**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙФЕДЕРАЦИИ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПБПУ»)

**Институт среднего профессионального образования**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отделение информационных технологий

Специальность: 09.02.07 Информационные системы и программирование

Дисциплина: МДК 11.01 Технологии разработки и защиты баз данных

Группа: 32919/3

**КУРСОВОЙ ПРОЕКТ**

**на тему:**

**«Проектирование и реализация базы данных отеля»**

Студент Пешенкова Альбина Валентиновна

(ФИО) (подпись)

Руководитель Тузова Диана Анатольевна

(ФИО) (подпись)

Оценка

(оценка) (подпись) (ФИО

Санкт - Петербург

2021

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙФЕДЕРАЦИИ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПБПУ»)

**Институт среднего профессионального образования**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отделение информационных технологий

**Задание на курсовое проектирование**

**по МДК 11.01 «Технологии разработки и защиты баз данных»**

Специальность: 09.02.07 Информационные системы и программирование

**Студенту** Пешенковой А.В. группа\_\_32919/3\_\_\_\_

(фамилия, инициалы)

**Тема:** **Проектирование и реализация** **базы данных отеля**

**Исходные данные к проекту:**

1. Литература по описанию объекта информатизации
2. Литература по инструментальным средствам разработки прикладных программ
3. Лекции, стандарты.
4. Интернет – сайты по предметной области

**Перечень подлежащих разработке вопросов:**

1. Аналитический обзор программных средств, технологий, аналогов ПП для создания базы данных и интерфейсов пользователей.
2. Проектирование реляционной базы данных (инфологическая и даталогическая модели с указанием связей (с пояснениями) между таблицами).
3. Разработка механизмов управления данными в БД (добавление, удаление и обновление

данных).

1. Организация выборки информации из разработанной базы данных (сформулировать не менее 5 запросов всех типов, реализуемых средствами выбранной СУБД).
2. Проработка технологии доступа к базе данных (определение круга пользователей базы данных и оценка возможности разграничения полномочий пользователей).
3. Разработка ПО для решения поставленной задачи.
4. Разработка алгоритмов программных модулей.
5. Тестирование работы приложения.
6. Оформление документации (руководство системного программиста).
7. Оформление пояснительной записки.

**Руководитель Д.А. Тузова**

(подпись, дата) (инициалы, фамилия)

**Задание принял к выполнению:**

**Студент А.В. Пешенкова**

(подпись, дата) (инициалы, фамилия)

Содержание

[ВВЕДЕНИЕ 4](#_Toc88507355)

[Глава 1. Сравнительная характеристика СУБД 5](#_Toc88507356)

[1.1 MySQL 5](#_Toc88507357)

[1.2 Oracle Database 6](#_Toc88507358)

[1.3 Microsoft SQL Server 8](#_Toc88507359)

[Глава 2. Проектирование БД 11](#_Toc88507360)

[2.1 Описание предметной области 11](#_Toc88507361)

[2.2 Выделение сущностей 12](#_Toc88507362)

[2.3 Описание связей 13](#_Toc88507363)

[2.4 ER-модель 14](#_Toc88507364)

[2.5 Создание БД в СУБД MS SQL Server 14](#_Toc88507365)

[2.6 Заполнение БД тестовыми значениями 15](#_Toc88507366)

[Глава 3. Работа с БД 21](#_Toc88507367)

[3.1. Создание запросов 21](#_Toc88507368)

[3.2 Интерфейс 25](#_Toc88507369)

[ЗАКЛЮЧЕНИЕ 41](#_Toc88507370)

[Приложение](#_Toc88507371)

# ВВЕДЕНИЕ

База данных - важнейший компонент любой информационной системы. Она позволяет структурировано хранить большие объемы информации конкретного предприятия, что значительно рационализирует ведение отчетов и создание архивов. Оптимизированные базы данных значительно увеличивают производительность, построенных на их использовании, программ.

Темой данного курсового проекта является «Проектирование и реализация базы данных отеля» для автоматизации деятельности и обеспечения хранения, накопления и предоставления информации.

Целью курсовой работы являются закрепление практических умений, полученных на лабораторных занятиях, получение дополнительных навыков обследования предметной области, концептуального, логического и физического проектирования базы данных, освоение средств поддержания целостности БД, формирования разнообразных запросов

Задачами курсового проекта являются проектирование и создание прикладной программы, обеспечивающей создание и ведение базы данных по выбранной предметной области. На примере этой программы должны быть показаны примеры реализации вопросов, указанных выше, а также связанных с конкретной СУБД и операционной системой.

# Глава 1. Сравнительная характеристика СУБД

## MySQL

СУБД MySQL - разработка шведской компании MySQL AB, разработку и поддержку которой в настоящий момент осуществляет корпорация Oracle. Данная СУБД является программным обеспечением с открытым исходным кодом, распространяемым по лицензии GNU (GPL) и коммерческой лицензии для ситуаций, не подпадающих под действие лицензии GPL. MySQL поддерживает реляционную модель данных, т. е. представляет собой реляционную СУБД.

В версии MySQL 4.1 появились такие нововведения, как полная поддержка вложенных запросов и поддержка транзакций. В MySQL 5.0 стали доступными следующие важные механизмы:

* хранимые процедуры и функции;
* триггеры;
* представления;
* курсоры;
* информационная схема, т. е. переносимый набор представлений системной таблицы, в которой хранится разнообразная внутренняя информация;
* обработчики ошибок.

Подходит для организаций, которым требуется надежный инструмент управления базами данных, но бесплатный.

Основные достоинства СУБД MySQL:

* устойчивость работы;
* простота в использовании;
* считается одной из самых быстрых СУБД;
* масштабируемость;
* открытый код доступен для просмотра и модернизации, что позволяет постоянно улучшать программный продукт;
* разработана с использованием языков C/C++, протестирована на многих платформах, среди которых Windows, Linux, FreeBSD, Mac OS X, OS/2, Solaris и др;
* поддерживает API (Application Programming Interface, программный интерфейс приложения) для С, C++, Eiffel, Java, Perl, PHP, Python, Ruby и Tcl. Можно успешно применять как для построения Web-страниц с использованием Perl, PHP и Java, так и для работы прикладной программы, созданной с использованием Delphi, Builder C++ или платформы .NET;
* широкий выбор типов таблиц, в том числе и сторонних разработчиков;
* хорошо документирована.

Недостатки СУБД MySQL:

* низкая скорость разработки. Как и многим другим программным продуктам с открытым кодом, MySQL не достает некоторого технического совершенства, что порой сказывается на эффективности процессов разработки;
* отсутствует встроенная поддержка XML или OLAP;
* недостаточная надежность. В вопросах надежности некоторых процессов по работе с данными (например, связь, транзакции, аудит) MySQL уступает некоторым другим СУБД;
* платная поддержка даже для бесплатной версии.

## Oracle Database

На мировом рынке корпоративных СУБД Oracle занимает одно из доминирующих положений. По статистическим данным на рынке России лидирующее положение занимает Oracle, так как по статистическим данным за 2010 год, данная СУБД занимает более 60% всего рынка, среди других СУБД и около 30% мирового рынка СУБД.

Одной из основных характеристик СУБД Oracle является функционирование системы на большинстве платформ. В том числе на больших ЭВМ, UNIX-серверах, персональных компьютерах и т. д.

Другой важной характеристикой является поддержка Oracle всех возможных вариантов архитектур, в том числе симметричных многопроцессорных систем, кластеров, систем с массовым параллелизмом и т. д. Очевидна значимость этих характеристик для крупномасштабных организаций, где эксплуатируется множество компьютеров различных моделей и производителей. Возможность переноса Oracle обеспечивается специфической структурой исходного программного кода сервера баз данных. Приблизительно 80% программного кода Oracle — это программы на языке программирования C, является платформо-независимым. Примерно 20% кода, представляющее собой ядро сервера, реализовано на машинно-зависимых языках; и эта часть кода, разумеется, переписывается для различных платформ.

Одна из отличительных особенностей сервера Oracle — возможность хранения и обработки различных типов данных. Данная функциональность интегрирована в ядро СУБД и поддерживается модулем interMedia в составе Oracle Database. Он обеспечивает работу с текстовыми документами, включая различные виды поиска, в том числе контекстного; работу с графическими образами более 20–ти форматов; работу с аудио и видеоинформацией. СУБД Oracle также позволяет за счет использования Object Option конструировать новые типы данных со спецификацией методов доступа к ним.

Подходит для крупных организаций, которые работают с огромными базами данных и разнообразными функциями.

Преимущества СУБД Oracle:

* самые свежие инновации и впечатляющий функционал уже внедрены в этом продукте, поскольку компания Oracle стремится держать планку даже на фоне других разработчиков СУБД;
* СУБД от Oracle является крайне надёжной, фактически это эталон надёжности среди подобных систем;
* Oracle поддерживает самые большие базы данных. Большое количество пользователей для этой системы также не помеха. СУБД способна поддерживать любых пользователей, в любом количестве, которые при этом одновременно выполняют разные задачи;
* СУБД Oracle хорошо обрабатывает транзакции. Система сохраняет высокую производительность, в результате чего пользователи не страдают от низкой скорости обработки.

Недостатки СУБД Oracle:

* высокая стоимость, особенно для небольших организаций;
* система может потребовать значительных ресурсов уже сразу после установки, поэтому возможно потребуется модернизировать оборудование для внедрения Oracle;
* Oracle не просто настроить. Крупные компании, платящие деньги, обычно постоянно нанимают администраторов Oracle на сотрудников;
* большинство документов строго из Oracle – в документации по веб-интерфейсу/справки/примерах очень мало.

## Microsoft SQL Server

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

Движок SQL Server работает на облачных серверах, а также локальных серверах, причем можно комбинировать типы применяемых серверов одновременно. SQL Server поддерживает зеркалирование и кластеризацию баз данных. Кластер сервера SQL — это совокупность одинаково конфигурированных серверов; такая схема помогает распределить рабочую нагрузку между несколькими серверами.

Вскоре после выпуска Microsoft SQL сервер 2016, Microsoft адаптировала продукт для операционной системы Linux, а на Windows-платформе он работал изначально.

Одной из уникальных особенностей версии 2016 года является temporal data support (временная поддержка данных), которая позволяет отслеживать изменения данных с течением времени. Последняя версия Microsoft SQL-сервер поддерживает dynamic data masking (динамическую маскировку данных), которая гарантирует, что только авторизованные пользователи будут видеть конфиденциальные данные.

Подходит для крупных организаций, которые уже используют ряд продуктов Microsoft.

Преимущества СУБД MS SQL Server:

* масштабируется, поэтому работать с ней можно на портативных ПК или мощной мультипроцессорной технике. Процессор может одновременно обрабатывать большой объем запросов;
* данные извлекаются быстро. Система позволяет обрабатывать транзакции в интерактивном режиме, есть динамическая блокировка;
* прост в использовании;
* работает быстро и стабильно;
* хорошо взаимодействует с другими продуктами Microsoft;
* рутинные административные задачи автоматизированы: это управление блокировками, памятью, редактура размеров файлов. У системы продуманы настройки, можно создать профили пользователей;
* интегрирован сервер интерактивного анализа для принятия решений, создания корпоративных отчетов. Есть службы преобразования информации.

Недостатки СУБД MS SQL Server:

* даже при тщательной настройке производительности корпорация SQL Server способен занять все доступные ресурсы;
* проблемы с использованием службы интеграции для импорта файлов;
* высокая цена.

Анализируя полученную информацию о различных СУБД, делаем вывод, что бесплатность, хорошая совместимость с продукцией Microsoft, а также простота в использовании делает MS SQL Server наилучшим выбором для выполнения данной работы.

# Глава 2. Проектирование БД

## 2.1 Описание предметной области

Необходимо разработать базу данных, а также приложение для работы с ней, для отеля «Гранд Отель Чайковский». В базе будут определены два уровня доступа: для администратора и сотрудников отеля, причём доступ сотрудников будет отличаться в зависимости от отдела, в котором они работают.

Сотрудники при приёме на работу определяются на должность, относящуюся к одному из отделов отеля.

В отеле присутствует три отдела, каждый из которых заведует определённой сферой работы. Управляющий отдел занимается общим управлением работой отеля, наймом кадров, заключением сделок со сторонними организациями, закупками. Отдел по работе с клиентами занимается регистрацией гостей отеля, консультированием по вопросам проживания, бронированием номеров и помощью гостям при заселении в номера. Поддерживающий отдел занимается ремонтом и уборкой отеля. Также в качестве подразделения поддерживающего отдела присутствует кухня, которая осуществляет приготовление трёх приёмов пищи и готовит заказы посетителей ресторана отеля в течение дня.

Гости имеют возможность заранее забронировать номер, позвонив администратору из отдела по работе с клиентами, или же занять свободные на данный момент номера, зарегистрировавшись на стойке администрации. При регистрации гостя в базу заносятся его личные данные, выбранный тип питания, номер отеля, а также дата выселения.

В случае поломки в номере гость может обратиться к администратору. Он составит запрос на ремонт, который будет отправлен в поддерживающий отдел для выполнения ремонтных работ.

Также отелю требуется закупать продукты питания для кухни, канцелярию сотрудникам, расходные материалы в номера отеля и многое другое. Все эти закупки осуществляются управляющим отделом.

Отель находится по адресу улица Чайковского 55, Центральный район, Санкт-Петербург, Россия.

В результате БД должна обеспечить решение следующих задач:

* заселение гостей;
* осуществление ремонта;
* осуществление питания;
* осуществление закупок.

## 2.2 Выделение сущностей

После анализа ПО, можем выделить следующие сущности:

* Сотрудник;
* Должности;
* Управляющий отдел;
* Отдел по работе с клиентами;
* Поддерживающий отдел;
* Закупки;
* Клиенты;
* Номера отеля;
* Запросы на ремонт;
* Кухня;
* Расписание питания.

Перечень атрибутов выделенных сущностей

* Сотрудник: id, фамилия, имя, отчество, дата рождения, паспортные данные, дата устройства, должность;
* Должность: Название позиции, количество ставок, отделение, месячная ставка;
* Отдел по работе с клиентами: сотрудник, должность;
* Гости: фамилия, имя, отчество, паспортные данные, дата рождения, номер телефона, номер отеля, количество человек, тип питания, дата заселения, дата выселения, сумма к оплате, администратор;
* Номер отеля: номер, число мест, класс, стоимость за сутки;
* Управляющий отдел: сотрудник, должность;
* Заказ: id, товар, количество, цена, ответственный, дата;
* Поддерживающий отдел: сотрудник, должность;
* Запрос на ремонт: id, сотрудник, дата, номер в отеле;
* Кухня: сотрудник, место работы, управляющий;
* Расписание питания: приём пищи, время начала, время окончания, ответственный.

Перечень необходимых ограничений:

* Паспортные данные должны иметь ограничение по количеству символов и иметь пробел, разделяющий серию и номер паспорта;
* Номер телефона должен иметь вид +7(000) 000-00-00.

## 2.3 Описание связей

* Сотрудники принимаются на должность. Связь Должность (название позиции) и Сотрудник (должность). Связь один ко многим;
* Сотрудники распределяются по отделам. Связи один к одному между таблицами Сотрудник и Отдел по работе с клиентами, Сотрудник и Управляющий отдел, Сотрудник и Поддерживающий отдел;
* Администратор регистрирует гостей. Связь один ко многим между Отдел по работе с клиентами (сотрудник) и Гости (администратор);
* Гости заселяются в номера отеля. Связь один ко многим между Номера отеля (номер) и Гости (номер отеля);
* Управляющий отдел осуществляет заказы. Связь один ко многим между Управляющий отдел (сотрудник) и Заказ (ответственный);
* Поддерживающий отдел осуществляет ремонт. Связь один ко многим между Поддерживающий отдел (сотрудник) и Запрос на ремонт (сотрудник);
* Выделяем сотрудников, работающих на кухне, в отдельную таблицу. Связь один к одному между Поддерживающий отдел (сотрудник) и Кухня (сотрудник);
* Кухня назначает расписание приёмов пищи. Связь один ко многим между Кухня (сотрудник) и Расписание питания (ответственный).

## 2.4 ER-модель

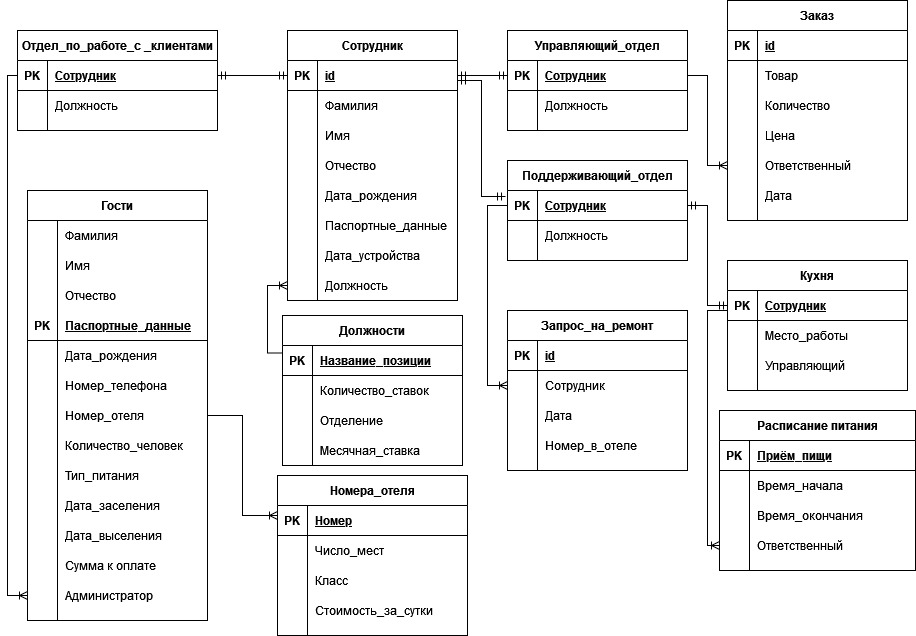


Рисунок 1 – ER-модель БД

## 2.5 Создание БД в СУБД MS SQL Server

Осуществляем создание БД, описанной ER-диаграммой (см. рисунок 1).

