МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гродненский государственный университет имени Янки Купалы»

Факультет математики и информатики

**Кафедра современных технологий программирования**

ЮРАШЕВИЧ ЭЛЬВИРА НИКОЛАЕВНА

**Проектирование информационной системы**   
**«Домашняя библиотека»**

Курсовая работа по дисциплине

«Системы баз данных»

студентки 3 курса специальности

1-26 03 01 «Управление информационными ресурсами» дневной формы получения образования

Научный руководитель  
Гуща Юлия Вальдемаровна, старший преподаватель   
кафедры современных технологий программирования

Гродно, 2023

**РЕЗЮМЕ**

Тема курсовой работы

**Проектирование информационной системы «Домашняя библиотека»**

Работа содержит: 36 страниц, 12 рисунков, 32 таблицы, 5 использованных источников.

Ключевые слова: информационная система, домашняя библиотека, база данных, MySQL, реляционная модель.

**Цель курсовой работы** — проектирование и разработка информационной системы для учета книг в домашней библиотеке.

Объектом исследования выступает информационная система «Домашняя библиотека».

Предмет исследования является процесс проектирования и разработки информационной системы «Домашняя библиотека».

Информационная система и приложение «Домашняя библиотека» позволяет отслеживать новые поступления книг в библиотеке пользователя, а также ее передвижение по другим пользователям.

# СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc136360680)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6](#_Toc136360681)

[1.1 Описание предметной области 6](#_Toc136360682)

[1.2 Используемые программные средства 7](#_Toc136360683)

[1.2.1 Power Designer 7](#_Toc136360684)

[1.2.2 MySQL 9](#_Toc136360685)

[1.2.3 C# 10](#_Toc136360686)

[2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ 12](#_Toc136360688)

[2.1 Концептуальная модель базы данных 12](#_Toc136360689)

[2.2 Логическая модель базы данных 14](#_Toc136360690)

[2.3 Физическая модель базы данных 15](#_Toc136360691)

[2.4 Нормализация отношений 17](#_Toc136360692)

[2.5 Сущности и взаимосвязи 23](#_Toc136360693)

[3 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ 28](#_Toc136360694)

[ЗАКЛЮЧЕНИЕ 35](#_Toc136360695)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc136360696)

# ВВЕДЕНИЕ

Ежедневно люди собирают и обрабатывают огромное количество информации. Это касается не только работы в организациях, но и обычной жизни.

Коллекции собранных данных хранят в себе большие потенциальные возможности по извлечению новой аналитической информации, с помощью которой можно строить тактику организации, выявлять склонности улучшения рынка, находить новые решения, обусловливающие успешное становление в условиях конкурентной борьбы.

Основные идеи современных информационных технологий базируются на концепции, согласно которой данные должны быть организованы в базы данных с целью адекватного отображения изменяющегося реального мира и удовлетворения информационных потребностей пользователей.

Базы данных — это совокупность структур, предназначенных для хранения больших объемов информации и программных модулей, осуществляющих управление данными, их выборку, сортировку и другие подобные действия.

Информация базы данных хранится в одной или нескольких таблицах. Любая таблица с данными состоит из набора однотипных записей, расположенных друг за другом. Они представляют собой строки таблицы, которые можно добавлять, удалять или изменять.

Каждая запись является набором именованных полей, или ячеек, которые могут хранить самую разнообразную информацию, начиная от даты рождения и заканчивая подробным описанием кулинарного рецепта. Однотипные поля разных записей образуют столбец таблицы.

Для хранения, упорядочения и анализа больших объемов информации предназначены комплексы средств, именуемых информационными системами (ИС). Одними из видов таких информационных систем и являются базы данных, управляемые с помощью систем управления базами данных (СУБД), и информационно-аналитические системы, предназначенные как для хранения, так и для анализа хранимой информации.

Современные автоматизированные информационные системы:

1. ориентируются на конкретного пользователя, предоставляя ему удобный интерфейс и необходимые для работы функции;
2. должны обеспечивать надежность и продолжительность хранения данных, обладающих разными структурами;
3. должны обладать функциями, обеспечивающими ввод, обновление и удаление данных;
4. должны предоставлять данные для коллективной работы, обеспечивая согласованность их действий.

