



**МИНОБРНАУКИ РОССИИ**  
**федеральное государственное бюджетное образовательное учреждение**  
**высшего образования**  
**«Московский государственный технологический университет «СТАНКИН»**  
**(ФГБОУ ВО МГТУ «СТАНКИН»)**

---

**Институт**  
информационных технологий

**Кафедра**  
информационных систем

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Управление данными»  
на тему: «Проектирование БД библиотеки»

**Студент**  
группа ИДБ–21–06

**Анчаев А. А.**

---

подпись

**Руководитель**  
старший преподаватель

**Быстрикова В. А.**

---

подпись

Москва 2023 г.

## Оглавление

ВВЕДЕНИЕ.....	3
ГЛАВА 1. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ БАЗЫ ДАННЫХ .....	4
1.1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ .....	4
1.2. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ .....	12
1.3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ.....	16
1.4. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ .....	21
ГЛАВА 2. ОПИСАНИЕ ФУНКЦИОНИРОВАНИЯ БАЗЫ ДАННЫХ.....	30
2.1. НАЗНАЧЕНИЕ И ПЕРЕЧЕНЬ ФУНКЦИЙ БАЗЫ ДАННЫХ.....	30
2.2 ОПИСАНИЕ РАБОТЫ С БАЗОЙ ДАННЫХ .....	31
ЗАКЛЮЧЕНИЕ .....	72
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	74
ПРИЛОЖЕНИЕ А. Заполнение таблиц данными.....	75
ПРИЛОЖЕНИЕ Б. SQL-операторы создания основных объектов БД.....	79
ПРИЛОЖЕНИЕ В. Код программы .....	90

## **ВВЕДЕНИЕ**

Объектом исследования курсового проекта является библиотека. Библиотека представляет собой организацию, осуществляющее учет, хранение и предоставление доступа к разнообразным информационным ресурсам, таким как книги и периодические издания. Основной целью библиотек является обеспечение общества культурными и образовательными ресурсами.

Актуальность темы проектирования базы данных для библиотек обусловлена ростом потребности в организации информационных ресурсов и обеспечении эффективного управления библиотечными коллекциями. В условиях современного общества, где информационные потоки постоянно увеличиваются, библиотеки играют важную роль в предоставлении доступа к знаниям и культурным ценностям.

Целью проектирования базы данных для библиотеки является создание системы, способной эффективно управлять каталогом книг, учетом выдачи и возврата книг, а также обеспечивать удобный доступ к информации для пользователей. Автоматизация библиотечных процессов позволяет сократить ручной труд, улучшить точность учета и повысить качество обслуживания читателей.

При проектировании курсового проекта были выделены определенные этапы. Начальным этапом является проведение анализа предметной области, выявление основных функций и требований к базе данных библиотеки. Концептуальное проектирование включает формализованное описание структуры данных без привязки к конкретной системе управления базами данных (СУБД). Логическое проектирование включает создание схемы данных, используя ER-диаграммы и определяя отношения между ключевыми сущностями. Физическое проектирование затрагивает выбор средств защиты данных, оптимизацию доступа к информации и обеспечение целостности данных. Завершающим этапом является разработка интерфейса для удобного взаимодействия с базой

данных библиотеки.

## **ГЛАВА 1. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ БАЗЫ ДАННЫХ**

### **1.1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

Под предметной областью принято понимать часть реального мира, подлежащую изучению для организации управления и автоматизации (предприятия, организации). Анализ предметной области позволяет определить, какие данные содержатся в базе данных (БД). Пользователями БД могут быть различные прикладные программы, программы-комплексы, а также специалисты предметной области, которые называются конечными пользователями.

Модель предметной области - это наши знания о предметной области. Знания могут быть как в виде неформальных знаний в мозгу эксперта, так и выражены формально при помощи каких-либо средств. В качестве таких средств могут выступать текстовые описания предметной области, наборы должностных инструкций, правила ведения дел в компании и т.п.

В данной работе в качестве предметной области рассматривается деятельность книжной библиотеки.

Библиотека – это учреждение культуры, организующее сбор, хранение и общественное пользование произведениями печати и другими документами. Библиотеки систематически занимаются сбором, хранением, и выдачей читателям произведений печати, а также информационно-библиографической работой, являются общедоступным источником знаний и основной базой для самообразования [1].

Характер требований, предъявляемых в настоящее время к информационно-библиотечной деятельности, обуславливают необходимость совершенствования информационно-библиографического обслуживания, от эффективной организации которого зависит успешное выполнение главной задачи библиотеки – содействовать информационно-документному обеспечению произведений печати и письменности для

общественного пользования, а также вести справочно-библиографическую работу. Актуальность данной темы заключается в том, что многие библиотеки до сих пор придерживаются, традиционных библиотечных систем и процессов. Необходимо, переход от бумажной коммуникации к коммуникации безбумажной, чтоб сократить время поиска необходимой литературы, информации о читателе и т.д.

Основными задачами, решаемыми в данной предметной области, являются:

- Учет читателей библиотеки.
- Пополнение и списание книг.
- Поиск и выдача книги читателю.
- Систематизация книг по разделам.
- Хранение информации о книгах и других материалах, находящихся в библиотеке, включая их названия, авторов, издательства, годы издания, ISBN и другие характеристики.
- Учет выдачи книг и контроля их возврата, включая информацию о читателях, датах выдачи и сроках возврата.
- Контроль за исполнением сформированных заказов.

Существует множество программных продуктов, решающих задачи в данной области:

Доступно много различных программных продуктов, предназначенных для решения задач в рассматриваемой области:

- сайты российских библиотек: [rsl.ru](http://rsl.ru), [nlr.ru](http://nlr.ru), [rasl.ru](http://rasl.ru), [nbmgu.ru](http://nbmgu.ru), [shpl.ru](http://shpl.ru) и другие;
- программный продукт «1С:Библиотека»;
- АИБС «МегаПро», разработанная ООО «Дата Экспресс»;
- ОПАС-Global разработанная компанией «ДИТ-ИБИС»;
- Интернет-сервисы Литнет, ЛитРес, Author.Today и т.п.

Книжные сервисы предназначены для автоматизации и оптимизации всего процесса информационно-библиографического обслуживания,

начиная с обеспечения удобного поиска и выбора необходимой литературы и заканчивая арендой книг пользователем.

Первым программным продуктом рассмотрим программное обеспечение «1С:Библиотека» (<https://solutions.1c.ru/catalog/library>).

1С:Библиотека — программа для автоматизирования рабочих процессов библиотеки, в зависимости от ее назначения, типа, состава фондов. Программа позволяет оперативно автоматизировать процессы с учетом их типа, а также в разрезе параметров библиотеки: количества фондов и их составляющих. Программное обеспечение подходит для внедрения в библиотеках и комплексах любых видов и форм, в том числе и в организациях, чья деятельность сопряжена с постоянным использованием библиотечных хранилищ и каталогов [2].

Программа 1С:Библиотека предоставляет следующие основные возможности:

- Администрирование и настройка систем под разные условия эксплуатации (рис.1.1).



Рис 1.1 Настройка системы

- Возможности инвентаризации книжного фонда (рис.1.2) и автоматизированного поиска нужных изданий и публикаций (рис.1.3).

Инвентарная книга

Сформировать Настройка... Поместить в архив Извлечь из архива

Инвентарная книга: Основного фонда

Параметры данных: Начало периода =  
Конец периода = 28.09.2010 23:59:59  
Отбор: Инв номер ссылка Инвентарная книга Равно "Основного фонда"

№ п/п	Дата	Инв. номер	Автор	Заглавие	Цена	Год издания	Место хранения	№ в КСУ	Выбытие (перемещение)		Отм. о проверке
									дата	номер	
13	22.03.2010	000033879	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
14	22.03.2010	000033880	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
15	22.03.2010	000033881	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
16	22.03.2010	000033882	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
17	22.03.2010	000033883	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
18	22.03.2010	000033884	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
19	22.03.2010	000033885	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
20	22.03.2010	000033886	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
21	22.03.2010	000033887	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
22	22.03.2010	000033888	Свириен	До Земли еще далеко	121,34		Основной ф.	1			
23	22.03.2010	000033889	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
24	22.03.2010	000033890	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
25	22.03.2010	000033891	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
26	22.03.2010	000033892	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
27	22.03.2010	000033893	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
28	22.03.2010	000033894	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
29	22.03.2010	000033895	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			
30	22.03.2010	000033896	Лакшин	Толстой и Чехов 2	312,08	2010	Основной ф.	1			

Рис 1.2 Инвентарная книга

Текстовый, монографический, физ. единица (Шаблон библиографическо...)

Записать и закрыть Показать обязательные поля Все действия

Наименование: Текстовый, монографический, физ. единица

Элементы библиографической записи Маркер Ограничения на поля Описание

Добавить

N	Элемент библиографической записи	Заголовок	Поле	Подполе	Справочник
1	Автор	Автор	700	a	Авторы
2	Редактор	Редактор	702	a	Авторы
3	Художник	Художник	702	a	Авторы
4	Составитель	Составитель	702	a	Авторы
5	Основное заглавие	Основное заглавие	200	a	
6	Сведения, относящиеся к заглавию	Сведения, относящиеся к заглавию	200	e	Сведения
7	Обозначение тома	Обозначение тома	200	v	
8	Место издания, распространения и т.д.	Место издания	210	a	Места издания
9	Имя издателя, распространителя и т.д.	Издательство	210	c	Издательство

Комментарий:

Код: 3 Группа:

Итого

Выбрать вариант... Все действия

50

Заглавие

Былины, старины, ст...  
Социально-экологиче...  
Находчивая зебра Ск...  
Встреча с писателем...  
Писатели и поэты Ро...  
Лоскутик и облако Ск...  
Летучая корова  
История России Учеб...  
История России Учеб...  
История России Учеб...

Рис 1.3 Автоматизированный поиск

- Возможность формирования и оформления заказа на покупку книг, с оперативным контролем за ходом выполнения заказа (рис.1.4).



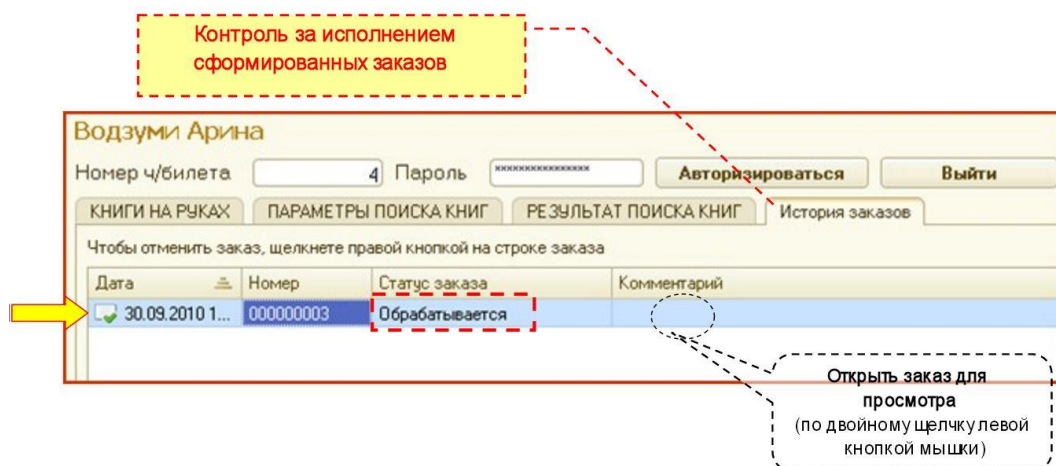


Рис 1.4 Оформление заказа

- Возможность осуществления работы по обслуживанию читателей (рис.1.5).

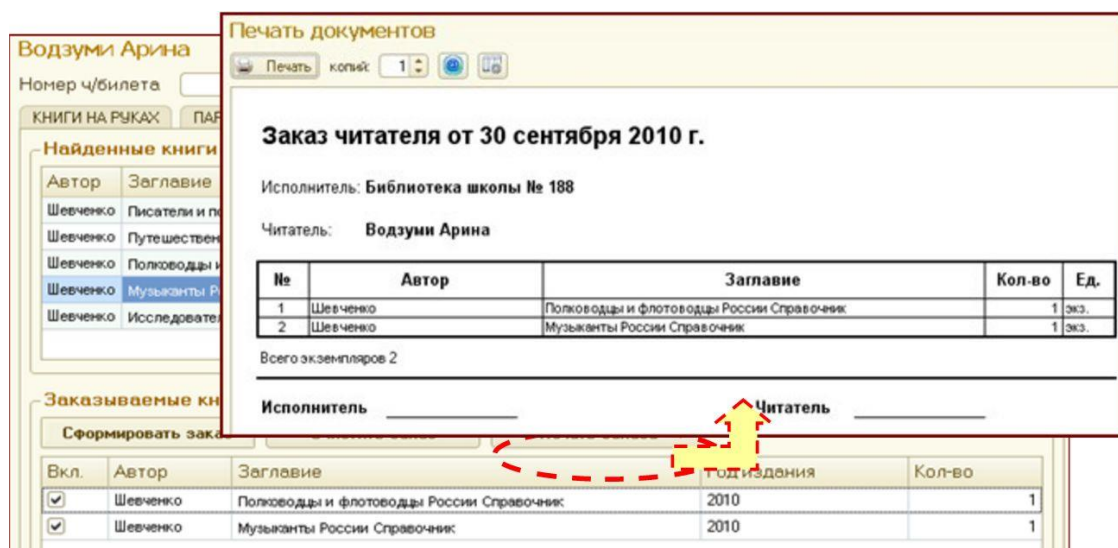


Рис 1.5 Заказ читателя

- Удаленный доступ читателя к электронному каталогу и поиск изданий по любым элементам библиографического описания, удаленный доступ читателя к своему формуляру (рис.1.6).

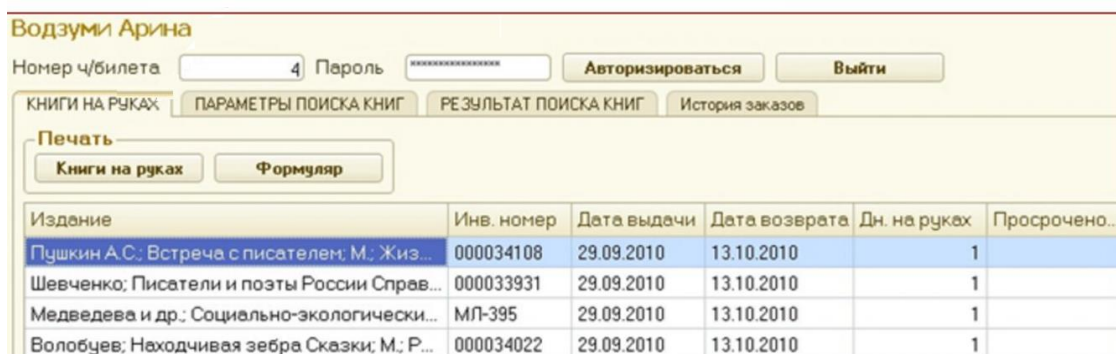


Рис 1.6 Электронный каталог



В качестве второго программного продукта рассмотрим OPAC-Global (<https://opac-global.ru>). OPAC-Global — автоматизированная библиотечная информационная система, основанная на облачных технологиях [3].

Среди основных функций данной программы стоит выделить:

- Поиск каталогизируемого документа в собственных базах данных (рис.1.7).

Рис 1.7 Поиск

- Формирование поискового запроса с помощью фильтра для упрощения поиска нужных экземпляров (рис.1.8).

▼ Фильтр

<b>Местонахождение</b> ФУБ ПМГМУ ФУБ МГУ ФУБ	<b>Фонд</b> <input type="text"/>	<b>Инвентарный номер</b> с <input type="text"/> по <input type="text"/>
<b>Дата списания</b> с <input type="text"/> по <input type="text"/>	<b>Радиометка</b> <input type="text"/>	<b>Акт списания</b> <input type="text"/>
<b>Местонахождение стеллажа</b> <input type="text"/>	<b>Причина списания</b> Ветхое Устаревшее Утерянное	<b>Направление выбытия</b> Утилизация Макулатура Передача в другой отдел
<b>Шифр хранения</b> <input type="text"/>	<b>Классификационная часть шифра</b> <input type="text"/>	<b>Часть, характеризующая документ</b> <input type="text"/>
<b>Номер экземпляра</b> <input type="text"/>	<b>Полочная форма заглавия</b> <input type="text"/>	<b>Суффикс шифра хранения</b> <input type="text"/>
<b>Номер записи в КСУ</b> с <input type="text"/> по <input type="text"/>	<b>Публикуемое примечание</b> <input type="text"/>	<b>Дата поступления</b> с <input type="text"/> по <input type="text"/>
<b>Дата создания метки</b> с <input type="text"/> по <input type="text"/>	<b>Цена</b> <input type="text"/>	<b>Имеет метку</b> Потеряные К списанию Особенности

Применить Сбросить

Найдено 28 экземпляров.

Рис 1.8 Фильтр поискового запроса

- Добавление документа в каталог.

Выбор листа ввода

- ☒ КНИГА. Одночастный.
- ☐ КНИГА. Многочастный. Общая часть
- ☐ КНИГА. Многочастный. Том
- ☐ НОТЫ. Одночастный.
- ☐ НОТЫ. Многочастный. Общая часть
- ☐ НОТЫ. Многочастный. Том
- ☐ КАРТЫ. Одночастный.
- ☐ КАРТЫ. Многочастный. Общая часть
- ☐ КАРТЫ. Многочастный. Том
- ☐ АУДИОВИЗУАЛЬНЫЙ. Одночастный.
- ☐ АУДИОВИЗУАЛЬНЫЙ. Многочастный.

Выбрать Отмена

Рис 1.9 Лист ввода

- Печать библиографической карточки.

<b>Применить редактирование</b>  <b>Параметры страницы</b>  <b>Предварительный просмотр</b>  <b>Печать основной карточки</b>  <b>Печать добавочных карточек</b>  <b>Удалить активную карточку</b>  <b>Заккрыть окно</b>	шрифт    ▼ Ж К Ч А^ А^    [иконки]
	<p><b>Шмелев, Иван Сергеевич (1873–1950).</b></p> <p>Лето Господне ; Избранное / И. С. Шмелев ; [худож. Комаров Г. А. и Комаров А. Г.]. – М. : изд. Сретенского монастыря, 2005. – 1134, [1] с., [1] л. ил. : ил. ; 17 см.</p> <p>Содерж.: Лето Господне; Избранное: Богомолье; Старый Валаам; Неупиваемая чаша; Куликово поле. – 10000 экз. – ISBN 5-7533-0240-8 (в пер.).</p>

Рис 1.10 Печать карточки

- Запись библиографического описания для использования в электронном каталоге.

Уровень готовности: <b>Незаконченный</b> Идентификатор: <b>RUNLR\b_ik145959</b> Лист ввода: <b>Книга. Однотомник</b> Формат вывода: Биб. описание ▼ <a href="#">Показать</a> Редактировать   Прототип   В список выдачи   Печать БК <a href="#">Заккрыть</a>
<div>Запись готова</div> <p><b>Маркер: 01215nam0 22003253i 450</b></p> <p>005 20101209120252.9</p> <p>010 ##\$a0-521-62633-1\$bhardback</p> <p>100 ##\$a20060504d1999 u y0rusy0189 ba</p> <p>101 0#\$aeng</p> <p>102 ##\$aGB</p> <p>105 ##\$ay      </p> <p>200 1#\$aShakespeare and domestic loss\$eforms of deprivation, mourning, a</p> <p>210 ##\$aCambridge [etc.]\$cCambridge univ. press\$d1999</p> <p>215 ##\$aXIII, 242 c.\$d24</p> <p>225 1#\$aCambridge studies in Renaissance literature and culture\$v32</p> <p>320 ##\$aБиблиогр. в примеч.: с. 202-232</p> <p>320 ##\$aУказ.: с. 233-242</p> <p>541 1#\$aШекспир и утрата дома\$zrus</p> <p>600 #1\$3RUNLR\auth\661405665\$aШекспир\$bУ. \$gУильям\$f1564 - 1</p> <p>606 1#\$aHomelessness in literature.\$2lcsh</p> <p>606 1#\$aGrief in literature.\$2lcsh</p> <p>606 1#\$aDrama - Psychological aspects.\$2lcsh</p> <p>676 ##\$a822.3'3\$v21</p> <p>686 ##\$aIII5(4Вл)43-4Шекспир 54\$2rubbk</p>

Рис 1.11 Библиографическое описание

Особенностью программы 1С:Библиотека является наличие

виртуального кабинета читателя и возможность интегрирования с другими решениями фирмы «1С». У OPAC-Global также обладает возможностью интеграции с другими системами управления данными, имеет более гибкий поиск и оптимизацией по инвентаризации книг. По сути, является онлайн-каталогизатором.

После проведения анализа предметной области был выделен перечень функций, которые будут реализованы в данной работе:

- Каталогизация и классификация книг.
- Возможность автоматизированного поиска нужных изданий и публикаций.
- Возможность осуществлять работы по обслуживанию читателей.
- Возможность формирования и оформления заказа на покупку книг.
- Списание книг.
- Учет читателей.

## **1.2. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

Цель концептуального проектирования – определение содержания базы данных. Концептуальное проектирование заключается в формализованном описании предметной области, чтобы можно было проанализировать корректность схемы БД, учитывая отсутствие привязки к СУБД. На этом этапе необходимо проанализировать задачи и построить диаграмму вариантов использования (ДВИ).

Диаграмма вариантов использования (Use Case Diagram) - это графический инструмент, который используется на этапе концептуального проектирования для описания потенциальных вариантов использования системы и ее функциональности. Она является частью анализа требований, который позволяет разработчикам конкретизировать требования пользователей к системе. На диаграмме вариантов использования представлены актеры, пользователи, которые взаимодействуют со системой, и их варианты использования системы, т.е. различные сценарии,

в которых актеры используют систему для достижения своих целей. Диаграмма вариантов использования позволяет увидеть весь набор возможностей, которые предоставляет система пользователю. Она также может использоваться для выявления потенциальных ошибок, необходимости добавления новых функциональных требований или уточнения уже существующих.

Составим словесное описание проектируемой БД. По телефону читатели могут узнать о наличии книг и их характеристике, оформить заказ на получение книги в читальном зале или забронировать ее (забронированная книга должна быть получена читателем в течение дня, забронированная, но не полученная книга после закрытия библиотеки считается свободной для других читателей).

Доставка книг клиентам библиотеки не осуществляется.

Читатель, пришедший в библиотеку, может получать у библиотекаря информацию о наличии книг и их характеристике, оформить заказ на получение книги в читальном зале.

Библиотекарь может производить запись и отмену записей на получение книги, предоставлять информацию о услугах (выдаче книг на руки, возможности чтения в читальном зале и т.д.) Также занимается заполнением ведомости после факта выдачи и получения книг [1].

С проектируемой системой для библиотеки будут взаимодействовать следующие действующие лица:

- консультант у справочного телефона;
- библиотекарь;
- библиограф;
- главный библиотекарь;

В реальной жизни читатель библиотеки может просматривать электронный каталог книг. Однако в проектируемой системе такой возможности у него нет, поэтому читатель не будет являться действующим лицом.

Рассмотрим варианты использования каждого из действующих лиц.

Для *консультанта у справочного телефона* доступны следующие функции:

1. Выдача информации о характеристиках и наличии книг.
2. Резервирование книги (включает предварительную проверку наличия книги на складе).
3. Аннулирование брони по желанию читателя.
4. Добавление нового читателя.
5. Изменение данных читателей.
6. Выдача информации о читателях.

Для *библиотекаря* доступны следующие функции:

1. Выдача информации о характеристиках и наличии книг.
2. Выдача книг читателям и их принятие.
3. Добавление нового читателя.
4. Выдача информации о читателях.
5. Изменение данных читателей.

Для *библиографа*, ответственного за книжный фонд, доступны следующие функции:

1. Принятие поступления книг на склад.
2. Добавление книг в каталог.
3. Выдача информации о характеристиках и наличии книг.

Для *главного библиотекаря* доступны следующие функции:

1. Добавление нового сотрудника.
2. Увольнение сотрудника.
3. Выдача информации о характеристиках и наличии книг.
4. Выдача информации о читателях.

Составим общую диаграмму вариантов использования (рис.1.12).

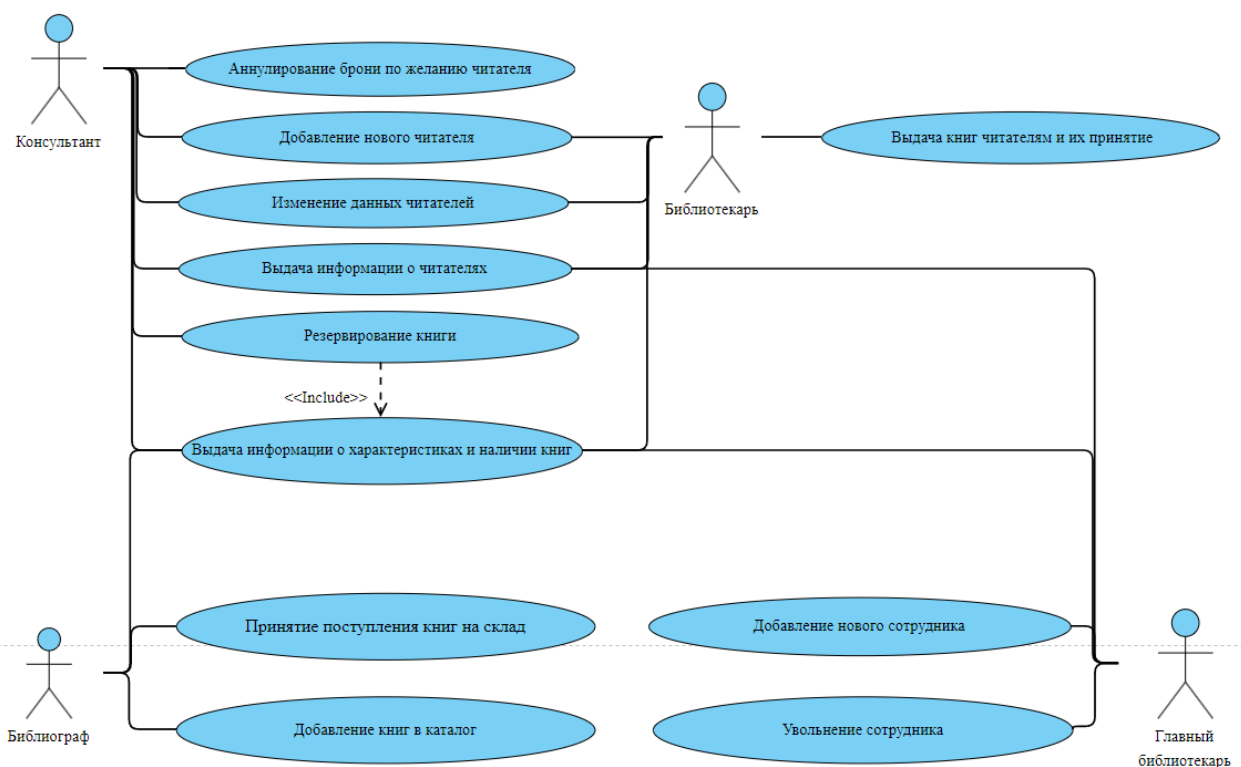


Рис 1.12 Диаграмма вариантов использования

В первом приближении для решения выделенных задач необходимо хранение данных о следующих объектах:

1. Книга
2. Выдача и сдача книги.
3. Регистрация поступления книги на склад.
4. Резервирование книги.
5. Читатели.
6. Сотрудники.

Необходимые данные можно классифицировать по частоте их изменения:

- условно-постоянные данные (например, характеристики выдаваемых книг, информация о сотрудниках);
- данные, которые оперативно обновляются при каждом решении задачи (дата выдачи, дата возврата, факт наличия задолженностей по аренде).



### 1.3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

Вторым этапом проектирования базы данных является логическое проектирование. Логическим проектированием базы данных является преобразование требований к данным в структуры данных. Результатом логического проектирования является СУБД-ориентированная структура базы данных и спецификации прикладных программ. Целью логического проектирования служит определение состава и структуры таблицы БД на основе результатов концептуального проектирования и проверка полученной модели с помощью методов нормализации.

В стадии логического проектирования входит:

- формирование отношений на основе логической модели данных;
- проверка отношений с использованием средств нормализации;
- определение ограничений целостности.

Для построения отношений используются диаграммы ER-типа сущность - связь. Основные ее определения:

- сущность – объект, информация о котором должна храниться в базе данных. Для ее изображения используется прямоугольный блок с именем сущности в виде существительного;

- связь – ассоциация между двумя сущностями, она предполагает наличие общих атрибутов. Для ее изображения используется ромб с именем связи в виде глагола [1].

Связи бывают типов: 1:1, 1:M (или M:1), M:M. Класс принадлежности может быть обязательным и необязательным.

На предыдущем этапе были выделены объекты, которые необходимо хранить в БД. Эти объекты становятся сущностями при ER-моделировании.

Построим ER-диаграммы всех сущностей и связей между ними.

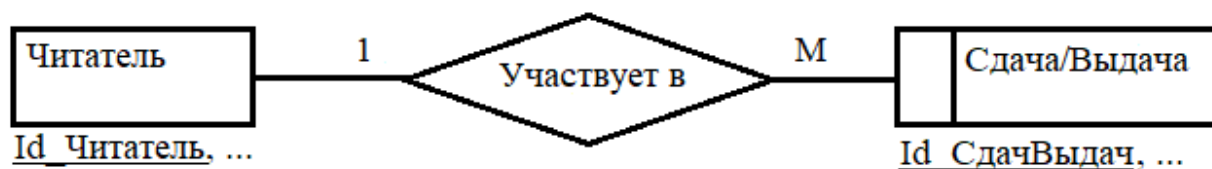


Рис. 1.13 Связь «Участвует в»

Связь «Участвует в» (рис. 1.13) имеет тип связи 1:M, так как каждый читатель может несколько раз оформить Сдачу/Выдачу книг, а в каждой записи о сдаче или выдаче может быть указан только один Читатель. Сущность Читатель имеет необязательный класс принадлежности, так как читатель может не участвовать в сдаче или выдаче книг. Сущность Сдача или Выдача имеет обязательный класс принадлежности, так как в каждой сдаче или выдаче книги должен обязательно указываться читатель.

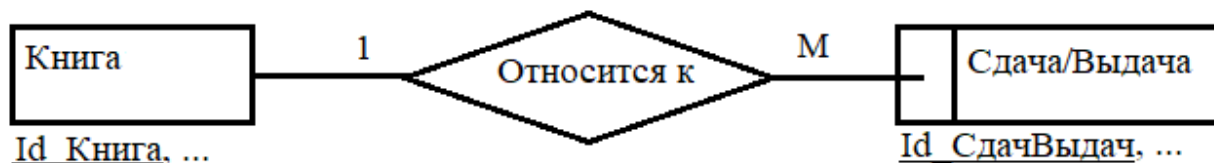


Рис. 1.14 Связь «Относится к»

Связь «Относится к» (рис. 1.14) имеет тип связи 1:M, так как каждая книга может Сдаваться/Выдаваться несколько раз, а в каждой сдаче/выдаче может быть указана только одна Книга. Сущность Книга имеет необязательный класс принадлежности, так как не все книги могут быть сданы/выданы. Сущность Сдача/Выдача имеет обязательный класс принадлежности, так как в сдаче/выдаче обязательно присутствует книга.

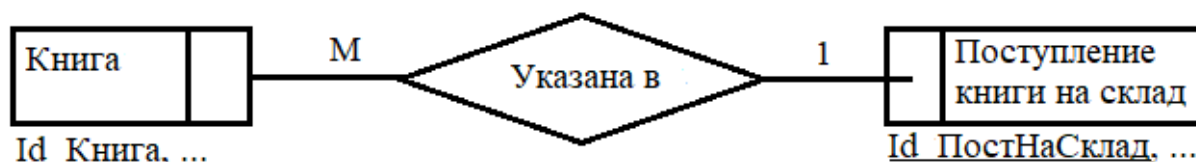


Рис. 1.15 Связь «Указана в»

Связь «Указана в» (рис. 1.15) имеет тип связи M:1, так как в каждом поступлении на склад может быть несколько Книг. Книга относится к одному Поступлению. Сущность Книга имеет обязательный класс принадлежности, так как книга обязательно поступает на склад. Сущность

Поступление книги на склад имеет обязательный класс принадлежности, так как каждое поступление на склад обязательно содержит книги.

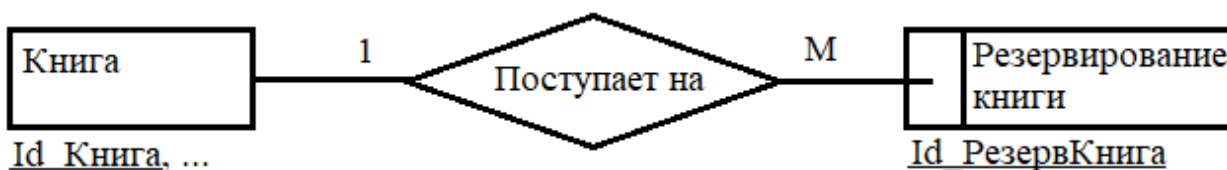


Рис. 1.16 Связь «Поступает на»

Связь «Поступает на» (рис. 1.16) имеет тип связи 1:M, так как каждая книга может быть зарезервирована несколько раз. Сущность Книга имеет необязательный класс принадлежности, так как не все книги могут быть зарезервированы. Сущность Резервирование книги имеет обязательный класс принадлежности, так как в резервировании книги обязательно присутствует книга.



Рис. 1.17 Связь «Создает»

Связь «Создает» (рис. 1.17) имеет тип связи 1:M, так как каждый Читатель может создать несколько резервирований книг, а в каждой записи о резервировании может быть указан только один оформивший ее Читатель. Сущность Резервирование книги имеет обязательный класс принадлежности, так как в каждом резервировании книги должен указываться читатель. Сущность Читатель имеет необязательный класс принадлежности, так как читатель может не создавать резервирование книги.

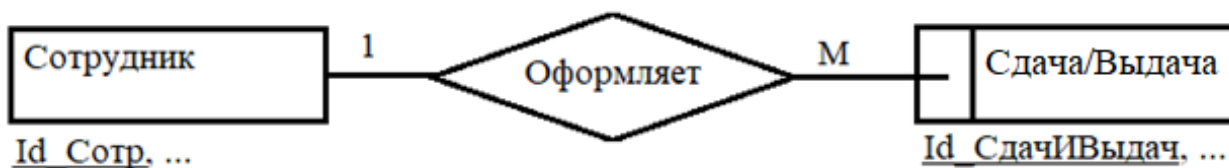


Рис. 1.18 Связь «Оформляет»

Связь «Оформляет» (рис. 1.18) имеет тип связи 1:M, так как каждый Сотрудник может оформить несколько записей о сдаче и выдаче книг, а в каждой сдаче и выдаче может быть указан только один выполнивший их Сотрудник. Сущность Сдача и выдача имеет обязательный класс принадлежности, так как в каждой оказываемой услуге должен указываться сотрудник, оказавший её. Сущность Сотрудник имеет необязательный класс принадлежности, так как сотрудник может не оформлять сдачу и выдачу книг.

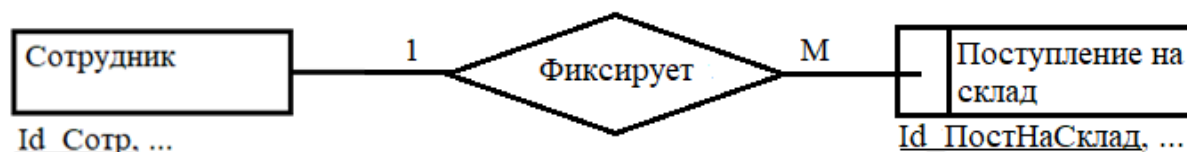


Рис. 1.19 Связь «Фиксирует»

Связь «Фиксирует» (рис. 1.19) имеет тип связи 1:M, так как каждый Сотрудник может зафиксировать несколько записей о поступлении книг на склад, а в каждой записи о поступлении на склад может быть указан только один принявший книги Сотрудник. Сущность Поступление на склад имеет обязательный класс принадлежности, так как в каждом поступлении на склад должен указываться сотрудник, оказавший её. Сущность Сотрудник имеет необязательный класс принадлежности, так как сотрудник может не принимать поступление на склад.

А) Сформируем набор предварительных отношений с указанием предполагаемого первичного ключа для каждого отношения.

Связь «Участствует в» (рис. 1.13) удовлетворяет условиям правила 4, в соответствии с которым получается 2 отношения:

1. Читатель (Id\_Читатель, ...)
2. Сдача/Выдача (Id\_СдачВыдач, Id\_Читатель, ...)

Связь «Относится к» (рис. 1.14) удовлетворяет условиям правила 4, в соответствии с которым получается 3 отношения:

1. Книга (Id\_Книга, ...)
2. Сдача/Выдача (Id\_СдачВыдач, Id\_Книга ...)

Связь «Указана в» (рис. 1.15) удовлетворяет условиям правила 4, в соответствии с которым получается 3 отношения:

1. Книга (Id\_Книга, ...)
2. ПоступлениеНаСклад (Id\_ПостНаСклад, Id\_Книга, ...)

Связь «Поступает на» (рис. 1.16) удовлетворяет условиям правила 4, в соответствии с которым получается 2 отношения:

1. Книга (Id\_Книга, ...)
2. РезервированиеКниги (Id\_РезервКнига, Id\_Книга, ...)

Связь «Создает» (рис. 1.17) удовлетворяет условиям правила 4, в соответствии с которым получается 2 отношения:

1. Читатель (Id\_Читатель, ...)
2. РезервированиеКниги (Id\_РезервКнига, Id\_Читатель, ...)

Связь «Оформляет» (рис. 1.18) удовлетворяет условиям правила 4, в соответствии с которым получается 2 отношения:

3. Сотрудник (Id\_Сотр, ...)
4. Сдача/Выдача (Id\_Сдач/Выдач, Id\_сотр, ...)

Связь «Фиксирует» (рис. 1.19) удовлетворяет условиям правила 4, в соответствии с которым получается 2 отношения:

1. Сотрудник (Id\_Сотр, ...)
2. ПоступлениеНаСклад (Id\_ПостНаСклад, Id\_Сотр, ...)

Б) Добавим не ключевые атрибуты в каждое из предварительных отношений с условием, чтобы отношения отвечали требованиям третьей нормальной формы.

Книга (Id\_Книга, Автор, Название, Жанр, ISBN, УДК, Тираж, Количество)

Сдача/Выдача (Id\_СдачВыдач, Id\_Книга, Id\_Читатель, Id\_Сотр, ДатаВыдачи, ДатаСдачи, Статус)

ПоступлениеНаСклад (Id\_ПостНаСклад, Id\_Книга, Id\_Сотр, ДатаПоступ, Количество)

РезервированиеКниги (Id\_Книга, Id\_РезервКнига, Id\_Читатель, ДатаРезерв)

Читатель (Id\_Читатель, ФИО, Пол, Телефон, ДеньРождения)

Сотрудник (Id\_Сотр, ФИО, Должность, №Телефона, №Паспорта)

#### **1.4. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ**

Третьим и заключительным этапом проектирования базы данных является физическое проектирование. Физическое проектирование – это процесс подготовки описания реализации базы данных. На этом этапе рассматриваются основные отношения и организация, предназначенных для обеспечения эффективного доступа к данным, а также все связанные с этим ограничения целостности и средства защиты.

Физическое проектирование является этапом создания проекта базы данных, при выполнении которого проектировщик принимает решения о способах реализации разрабатываемой базы данных. Во время предыдущего этапа проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области). Хотя эта структура не зависит от конкретной целевой СУБД, она создается с учетом выбранной модели хранения данных, например реляционной, сетевой или иерархической. Однако, приступая к физическому проектированию базы данных, прежде всего необходимо выбрать конкретную целевую СУБД. Поэтому физическое проектирование неразрывно связано с конкретной СУБД. Между логическим и физическим проектированием существует постоянная обратная связь, так как решения, принимаемые на этапе физического проектирования с целью повышения производительности системы, способны повлиять на структуру логической модели данных [7].

Как правило, основной целью физического проектирования базы данных является описание способа физической реализации логического проекта базы данных.

В качестве СУБД для проектирования базы данных выбран Microsoft SQL Server. Среда SQL Server Management Studio – это основной, стандартный и полнофункциональный инструмент для работы с Microsoft SQL Server, разработанный компанией Microsoft, который предназначен как для разработчиков, так и для администраторов SQL Server.

Так как в задачи входит полное сопровождение Microsoft SQL Server, начиная от создания баз данных, написания SQL запросов, создания хранимых процедур и функций, и заканчивая администрированием SQL Server, включая управление безопасностью, то в качестве основного инструмента отлично подходит среда SQL Server Management Studio.

На основе сведений, полученных в ходе выполнения предыдущего этапа, где были построены ER-диаграммы, а также сформированы отношения и добавлены не ключевые атрибуты, необходимо составить структуры таблиц, с помощью которых можно будет физически спроектировать базу данных. Для создания БД необходимо составить структуры вышеперечисленных таблиц. Необходимо установить типы данных, возможность оставить поля незаполненным, назначить первичные и внешние ключи, установить ограничения, значения по умолчанию, ссылки на другие таблицы.

