|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ ИМЕНИ К.Г. РАЗУМОВСКОГО (ПЕРВЫЙ КАЗАЧИЙ УНИВЕРСИТЕТ)»**  **(ФГБОУ ВО «МГУТУ ИМ. К.Г. РАЗУМОВСКОГО (ПКУ)»)** |
|  |
| **УНИВЕРСИТЕТСКИЙ КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ** |

|  |
| --- |
| **КУРСОВОЙ ПРОЕКТ** |
|  |
| по междисциплинарному курсу МДК.11.01 Технология разработки и защиты баз данных |
|  |
| на тему: Разработка базы данных для автоматизированного ведения и учёта финансовой документации по движению денежных средств в банке |
|  |
| студента группы 090207-9о-20/2  специальности 09.02.07 Информационные системы и программирование |
| Демёхина Даниил Валентиновича |

|  |  |  |
| --- | --- | --- |
| Студент |  | Д. В. Демёхин |
| Руководитель курсового проекта |  | В. А. Подобин |
| Председатель ПЦК специальности 09.02.07 Информационные системы и программирование |  | А. И. Глускер |

|  |  |  |
| --- | --- | --- |
| Дата защиты «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г. | | |
| Оценка:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| Заведующий отделением № 3 |  | И. Г. Дзюба |

Москва

2022

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc104587599)

[**1.** **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ** 5](#_Toc104587600)

[**1.1.** **Реляционная модель данных** 5](#_Toc104587601)

[**1.1.1.** **Понятие Реляционной модели данных** 5](#_Toc104587602)

[**1.1.2.** **Реляционные термины** 5](#_Toc104587603)

[**1.2.** **Логическая модель** 6](#_Toc104587604)

[**1.2.1.** **Основные требования** 6](#_Toc104587605)

[**1.3.** **Анализ предметной области** 7](#_Toc104587606)

[**1.3.1.** **Потенциальные таблицы** 7](#_Toc104587607)

[**2.** **ПРАКТИЧЕСКАЯ ЧАСТЬ** 11](#_Toc104587608)

[**2.1.** **Проектирование базы данных** 11](#_Toc104587609)

[**2.1.1.** **Построение логической модели** 11](#_Toc104587610)

[**2.1.2.** **Построение физической модели** 12](#_Toc104587611)

[**2.1.2.1.** **Характеристика таблиц базы данных** 12](#_Toc104587612)

[**2.2.** **Разработка информационной системы** 16](#_Toc104587613)

[**2.2.1.** **Создание окна авторизации** 16](#_Toc104587614)

[**2.2.2.** **Создание основной формы** 16](#_Toc104587615)

[**2.2.3.** **Создание дочерней формы для просмотра таблиц** 18](#_Toc104587616)

[**2.2.4.** **Создания окон добавления и удаления записей** 19](#_Toc104587617)

[**СПИСОК ЛИТЕРАТУРЫ** 21](#_Toc104587618)

[**ПРИЛОЖЕНИЕ А** 22](#_Toc104587619)

[**ПРИЛОЖЕНИЕ Б** 26](#_Toc104587620)

[**ПРИЛОЖЕНИЕ В** 31](#_Toc104587621)

[**ПРИЛОЖЕНИЕ Г** 45](#_Toc104587622)

[**ПРИЛОЖЕНИЕ Д** 49](#_Toc104587623)

[**ПРИЛОДЕНИЕ Е** 51](#_Toc104587624)

[**ПРИЛОДЕНИЕ Ё** 52](#_Toc104587625)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ФГБОУ ВО «МГУТУ им. К. Г. Разумовского (ПКУ)»-09.02.07-ИСП-207-06-2022-КП | | | |
|  |  |  |  |  |
| Изм. | Лист | ФИО | Подп. | Дата |
| Разраб | | Д.В. Демёхин |  |  | Разработка базы данных для автоматизированного ведения и учёта финансовой документации по движению денежных средств в банке | Литера | Лист | Листов |
| Руководит | | В.А. Подобин |  |  | КП | 2 | 53 |
|  | |  |  |  | УниКИТ  ИСП-207 | | |
|  | |  |  |  |
|  | |  |  |  |

# **ВВЕДЕНИЕ**

Информационные системы – один из самых необходимых элементов знания, который требует умения работать с большими объёмами информации в современном мире. Актуальность Информационных систем заключается в структуризации и упорядоченности необходимой информации.

Целью курсового проекта является разработка базы данных для учёта финансовых средств в банке и информационной системы для её эксплуатирования.

База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется [системой управления базами данных (СУБД)](https://www.oracle.com/cis/database/what-is-database/#WhatIsDBMS). Данные вместе с СУБД, а также приложения, которые с ними связаны, называются системой баз данных.

Предметная область в данной работе является деятельность банка. Её целью является разработка автоматизированной базы данных для учёта финансовых средств в банке. Прежде чем разработать базу данных, будет выполнен анализ предметной области. Создание базы данных (Далее БД) будет включать себе нижеперечисленные этапы:

* Концептуальное проектирование БД:

1. Сбор требований к данным;
2. Анализ требований к данным;
3. Редактирование требований к данным;

* Логической проектирование БД:

1. Преобразование требований к данным в структуры данных;

* Физическое проектирование БД:

1. Определение особенностей хранения данных, метода доступа и т.д.

Информационная система должна будет выполнять чтение, редакцию, удаление и добавление данных БД. Для её разработки выбран язык программирования C#.

За систему управления базами данных (СУБД) в этом проекте будет MS SQL MANAGEMENT STUDIO 2022, а платформой разработки информационной системы взята MS VISUAL STUDIO 2022.

# **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

* 1. **Реляционная модель данных**
     1. **Понятие Реляционной модели данных**

Реляционной называется база данных, к которой все данные, допустимые пользователю, организованы в виде таблиц, а все операции базы данных выполняются нам этими таблицами.

В реляционной базе данных информация хранится организована в виде таблиц, разделённых на строки и столбцы, на пересечении которых содержатся значения данных

* + 1. **Реляционные термины**

База данных – набор объектов.

Схема базы данных – набор таблиц и связей между ними.

Тело отношения – совокупность заголовков и столбцов.

Атрибут отношения – наименование столбца таблицы.

Домен – Совокупность допустимых значений.

Кортеж отношений – строка, запись.

Степень отношений – количество столбцов отношений.

Первичный ключ (Primary Key) – минимальный набор атрибутов**,** который единственным образом идентифицирует кортеж отношения.

Внешний ключ (Foreign Key) – атрибут или набор атрибутов внутри отношения, которые соответствуют первичному ключу некоторого отношения.

Отношение (Relationship) – связь между первичным ключом одной таблицы к внешнему ключу другой таблицы.

* 1. **Логическая модель** 
     1. **Основные требования**

Логическая модель должна отображать все сущности и связи, значимые для той цели, ради которой её проектируют.

Сущность – это отдельный элемент деятельности организации, например, клиенты или сотрудник, который должен быть представлен в БД.

Все объекты модели должны быть именованы. Именование сущностей и связей должны выполняться в терминах предметной области.

Для связей необходимо указать кратность (один ко многим, один к одному).