Целью курсовой работы является закрепление теоретических и практических навыков по разработке и работе с базами данных в процессе построения собственной базы данных «Домашняя библиотека» на основе реляционной модели данных.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## 1.1 Описание предметной области

Создание базы данных начинается с изучения и анализа предметной области. Анализ предметной области позволяет выделить ее сущности, определить первоначальные требования к функциональности и границы проекта. В нашем случае необходимо создать базу данных для книг, хранящихся в домашней библиотеке пользователя.

Система «Домашняя библиотека» используется для экономии времени и облегчения процесса отслеживания книг, различных серийных изданий и другой литературы в личной библиотеке. Также она позволяет хранить данные о пользователях библиотеки, а именно: кто взял книгу, когда взял, когда вернул и в каком состоянии.

Учитывая это, становится понятно, что владелец библиотеки также сможет:

1. Записывать и удалять пользователей в базу,
2. Добавлять и удалять книги из базы,
3. Добавлять новые жанры для книг, если необходимых не было в стандартном списке,
4. Добавлять новых авторов для книг, если необходимых не было в стандартном списке,
5. Добавлять новых авторов для книг, если необходимых не было в списке.
6. Добавлять новые издательства для книг, если необходимых не было в списке.

Каждое издание обладает такими характеристиками как: ее название, автор произведения, год издания, издательство, жанр произведения (самые разнообразные жанры, но основными являются: детектив, фантастика, фэнтези, драма, романтика, комедия, триллер, ужасы, экшен), вид обложки (мягкая, твердая, интегральная), находится ли книга в библиотеке или ее кто-то взял, и, наконец, количество страниц.

Следующий тип книг – зарубежные книги. Обладает всеми основными характеристиками книг, но при этом имеет свои собственные, такие как: язык оригинала, на котором была изначально написана книга, лицензиат.

Еще один тип – Иллюстрированная литература. Имеет основные характеристики книги, но также обладает своими: иллюстратор; в каком виде представлены иллюстрации, а именно: цветные или черно-белые.

Следует учесть, что названия книг могут совпадать, однако они различаются своим уникальным учетным номером, однозначно характеризующим каждый экземпляр книги.

Важно уточнить, что книга может находиться либо в свободном доступе в библиотеке, либо у одного пользователя, который ее взял.

Необходимо описать работу системы так, чтобы пользователь не мог взять уже занятую кем-то книгу.

Еще одним важным моментом является дата выдачи и дата возврата. Дата возврата всегда должна быть позднее даты выдачи.

О каждом пользователе, который воспользовался (либо пользуется периодически) библиотекой, имеются следующие данные: фамилия, имя, отчество и небольшие заметки о том, как пользователь обращается с книгой.

Для учета движения книг между членами семьи учитывается следующая информация: дата выдачи издания пользователю, дата возврата издания обратно в библиотеку, при этом можно оставить заметки о том, в каком состоянии пользователь вернул книгу.

Итак, целью создания автоматизированной системы «Частная библиотека» должен стать программный продукт, удовлетворяющий перечисленным ранее требованиям, а также реализованный с использованием соответствующих СУБД и программного обеспечения.

## 1.2 Используемые программные средства

**1.2.1 Power Designer**

PowerDesigner — это программный продукт, предоставляющий платформу для моделирования данных, процессов и архитектуры. Это помогает организациям понимать и визуализировать свои бизнес-процессы, разрабатывать модели данных и создавать архитектуры приложений. PowerDesigner включает в себя интегрированный набор инструментов, которые позволяют пользователям проектировать, документировать и управлять сложными корпоративными системами в простой в использовании среде. Его можно использовать для проектирования баз данных, объектного моделирования, управления бизнес-процессами, разработки приложений и многого другого. PowerDesigner доступен как для операционных систем Windows, так и для Mac.

PowerDesigner дает возможность управления изменениями на этапе проектирования, предлагает технику управления метаданными и содержит уникальную технологию анализа взаимосвязей моделей. Одновременно с поддержкой ведущих техник моделирования и управления метаданными, PowerDesigner также позволяет работать с моделями любых типов в единой интегрированной среде, а репозиторий метаданных PowerDesigner помогает наладить взаимодействие между всеми заинтересованными лицами компании, что обеспечивает более быстрый отклик на изменения в существующей бизнес-среде.

