGridView1.Columns.Add("guest\_id", "Код гостя");

dataGridView1.Columns.Add("number\_id", "ID Номера");

dataGridView1.Columns.Add("personnel\_id", "Код персонала");

dataGridView1.Columns.Add("appart\_name", "Апартаменты");

dataGridView1.Columns.Add("IsNew", String.Empty);

}

private void ReadSingleRow(DataGridView dgw, IDataRecord record)

{

dgw.Rows.Add(record.GetInt32(0), record.GetInt32(1), record.GetInt32(2), record.GetInt32(3), record.GetString(4), RowState.ModifiedNew);

}

private void RefreshDataGrid(DataGridView dgw)

{

dgw.Rows.Clear();

string queryString = $"select \* from reservation where number\_id!=29 and number\_id!=30 and number\_id!=31;";

SqlCommand command = new SqlCommand(queryString, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ReadSingleRow(dgw, reader);

}

reader.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void MRIYA\_Load(object sender, EventArgs e)

{

CreateColumns();

RefreshDataGrid(dataGridView1);

}

private void Search(DataGridView dgw)

{

dgw.Rows.Clear();

string searchString = $"select \* from reservation where concat(guest\_id,number\_id,personnel\_id,appart\_name) like '%" + textBox\_search.Text + "%'";

SqlCommand com = new SqlCommand(searchString, database.getConnection());

database.openConnection();

SqlDataReader read = com.ExecuteReader();

while (read.Read())

{

ReadSingleRow(dgw, read);

}

read.Close();

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

Search(dataGridView1);

}

}

}

MRIYA\_admin

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 test\_Database

{

enum RowState

{

Existed,

New,

ModifiedNew,

Deleted

}

public partial class MRIYA\_admin : Form

{

Database database = new Database();

public MRIYA\_admin()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void MRIYA\_admin\_Load(object sender, EventArgs e)

{

CreateColumns();

RefreshDataGrid(dataGridView1);

}

private void CreateColumns()

{

dataGridView1.Columns.Add("id","id");

dataGridView1.Columns.Add("guest\_id", "Код гостя");

dataGridView1.Columns.Add("number\_id", "ID Номера");

dataGridView1.Columns.Add("personnel\_id", "Код персонала");

dataGridView1.Columns.Add("appart\_name", "Апартаменты");

dataGridView1.Columns.Add("IsNew",String.Empty);

}

private void ClearFields()

{

textBox\_id.Text = "";

textBox\_guest\_id.Text = "";

textBox\_number\_id.Text = "";

textBox\_personnel\_id.Text = "";

textBox\_appart\_name.Text = "";

}

private void ReadSingleRow(DataGridView dgw, IDataRecord record)

{

dgw.Rows.Add(record.GetInt32(0),record.GetInt32(1), record.GetInt32(2),record.GetInt32(3),record.GetString(4),RowState.ModifiedNew);

}

private void RefreshDataGrid(DataGridView dgw)

{

dgw.Rows.Clear();

string queryString = $"select \* from reservation where number\_id!=29 and number\_id!=30 and number\_id!=31;";

SqlCommand command = new SqlCommand(queryString, database.getConnection());

database.openConnection();

SqlDataReader reader = command.ExecuteReader();

while(reader.Read())

{

ReadSingleRow (dgw,reader);

}

reader.Close();

}

private void link\_back\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Admin\_form frm\_admin = new Admin\_form();

this.Hide();

frm\_admin.ShowDialog();

}

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

var selectedRow = e.RowIndex;

if(selectedRow >= 0)

{

DataGridViewRow row = dataGridView1.Rows[selectedRow];

textBox\_id.Text = row.Cells[0].Value.ToString();

textBox\_guest\_id.Text = row.Cells[1].Value.ToString();

textBox\_number\_id.Text = row.Cells[2].Value.ToString();

textBox\_personnel\_id.Text = row.Cells[3].Value.ToString();

textBox\_appart\_name.Text = row.Cells[4].Value.ToString();

}

}

private void picture\_update\_Click(object sender, EventArgs e)

{

RefreshDataGrid(dataGridView1);

ClearFields();

}

private void button\_new\_Click(object sender, EventArgs e)

{

Add\_Form addfrm = new Add\_Form();

addfrm.Show();

}

private void Search(DataGridView dgw)

{

dgw.Rows.Clear();

string searchString = $"select \* from reservation where concat(guest\_id,number\_id,personnel\_id,appart\_name) like '%" + textBox\_search.Text + "%'";

SqlCommand com = new SqlCommand(searchString,database.getConnection());

database.openConnection();

SqlDataReader read = com.ExecuteReader();

while(read.Read())

{

ReadSingleRow(dgw, read);

}

read.Close();

}

private void textBox\_search\_TextChanged(object sender, EventArgs e)

{

Search(dataGridView1);

}

private void deleteRow()

{

int index = dataGridView1.CurrentCell.RowIndex;

dataGridView1.Rows[index].Cells[5].Value = RowState.Deleted;

}

private void button1\_Click(object sender, EventArgs e)

{

deleteRow();

ClearFields();

}

private void Updatetable()

{

database.openConnection();

for(int index =0; index < dataGridView1.Rows.Count; index++)

{

var rowState = (RowState)dataGridView1.Rows[index].Cells[5].Value;

if (rowState == RowState.Existed)

continue;

if(rowState == RowState.Deleted)

{

var id = Convert.ToInt32(dataGridView1.Rows[index].Cells[0].Value);

var deleteQuery = $"delete from reservation where id = {id}";

var command = new SqlCommand(deleteQuery, database.getConnection());

command.ExecuteNonQuery();

}

}

database.closeConnection();

}

private void button\_save\_Click(object sender, EventArgs e)

{

Updatetable();

}

private void picture\_Clean\_Click(object sender, EventArgs e)

{

ClearFields();

}

}

}

Sevsu

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 test\_Database

{

public partial class Sevsu : Form

{

public Sevsu()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void link\_back\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Guest\_form frm\_guest = new Guest\_form();

this.Hide();

frm\_guest.ShowDialog();

}

private void Sevsu\_Load(object sender, EventArgs e)

{

}

}

}

Sevsu\_admin

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 test\_Database

{

public partial class Sevsu\_admin : Form

{

public Sevsu\_admin()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void Sevsu\_admin\_Load(object sender, EventArgs e)

{

}

private void link\_back\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Admin\_form frm\_admin = new Admin\_form();

this.Hide();

frm\_admin.ShowDialog();

}

}

}