Модель должна читаться по схеме:

**<**Сущность 1**> - <**Отношение / связь**> - <**Сущность 2**>**.

Модель должна быть структурирована, сущности должны быть сгруппированы по логическому смыслу.

Расположение объектов модели недолжно препятствовать её чтению.

* 1. **Анализ предметной области**
     1. **Потенциальные таблицы**

Автоматизация информационной системы (ИС) «Банка» позволяет хранить данные в единой БД, а также управлять потоками данных в системе, основываясь на единых принципах и методах, позволяющие выполнять различные манипуляции над данными.

Основные функции банковской деятельности является выдача кредитов, дебетовых карт и открытие вкладов. Для учёта денежных средств будет необходимо учитывать личные данные клиентов и их договора, так как один клиент может оформить несколько банковских услуг, будет учитываться возможность хранения нескольких договоров на одного клиента. Также каждый клиент может иметь по несколько открытых счетов, поэтому также, как и договора, счета клиентов будут отделены от самых клиентов в другой таблицу, но будут связанны между собой отношением (Relation). Данные о клиентах и их история оформленных услуг поделится на:

* Личные данные клиентов, имеющие следующие записи:

1. Номер клиента;
2. Имя клиента;
3. Фамилия клиента;
4. Отчество клиента;
5. Дата рождения;
6. Серия и номер паспорта;
7. Фактический адрес;
8. Адрес регистрации;
9. Email;
10. Телефон;
11. Пол;

* Договора клиентов:

1. Номер договора;
2. Номер клиента;
3. Дата заключения;
4. Дата окончания;
5. Номер счёта;
6. Номер сотрудника;

* История операций:

1. Номер операции;
2. Код операции;
3. Номер счёта;
4. Сумма операции;

* Счета клиентов:

1. Номер счёта;
2. Номер клиента.

Так как в каждом договоре клиента будет прописан сотрудник банка, продающий услуги банка клиенту, БД будет иметь:

* Личные данные сотрудников, в которых необходимо записывать:

1. Номер сотрудника;
2. Имя сотрудника;
3. Фамилия сотрудника;
4. Отчество сотрудника;
5. Дата рождения;
6. Серия и номер паспорта;
7. Фактический адрес;
8. Адрес регистрации;
9. Email;
10. Телефон;
11. Пол;

* Специальность:

1. Номер отдела;
2. Код специальности:
3. Наименование специальности;
4. Минимальный должностной оклад;

Так как в каждый договор невозможно будет записать Оформленные кредиты, вклады, дебетовые карты из-за разных атрибутов, придется их разделить на три разных таблицы:

* Оформленные кредиты, состоящие из:

1. Номер пластика;
2. Код услуги;
3. Номер договора;
4. Сумма;
5. ПИН код;

* Оформленные вклады:

1. Номер вклада;
2. Код услуги;
3. Номер договора;
4. Сумма;

* Оформленные дебетовые карты:

1. Номер пластика;
2. Номер договора;
3. Код услуги;
4. ПИН код.

Нельзя забывать, что в банках есть различные предложения по различным услугам, поэтому они будут записаны в разные категории, у которых будут разные. И, для объединения всех услуг, надо добавить вспомогательную таблицу в которой будут прописаны весь набор услуг, который предоставляет банк:

* Виды кредитов:

1. Код услуги;
2. Минимальный срок кредита;
3. Максимальные срок кредита;
4. Проценты;
5. Минимальная сумма;
6. Максимальная сумма;

* Виды вкладов:

1. Код услуги;
2. Минимальная сумма;
3. Максимальная сумма;
4. Неснижаемый остаток;
5. Срок;
6. Проценты;
7. Проценты с капитализацией;
8. Минимальный возраст;
9. Валюта;

* Виды дебетовых карт:

1. Код услуги
2. Стоимость обслуживания
3. Минимальный возраст
4. Максимальный возраст
5. КЭШБЭК;

* Банковские услуги:

1. Номер услуги;
2. Код услуги;
3. Название услуги.
4. **ПРАКТИЧЕСКАЯ ЧАСТЬ**
   1. **Проектирование базы данных**
      1. **Построение логической модели**

|  |  |  |
| --- | --- | --- |
| Сущность 1 | Отношение | Сущность 2 |
| КЛИЕНТЫ | ОДИН КО МНОГИМ | ДОГОВОРЫ\_КЛИЕНТОВ |
| СОТРУДНИКИ | ОДИН КО МНОГИМ | ДОГОВОРЫ\_КЛИЕНТОВ |
| КЛИЕНТЫ | ОДИН КО МНОГИМ | СЧЕТА |
| СЧЕТА | ОДИН КО МНОГИМ | ИСТРОРИЯ\_ОПЕРАЦИЙ |
| ВИДЫ\_ОПЕРАЦИИ | ОДИН КО МНОГИМ | ИСТРОРИЯ\_ОПЕРАЦИЙ |
| СОТРУДНИКИ | ОДИН К ОДНОМУ | ДАННЫЕ\_ВХОДА |
| СЧЕТА | ОДИН КО МНОГИМ | ДОГОВОРЫ\_КЛИЕНТОВ |
| ДЕБ\_КАРТЫ | ОДИН К ОДНОМУ | БАНКОВСКИЕ\_УСЛУГИ |
| ВИДЫ\_ВКЛАДОВ | ОДИН К ОДНОМУ | БАНКОВСКИЕ\_УСЛУГИ |
| ВИДЫ\_КРЕДИТОВ | ОДИН К ОДНОМУ | БАНКОВСКИЕ\_УСЛУГИ |
| ДЕБ\_КАРТЫ | ОДИН КО МНОГИМ | ОФОРМЛЕННЫЕ\_ДЕБ\_КАРТЫ |
| ВИДЫ\_ВКЛАДОВ | ОДИН КО МНОГИМ | ОФОРМЛЕННЫЕ\_ВКЛАДЫ |
| ВИДЫ\_КРЕДИТОВ | ОДИН КО МНОГИМ | ОФОРМЛЕННЫЕ\_КРЕДИТЫ |
| ДОГОВОРЫ\_КЛИЕНТОВ | ОДИН К ОДНОМУ | ОФОРМЛЕННЫЕ\_ДЕБ\_КАРТЫ |
| ДОГОВОРЫ\_КЛИЕНТОВ | ОДИН К ОДНОМУ | ОФОРМЛЕННЫЕ\_ВКЛАДЫ |
| ДОГОВОРЫ\_КЛИЕНТОВ | ОДИН К ОДНОМУ | ОФОРМЛЕННЫЕ\_КРЕДИТЫ |

****

* + 1. **Построение физической модели**
       1. **Характеристика таблиц базы данных**

Таблица 1 Структура Таблицы "БАНКОВСКИЕ\_УСЛУГИ"

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Нумерация услуги | НОМЕР\_УСЛУГИ | int |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Название | НАЗВАНИЕ | varchar(75) |

Таблица 2Структура Таблицы "ВИДЫ\_ВКЛАДОВ"