В таблице 1.1 представлена структура таблицы «Книга», в приложении А (рис. А.1) содержатся данные таблицы «Книга».

Таблица 1.1

Требования к структуре таблицы «Книга»

Столбец	Тип данных	Ноль	Ключ	По умолчанию	Ограничение
Id_Книга	INT	Нет	Первичный		
Автор	VARCHAR(50)	Нет			
Название	VARCHAR(50)	Нет			
Жанр	VARCHAR(50)	Нет			
ISBN	VARCHAR(50)	Нет			
УДК	VARCHAR(50)	Нет			



Тираж	INT	Нет		100	>0
Количество	INT	Нет		1	>=0

В таблице 1.2 представлена структура таблицы «Читатель», в приложении А (рис. А.2) содержатся данные таблицы «Читатель».

Таблица 1.2

#### Требования к структуре таблицы «Читатель»

Столбец	Тип данных	Ноль	Ключ	По умолчанию	Ограничение
Id_Читатель	INT	Нет	Первичный		
ФИО	VARCHAR(50)	Нет			
Пол	CHAR(1)	Нет		м	«М» или «Ж»
Телефон	VARCHAR(12)	Нет			Уникальный
ДеньРождения	DATE	Нет			

В таблице 1.3 представлена структура таблицы «Сотрудник», в приложении А (рис. А.3) содержатся данные таблицы «Сотрудник».

Таблица 1.3

#### Требования к структуре таблицы «Сотрудник»

Столбец	Тип данных	Ноль	Ключ	По умолчанию	Ограничение
Id_Сотр	INT	Нет	Первичный		
ФИО	VARCHAR(50)	Нет			
Должность	VARCHAR(50)	Нет			
№Телефона	VARCHAR(12)	Да			Уникальный
№Паспорта	INT	Нет			Уникальный

В таблице 1.4 представлена структура таблицы «Сдача/Выдача», в приложении А (рис. А.4) содержатся данные таблицы «Сдача/Выдача».

Таблица 1.4

#### Требования к структуре таблицы «Сдача/Выдача»

Столбец	Тип данных	Нуль	Ключ	По умолчанию	Ограничение	Ссылка
Id_СдачВыдач	INT	Нет	Первичный			
Id_Книга	INT	Нет	Внешний			Книга (Id_Книга)
Id_Читатель	INT	Нет	Внешний			Читатель (Id_Читатель)
Id_Сотр	INT	Нет	Внешний			Сотрудник (Id_Сотр)
ДатаВыдачи	DATE	Нет		Текущая дата	ДатаСдачи > ДатаВыдачи	
ДатаСдачи	DATE	Нет				
Статус	BIT	Нет		0		

В таблице 1.5 представлена структура таблицы «ПоступлениеНаСклад», в приложении А (рис. А.5) содержатся данные таблицы «ПоступлениеНаСклад».

Таблица 1.5

Требования к структуре таблицы «ПоступлениеНаСклад»

Столбец	Тип данных	Нуль	Ключ	По умолчанию	Ограничение	Ссылка
Id_ПостНаСклад	INT	Нет	Первичный			
Id_Книга	INT	Нет	Внешний			Книга (Id_Книга)
Id_Сотр	INT	Нет	Внешний			Сотрудник (Id_Сотр)
ДатаПоступ	DATE	Нет		Текущая дата		
Количество	INT	Нет		1	>=1	

В таблице 1.6 представлена структура таблицы «РезервированиеКниги», в приложении А (рис. А.6) содержатся данные таблицы «РезервированиеКниги».

Таблица 1.6

Требования к структуре таблицы «РезервированиеКниги»

Столбец	Тип данных	Ноль	Ключ	По умолчанию	Ограничение	Ссылка
Id_Книга	INT	Нет	Внешний			Книга (Id_Книга)
Id_Резерв Книга	INT	Нет	Первичный			
Id_Читатель	INT	Нет	Внешний			Читатель (Id_Читатель)
ДатаРезерв	DATE	Нет		Текущая дата	ДатаРезерв >=Текущая Дата	

На рис. 1.20 отображена схема базы данных библиотеки.

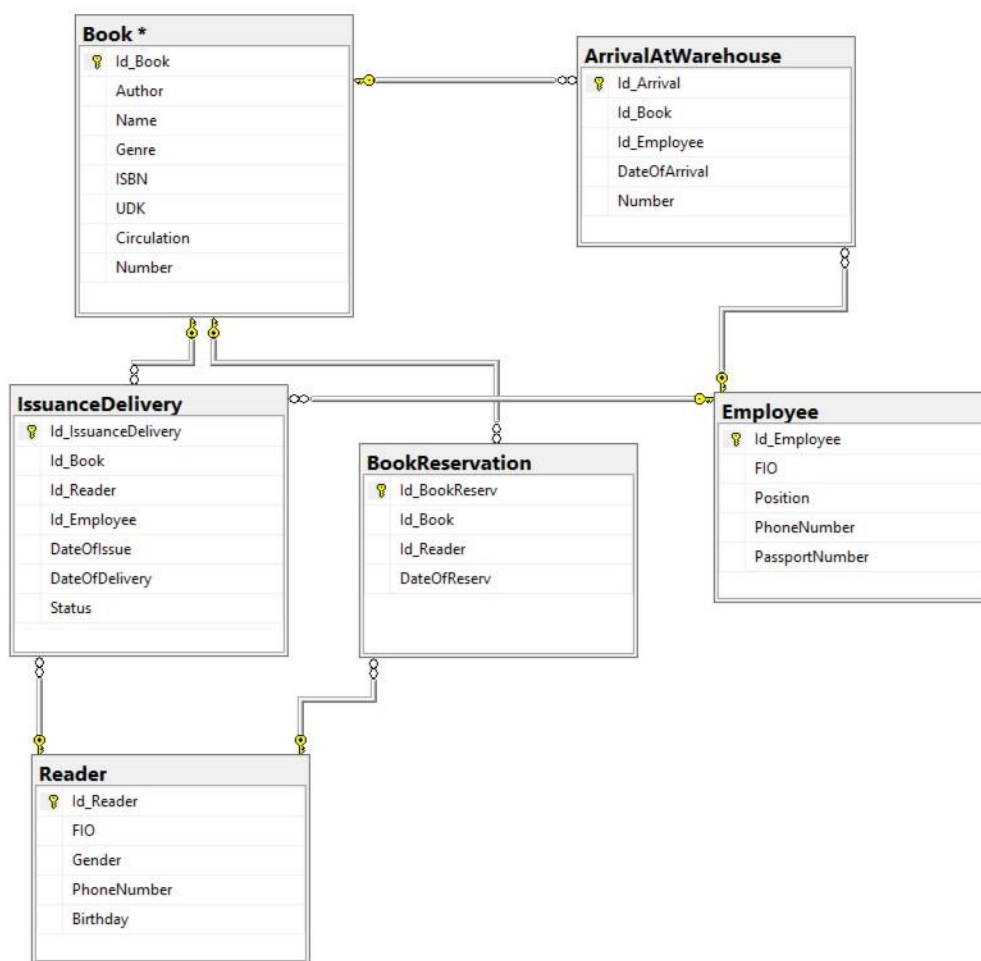


Рис. 1.20 Схема БД библиотеки

У таблиц также есть собственные триггеры и процедуры.

- У таблицы «ArrivalAtWarehouse» есть триггер «trigAddArrivalAtWarehouse», а также хранимая процедура «AddArrivalAtWarehouse» (Приложение Б, п.1, п.2). Процедура «AddArrivalAtWarehouse» необходима для добавления в БД поступления книги. Входными данными являются id книги, номер телефона работника, принявшего поступление, дата поступления и количество поступивших книг. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Триггер «trigAddArrivalAtWarehouse» предназначен для присваивания поступлению ID, а также для обновления поля «количество» при поступлении книги в таблице «Book».

- У таблицы «Reader» есть триггер «RegReader», а также хранимые процедуры «RegisterReader», «UpdateReader», «SearchReader» (Приложение Б, п.3, п.4, п.5, п.6). Процедура «RegisterReader» необходима для добавления читателя в таблицу «Reader». Входными данными являются ФИО читателя, пол, номер телефона читателя, дата рождения. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Процедура «UpdateReader» необходима для обновления номера телефона читателя в БД. Входными данными являются ID читателя, новый номер телефона. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Процедура «SearchReader» предназначена для поиска в БД читателя. Входными данными являются ФИО читателя, номер телефона читателя. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде.

- У таблицы «Book» есть хранимые процедуры «AddBook» и «SearchBook» (Приложение Б, п.7, п.8). Процедура «AddBook» необходима для добавления книги в таблицу. Входными данными являются автор книги, название книги, жанр, ISBN, УДК. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Процедура «SerachBook» необходима для поиска книги в таблице. Входными данными являются автор книги, название книги. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде.

- У таблицы «BookReservation» есть хранимые процедуры «AddReservation» и «DeleteReservation» (Приложение Б, п.9, п.10). Процедура «AddReservation» необходима для добавления резервирования книги в таблицу. Входными данными являются автор книги, название книги, номер телефона читателя, дата резервирования. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Процедура «DeleteReservation» необходима для удаления резервирования книги. Входными данными являются автор книги, название книги, номер телефона читателя, дата резервирования. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде.

- У таблицы «Employee» есть хранимые процедуры «AddEmployee» и «FireEmployee» (Приложение Б, п.11, п.12). Процедура «AddEmployee» необходима для добавления работника в таблицу. Входными данными являются ФИО, должность, номер телефона, номер и серия паспорта. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Процедура «FireEmployee» необходима для увольнения сотрудника. Входным параметром являются данные паспорта. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде.

- У таблицы «IssuanceDelivery» есть хранимые процедуры «BookIssuance» и «AcceptTheBook» (Приложение Б, п.13, п.14). Процедура «BookIssuance» необходима для добавления записи в таблицу о выдаче книги. Входными данными являются автор книги, название книги, номер телефона читателя, номер телефона работника, дата выдачи, дата возврата. Процедура проверяет входные данные, устанавливая значение выходного

параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде. Процедура «AcceptTheBook» необходима для добавления в таблицу записи о возврате книги. Входным параметром является ID выдачи книги. Процедура проверяет входные данные, устанавливая значение выходного параметра @message в зависимости от корректности введенных значений. Это значение затем обрабатывается в С#-коде.



## **ГЛАВА 2. ОПИСАНИЕ ФУНКЦИОНИРОВАНИЯ БАЗЫ ДАННЫХ**

### **2.1. НАЗНАЧЕНИЕ И ПЕРЕЧЕНЬ ФУНКЦИЙ БАЗЫ ДАННЫХ**

Основной целью при разработке программного интерфейса для работы с базой данных является создание интуитивно понятного, простого и удобного в использовании инструмента для пользователей. Взаимодействовать с базой данных должны четыре вида пользователей: консультант, библиотекарь, библиограф и главный библиотекарь. Каждая категория сотрудников должна иметь свой интерфейс и функционал, соответствующий их должностным обязанностям.

Консультанту доступны следующие функции:

- Выдача информации о характеристиках и наличии книг.
- Резервирование книги (включает предварительную проверку наличия книги на складе).
- Аннулирование брони по желанию читателя.
- Добавление нового читателя.
- Изменение данных читателей.
- Выдача информации о читателях.

Библиотекарю доступны следующие функции:

- Выдача информации о характеристиках и наличии книг.
- Выдача книг читателям и их принятие.
- Добавление нового читателя.
- Выдача информации о читателях.
- Изменение данных читателей.

Библиографу доступны следующие функции:

- Принятие поступления книг на склад.
- Добавление книг в каталог.
- Выдача информации о характеристиках и наличии книг.

Главному библиотекарю доступны следующие функции:

- Добавление нового сотрудника.

- Увольнение сотрудника.
- Выдача информации о характеристиках и наличии книг.
- Выдача информации о читателях.

Для реализации пользовательского интерфейса использовались: среда разработки Visual Studio [4], СУБД Microsoft SQL Server, SQL Server Management Studio [7], язык программирования C# [5]

## 2.2 ОПИСАНИЕ РАБОТЫ С БАЗОЙ ДАННЫХ

При входе в приложение открывается главная страница (рис. 2.1). Вход на форму сотрудника осуществляется нажатием на соответствующую сотруднику кнопку.

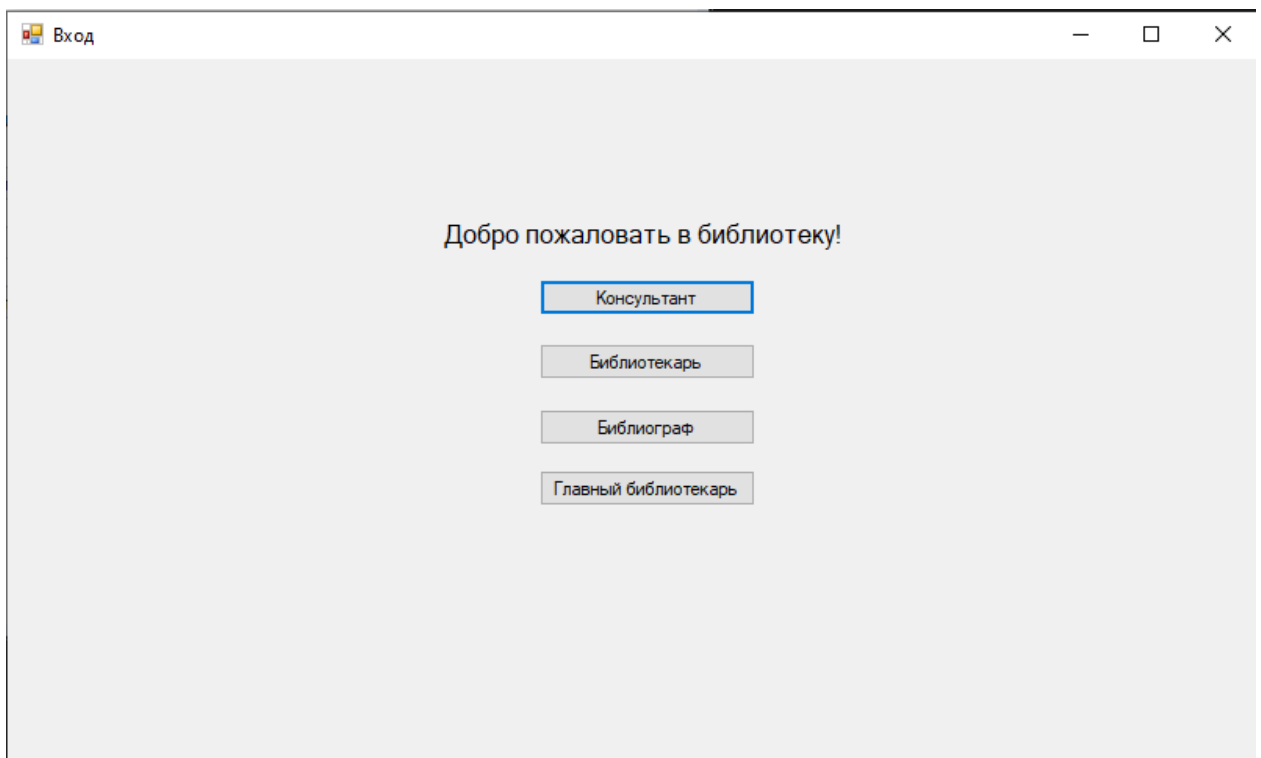


Рис. 2.1 Главная страница приложения

Войдем на форму консультанта и продемонстрируем его функционал возможностей работы с системой (рис. 2.2).

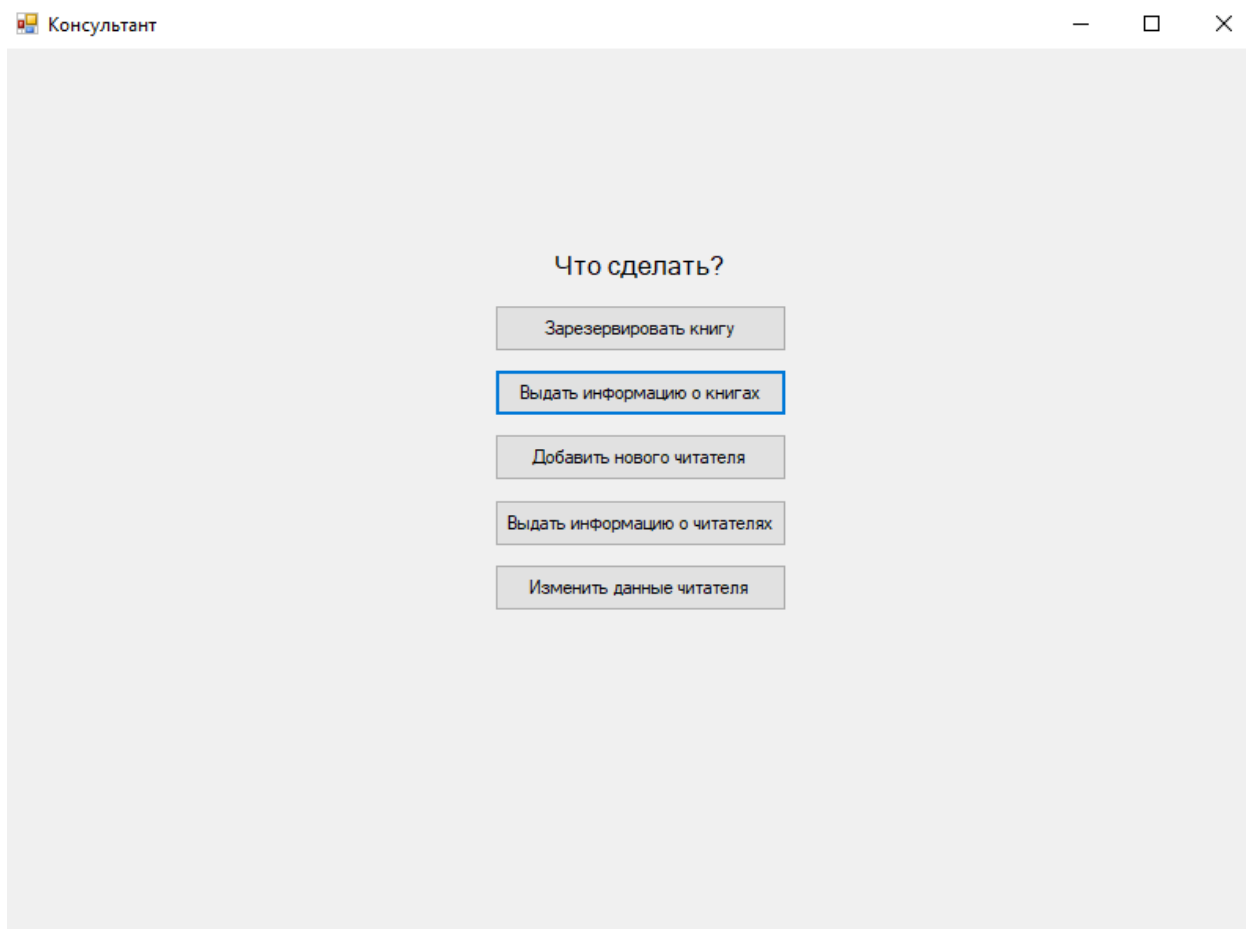


Рис. 2.2 Главная форма консультанта

Перейдем на форму «Резервирование книги» нажатием на первую сверху в списке кнопку. После чего откроется соответствующая форма (рис. 2.3). На форме доступны 4 функции: добавление записи о резервировании книги, удалении записи, обновление данных, и поиск по ФИО читателя.

Резервирование книги

ФИО Читателя	Номер телефона	Автор книги
Иванов Иван Иванович	22222	Фёдор Достоевский
Петрова Елена Сергеевна	22222222	Лев Толстой
Смирнов Алексей Андреевич	73456789012	Анна Ахматова
Козлова Мария Дмитриевна	74567890123	Александр Блок
Николаев Игорь Владимирович	75678901234	Иван Бунин
Егорова Ольга Александровна	76789012345	Антон Чехов
Федоров Денис Сергеевич	77890123456	Николай Гоголь
Сергеева Анастасия Ивановна	78901234567	Владимир Маяковский
Кузнецов Павел Алексеевич	79012345678	Борис Пастернак
Александрова София Дмитриевна	71233456789	Михаил Шолохов
Александров Анатолий Петрович	71112223344	Игорь Стругацкий
Иванова Анна Владимировна	72223334455	Аркадий Стругацкий
Сергеев Алексей Сергеевич	73334445566	Макс Фрай
Кузнецова Татьяна Алексеевна	74445556677	Дмитрий Емец
Петров Артем Владимирович	75556667788	Сергей Лукьяненко
Николаева Виктория Игоревна	76667778899	Татьяна Толстая
Егоров Павел Дмитриевич	77778889900	Иван Бунин
Федорова Екатерина Сергеевна	88889990011	Фёдор Достоевский
Смирнов Денис Иванович	99990001122	Лев Толстой
Козлова Маргарита Алексеевна	00011122333	Антон Чехов
Александрова Ирина Анатольевна	11122233444	Александр Солженицын
Смирнова Алена Андреевна	33344455566	Лев Толстой
Кузнецов Владислав Дмитриевич	44455566777	Антуан де Сент-Экзюпер
Петрова Оксана Александровна	55566677888	Игорь Стругацкий
Николаев Пётр Владимирович	66677788999	Аркадий Стругацкий

Зарезервировать книгу

Удалить резервирование

Обновить данные

ФИО

Рис. 2.3 Форма «Резервирование книги»

При нажатии на кнопку «Зарезервировать книгу» открывается форма добавления записи о резервировании книги. При нажатии на кнопку «Зарезервировать» срабатывает хранимая процедура «AddReservation» (рис. 2.4).

Резервирование книги

Автор:

Книга:

Номер телефона:

Дата:

Рис. 2.4 Форма добавления записи о резервировании книги

При введении некорректных данных:

1. Неправильной даты (рис. 2.5);
2. Книги, которой нет в наличии (рис. 2.6);
3. Несуществующей книги (рис. 2.7);
4. Повторной резервации одной и той же книги одним читателем (рис. 2.8);
5. Несуществующего номера телефона (рис. 2.9);
6. Пустых значений (рис. 2.10);

Появится окно, сообщающее об одной из данных ошибок.

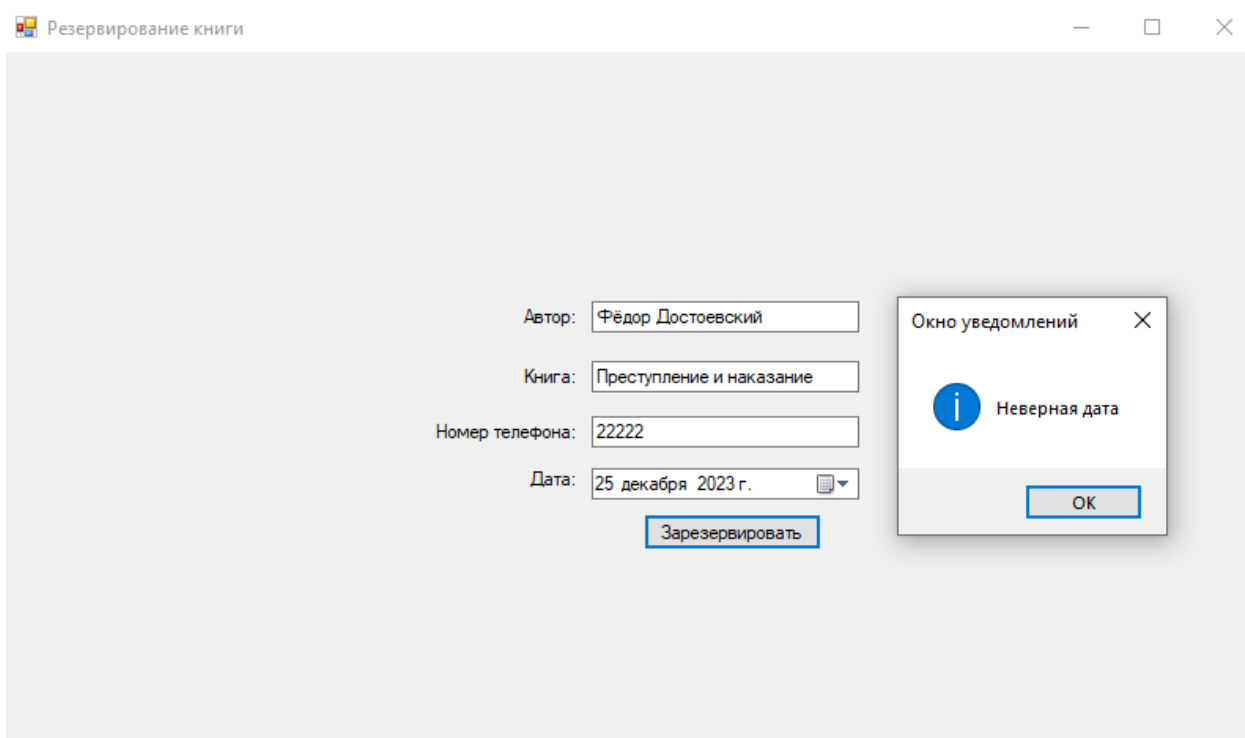


Рис. 2.5 Результат ввода некорректной даты

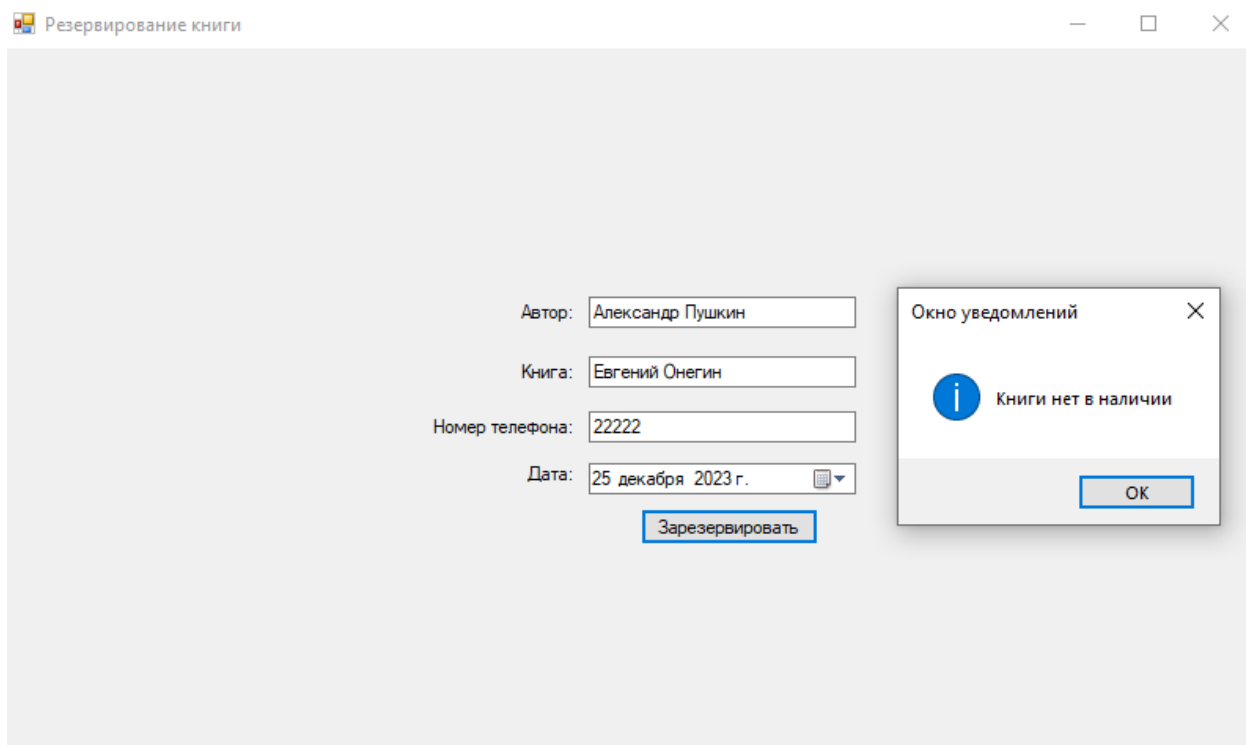


Рис. 2.6 Результат ввода книги, которой нет в наличии

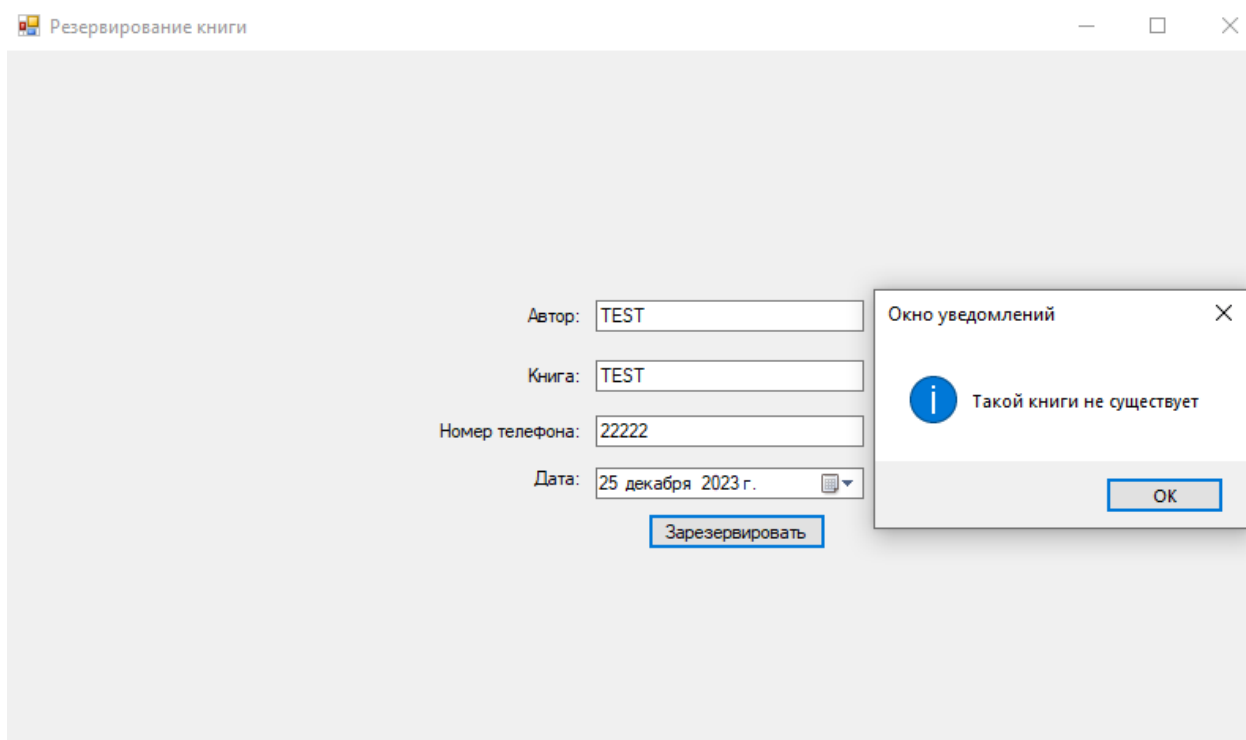


Рис. 2.7 Результат ввода несуществующей книги

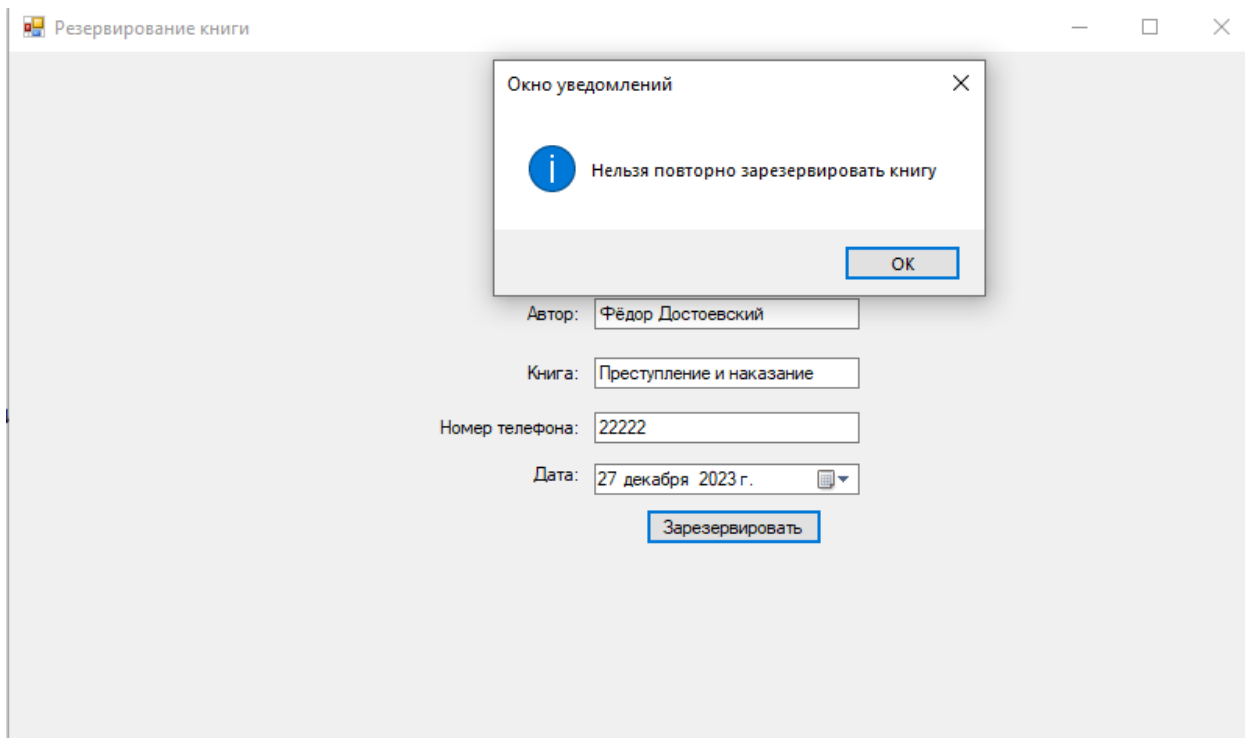


Рис. 2.8 Результат повторной резервации одной и той же книги одним читателем

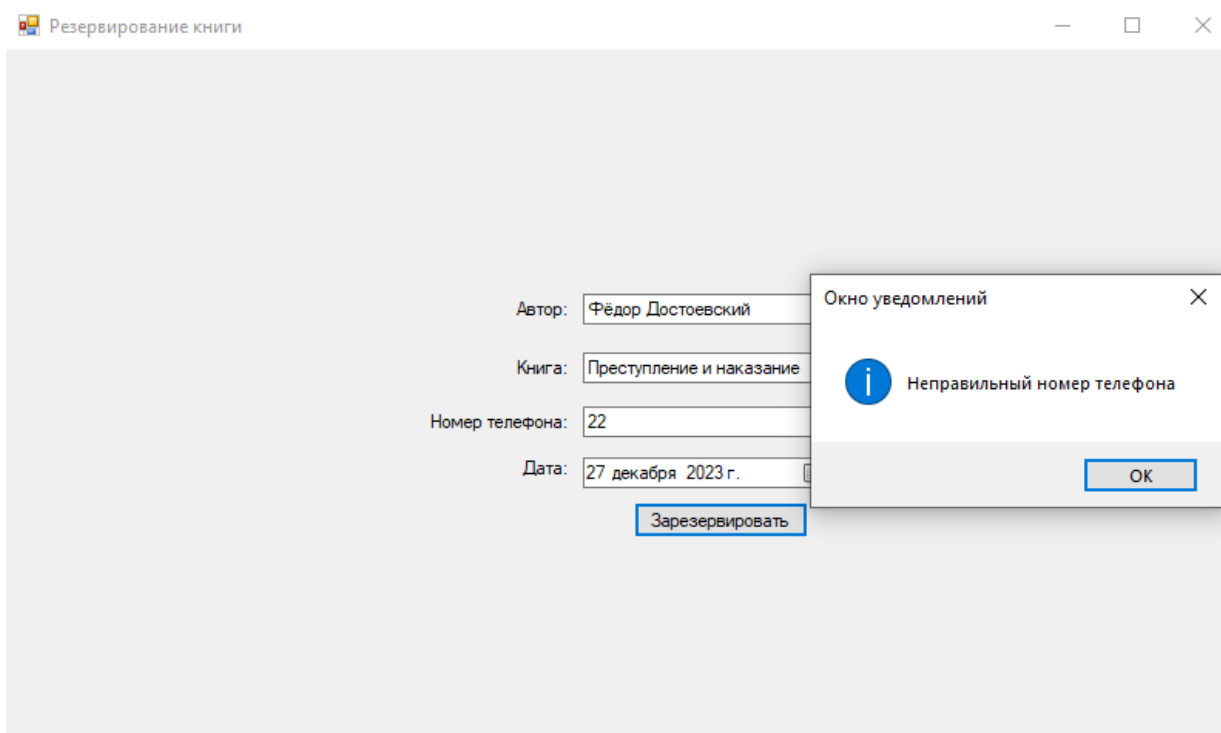


Рис. 2.9 Результат ввода некорректного номера телефона



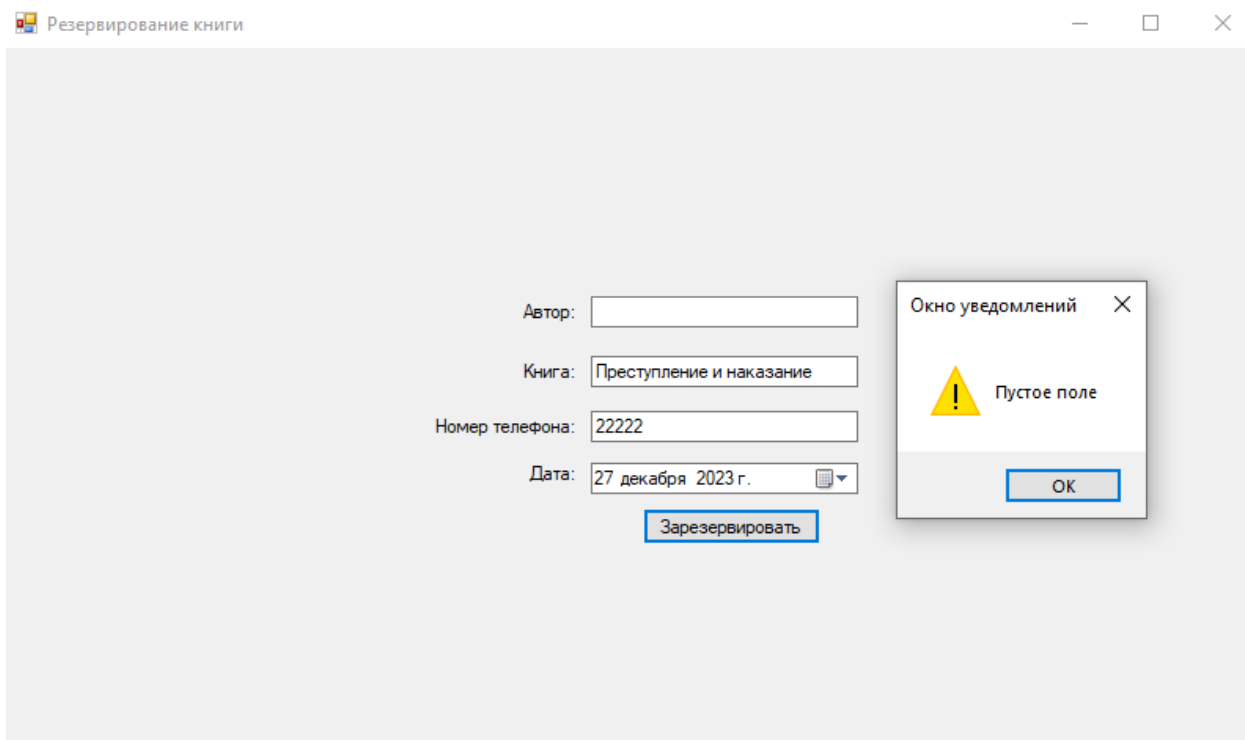


Рис. 2.10 Результат ввода пустых значений

Теперь введем корректные данные и продемонстрируем правильность работы приложения. Выводится сообщение об успешной резервации книги (рис. 2.11).