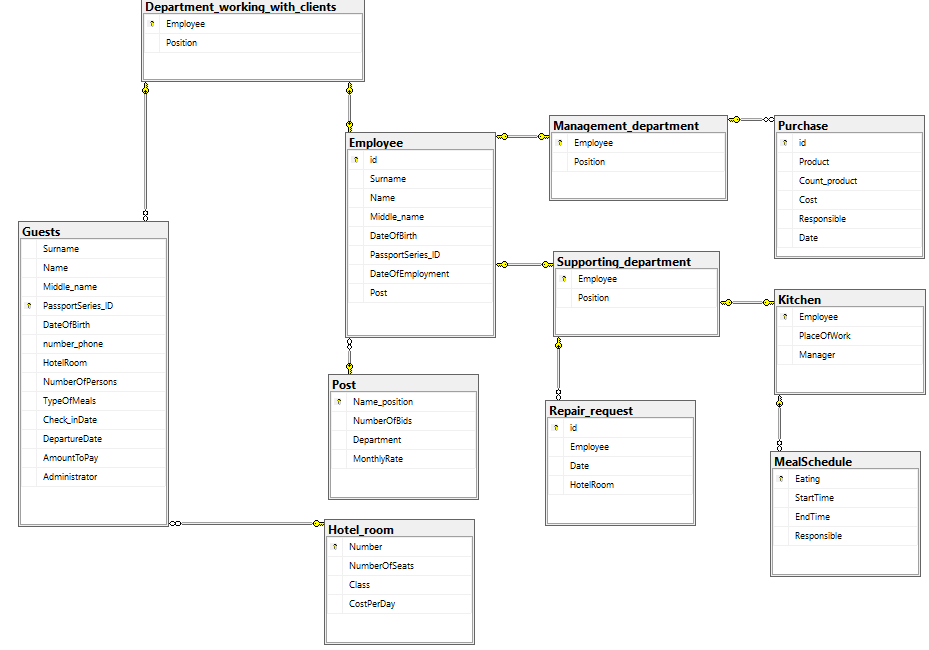


Рисунок 2 – Схема данных БД в MS SQL Server

## 2.6 Заполнение БД тестовыми значениями

Заполнение таблицы Post (рисунок 3).

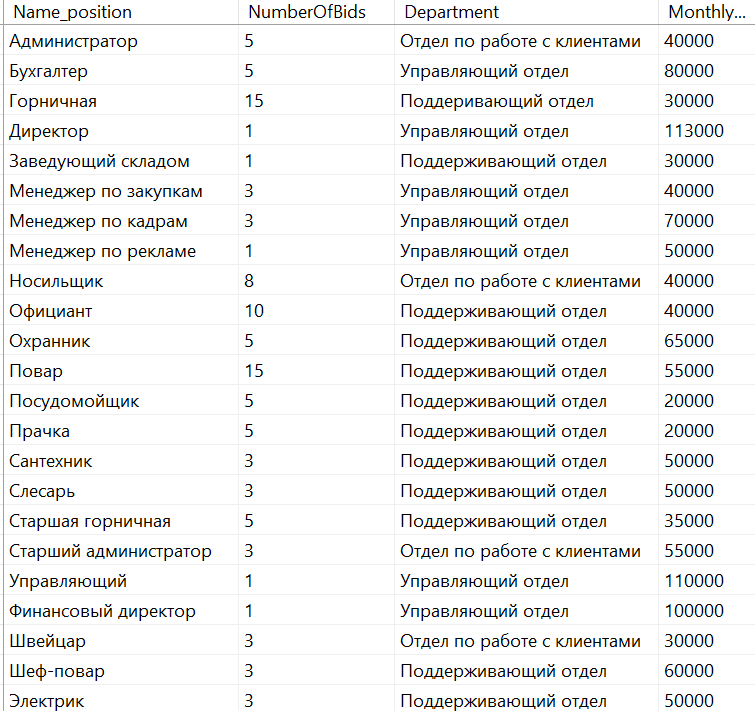


Рисунок 3 – таблица Post

Заполнение таблицы Employee (рисунки 4-5).



Рисунок 4 – таблица Employee

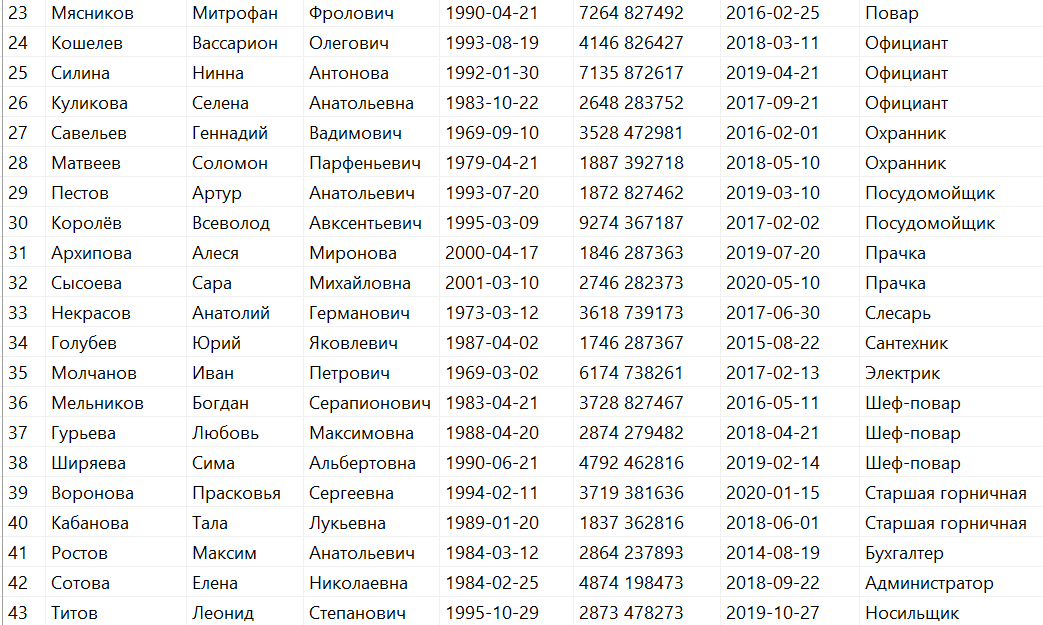


Рисунок 5 – продолжение таблицы Employee

Заполнение таблицы Management\_department (рисунок 6).

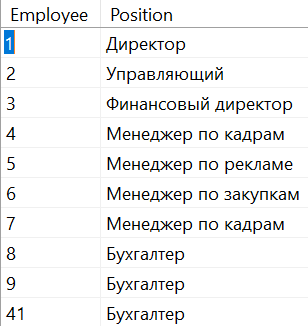


Рисунок 6 – таблица Management\_department

Заполнение таблицы Department\_working\_with\_clients (рисунок 7).

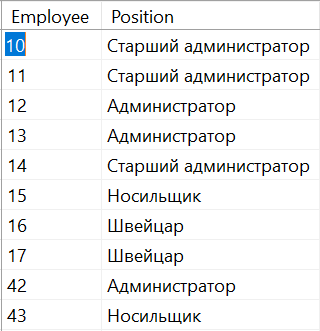


Рисунок 7 – таблица Department\_working\_with\_clients

Заполняем таблицу Supporting\_department (рисунок 8).

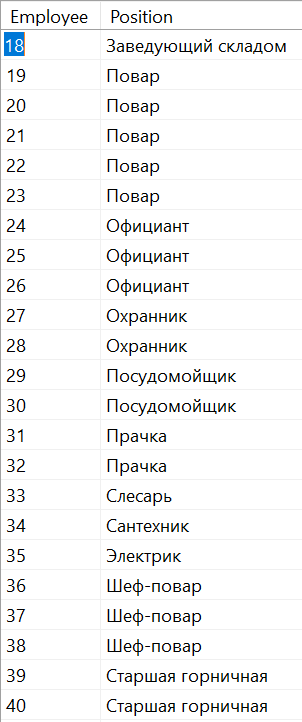


Рисунок 8 – таблица Supporting\_department

Заполняем таблицу Kitchen (рисунок 9).



Рисунок 9 – таблица Kitchen

Заполняем таблицу Guests (рисунок 10).

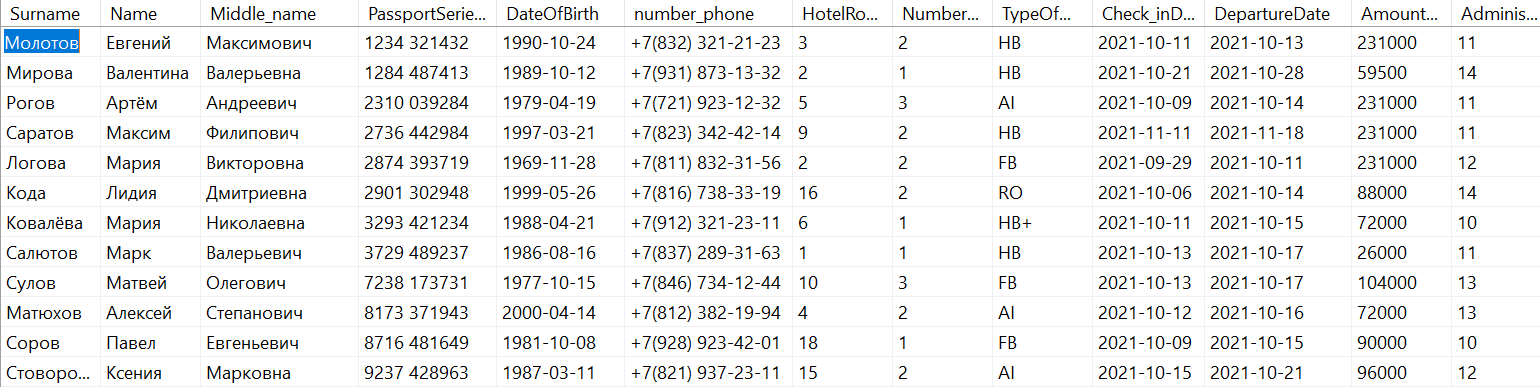


Рисунок 10 – таблица Guests

Заполняем таблицу Hotel\_room (рисунок 11).



Рисунок 11 – таблица Hotel\_room

Заполняем таблицу Purchase (рисунок 12).

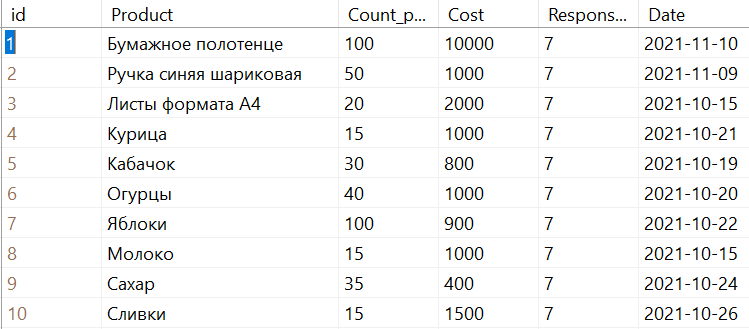


Рисунок 12 – таблица Purchase

Заполняем таблицу Repair\_request (рисунок 13).

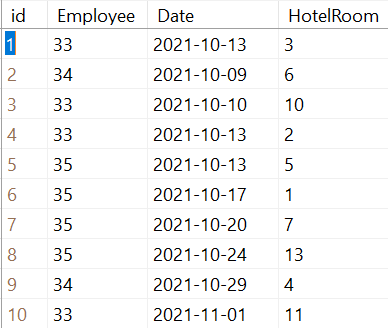


Рисунок 13 – таблица Repair\_request

Заполняем таблицу MealSchedule

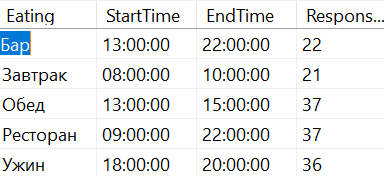


Рисунок 14 – таблица MealSchedule

# Глава 3. Работа с БД

## 3.1. Создание запросов

Для удобства заполнения базы напишем триггер для распределения добавленных сотрудников по отделам, а также триггер, который запишет сотрудников поддерживающего отдела в таблицу Кухня, если их должность относится к этому подотделу.

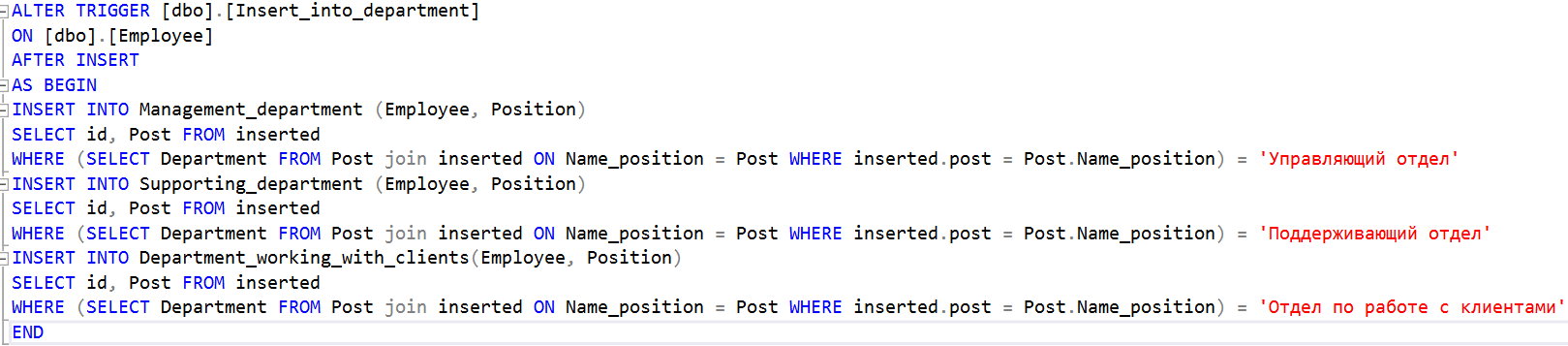


Рисунок 15 – код триггера Insert\_into\_department

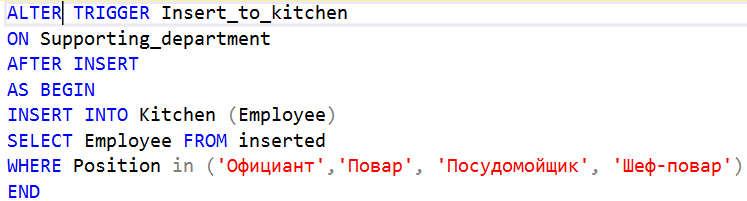


Рисунок 16 – код триггера Insert\_into\_kitchen

Проверим работоспособность данных триггеров.

Изначальную таблицу сотрудников можем увидеть на рисунке 4 и рисунке 5. Добавим в неё нового сотрудника.



Рисунок 17 – добавление сотрудника

Просмотрим результат в таблице поддерживающего отдела, куда должен был быть добавлен новый сотрудник.

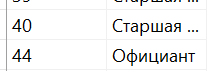


Рисунок 18 – добавленный с помощью сотрудник в таблице Supporting\_department

Также проверим таблицу для сотрудников кухни, т.к. должность Официант относится к этому подразделу. Видим, что id нового сотрудника есть в данной таблице.



Рисунок 19 – добавленная запись в таблице Kitchen

Следующая группа триггеров предназначена для изменения должности в таблице сотрудников, если в таблице, где он приписан к отделу, его должность будет изменена.

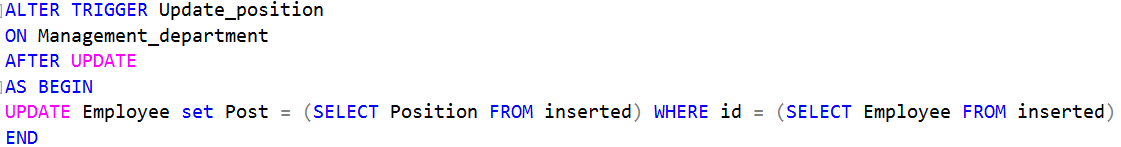


Рисунок 20 – триггер Update\_position для сотрудников управляющего отдела

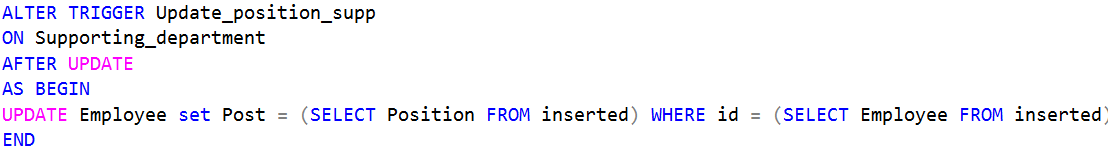


Рисунок 21 – триггер Update\_position\_supp для сотрудников поддерживающего отдела

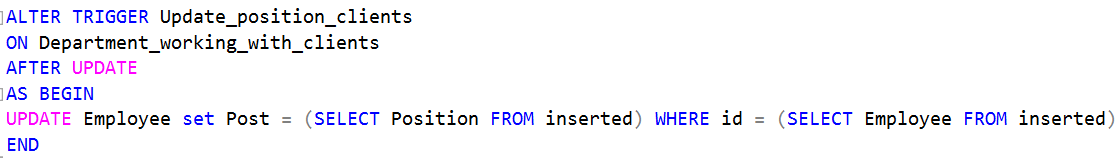


Рисунок 22 – триггер Update\_position\_clients для сотрудников отдела по работе с клиентами

Проверим работу триггеров на примере сотрудника, добавленного на рисунке 17. Переведём его на должность Повар, и проверим, какая должность у него будет в таблице сотрудников.