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Минимальная сумма вклада | МИНИМАЛЬНАЯ\_СУММА | money |
| Максимальная сумма вклада | МАКСИМАЛЬНАЯ\_СУММА | money |
| Неснижаемый остаток | НЕСНИЖАЕМЫЙ\_ОСТАТОК | money |
| Срок вклада | СРОК\_В\_НЕДЕЛЯХ | int |
| Проценты по вкладу | ПРОЦЕНТЫ | decimal(5, 2) |
| Проценты с капитализацией | С\_КАПИТАЛИЗАЦИЕЙ | decimal(5, 2) |
| Минимальный возраст | МИНИМАЛЬНЫЙ\_ВОЗРАСТ | int |
| Валюта вклада | ВАЛЮТА | varchar(20) |

Таблица 3Структура Таблицы "ВИДЫ\_ДЕБЕТОВЫХ\_КАРТ"

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Стоимость обслуживания | ОБСЛУЖИВАНИЕ\_В\_МЕСЯЦ | money |
| Минимальный возраст | МИНИМАЛЬНЫЙ\_ВОЗРАСТ | int |
| Максимальный возраст | МАКСИМАЛЬНЫЙ\_ВОЗРАСТ | int |
| Начисляемый кэшбэк в процентах | КЭШБЭК | decimal(5, 2) |

Таблица 4 Структура таблицы «ВИДЫ\_КРЕДИТОВ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Минимальный срок кредита | МИНИМАЛЬНЫЙ\_СРОК\_КРЕДИТА | int |
| Максимальный срок кредита | МАКСИМАЛЬНЫЙ\_СРОК\_КРЕДИТА | int |
| ПРОЦЕНТЫ кредиту | ПРОЦЕНТЫ | decimal(5, 2) |
| МИНИМАЛЬНАЯ СУММА взятия кредита | МИНИМАЛЬНАЯ\_СУММА | money |
| МАКСИМАЛЬНАЯ\_СУММА взятия кредита | МАКСИМАЛЬНАЯ\_СУММА | money |

Таблица 5 Структура таблицы «ВИДЫ\_ОПЕРАЦИИ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Id операции | КОД\_ОПЕРАЦИИ | varchar(3) |
| Вид операции | ВИД\_ОПЕРАЦИИ | varchar(75) |

Таблица 6 Структура таблицы «ДАННЫЕ\_ВХОДА»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Login сотрудника | LOGIN | varchar(25) |
| Пароль сотрудника | ПАРОЛЬ | varchar(30) |
| Номер сотрудника | НОМЕР\_СОТРУДНИКА | int |

Таблица 7 Структура таблицы «ДОГОВОРЫ\_КЛИЕНТОВ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Номер договора | НОМЕР\_ДОГОВОРА | int |
| Номер клиента | НОМЕР\_КЛИЕНТА | int |
| Дата заключения договора | ДАТА\_ЗАКЛЮЧЕНИЯ | date |
| Дата окончания договора | ДАТА\_ОКОНЧАНИЯ | date |
| Счёт клиента | НОМЕР\_СЧЁТА | varchar(20) |
| Номер сотрудника | НОМЕР\_СОТРУДНИКА | int |

Таблица 8 Структура таблицы «ИСТОРИЯ\_ОПЕРАЦИЙ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Нумерация операции | НОМЕР\_ОПЕРАЦИИ | int |
| Id операции | КОД\_ОПЕРАЦИИ | varchar(3) |
| Счёт клиента | НОМЕР\_СЧЁТА | varchar(20) |
| Сумма операции | СУММА\_ОПЕРАЦИИ | money |

Таблица 9 Структура таблицы «КЛИЕНТЫ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Номерация клиента | НОМЕР\_КЛИЕНТА | int |
| Фамилия клиента | ФАМИЛИЯ | char(50) |
| Имя клиента | ИМЯ | char(30) |
| Отчество клиента | ОТЧЕСТВО | char(30) |
| Телефон клиента | ТЕЛЕФОН | varchar(50) |
| Серия и номер паспорта клиента | СЕРИЯ\_НОМЕР | varchar(10) |
| Пол клиента | ПОЛ | char(1) |
| Адрес регистрации клиента | АДРЕС\_РЕГИСТРАЦИИ | varchar(150) |
| Email клиента | EMAIL | varchar(50) |
| Фактический адрес клиента | ФАКТИЧЕСКИЙ\_АДРЕС | varchar(150) |
| Дата рождения клиента | ДАТА\_РОЖДЕНИЯ | date |

Таблица 10 Структура таблицы «ОФОРМЛЕННЫЕ\_ВКЛАДЫ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Нумерация вклада | НОМЕР\_ВКЛАДА | int |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Номер договора | НОМЕР\_ДОГОВОРА | int |
| Сумма вклада | СУММА | money |

Таблица 11 Структура таблицы «ОФОРМЛЕННЫЕ\_ДЕБЕТОВЫЕ\_КАРТЫ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Номер дебетовой карты | НОМЕР\_ПЛАСТИКА | varchar(16) |
| Номер договора | НОМЕР\_ДОГОВОРА | int |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Пароль от карты | ПИН\_КОД | varchar(4) |

Таблица 12 Структура таблицы «ОФОРМЛЕННЫЕ\_КРЕДИТЫ»

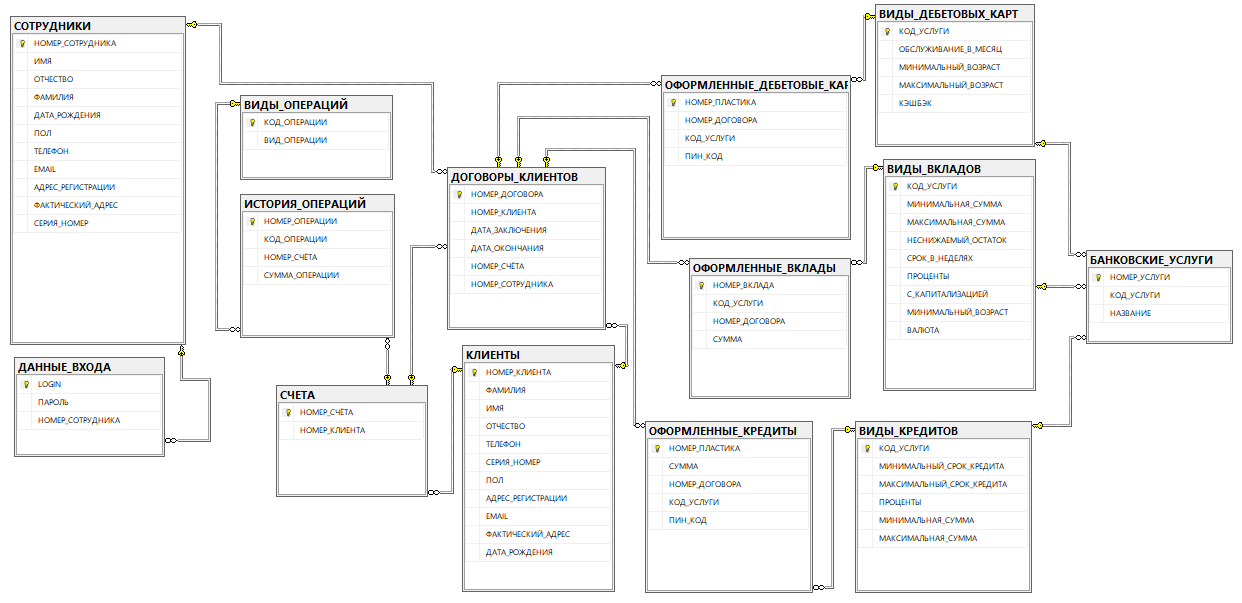
|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Номер КРЕДИТНОЙ карты | НОМЕР\_ПЛАСТИКА | varchar(16) |
| СУММА КРЕДИТА | СУММА | money |
| НОМЕР ДОГОВОРА | НОМЕР\_ДОГОВОРА | int |
| ID услуги | КОД\_УСЛУГИ | varchar(3) |
| Пароль от карты | ПИН\_КОД | varchar(4) |