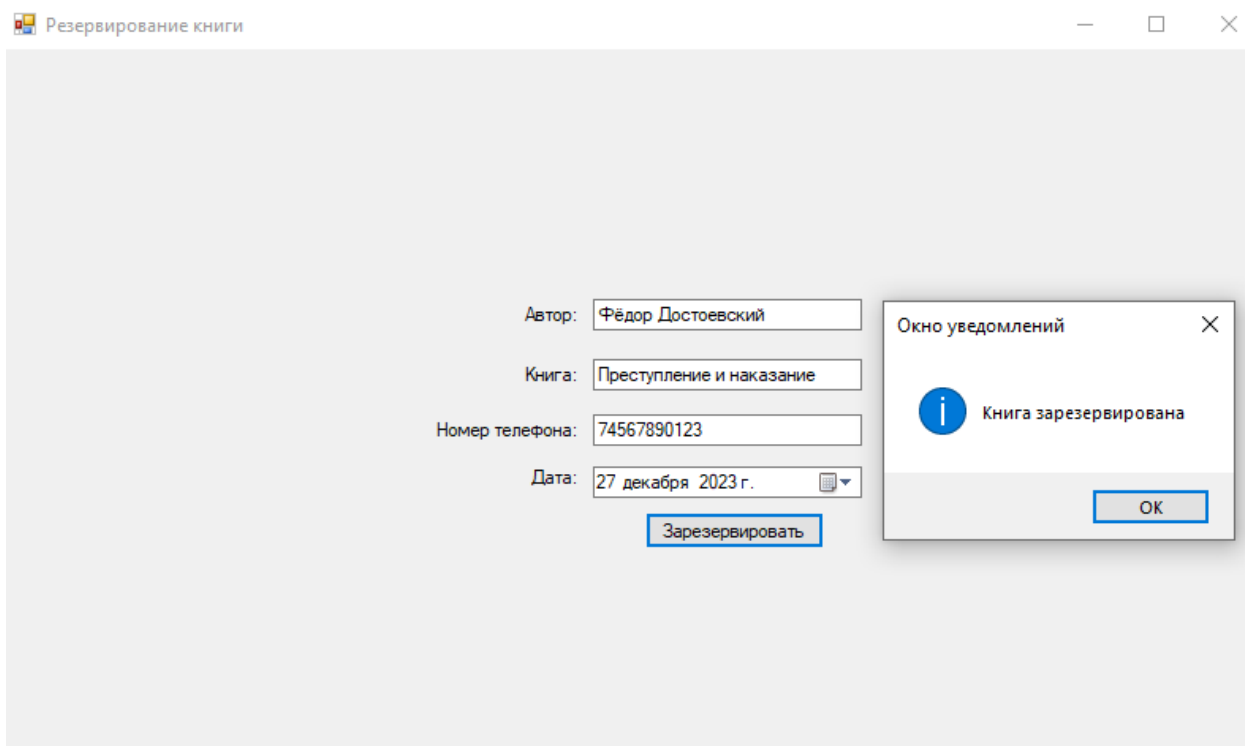


Рис. 2.11 Результат ввода корректных данных

Следующей на форме «Резервирование книги» располагается кнопка

«Удалить резервирование». При нажатии, система переводит пользователя на форму «Удаление резервирования» (рис. 2.12). При нажатии на кнопку «Удалить» срабатывает хранимая процедура «DeleteReservation».

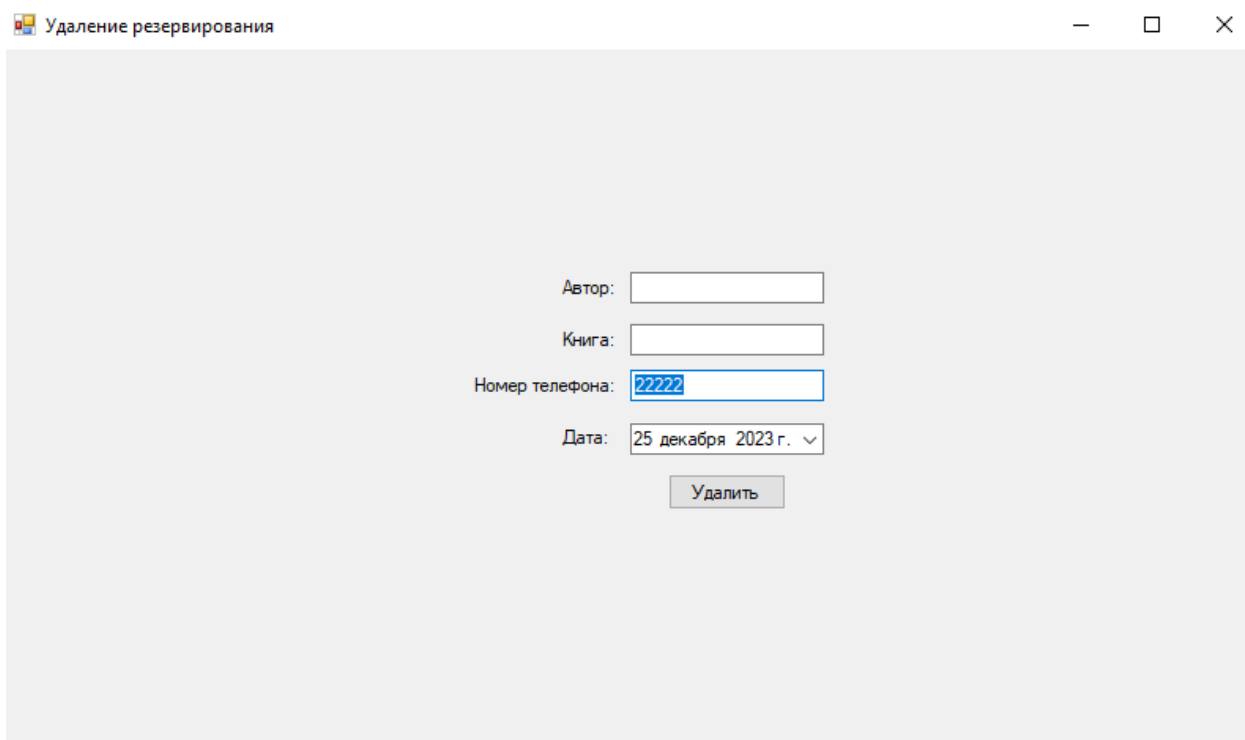


Рис. 2.12 Форма «Удаление резервирования»

При введении некорректных данных:

1. Несуществующей резервации (рис. 2.13);
2. Несуществующей книги (рис. 2.14);
5. Несуществующего номера телефона (рис. 2.15);
6. Пустых значений (рис. 2.16);

Появится окно, сообщающее об одной из данных ошибок.

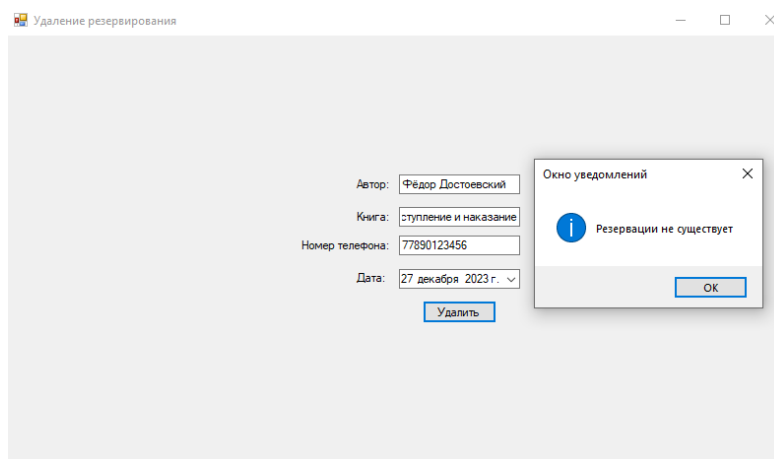


Рис. 2.13 Результат ввода несуществующей резервации

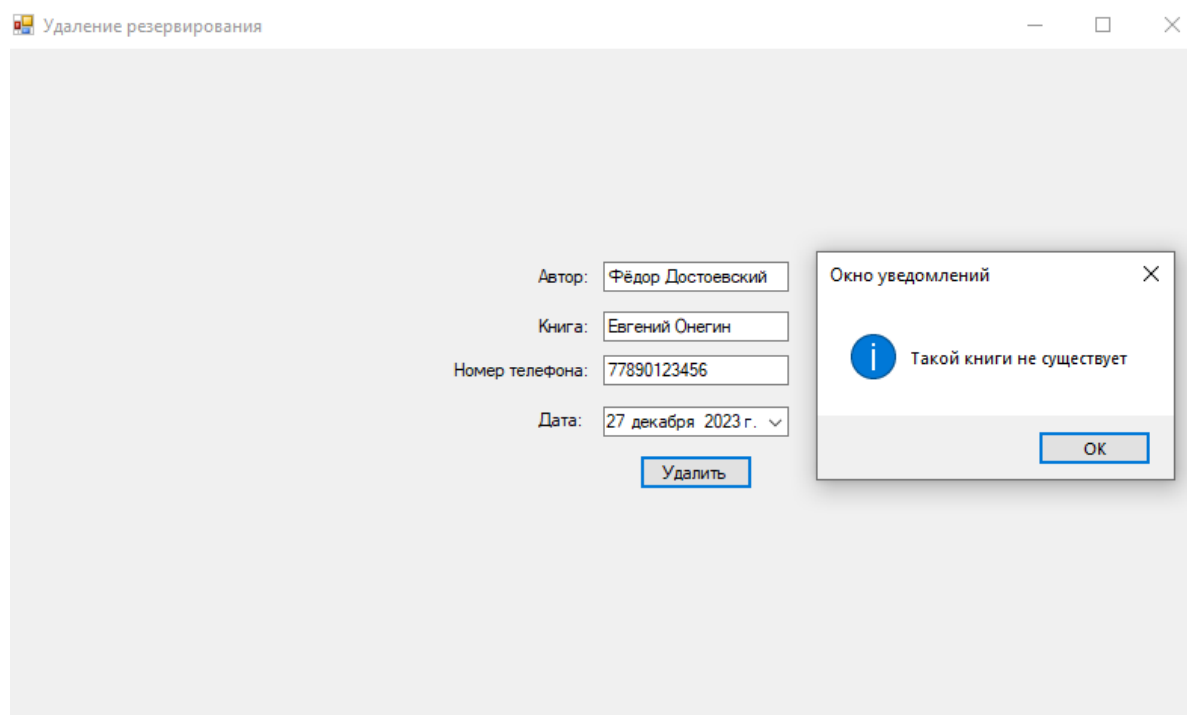


Рис. 2.14 Результат ввода несуществующей книги

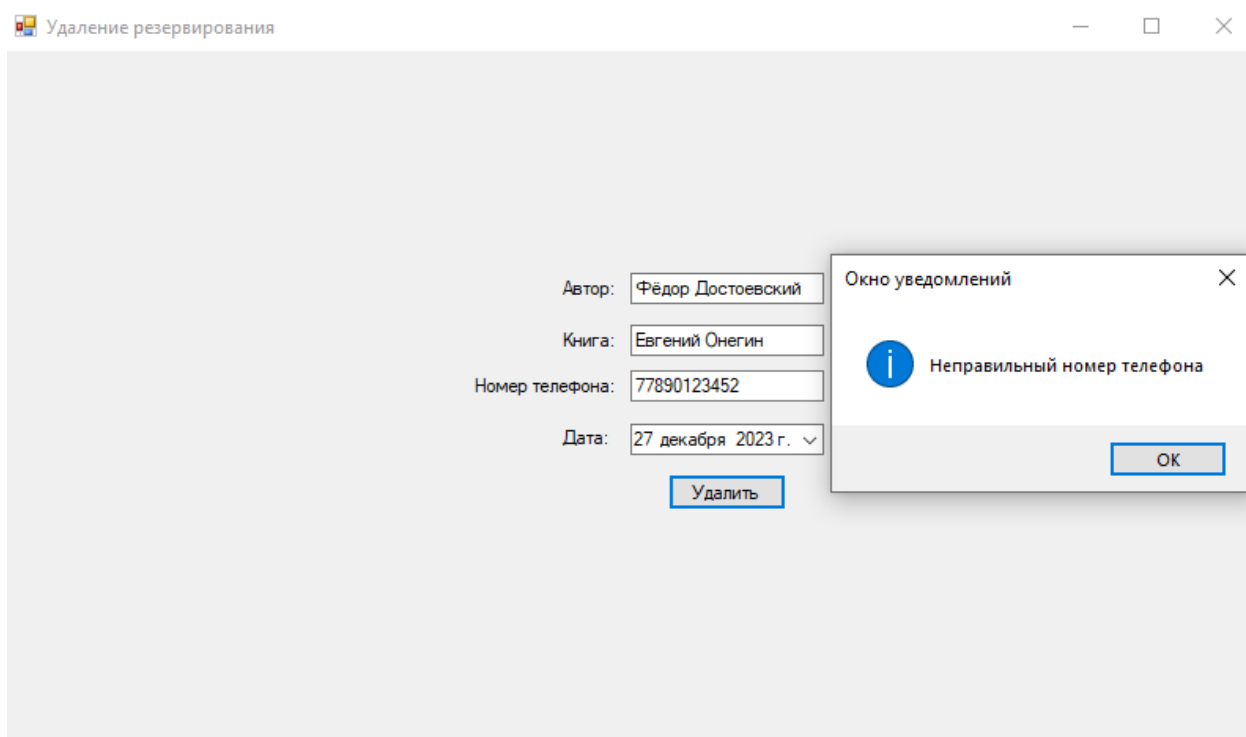


Рис. 2.15 Результат ввода несуществующего номера телефона

Хочется отметить, что обработка ввода пустых значений присутствует на всех формах, поэтому далее эта ошибка рассматриваться не будет.

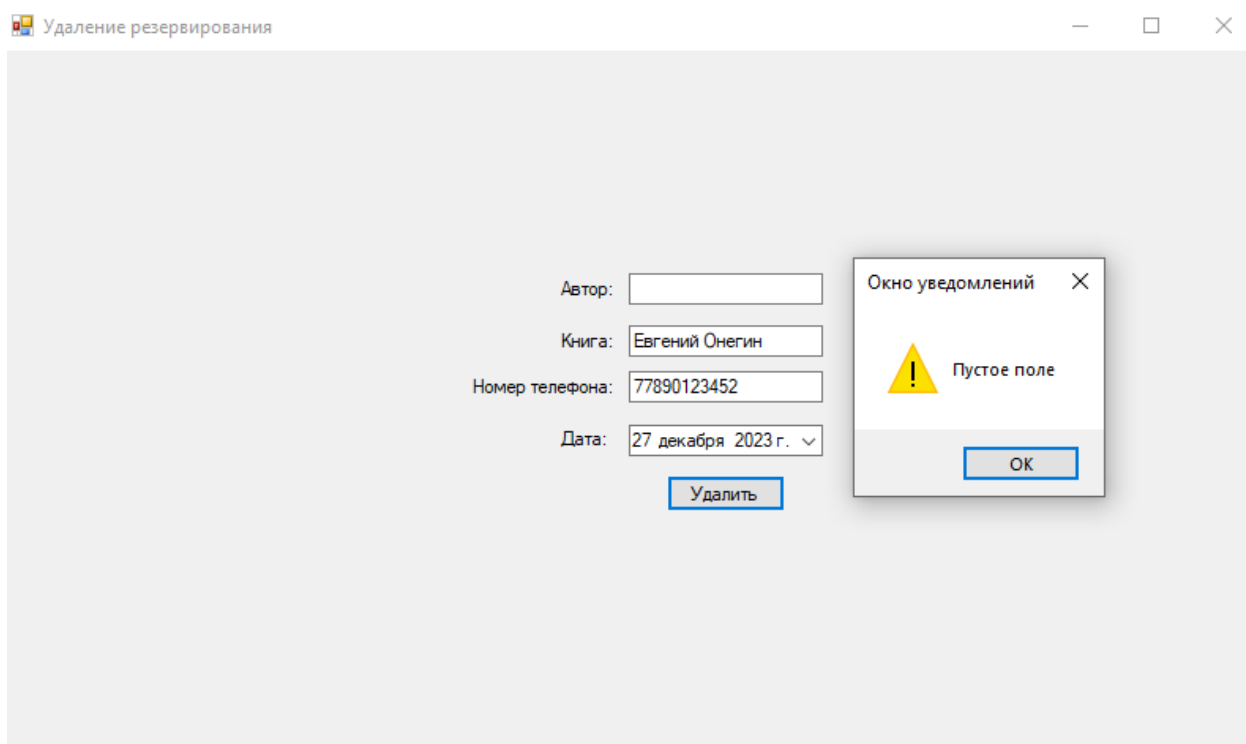


Рис. 2.16 Результат ввода пустого значения

Введем корректные данные. На рисунке 2.17 изображен вывод сообщения об успешном удалении записи о резервировании книги.

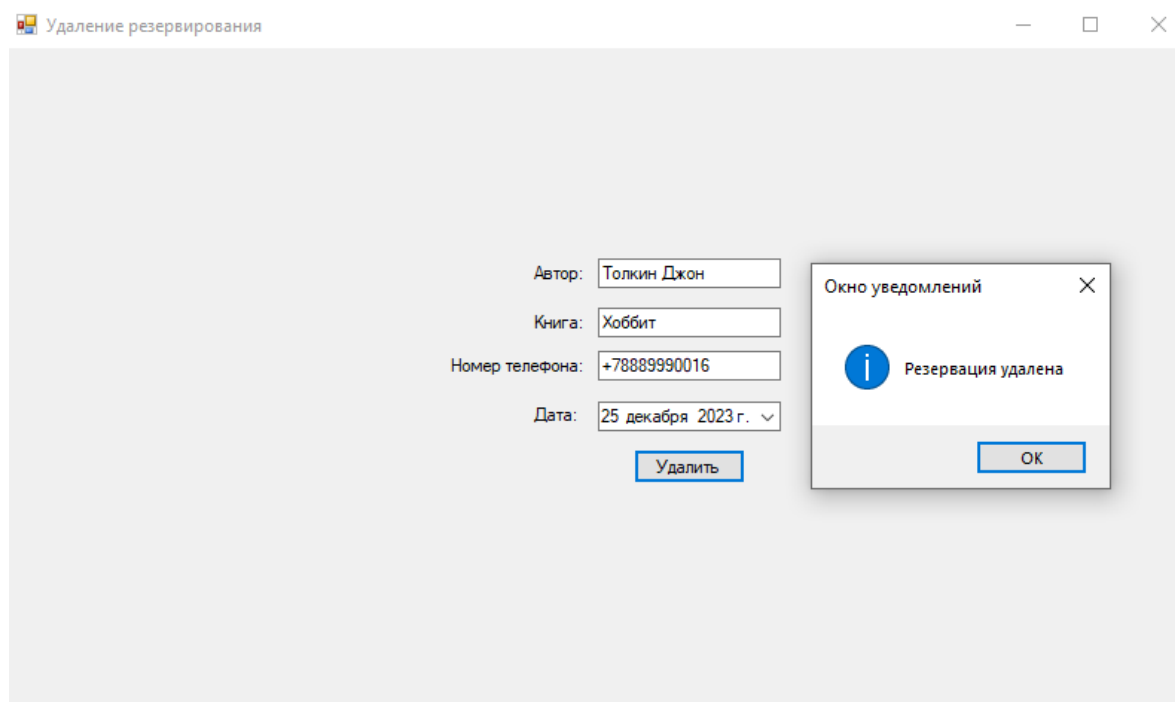


Рис. 2.17 Результат ввода корректных данных

Вернемся на форму добавления записи о резервировании. На ней располагается кнопка «Найти». Введем фамилию читателя и увидим выведенный список читателей с данной фамилией (рис. 2.18).

Резервирование книги

	ФИО Читателя	Номер телефона	Автор книги	Название книги
▶	Козлова Мария Дмитриевна	74567890123	Александр Блок	Стихи
	Козлова Маргарита Алексеевна	00011122333	Антон Чехов	Вишневый сад
	Козлова Алена Алексеевна	+70001132233	Агата Кристи	Десять негров
	Козлова Анастасия Дмитриевна	+74445552677	Иван Бунин	Темные аллеи
	Козлова Елизавета Алексеевна	+70004112233	Владимир Маяковский	Лирика
*				

Зарезервировать книгу

Удалить резервирование

Обновить данные

ФИО

Рис. 2.18 Результат ввода фамилии «Козлова» в поиск

Нажмем на кнопку «Обновить данные» и увидим список всех читателей (рис. 2.19). Хочется отметить, что кнопка «Обновить данные» присутствует на большинстве формах, поэтому далее рассматриваться данная кнопка не будет.

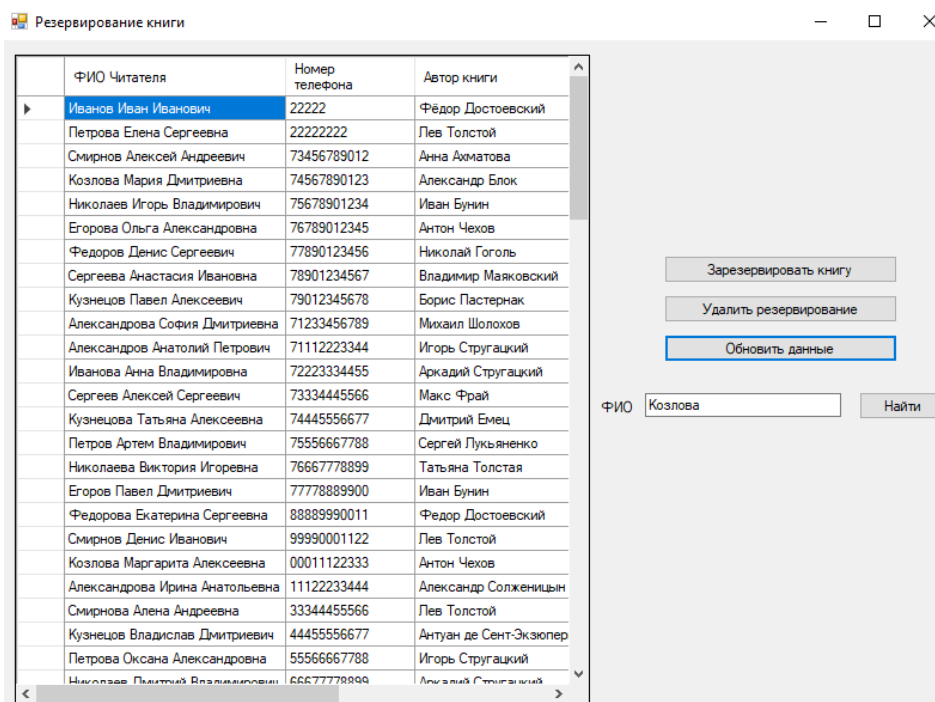


Рис. 2.19 Результат нажатия кнопки «Обновить данные»

Следующей на форме консультанта располагается кнопка «Выдать информацию о книгах». При нажатии на данную кнопку система переводит пользователя на форму «Поиск книги» (рис. 2.20). При нажатии на кнопку «Найти» срабатывает хранящая процедура «SearchBook».

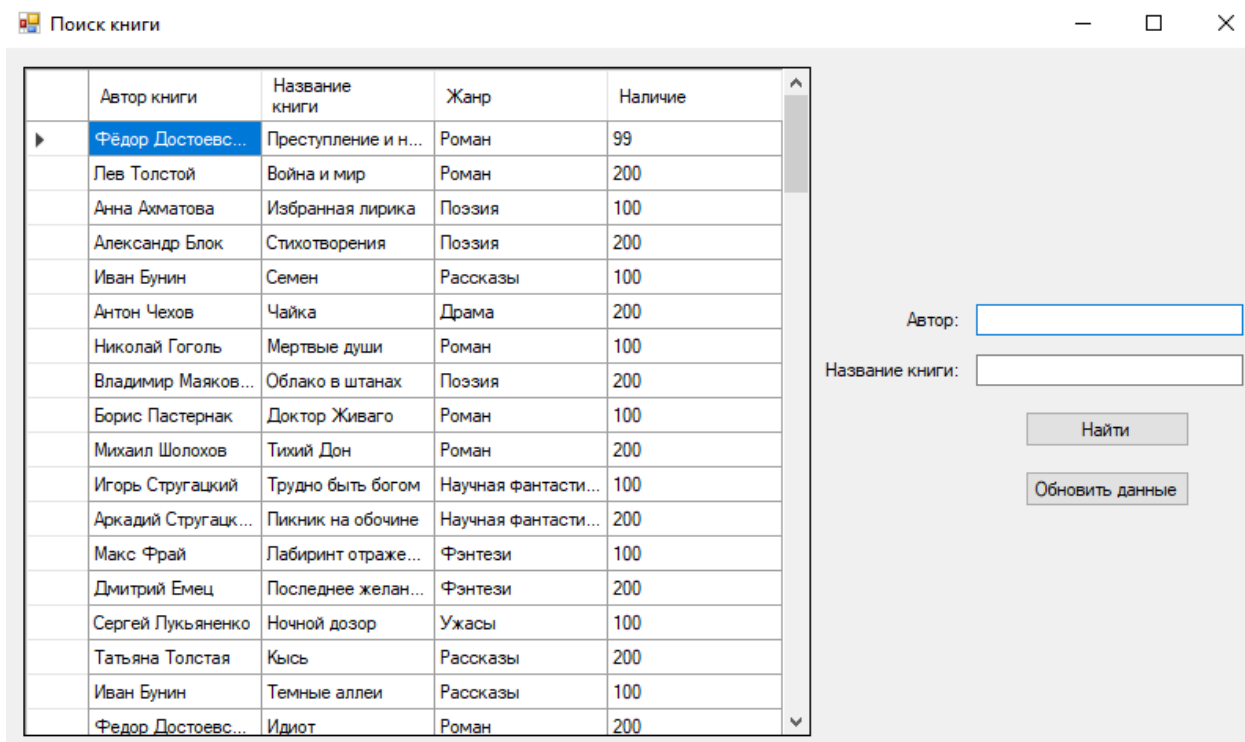


Рис. 2.20 Форма «Поиск книги»

При введении некорректных данных, а именно несуществующей книги,  
появится окно, сообщающее об ошибке (рис 2.21).

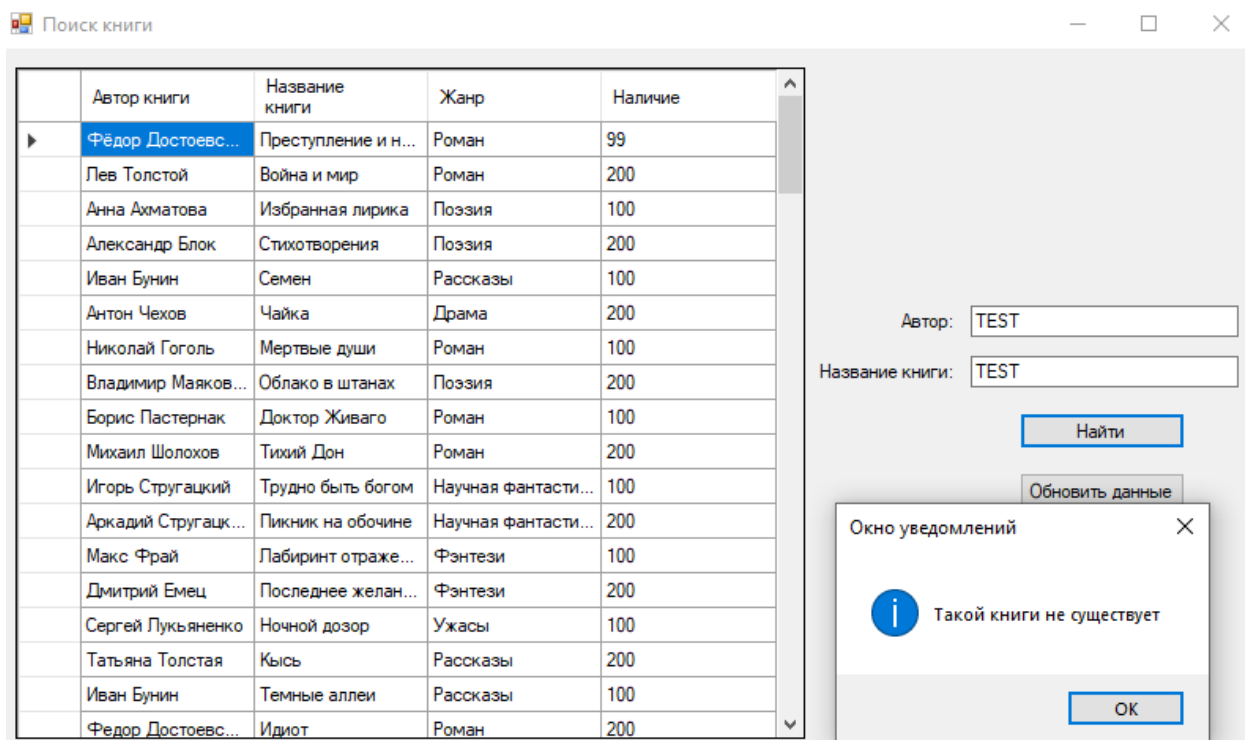


Рис. 2.21 Результат ввода несуществующей книги

Введем корректные значения, заполнив два поля. Выведется сообщение об успешном поиске книги и отображение книги в dataGridViewView (рис. 2.22, 2.23).

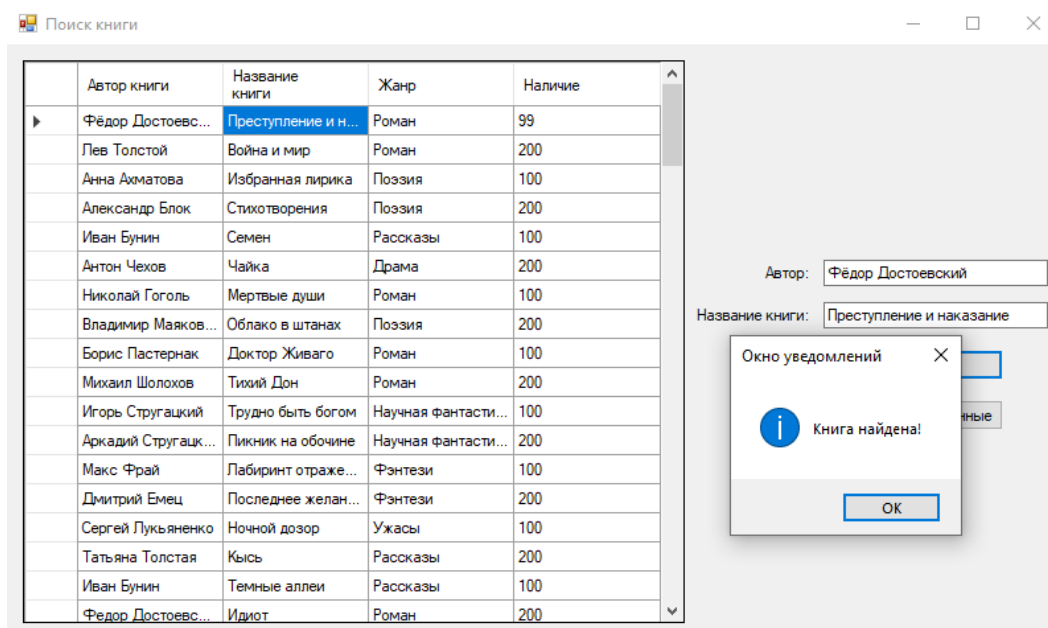


Рис. 2.22 Сообщение о нахождении книги

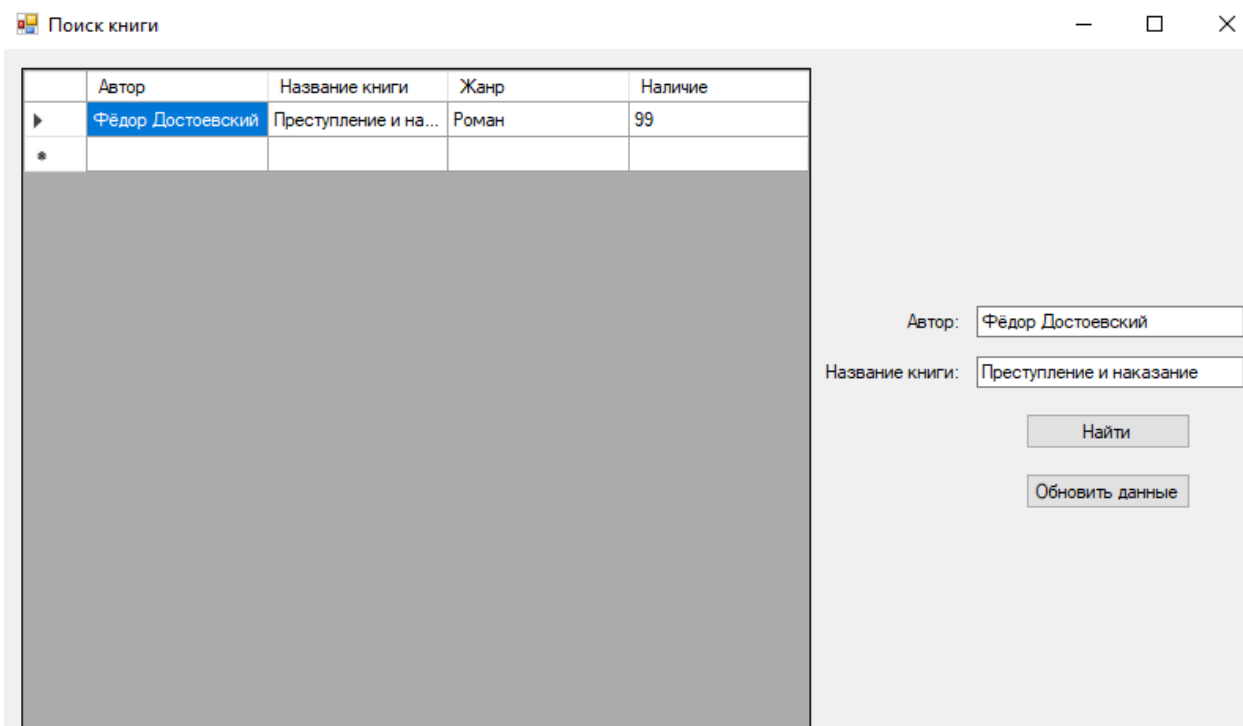


Рис. 2.23 Результат ввода корректных данных

Теперь введем корректное значение, заполнив одно из полей. Книги успешно отобразятся в dataGridView (рис. 2.24, 2.25).

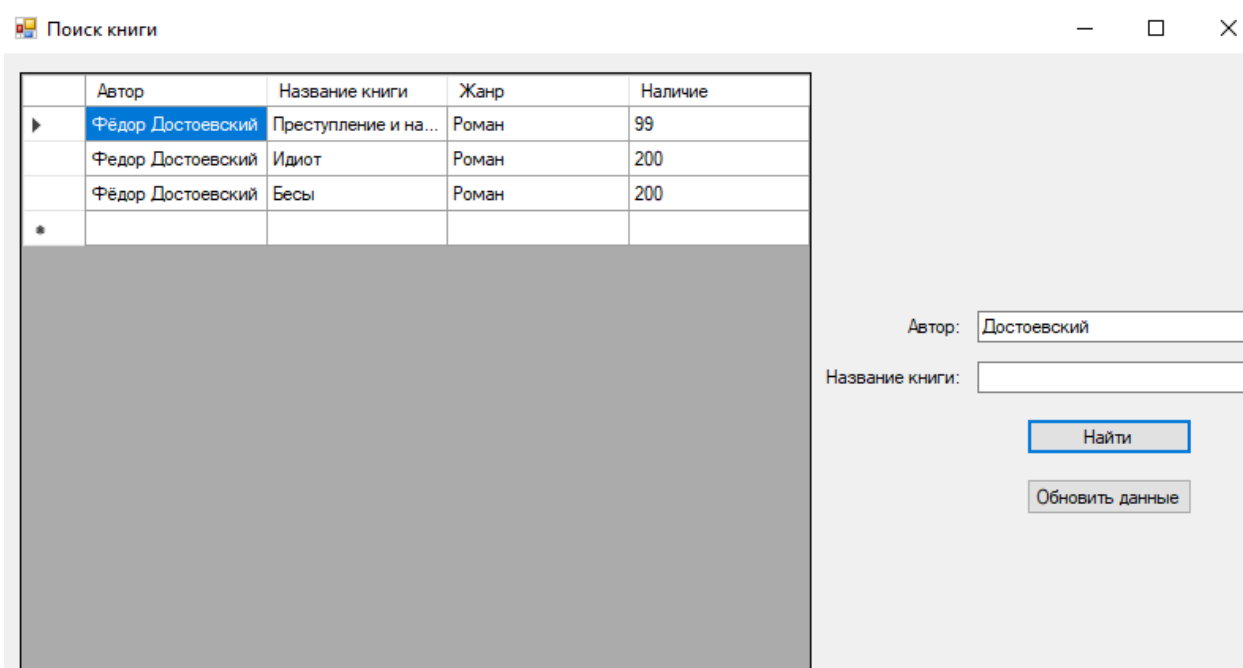
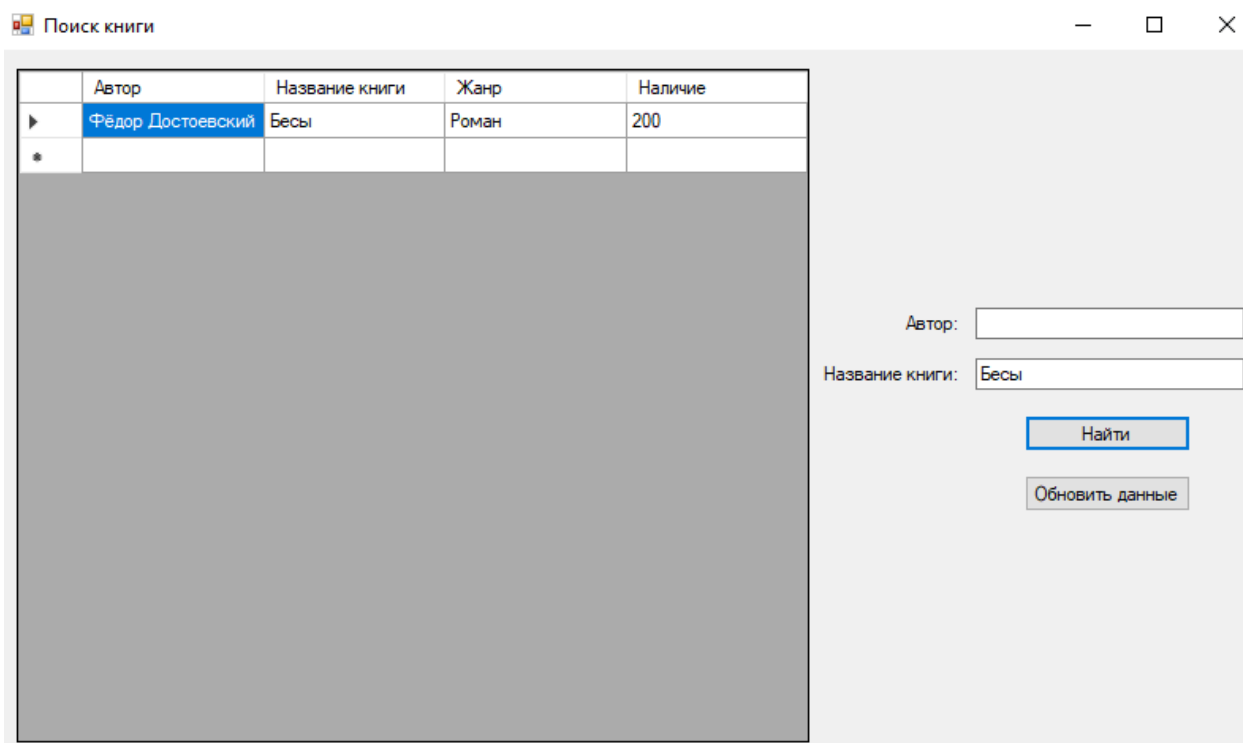


Рис. 2.24 Результат ввода фамилии «Достоевский»





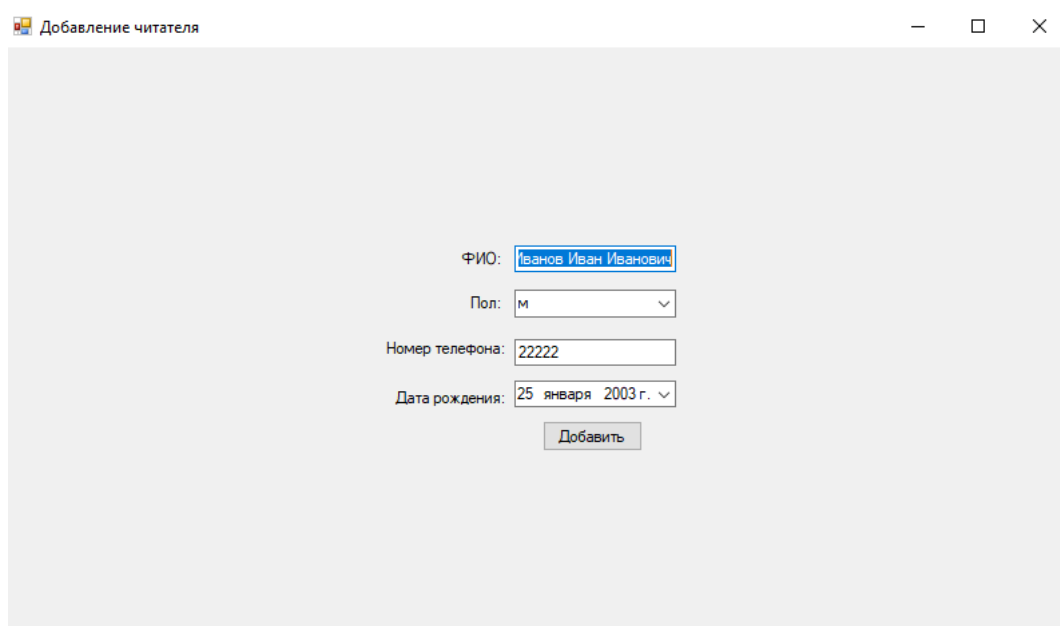
	Автор	Название книги	Жанр	Наличие
▶	Фёдор Достоевский	Бесы	Роман	200
*				

Автор:

Название книги:

Рис. 2.25 Результат ввода названия книги «Бесы»

Следующей на форме консультанта располагается кнопка «Добавить нового читателя». При нажатии на данную кнопку система переводит пользователя на форму «Добавление читателя» (рис. 2.26). При нажатии на кнопку «Добавить» срабатывает хранимая процедура «RegisterReader» и триггер «RegReader».