Рисунок 23 – изменение должности сотруднику



Рисунок 24 – изменённая должность в таблице сотрудников

Следующая группа триггеров рассчитывает стоимость, которую гость отеля должен оплатить за проживание в отеле, при добавлении гостя, а также при изменении данных о нём.

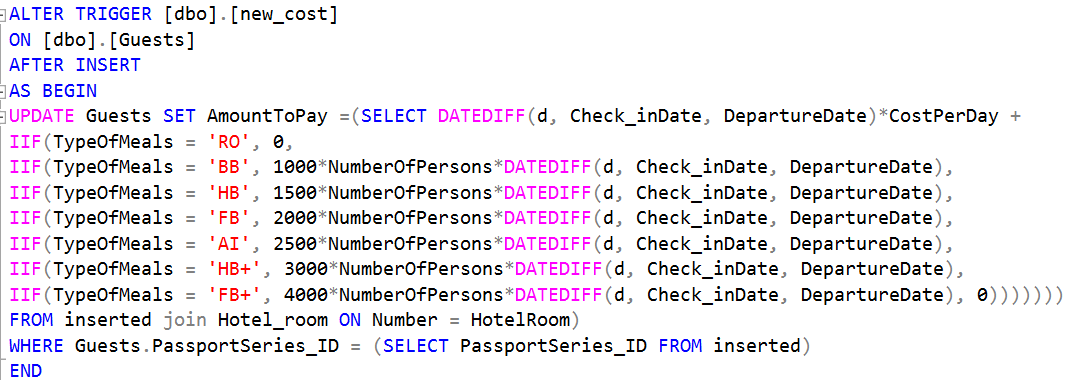


Рисунок 25 – триггер new\_cost для расчета при добавлении гостя

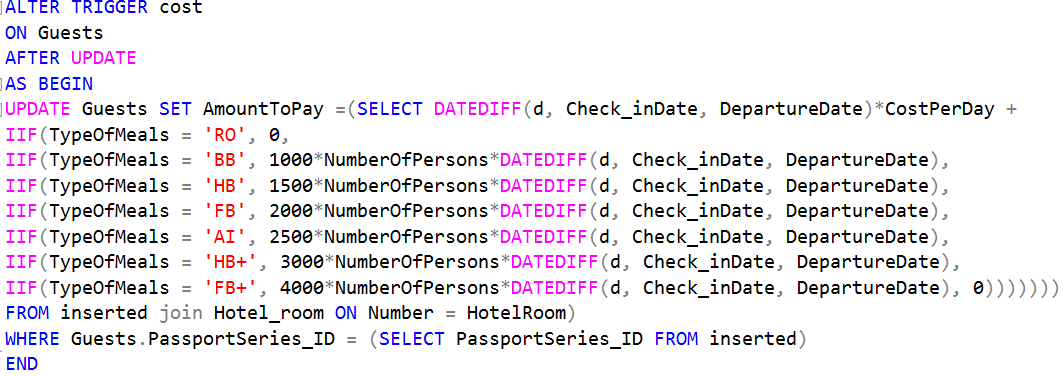


Рисунок 26 – триггер cost для расчета при изменении данных гостя

Проверим работу триггера, добавив нового гостя.



Рисунок 27 – добавление нового гостя



Рисунок 28 – результат работы триггера

Затем создаём процедуру для получения списка и данных о номерах отеля, которые свободны на данный момент

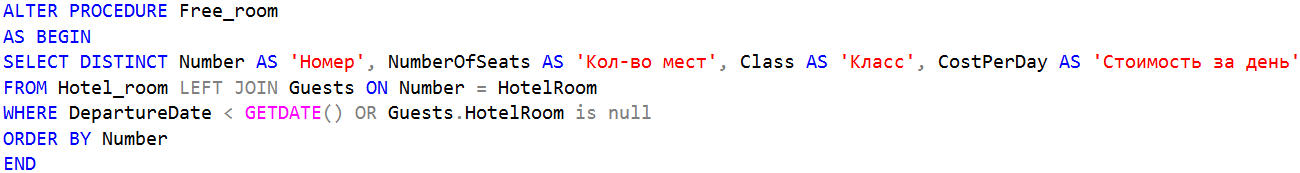


Рисунок 29 – процедура Free\_room

Проверим работу процедуры. Ожидаемый результат – все номера, за исключением номера 9, в котором ещё числится гость.



Рисунок 30 – запуск процедуры Free\_room

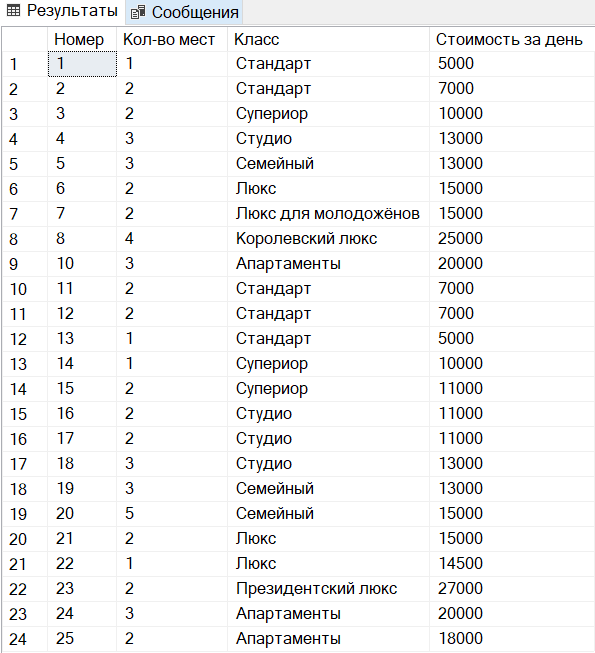


Рисунок 31 – результат запуска процедуры

Последняя процедура выполняет подсчет затрат отеля на закупки в определённый месяц.

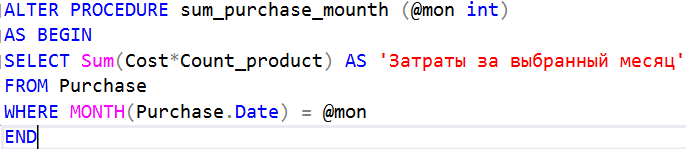


Рисунок 32 – код процедуры sum\_purchase\_month

Проверим работу процедуры, запросив подсчёт затрат за октябрь



Рисунок 33 – запуск процедуры



Рисунок 34 – результат выполнения процедуры

## 3.2 Интерфейс

Создаём графический интерфейс для работы с БД. Создаём проект Windows Forms, подключаем к нему базу данных из MS SQL Server. Расставляем элементы управления на форме, настраиваем их параметры. Полученная форма представлена на рисунках 35 и 36.

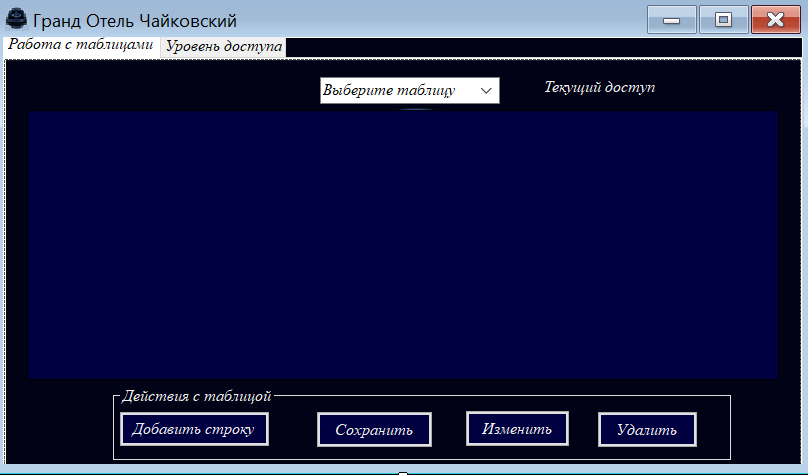


Рисунок 35 – первое окно формы

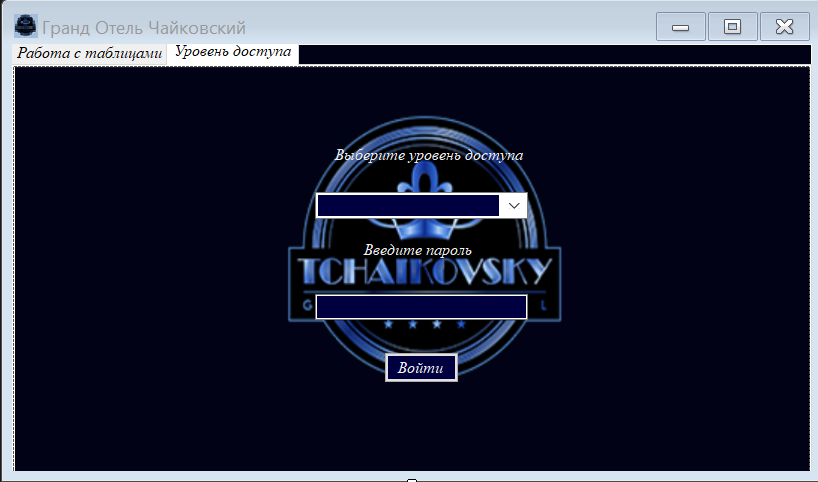


Рисунок 36 – второе окно формы

Затем прописываем события, происходящие при взаимодействии с формой. Написанный код формы представлен в Приложении. Продемонстрируем работу приложения при запуске на разных уровнях доступа.

Запуск от имени администратора предоставляет доступ ко всем функциям приложения, открывает все созданные таблицы и позволяет их изменять.