В SAP PowerDesigner можно сформировать различные модели – физическую, логическую и концептуальную. Главным преимуществом является то, что создав концептуальную модель, с помощью инструментов программы можно автоматически сформировать физическую и логическую модели.

Концептуальная модель в SAP PowerDesigner — это графическое представление системы, состоящей из сущностей и их взаимосвязей. Он используется для визуального представления компонентов системы, таких как ее объекты, процессы, хранилища данных и взаимодействия между различными компонентами. Модель можно использовать для анализа системы и выявления проблем проектирования или возможностей для улучшения. Его также можно использовать для документирования дизайнерских решений, принятых в процессе разработки.

Логическая модель в SAP PowerDesigner представляет собой представление логической структуры данных, хранящихся в базе данных. Он обеспечивает общий обзор организации данных и их взаимосвязей, включая сущности и их атрибуты, а также взаимосвязи между сущностями. Модель можно использовать для создания физических проектов баз данных, а также для помощи в проектировании и разработке приложений, которые получают доступ к данным и управляют ими.

Физическая модель в SAP PowerDesigner — это представление существующей системы, включая ее объекты, отношения и атрибуты. Он также может представлять собой проект новой системы, которая будет реализована с использованием определенной технологии базы данных. Физическая модель содержит объекты, представляющие фактические физические элементы системы, такие как таблицы, столбцы, первичные и внешние ключи и индексы. Физические модели используются для создания кода, специфичного для базы данных, который можно использовать для создания базы данных.

Модель строится в виде графической схемы, называемой диаграммой. Диаграмма кликабельна, можно посмотреть подробно любую таблицу, поле, связь.

**1.2.2 MySQL**

MySQL — это система управления реляционными базами данных (RDBMS) с открытым исходным кодом, разработанная Oracle. MySQL — одна из самых популярных баз данных в мире, используемая многими крупными и малыми предприятиями, а также отдельными веб-разработчиками. MySQL основан на языке структурированных запросов (SQL), который является наиболее распространенным языком для управления базами данных. Он поддерживает хранение, поиск и манипулирование данными как из структурированных, так и из неструктурированных источников. MySQL — это мощное и универсальное решение, которое можно использовать для создания приложений, начиная от небольших веб-сайтов и заканчивая крупными корпоративными приложениями.

MySQL — это надежный многопоточный сервер баз данных, отличающийся впечатляющей скоростью и простотой использования. Он известен своей надежностью, стабильностью и удобством использования.

MySQL — идеальный выбор для приложений малого и среднего размера. Он может быть скомпилирован на различных платформах и лучше всего работает на серверах Unix, где включена многопоточность, что помогает повысить его производительность.

MySQL поддерживает язык запросов SQL в стандарте ANSI 92, и, кроме этого, имеет множество расширений к этому стандарту, которых нет ни в одной другой СУБД.

Краткий перечень возможностей MySQL.

1. Поддерживается неограниченное количество пользователей, одновременно работающих с базой данных.
2. Количество строк в таблицах может достигать 50 млн.
3. Быстрое выполнение команд. Возможно, MySQL самый быстрый сервер из существующих.
4. Простая и эффективная система безопасности.

Также следует отметить, что MySQL - система с открытым исходным кодом: любой желающий имеет возможность использовать и модифицировать это программное обеспечение по своему усмотрению. Каждый пользователь имеет право получить данное программное обеспечение посредством сети Интернет бесплатно. Программное обеспечение MySQL распространяется по лицензии GPL (GNU General Public License), которая регламентирует, что разрешено, а что нет в отношении программного обеспечения.

**1.2.3 C#**

Исходя из требований к приложению, имеющихся навыков и доступных средств, для реализации приложения был выбран язык C#.

C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

В результате язык получился крайне гибкий, мощный и универсальный. На нем пишут практически все, что угодно, от небольших веб-приложений до мощных программных систем, объединяющих в себе веб-структуры, приложения для десктопов и мобильных устройств. Все это стало возможным благодаря удобному Си-подобному синтаксису, строгому структурированию, огромному количеству фреймворков и библиотек (их число достигает нескольких сотен).