ФИО:

Пол:

Номер телефона:

Дата рождения:

Рис. 2.26 Форма «Добавление читателя»

При введении некорректных данных:

1. Уже зарегистрированного номера телефона (рис. 2.27);
2. Некорректного пола (рис. 2.28);

Появится окно, сообщающее об одной из данных ошибок.

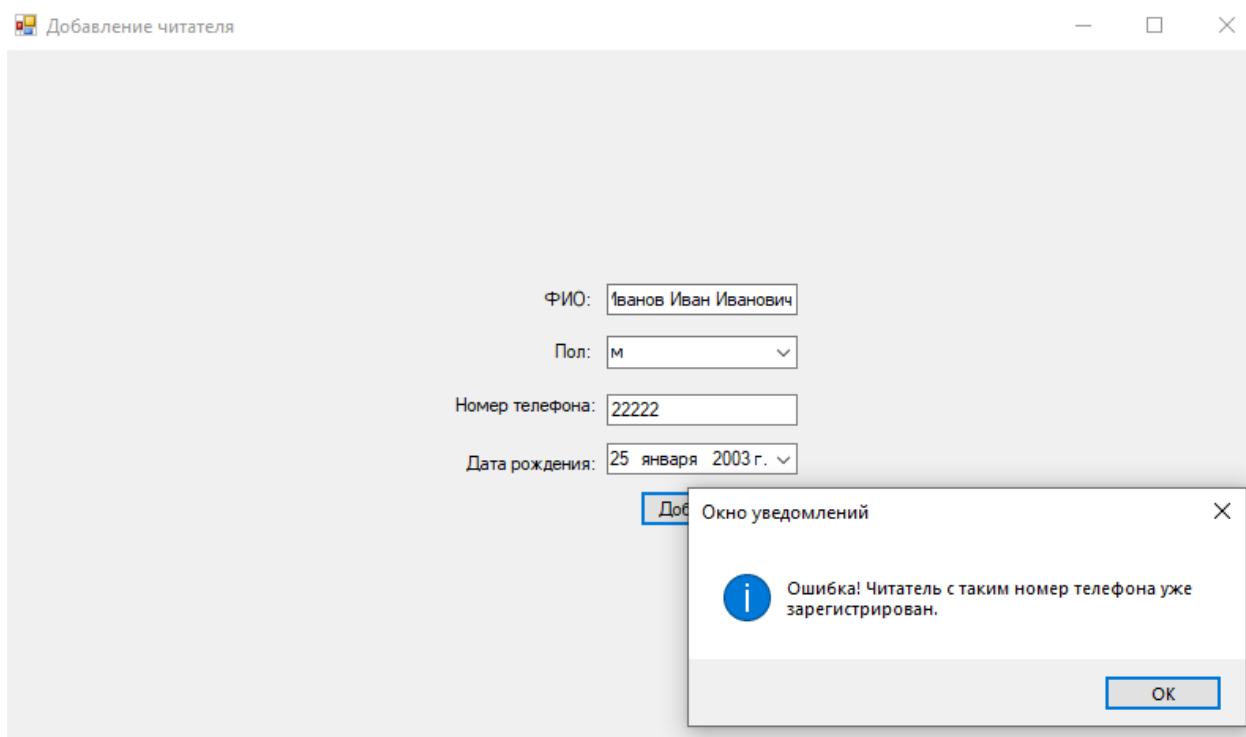


Рис. 2.27 Результат ввода зарегистрированного номера телефона

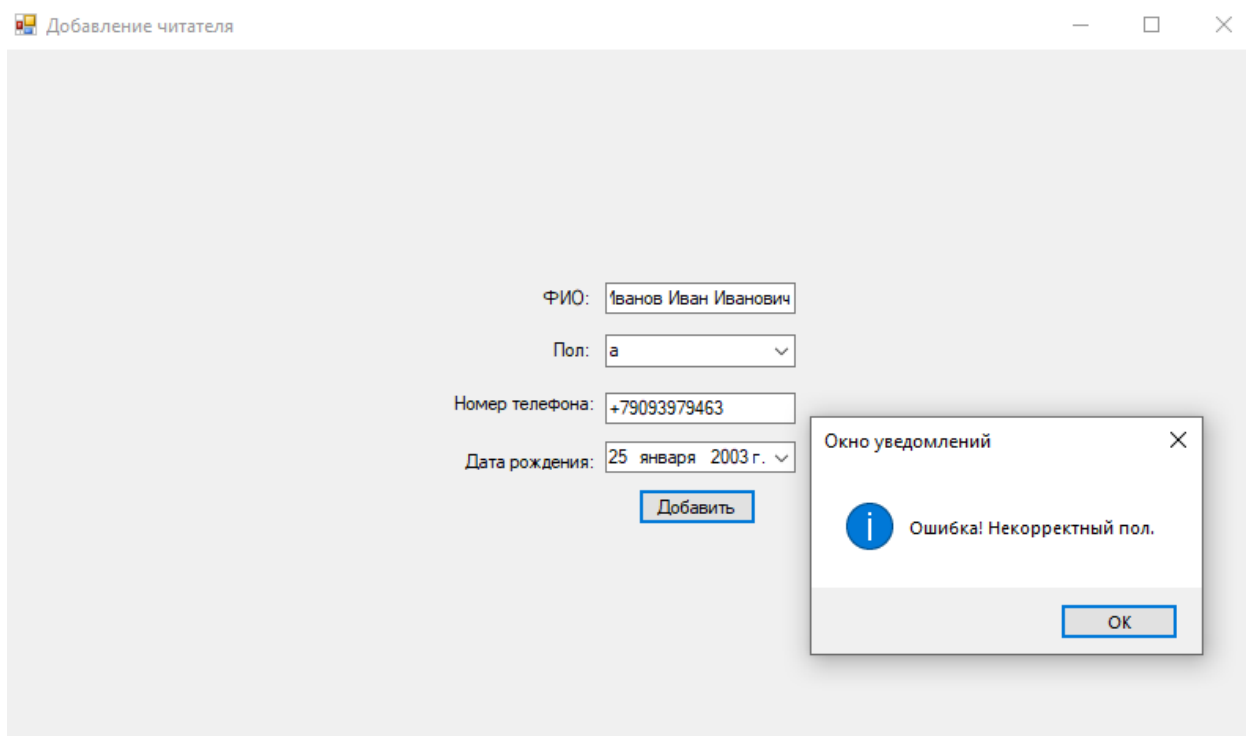


Рис. 2.28 Результат ввода некорректного пола

Теперь введем корректные значения, заполнив все поля. Выведется

сообщение об успешной регистрации читателя (рис. 2.29).

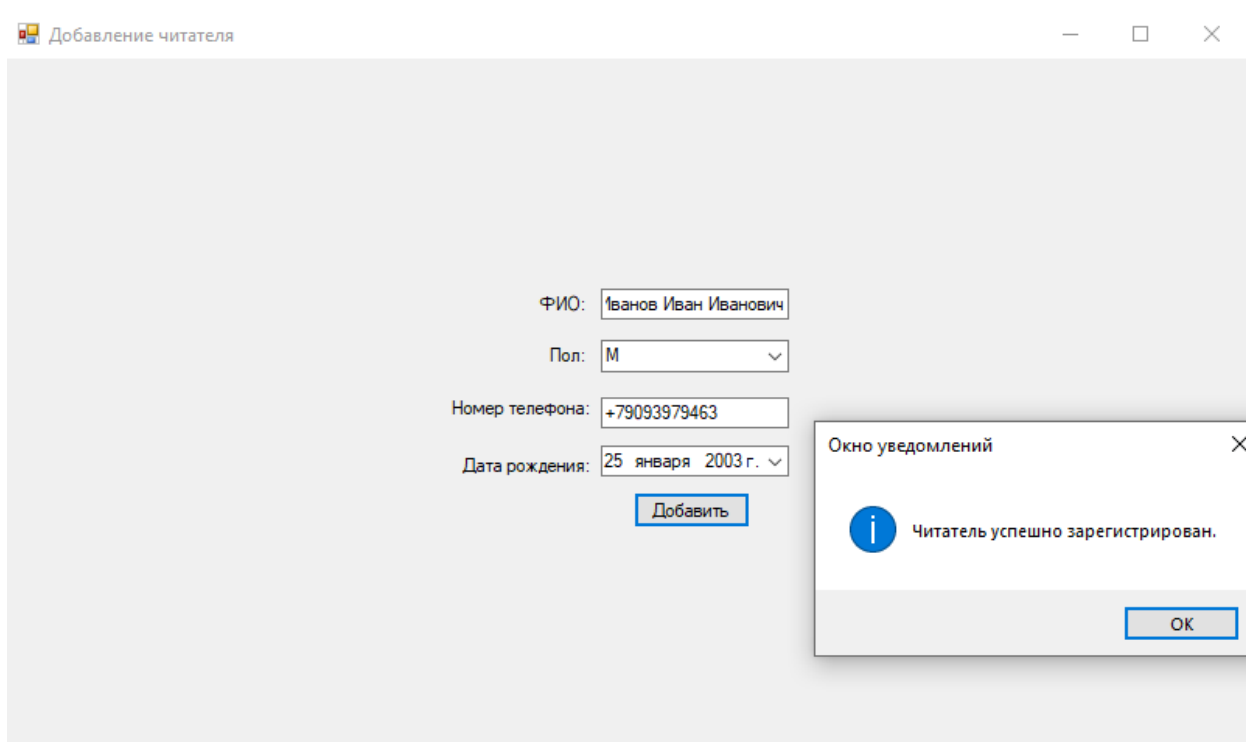


Рис. 2.29 Результат ввода корректных значений

Следующей на форме консультанта располагается кнопка «Выдать информацию о читателях». При нажатии на данную кнопку система переводит пользователя на форму «Поиск читателя» (рис. 2.30). При нажатии на кнопку «Найти» срабатывает хранимая процедура «SearchReader».

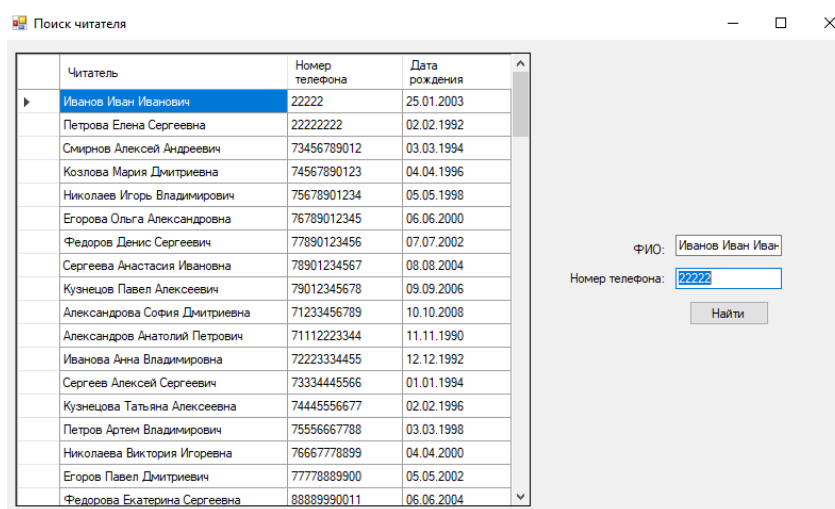


Рис. 2.30 Форма «Поиск читателя»

При введении некорректных ФИО или номера телефона, появляется ошибка (рис. 2.31).

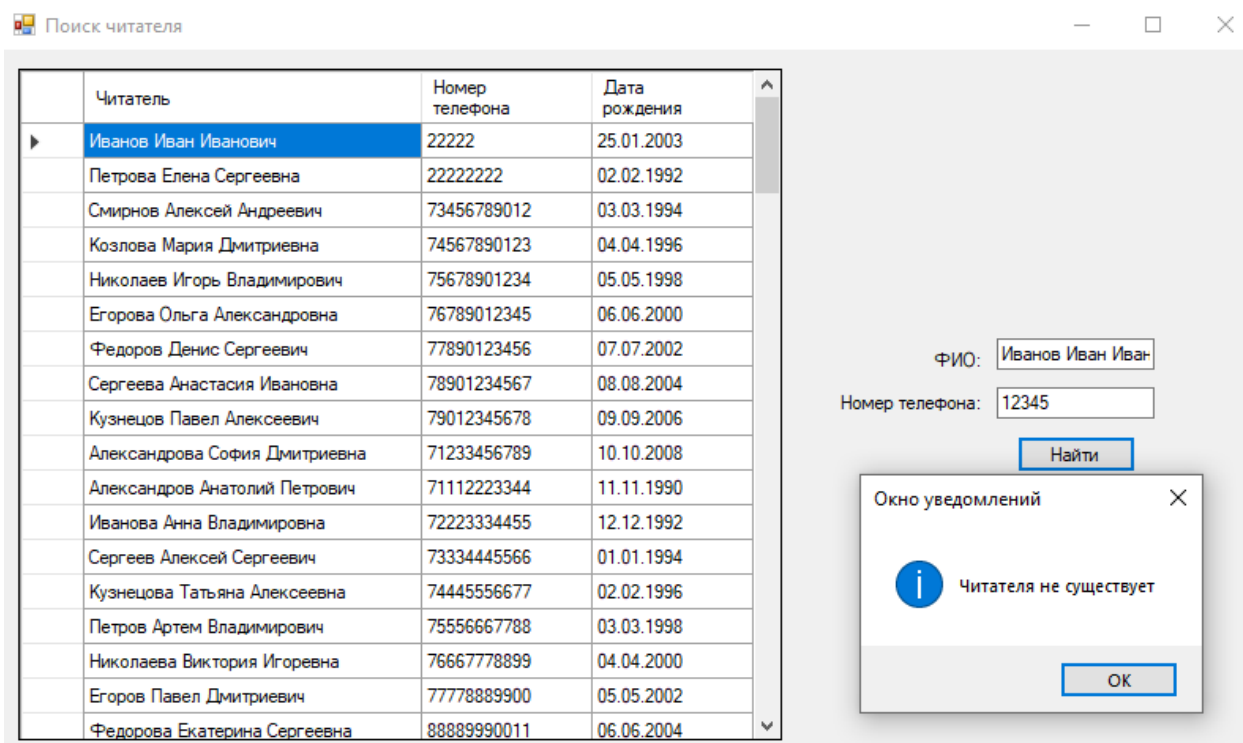


Рис. 2.31 Результат ввода несуществующего номера телефона

Введем корректные ФИО и номер телефона, появляется сообщение о нахождении читателя и вывод в dataGridView (рис. 2.32, 2.33).

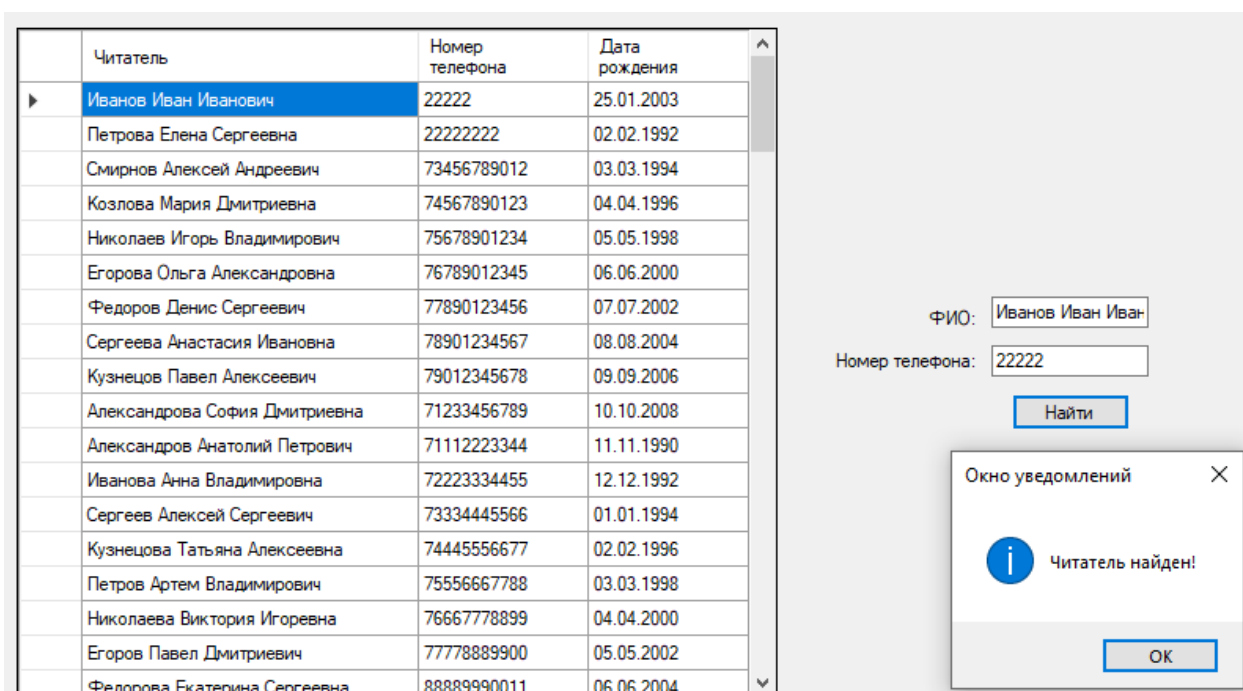


Рис. 2.32 сообщение о нахождении читателя

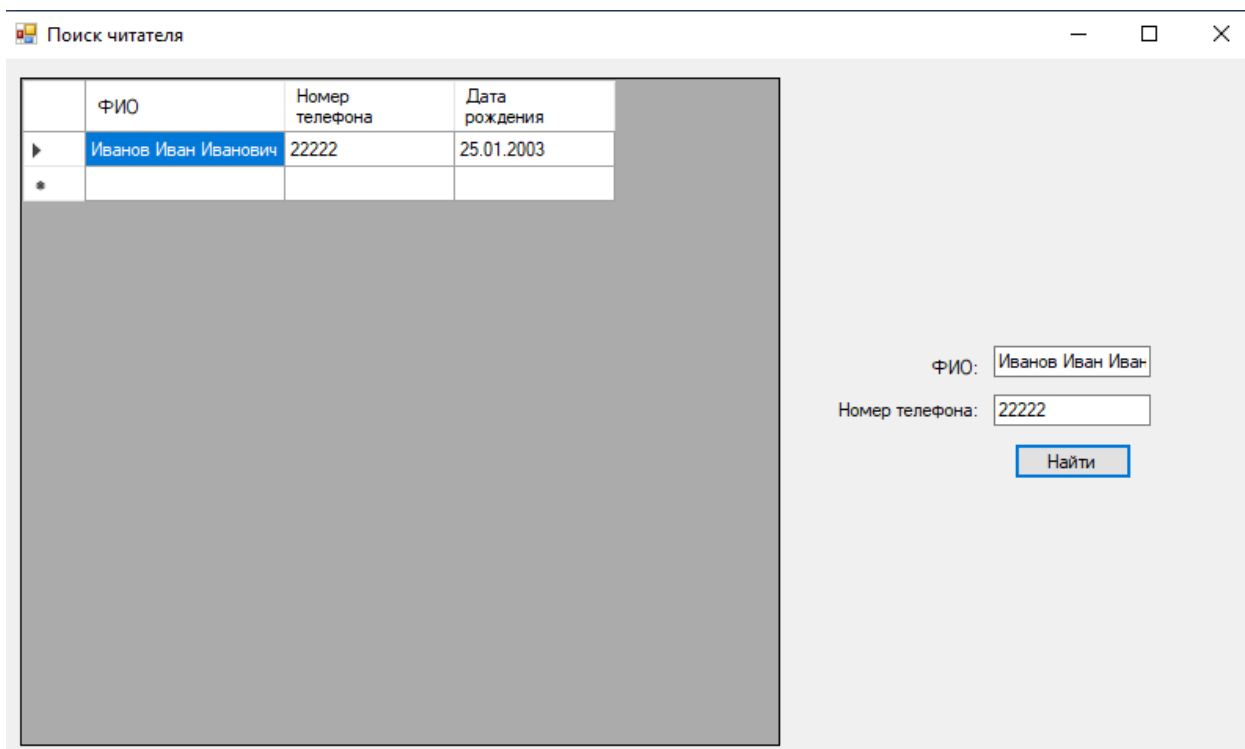


Рис. 2.33 Результат ввода корректных ФИО и номера телефона

Заполним поле ФИО, поле номер телефона оставим пустым, происходит вывод читателей в dataGridView (рис. 2.34).

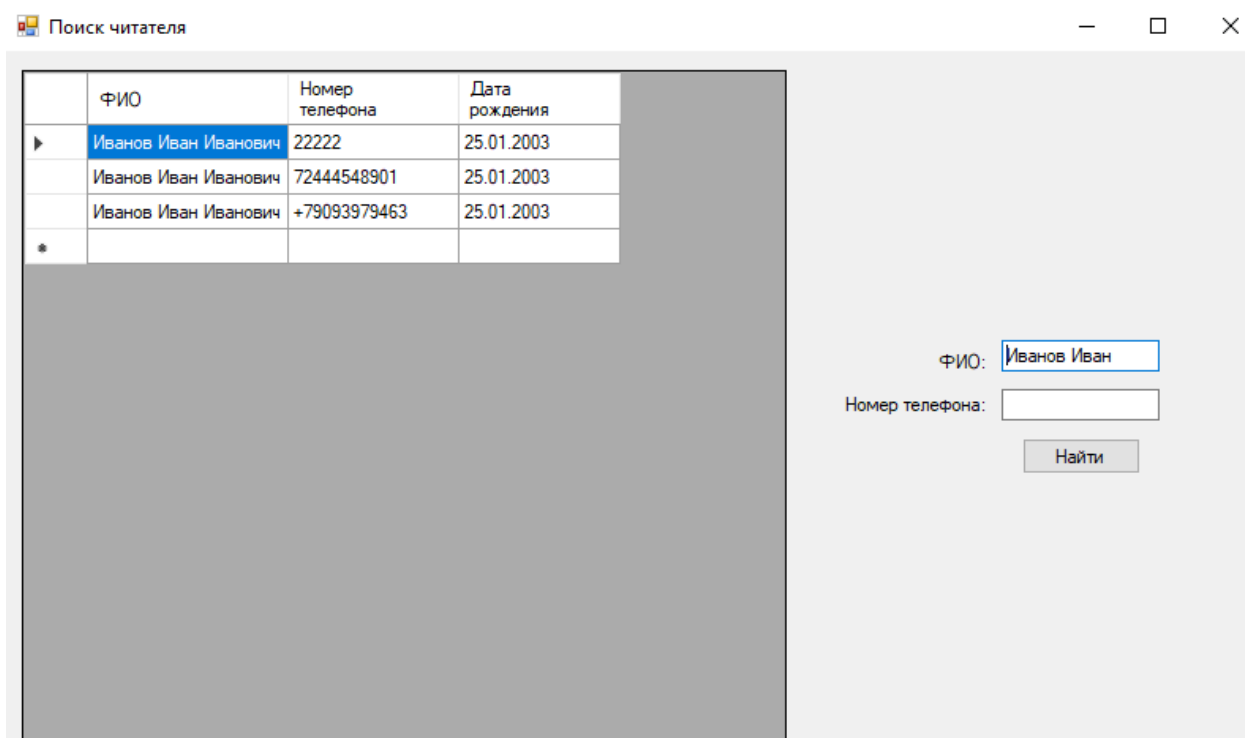


Рис. 2.34 Результат ввода корректного ФИО

Заполним поле номер телефона, поле ФИО оставим пустым, происходит вывод читателей в dataGridView (рис. 2.35).

	ФИО	Номер телефона	Дата рождения
▶	Иванов Иван Иванович	72444548901	25.01.2003
*			

ФИО:

Номер телефона:

Рис. 2.35 Результат ввода корректного номера телефона

Следующей на форме консультанта располагается кнопка «Изменить данные читателя». При нажатии на данную кнопку система переводит пользователя на форму «Изменить данные читателя» (рис. 2.36). При нажатии на кнопку «Изменить данные» срабатывает хранимая процедура «UpdateReader».

	ID	Читатель	Номер телефона	Дата рождения
▶	1	Иванов Иван И...	22222	25.01.2003
	2	Петрова Елена ...	22222222	02.02.1992
	3	Смирнов Алекс...	73456789012	03.03.1994
	4	Козлова Мария ...	74567890123	04.04.1996
	5	Николаев Игор...	75678901234	05.05.1998
	6	Егорова Ольга ...	76789012345	06.06.2000
	7	Федоров Денис...	77890123456	07.07.2002
	8	Сергеева Анаст...	78901234567	08.08.2004
	9	Кузнецов Павел...	79012345678	09.09.2006
	10	Александрова С...	71233456789	10.10.2008
	11	Александров Ан...	71112223344	11.11.1990
	12	Иванова Анна В...	72223334455	12.12.1992
	13	Сергеев Алексе...	73334445566	01.01.1994
	14	Кузнецова Тать...	74445556677	02.02.1996
	15	Петров Артем В...	75556667788	03.03.1998
	16	Николаева Викт...	76667778899	04.04.2000
	17	Егоров Павел Д...	77778889900	05.05.2002

Номер телефона:

ID Читателя

Новый номер телефона:

Рис. 2.36 Форма «Изменить данные читателя»

При введении некорректных данных:

1. Уже зарегистрированного номера телефона (рис. 2.37);
2. Некорректного ID читателя (рис. 2.38);

Появится окно, сообщающее об одной из данных ошибок.

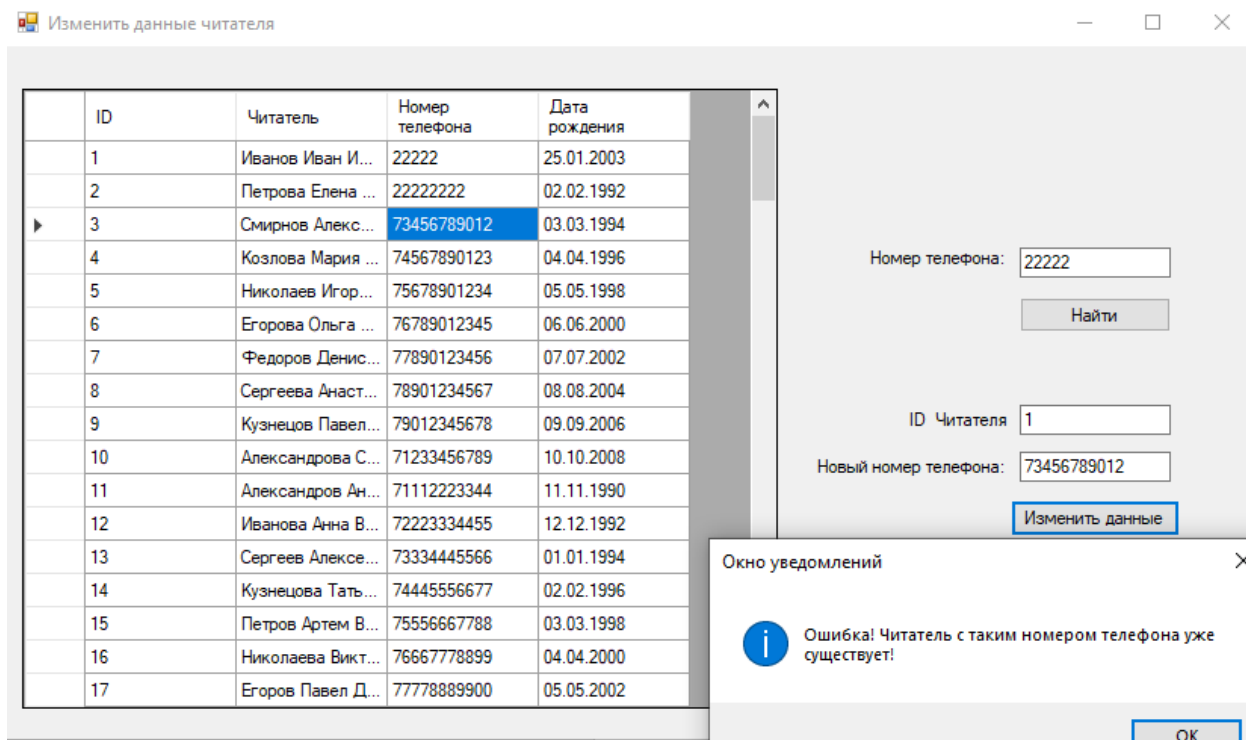


Рис. 2.37 Результат ввода зарегистрированного номера телефона

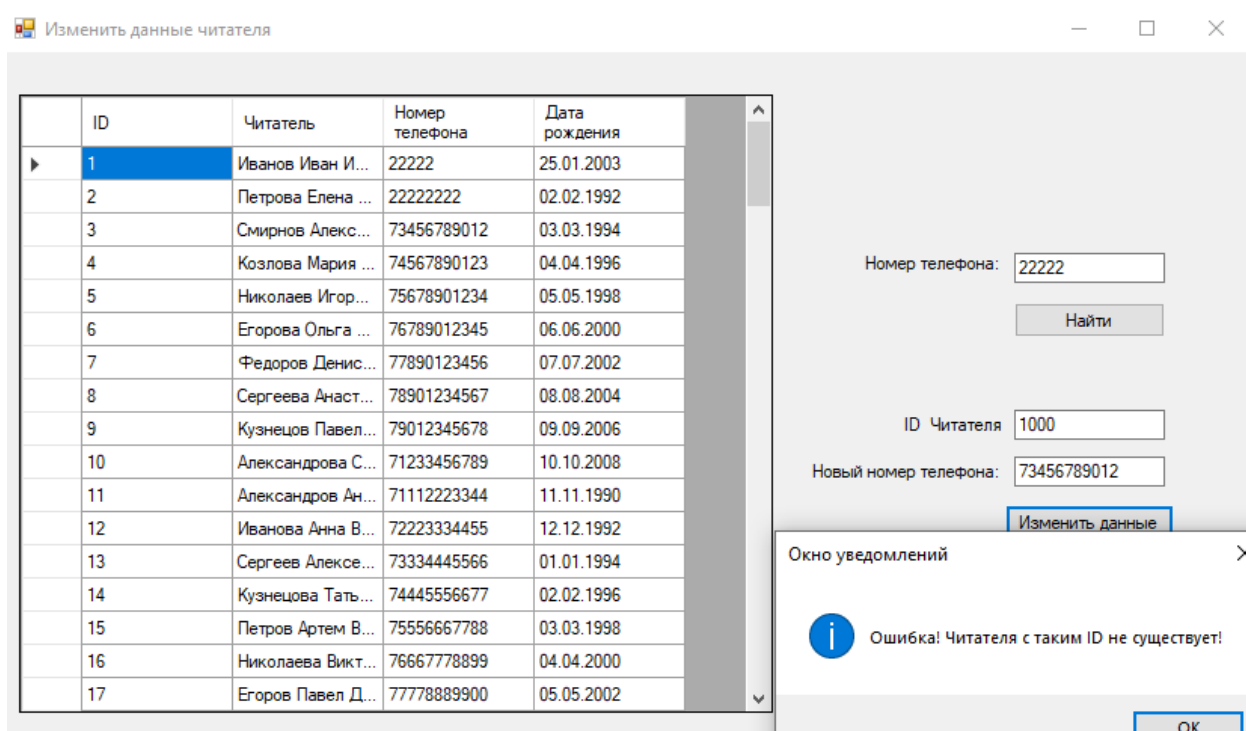


Рис. 2.38 Результат ввода несуществующего ID читателя

Введем корректные ID читателя и новый номер телефона, появляется сообщение об успешном изменении номера и происходит обновление в dataGridViewView (рис. 2.39, 2.40).

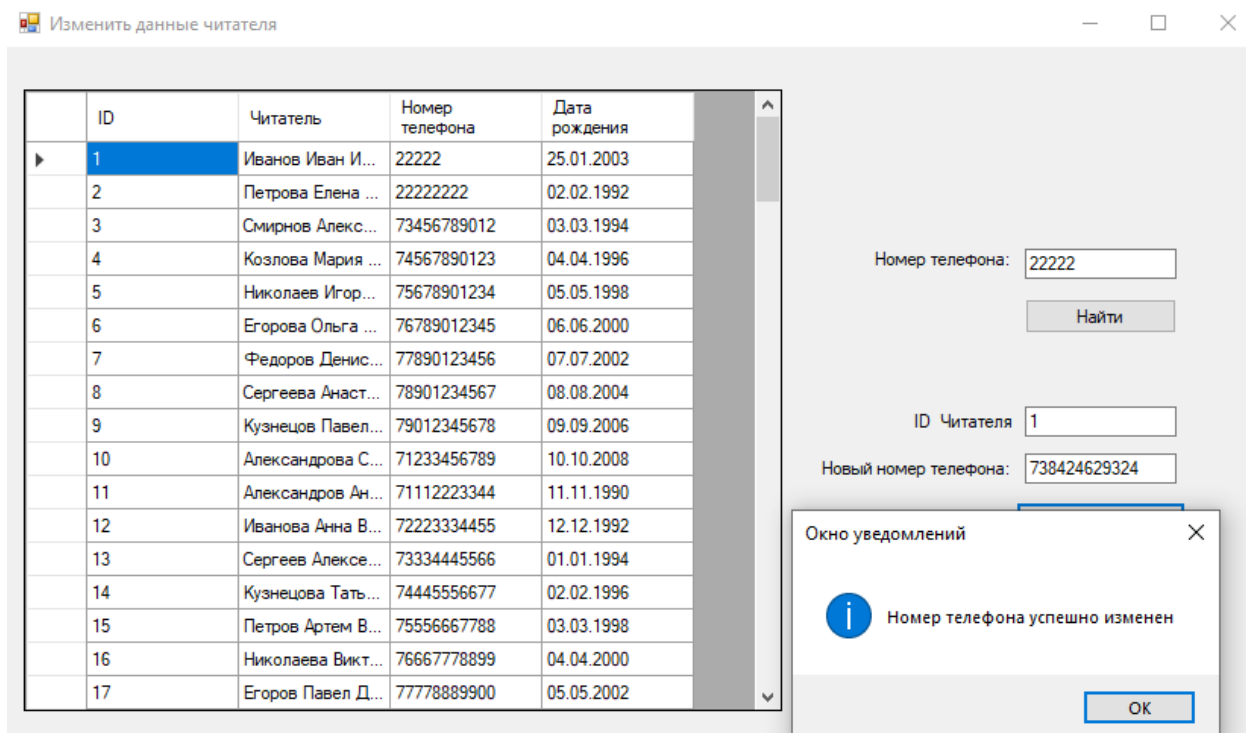


Рис. 2.39 Вывод сообщения об изменении номера телефона

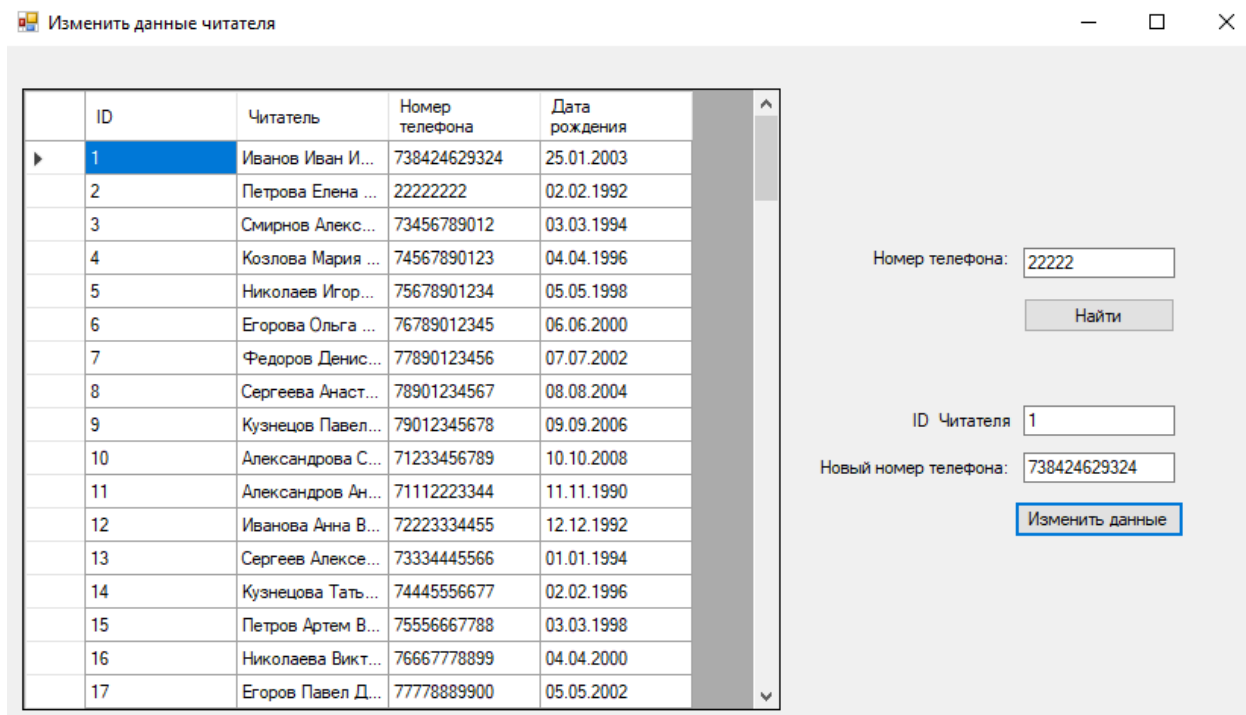


Рис. 2.39 Обновление номера телефона читателя с ID 1 в dataGridViewView

На этом функции консультанта заканчиваются. Перейдем к функциям библиотекаря (рис. 2.40). Последние четыре действия были



описаны в функциях консультанта. Поэтому перейдем к выдаче и принятию книг.

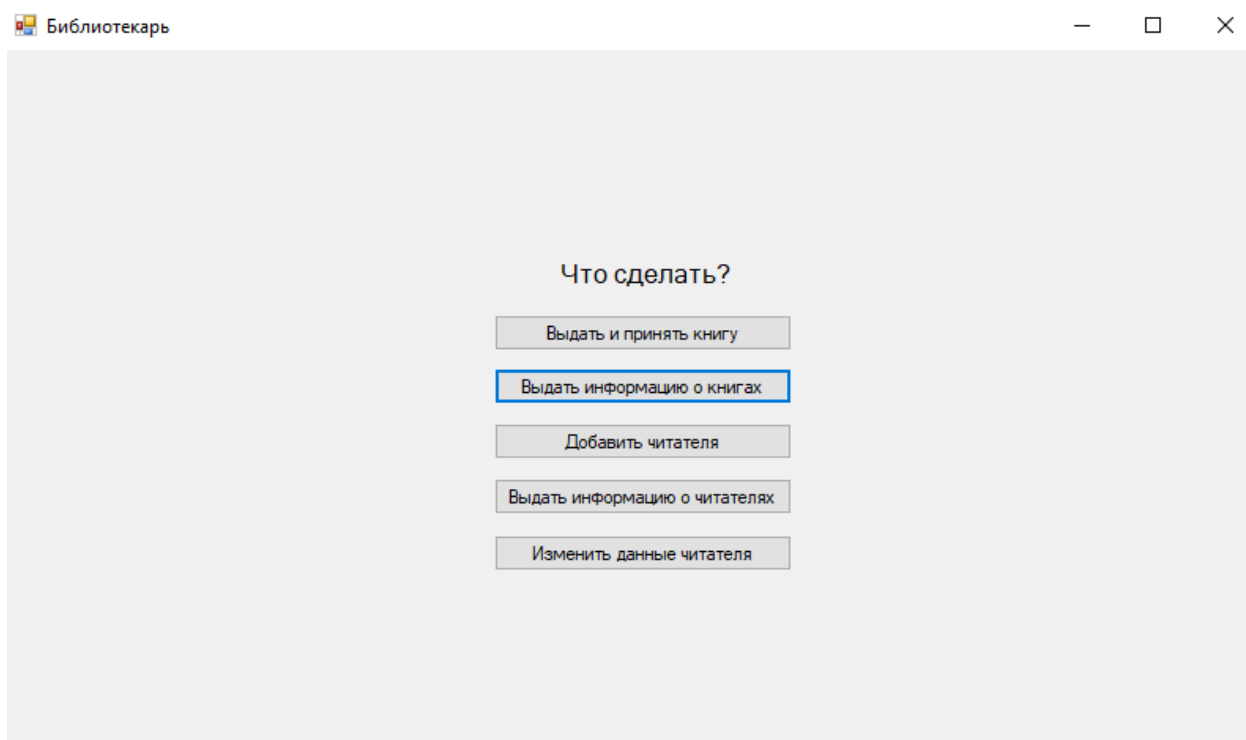


Рис. 2.40 Форма «Библиотекарь»

При нажатии на кнопку «Выдать и принять книгу» открывается форма «Выдача книги» (рис. 2.41) При нажатии на кнопку «Выдать книгу» происходит добавление записи о выдаче книги и срабатывают хранимая процедура «BookIssuance». При нажатии на кнопку «Принять книгу» происходит добавление записи о сдаче книги и срабатывают хранимая процедура «AcceptBook».