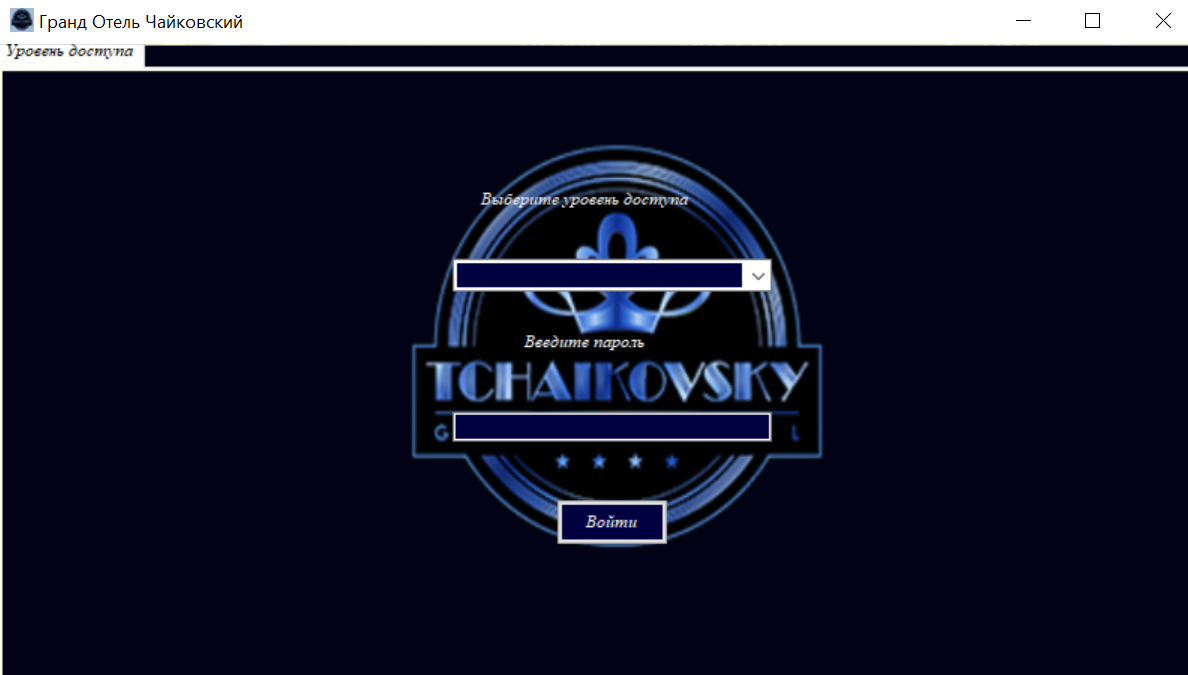


Рисунок 37 – запуск приложения

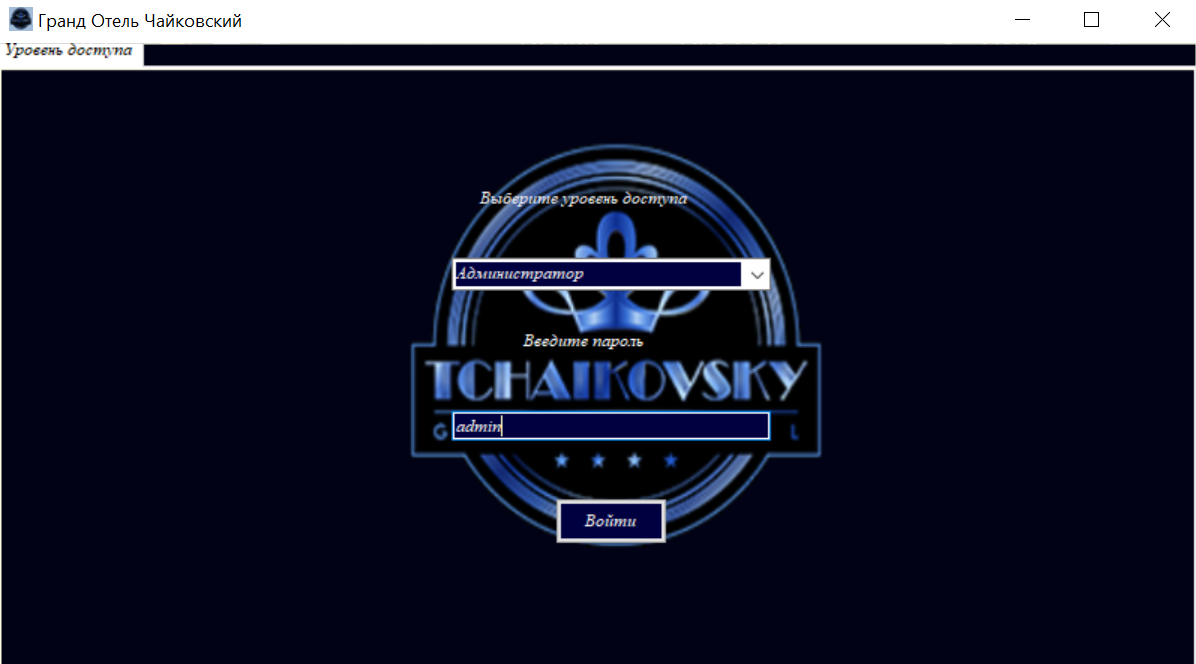


Рисунок 38 – выбор уровня доступа Администратора

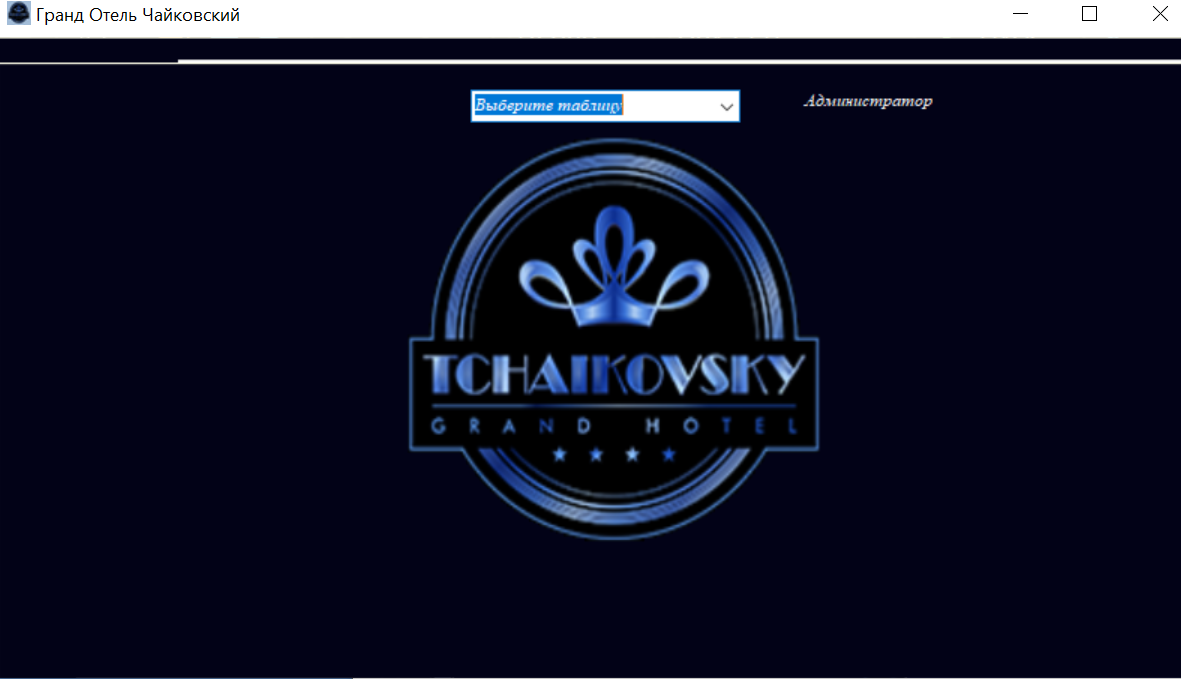


Рисунок 39 – успешный вход в аккаунт Администратора

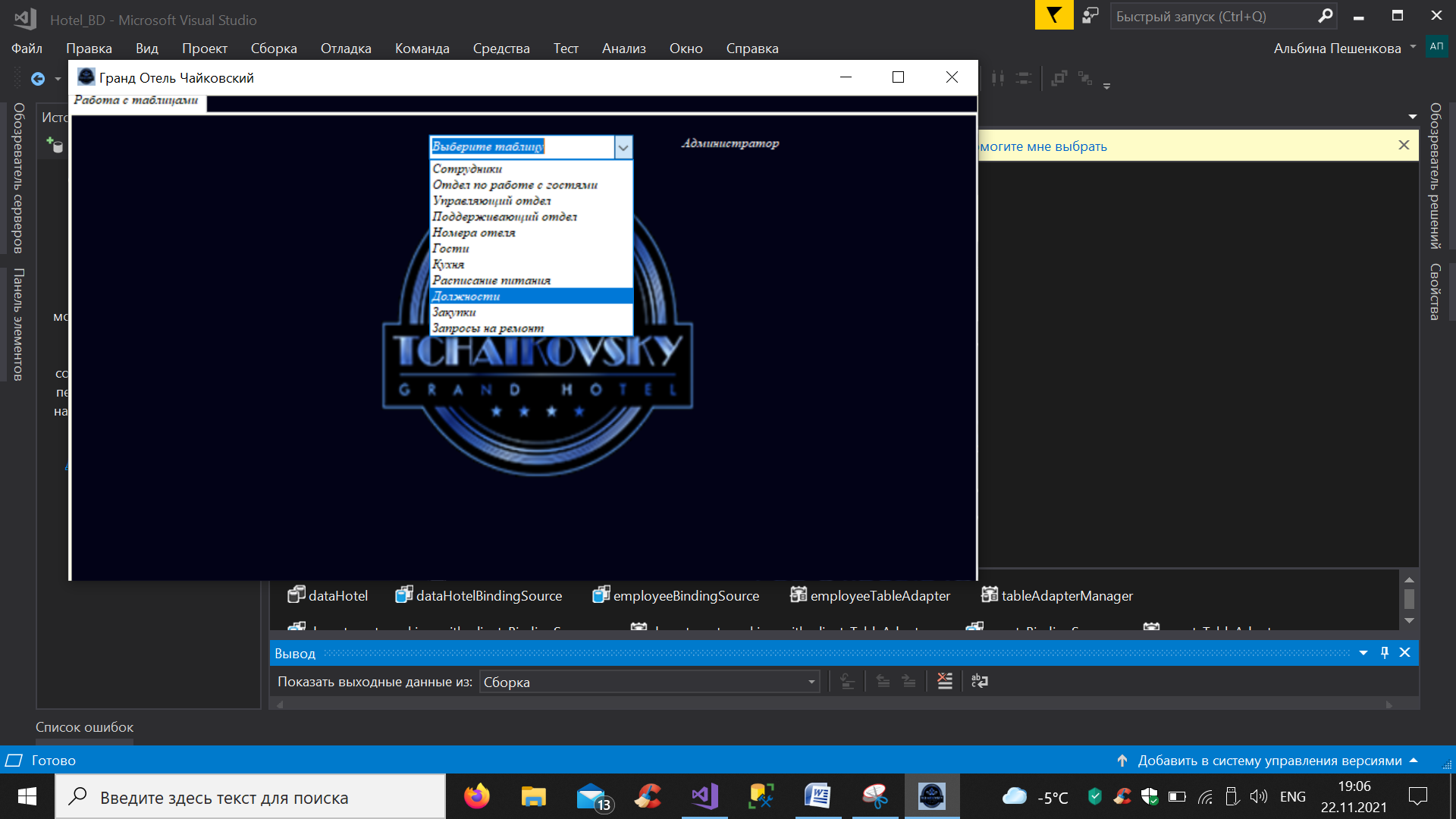


Рисунок 40 – доступный список таблиц

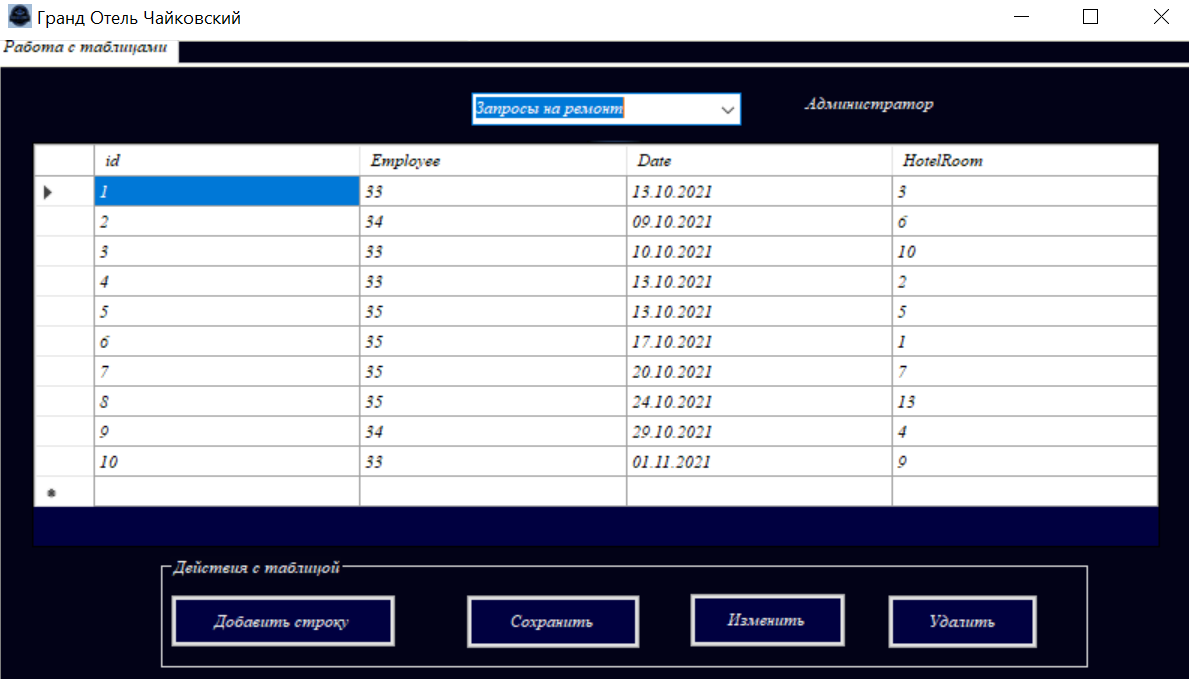


Рисунок 41 – вид окна при выборе таблицы

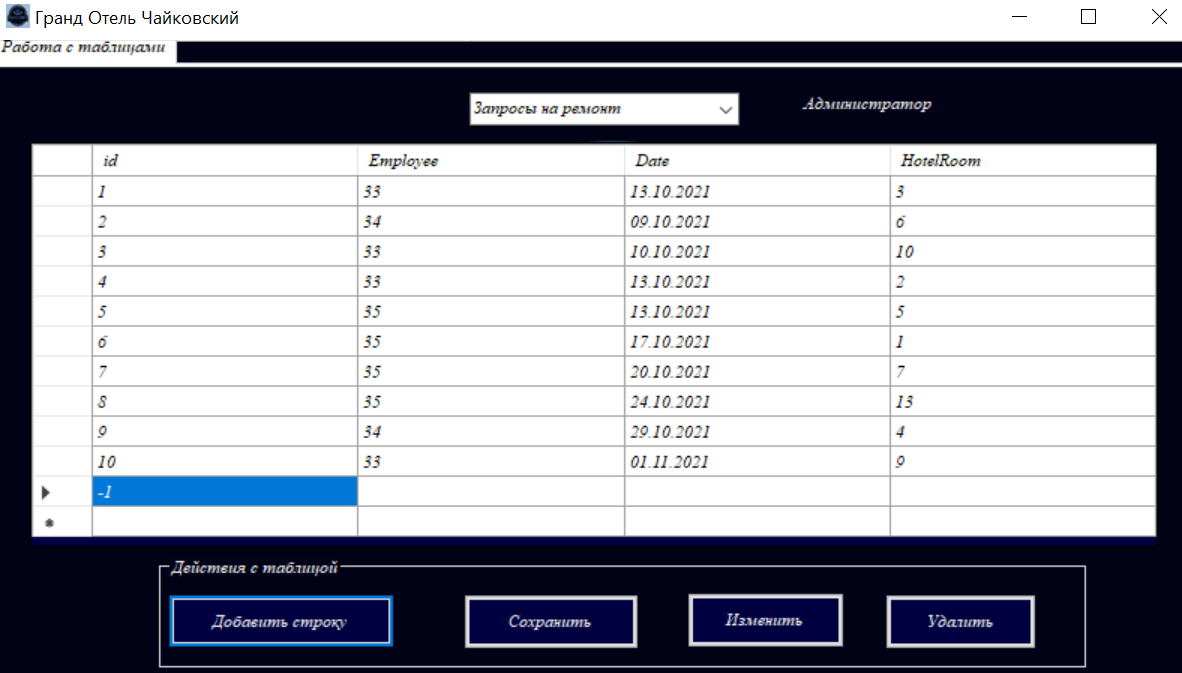


Рисунок 42 – вид при нажатии «Добавить строку»

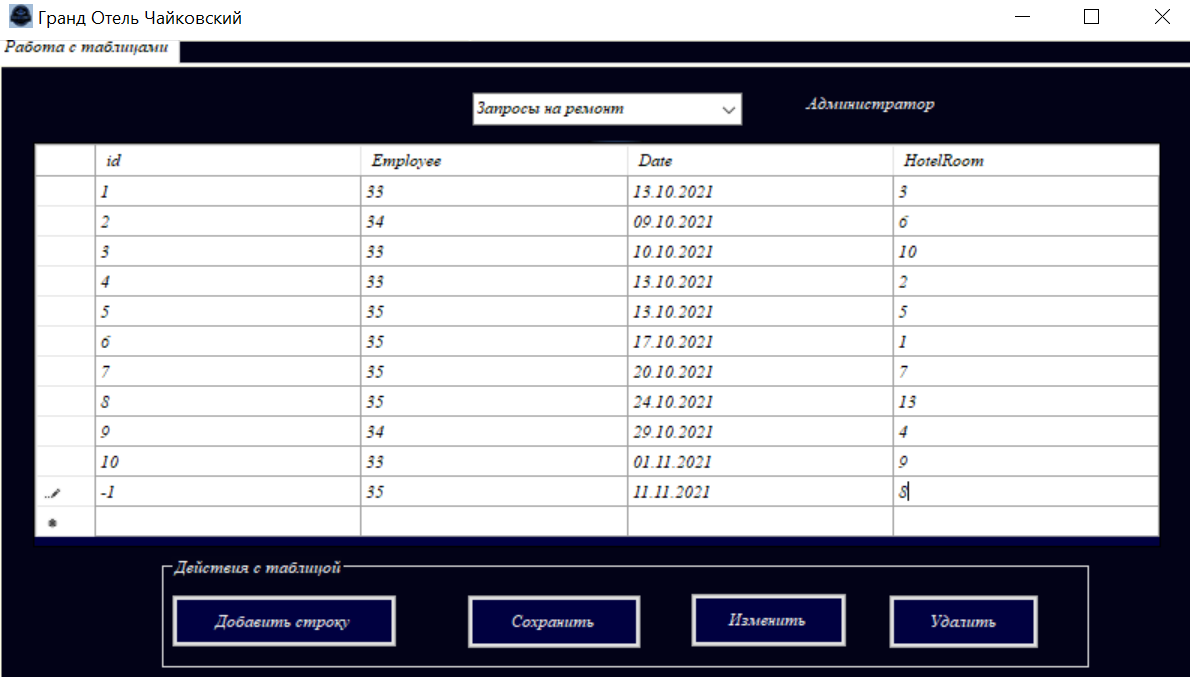


Рисунок 43 – заполняем данные для новой записи

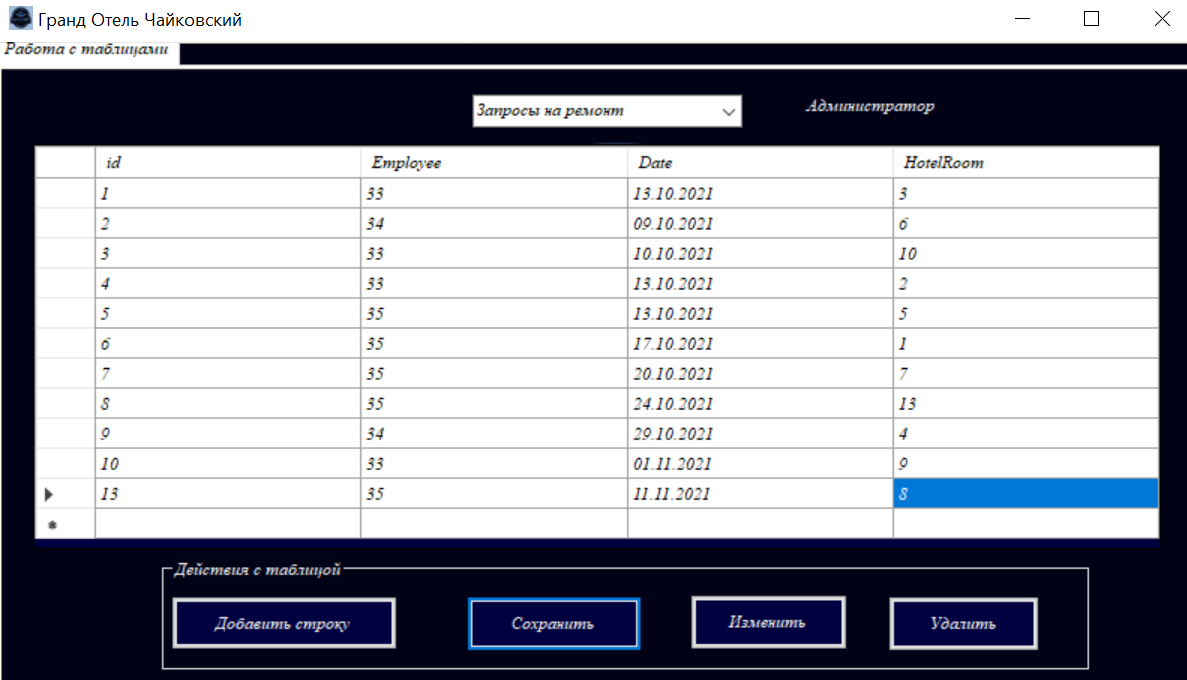


Рисунок 44 – сохраняем записанные данные в БД

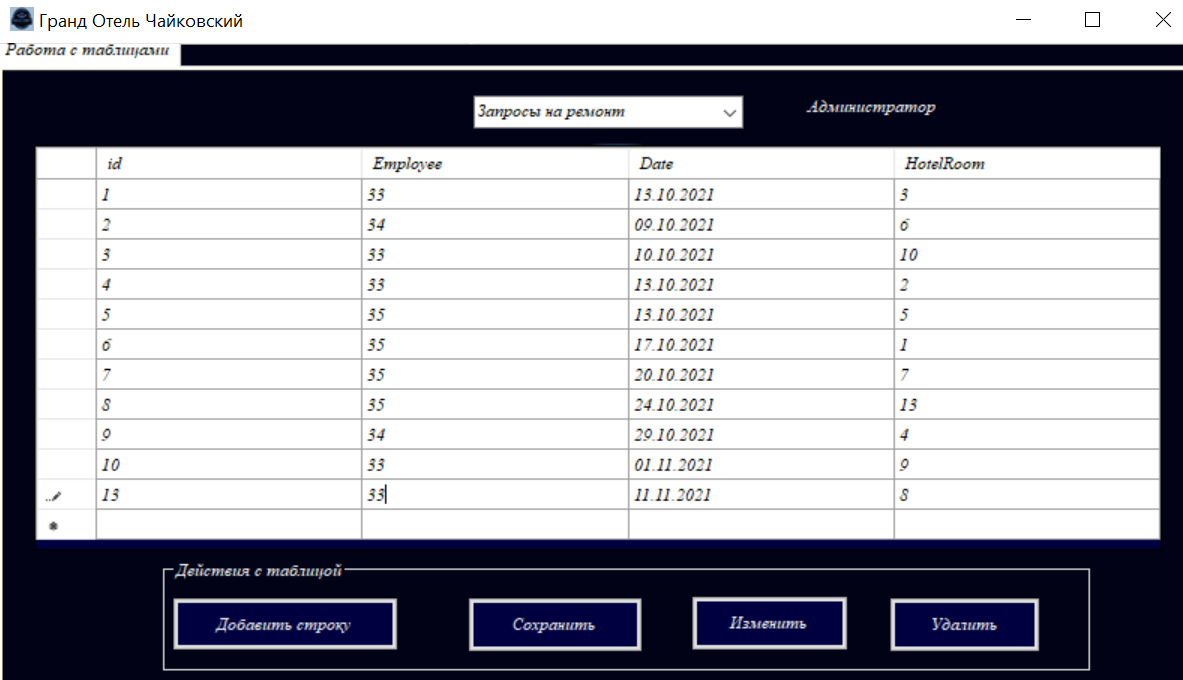


Рисунок 45 – нажатие на кнопки «Изменить», после чего появляется возможность менять значение