C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов. С момента создания язык C# обогатился функциями для поддержки новых рабочих нагрузок и современными рекомендациями по разработке ПО.

C# является объектно-ориентированным и в этом плане много перенял у Java и С++. Например, C# поддерживает полиморфизм, наследование, перегрузку операторов, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. И C# продолжает активно развиваться, и с каждой новой версией появляется все больше интересного функционала.

В результате проведенного анализа и того факта, что некоторые навыки по написанию программ на языке программирования C# у нас уже имеются, можно сделать вывод о том, что наиболее подходящим языком программирования является С#.

В качестве среды разработки была выбрана Microsoft Visual Studio 2019. Microsoft Visual Studio 2019 является мощным инструментом для разработки программного обеспечения в основном на C# и C++ и их Фреймворках. Большим плюсом является наличие студенческой лицензии, что позволяет бесплатно пользоваться данной средой разработки.

Для разработки приложения было выбрано решение использовать Windows Forms. Библиотека Windows Forms доступна в .NET Framework, кроссплатформенной среде с открытым исходным кодом. Это означает, что вы можете использовать C для написания приложений Windows Forms, которые будут работать как на Windows, так и на других платформах.

# 2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ

## 2.1 Концептуальная модель базы данных

Концептуальная модель базы данных – это высокоуровневое представление о том, как данные взаимодействуют друг с другом и как они организованы в базе данных. Она описывает сущности, их атрибуты и отношения между ними, но не уточняет детали реализации.

Концептуальная модель базы данных является первым этапом проектирования базы данных, на котором определяются основные объекты и связи между ними. Она создается на основе требований к системе и бизнес-процессов, которые должна поддерживать база данных.

Примером концептуальной модели может служить ER-диаграмма, которая показывает сущности, их атрибуты и отношения между ними в виде графической схемы. На этом этапе не учитывается, как данные будут храниться и обрабатываться в базе данных, а только определяется их логическая структура и связи между ними.

Концептуальная модель базы данных является важным инструментом для понимания бизнес-процессов и требований к системе, а также для обеспечения правильного проектирования базы данных на следующих этапах.

Правила разработки концептуальной модели базы данных.

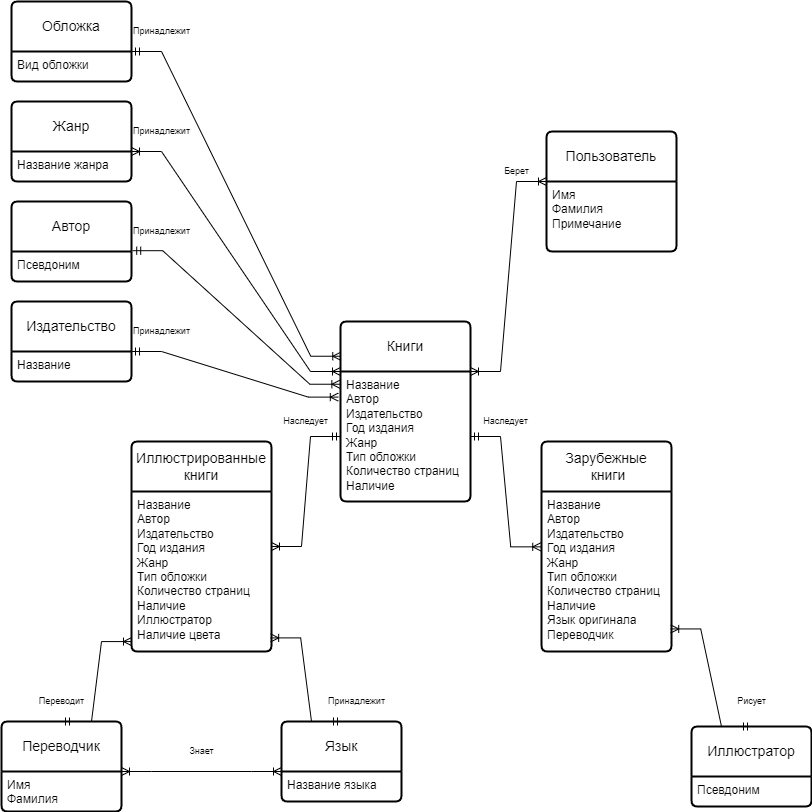
1. Определить сущности (таблицы), которые будут частью базы данных.
2. Определить атрибуты (столбцы) каждой сущности.
3. Определить отношения между сущностями.
4. Нормализовать данные, чтобы уменьшить избыточность данных.
5. Определить любые ограничения, которые необходимо применить к данным.
6. Определить, как пользователи и приложения будут получать доступ к данным и манипулировать ими.
7. Разработать физическую структуру базы данных на основе концептуальной модели