	Айди	Автор	Название книги	Номер телефона читателя
▶	1	Фёдор Достоев...	Преступление и...	738424629324
	2	Лев Толстой	Война и мир	22222222
	3	Анна Ахматова	Избранная лири...	73456789012
	4	Александр Блок	Стихотворения	74567890123
	5	Иван Бунин	Семен	75678901234
	6	Антон Чехов	Чайка	76789012345
	7	Николай Гоголь	Мертвые души	77890123456
	8	Владимир Маяк...	Облако в штанах	78901234567
	9	Борис Пастернак	Доктор Живаго	79012345678
	10	Михаил Шолохов	Тихий Дон	71233456789
	11	Игорь Стругацкий	Трудно быть бо...	71112223344
	12	Аркадий Струга...	Пикник на обоч...	72223334455
	13	Макс Фрай	Лабиринт отраж...	73334445566
	14	Дмитрий Емец	Последнее жел...	74445556677
<	15	Сергей Лукьянч...	Ночный взор...	75556667788

Автор:

Название книги:

Телефон читателя:

Телефон работника:

Дата выдачи:  ▼

Дата сдачи:  ▼

Id Выдачи/Сдачи:

Рис. 2.41 Форма «Выдача книги»

При введении некорректных данных:

1. Неправильной даты (рис. 2.42, 2.43);
2. Сотрудника, который был уволен (рис. 2.44);
3. Книги, которая уже выдана читателю (рис. 2.45);
4. Книги, которой нет в наличии (рис. 2.46);
5. Несуществующей книги (рис. 2.47)
6. Неправильного номера телефона читателя или сотрудника (рис. 2.48, 2.49)
7. Несуществующего ID выдачи (рис. 2.50)

Появится окно, сообщающее об одной из данных ошибок.

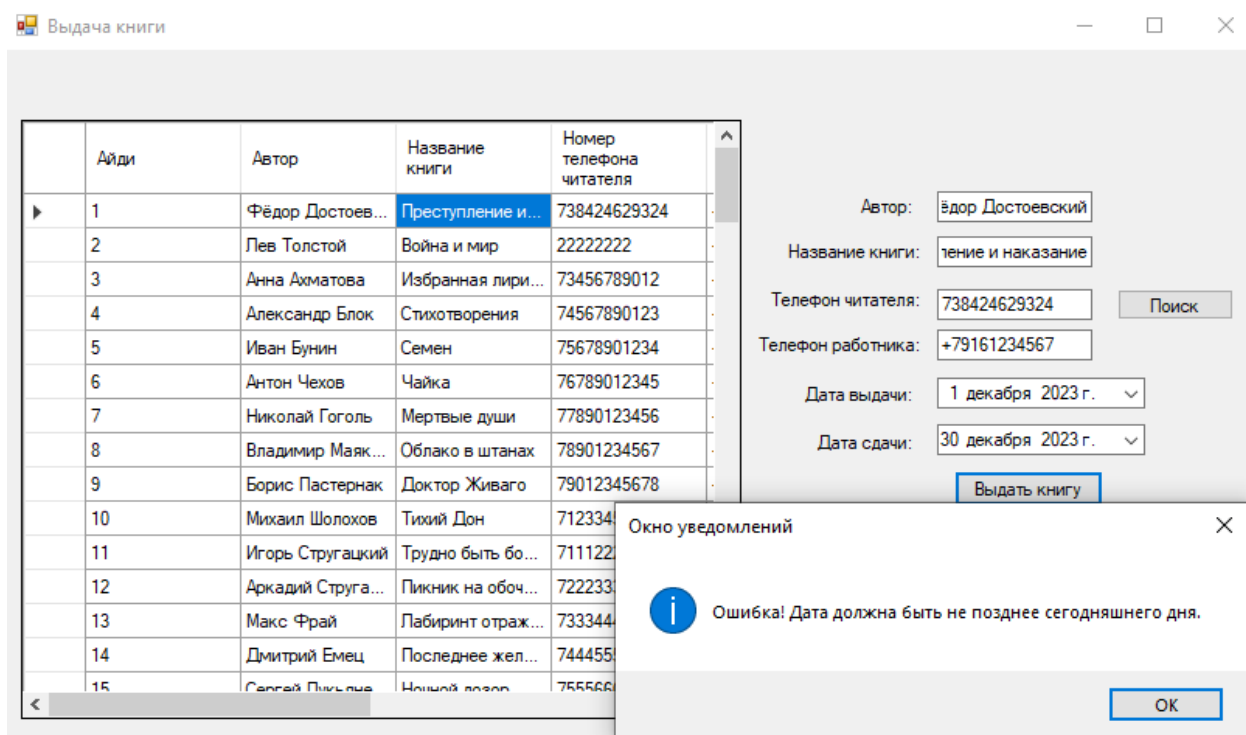


Рис. 2.42 Вывод сообщения о некорректной дате

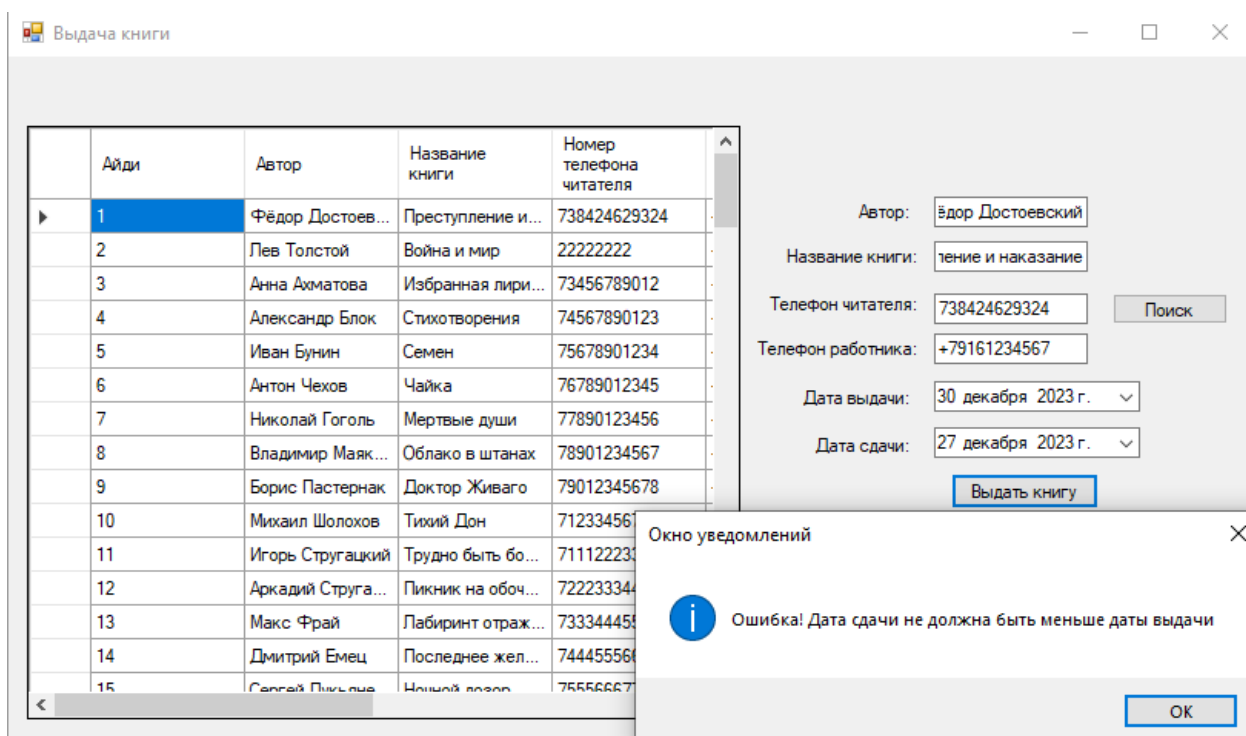


Рис. 2.43 Вывод сообщения о некорректной дате

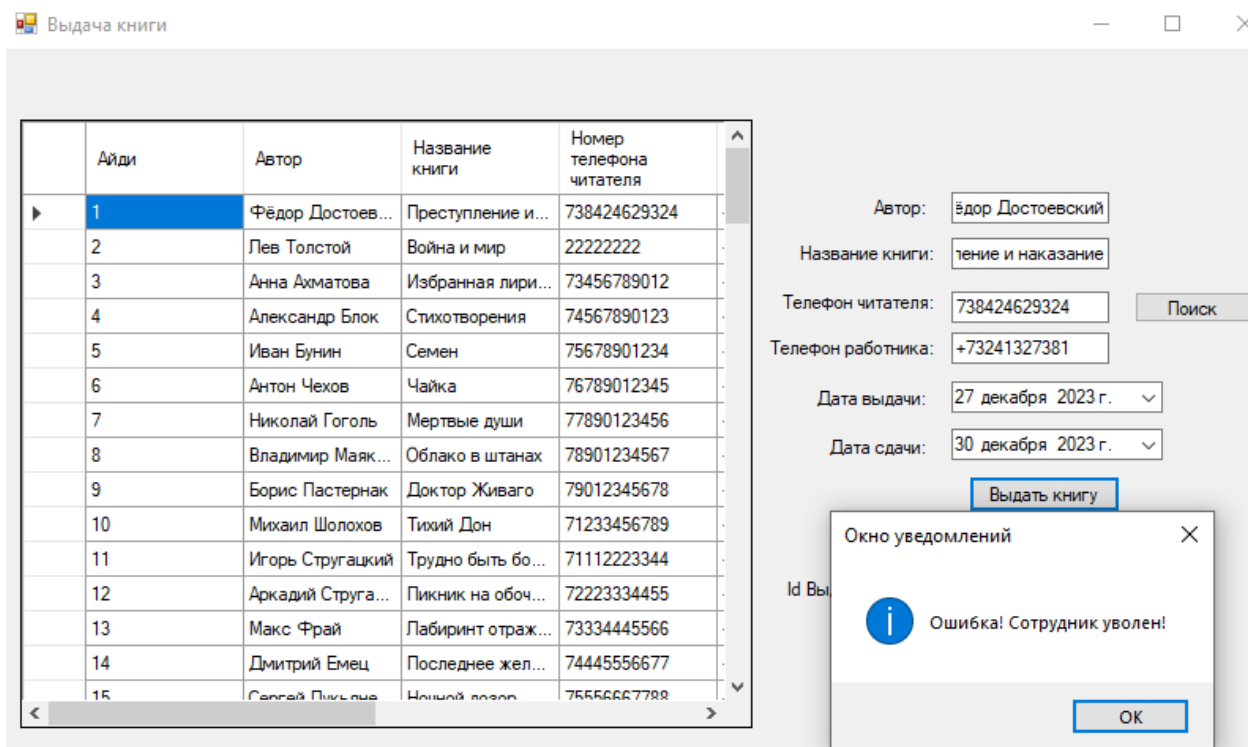


Рис. 2.44 Вывод сообщения о указании номера уволенного сотрудника

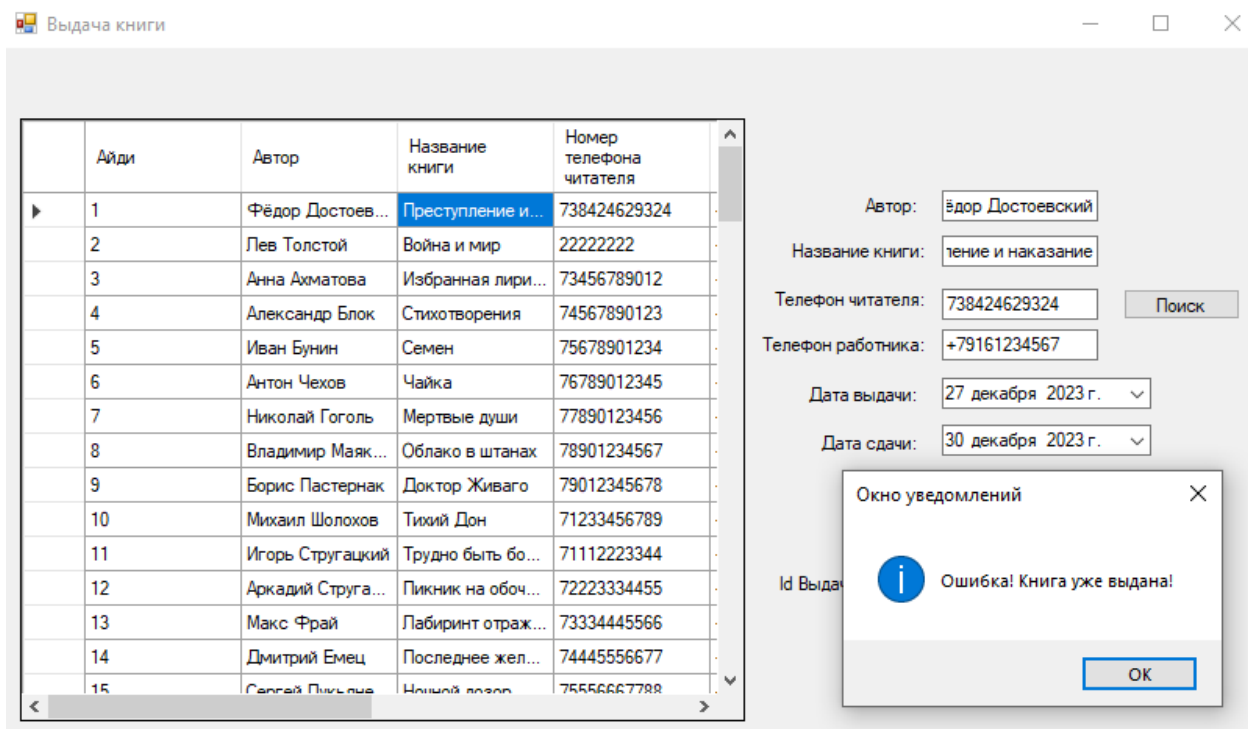


Рис. 2.45 Вывод сообщения о выдаче уже выданной читателю книги

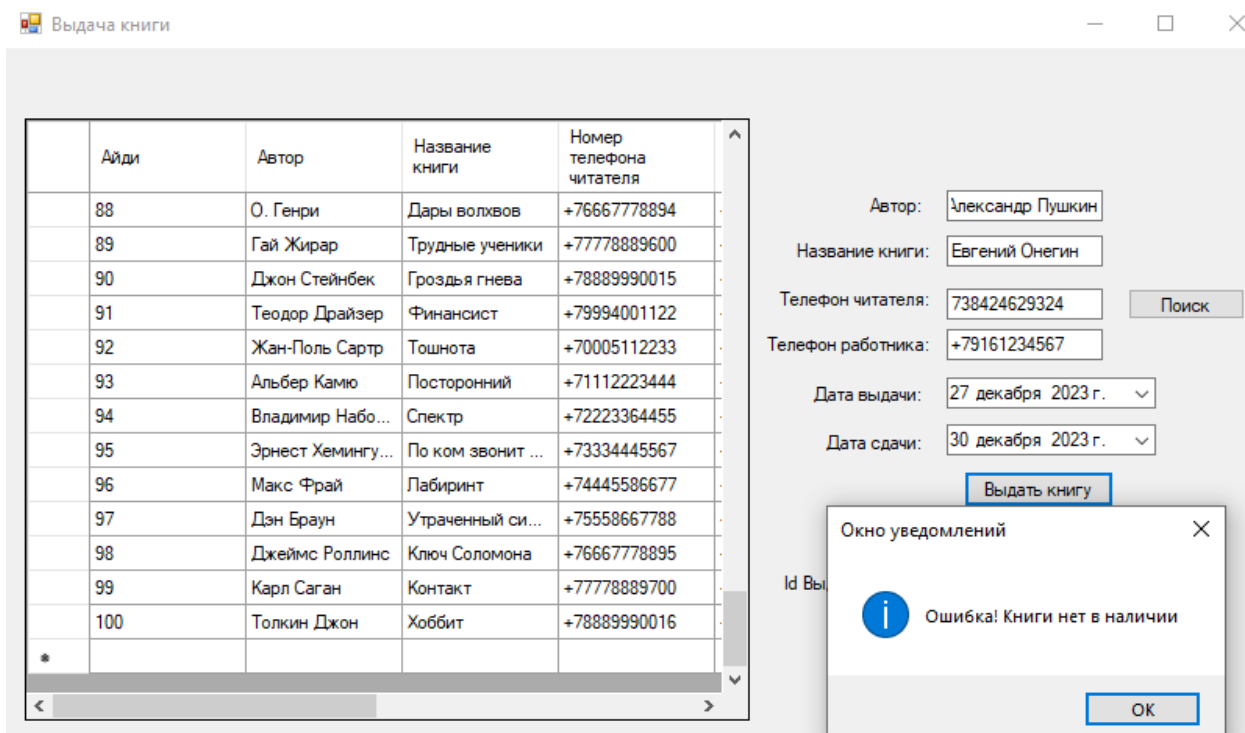


Рис. 2.46 Вывод сообщения о выдаче книги, которой нет в наличии

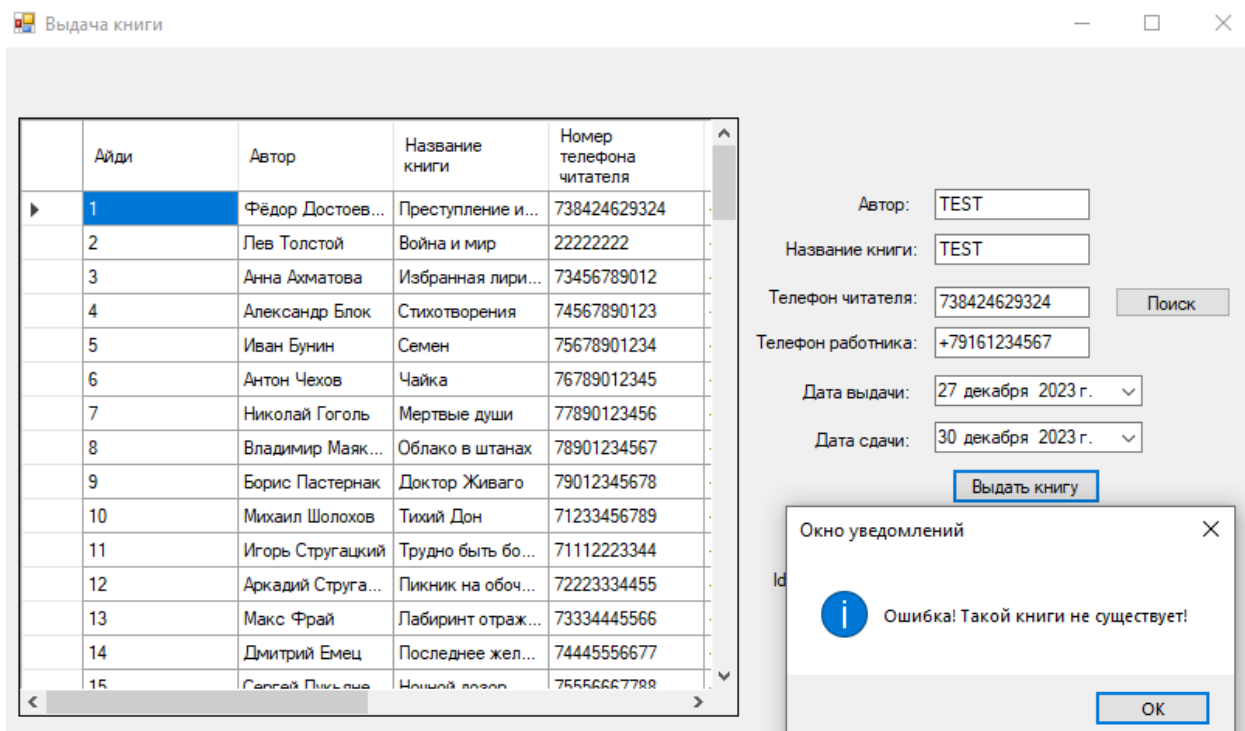


Рис. 2.47 Вывод сообщения о выдаче несуществующей книги

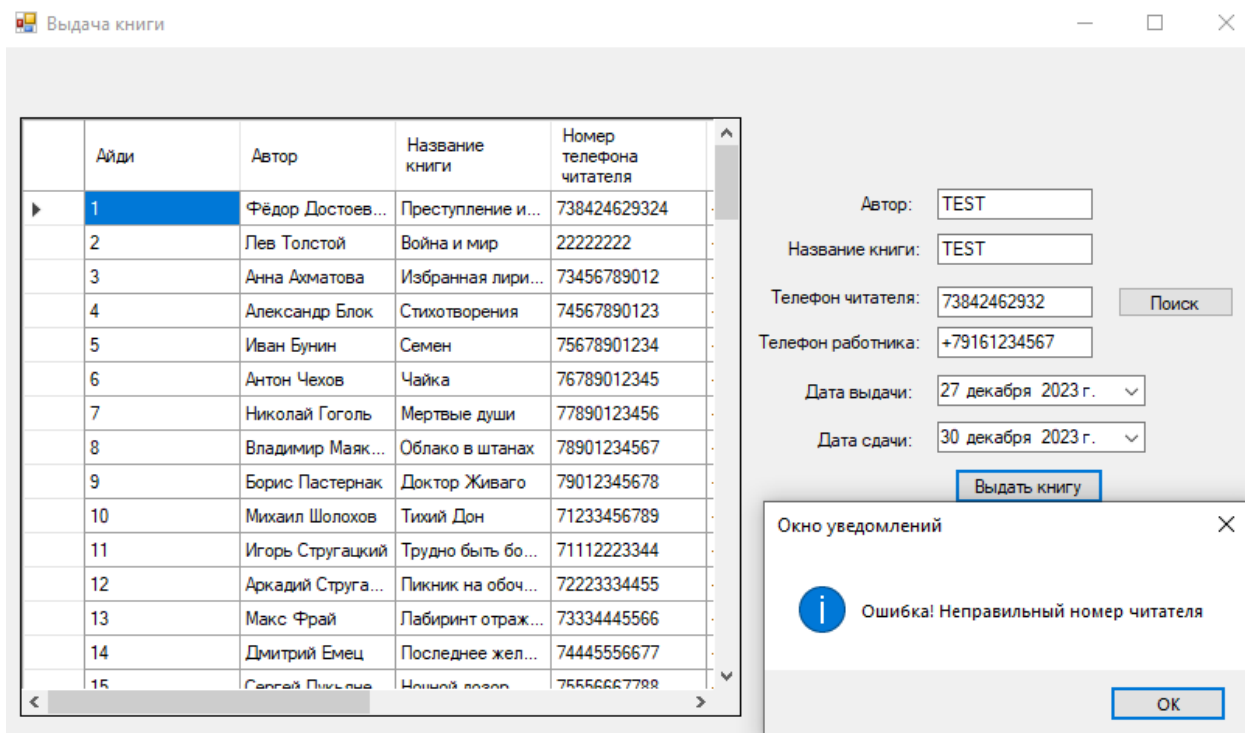


Рис. 2.48 Вывод сообщения о неправильном номере телефона читателя

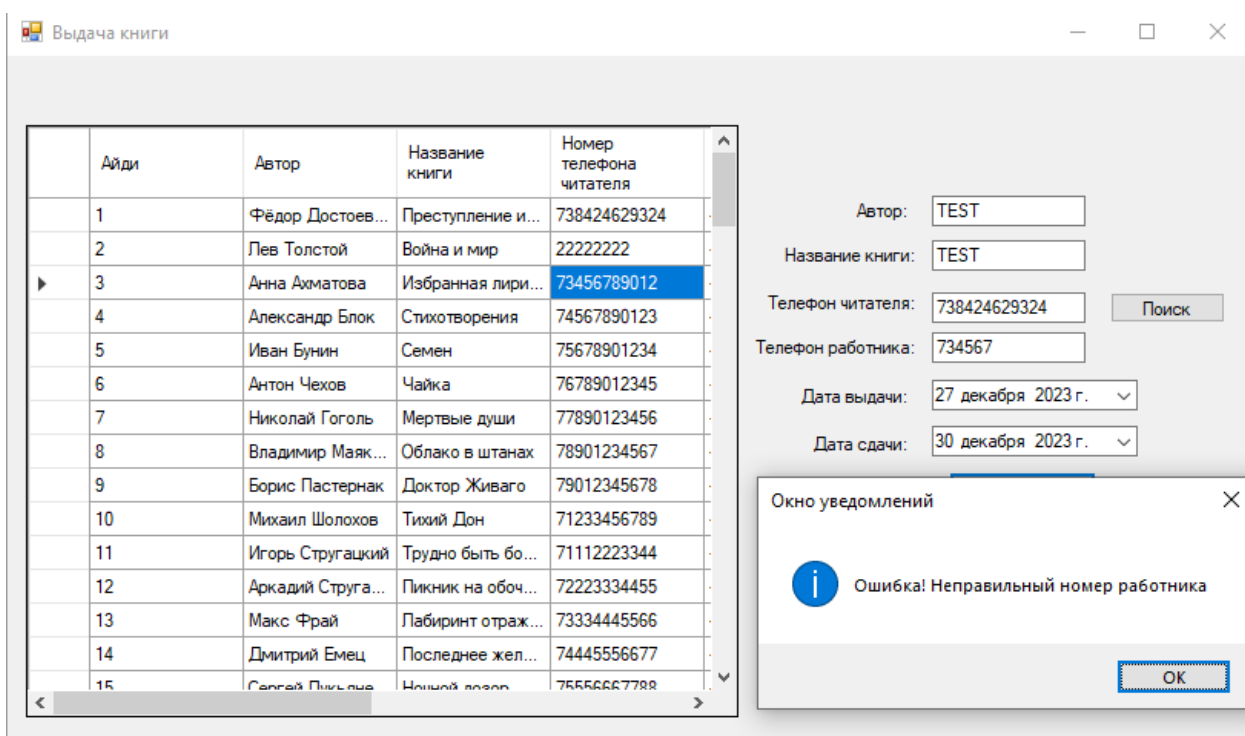


Рис. 2.49 Вывод сообщения о неправильном номере телефона работника

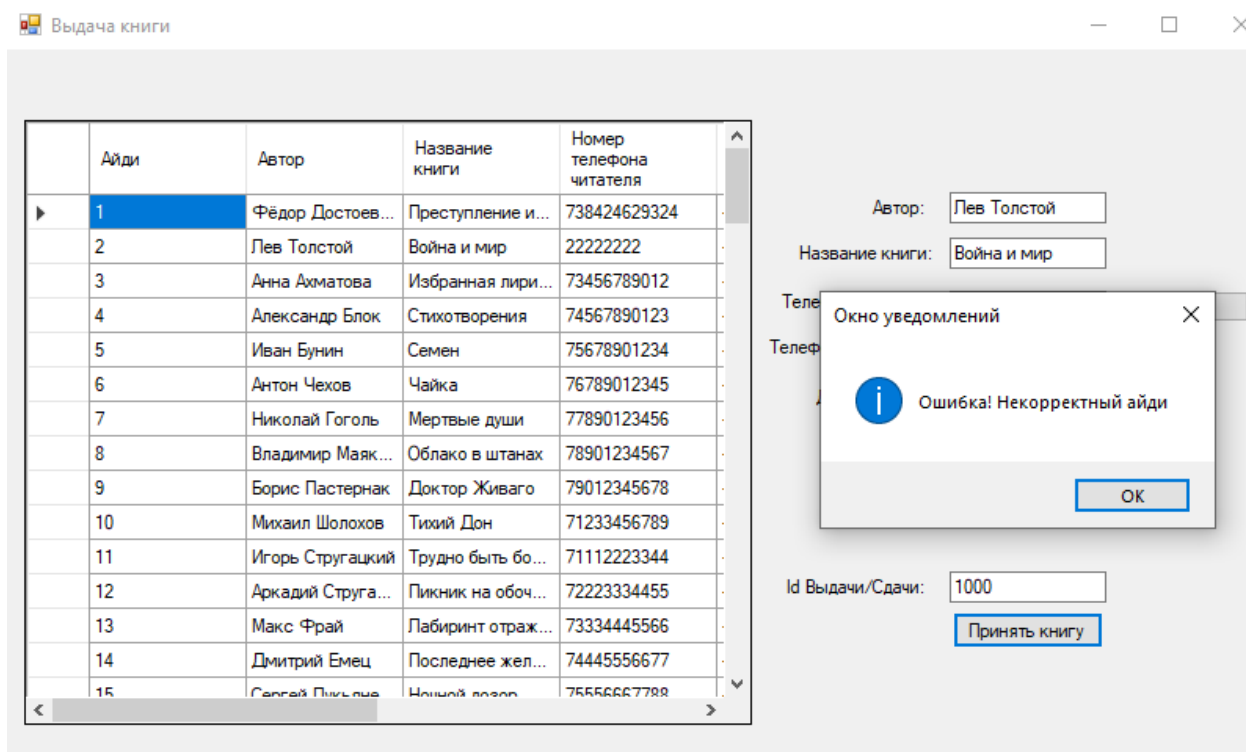


Рис. 2.50 Вывод сообщения о неправильном Айди книги

Введем корректные данные при выдаче книги, появляется сообщение об успешной выдаче книги (рис. 2.51).

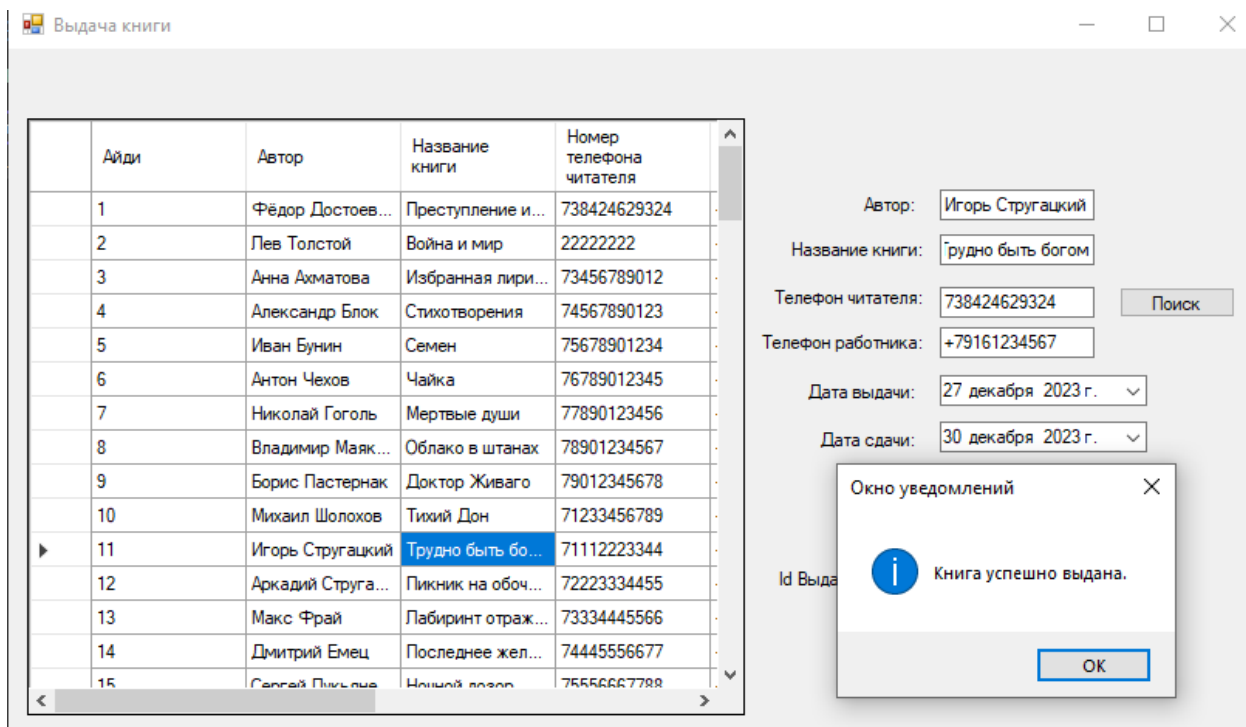


Рис. 2.51 Вывод сообщения о выдаче книги

Введем корректные данные при выдаче книги, появляется сообщение об успешной выдаче книги (рис. 2.52).

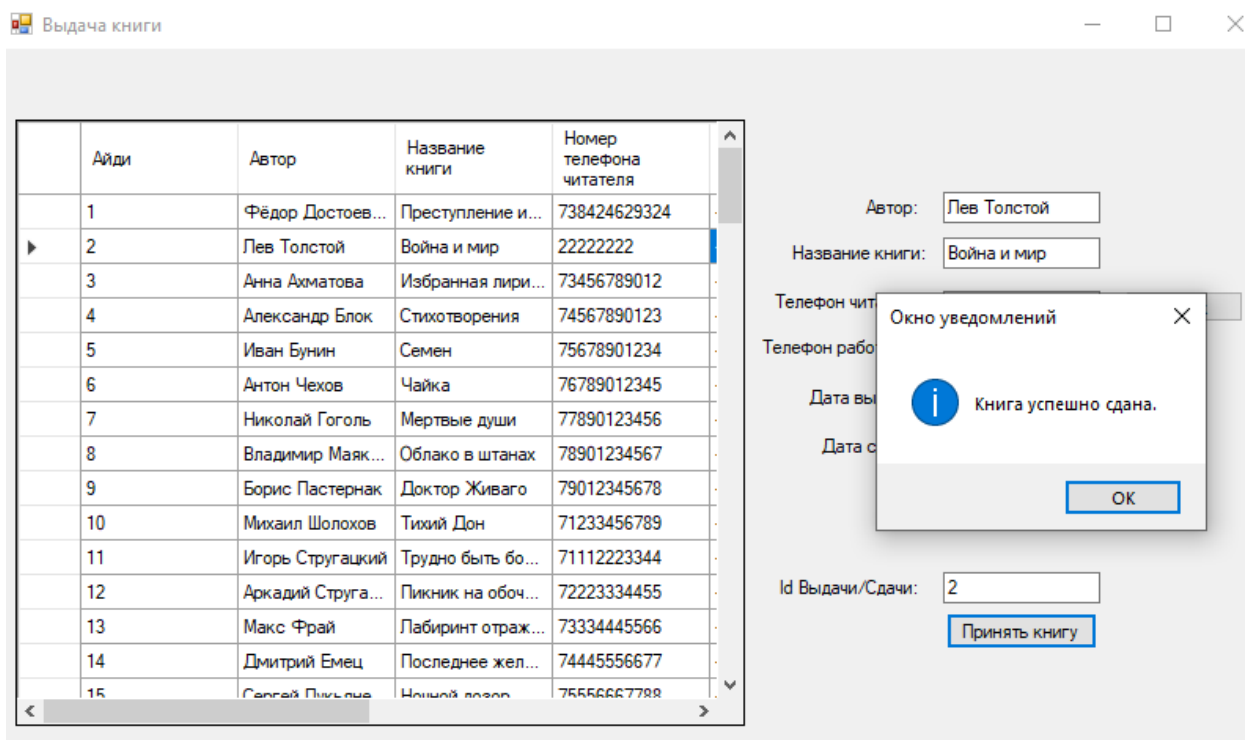


Рис. 2.52 Вывод сообщения о сдаче книги

В правой части формы располагается кнопка поиска информации о сдаче и выдаче книг читателю (рис. 2.53).

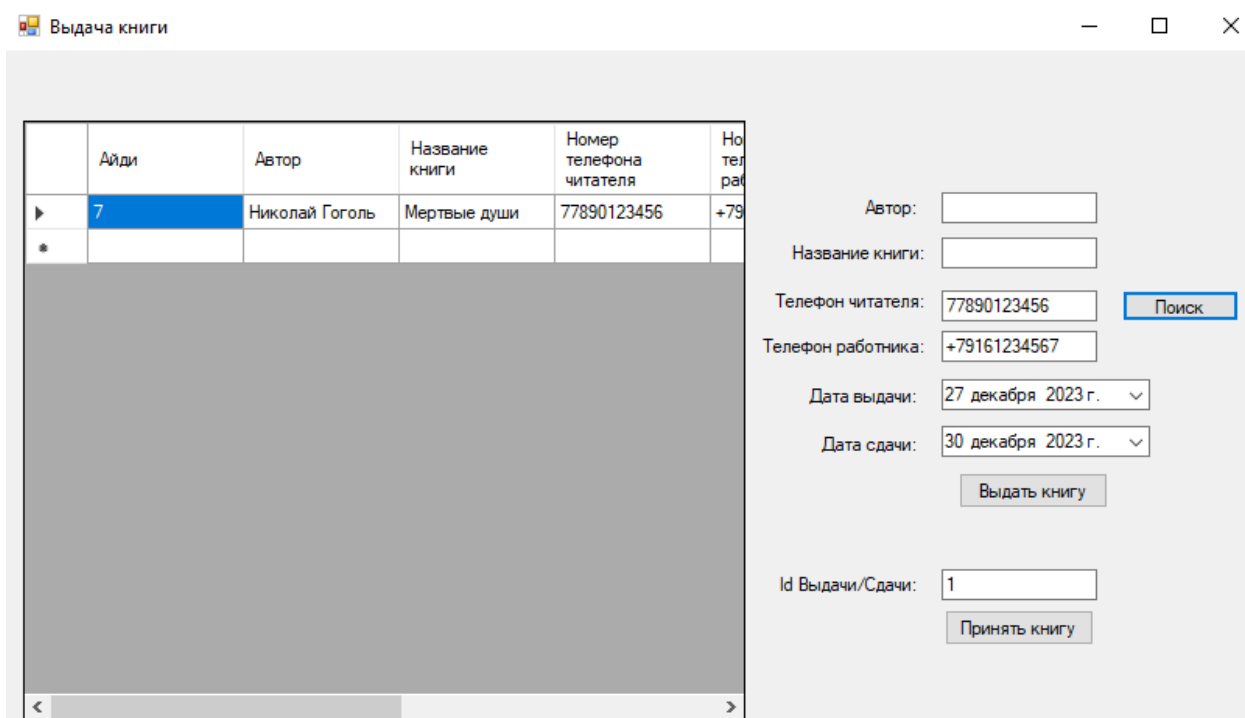


Рис. 2.53 Результат поиска по номеру телефона читателя

Перейдем к функциям библиографа (рис. 2.54). Последнее действие уже было описано у консультанта. Поэтому рассмотрим только принятие



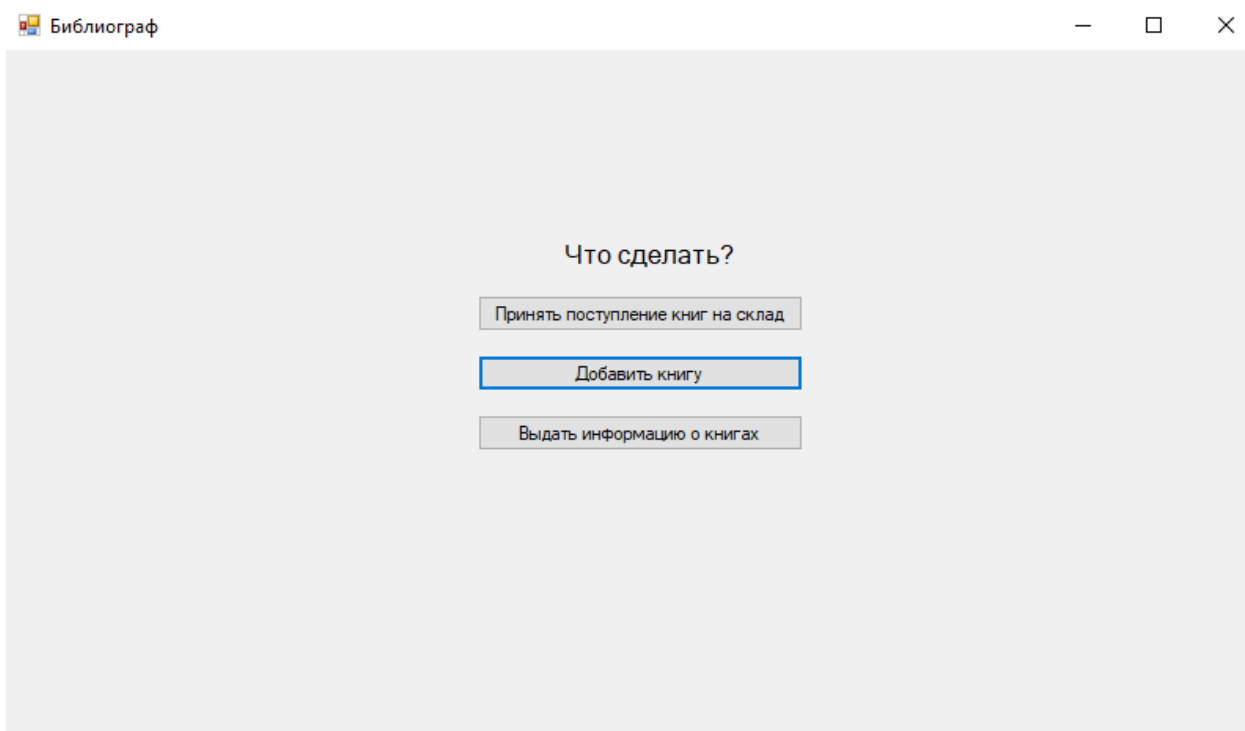


Рис. 2.54 Форма «Библиограф»

Нажмем на кнопку «Принять поступление книг на склад». Откроется форма «Принятие книг на склад» (рис 2.55) На форме находится поиск книг, который уже был описан и кнопка «Добавить запись о пополнении». При нажатии на кнопку срабатывает процедура «AddArrivalAtWarehouse» и триггер «trigAddArrivalAtWarehouse».