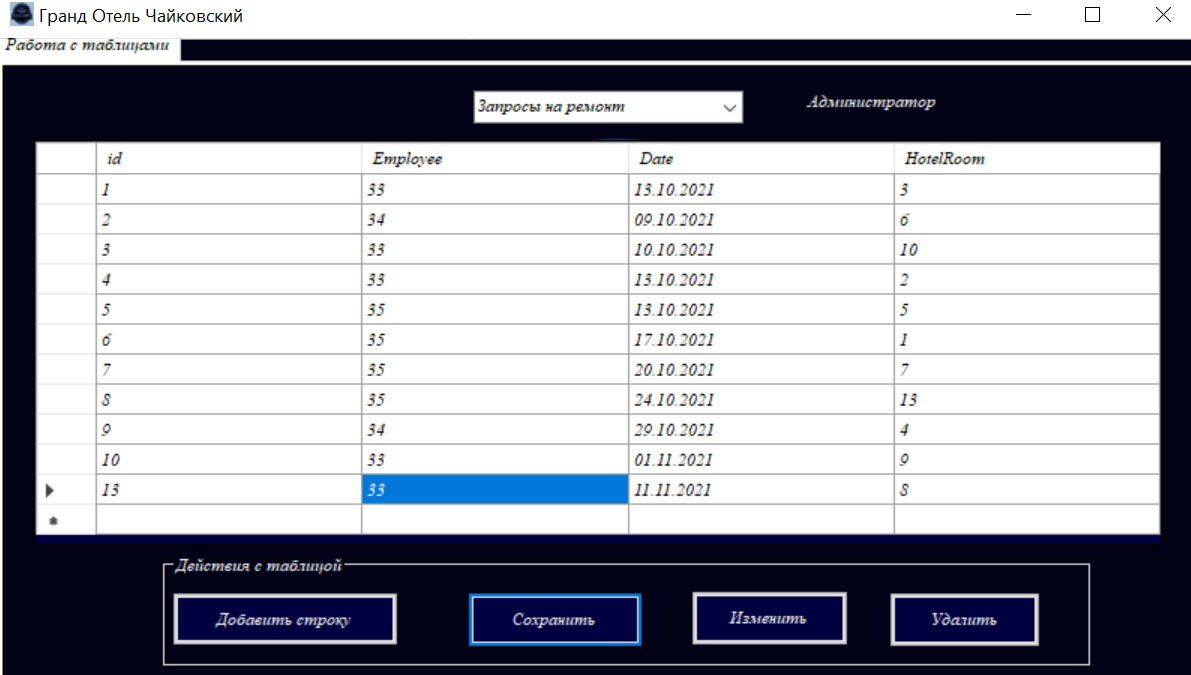


Рисунок 46 – нажимаем кнопку «Сохранить», чтобы сохранить изменённые значения

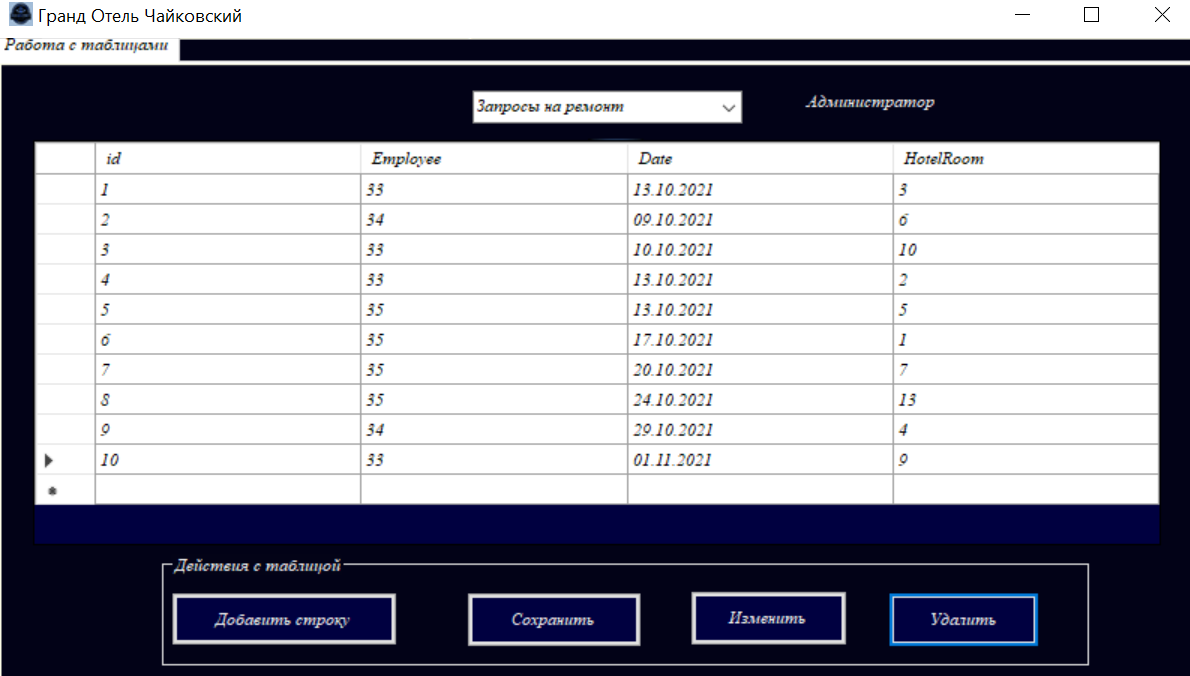


Рисунок 47 – удаление записи

Запускаем приложение от имени сотрудника управляющего отдела.

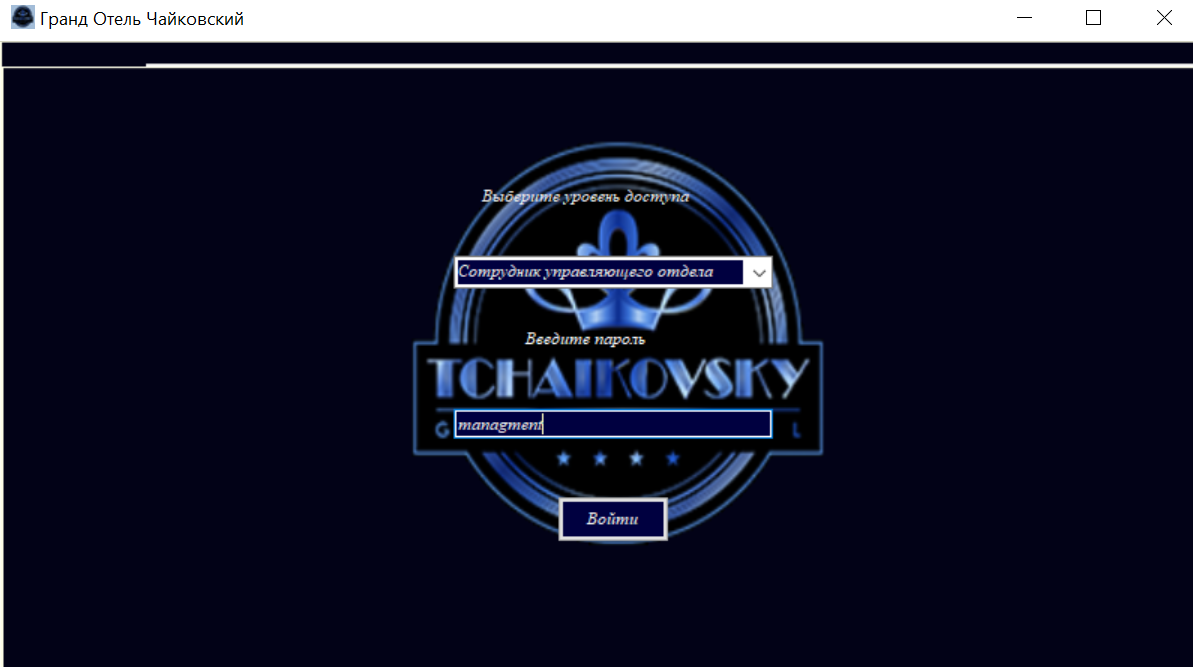


Рисунок 48 – вход в аккаунт сотрудника управляющего отдела

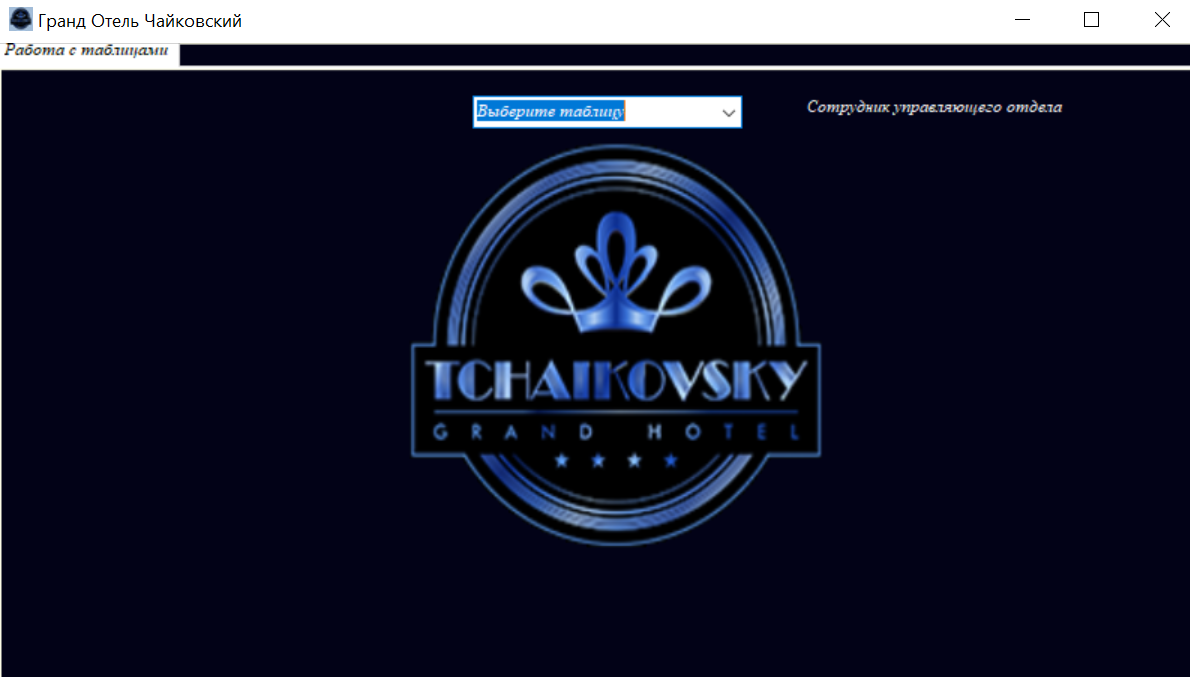


Рисунок 49 – окно при успешном входе в аккаунт

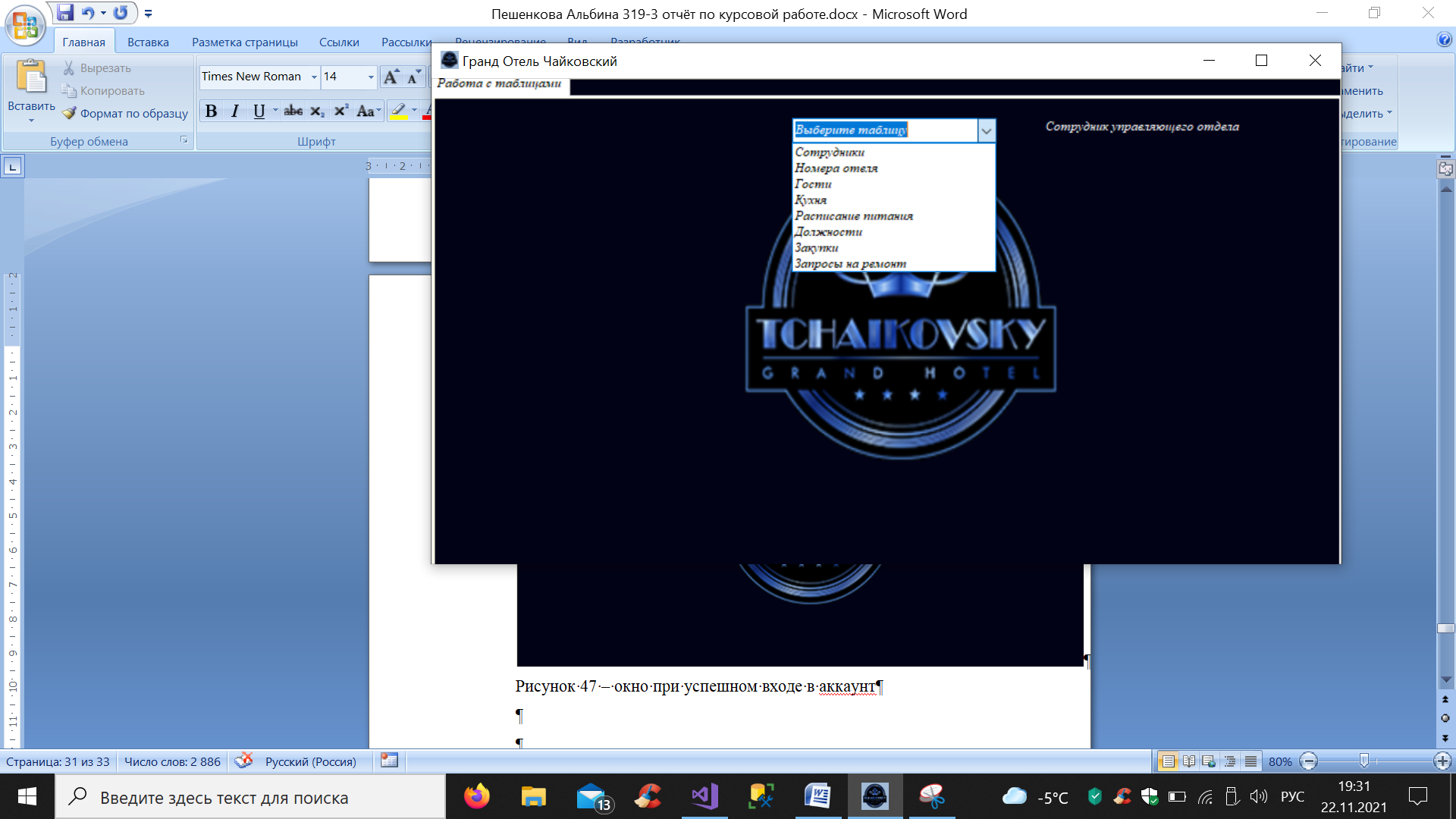


Рисунок 50 – список доступных таблиц для сотрудников управляющего отдела

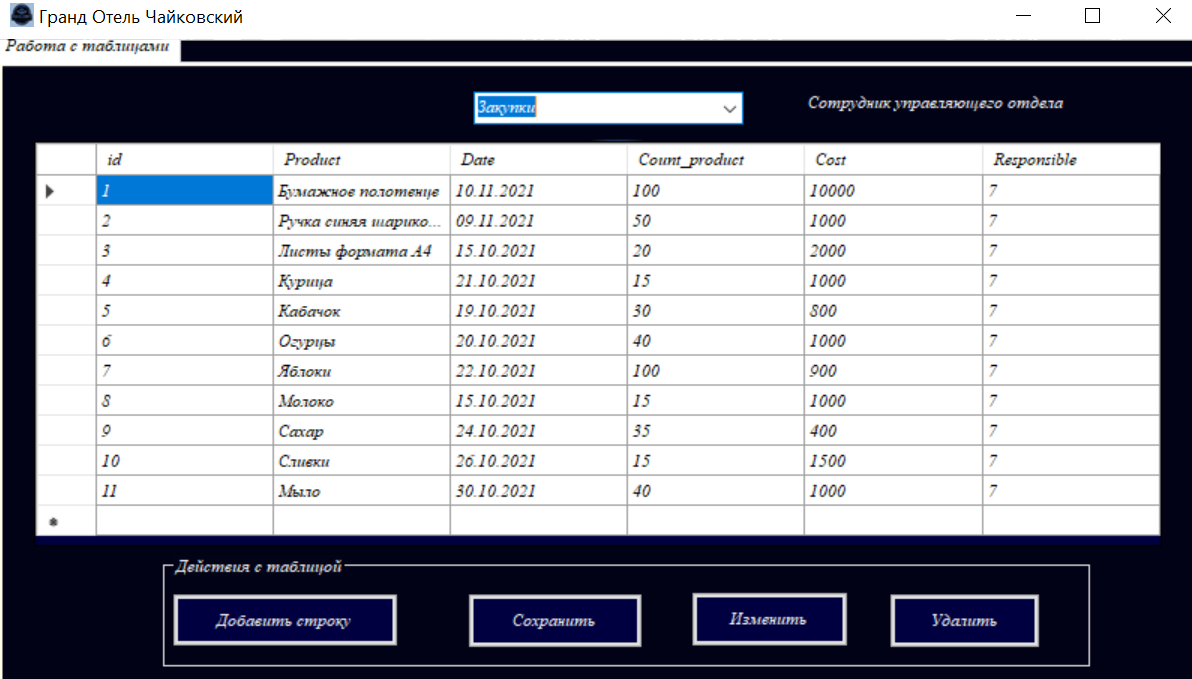


Рисунок 51 – открытие выбранной таблицы

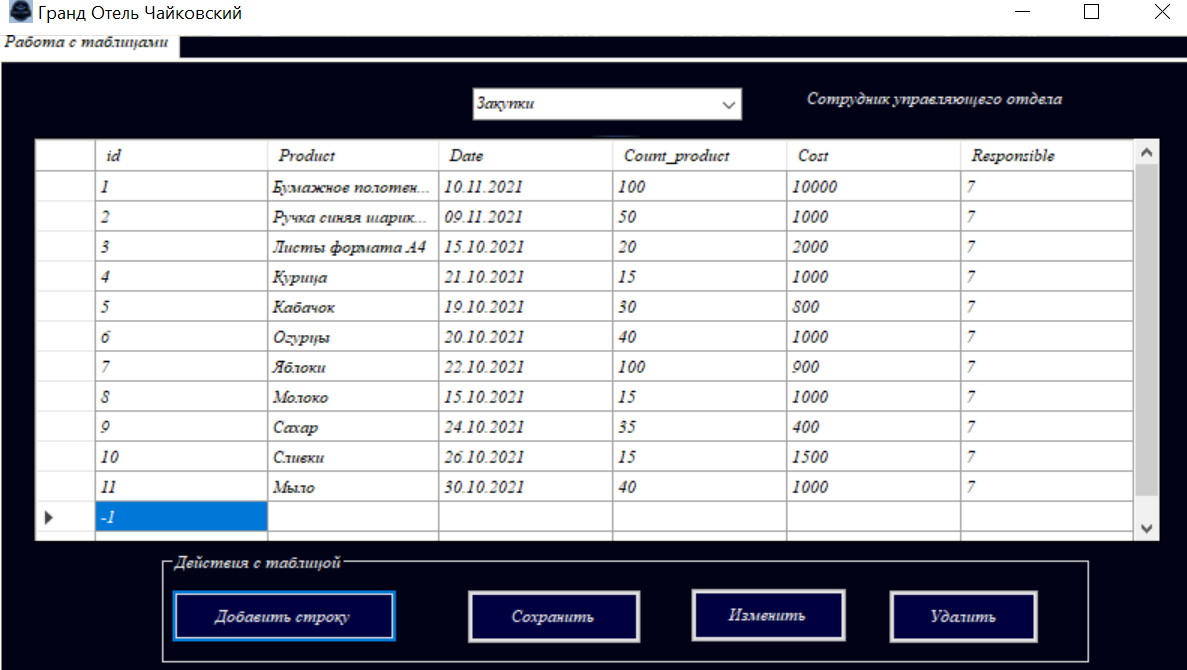


Рисунок 52 – результат нажатия кнопки «Добавить строку»