Для разработки схемы концептуальной модели базы данных для информационной системы «Домашняя библиотека» использовалась нотация «Crow’s foot»

Нотация "Crow's foot" (в переводе с английского "воронья лапка") – это одна из наиболее распространенных нотаций для проектирования баз данных. Она используется для создания ER-диаграмм (Entity-Relationship Diagrams), которые позволяют описывать сущности (entities) в предметной области, их атрибуты (attributes) и связи между ними.

Нотация "Crow's Foot" получила свое название из-за визуального сходства ее символов с лапами ворон. В этой нотации символ "вороньей лапки" используется для обозначения кардинальности (cardinality) связи между двумя сущностями. Например, одна "воронья лапка" означает, что между двумя сущностями существует один к одному (1:1) отношение, а три "вороньи лапки" ­– что между ними существует один ко многим (1:N) отношение.

Нотация "Crow's Foot" является стандартом в индустрии и широко используется в инструментах для проектирования и управления базами данных, таких как Microsoft Visio, ERwin и другие (Рисунок 2.1).



**Рисунок 2.1** – Схема концептуальной модели данных

Примечание **–**  Источник: собственная разработка

## 2.2 Логическая модель базы данных

Логическая модель базы данных – это описание структуры и связей между данными в базе данных. Она определяет, какие данные будут храниться в базе данных, как они будут организованы и как они будут связаны друг с другом.