Таблица 13 Структура таблицы «СОТРУДНИКИ»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Номер сотрудника | НОМЕР\_СОТРУДНИКА | int |
| Имя сотрудника | ИМЯ | varchar(40) |
| Отчество сотрудника | ОТЧЕСТВО | varchar(40) |
| Фамилия сотрудника | ФАМИЛИЯ | varchar(40) |
| Дата рождения сотрудника | ДАТА\_РОЖДЕНИЯ | date |
| Пол сотрудника | ПОЛ | char(1) |
| Телефон сотрудника | ТЕЛЕФОН | varchar(11) |
| Email сотрудника | EMAIL | varchar(50) |
| Адрес регистрации сотрудника | АДРЕС\_РЕГИСТРАЦИИ | varchar(150) |
| Фактический адрес сотрудника | ФАКТИЧЕСКИЙ\_АДРЕС | varchar(150) |
| Серия и номер паспорта сотрудника | СЕРИЯ\_НОМЕР | varchar(10) |

Таблица 14 Структура таблицы «СЧЕТА»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Идентификатор | Тип данных |
| Счёта клиента | НОМЕР\_СЧЁТА | varchar(20) |
| Нумерация клиента | НОМЕР\_КЛИЕНТА | int |

****

* 1. **Разработка информационной системы**
     1. **Создание окна авторизации**

Для окна авторизации будет применена таблица из БД, в которой записаны Логин и Пароль сотрудника. Авторизация будет происходить путём выборки строк данных из этой таблицы, у которых будет сходится ячейка из столбца «Логин» и набранным пользователем логином, а также будет проводиться рассмотрение схожести ячеек столбца «ПАРОЛЬ» с соответствующей записью пользователя в элементе «Textbox».

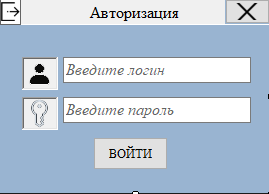


Рисунок 1Форма авторизации "FormLogin"

* + 1. **Создание основной формы**

Основная форма будет доступна для управления сразу после авторизации пользователя. В Основной форме представлено верхняя панель меню, где можно развернуть раздел «Данные», и выбрать нужную область данных. Область данных, которая предоставляется пользователю:

* Раздел «Сотрудники»:

1. Таблица «СОТРУДНИКИ»;
2. Таблица «ДАННЫЕ\_ВХОДА» - страница ControlBox’а «Пользователи»;

* Раздел «Клиенты»:

1. Таблица «КЛИЕНТЫ»;
2. Таблица «СЧЕТА»;
3. Таблица «ИСТОРИЯ\_ОПЕРАЦИЙ»;
4. Таблица «ДОГОВОРЫ\_КЛИЕНТОВ»;

* Раздел «Банковские услуги»:

1. Таблица «ВИДЫ\_КРЕДИТОВ»;
2. Таблица «ВИДЫ\_ВКЛАДОВ»;
3. Таблица «ВИДЫ\_ДЕБЕТОВЫХ\_КАРТ»;

* Раздел «Оформленные банковские услуги»:

1. Таблица «БАНКОВСКИЕ\_УСЛУГИ»;
2. Таблица «ОФОРМЛЕННЫЕ\_ВКЛАДЫ»;
3. Таблица «ОФОРМЛЕННЫЕ\_ДЕБЕТОВЫЕ\_КАРТЫ»;
4. Таблица «ОФОРМЛЕННЫЕ\_КРЕДИТЫ».

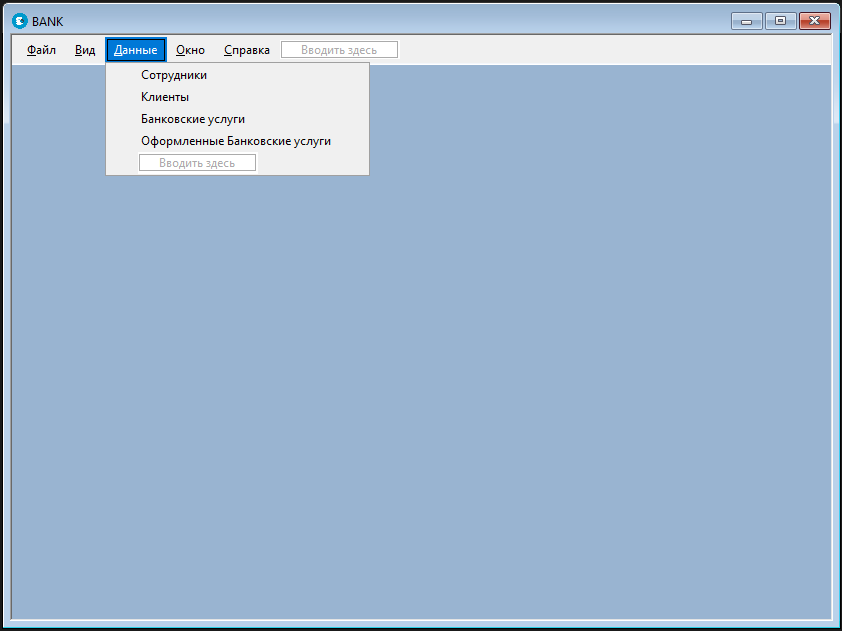


Рисунок 2 Основная форма "FormMain"

* + 1. **Создание дочерней формы для просмотра таблиц**

Дочерняя форма должная быть удобной для просмотра информации и возможности поиска по определённым критериям этой информации. Отображать данные из таблицы будет элемент «DataGridView», с помощью запроса на вывод всех данных из таблицы. Учитывая необходимость удаления и добавления записей, будут элементы управления «Button» - служат кнопками. Кнопка «Удалить» будет вызывать форму в которой пользователь сможет выбрать какую строку ему надо удалить, а для вызова окна добавления будет кнопка «добавить».