Рисунок 53 – заполнение добавленной строки

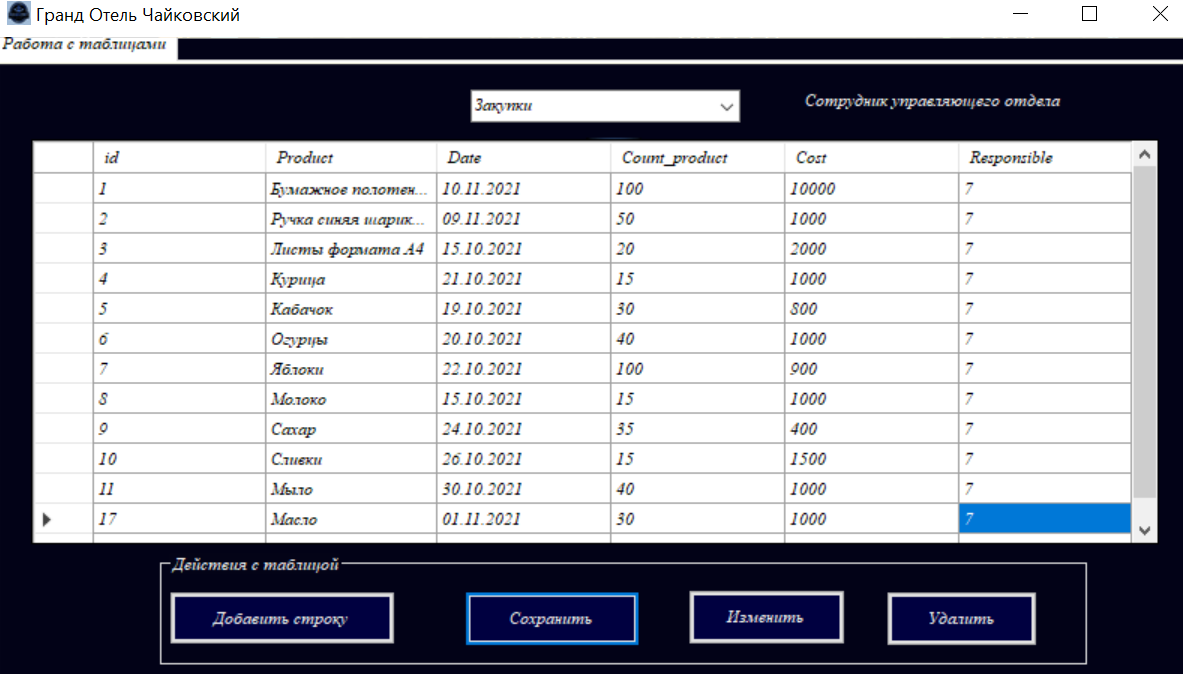


Рисунок 54 – сохранение добавленной строки

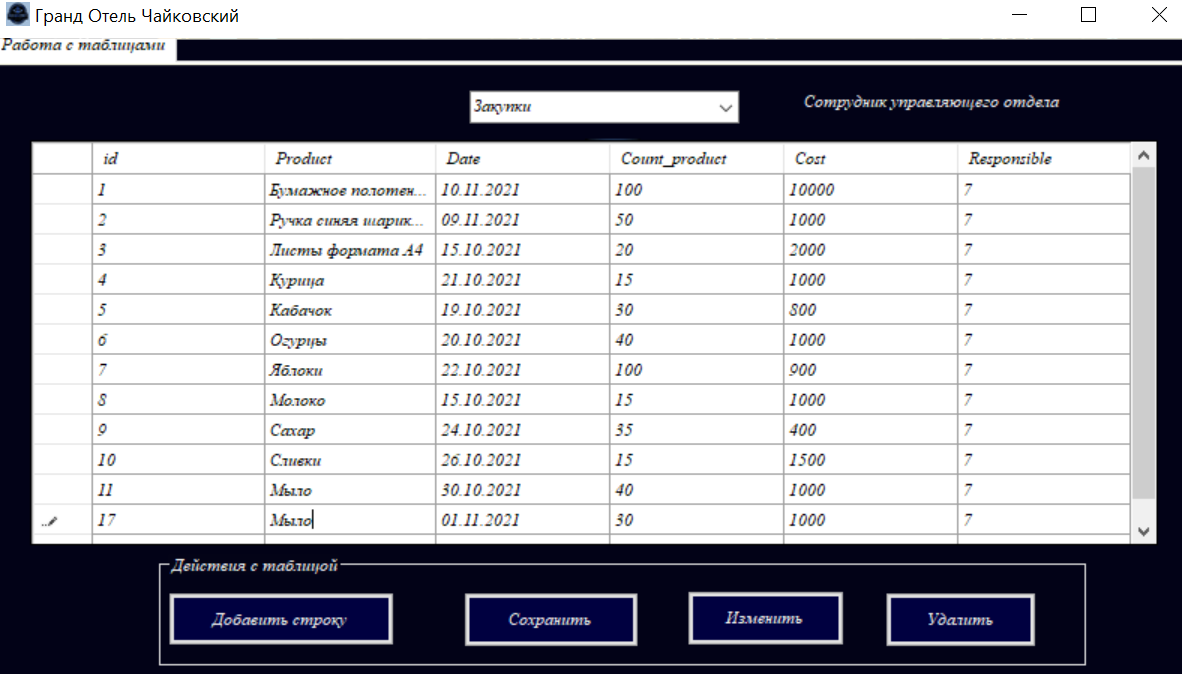


Рисунок 55 – изменение записи

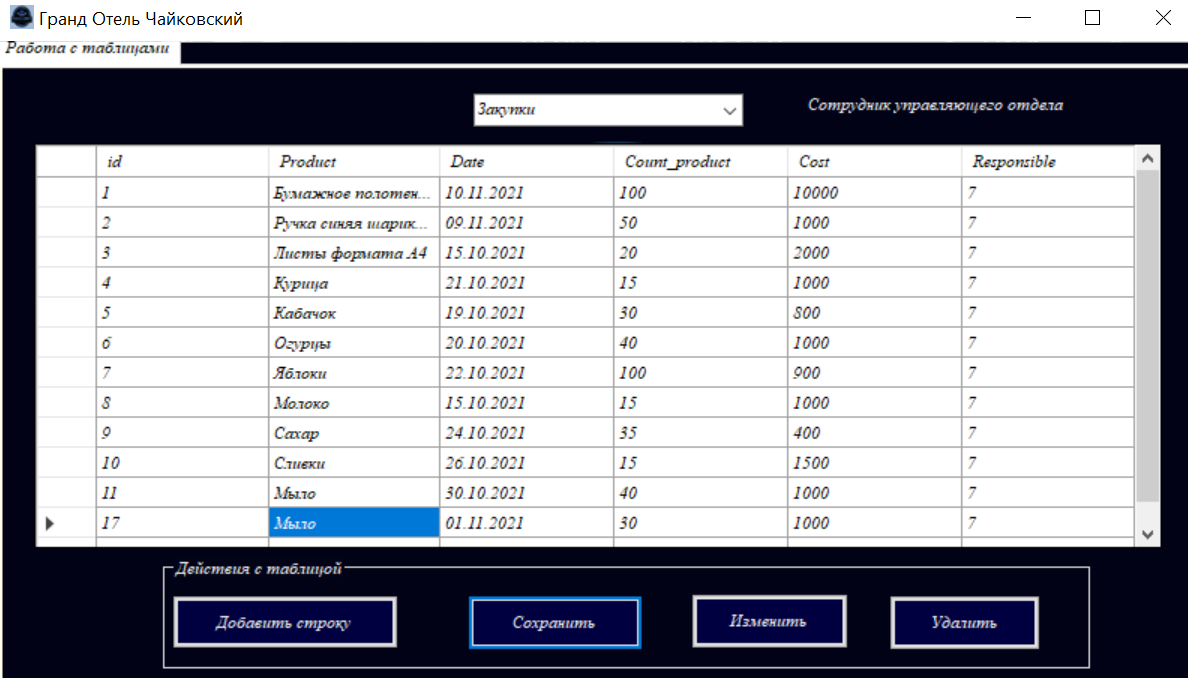


Рисунок 56 – сохранение изменения записей

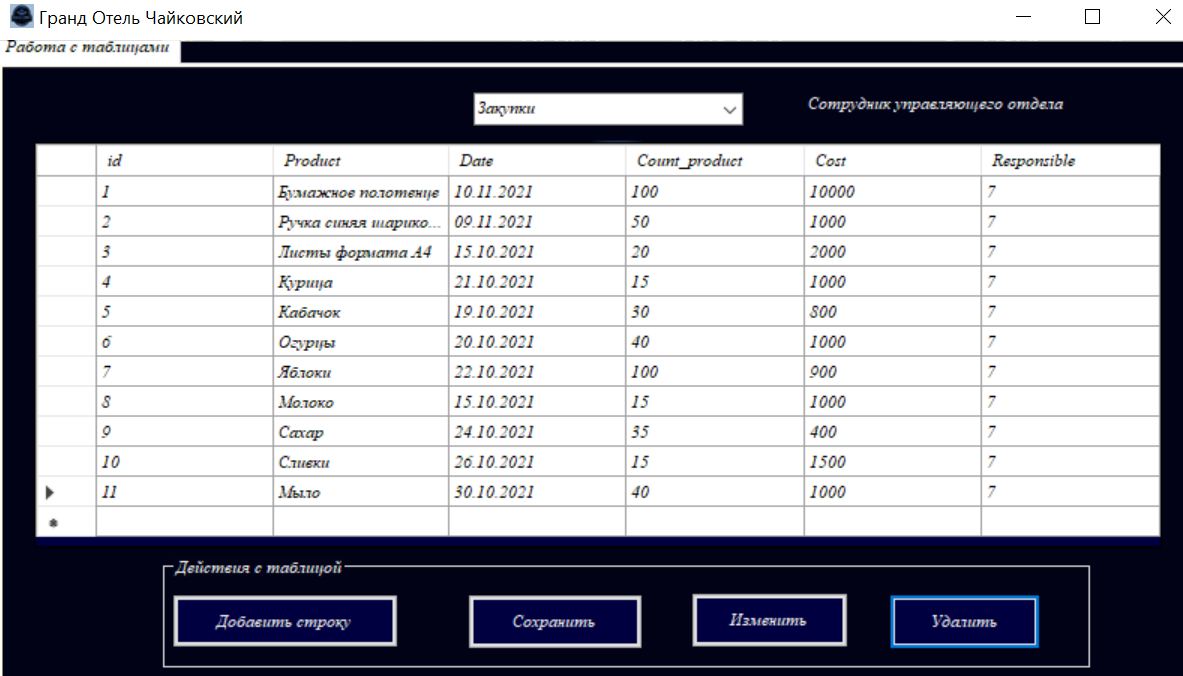


Рисунок 57 – удаление записи

Запустим приложение для уровня доступа «Сотрудник отдела по работе с клиентами».