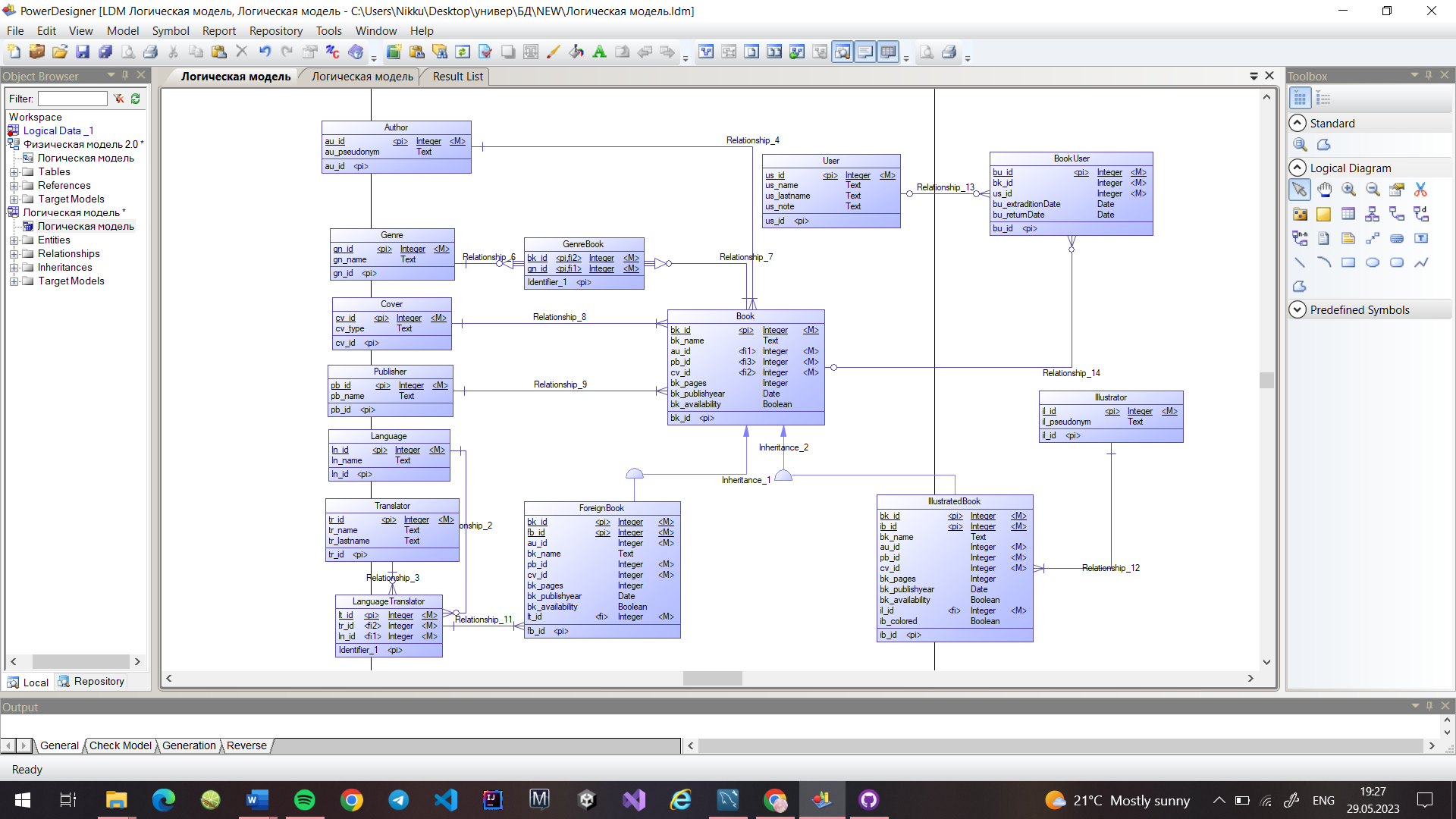
Логическая модель базы данных обычно представляется в виде диаграммы, которая показывает таблицы и связи между ними. Каждая таблица представляет собой набор записей, содержащих информацию об объектах или сущностях, которые хранятся в базе данных. Связи между таблицами определяют, как эти объекты связаны друг с другом.

Создание логической модели базы данных – это важный этап в проектировании базы данных, который помогает определить требования к данным и обеспечить эффективное хранение и доступ к данным.

Существует несколько правил, которые следует учитывать при создании логической модели базы данных. Некоторые из них включают:

1. Определение сущностей и их атрибутов. Сущности – это объекты, которые вы хотите хранить в базе данных, а их атрибуты – это свойства, которые вы хотите сохранить для каждой сущности.
2. Использование правильных типов данных для каждого атрибута. Например, если вы хотите сохранить дату, вам следует использовать тип данных "дата", а не "текст".
3. Определение отношений между сущностями. Например, если у вас есть таблица клиентов и таблица заказов, то вы можете установить отношение "один ко многим" между ними, так как каждый клиент может иметь несколько заказов, но каждый заказ относится только к одному клиенту.
4. Нормализация базы данных. Это процесс разделения больших таблиц на более мелкие таблицы, чтобы избежать повторения данных и обеспечить более эффективное хранение и поиск информации.
5. Использование правильных инструментов для создания логической модели базы данных. Существует множество инструментов, которые могут помочь вам создать логическую модель, такие как ER-диаграммы и UML-диаграммы.

Для построения схемы логической модели базы данных использовалось программное решение SAP PowerDesigner (Рисунок 2.2).



**Рисунок 2.2** – Схема логической модели данных

Примечание **–**  Источник: собственная разработка

## 2.3 Физическая модель базы данных

Физическая модель базы данных – это модель, которая описывает конкретную реализацию базы данных в определенной СУБД. Схема базы данных состоит из подробной информации об отдельных таблицах, полях, столбцах, индексах и других реализованных ограничениях целостности данных. Это также включает правила, касающиеся того, как данные должны храниться и манипулироваться в базе данных. Физическая модель базы данных является конкретной реализацией логической модели базы данных, которая описывает структуру данных и их отношения.