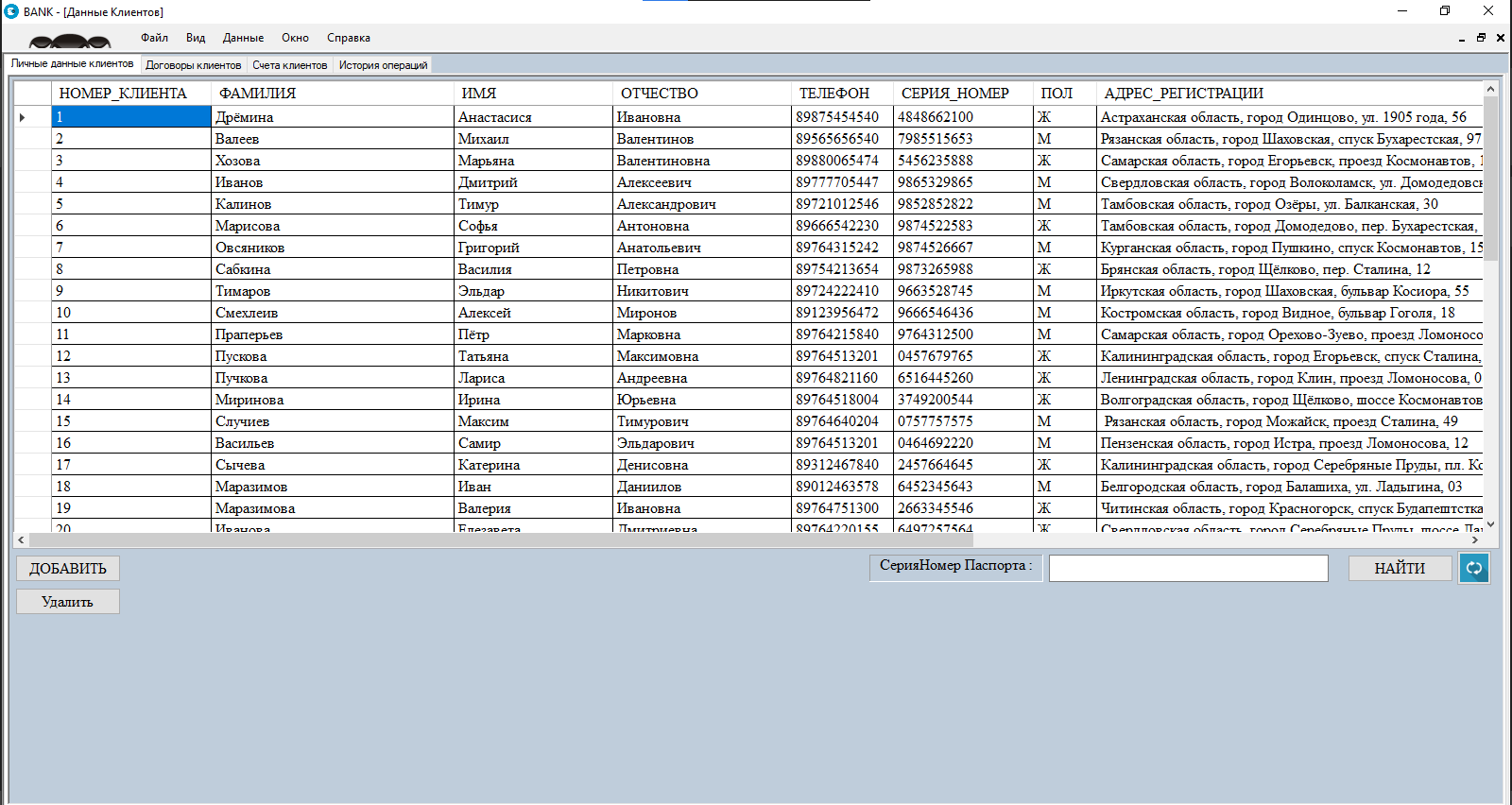


Рисунок 3 Дочерняя форма "tablesWorks"

* + 1. **Создания окон добавления и удаления записей**

Чтобы осуществить добавление записей введённых пользователем будет использован запрос SQL на добавление записей, вставляющий в себя эти данные.

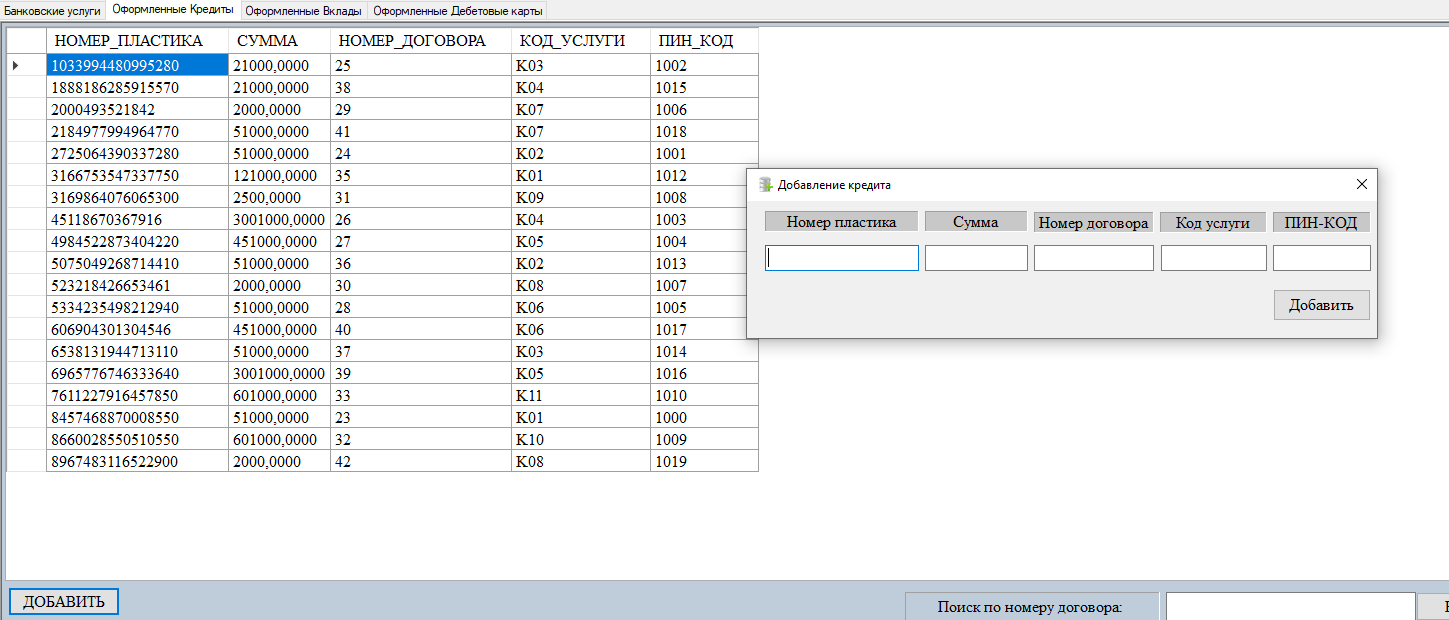


Рисунок 4 Форма добавления записей "AddCredit"

Для удаления строки используется элемент «ComboBox», из него вылезет список в котором можно выбрать необходимую строку.