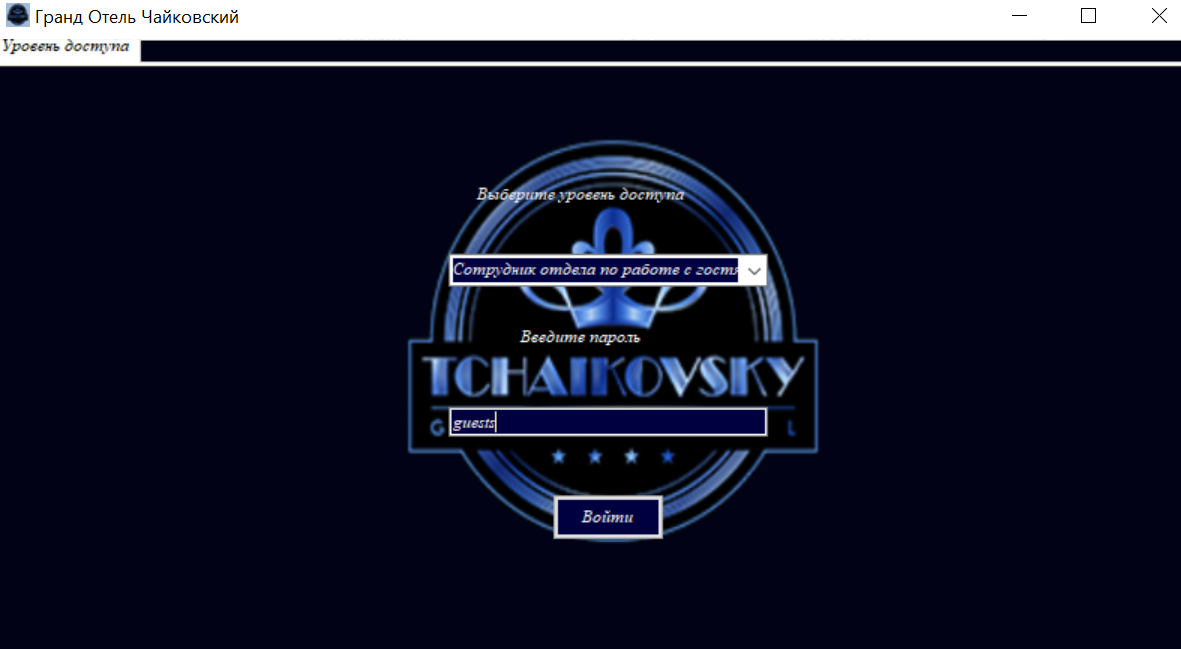


Рисунок 58 – вводим данные аккаунта сотрудника отдела по работе с клиентами

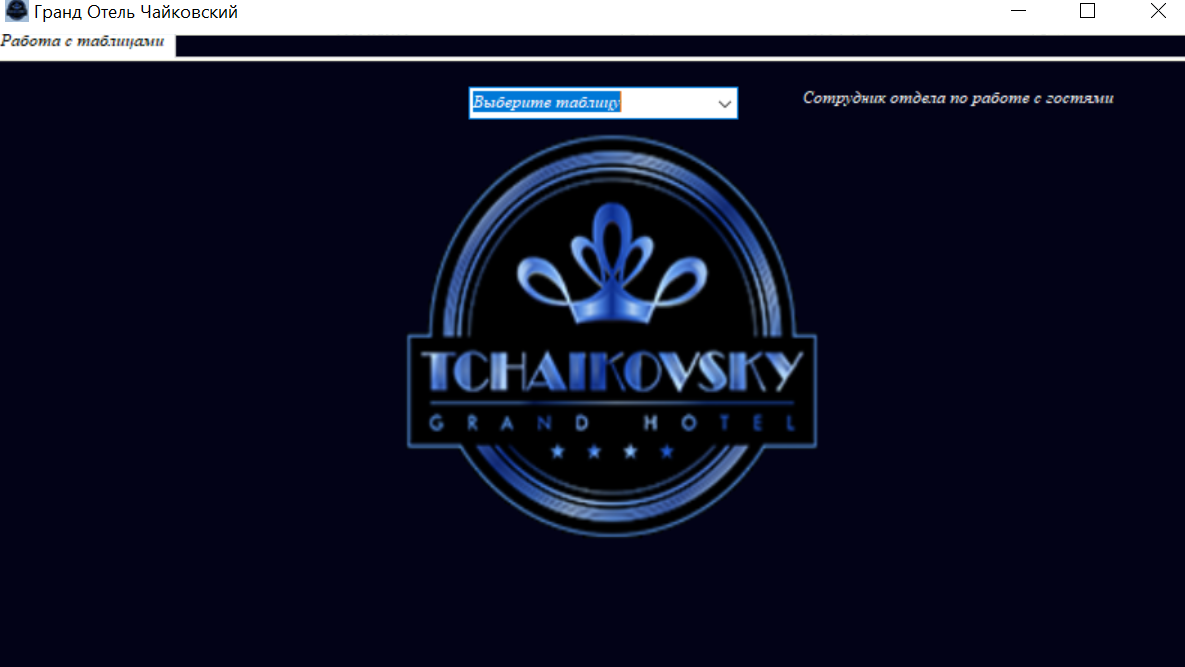


Рисунок 59 – окно при входе в аккаунт

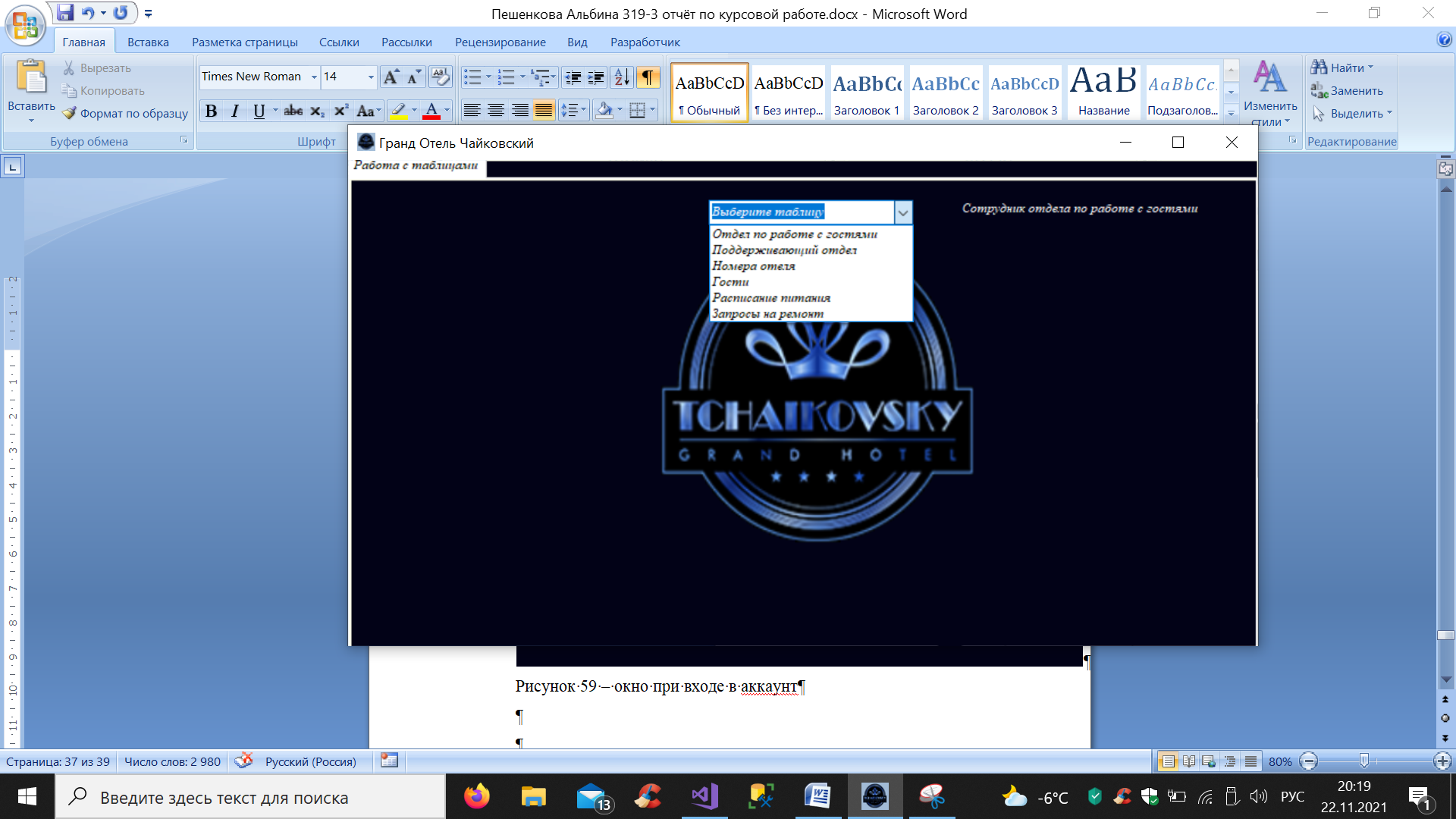


Рисунок 60 –доступные таблицы

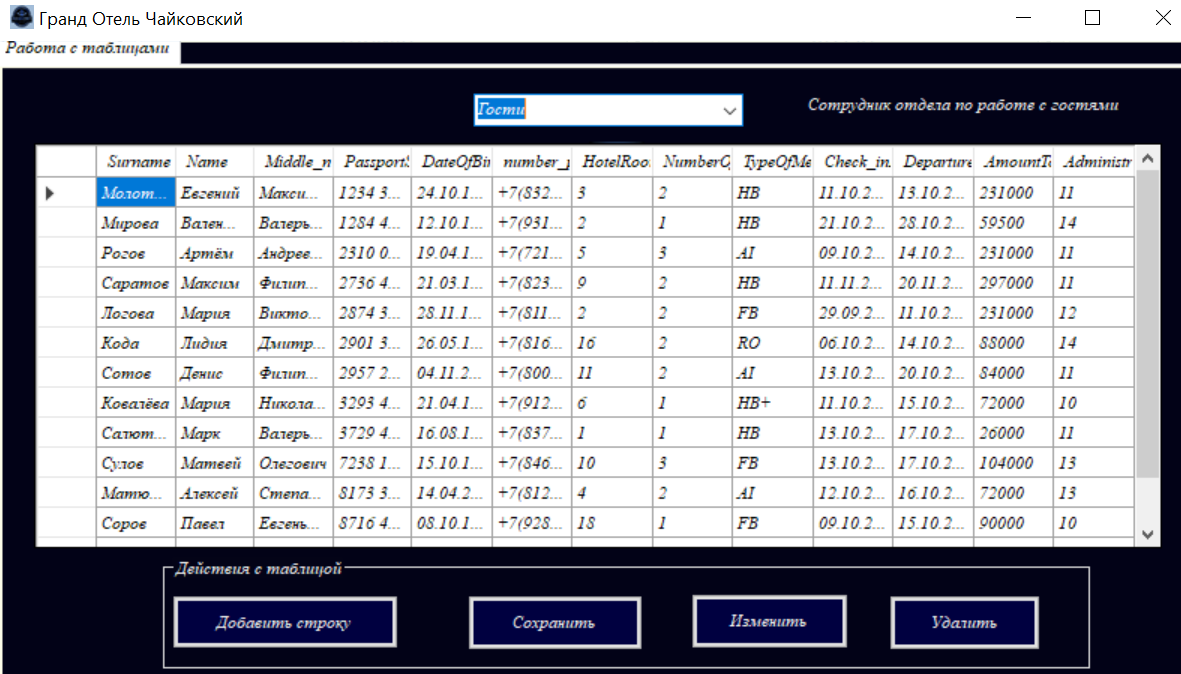


Рисунок 61 – открытие таблицы

Функции «Добавление», «Изменение», «Сохранение» и «Удаление» работают также, как в предыдущих уровнях доступа.

Запустим приложение для уровня доступа сотрудника поддерживающего отдела.

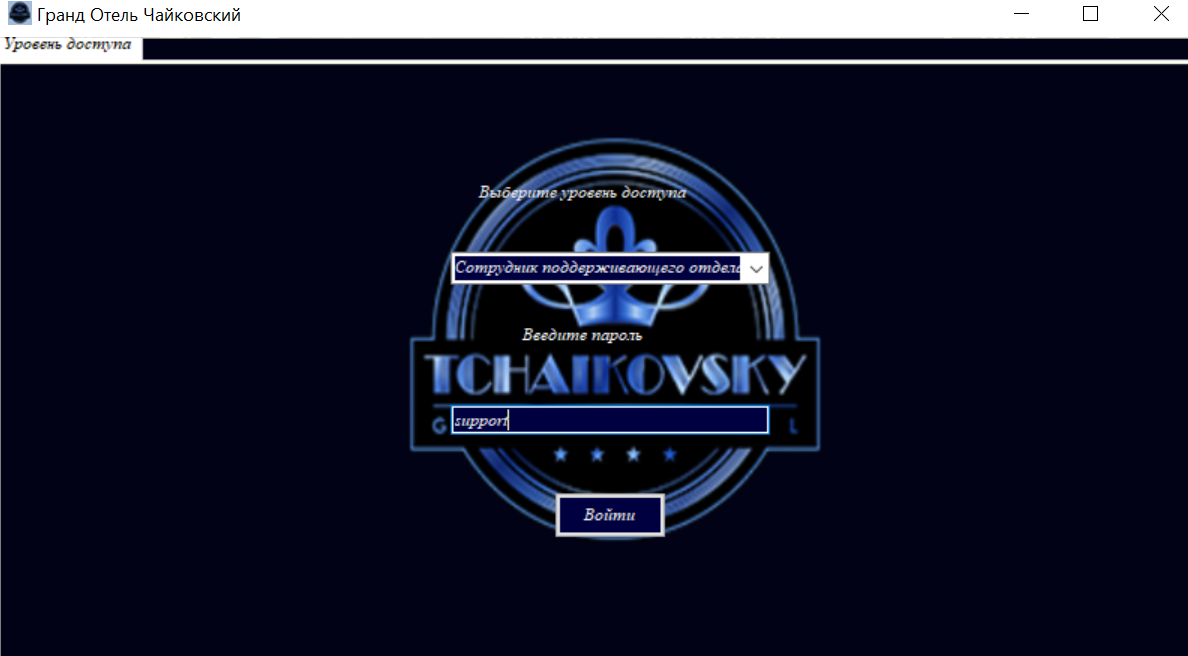


Рисунок 62 – ввод данных аккаунта сотрудника поддерживающего отдела

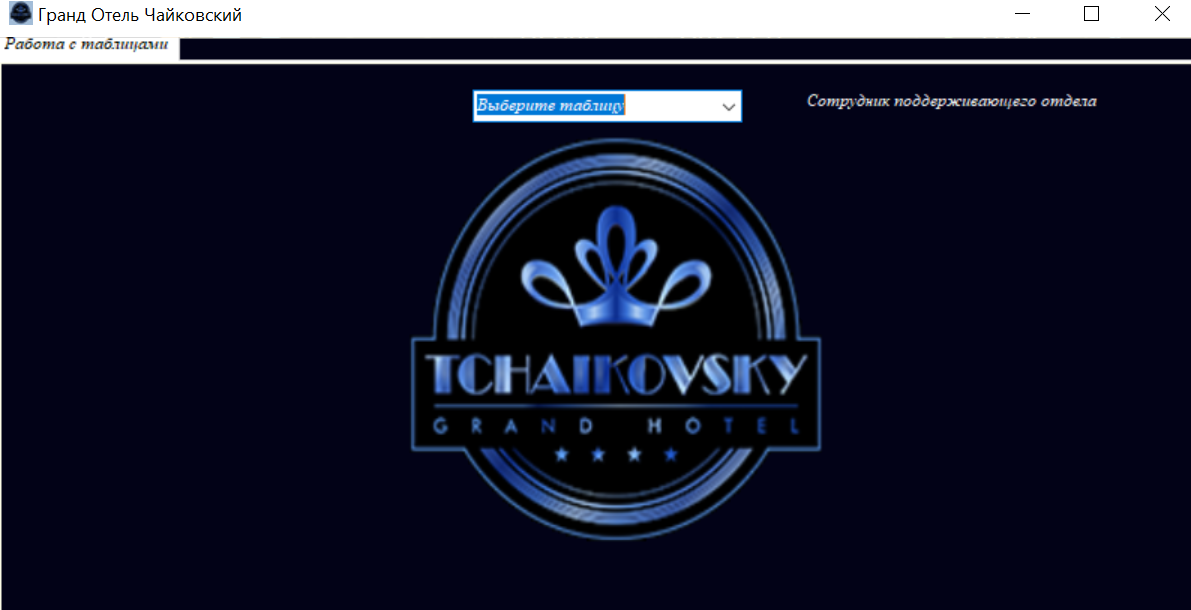


Рисунок 63 – окно при входе в аккаунт

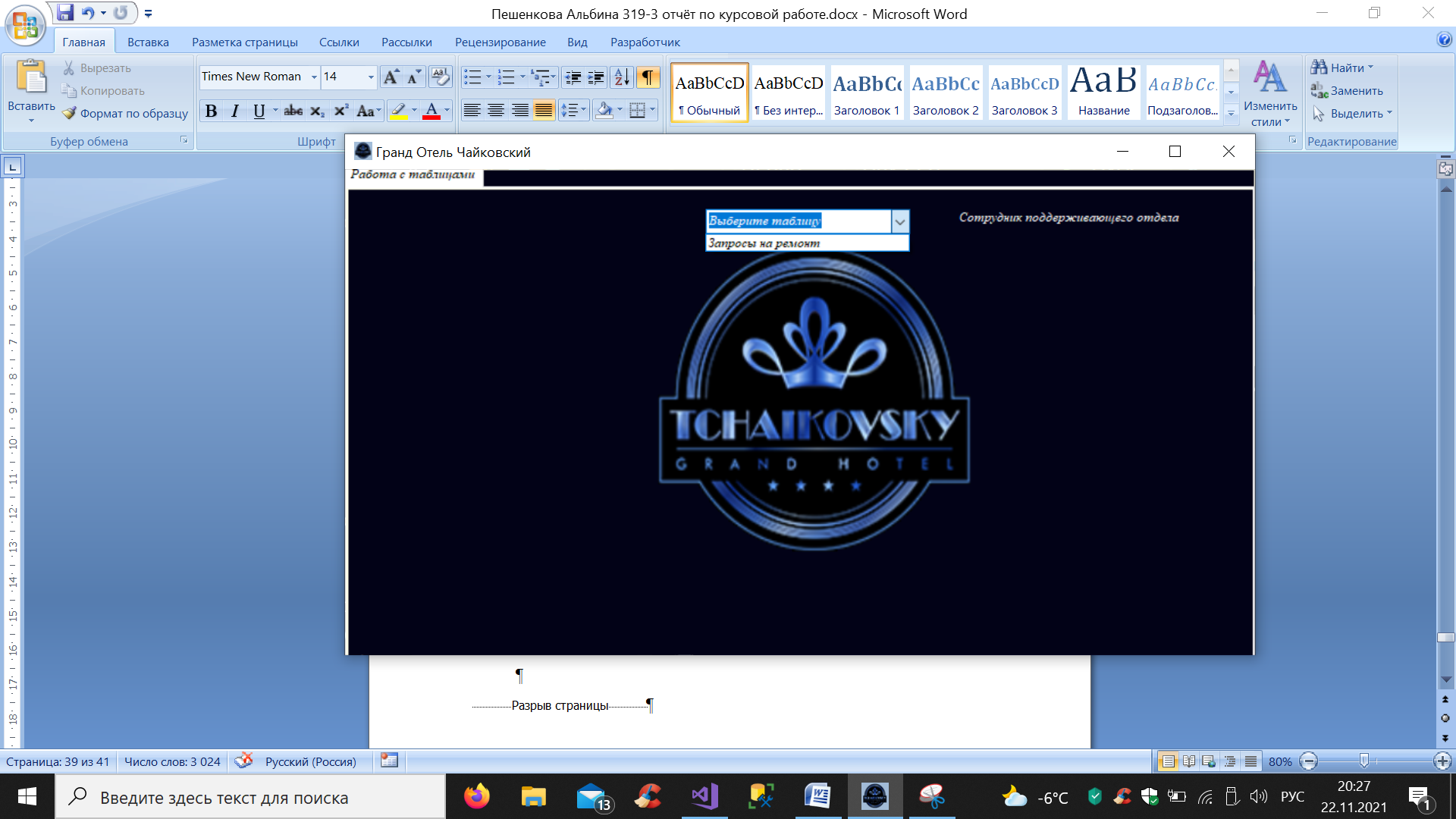


Рисунок 64 – доступные таблицы

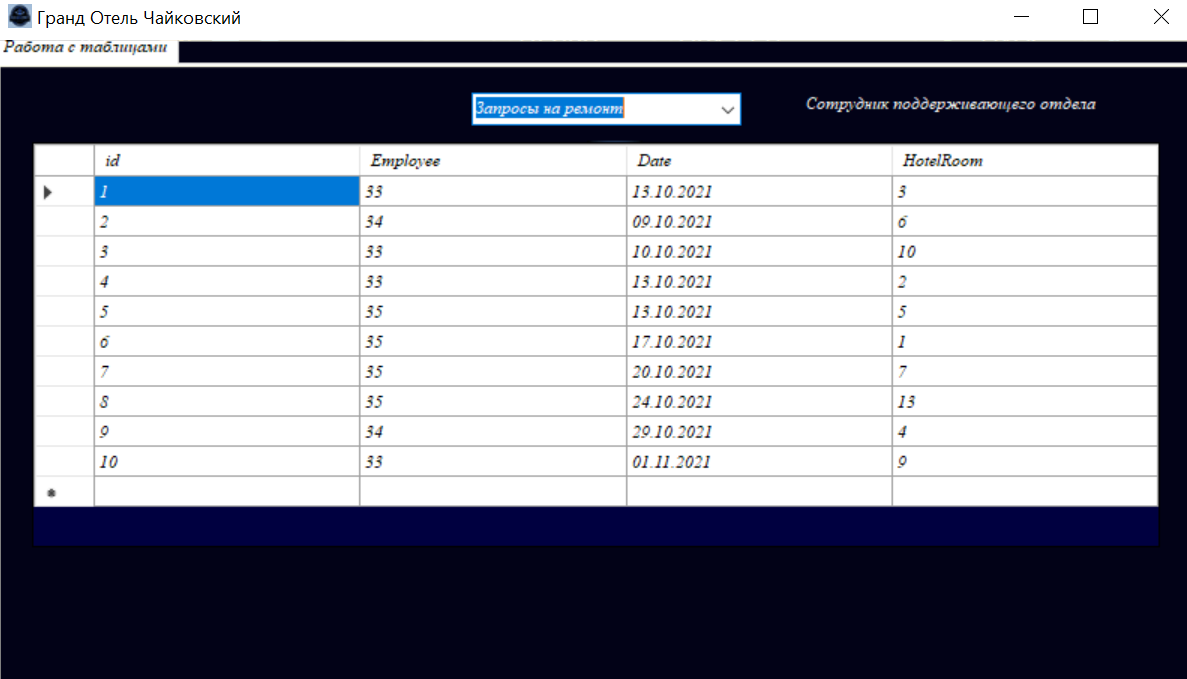


Рисунок 65 – вид окна при выборе таблицы

В данном уровне доступа изменение записей не доступно.