Принятие книг на склад

Айди книги	Автор книги	Название книги	Жанр
1	Фёдор Достоев...	Преступление и...	Роман
2	Лев Толстой	Война и мир	Роман
3	Анна Ахматова	Избранная лири...	Поэзия
4	Александр Блок	Стихотворения	Поэзия
5	Иван Бунин	Семен	Рассказы
6	Антон Чехов	Чайка	Драма

Автор:

Название книги:

Найти

Айди книги	Номер телефона	Количество
1	+79161234567	100
2	+79259876543	200
3	+7910987654	100
4	+74955555555	200
5	+74998889900	100
6	+79161234567	200
7	+79259876543	100

Id Книги:

Номер телефона сотрудника:

Дата поступления:  ▾

Количество:

Добавить запись о пополнении

Рис. 2.55 Форма «Принятие книг на склад»

При введении некорректных данных:

1. Неправильной даты (рис. 2.56);
2. Некорректного количества книг (рис. 2.57);
3. Номера телефона уволенного сотрудника (рис. 2.58);
4. Неправильного номера телефона сотрудника (рис. 2.59)
5. Несуществующей книги (рис. 2.60)

Появится окно, сообщающее об одной из данных ошибок.

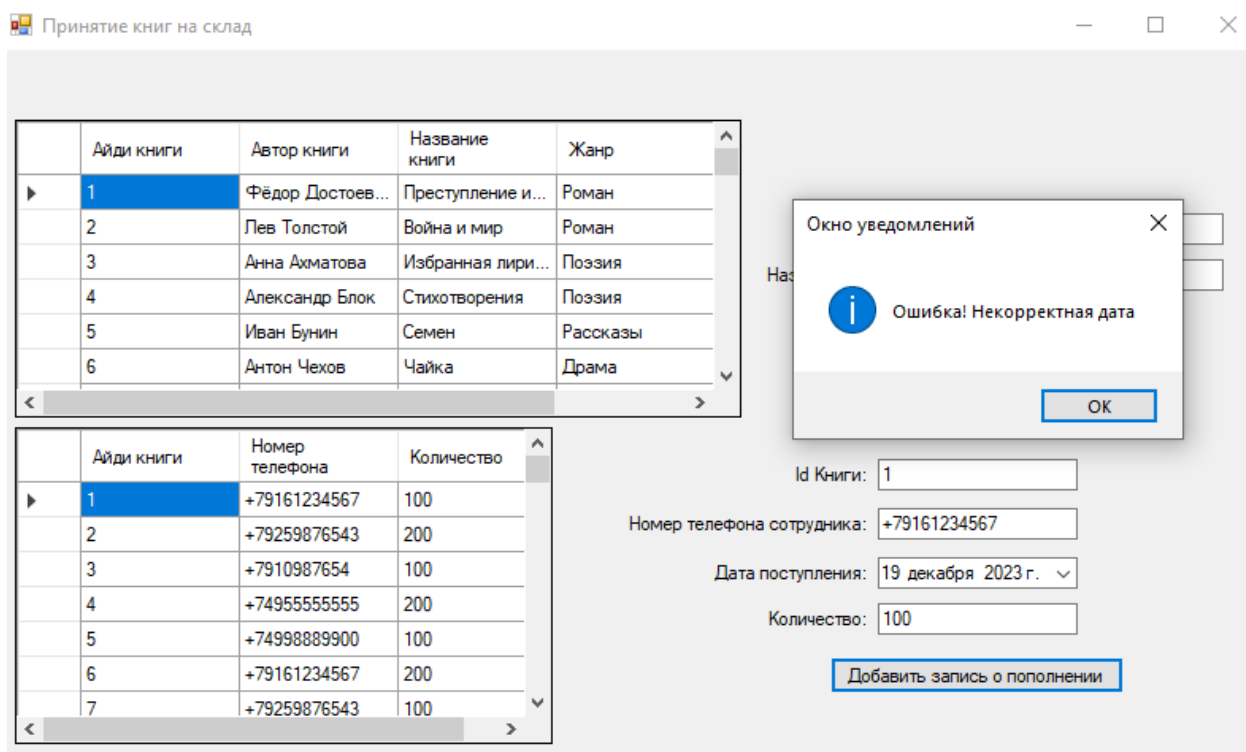


Рис. 2.56 Результат ввода некорректной даты

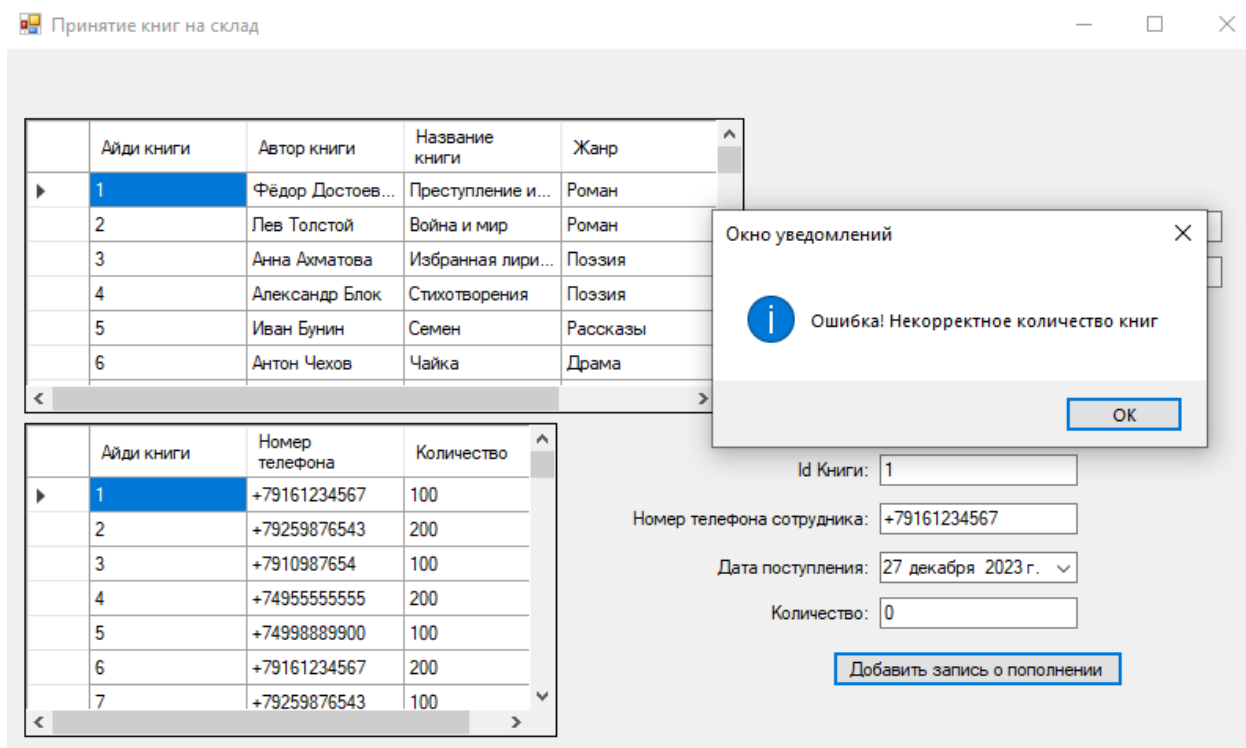


Рис. 2.57 Результат ввода количества книг равного 0

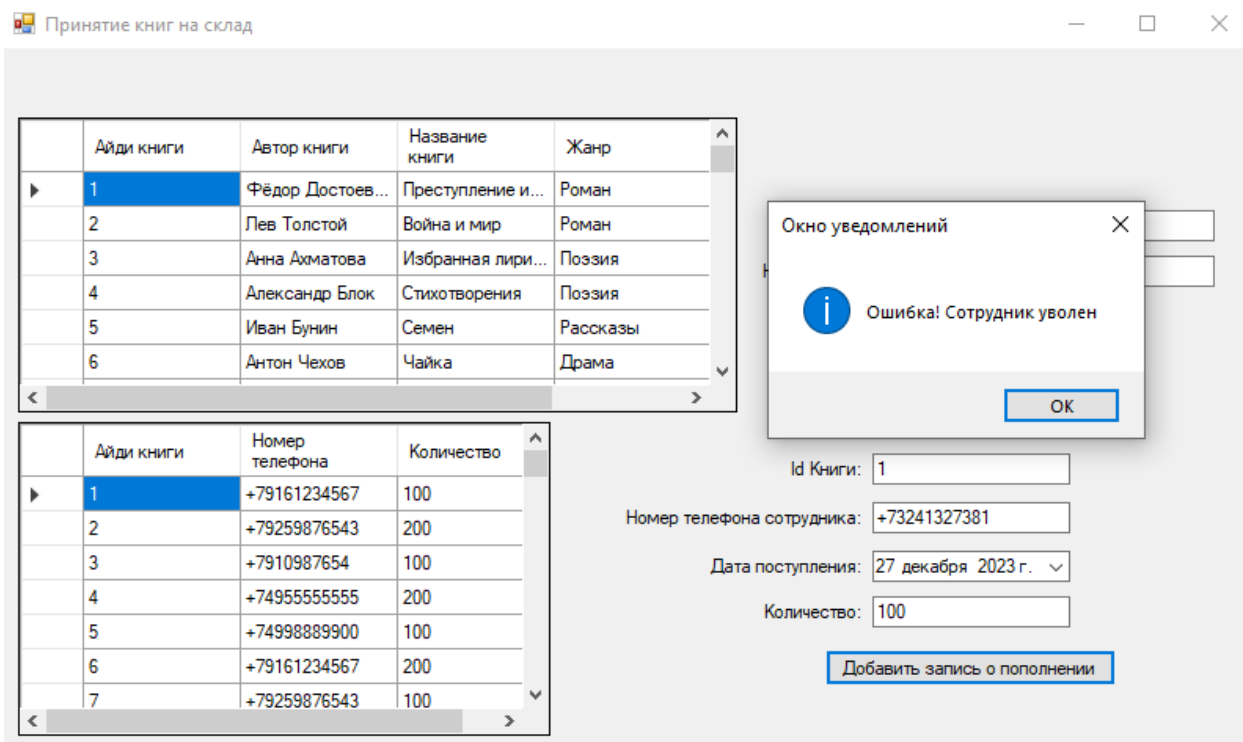


Рис. 2.58 Результат ввода номера телефона уволенного сотрудника

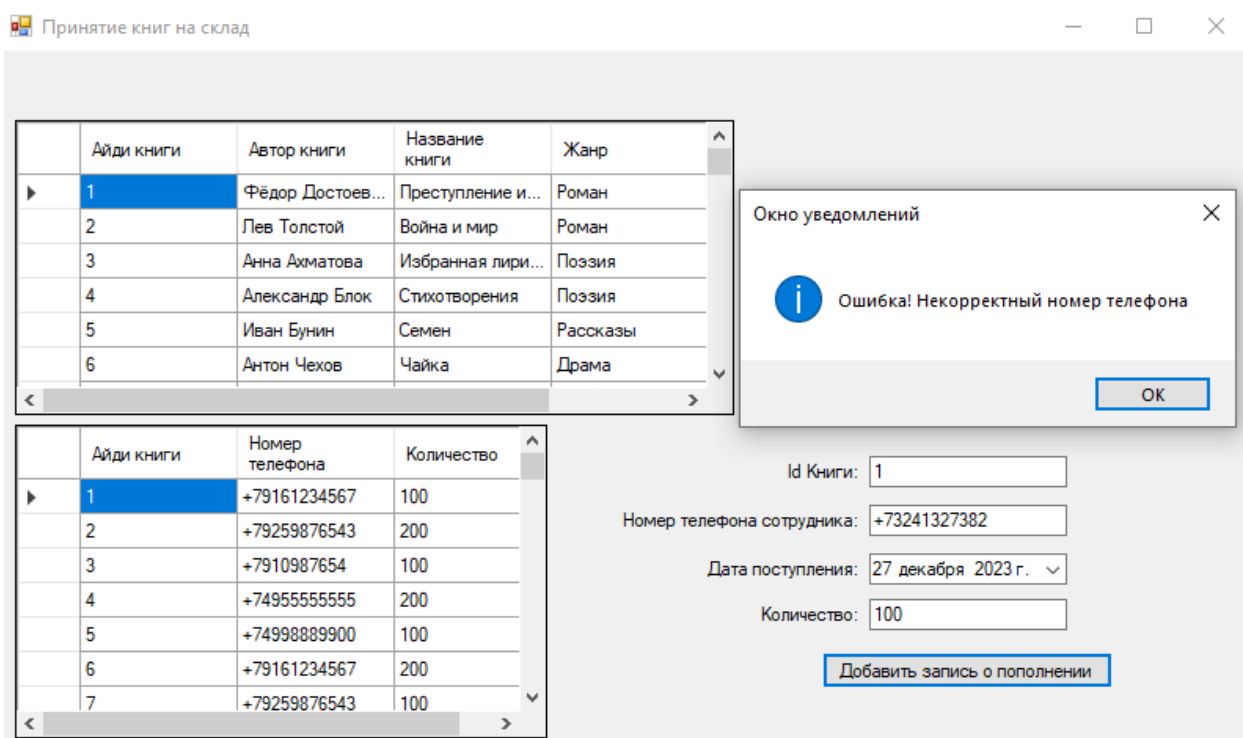


Рис. 2.59 Результат ввода некорректного номера телефона

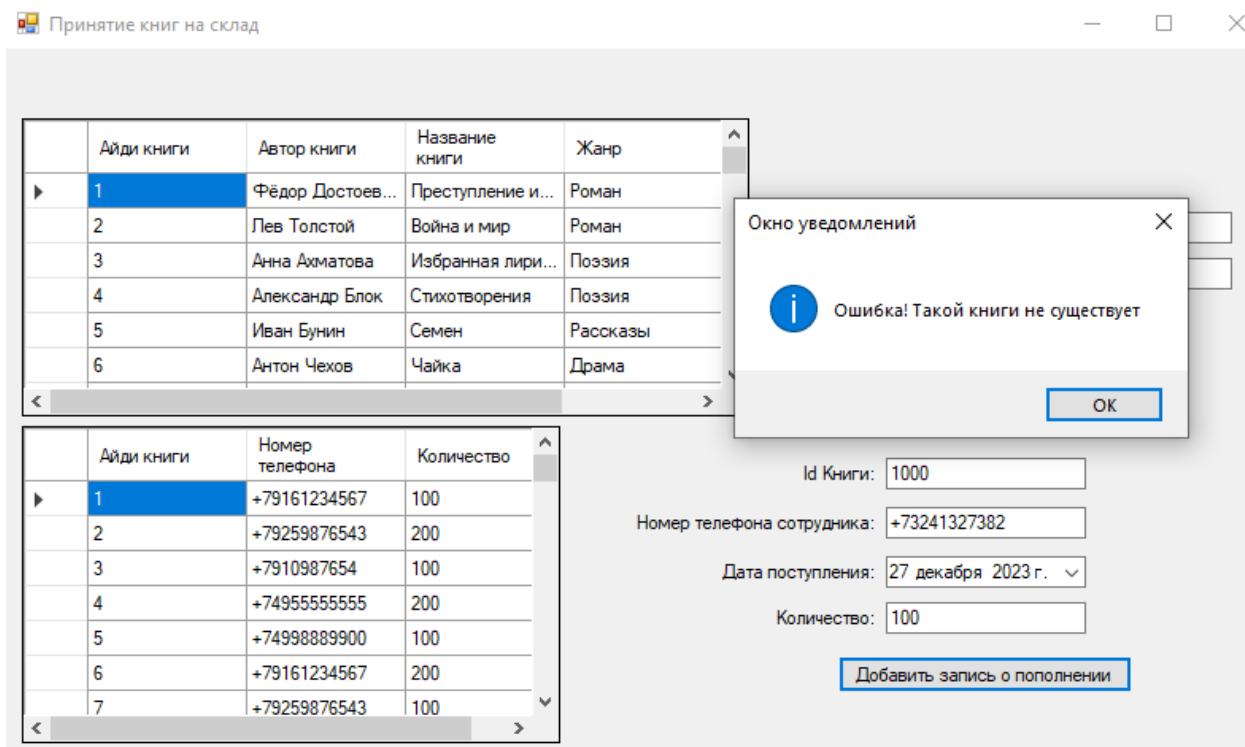


Рис. 2.60 Результат ввода несуществующего Id книги

Введем корректные данные при пополнении книги, появляется сообщение об успешном поступлении книги (рис. 2.61).

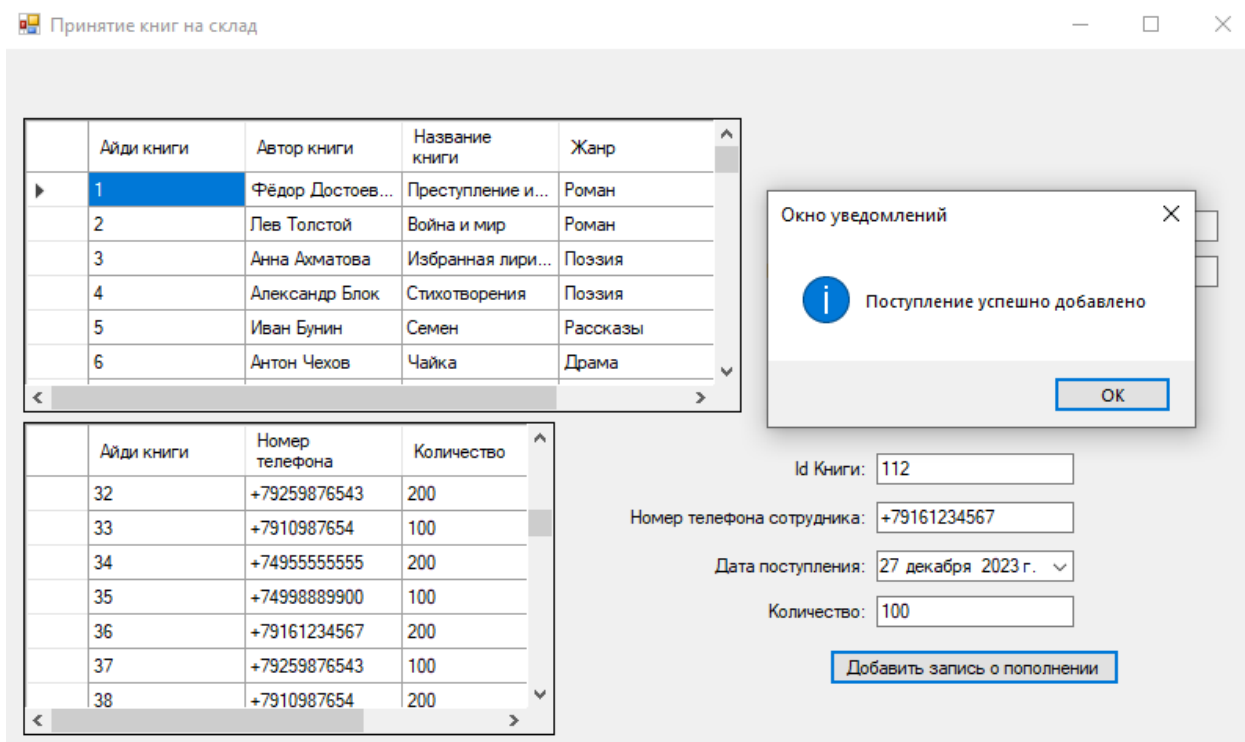


Рис. 2.61 Результат ввода корректных данных

На этом функции консультанта заканчиваются. Перейдем к функциям библиотекаря (рис. 2.62). Последние два действия были описаны в функциях консультанта. Поэтому рассмотрим добавление и увольнение сотрудника.

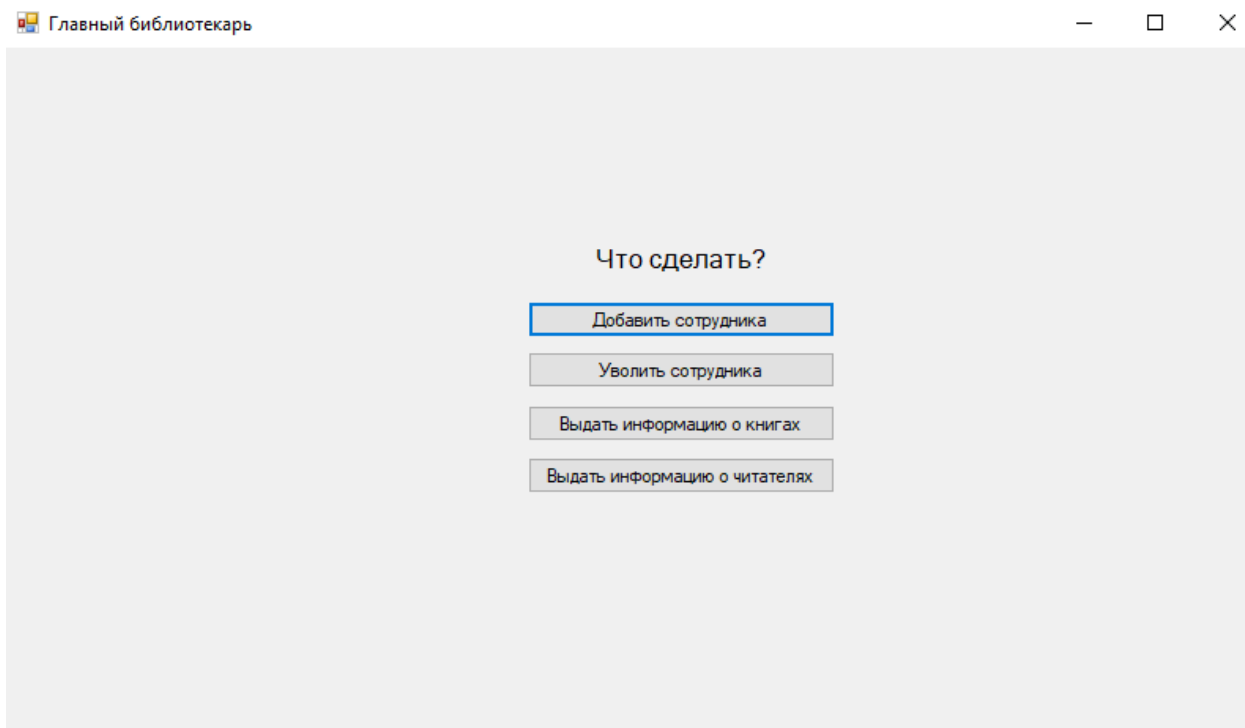


Рис. 2.62 Форма «Главный библиотекарь»

Нажмем на кнопку «Добавить сотрудника». Откроется форма «Добавление сотрудника» (рис 2.63) При нажатии на кнопку «Добавить» срабатывает процедура «AddEmployee».

ФИО	Должность	Номер телефона	Паспорт
Александров Р.В.	Консультант	+79161234567	1234567890
Петров Петр Пе...	Библиотекарь	+79259876543	4567123456
Смирнова Анна ...	Библиотекарь	+7910987654	8910987654
Козлова Мария ...	Библиограф	+74955555555	2345678901
Николаев Андр...	Главный библио...	+74998889900	6789234567
Корчагин Дании...	Уволен	+73241327381	3234277424

ФИО:   
 Должность:   
 Номер телефона:   
 Паспорт:

Рис. 2.63 Форма «Добавление сотрудника»

При введении некорректных данных:

1. Зарегистрированного номера телефона и паспорта (рис. 2.64, 2.65);
2. Несуществующей должности (рис. 2.66);

Появится окно, сообщающее об одной из данных ошибок.

ФИО	Должность	Номер телефона	Паспорт
Александров Р.В.	Консультант	+79161234567	1234567890
Петров Петр Пе...	Библиотекарь	+79259876543	4567123456
Смирнова Анна ...	Библиотекарь	+7910987654	8910987654
Козлова Мария ...	Библиограф	+74955555555	2345678901
Николаев Андр...	Главный библио...	+74998889900	6789234567
Корчагин Дании...	Уволен	+73241327381	3234277424

ФИО:   
 Должность:   
 Номер телефона:

Окно уведомлений

Ошибка! Сотрудник с таким телефоном уже существует!

Рис. 2.64 Результат ввода существующего номера телефона сотрудника

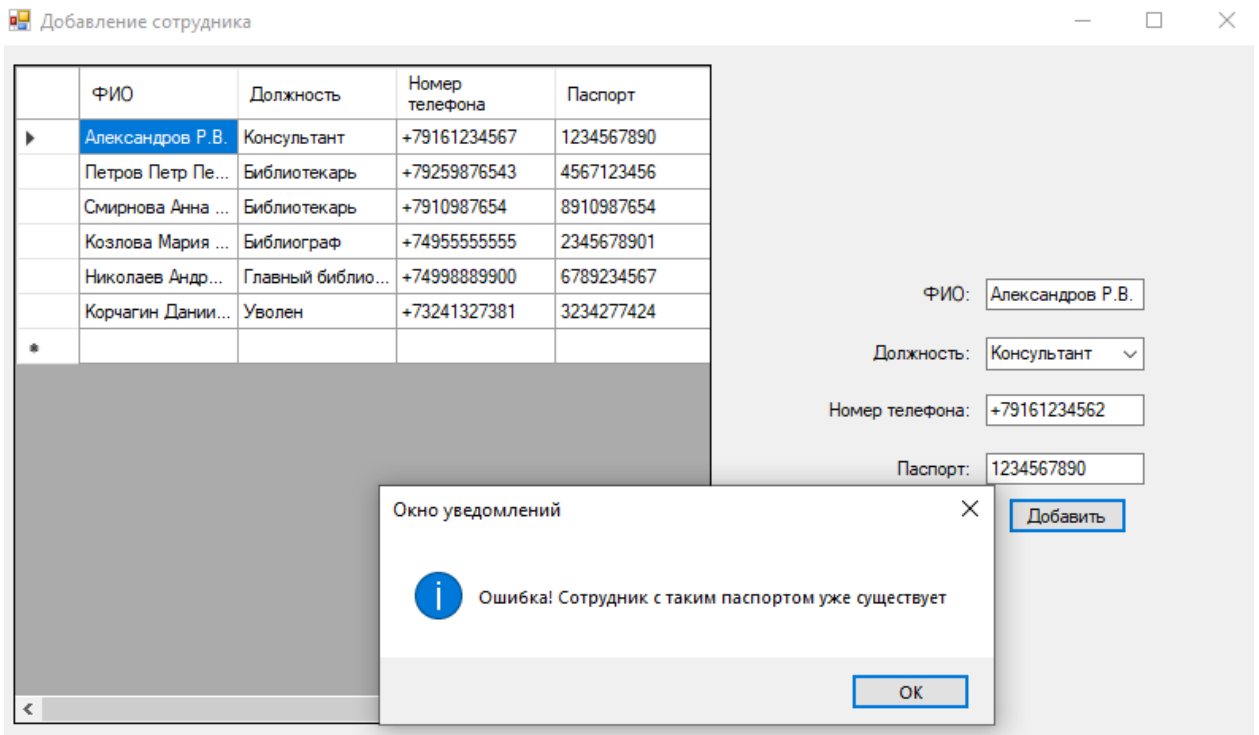


Рис. 2.65 Результат ввода существующего паспорта сотрудника

Введем корректные данные при добавлении сотрудника, появляется сообщение об успешном добавлении сотрудника (рис. 2.66) и вывод добавленного сотрудника в dataGridView (рис 2.67).

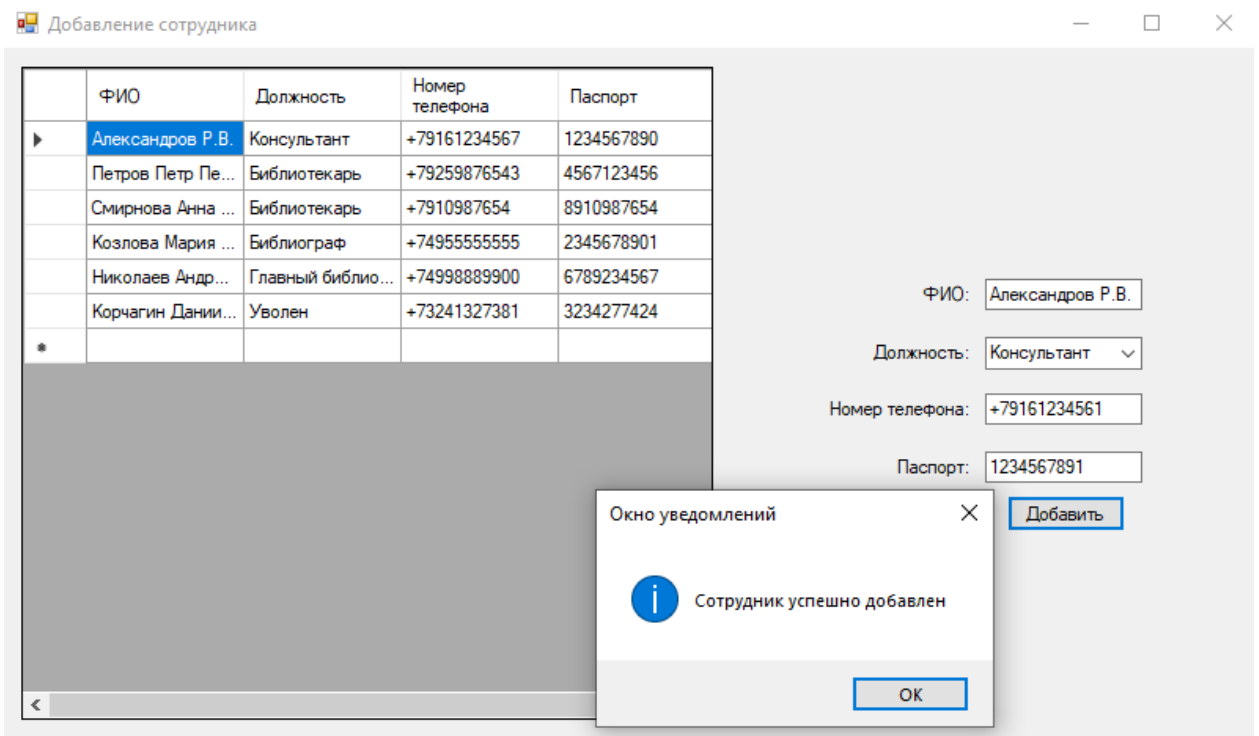


Рис. 2.66 Результат ввода корректных значений сотрудника



Добавление сотрудника

	ФИО	Должность	Номер телефона	Паспорт
▶	Александров Р.В.	Консультант	+79161234567	1234567890
	Петров Петр Пе...	Библиотекарь	+79259876543	4567123456
	Смирнова Анна ...	Библиотекарь	+7910987654	8910987654
	Козлова Мария ...	Библиограф	+74955555555	2345678901
	Николаев Андр...	Главный библио...	+74998889900	6789234567
	Корчагин Дании...	Уволен	+73241327381	3234277424
	Александров Р.В.	Консультант	+79161234561	1234567891
*				

Рис. 2.67 Результат вывода сотрудника в dataGridView

Продолжим рассматривать функции главного библиотекаря. Нажмем на кнопку «Уволить сотрудника». Откроется форма «Увольнение сотрудника» (рис 2.68). При нажатии на кнопку «Уволить» срабатывает процедура «FireEmployee».

Увольнение сотрудника

	ФИО	Должность	Номер телефона	Паспорт
▶	Александров Р.В.	Консультант	+79161234567	1234567890
	Петров Петр Пе...	Библиотекарь	+79259876543	4567123456
	Смирнова Анна ...	Библиотекарь	+7910987654	8910987654
	Козлова Мария ...	Библиограф	+74955555555	2345678901
	Николаев Андр...	Главный библио...	+74998889900	6789234567
	Корчагин Дании...	Уволен	+73241327381	3234277424
	Александров Р.В.	Консультант	+79161234561	1234567891
*				

Паспорт:

Рис. 2.68 Форма «Увольнение сотрудника»

При введении некорректных данных:

1. Несуществующего сотрудника (рис. 2.69);
2. Уже уволенного сотрудника (рис. 2.70);

Появится окно, сообщающее об одной из данных ошибок.

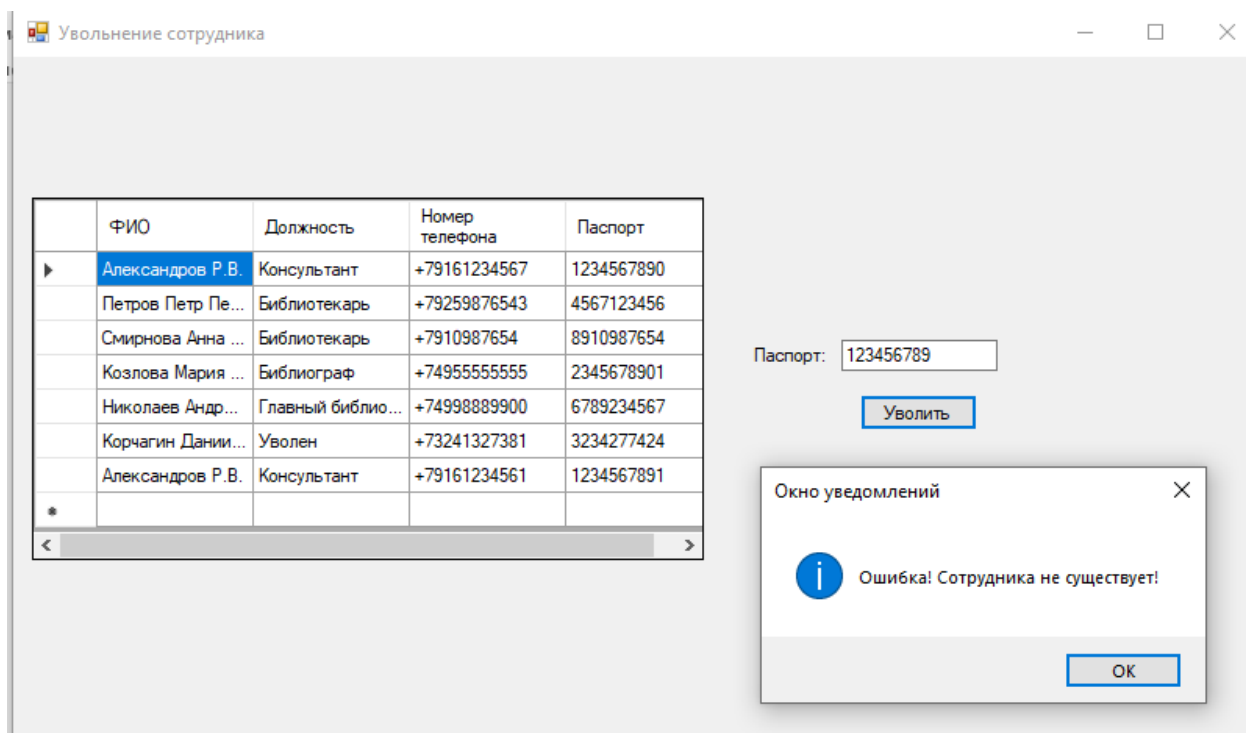


Рис. 2.69 Результат ввода несуществующего паспорта сотрудника

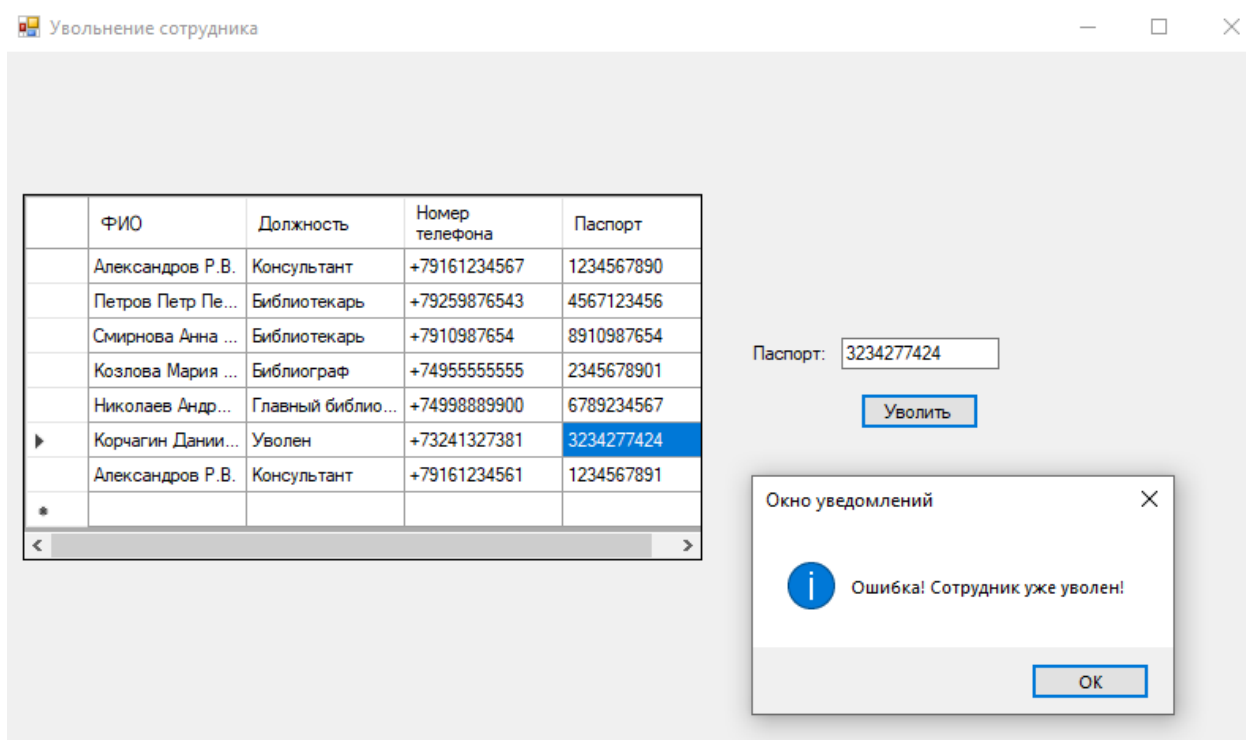


Рис. 2.70 Результат ввода паспорта уволенного сотрудника

Введем корректные данные при увольнении сотрудника, появляется сообщение об увольнении сотрудника (рис. 2.71) и обновление должности сотрудника на «Уволен» в dataGridView (рис 2.72).

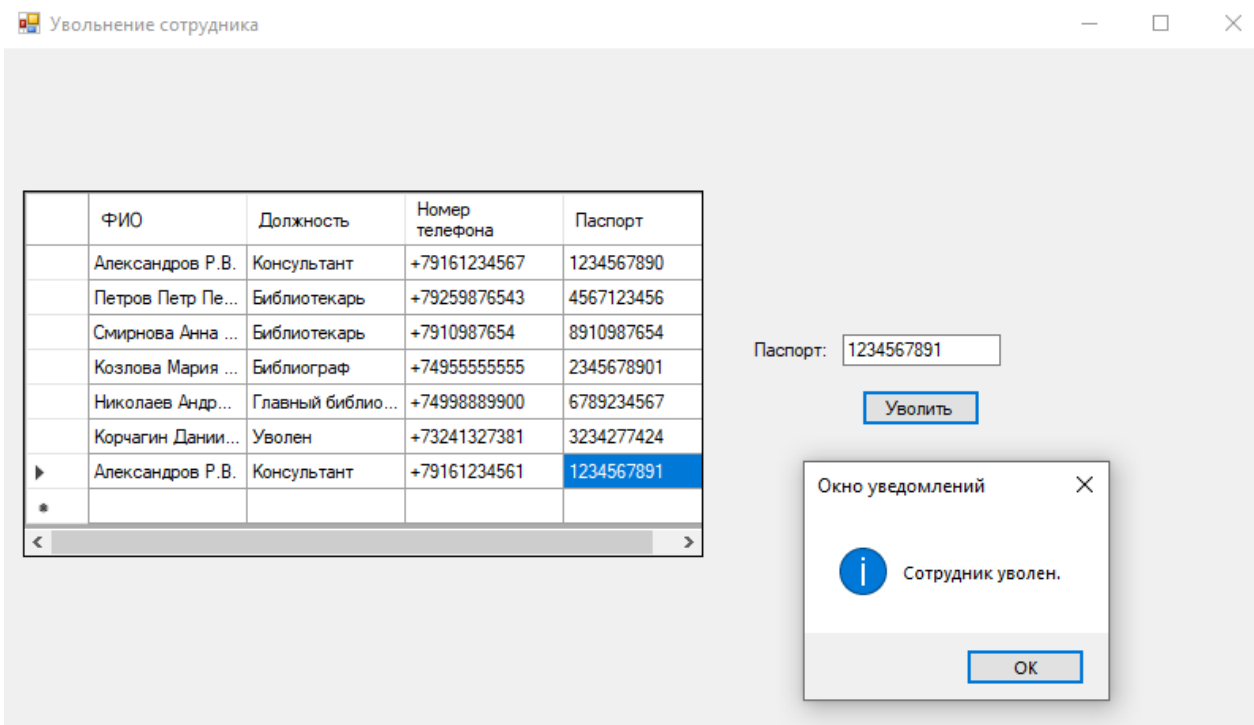


Рис. 2.71 Результат ввода корректного паспорта сотрудника

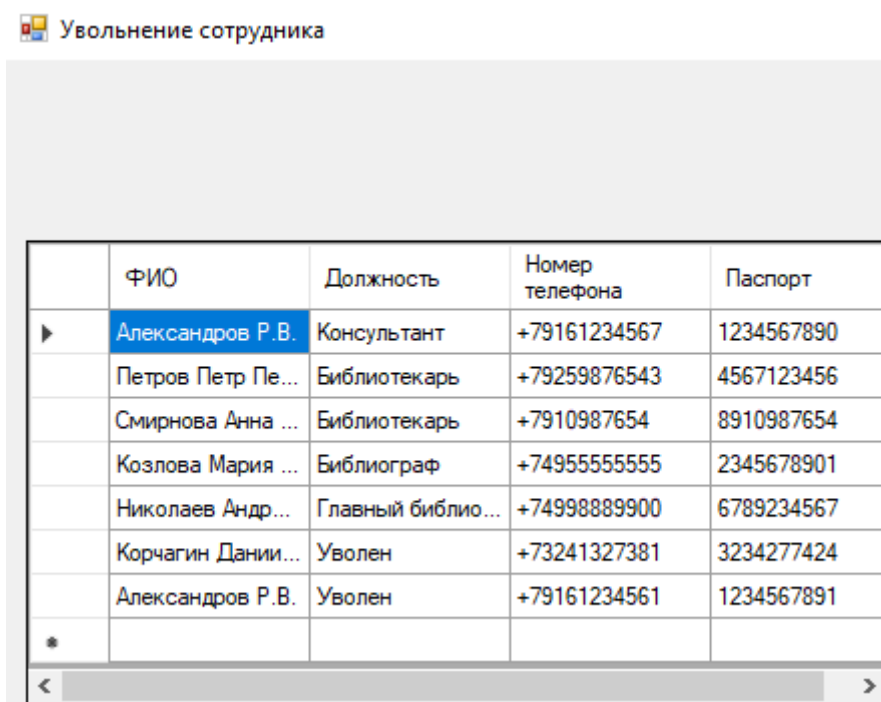


Рис. 2.72 Результат обновления должности на «Уволен» в dataGridView

## ЗАКЛЮЧЕНИЕ

Тема курсового проекта - проектирование базы данных для упрощения работы персонала библиотеки и автоматизации процессов учета книг и ведения истории читателей. Этот проект оказывается востребованным как для небольших библиотек, так и для крупных библиотечных центров, поскольку он не только прост и удобен в использовании, но также существенно сокращает время обслуживания каждого читателя, заменяя ручное ведение списков и карточек на автоматизированный процесс. В ходе выполнения самостоятельной работы были получены следующие результаты:

1. При проведении анализа предметной области был проведен подробный анализ уже существующих программных решений. Первое из них является конфигурацией «1С:Библиотека», а второе – программой «OPAC-Global». Выделены основные функции, аспекты будущей системы.