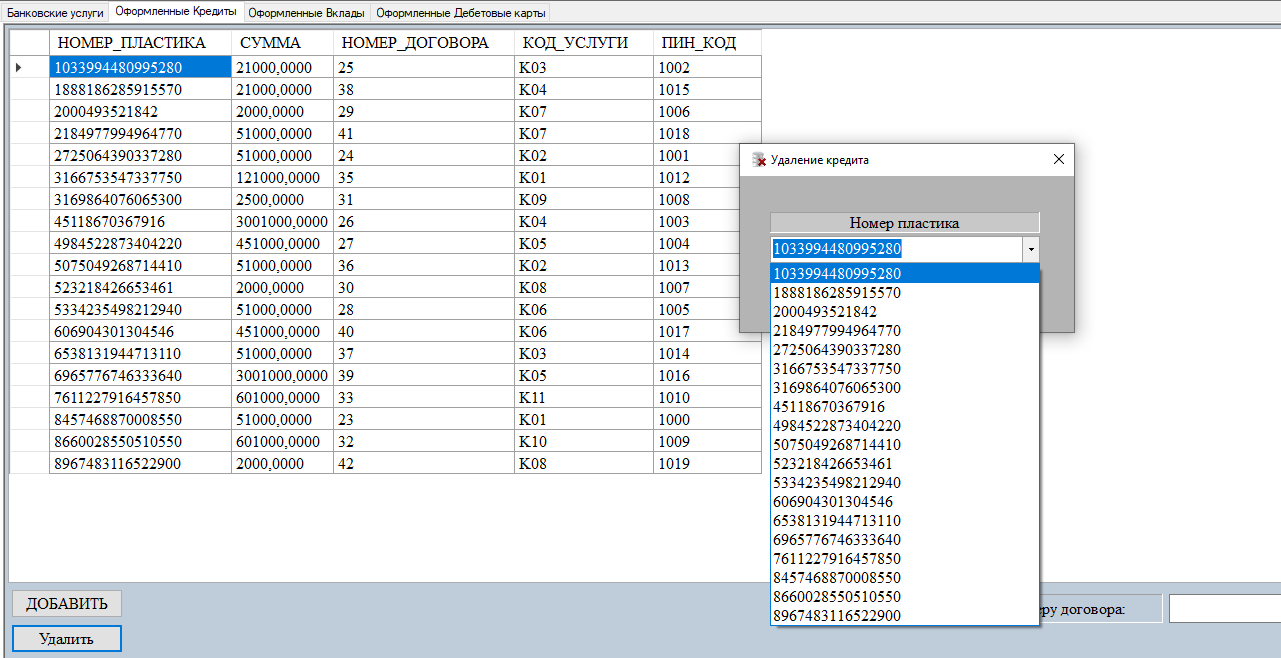


Рисунок 5 Форма удаления "DeleteCredit"

# **СПИСОК ЛИТЕРАТУРЫ**

1. Щеглов, Ю.А. Информационные системы и процессы /Ю.А. Щеглов, д.т.н. - Новосибирск: НИНХ, 2018. - 251 с.
2. Задорожный, В.Н. Информационные технологии и автоматизация управления / В. Н. Задорожный. - Омск : Изд-во ОмГТУ, 2019. - 269 с.
3. Баранов В. В., Горошко И. В., Лебедев В. Н. Информационные технологии управления и организация защиты информации: учебник / В. В. Баранов, И. В. Горошко, В. Н. Лебедев и др. - Москва: Академия управления МВД России, 2018. - 453 с.
4. А.Д. Хомоненко, В.М. Цыганков, М.Г. Мальцев

Базы данных: Учебник для высших учебных заведений / Под реф. Проф. А.Д. Хомоненко – 4-е издание, доп. и перераб. – СПБ.: КОРОНА принт, 2004. – 736 с.

1. Базы данных: учебник / И.А. Кумскова – 3-е издание, перераб. – Москва: КНОРУС, 2020. – 400 с. – (Среднее профессиональное образование)
2. Иванова Л. Н. Проектная деятельность в управлении персоналом: учебное пособие / Л. Н. Иванова. - Новосибирск: Сибирский институт управления - филиал РАНХиГС, 2019. – 197с.
3. Базаров Т. Ю. Управление персоналом : учебник / Т.Ю. Базаров. - 15-е изд., стер. - Москва : Академия, 2018. – 314с.

# **ПРИЛОЖЕНИЕ А**

Листинг Формы авторизации «FormLogin»**:**

namespace BANK

{

public partial class FormLogin : Form

{

Form form = new MainForm();

public FormLogin()

{

InitializeComponent();

CenterToScreen();

}

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

{

form.Enabled = false;

form.Show();

try

{

MyConnection.OpenConnection();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: Подключение отсутствует!" + ex);

}

}

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

{

DataTable table = new DataTable();

SqlDataAdapter adapter = new SqlDataAdapter();

SqlCommand command = new SqlCommand();

command.CommandText = @"select \* FROM ДАННЫЕ\_ВХОДА WHERE ((LEFT(LOGIN,5))) = 'ADMIN' AND [LOGIN] = @log AND [ПАРОЛЬ] = @pass";

command.Parameters.Add("@log", SqlDbType.VarChar).Value = user.Text;

command.Parameters.Add("@pass", SqlDbType.VarChar).Value = userword.Text;

table.Clear();

command.Connection = MyConnection.Connection;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

this.Close();

form.Enabled = true;

MyElements.LOCK = true;

}

else

{

command.CommandText = null;

command.CommandText = @"SELECT \* FROM ДАННЫЕ\_ВХОДА where [LOGIN] = @log AND [ПАРОЛЬ] = @pass";

command.Connection = MyConnection.Connection;

table.Clear();

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

this.Close();

form.Enabled = true;

}

else MessageBox.Show("Неправильный логин или пароль");

}

}

private void button2\_Click\_1(object sender, EventArgs e)

{

this.Close();

Application.Exit();

}

private void userword\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

in\_log.PerformClick();

}

}

private void user\_PreviewKeyDown(object sender, PreviewKeyDownEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

userword.Focus();

}

}

private void userword\_TextChanged(object sender, EventArgs e)

{

userword.PasswordChar = '\*';

}

private void userword\_MouseUp(object sender, MouseEventArgs e)

{

string b = userword.Text;

if (b == "Введите пароль")

{

userword.Text = "";

}

}

private void user\_MouseUp(object sender, MouseEventArgs e)

{

string a = user.Text;

if (a == "Введите логин")

{

user.Text = "";

}

}

}

}

# **ПРИЛОЖЕНИЕ Б**

Листинг основной формы «FormMain»:

namespace BANK

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

this.WindowState = FormWindowState.Maximized;

this.IsMdiContainer = true;

}

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

{

if (this.Enabled)

{

if (MyElements.LOCK)

{

MessageBox.Show("Вы вошли как Администратор!", "Вход выполнен");

}

else

MessageBox.Show("Авторизация прошла успешно!", "Вход выполнен");

}

}

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

{

DialogResult result = MessageBox.Show(

"Вы действительно хотите выйти?",

"Выход из приложения",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question,

MessageBoxDefaultButton.Button1

);

switch (result)

{

case DialogResult.Yes:

Cursor.Current = Cursors.WaitCursor;

Application.Exit();

break;

case DialogResult.No:

break;

}

}

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

{

try

{

do

{

MdiChildren[0].Close();

} while (MdiChildren.Count() > 0);

}

catch {

MessageBox.Show("Все окна закрыты");

}

}

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

{

this.WindowState = FormWindowState.Minimized;