При вводе некорректных данных аккаунта появляется окно, уведомляющее о ошибке вводе данных.

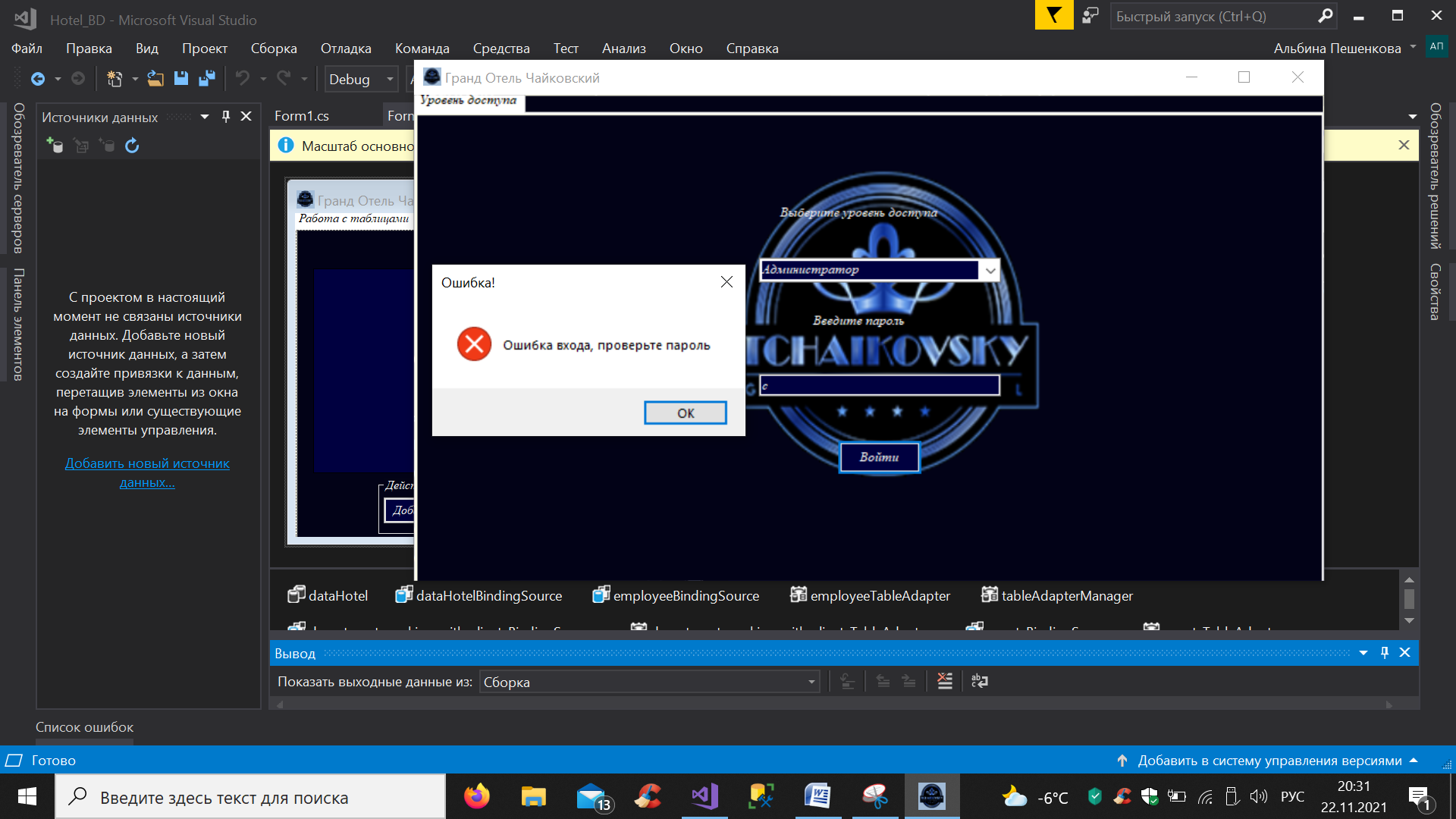


Рисунок 66 – окно ошибки

# ЗАКЛЮЧЕНИЕ

В ходе курсовой работы была разработана и спроектирована база данных "Отель”, содержащая необходимые данные, а также подключённая к пользовательскому интерфейсу. База данных разработана в программной системе разработки баз данных MS SQL Server Management Studio 18, что позволяет легко понять ее организацию и простоту управления. Получить необходимую информацию из базы данных можно, используя SQL – запросы или интерфейс, выполненный с помощью языка C# в среде программирования Visual Studio 2017.

Были закреплены навыки в работе с СУБД SQL Server, выполнено повторение процесса создания таблиц, установления связей между таблицами, разработки запросов, заполнения таблиц данными. Также в ходе курсовой работы был получен навык связывания базы данных с определенным кодом, работы с базой посредством языка C# и разработки пользовательского приложения.

Источники:

1. <http://openhospitality.org/blog/razdel-1/kategorii-nomerov-v-otele-vidy-tipy-i-rasshifrovka/>
2. [https://www.booking.com/hotel/ru/grand-otiel-chaikovskii-sankt-peterburg.ru.html?aid=392478;label=yandex-zV\*C\*WI8A8R2bd0KE\*mHpg-7827254018;sid=f589bf3f6fed435f68be4e29bf952d56;sig=v18s\_ccj\_1](https://www.booking.com/hotel/ru/grand-otiel-chaikovskii-sankt-peterburg.ru.html?aid=392478;label=yandex-zV*C*WI8A8R2bd0KE*mHpg-7827254018;sid=f589bf3f6fed435f68be4e29bf952d56;sig=v18s_ccj_1)
3. <https://revolution.allbest.ru/programming/00597528_0.html>
4. <https://depix.ru/articles/sistema_upravleniya_bazami_dannyh_mysql>
5. <https://puzat.ru/knowledge/mysql>
6. <https://oracle-patches.com/oracle/prof/%D1%81%D1%83%D0%B1%D0%B4-oracle-%D0%BE%D0%B1%D0%B7%D0%BE%D1%80-%D1%85%D0%B0%D1%80%D0%B0%D0%BA%D1%82%D0%B5%D1%80%D0%B8%D1%81%D1%82%D0%B8%D0%BA-%D0%B8-%D0%B2%D0%BE%D0%B7%D0%BC%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B5%D0%B9-%D0%B1%D0%B0%D0%B7%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85>
7. <http://www.winblog.ru/admin/1147770778-al15011701.html>
8. <https://drach.pro/blog/hi-tech/item/145-db-comparison>
9. <https://techarks.ru/qa/oracle/nedostatki-bazi-dannih-oracl-NL/>
10. <https://knowledge.allbest.ru/programming/3c0b65625a3bc79b4c53b88521206d27_0.html>
11. Майк Гандерлой, Джозеф Джорден, Дейвид Чанц. Освоение Microsoft SQL Server 2005 = Mastering Microsoft SQL Server 2005. — М.: «Диалектика», 2007. — С. 1104. — ISBN 0-7821-4380-6. 2. Microsoft SQL Server 2008.
12. Руководство для начинающих — Душан Петкович.
13. Ковязин, А.Н. Архитектура, администрирование и разработка приложений баз данных в InterBase/FireBird/Yaffil / А.Н. Ковязин, С.М. Востриков. - М.: Кудиц-образ; Издание 4-е, 2006

# Приложение

Текст файла Form1:

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.Data.SqlClient;

using System.Windows.Forms;

namespace Hotel\_BD

{

public partial class Form1 : Form

{

SqlConnection CONNECT;

string connection = @" Data Source= LAPTOP-VQTN067N\SQLEXPRESS; Initial catalog=Hotel; Integrated Security=True";

string chooseTable;

TabPage page\_Table;

TabPage page\_Access;

string passAdmin = "admin";

string passManagment = "managment";

string passSupport = "support";

string passGuest = "guests";

string writePass = "";

string[] admin = { "Сотрудники", "Отдел по работе с гостями", "Управляющий отдел", "Поддерживающий отдел" , "Номера отеля" , "Гости" , "Кухня", "Расписание питания" , "Должности" , "Закупки" , "Запросы на ремонт" };

string[] managment = { "Сотрудники", "Номера отеля", "Гости", "Кухня", "Расписание питания", "Должности", "Закупки", "Запросы на ремонт" };

string[] support = { "Запросы на ремонт" };

string[] guests = { "Отдел по работе с гостями", "Поддерживающий отдел", "Номера отеля", "Гости", "Расписание питания", "Запросы на ремонт" };

public BindingSource getChooseTable(string choose)

{

BindingSource nameTable = null;

switch (choose)

{

case "Сотрудники": nameTable = employeeBindingSource; break;

case "Отдел по работе с гостями": nameTable = department\_working\_with\_clientsBindingSource; break;

case "Управляющий отдел": nameTable = management\_departmentBindingSource; break;

case "Поддерживающий отдел": nameTable = supporting\_departmentBindingSource; break;

case "Номера отеля": nameTable = hotel\_roomBindingSource; break;

case "Гости": nameTable = guestsBindingSource; break;

case "Кухня": nameTable = kitchenBindingSource; break;

case "Расписание питания": nameTable = mealScheduleBindingSource; break;

case "Должности": nameTable = postBindingSource; break;

case "Закупки": nameTable = purchaseBindingSource; break;

case "Запросы на ремонт": nameTable = repair\_requestBindingSource; break;

}

return nameTable;

}

public Form1()

{

InitializeComponent();

table.Visible = false;

table.ReadOnly = true;

actons\_table.Visible = false;

page\_Table = tabControl1.TabPages[0];

page\_Access = tabControl1.TabPages[1];

tabControl1.TabPages.Remove(page\_Table);

}

private void choose\_table\_SelectedIndexChanged(object sender, EventArgs e)

{

chooseTable = choose\_table.Text;

table.Visible = true;

switch (accessEmployee.Text)

{

case "Администратор":

actons\_table.Visible = true;

break;

case "Сотрудник управляющего отдела":

actons\_table.Visible = true;

break;

case "Сотрудник отдела по работе с гостями":

actons\_table.Visible = true;

break;

case "Сотрудник поддерживающего отдела":

break;

}

CONNECT = new SqlConnection(connection);

CONNECT.Open();

table.DataSource = getChooseTable(chooseTable);

CONNECT.Close();

}

private void Form1\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Supporting\_department". При необходимости она может быть перемещена или удалена.

this.supporting\_departmentTableAdapter.Fill(this.dataHotel.Supporting\_department);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Repair\_request". При необходимости она может быть перемещена или удалена.

this.repair\_requestTableAdapter.Fill(this.dataHotel.Repair\_request);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Purchase". При необходимости она может быть перемещена или удалена.

this.purchaseTableAdapter.Fill(this.dataHotel.Purchase);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Post". При необходимости она может быть перемещена или удалена.

this.postTableAdapter.Fill(this.dataHotel.Post);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.MealSchedule". При необходимости она может быть перемещена или удалена.

this.mealScheduleTableAdapter.Fill(this.dataHotel.MealSchedule);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Management\_department". При необходимости она может быть перемещена или удалена.

this.management\_departmentTableAdapter.Fill(this.dataHotel.Management\_department);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Kitchen". При необходимости она может быть перемещена или удалена.

this.kitchenTableAdapter.Fill(this.dataHotel.Kitchen);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Hotel\_room". При необходимости она может быть перемещена или удалена.

this.hotel\_roomTableAdapter.Fill(this.dataHotel.Hotel\_room);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Guests". При необходимости она может быть перемещена или удалена.

this.guestsTableAdapter.Fill(this.dataHotel.Guests);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Department\_working\_with\_clients". При необходимости она может быть перемещена или удалена.

this.department\_working\_with\_clientsTableAdapter.Fill(this.dataHotel.Department\_working\_with\_clients);

// TODO: данная строка кода позволяет загрузить данные в таблицу "dataHotel.Employee". При необходимости она может быть перемещена или удалена.

this.employeeTableAdapter.Fill(this.dataHotel.Employee);

}

private void INSERT\_Click(object sender, EventArgs e) //Добавление строки

{

getChooseTable(chooseTable).AddNew();

table.ReadOnly = false;

}

private void UPDATE\_Click(object sender, EventArgs e) //Сохранение изменений

{

getChooseTable(chooseTable).EndEdit();

switch (chooseTable)

{

case "Сотрудники": employeeTableAdapter.Update(dataHotel); break;

case "Отдел по работе с гостями": department\_working\_with\_clientsTableAdapter.Update(dataHotel); break;

case "Управляющий отдел": management\_departmentTableAdapter.Update(dataHotel); break;

case "Поддерживающий отдел": supporting\_departmentTableAdapter.Update(dataHotel); break;

case "Номера отеля": hotel\_roomTableAdapter.Update(dataHotel); break;

case "Гости": guestsTableAdapter.Update(dataHotel); break;

case "Кухня": kitchenTableAdapter.Update(dataHotel); break;

case "Расписание питания": mealScheduleTableAdapter.Update(dataHotel); break;

case "Должности": postTableAdapter.Update(dataHotel); break;

case "Закупки": purchaseTableAdapter.Update(dataHotel); break;

case "Запросы на ремонт": repair\_requestTableAdapter.Update(dataHotel); break;

}

table.ReadOnly = true;

}

private void UPDATE\_Click\_1(object sender, EventArgs e) //Изменить

{

table.ReadOnly = false;

}

private void DELETE\_Click(object sender, EventArgs e) //Удаить

{

getChooseTable(chooseTable).RemoveCurrent();

getChooseTable(chooseTable).EndEdit();

switch (chooseTable)

{

case "Сотрудники": employeeTableAdapter.Update(dataHotel); break;

case "Отдел по работе с гостями": department\_working\_with\_clientsTableAdapter.Update(dataHotel); break;

case "Управляющий отдел": management\_departmentTableAdapter.Update(dataHotel); break;

case "Поддерживающий отдел": supporting\_departmentTableAdapter.Update(dataHotel); break;

case "Номера отеля": hotel\_roomTableAdapter.Update(dataHotel); break;

case "Гости": guestsTableAdapter.Update(dataHotel); break;

case "Кухня": kitchenTableAdapter.Update(dataHotel); break;

case "Расписание питания": mealScheduleTableAdapter.Update(dataHotel); break;

case "Должности": postTableAdapter.Update(dataHotel); break;

case "Закупки": purchaseTableAdapter.Update(dataHotel); break;

case "Запросы на ремонт": repair\_requestTableAdapter.Update(dataHotel); break;

}

}

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

{

string access = accessEmployee.Text;

writePass = password.Text;

switch (access)

{

case "Администратор":

if (writePass == passAdmin)

{

tabControl1.TabPages.Add(page\_Table);

for (int i = 0; i < admin.Length; i++)

choose\_table.Items.Add(admin[i]);

tabControl1.TabPages.Remove(page\_Access);

}

else {

MessageBox.Show("Ошибка входа, проверьте пароль", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

break;

case "Сотрудник управляющего отдела":

if (writePass == passManagment)

{

tabControl1.TabPages.Add(page\_Table);

for (int i = 0; i < managment.Length; i++)

choose\_table.Items.Add(managment[i]);

tabControl1.TabPages.Remove(page\_Access);

}

else

{

MessageBox.Show("Ошибка входа, проверьте пароль", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

break;

case "Сотрудник отдела по работе с гостями":

if (writePass == passGuest)

{

tabControl1.TabPages.Add(page\_Table);

for (int i = 0; i < guests.Length; i++)

choose\_table.Items.Add(guests[i]);

tabControl1.TabPages.Remove(page\_Access);

}

else

{

MessageBox.Show("Ошибка входа, проверьте пароль", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

break;

case "Сотрудник поддерживающего отдела":

if (writePass == passSupport)

{

tabControl1.TabPages.Add(page\_Table);

for (int i = 0; i < support.Length; i++)

choose\_table.Items.Add(support[i]);

tabControl1.TabPages.Remove(page\_Access);

}

else

{

MessageBox.Show("Ошибка входа, проверьте пароль", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

break;

}

accessNow.Text = access;

}

}

}