2. На этапе концептуального проектирования были выделены действующие лица (консультант, библиотекарь, библиограф, главный библиотекарь) и их функции, на основе которых была построена диаграмма вариантов использования, с помощью которой были определены объекты, информацию о которых следует хранить в БД.

3. Основная задача этапа логического проектирования заключалась в построении и описании ER-диаграмм, на основе которых, при помощи правил, были построены отношения, определяющие реляционную схему базы данных библиотеки: Книга, Сотрудник, Читатель, Сдача/Выдача, ПоступлениеНаСклад, РезервированиеКниги.

4. На заключительном этапе физического проектирования были сформулированы требования к структурам таблиц базы данных, реализованных с помощью MS SQL Server Management Studio. Так же была создана БД библиотеки, отражающая связи между таблицами.

5. Разработан пользовательский интерфейс для взаимодействия с базой данных с использованием среды разработки Visual Studio 2022 и

языка программирования C#. Создана база данных, позволяющая выполнять различные функции в зависимости от роли пользователя. Доступные функции включают изменение информации о книгах, взаимодействие с читателями и сотрудниками.

В результате был создан программный продукт, полностью соответствующий поставленным задачам библиотеки.

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. WikipediA [Электронный ресурс] – 2023. – Режим доступа: <https://ru.wikipedia.org/wiki/Библиотека>, свободный. Дата обращения: 1.12.2023 г.
2. 1с [Электронный ресурс] – 2023. – Режим доступа: <https://solutions.1c.ru/catalog/library>, свободный. Дата обращения: 1.12.2023 г.
3. Орас-Global [Электронный ресурс] – 2023 – Режим доступа: <https://орас-global.ru>, свободный. Дата обращения: 3.12.2023 г.
4. Документация по с# [Электронный ресурс] – Microsoft, 2022. – Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/csharp/>. Дата обращения: 15.12.2023 г.
5. Программирование на с# [Электронный ресурс] – 2019. – Режим доступа: <https://metanit.com/>, свободный. Дата обращения: 17.12.2023 г.
6. Документация по Работе с Com сервером Word [Электронный ресурс]  
Microsoft, 2022. – Режим доступа: [http://www.wladm.narod.ru/C\\_Sharp/comword.html](http://www.wladm.narod.ru/C_Sharp/comword.html). Дата обращения: 19.12.2023 г.
7. Документация по SQL [Электронный ресурс] – Microsoft, 2022. – Режим доступа: <https://learn.microsoft.com/ru-ru/sql/?view=sql-server-ver16>. Дата обращения: 21.12.2023 г.

## ПРИЛОЖЕНИЕ А

### Заполнение таблиц данными

Id_Book	Author	Name	Genre	ISBN	UDK	Number
13	Макс Фрай	Лабиринт отра...	Фэнтези	978-5171087429	813.52	100
14	Дмитрий Емец	Последнее же...	Фэнтези	978-5171185019	813.52	200
15	Сергей Лукьян...	Ночной дозор	Ужасы	978-5170866945	813.54	100
16	Татьяна Толстая	Кысь	Рассказы	978-5170415795	821.161.1	200
17	Иван Бунин	Темные аллеи	Рассказы	978-5170866648	821.161.1	100
18	Федор Достоев...	Идиот	Роман	978-5170193001	821.161.2	200
19	Лев Толстой	Анна Каренина	Роман	978-5171087528	821.161.2	100
20	Антон Чехов	Вишневый сад	Драма	978-5170239811	821.161.2	200
21	Александр Со...	Раковый корпус	Роман	978-5170384121	891.7	100
23	Лев Толстой	Война и мир	Роман	978-5171036861	821.161.2	100
24	Антуан де Сент...	Земля людей	Философия	978-5170708336	111.85	200
25	Игорь Стругац...	Хищные вещи ...	Научная фанта...	978-5171087429	813.54	100
26	Аркадий Струг...	Малыш	Научная фанта...	978-5170426098	813.54	200
27	Габриэль Гарс...	Осенний патру...	Магический р...	978-5170921001	861.14	100
28	Эрих Мария Ре...	Триумфальная...	Роман	978-5170168211	833.912	200
29	Джордж Оруэлл	Скотный двор	Антиутопия	978-5171070032	823.912	100
30	Джоан Роулинг	Гарри Поттер ...	Фэнтези	978-1408812826	823.912	200
31	Исаак Азимов	Я, робот	Научная фанта...	978-5170413708	813.54	100
32	Михаил Булга...	Собачье сердце	Роман	978-5170127987	891.7	200
33	Владимир Наб...	Лолита	Роман	978-5170829403	891.7	100
34	Борис Пастерн...	Доктор Живаго	Роман	978-5170183298	821.161.1	200
35	Герберт Уэллс	Война миров	Научная фанта...	978-5170218953	813.54	100
36	Жюль Верн	Двадцать тыся...	Научная фанта...	978-5170863217	813.54	200
37	Александр Дю...	Граф Монте-К...	Роман	978-5170106043	843.5	100
38	Марк Твен	Приключения ...	Роман	978-5170709203	813.4	200
39	Джейн Остин	Гордость и пре...	Роман	978-5170050582	823.7	100
40	Агата Кристи	Десять негритят	Детектив	978-5170242439	823.912	200
41	Исаак Азимов	Надежда на Зе...	Научная фанта...	978-5170427774	813.54	100
42	Фёдор Достоев...	Бесы	Роман	978-5170261192	821.161.2	200
43	Лев Толстой	Кавказский пл...	Повесть	978-5170629924	821.161.2	100
44	Антуан де Сент...	Война в небе	Военный роман	978-5170711329	111.85	200
45	Игорь Стругац...	Операция "Кр...	Научная фанта...	978-5171200286	813.54	100
46	Аркадий Струг...	Трудно быть б...	Научная фанта...	978-5170093176	813.54	200
47	Габриэль Гарс...	Столкновение ...	Магический р...	978-5170960581	861.14	100
48	Эрих Мария Ре...	Черный обели...	Роман	978-5170235370	833.912	200
49	Джордж Оруэлл	Ферма животн...	Аллегория	978-5170107743	823.912	100
50	Джоан Роулинг	Гарри Поттер ...	Фэнтези	978-5353003650	823.912	200
51	Исаак Азимов	Космический ...	Научная фанта...	978-5170573762	813.54	100
52	Михаил Булга...	Мастер и Марг...	Роман	978-5170826419	891.7	200

Рис. А.1 Данные таблицы «Книга»

Id_Reader	FIO	Gender	PhoneNumber	Birthday
1	Иванов Иван ...	м	738424629324	2003-01-25
2	Петрова Елена...	ж	22222222	1992-02-02
3	Смирнов Алек...	м	73456789012	1994-03-03
4	Козлова Мари...	ж	74567890123	1996-04-04
5	Николаев Иго...	м	75678901234	1998-05-05
6	Егорова Ольга ...	ж	76789012345	2000-06-06
7	Федоров Дени...	м	77890123456	2002-07-07
8	Сергеева Анас...	ж	78901234567	2004-08-08
9	Кузнецов Паве...	м	79012345678	2006-09-09
10	Александрова ...	ж	71233456789	2008-10-10
11	Александров А...	м	71112223344	1990-11-11
12	Иванова Анна ...	ж	72223334455	1992-12-12
13	Сергеев Алекс...	м	73334445566	1994-01-01
14	Кузнецова Тать...	ж	74445556677	1996-02-02
15	Петров Артем ...	м	75556667788	1998-03-03
16	Николаева Вик...	ж	76667778899	2000-04-04
17	Егоров Павел ...	м	77778889900	2002-05-05
18	Федорова Екат...	ж	88889990011	2004-06-06
19	Смирнов Дени...	м	99990001122	2006-07-07
20	Козлова Марга...	ж	00011122333	2008-08-08
21	Александрова ...	ж	11122233444	1990-09-21
22	Иванов Антон ...	м	22233344555	1992-08-22
23	Смирнова Але...	ж	33344455566	1994-07-23
24	Кузнецов Влад...	м	44455556677	1996-06-24
25	Петрова Оксан...	ж	55566667788	1998-05-25
26	Николаев Дми...	м	66677778899	2000-04-26
27	Егорова Натал...	ж	77788889900	2002-03-27
28	Федоров Викто...	м	+78889990013	2004-02-28
29	Сергеева Елен...	ж	+79990001112	2006-01-29
30	Козлов Станис...	м	+70001162233	2008-12-30
31	Александрова ...	ж	+71112223334	1990-09-30
32	Иванов Артем ...	м	+72223334435	1992-08-31
33	Смирнов Арте...	м	+73334442566	1994-07-01
34	Кузнецов Вадим...	м	+74445556277	1996-06-02
35	Петрова Анаст...	ж	+75556267788	1998-05-03
36	Николаев Викт...	м	+76667778399	2000-04-04
37	Егоров Алексе...	м	+77778889920	2002-03-05
38	Федорова Мар...	ж	+78289990011	2004-02-06
39	Сергеев Влади...	м	+79990001132	2006-01-07

Рис. А.2 Данные таблицы «Читатель»



Id_Employee	FIO	Position	PhoneNumber	PassportNum...
1	Александров Р...	Консультант	+79161234567	1234567890
2	Петров Петр П...	Библиотекарь	+79259876543	4567123456
3	Смирнова Анн...	Библиотекарь	+7910987654	8910987654
4	Козлова Мари...	Библиограф	+74955555555	2345678901
5	Николаев Анд...	Главный библ...	+74998889900	6789234567
6	Корчагин Дан...	Уволен	+73241327381	3234277424
7	Александров Р...	Уволен	+79161234561	1234567891

Рис. А.3 Данные таблицы «Сотрудник»

Id_IssuanceDel...	Id_Book	Id_Reader	Id_Employee	DateOfIssue	DateOfDelivery	Status
1	1	1	1	2023-12-27	2023-12-30	True
2	2	2	2	2023-12-28	2023-12-30	True
3	3	3	3	2023-12-29	2023-12-30	False
4	4	4	4	2023-12-21	2023-12-27	False
5	5	5	5	2023-12-22	2023-12-28	False
6	6	6	1	2023-12-23	2023-12-29	False
7	7	7	2	2023-12-27	2023-12-27	False
8	8	8	3	2023-12-28	2023-12-28	False
9	9	9	4	2023-12-29	2023-12-29	False
10	10	10	5	2023-12-21	2023-12-27	False
11	11	11	1	2023-12-22	2023-12-28	False
12	12	12	2	2023-12-23	2023-12-29	False
13	13	13	3	2023-12-27	2023-12-27	False
14	14	14	4	2023-12-28	2023-12-28	False
15	15	15	5	2023-12-29	2023-12-29	False
16	16	16	1	2023-12-21	2023-12-27	False
17	17	17	2	2023-12-22	2023-12-28	False
18	18	18	3	2023-12-23	2023-12-29	False
19	19	19	4	2023-12-27	2023-12-27	False
20	20	20	5	2023-12-28	2023-12-28	False
21	21	21	1	2023-12-29	2023-12-29	False
23	23	23	3	2023-12-28	2023-12-28	False
24	24	24	4	2023-12-29	2023-12-29	False
25	25	25	5	2023-12-21	2023-12-27	False
26	26	26	1	2023-12-22	2023-12-28	False
27	27	27	2	2023-12-23	2023-12-29	False
28	28	28	3	2023-12-27	2023-12-27	False
29	29	29	4	2023-12-28	2023-12-28	False
30	30	30	5	2023-12-29	2023-12-29	False
31	31	31	1	2023-12-21	2023-12-27	False
32	32	32	2	2023-12-22	2023-12-28	False
33	33	33	3	2023-12-23	2023-12-29	False
34	34	34	4	2023-12-27	2023-12-27	False
35	35	35	5	2023-12-28	2023-12-28	False
36	36	36	1	2023-12-29	2023-12-29	False
37	37	37	2	2023-12-21	2023-12-27	False
38	38	38	3	2023-12-22	2023-12-28	False
39	39	39	4	2023-12-23	2023-12-29	False
40	40	40	5	2023-12-27	2023-12-27	False

Рис. А.4 Данные таблицы «Выдача/Сдача»

Id_Arrival	Id_Book	Id_Employee	DateOfArrival	Number
1	1	1	2023-12-19	100
2	2	2	2023-12-19	200
3	3	3	2023-12-19	100
4	4	4	2023-12-19	200
5	5	5	2023-12-19	100
6	6	1	2023-12-19	200
7	7	2	2023-12-19	100
8	8	3	2023-12-19	200
9	9	4	2023-12-19	100
10	10	5	2023-12-19	200
11	11	1	2023-12-19	100
12	12	2	2023-12-19	200
13	13	3	2023-12-19	100
14	14	4	2023-12-19	200
15	15	5	2023-12-19	100
16	16	1	2023-12-19	200
17	17	2	2023-12-19	100
18	18	3	2023-12-19	200
19	19	4	2023-12-19	100
20	20	5	2023-12-19	200
21	21	1	2023-12-19	100
23	23	3	2023-12-19	100
24	24	4	2023-12-19	200
25	25	5	2023-12-19	100
26	26	1	2023-12-19	200
27	27	2	2023-12-19	100
28	28	3	2023-12-19	200
29	29	4	2023-12-19	100
30	30	5	2023-12-19	200
31	31	1	2023-12-19	100
32	32	2	2023-12-19	200
33	33	3	2023-12-19	100
34	34	4	2023-12-19	200
35	35	5	2023-12-19	100
36	36	1	2023-12-19	200
37	37	2	2023-12-19	100
38	38	3	2023-12-19	200
39	39	4	2023-12-19	100
40	40	5	2023-12-19	200

Рис. А.5 Данные таблицы «ПоступлениеНаСклад»

Id_BookReserv	Id_Book	Id_Reader	DateOfReserv
1	1	1	2023-12-25
2	2	2	2023-12-25
3	3	3	2023-12-25
4	4	4	2023-12-25
5	5	5	2023-12-25
6	6	6	2023-12-25
7	7	7	2023-12-25
8	8	8	2023-12-25
9	9	9	2023-12-25
10	10	10	2023-12-25
11	11	11	2023-12-25
12	12	12	2023-12-25
13	13	13	2023-12-25
14	14	14	2023-12-25
15	15	15	2023-12-25
16	16	16	2023-12-25
17	17	17	2023-12-25
18	18	18	2023-12-25
19	19	19	2023-12-25
20	20	20	2023-12-25
21	21	21	2023-12-25
23	23	23	2023-12-25
24	24	24	2023-12-25
25	25	25	2023-12-25
26	26	26	2023-12-25
27	27	27	2023-12-25
28	28	28	2023-12-25
29	29	29	2023-12-25
30	30	30	2023-12-25
31	31	31	2023-12-25
32	32	32	2023-12-25
33	33	33	2023-12-25
34	34	34	2023-12-25
35	35	35	2023-12-25
36	36	36	2023-12-25
37	37	37	2023-12-25
38	38	38	2023-12-25
39	39	39	2023-12-25
40	40	40	2023-12-25

Рис. А.6 Данные таблицы «РезервированиеКниги»

## ПРИЛОЖЕНИЕ Б

### SQL-операторы создания основных объектов БД

1. Триггер «trigAddArrivalAtWarehouse»

```
ALTER TRIGGER [dbo].[trigAddArrivalAtWarehouse] ON [dbo].[ArrivalAtWarehouse]
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @id_arrival INT
    IF NOT EXISTS (SELECT Id_Arrival FROM ArrivalAtWarehouse)
    BEGIN
```

```

SET @id_arrival = 1

INSERT INTO ArrivalAtWarehouse SELECT @id_arrival, Id_Book, Id_Employee,
DateOfArrival, Number FROM inserted

DECLARE @num INT, @id_book INT

SET @id_book = (SELECT Id_Book FROM ArrivalAtWarehouse WHERE Id_Arrival =
@id_arrival)

SET @num = (SELECT Number FROM ArrivalAtWarehouse WHERE Id_Arrival =
@id_arrival)

UPDATE Book

SET Number = Number + @num WHERE Id_Book = @id_book

END

ELSE

BEGIN

SELECT @id_arrival = MAX(Id_Arrival) FROM ArrivalAtWarehouse

INSERT INTO ArrivalAtWarehouse SELECT @id_arrival+1, Id_Book, Id_Employee,
DateOfArrival, Number FROM inserted

SET @id_book = (SELECT Id_Book FROM ArrivalAtWarehouse WHERE Id_Arrival =
@id_arrival+1)

SET @num = (SELECT Number FROM ArrivalAtWarehouse WHERE Id_Arrival =
@id_arrival+1)

UPDATE Book

SET Number = Number + @num WHERE Id_Book = @id_book

END

END

```

## 2. Хранимая процедура «AddArrivalAtWarehouse»

```

ALTER PROCEDURE [dbo].[AddArrivalAtWarehouse]
@id_book INT, @phone VARCHAR(12), @date DATE, @number INT, @message
VARCHAR(50) OUTPUT
AS
BEGIN
IF EXISTS (SELECT Id_Book FROM Book WHERE Id_Book = @id_book)
BEGIN
IF EXISTS (SELECT PhoneNumber FROM Employee WHERE PhoneNumber = @phone)
BEGIN
declare @id_employee INT = (SELECT Id_Employee FROM Employee WHERE
PhoneNumber = @phone)
IF (@date < CAST(GETDATE() AS DATE))
SET @message = 'Incorrect date of arrival'
ELSE

```

```

BEGIN
IF (@number <= 0)
    SET @message = 'Incorrect number of books'
ELSE
BEGIN
    Declare @status VARCHAR(50) = (SELECT Position FROM Employee
WHERE PhoneNumber = @phone)
    IF ( @status = 'Уволен' )
        SET @message = 'Employee fired'
    ELSE
    BEGIN
        SET @message = 'Arrival succeeded'
        INSERT INTO ArrivalAtWarehouse VALUES (0, @id_book,
@id_employee, @date, @number)
    END
    END
END
END
ELSE
    SET @message = 'Incorrect ID employee'
END
ELSE
    SET @message = 'Incorrect ID book'
END
END

3. Триггер «RegReader»
ALTER TRIGGER [dbo].[RegReader] ON [dbo].[Reader] INSTEAD OF INSERT
AS
BEGIN
    DECLARE @id_reader INT
    IF NOT EXISTS (SELECT Id_Reader FROM Reader)
    BEGIN
        SET @id_reader = 1
        INSERT INTO Reader SELECT @id_reader, FIO, Gender, PhoneNumber, Birthday
FROM inserted
    END
    ELSE
    BEGIN
        SELECT @id_reader = MAX(Id_Reader) FROM Reader
        INSERT INTO Reader SELECT @id_reader+1, FIO, Gender, PhoneNumber, Birthday
FROM inserted
    END
END
END

4. Хранимая процедура «RegisterReader»
ALTER PROCEDURE [dbo].[RegisterReader]
@fio VARCHAR(50), @gender CHAR(1), @phone VARCHAR(12), @birth DATE,
@message VARCHAR(50) OUTPUT
AS
BEGIN
    IF EXISTS (SELECT PhoneNumber FROM Reader WHERE PhoneNumber = @phone)
        SET @message = 'Reader already registered'
    ELSE

```

```

IF (@gender = 'м' OR @gender = 'ж')
BEGIN
    SET @message = 'Reader registered'
    INSERT INTO Reader VALUES (0, @fio, @gender, @phone, @birth)
END
ELSE
    SET @message = 'Incorrect gender'
END

```

#### 5. Хранимая процедура «UpdateReader»

```

ALTER PROCEDURE [dbo].[UpdateReader]
@Id_Reader INT, @newPhone VARCHAR(12), @message VARCHAR(50) OUTPUT
AS
BEGIN
    IF NOT EXISTS (SELECT Id_Reader FROM Reader WHERE Id_Reader = @Id_Reader)
    SET @message = 'Reader not found'
    ELSE
        IF EXISTS (SELECT PhoneNumber FROM Reader WHERE PhoneNumber =
@newPhone AND Id_Reader <> @Id_Reader)
            SET @message = 'Phone number is already use'
        ELSE
            BEGIN
                SET @message = 'Reader updated'
                UPDATE Reader
                SET PhoneNumber = @newPhone
                WHERE Id_Reader = @Id_Reader
            END
END

```

#### 6. Хранимая процедура «SearchReader»

```

ALTER PROCEDURE [dbo].[SearchReader]
@FIO VARCHAR(50),
@phone VARCHAR(12),
@message VARCHAR(50) OUTPUT
AS
BEGIN
    IF @FIO IS NOT NULL AND @phone IS NOT NULL
    BEGIN
        IF EXISTS (SELECT FIO, PhoneNumber FROM Reader WHERE FIO LIKE '%' +
@FIO + '%' AND PhoneNumber LIKE '%' + @phone + '%')
        BEGIN
            SET @message = 'Reader found'
            SELECT FIO AS [ФИО], PhoneNumber AS [Номер телефона], Birthday AS [Дата
рождения]
            FROM Reader
            WHERE FIO LIKE '%' + @FIO + '%' AND PhoneNumber LIKE '%' + @phone + '%'
        END
        ELSE
            SET @message = 'Reader not exist'
        END
    ELSE IF @FIO IS NOT NULL
    BEGIN

```

```

        IF EXISTS (SELECT FIO, PhoneNumber FROM Reader WHERE FIO LIKE '%' +
@FIO + '%')
        BEGIN
            SET @message = 'Reader found'
            SELECT FIO AS [ФИО], PhoneNumber AS [Номер телефона], Birthday AS [Дата
рождения]
            FROM Reader
            WHERE FIO LIKE '%' + @FIO + '%'
        END
        ELSE
            SET @message = 'Reader with FIO not exist'
        END
        ELSE IF @phone IS NOT NULL
        BEGIN
            IF EXISTS (SELECT FIO, PhoneNumber FROM Reader WHERE PhoneNumber LIKE
'%' + @phone + '%')
            BEGIN
                SET @message = 'Reader found'
                SELECT FIO AS [ФИО], PhoneNumber AS [Номер телефона], Birthday AS [Дата
рождения]
                FROM Reader
                WHERE PhoneNumber LIKE '%' + @phone + '%'
            END
            ELSE
                SET @message = 'Reader with phone not exist'
            END
            ELSE
                SET @message = 'Invalid patametres'
            END
        END
    
```

#### 7. Хранимая процедура «AddBook»

```

ALTER PROCEDURE [dbo].[AddBook]
@author VARCHAR(50), @name VARCHAR(50), @genre VARCHAR(50), @ISBN
VARCHAR(20), @UDK VARCHAR(20), @message VARCHAR(50) OUTPUT
AS
BEGIN
    DECLARE @id_book INT
    IF NOT EXISTS (SELECT Author, [Name], Genre, ISBN, UDK FROM Book WHERE
Author = @author AND [Name] = @name AND Genre = @genre AND ISBN = @ISBN
AND UDK = @UDK)
    BEGIN
        IF EXISTS (SELECT Id_Book From Book)
        BEGIN
            SET @message = 'Book added'
            SET @id_book = (SELECT MAX(Id_Book) FROM Book)
            INSERT INTO Book VALUES (@id_book+1, @author, @name, @genre, @ISBN,
@UDK, 0)
        END
        ELSE
            BEGIN
                SET @message = 'Book added'
            END
        END
    
```

```

        SET @id_book = 1
        INSERT INTO Book VALUES (@id_book, @author, @name, @genre, @ISBN,
@UDK, 0)
    END
END
ELSE
    SET @message = 'The book already exists'
END

```

#### 8. Хранимая процедура «SearchBook»

```

ALTER PROCEDURE [dbo].[SearchBook]
@author VARCHAR(50), @name VARCHAR(50), @message VARCHAR(50) OUTPUT
AS
BEGIN
    IF EXISTS (SELECT Author, Name FROM Book WHERE Author LIKE '%' + @author +
 '%' OR [Name] LIKE '%' + @name + '%')
    BEGIN
        SET @message = 'Books shown'
        SELECT Author AS [Автор книги], [Name] AS [Название книги], Genre AS [Жанр],
Number AS [Наличие] FROM Book WHERE Author = @author OR [Name] = @name
    END
    ELSE
        SET @message = 'No such book'
    END

```

#### 9. Хранимая процедура «AddReservation»

```

ALTER PROCEDURE [dbo].[AddReservation]
@author VARCHAR(50), @name VARCHAR(50), @phone VARCHAR(12), @date DATE,
@message VARCHAR(50) OUTPUT
AS
BEGIN
    DECLARE @id_book INT
    IF EXISTS (SELECT PhoneNumber FROM Reader WHERE PhoneNumber = @phone)
    BEGIN
        IF EXISTS (SELECT Author, [Name] FROM Book WHERE Author = @author AND
[Name] = @name)
        BEGIN
            IF EXISTS (SELECT Author, [Name] FROM Book WHERE Author = @author AND
[Name] = @name AND Number > 1)
            BEGIN
                --SET @id_book = (SELECT Id_Book FROM Book WHERE Author = @author AND
[Name] = @name)
                IF (@date < CAST(GETDATE() AS DATE))
                    SET @message = 'Date must be at least today'
                ELSE
                BEGIN
                    SET @message = 'Added to reservation'
                    DECLARE @id_reader INT, @id_bookreserv INT
                    SET @id_reader = (SELECT Id_Reader FROM Reader WHERE PhoneNumber =
@phone)
                    SET @id_book = (SELECT Id_Book FROM Book WHERE Author = @author AND
[Name] = @name)

```



```

        IF EXISTS (SELECT Id_Book, Id_Reader, DateOfReserv From
BookReservation WHERE Id_Book = @id_book and Id_Reader = @id_reader and
DateOfReserv = @date)
            SET @message = 'You cannot re-reserve a book'
        ELSE
        BEGIN
            IF NOT EXISTS (SELECT Id_BookReserv FROM BookReservation)
            BEGIN
                SET @id_bookreserv = 1
                INSERT INTO BookReservation VALUES (@id_bookreserv,
@id_book, @id_reader, @date)
            END
        ELSE
        BEGIN
            SET @id_bookreserv = (SELECT MAX(Id_BookReserv)
FROM BookReservation)
            INSERT INTO BookReservation VALUES
(@id_bookreserv+1, @id_book, @id_reader, @date)
        END
        UPDATE Book
        SET Number = Number - 1 WHERE Id_Book = @id_book
    END
END
END
ELSE
    SET @message = 'Quantity lower than 1'
END
ELSE
    SET @message = 'No such book exists'
END
ELSE
    SET @message = 'Incorrect phone'
END

```

10. Хранимая процедура «DeleteReservation»

```
ALTER PROCEDURE [dbo].[DeleteReservation]
```

```
@author VARCHAR(50), @name VARCHAR(50), @phone VARCHAR(12), @date DATE,
@message VARCHAR(50) OUTPUT
```

```
AS
```

```
BEGIN
```

```
    DECLARE @id_book INT, @id_reader INT, @id_bookreserv INT
```

```
    IF EXISTS (SELECT PhoneNumber FROM Reader WHERE PhoneNumber = @phone)
```

```
    BEGIN
```

```
        SET @id_reader = (SELECT Id_Reader FROM Reader WHERE PhoneNumber =
@phone)
```

```
        IF EXISTS (SELECT Author, [Name] FROM Book WHERE Author = @author AND
[Name] = @name)
```

```
        BEGIN
```

```
            SET @id_book = (SELECT Id_Book FROM Book WHERE Author = @author AND
[Name] = @name)
```

```
            IF EXISTS (SELECT Id_Book, Id_Reader, DateOfReserv FROM BookReservation
WHERE Id_Book = @id_book AND Id_Reader = @id_reader AND DateOfReserv = @date)
```

```
            BEGIN
```

```

        SET @message = 'Reservation deleted'
        DELETE FROM BookReservation WHERE Id_Book = @id_book AND Id_Reader =
@id_reader AND DateOfReserv = @date
        UPDATE Book
        SET Number = Number + 1 WHERE Id_Book = @id_book
    END
    ELSE
        SET @message = 'No such registration'
    END
    ELSE
        SET @message = 'No such book exists'
    END
    ELSE
        SET @message = 'Incorrect phone'
    END
END

```

11. Хранимая процедура «AddEmployee»

```

ALTER PROCEDURE [dbo].[AddEmployee]
    @fio VARCHAR(50), @position VARCHAR(50), @phone VARCHAR(12), @passport
VARCHAR(10), @message VARCHAR(50) OUTPUT
AS
BEGIN
    DECLARE @id_employee INT
    IF NOT EXISTS (SELECT PhoneNumber FROM Employee WHERE PhoneNumber =
@phone)
    BEGIN
        IF NOT EXISTS (SELECT PassportNumber FROM Employee WHERE PassportNumber =
@passport)
        BEGIN
            If (@position = 'Уволен')
            Begin
                SET @message = 'no fired position'
            End
            Else
            Begin
                IF NOT EXISTS (SELECT Position From Employee WHERE Position = @position)
                SET @message = 'Incorrect position'
            END
        ELSE
            BEGIN
                Set @message = 'The employee added'
                IF NOT EXISTS (SELECT Id_Employee FROM Employee)
                BEGIN
                    SET @id_employee = 1
                    INSERT INTO Employee VALUES (@id_employee, @fio, @position,
@phone, @passport)
                END
            ELSE
                BEGIN
                    SET @id_employee = (SELECT MAX(Id_Employee) FROM
Employee)

```

```

INSERT INTO Employee VALUES (@id_employee+1, @fio,
@position, @phone, @passport)
END
END
END
ELSE
SET @message = 'Passport already exists'
END
ELSE
SET @message = 'Number already exists'
END

```

## 12. Хранимая процедура «FireEmployee»

```

ALTER PROCEDURE [dbo].[FireEmployee]
@passport VARCHAR(10),
@message VARCHAR(50) OUTPUT
AS
BEGIN
DECLARE @id_employee INT
IF EXISTS (SELECT Id_Employee FROM Employee WHERE PassportNumber =
@passport)
BEGIN
SET @id_employee = (SELECT Id_Employee FROM Employee WHERE
PassportNumber = @passport)
IF EXISTS (SELECT Id_Employee FROM Employee WHERE Id_Employee =
@id_employee AND Position = 'Уволен')
BEGIN
SET @message = 'Employee already fired'
END
ELSE
BEGIN
-- Обновляем столбец Position на "уволен"
UPDATE Employee
SET Position = 'Уволен'
WHERE Id_Employee = @id_employee

SET @message = 'Employee fired'
END
END
ELSE
BEGIN
SET @message = 'Employee does not exist'
END
END

```

## 13. Хранимая процедура « BookIssuance »

```

ALTER PROCEDURE [dbo].[BookIssuance]
@author VARCHAR(50), @name VARCHAR(50), @phoneReader VARCHAR(12),
@phoneEmployee VARCHAR(12), @dateIssuance DATE, @dateDelivery DATE,
@message VARCHAR(50) OUTPUT
AS

```

```

BEGIN
    Declare @status VARCHAR(50) = (SELECT Position FROM Employee WHERE
    PhoneNumber = @phoneEmployee)
    IF ( @status = 'Уволен' )
        SET @message = 'Employee fired'
    ELSE
        BEGIN
            DECLARE @id_book INT
            IF EXISTS (SELECT PhoneNumber FROM Reader WHERE PhoneNumber =
            @phoneReader)
                BEGIN
                    IF EXISTS (SELECT PhoneNumber FROM Employee WHERE PhoneNumber =
                    @phoneEmployee)
                        BEGIN
                            IF EXISTS (SELECT Author, [Name] FROM Book WHERE Author = @author AND
                            [Name] = @name)
                                BEGIN
                                    IF EXISTS (SELECT Author, [Name] FROM Book WHERE Author = @author AND
                                    [Name] = @name AND Number > 1)
                                        BEGIN
                                            --SET @id_book = (SELECT Id_Book FROM Book WHERE Author = @author AND
                                            [Name] = @name)
                                            IF (@dateIssuance < CAST(GETDATE() AS DATE))
                                                SET @message = 'Date must be at least today'
                                            ELSE
                                                BEGIN
                                                    IF (@dateIssuance > @dateDelivery)
                                                        SET @message = 'dateDelivery less dateIssuance'
                                                    ELSE
                                                        BEGIN
                                                            DECLARE @id_reader INT, @id_issuanceDelivery INT, @id_employee INT
                                                            SET @id_reader = (SELECT Id_Reader FROM Reader WHERE PhoneNumber =
                                                            @phoneReader)
                                                            SET @id_employee = (SELECT Id_Employee FROM Employee WHERE
                                                            PhoneNumber = @phoneEmployee)
                                                            SET @id_book = (SELECT Id_Book FROM Book WHERE Author = @author AND
                                                            [Name] = @name)
                                                            IF EXISTS (SELECT Id_Book, Id_Reader, Status From IssuanceDelivery
                                                            WHERE Id_Book = @id_book and Id_Reader = @id_reader and Status = 0)
                                                                SET @message = 'Book cannot be issued a second time'
                                                            ELSE
                                                                BEGIN
                                                                    SET @message = 'Book has been issued'
                                                                    IF NOT EXISTS (SELECT Id_IssuanceDelivery FROM
                                                                    IssuanceDelivery)
                                                                        BEGIN
                                                                            SET @id_issuanceDelivery = 1
                                                                            INSERT INTO IssuanceDelivery VALUES
                                                                            (@id_issuanceDelivery, @id_book, @id_reader, @id_employee, @dateIssuance,
                                                                            @dateDelivery, 0)
                                                                        END
                                                                    ELSE

```

```

        BEGIN
            SET @id_issuanceDelivery = (SELECT
MAX(Id_IssuanceDelivery) FROM IssuanceDelivery)
            INSERT INTO IssuanceDelivery VALUES
(@id_issuanceDelivery+1, @id_book, @id_reader, @id_employee, @dateIssuance,
@dateDelivery, 0)
            END
            UPDATE Book
            SET Number = Number - 1 WHERE Id_Book = @id_book
        END
    END
END
END
ELSE
    SET @message = 'Quantity lower than 1'
END
ELSE
    SET @message = 'No such book exists'
    END
    ELSE
    SET @message = 'Incorrect phone Employee'
    END
ELSE
    SET @message = 'Incorrect phone Reader'
END
END

```

#### 14. Хранимая процедура «AcceptTheBook»

```

ALTER PROCEDURE [dbo].[AcceptTheBook]
    @id INT, @message VARCHAR(50) OUTPUT
AS BEGIN
    IF EXISTS (SELECT Id_IssuanceDelivery FROM IssuanceDelivery WHERE
Id_IssuanceDelivery = @id)
        BEGIN
            declare @id_book INT
            SET @id_book = (SELECT Id_Book FROM IssuanceDelivery WHERE
IssuanceDelivery.Id_IssuanceDelivery = @id)
            BEGIN
                IF (1 = (SELECT [Status] FROM IssuanceDelivery WHERE
IssuanceDelivery.Id_IssuanceDelivery = @id))
                    SET @message = 'Book already returned'
                ELSE
                    BEGIN
                        SET @message = 'Book returned'
                    END
            END
            UPDATE IssuanceDelivery
            SET [Status] = 1 WHERE Id_IssuanceDelivery = @id
            UPDATE Book
            SET Number = Number + 1 WHERE Id_Book = @id_book
        END
    END
END
ELSE

```

```
SET @message = 'Incorrect ID'
END
```

## ПРИЛОЖЕНИЕ В

### Код программы

#### 1. Форма «LoginForm»

```
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 LibraryBD
{
    public partial class LoginForm : Form
    {
        public LoginForm()
        {
            InitializeComponent();

            private void label1_Click(object sender, EventArgs e)
            {

            }

            private void button1_Click(object sender, EventArgs e)
            {
                Form form2 = new Form2();
                form2.Show();
            }

            private void button2_Click(object sender, EventArgs e)
            {
                Form form9 = new Form9();
                form9.Show();
            }

            private void button3_Click(object sender, EventArgs e)
            {
                Form form12 = new Form12();
                form12.Show();
            }

            private void button4_Click(object sender, EventArgs e)
            {
                Form form16 = new Form16();
                form16.Show();
            }

            private void LoginForm_Load(object sender, EventArgs e)
            {

            }
        }
    }
}
```

#### 2. Форма «Консультант»

```

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 LibraryBD
{
    public partial class Form2 : Form
    {
        public Form2()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Form form7 = new Form7();
            form7.Show();
        }

        private void bookReservationBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.bookReservationBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.libraryDataSet);
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.BookReservation".
            При необходимости она может быть перемещена или удалена.
            this.bookReservationTableAdapter.Fill(this.libraryDataSet.BookReservation);
        }

        private void button3_Click(object sender, EventArgs e)
        {
            Form form8 = new Form8();
            form8.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form form3 = new Form3();
            form3.Show();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            Form form6 = new Form6();
            form6.Show();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            Form form18 = new Form18();
            form18.Show();
        }
    }
}

```

### 3. Форма «Резервирование книги»

```
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 LibraryBD
{
    public partial class Form3 : Form
    {
        public Form3()
        {
            InitializeComponent();

            private void bookReservationBindingNavigatorSaveItem_Click(object sender, EventArgs e)
            {
                this.Validate();
                this.bookReservationBindingSource.EndEdit();
                this.tableAdapterManager.UpdateAll(this.libraryDataSet);
            }

            private void Form3_Load(object sender, EventArgs e)
            {
                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Employee". При
                // необходимости она может быть перемещена или удалена.
                this.employeeTableAdapter.Fill(this.libraryDataSet.Employee);
                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.BookReservation".
                // При необходимости она может быть перемещена или удалена.
                this.bookReservationTableAdapter.Fill(this.libraryDataSet.BookReservation);
                string connString = @"Data Source=ANCHA\EVPC\SQLALDAR;Initial Catalog=Library;Integrated
                Security=True";
                SqlConnection connection = new SqlConnection(connString);
                SqlCommand command = new SqlCommand();
                command.Connection = connection;
                command.CommandType = System.Data.CommandType.Text;
                command.CommandText = " SELECT Reader.FIO AS [ФИО Читателя], Reader.PhoneNumber AS [Номер
                телефона], Book.Author AS [Автор книги], Book.Name AS [Название книги], DateOfReserv AS [Дата
                резервирования] FROM Reader JOIN BookReservation ON Reader.Id_Reader = BookReservation.Id_Reader
                JOIN Book ON BookReservation.Id_Book = Book.Id_Book";
                connection.Open();
                SqlDataAdapter adapter = new SqlDataAdapter(command);
                DataTable dt = new DataTable();
                adapter.Fill(dt);
                dataGridView1.DataSource = dt;
                connection.Close();
            }

            private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
            {
            }

            private void button1_Click(object sender, EventArgs e)
            {
                Form form4 = new Form4();
            }
        }
    }
}
```



```

        form4.Show();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        Form form5 = new Form5();
        form5.Show();
    }

    private void button4_Click(object sender, EventArgs e)
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = " SELECT Reader.FIO AS [ФИО Читателя], Reader.PhoneNumber AS [Номер
телефона], Book.Author AS [Автор книги], Book.Name AS [Название книги], DateOfReserv AS [Дата
резервирования] FROM Reader JOIN BookReservation ON Reader.Id_Reader = BookReservation.Id_Reader
JOIN Book ON BookReservation.Id_Book = Book.Id_Book";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void label1_Click(object sender, EventArgs e)
    {
    }

    private void button5_Click(object sender, EventArgs e)
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Reader.FIO AS [ФИО Читателя], Reader.PhoneNumber AS [Номер
телефона], Book.Author AS [Автор книги], Book.Name AS [Название книги], DateOfReserv AS [Дата
резервирования] FROM Reader JOIN BookReservation ON Reader.Id_Reader = BookReservation.Id_Reader
JOIN Book ON BookReservation.Id_Book = Book.Id_Book WHERE Reader.FIO LIKE '%" + textBox1.Text + "%'";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }
}
}
}