}

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

{

if (this.WindowState == FormWindowState.Normal)

{

this.WindowState = FormWindowState.Maximized;

}else

this.WindowState = FormWindowState.Normal;

}

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

{

colorDialog1.ShowDialog();

}

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

{

fontDialog1.ShowDialog();

}

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

{

Form form = new tablesdealsform();

form.MdiParent = this;

form.WindowState = FormWindowState.Maximized;

form.Show();

}

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

{

Form form = new tablesTypesdeals();

form.MdiParent = this;

form.WindowState = FormWindowState.Maximized;

form.Show();

}

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

{

Form form = new tablesWorks();

form.MdiParent = this;

form.WindowState = FormWindowState.Maximized;

form.Show();

}

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

{

Form form = new tablesClients();

form.MdiParent = this;

form.WindowState = FormWindowState.Maximized;

form.Show();

}

}

}

# **ПРИЛОЖЕНИЕ В**

Листинг Формы дочернего окна «tablesdealsform»:

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 BANK

{

public partial class tablesdealsform : Form

{

SqlDataAdapter adapter = new SqlDataAdapter();

string selectdeal = @"SELECT \* FROM [БАНКОВСКИЕ\_УСЛУГИ]";

string selectcredit = @"SELECT \* FROM [ОФОРМЛЕННЫЕ\_КРЕДИТЫ]";

string selectvclads = @"SELECT \* FROM [ОФОРМЛЕННЫЕ\_ВКЛАДЫ]";

string selectcards = @"SELECT \* FROM [ОФОРМЛЕННЫЕ\_ДЕБЕТОВЫЕ\_КАРТЫ]";

private void UpdateDeals()

{

adapter.SelectCommand = null;

if (dataGridViewdealse.Rows.Count > 0)

{

MyElements.tablefordeals.Clear();

}

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = selectdeal;

sqlCommand.Connection = MyConnection.Connection;

adapter.SelectCommand = sqlCommand;

adapter.Fill(MyElements.tablefordeals);

dataGridViewdealse.DataSource = MyElements.tablefordeals;

}

private void UpdateCredits()

{

adapter.SelectCommand = null;

if (dataGridViewCredits.Rows.Count > 0)

{

MyElements.tableforCreadits.Clear();

}

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = selectcredit;

sqlCommand.Connection = MyConnection.Connection;

adapter.SelectCommand = sqlCommand;

adapter.Fill(MyElements.tableforCreadits);

dataGridViewCredits.DataSource = MyElements.tableforCreadits;

}

private void UpdateVclads()

{

adapter.SelectCommand = null;

if (dataGridViewVclads.Rows.Count > 0)

{

MyElements.tableforVclads.Clear();

}

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = selectvclads;

sqlCommand.Connection = MyConnection.Connection;

adapter.SelectCommand = sqlCommand;

adapter.Fill(MyElements.tableforVclads);

dataGridViewVclads.DataSource = MyElements.tableforVclads;

}

private void UpdateCards()

{

adapter.SelectCommand = null;

if (dataGridViewCards.Rows.Count > 0)

{

MyElements.tableforCards.Clear();

}

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = selectcards;

sqlCommand.Connection = MyConnection.Connection;

adapter.SelectCommand = sqlCommand;

adapter.Fill(MyElements.tableforCards);

dataGridViewCards.DataSource = MyElements.tableforCards;

}

public tablesdealsform()

{

InitializeComponent();

}

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

{

UpdateDeals();

UpdateCredits();

UpdateVclads();

UpdateCards();

}

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

{

adapter.SelectCommand = null;

string sel = select\_deals\_text.Text;

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = @"SELECT \* FROM [БАНКОВСКИЕ\_УСЛУГИ] WHERE [НАЗВАНИЕ] = @sel";

sqlCommand.Connection = MyConnection.Connection;

sqlCommand.Parameters.Add("@sel", SqlDbType.VarChar).Value = sel;

adapter.SelectCommand = sqlCommand;

MyElements.tablefordeals.Clear();

adapter.Fill(MyElements.tablefordeals);

if (MyElements.tablefordeals.Rows.Count < 1)

{

MessageBox.Show("Услуги с таким названием не существует");

}

}

private void button2\_Click(object sender, EventArgs e)

{

UpdateDeals();

}

private void button3\_Click(object sender, EventArgs e)

{

Form add = new AddDeal();

add.StartPosition = FormStartPosition.CenterScreen;

add.Size = new Size(655, 185);

add.Show();

}

private void button4\_Click(object sender, EventArgs e)

{

UpdateDeals();

}

private void button5\_Click(object sender, EventArgs e)

{

Form form = new DeleteDeal();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

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

{

try

{

adapter.SelectCommand = null;

string sel = textCredits.Text;

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = @"SELECT \* FROM [ОФОРМЛЕННЫЕ\_КРЕДИТЫ] WHERE [НОМЕР\_ДОГОВОРА] = @sel";

sqlCommand.Connection = MyConnection.Connection;

sqlCommand.Parameters.Add("@sel", SqlDbType.Int).Value = textCredits.Text;

adapter.SelectCommand = sqlCommand;

MyElements.tableforCreadits.Clear();

adapter.Fill(MyElements.tableforCreadits);

}

catch

{

if (MyElements.tableforCreadits.Rows.Count < 1)

{

MessageBox.Show("Услуги с таким названием не существует");

}

}

}

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

{

UpdateCredits();

}

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

{

UpdateCredits();

}

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

{

Form form = new creditAdd();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

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

{

Form form = new DeleteCredit();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

private void button4\_Click\_1(object sender, EventArgs e)

{

UpdateVclads();

}

private void button2\_Click\_1(object sender, EventArgs e)

{

UpdateVclads();

}

private void button5\_Click\_1(object sender, EventArgs e)

{

try

{

adapter.SelectCommand = null;

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = @"SELECT \* FROM [ОФОРМЛЕННЫЕ\_ВКЛАДЫ] WHERE [НОМЕР\_ДОГОВОРА] = @sel";

sqlCommand.Connection = MyConnection.Connection;

sqlCommand.Parameters.Add("@sel", SqlDbType.Int).Value = textBoxVclad.Text;

adapter.SelectCommand = sqlCommand;

MyElements.tableforVclads.Clear();

adapter.Fill(MyElements.tableforVclads);

}

catch

{

if (MyElements.tableforVclads.Rows.Count < 1)

{

MessageBox.Show("Услуги с таким названием не существует");

}

}

}

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

{

Form form = new AddVclad();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

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

{

Form form = new DeleteVclads();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

private void button5\_Click\_2(object sender, EventArgs e)

{

try

{

adapter.SelectCommand = null;

string sel = textCredits.Text;

SqlCommand sqlCommand = new SqlCommand();

sqlCommand.CommandText = @"SELECT \* FROM [ОФОРМЛЕННЫЕ\_ДЕБЕТОВЫЕ\_КАРТЫ] WHERE [НОМЕР\_ДОГОВОРА] = @sel";

sqlCommand.Connection = MyConnection.Connection;

sqlCommand.Parameters.Add("@sel", SqlDbType.Int).Value = textBoxCards.Text;

adapter.SelectCommand = sqlCommand;

MyElements.tableforCards.Clear();

adapter.Fill(MyElements.tableforCards);

}

catch

{

if (MyElements.tableforCreadits.Rows.Count < 1)

{

MessageBox.Show("Услуги с таким названием не существует");

}

}

}

private void button4\_Click\_2(object sender, EventArgs e)

{

UpdateCards();

}

private void button2\_Click\_2(object sender, EventArgs e)

{

UpdateCards();

}

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

{

Form form = new AddCards();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

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

{

Form form = new DeleteCard();

form.StartPosition = FormStartPosition.CenterScreen;

form.Show();

}

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

{

UpdateDeals();

select\_deals\_text.Text = null;

}

private void button1\_Click\_1(object sender, EventArgs e)

{

UpdateCredits();

textCredits.Text = null;

}

private void button2\_Click\_3(object sender, EventArgs e)

{

UpdateVclads();

textBoxVclad.Text = null;

}

private void button3\_Click\_1(object sender, EventArgs e)

{

UpdateCards();

textBoxCards.Text = null;

}

}

}

# **ПРИЛОЖЕНИЕ Г**

Листинг Формы окна добавления данных «AddClients»:

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 BANK

{

public partial class AddClients : Form

{

SqlDataAdapter sqlData = new SqlDataAdapter();

SqlCommand command = new SqlCommand(@"Insert into [КЛИЕНТЫ]([НОМЕР\_КЛИЕНТА], [ИМЯ], [ОТЧЕСТВО], [ФАМИЛИЯ], [ДАТА\_РОЖДЕНИЯ], [ПОЛ], [ТЕЛЕФОН], [EMAIL], [АДРЕС\_РЕГИСТРАЦИИ], [ФАКТИЧЕСКИЙ\_АДРЕС], [СЕРИЯ\_НОМЕР]) VALUES(@NUMBER, @NAME, @OTCHE, @FAMILY, @DATEBIRTHDAY, @POL, @PHONE, @EMAIL, @ADRESSREG, @ADRESSFACT, @PASS)", MyConnection.GetConnection());

public AddClients()

{

InitializeComponent();

}

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

{

string num = Номер.Text;

Convert.ToInt32(num);

string name = Имя.Text;

string fam = Фамилия.Text;

string otch = Отчество.Text;

string phone = Телефон.Text;

string adressReg = Адресрег.Text;

string adressFact = Адресфакт.Text;

string Email = email.Text;

string pol = Пол.Text;

string DateB = Рождение.Text;

string SeriaNum = СерияНомер.Text;

try

{

command.Parameters.Add("@NUMBER", SqlDbType.Int).Value = num;

command.Parameters.Add("@NAME", SqlDbType.VarChar).Value = name;

command.Parameters.Add("@OTCHE", SqlDbType.VarChar).Value = otch;

command.Parameters.Add("@FAMILY", SqlDbType.VarChar).Value = fam;

command.Parameters.Add("@DATEBIRTHDAY", SqlDbType.Date).Value = DateB;

command.Parameters.Add("@POL", SqlDbType.Char).Value = pol;

command.Parameters.Add("@PHONE", SqlDbType.VarChar).Value = phone;

command.Parameters.Add("@EMAIL", SqlDbType.VarChar).Value = Email;

command.Parameters.Add("@ADRESSREG", SqlDbType.VarChar).Value = adressReg;

command.Parameters.Add("@ADRESSFACT", SqlDbType.VarChar).Value = adressFact;

command.Parameters.Add("@PASS", SqlDbType.VarChar).Value = SeriaNum;

MyConnection.OpenConnection();

MyConnection.CloseConnection();

MessageBox.Show("Запись добавлена !", "Выполнение добавления записи");

Номер.Text = null;

Имя.Text = null;

Фамилия.Text = null;

Отчество.Text = null;

Телефон.Text = null;

Адресрег.Text = null;

Адресфакт.Text = null;

email.Text = null;

Пол.Text = null;

Рождение.Text = null;

СерияНомер.Text = null;

}

catch

{

MessageBox.Show("Не удалось добавить запись!", "Ошибка добавления");

}

}

}

}

# **ПРИЛОЖЕНИЕ Д**

Листинг Формы окна удпления данных **«**DeleteClients**»:**

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 BANK

{

public partial class DeleteClients : Form

{

DataTable table = new DataTable();

SqlDataAdapter adapter = new SqlDataAdapter();

SqlCommand sqlCommand = new SqlCommand();

public DeleteClients()

{

InitializeComponent();

}

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

{

try

{

string number = comboBox1.Text;

SqlDataAdapter adapter = new SqlDataAdapter();

sqlCommand.CommandText = @"DELETE [КЛИЕНТЫ] WHERE [НОМЕР\_КЛИЕНТА] = @NUM";

sqlCommand.Parameters.Add("@NUM", SqlDbType.VarChar).Value = number;

sqlCommand.Connection = MyConnection.Connection;

MyConnection.OpenConnection();

sqlCommand.ExecuteNonQuery();

MyConnection.CloseConnection();

MessageBox.Show("Удаление выполнено!", "Удаление записи");

}

catch

{

MessageBox.Show("Ошибка удаления!", "Удаление записи");

}

}

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

{

sqlCommand.CommandText = "SELECT [НОМЕР\_КЛИЕНТА] FROM [КЛИЕНТЫ]";

sqlCommand.Connection = MyConnection.Connection;

adapter.SelectCommand = sqlCommand;

adapter.Fill(table);

comboBox1.DataSource = table;

comboBox1.ValueMember = "НОМЕР\_КЛИЕНТА";

comboBox1.SelectAll();

}

}

}

# **ПРИЛОДЕНИЕ Е**

Листинг класса «MyElements»:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

using System.Data;

namespace BANK

{

static public class MyElements

{

static public DataTable tablefordeals = new DataTable();

static public DataTable tableforCreadits = new DataTable();

static public DataTable tableforVclads = new DataTable();

static public DataTable tableforCards = new DataTable();

static public DataTable TypeCards = new DataTable();

static public DataTable TypeCreadits = new DataTable();

static public DataTable TypeVclads = new DataTable();

static public DataTable tableWorks = new DataTable();

static public DataTable tableUser = new DataTable();

static public DataTable tableClients = new DataTable();

static public DataTable tableChet = new DataTable();

static public DataTable tableHistoryOper = new DataTable();

static public DataTable tableDogovotClients = new DataTable();

static public bool LOCK = false;

}

}

# **ПРИЛОДЕНИЕ Ё**

Листинг класса «MyElements»:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

namespace BANK

{

public static class MyConnection

{

static public SqlConnection Connection = new SqlConnection(@"Data Source=DESKTOP-9N46EPK\DANIIL\_BANK1230;Initial Catalog=data\_bank;Integrated Security=True");

static public void OpenConnection()

{

if (Connection.State == System.Data.ConnectionState.Closed)

Connection.Open();

}

static public void CloseConnection()

{

if (Connection.State == System.Data.ConnectionState.Open)

Connection.Close();

}

static public System.Data.SqlClient.SqlConnection GetConnection()

{

return Connection;

} }}