```

#### 4. Форма «Резервирование книги»

```

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 LibraryBD
{
    public partial class Form4 : Form
    {
        public Form4()
        {
            InitializeComponent();

            private void readerBindingNavigatorSaveItem_Click(object sender, EventArgs e)
            {
                this.Validate();
                this.readerBindingSource.EndEdit();
                this.tableAdapterManager.UpdateAll(this.libraryDataSet);

            }

            private void Form4_Load(object sender, EventArgs e)
            {
                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.BookReservation".
                // При необходимости она может быть перемещена или удалена.
                this.bookReservationTableAdapter.Fill(this.libraryDataSet.BookReservation);
                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
                // необходимости она может быть перемещена или удалена.
                //this.bookTableAdapter.Fill(this.libraryDataSet.Book);
                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Reader". При
                // необходимости она может быть перемещена или удалена.
                this.readerTableAdapter.Fill(this.libraryDataSet.Reader);

            }

            private void fIOLabel_Click(object sender, EventArgs e)
            {

            }

            private void button1_Click(object sender, EventArgs e)
            {
                try
                {
                    if (string.IsNullOrEmpty(authorTextBox.Text) || string.IsNullOrEmpty(nameTextBox.Text) ||
string.IsNullOrEmpty(phoneNumberTextBox.Text))
                    {
                        MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
MessageBox.Icon.Warning);
                    }
                    else
                    {
                        myCommand.Parameters["@author"].Value = authorTextBox.Text;
                        myCommand.Parameters["@name"].Value = nameTextBox.Text;
                        myCommand.Parameters["@phone"].Value = phoneNumberTextBox.Text;
                        myCommand.Parameters["@date"].Value = Convert.ToDateTime(dateOfReserv.Text);
                        mySqlConnection.Open();
                        myCommand.ExecuteNonQuery();
                        mySqlConnection.Close();
                        if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Date must be at least today")
                        {
                            MessageBox.Show("Неверная дата", "Окно уведомлений", MessageBoxButtons.OK,
MessageBox.Icon.Information);
                        }
                        else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Added to reservation")
                        {

```

```

        MessageBox.Show("Книга зарезервирована", "Окно уведомлений", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
    }
    else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Quantity lower than 1")
    {
        MessageBox.Show("Книги нет в наличии", "Окно уведомлений", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
    }
    else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "No such book exists")
    {
        MessageBox.Show("Такой книги не существует", "Окно уведомлений", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
    }
    else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect phone")
    {
        MessageBox.Show("Неправильный номер телефона", "Окно уведомлений",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "You cannot re-reserve a
    book")
    {
        MessageBox.Show("Нельзя повторно зарезервировать книгу", "Окно уведомлений",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    }
}

catch
{
    MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

}

private void mySqlConnection_InfoMessage(object sender, SqlInfoMessageEventArgs e)
{
}

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

```

## 5. Форма «Удаление резервирования»

```

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 LibraryBD
{
    public partial class Form5 : Form
    {
        public Form5()
        {
            InitializeComponent();
        }
    }
}

```

```

private void readerBindingNavigatorSaveItem_Click(object sender, EventArgs e)
{
    this.Validate();
    this.readerBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(this.libraryDataSet);
}

private void Form5_Load(object sender, EventArgs e)
{
    // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
    // необходимости она может быть перемещена или удалена.
    //this.bookTableAdapter.Fill(this.libraryDataSet.Book);
    // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.BookReservation".
    // При необходимости она может быть перемещена или удалена.
    this.bookReservationTableAdapter.Fill(this.libraryDataSet.BookReservation);
    // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Reader". При
    // необходимости она может быть перемещена или удалена.
    this.readerTableAdapter.Fill(this.libraryDataSet.Reader);
}

private void sqlConnection1_InfoMessage(object sender, System.Data.SqlClient.SqlInfoMessageEventArgs e)
{
}

private void button1_Click(object sender, EventArgs e)
{
    try
    {
        if (string.IsNullOrEmpty(authorTextBox.Text) || string.IsNullOrEmpty(nameTextBox.Text) ||
            string.IsNullOrEmpty(phoneNumberTextBox.Text))
        {
            MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
        }
        else
        {
            myCommand.Parameters["@author"].Value = authorTextBox.Text;
            myCommand.Parameters["@name"].Value = nameTextBox.Text;
            myCommand.Parameters["@phone"].Value = phoneNumberTextBox.Text;
            myCommand.Parameters["@date"].Value = Convert.ToDateTime(dateOfReserv.Text);
            //myCommand.Parameters["@message"].Value = ();
            mySqlConnection.Open();
            myCommand.ExecuteNonQuery();
            mySqlConnection.Close();
            if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reservation deleted")
            {
                MessageBox.Show("Резервация удалена", "Окно уведомлений", MessageBoxButtons.OK,
                    MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "No such registration")
            {
                MessageBox.Show("Резервации не существует", "Окно уведомлений", MessageBoxButtons.OK,
                    MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "No such book exists")
            {
                MessageBox.Show("Такой книги не существует", "Окно уведомлений", MessageBoxButtons.OK,
                    MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect phone")
            {
            }
        }
    }
    catch
    {
    }
}

```

```

        MessageBox.Show("Неправильный номер телефона", "Окно уведомлений",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }

    }
}

catch
{
    MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

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

}
}
}

```

## 6. Форма «Добавление читателя»

```

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 LibraryBD
{
    public partial class Form6 : Form
    {
        public Form6()
        {
            InitializeComponent();
        }

        private void readerBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.readerBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.libraryDataSet);
        }

        private void Form6_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Reader". При
            // необходимости она может быть перемещена или удалена.
            this.readerTableAdapter.Fill(this.libraryDataSet.Reader);
        }

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

        private void button2_Click(object sender, EventArgs e)
        {
            try
            {

```

```

        if (string.IsNullOrEmpty(fioTextBox.Text) || string.IsNullOrEmpty(genderComboBox.Text) ||
            string.IsNullOrEmpty(phoneNumberTextBox.Text))
        {
            MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
        }
        else
        {
            myCommand.Parameters["@fio"].Value = fioTextBox.Text;
            myCommand.Parameters["@gender"].Value = genderComboBox.Text;
            myCommand.Parameters["@phone"].Value = phoneNumberTextBox.Text;
            myCommand.Parameters["@birth"].Value = Convert.ToDateTime(birthdayDate.Text);
            //myCommand.Parameters["@message"].Value = ();
            mySqlConnection.Open();
            myCommand.ExecuteNonQuery();
            mySqlConnection.Close();
            if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader already registered")
            {
                MessageBox.Show("Ошибка! Читатель с таким номер телефона уже зарегистрирован.", "Окно
уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader registered")
            {
                MessageBox.Show("Читатель успешно зарегистрирован.", "Окно уведомлений",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect gender")
            {
                MessageBox.Show("Ошибка! Некорректный пол.", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
        }
    }
}

catch
{
    MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}
}

```

## 7. Форма «Поиск книги»

```

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 LibraryBD
{
    public partial class Form7 : Form
    {
        public Form7()
        {
            InitializeComponent();
        }
    }
}

```

```

private void bookBindingNavigatorSaveItem_Click(object sender, EventArgs e)
{
    this.Validate();
    this.bookBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(this.libraryDataSet);
}

private void Form7_Load(object sender, EventArgs e)
{
    // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
    // необходимости она может быть перемещена или удалена.
    //this.bookTableAdapter.Fill(this.libraryDataSet.Book);
    string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
    Security=True";
    SqlConnection connection = new SqlConnection(connString);
    SqlCommand command = new SqlCommand();
    command.Connection = connection;
    command.CommandType = System.Data.CommandType.Text;
    command.CommandText = "SELECT Book.Author AS [Автор книги], Book.Name AS [Название книги],
    Book.Genre AS [Жанр], Book.Number AS [Наличие] From Book";
    connection.Open();
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    DataTable dt = new DataTable();
    adapter.Fill(dt);
    dataGridView1.DataSource = dt;
    connection.Close();
}

private void button1_Click(object sender, EventArgs e)
{
    try
    {
        if (string.IsNullOrEmpty(authorTextBox.Text) && string.IsNullOrEmpty(nameTextBox.Text))
        {
            MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
        }
        else
        {
            myCommand.Parameters["@author"].Value = authorTextBox.Text;
            myCommand.Parameters["@name"].Value = nameTextBox.Text;
            mySqlConnection.Open();
            myCommand.ExecuteNonQuery();
            mySqlConnection.Close();
            if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Books shown")
            {
                MessageBox.Show("Книга найдена!", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "No such book")
            {
                MessageBox.Show("Такой книги не существует", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
        }
    }
    catch
    {
        MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

```

        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Author AS [Автор], Name AS [Название книги], Genre AS [Жанр],
Number AS [Наличие] FROM Book WHERE Author LIKE '%" + authorTextBox.Text + "%' AND Name LIKE '%" +
nameTextBox.Text + "%'";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {
    }

    private void button2_Click(object sender, EventArgs e)
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Author AS [Автор], Name AS [Название книги], Genre AS [Жанр],
Number AS [Наличие] FROM Book";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }
}
}

```

## 8. Форма «Поиск читателя»

```

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 LibraryBD
{
    public partial class Form8 : Form
    {
        public Form8()
        {
            InitializeComponent();
        }
    }
}

```



```

private void readerBindingNavigatorSaveItem_Click(object sender, EventArgs e)
{
    this.Validate();
    this.readerBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(this.libraryDataSet);
}

private void Form8_Load(object sender, EventArgs e)
{
    // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Reader". При
    // необходимости она может быть перемещена или удалена.
    this.readerTableAdapter.Fill(this.libraryDataSet.Reader);
    string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
    Security=True";
    SqlConnection connection = new SqlConnection(connString);
    SqlCommand command = new SqlCommand();
    command.Connection = connection;
    command.CommandType = System.Data.CommandType.Text;
    command.CommandText = "SELECT Reader.FIO AS [Читатель], Reader.PhoneNumber AS [Номер
    телефона], Reader.Birthday AS [Дата рождения] From Reader";
    connection.Open();
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    DataTable dt = new DataTable();
    adapter.Fill(dt);
    dataGridView1.DataSource = dt;
    connection.Close();
}

private void button2_Click(object sender, EventArgs e)
{
    try
    {
        if (string.IsNullOrEmpty(fioTextBox.Text) &&
        string.IsNullOrEmpty(phoneNumberTextBox.Text))
        {
            MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
        }
        else
        {
            myCommand.Parameters["@FIO"].Value = fioTextBox.Text;
            myCommand.Parameters["@phone"].Value = phoneNumberTextBox.Text;
            mySqlConnection.Open();
            myCommand.ExecuteNonQuery();
            mySqlConnection.Close();
            if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader found")
            {
                MessageBox.Show("Читатель найден!", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader not exist")
            {
                MessageBox.Show("Читателя не существует", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader with FIO not exist")
            {
                MessageBox.Show("Читателя с такой фамилией не существует", "Окно уведомлений",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader with phone not
            exist")
            {

```

```

        MessageBox.Show("Читателя с таким номером телефона не существует", "Окно уведомлений",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}

catch
{
    MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
SqlConnection connection = new SqlConnection(connString);
SqlCommand command = new SqlCommand();
command.Connection = connection;
command.CommandType = System.Data.CommandType.Text;
command.CommandText = "SELECT FIO AS [ФИО], PhoneNumber AS [Номер телефона], Birthday AS
[Дата рождения] FROM Reader WHERE FIO LIKE '%" + fioTextBox.Text + "%' AND PhoneNumber LIKE '%" +
phoneNumberTextBox.Text + "%'";
connection.Open();
SqlDataAdapter adapter = new SqlDataAdapter(command);
DataTable dt = new DataTable();
adapter.Fill(dt);
dataGridView1.DataSource = dt;
connection.Close();
}

private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
}
}
}

```

## 9. Форма «Библиотекарь»

```

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 LibraryBD
{
    public partial class Form9 : Form
    {
        public Form9()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Form form7 = new Form7();
            form7.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form form11 = new Form11();
            form11.Show();
        }
    }
}

```

```

private void button3_Click(object sender, EventArgs e)
{
    Form form8 = new Form8();
    form8.Show();
}

private void button4_Click(object sender, EventArgs e)
{
    Form form6 = new Form6();
    form6.Show();
}

private void button5_Click(object sender, EventArgs e)
{
    Form form18 = new Form18();
    form18.Show();
}

private void Form9_Load(object sender, EventArgs e)
{
}
}
}

```

## 10. Форма «Поиск книги»

```

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 LibraryBD
{
    public partial class Form10 : Form
    {
        public Form10()
        {
            InitializeComponent();
        }

        private void Form9_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
            // необходимости она может быть перемещена или удалена.
            //this.bookTableAdapter.Fill(this.libraryDataSet.Book);
            string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
            Security=True";
            SqlConnection connection = new SqlConnection(connString);
            SqlCommand command = new SqlCommand();
            command.Connection = connection;
            command.CommandType = System.Data.CommandType.Text;
            command.CommandText = "SELECT Book.Author AS [Автор книги], Book.Name AS [Название книги],
            Book.Genre AS [Жанр], Book.Number AS [Наличие] From Book";
            connection.Open();
            SqlDataAdapter adapter = new SqlDataAdapter(command);
            DataTable dt = new DataTable();

```

```

        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void bookBindingNavigatorSaveItem_Click(object sender, EventArgs e)
    {
        this.Validate();
        this.bookBindingSource.EndEdit();
        this.tableAdapterManager.UpdateAll(this.libraryDataSet);

    }

    private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {

    }

    private void button1_Click(object sender, EventArgs e)
    {
        try
        {
            if (string.IsNullOrEmpty(authorTextBox.Text) || string.IsNullOrEmpty(nameTextBox.Text))
            {
                MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
            }
            else
            {
                string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
                SqlConnection connection = new SqlConnection(connString);
                SqlCommand command = new SqlCommand();
                command.Connection = connection;
                command.CommandType = System.Data.CommandType.Text;
                command.CommandText = "SELECT Author AS [Автор], Name AS [Название книги], Genre AS
[Жанр], Number AS [Наличие] FROM Book WHERE Author LIKE '%" + authorTextBox.Text + "%' AND Name
LIKE '%" + nameTextBox.Text + "%'";
                connection.Open();
                SqlDataAdapter adapter = new SqlDataAdapter(command);
                DataTable dt = new DataTable();
                adapter.Fill(dt);
                dataGridView1.DataSource = dt;
                connection.Close();
            }
        }
        catch
        {
            MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}

```

## 11. Форма «Выдача книги»

```

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 LibraryBD
{
    public partial class Form11 : Form
    {
        public Form11()
        {
            InitializeComponent();

            private void bookBindingNavigatorSaveItem_Click(object sender, EventArgs e)
            {
                this.Validate();
                this.bookBindingSource.EndEdit();
                this.tableAdapterManager.UpdateAll(this.libraryDataSet);

            }

            private void Form9_Load(object sender, EventArgs e)
            {
                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Employee". При
                // необходимости она может быть перемещена или удалена.
                this.employeeTableAdapter.Fill(this.libraryDataSet.Employee);

                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.IssuanceDelivery".
                // При необходимости она может быть перемещена или удалена.
                this.issuanceDeliveryTableAdapter.Fill(this.libraryDataSet.IssuanceDelivery);

                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Reader". При
                // необходимости она может быть перемещена или удалена.
                this.readerTableAdapter.Fill(this.libraryDataSet.Reader);

                // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
                // необходимости она может быть перемещена или удалена.
                // this.bookTableAdapter.Fill(this.libraryDataSet.Book);

                string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
                Security=True";

                SqlConnection connection = new SqlConnection(connString);
                SqlCommand command = new SqlCommand();
                command.Connection = connection;
                command.CommandType = System.Data.CommandType.Text;
                command.CommandText = "SELECT IssuanceDelivery.Id_IssuanceDelivery AS [Айди], Book.Author AS
                [Автор], Book.Name AS [Название книги], Reader.PhoneNumber AS [Номер телефона читателя],
                Employee.PhoneNumber AS [Номер телефона работника], IssuanceDelivery.DateOfIssue AS [Дата выдачи],

```

```

IssuanceDelivery.DateOfDelivery AS [Дата сдачи], Status AS [Книга сдана] FROM Employee INNER JOIN
IssuanceDelivery ON Employee.Id_Employee = IssuanceDelivery.Id_Employee INNER JOIN Book ON
IssuanceDelivery.Id_Book = Book.Id_Book INNER JOIN Reader ON IssuanceDelivery.Id_Reader =
Reader.Id_Reader";

        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();

    }

    private void fIOLabel_Click(object sender, EventArgs e)
    {

    }

    private void button1_Click(object sender, EventArgs e)
    {
        try
        {
            if (string.IsNullOrEmpty(authorTextBox.Text) || string.IsNullOrEmpty(nameTextBox.Text) ||
                string.IsNullOrEmpty(phoneNumberTextBox.Text) || string.IsNullOrEmpty(phoneNumberTextBox1.Text))
            {
                MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
            }
            else
            {
                myCommand.Parameters["@author"].Value = authorTextBox.Text;
                myCommand.Parameters["@name"].Value = nameTextBox.Text;
                myCommand.Parameters["@phoneReader"].Value = phoneNumberTextBox.Text;
                myCommand.Parameters["@phoneEmployee"].Value = phoneNumberTextBox1.Text;
                myCommand.Parameters["@dateIssuance"].Value = Convert.ToDateTime(dateOfIssueDate.Text);
                myCommand.Parameters["@dateDelivery"].Value = Convert.ToDateTime(dateOfDeliveryDate.Text);
                mySqlConnection.Open();
                myCommand.ExecuteNonQuery();
                mySqlConnection.Close();
                if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Date must be at least today")
                {
                    MessageBox.Show("Ошибка! Дата должна быть не позднее сегодняшнего дня.", "Окно
уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
            }
        }
        catch { }
    }

```

```

else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Employee fired")
{
    MessageBox.Show("Ошибка! Сотрудник уволен!", "Окно уведомлений", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "dateDelivery less
dateIssuance")
{
    MessageBox.Show("Ошибка! Дата сдачи не должна быть меньше даты выдачи", "Окно
уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Book cannot be issued a
second time")
{
    MessageBox.Show("Ошибка! Книга уже выдана!", "Окно уведомлений", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Quantity lower than 1")
{
    MessageBox.Show("Ошибка! Книги нет в наличии", "Окно уведомлений",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "No such book exists")
{
    MessageBox.Show("Ошибка! Такой книги не существует!", "Окно уведомлений",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect phone Employee")
{
    MessageBox.Show("Ошибка! Неправильный номер работника", "Окно уведомлений",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect phone Reader")
{
    MessageBox.Show("Ошибка! Неправильный номер читателя", "Окно уведомлений",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Book has been issued")
{
    MessageBox.Show("Книга успешно выдана.", "Окно уведомлений", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
}
}

```

```

        catch
        {
            MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        string connString = @"Data Source=ANCHAЕVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT IssuanceDelivery.Id_IssuanceDelivery AS [Айди], Book.Author AS
[Автор], Book.Name AS [Название книги], Reader.PhoneNumber AS [Номер телефона читателя],
Employee.PhoneNumber AS [Номер телефона работника], IssuanceDelivery.DateOfIssue AS [Дата выдачи],
IssuanceDelivery.DateOfDelivery AS [Дата сдачи], Status AS [Книга сдана] FROM Employee INNER JOIN
IssuanceDelivery ON Employee.Id_Employee = IssuanceDelivery.Id_Employee INNER JOIN Book ON
IssuanceDelivery.Id_Book = Book.Id_Book INNER JOIN Reader ON IssuanceDelivery.Id_Reader =
Reader.Id_Reader";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {
    }

    private void button2_Click(object sender, EventArgs e)
    {
        try
        {
            if (string.IsNullOrEmpty(id_IssuanceDeliveryTextBox.Text))
            {
                MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
            }
            else
            {
                myCommand2.Parameters["@id"].Value = Convert.ToInt32(id_IssuanceDeliveryTextBox.Text);
                mySqlConnection1.Open();
            }
        }
        catch
        {
            MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}

```



```

myCommand2.ExecuteNonQuery();
mySqlConnection1.Close();
if (Convert.ToString(myCommand2.Parameters["@message"].Value) == "Book returned")
{
    MessageBox.Show("Книга успешно сдана.", "Окно уведомлений", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand2.Parameters["@message"].Value) == "Book already returned")
{
    MessageBox.Show("Ошибка! Книга уже сдана", "Окно уведомлений", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
else if (Convert.ToString(myCommand2.Parameters["@message"].Value) == "Incorrect ID")
{
    MessageBox.Show("Ошибка! Некорректный айди", "Окно уведомлений", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}

}
}

catch
{
    MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";

SqlConnection connection = new SqlConnection(connString);
SqlCommand command = new SqlCommand();
command.Connection = connection;
command.CommandType = System.Data.CommandType.Text;
command.CommandText = "SELECT IssuanceDelivery.Id_IssuanceDelivery AS [Айди], Book.Author AS
[Автор], Book.Name AS [Название книги], Reader.PhoneNumber AS [Номер телефона читателя],
Employee.PhoneNumber AS [Номер телефона работника], IssuanceDelivery.DateOfIssue AS [Дата выдачи],
IssuanceDelivery.DateOfDelivery AS [Дата сдачи], Status AS [Книга сдана] FROM Employee INNER JOIN
IssuanceDelivery ON Employee.Id_Employee = IssuanceDelivery.Id_Employee INNER JOIN Book ON
IssuanceDelivery.Id_Book = Book.Id_Book INNER JOIN Reader ON IssuanceDelivery.Id_Reader =
Reader.Id_Reader";

connection.Open();
SqlDataAdapter adapter = new SqlDataAdapter(command);
DataTable dt = new DataTable();
adapter.Fill(dt);
dataGridView1.DataSource = dt;
connection.Close();

```

```

    }

    private void button3_Click(object sender, EventArgs e)
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT IssuanceDelivery.Id_IssuanceDelivery AS [Айди], Book.Author AS
[Автор], Book.Name AS [Название книги], Reader.PhoneNumber AS [Номер телефона читателя],
Employee.PhoneNumber AS [Номер телефона работника], IssuanceDelivery.DateOfIssue AS [Дата выдачи],
IssuanceDelivery.DateOfDelivery AS [Дата сдачи], Status AS [Книга сдана] FROM Employee INNER JOIN
IssuanceDelivery ON Employee.Id_Employee = IssuanceDelivery.Id_Employee INNER JOIN Book ON
IssuanceDelivery.Id_Book = Book.Id_Book INNER JOIN Reader ON IssuanceDelivery.Id_Reader =
Reader.Id_Reader WHERE Reader.PhoneNumber LIKE '%" + phoneNumberTextBox.Text + "%'";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void mySqlConnection_InfoMessage(object sender, SqlInfoMessageEventArgs e)
    {
    }
}
}

```

## 12. Форма «Библиограф»

```

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 LibraryBD
{
    public partial class Form12 : Form
    {
        public Form12()
        {
            InitializeComponent();
        }
    }
}

```

```

    }

    private void button2_Click(object sender, EventArgs e)
    {
        Form form13 = new Form13();
        form13.Show();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        Form form14 = new Form14();
        form14.Show();
    }

    private void Form12_Load(object sender, EventArgs e)
    {

    }

    private void button3_Click(object sender, EventArgs e)
    {
        Form form7 = new Form7();
        form7.Show();
    }
}

```

### 13. Форма «Принятие книг на склад»

```

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 LibraryBD
{
    public partial class Form13 : Form
    {
        public Form13()
        {
            InitializeComponent();
        }

        private void bookBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.bookBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.libraryDataSet);
        }
    }
}

```

```

    }

    private void Form13_Load(object sender, EventArgs e)
    {
        // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.ArrivalAtWarehouse".
        При необходимости она может быть перемещена или удалена.
        this.arrivalAtWarehouseTableAdapter.Fill(this.libraryDataSet.ArrivalAtWarehouse);
        // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Employee". При
        необходимости она может быть перемещена или удалена.
        this.employeeTableAdapter.Fill(this.libraryDataSet.Employee);
        // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
        необходимости она может быть перемещена или удалена.
        //this.bookTableAdapter.Fill(this.libraryDataSet.Book);
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
        Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command1 = new SqlCommand();
        command1.Connection = connection;
        command1.CommandType = System.Data.CommandType.Text;
        command1.CommandText = "SELECT Id_Book AS [Айди книги], Employee.PhoneNumber AS [Номер
        телефона], ArrivalAtWarehouse.Number AS [Количество], ArrivalAtWarehouse.DateOfArrival AS [Дата
        поступления] FROM ArrivalAtWarehouse INNER JOIN Employee ON ArrivalAtWarehouse.Id_Employee =
        Employee.Id_Employee";
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Id_Book AS [Айди книги], Book.Author AS [Автор книги],
        Book.Name AS [Название книги], Book.Genre AS [Жанр], Book.Number AS [Наличие] From Book";
        connection.Open();
        SqlDataAdapter adapter1 = new SqlDataAdapter(command1);
        DataTable dt1 = new DataTable();
        adapter1.Fill(dt1);
        dataGridView2.DataSource = dt1;
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        try

```

```

{
    if (string.IsNullOrEmpty(id_BookTextBox.Text) ||
string.IsNullOrEmpty(phoneNumberTextBox.Text) || string.IsNullOrEmpty(numberTextBox.Text))
    {
        MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
    }
    else
    {
        myCommand.Parameters["@id_book"].Value = id_BookTextBox.Text;
        myCommand.Parameters["@phone"].Value = phoneNumberTextBox.Text;
        myCommand.Parameters["@date"].Value = dateOfArrival.Text;
        myCommand.Parameters["@number"].Value = numberTextBox.Text;
        mySqlConnection.Open();
        myCommand.ExecuteNonQuery();
        mySqlConnection.Close();
        if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect date of arrival")
        {
            MessageBox.Show("Ошибка! Некорректная дата", "Окно уведомлений", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        }
        else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect number of books")
        {
            MessageBox.Show("Ошибка! Некорректное количество книг", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Employee fired")
        {
            MessageBox.Show("Ошибка! Сотрудник уволен", "Окно уведомлений", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        }
        else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Arrival succeeded")
        {
            MessageBox.Show("Поступление успешно добавлено", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect ID employee")
        {
            MessageBox.Show("Ошибка! Некорректный номер телефона", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect ID book")
        {

```

```

        MessageBox.Show("Ошибка! Такой книги не существует", "Окно уведомлений",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}

catch
{
    MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
}

string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";

SqlConnection connection = new SqlConnection(connString);
SqlCommand command = new SqlCommand();
command.Connection = connection;
command.CommandType = System.Data.CommandType.Text;
command.CommandText = "SELECT Id_Book AS [Айди книги], Employee.PhoneNumber AS [Номер
телефона], ArrivalAtWarehouse.Number AS [Количество] FROM ArrivalAtWarehouse INNER JOIN Employee ON
ArrivalAtWarehouse.Id_Employee = Employee.Id_Employee";

connection.Open();
SqlDataAdapter adapter = new SqlDataAdapter(command);
DataTable dt = new DataTable();
adapter.Fill(dt);
dataGridView2.DataSource = dt;
connection.Close();
}

private void button1_Click(object sender, EventArgs e)
{
    if (!(string.IsNullOrEmpty(authorTextBox.Text)) && !(string.IsNullOrEmpty(nameTextBox.Text)))
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";

        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Id_Book AS [Айди книги], Author AS [Автор], Name AS
[Название книги], Genre AS [Жанр], Number AS [Наличие] FROM Book WHERE Author LIKE '%" +
authorTextBox.Text + "%' AND Name LIKE '%" + nameTextBox.Text + "%'";

        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
    }
}

```

```

        dataGridView1.DataSource = dt;
        connection.Close();
    }
    else if (!(string.IsNullOrEmpty(authorTextBox.Text)) &&
string.IsNullOrEmpty(nameTextBox.Text))
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Id_Book AS [Айди книги], Author AS [Автор], Name AS
[Название книги], Genre AS [Жанр], Number AS [Наличие] FROM Book WHERE Author LIKE '%" +
authorTextBox.Text + "%'";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }
    else if (string.IsNullOrEmpty(authorTextBox.Text) &&
!(string.IsNullOrEmpty(nameTextBox.Text)))
    {
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Id_Book AS [Айди книги], Author AS [Автор], Name AS
[Название книги], Genre AS [Жанр], Number AS [Наличие] FROM Book WHERE Name LIKE '%" +
nameTextBox.Text + "%'";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }
    else if (string.IsNullOrEmpty(authorTextBox.Text) && string.IsNullOrEmpty(nameTextBox.Text))
    {

```

```

        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";

        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Id_Book AS [Айди книги], Author AS [Автор], Name AS
[Название книги], Genre AS [Жанр], Number AS [Наличие] FROM Book";

        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();

    }
}

private void dataGridView2_CellContentClick(object sender, DataGridViewCellEventArgs e)
{

}
}
}

```

## 14. Форма «Добавление книги»

```

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 LibraryBD
{
    public partial class Form14 : Form
    {
        public Form14()
        {
            InitializeComponent();

            private void bookBindingNavigatorSaveItem_Click(object sender, EventArgs e)
            {
                this.Validate();
                this.bookBindingSource.EndEdit();
                this.tableAdapterManager.UpdateAll(this.libraryDataSet);
            }
        }
    }
}

```



```

    }

    private void Form1_Load(object sender, EventArgs e)
    {
        // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Book". При
        // необходимости она может быть перемещена или удалена.
        //this.bookTableAdapter.Fill(this.libraryDataSet.Book);
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
        Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Author AS [Автор книги], Book.Name AS [Название книги],
        Book.Genre AS [Жанр], Book.ISBN AS [ISBN], Book.UDK AS [UDK], Book.Number AS [Наличие] From Book";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        try
        {
            if (string.IsNullOrEmpty(authorTextBox.Text) || string.IsNullOrEmpty(nameTextBox.Text) ||
            string.IsNullOrEmpty(genreTextBox.Text) || string.IsNullOrEmpty(isbnTextBox.Text) ||
            string.IsNullOrEmpty(udkTextBox.Text))
            {
                MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
            }
            else
            {
                myCommand.Parameters["@author"].Value = authorTextBox.Text;
                myCommand.Parameters["@name"].Value = nameTextBox.Text;
                myCommand.Parameters["@genre"].Value = genreTextBox.Text;
                myCommand.Parameters["@ISBN"].Value = isbnTextBox.Text;
                myCommand.Parameters["@UDK"].Value = udkTextBox.Text;

                mySqlConnection.Open();
                myCommand.ExecuteNonQuery();
                mySqlConnection.Close();
                if (Convert.ToString(myCommand.Parameters["@message"].Value) == "The book already exists")
                {
                    MessageBox.Show("Ошибка! Книга уже добавлена!", "Окно уведомлений",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
                else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Book added")
                {
                    MessageBox.Show("Книга успешно добавлена", "Окно уведомлений", MessageBoxButtons.OK,
                    MessageBoxIcon.Information);
                }
            }
        }
        catch
        {
            MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
        Security=True";
    }

```

```

        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Book.Author AS [Автор книги], Book.Name AS [Название книги],
Book.Genre AS [Жанр], Book.ISBN AS [ISBN], Book.UDK AS [UDK], Book.Number AS [Наличие] From Book";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {
    }

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

```

## 15. Форма «Добавление сотрудника»

```

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 LibraryBD
{
    public partial class Form15 : Form
    {
        public Form15()
        {
            InitializeComponent();
        }

        private void employeeBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.employeeBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.libraryDataSet);
        }

        private void Form15_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Employee". При
            // необходимости она может быть перемещена или удалена.
            this.employeeTableAdapter.Fill(this.libraryDataSet.Employee);
            string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
            Security=True";
            SqlConnection connection = new SqlConnection(connString);

```

```

SqlCommand command = new SqlCommand();
command.Connection = connection;
command.CommandType = System.Data.CommandType.Text;
command.CommandText = "SELECT Employee.FIO AS [ФИО], Employee.Position AS [Должность],
Employee.PhoneNumber AS [Номер телефона], Employee.PassportNumber AS [Паспорт] FROM Employee";
connection.Open();
SqlDataAdapter adapter = new SqlDataAdapter(command);
DataTable dt = new DataTable();
adapter.Fill(dt);
dataGridView1.DataSource = dt;
connection.Close();
}

private void button1_Click(object sender, EventArgs e)
{
    try
    {
        if (string.IsNullOrEmpty(fioTextBox.Text) || string.IsNullOrEmpty(positionComboBox.Text) ||
string.IsNullOrEmpty(phoneNumberTextBox.Text) ||
string.IsNullOrEmpty(passportNumberTextBox.Text))
        {
            MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
        }
        else
        {
            myCommand.Parameters["@fio"].Value = fioTextBox.Text;
            myCommand.Parameters["@position"].Value = positionComboBox.Text;
            myCommand.Parameters["@phone"].Value = phoneNumberTextBox.Text;
            myCommand.Parameters["@passport"].Value = passportNumberTextBox.Text;
            mySqlConnection.Open();
            myCommand.ExecuteNonQuery();
            mySqlConnection.Close();
            if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Number already exists")
            {
                MessageBox.Show("Ошибка! Сотрудник с таким телефоном уже существует!", "Окно
уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "no fired position")
            {
                MessageBox.Show("Ошибка! Некорректная должность ", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Passport already exists")
            {
                MessageBox.Show("Ошибка! Сотрудник с таким паспортом уже существует", "Окно
уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Incorrect position")
            {
                MessageBox.Show("Ошибка! Некорректная должность", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "The employee added")
            {
                MessageBox.Show("Сотрудник успешно добавлен", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch
    {
        MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

```

        string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Employee.FIO AS [ФИО], Employee.Position AS [Должность],
Employee.PhoneNumber AS [Номер телефона], Employee.PassportNumber AS [Паспорт] FROM Employee";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }

    private void mySqlConnection_InfoMessage(object sender, SqlInfoMessageEventArgs e)
    {
    }
}
}

```

## 16. Форма «Главный библиотекарь»

```

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 LibraryBD
{
    public partial class Form16 : Form
    {
        public Form16()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Form form15 = new Form15();
            form15.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            Form form17 = new Form17();
            form17.Show();
        }

        private void Form16_Load(object sender, EventArgs e)
        {
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form form8 = new Form8();
            form8.Show();
        }
    }
}

```

```

    }

    private void button4_Click(object sender, EventArgs e)
    {
        Form form7 = new Form7();
        form7.Show();
    }
}

```

## Форма «Увольнение сотрудника»

```

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 LibraryBD
{
    public partial class Form17 : Form
    {
        public Form17()
        {
            InitializeComponent();
        }

        private void employeeBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.employeeBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.libraryDataSet);
        }

        private void Form17_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Employee". При
            // необходимости она может быть перемещена или удалена.
            this.employeeTableAdapter.Fill(this.libraryDataSet.Employee);
            string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
            Security=True";
            SqlConnection connection = new SqlConnection(connString);
            SqlCommand command = new SqlCommand();
            command.Connection = connection;
            command.CommandType = System.Data.CommandType.Text;
            command.CommandText = "SELECT Employee.FIO AS [ФИО], Employee.Position AS [Должность],
            Employee.PhoneNumber AS [Номер телефона], Employee.PassportNumber AS [Паспорт] FROM Employee";
            connection.Open();
            SqlDataAdapter adapter = new SqlDataAdapter(command);
            DataTable dt = new DataTable();
            adapter.Fill(dt);
            dataGridView1.DataSource = dt;
            connection.Close();
        }

        private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {

```

```

    }

    private void button1_Click(object sender, EventArgs e)
    {
        try
        {
            if (string.IsNullOrEmpty(passportNumberTextBox.Text))
            {
                MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
            }
            else
            {
                myCommand.Parameters["@passport"].Value = passportNumberTextBox.Text;
                mySqlConnection.Open();
                myCommand.ExecuteNonQuery();
                mySqlConnection.Close();
                if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Employee fired")
                {
                    MessageBox.Show("Сотрудник уволен.", "Окно уведомлений", MessageBoxButtons.OK,
                    MessageBoxIcon.Information);
                }
                else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Employee already fired")
                {
                    MessageBox.Show("Ошибка! Сотрудник уже уволен!", "Окно уведомлений",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
                else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Employee does not exist")
                {
                    MessageBox.Show("Ошибка! Сотрудника не существует!", "Окно уведомлений",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
            }
        }
        catch
        {
            MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        this.employeeTableAdapter.Fill(this.libraryDataSet.Employee);
        string connString = @"Data Source=ANCHAЕVPC\SQLALDAR;Initial Catalog=Library;Integrated
        Security=True";
        SqlConnection connection = new SqlConnection(connString);
        SqlCommand command = new SqlCommand();
        command.Connection = connection;
        command.CommandType = System.Data.CommandType.Text;
        command.CommandText = "SELECT Employee.FIO AS [ФИО], Employee.Position AS [Должность],
        Employee.PhoneNumber AS [Номер телефона], Employee.PassportNumber AS [Паспорт] FROM Employee";
        connection.Open();
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        dataGridView1.DataSource = dt;
        connection.Close();
    }
}

```

## Форма «Изменить данные читателя»

```

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 LibraryBD
{
    public partial class Form18 : Form
    {
        public Form18()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            try
            {
                if (string.IsNullOrEmpty(id_ReaderTextBox.Text) ||
string.IsNullOrEmpty(phoneNumberTextBox1.Text))
                {
                    MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
                }
                else
                {
                    myCommand.Parameters["@Id_Reader"].Value = id_ReaderTextBox.Text;
                    myCommand.Parameters["@newPhone"].Value = phoneNumberTextBox1.Text;
                    mySqlConnection.Open();
                    myCommand.ExecuteNonQuery();
                    mySqlConnection.Close();
                    if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Phone number is already use")
                    {
                        MessageBox.Show("Ошибка! Читатель с таким номером телефона уже существует!", "Окно
уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    }
                    else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader not found")
                    {
                        MessageBox.Show("Ошибка! Читателя с таким ID не существует!", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                    }
                    else if (Convert.ToString(myCommand.Parameters["@message"].Value) == "Reader updated")
                    {
                        MessageBox.Show("Номер телефона успешно изменен", "Окно уведомлений",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                    }
                }
                string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
Security=True";
                SqlConnection connection = new SqlConnection(connString);
                SqlCommand command = new SqlCommand();
                command.Connection = connection;
                command.CommandType = System.Data.CommandType.Text;
                command.CommandText = "SELECT Reader.Id_Reader AS [ID], Reader.FIO AS [Читатель],
Reader.PhoneNumber AS [Номер телефона], Reader.Birthday AS [Дата рождения] FROM Reader";
                connection.Open();
                SqlDataAdapter adapter = new SqlDataAdapter(command);
                DataTable dt = new DataTable();
                adapter.Fill(dt);
                dataGridView1.DataSource = dt;
                connection.Close();
            }

            catch
            {
                MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
}

```

```

    }
}

private void button1_Click(object sender, EventArgs e)
{
    try
    {
        if (string.IsNullOrEmpty(phoneNumberTextBox.Text))
        {
            MessageBox.Show("Пустое поле", "Окно уведомлений", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
        }
        else
        {
            string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
            Security=True";
            SqlConnection connection = new SqlConnection(connString);
            SqlCommand command = new SqlCommand();
            command.Connection = connection;
            command.CommandType = System.Data.CommandType.Text;
            command.CommandText = "SELECT Reader.Id_Reader AS [ID], Reader.FIO AS [Читатель],
            Reader.PhoneNumber AS [Номер телефона], Reader.Birthday AS [Дата рождения] FROM Reader WHERE
            PhoneNumber LIKE '%" + phoneNumberTextBox.Text + "%'";
            connection.Open();
            SqlDataAdapter adapter = new SqlDataAdapter(command);
            DataTable dt = new DataTable();
            adapter.Fill(dt);
            dataGridView1.DataSource = dt;
            connection.Close();
        }
    }
    catch
    {
        MessageBox.Show("Ошибка", "Окно уведомлений", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

private void readerBindingNavigatorSaveItem_Click(object sender, EventArgs e)
{
    this.Validate();
    this.readerBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(this.libraryDataSet);
}

private void Form18_Load(object sender, EventArgs e)
{
    // TODO: данная строка кода позволяет загрузить данные в таблицу "libraryDataSet.Reader". При
    // необходимости она может быть перемещена или удалена.
    this.readerTableAdapter.Fill(this.libraryDataSet.Reader);
    string connString = @"Data Source=ANCHAEVPC\SQLALDAR;Initial Catalog=Library;Integrated
    Security=True";
    SqlConnection connection = new SqlConnection(connString);
    SqlCommand command = new SqlCommand();
    command.Connection = connection;
    command.CommandType = System.Data.CommandType.Text;
    command.CommandText = "SELECT Reader.Id_Reader AS [ID], Reader.FIO AS [Читатель],
    Reader.PhoneNumber AS [Номер телефона], Reader.Birthday AS [Дата рождения] FROM Reader";
    connection.Open();
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    DataTable dt = new DataTable();
    adapter.Fill(dt);
    dataGridView1.DataSource = dt;
    connection.Close();
}

```



}  
}