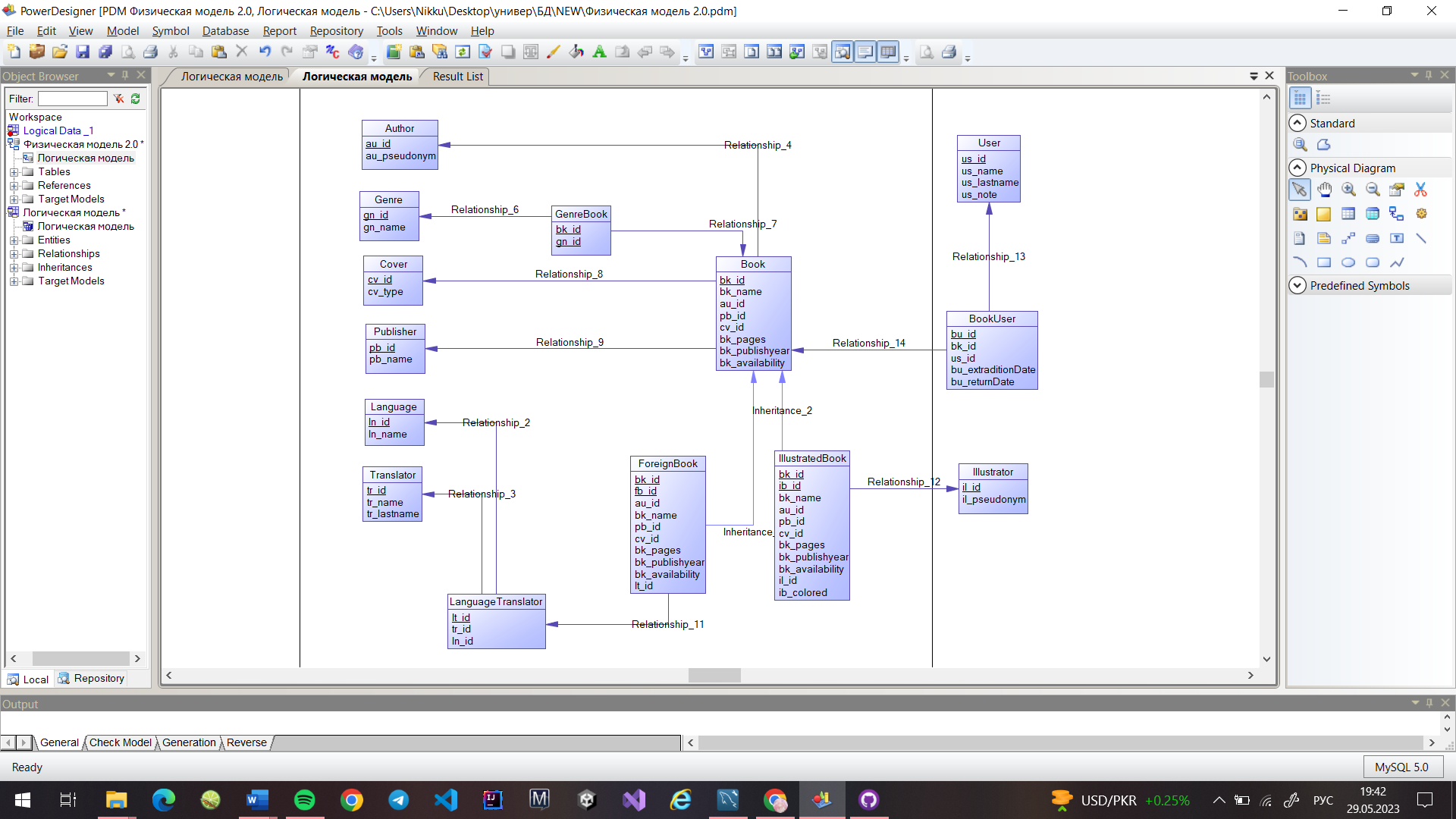
Физическая модель базы данных – это определение базы данных состоит из сведений о различных типах данных, соответствующих им размерах полей, ключах, индексах и других подобных атрибутах, определяющих хранение и обработку данных. Кроме того, он также обеспечивает точность и согласованность, вводя определенные ограничения целостности данных.

Неотъемлемой частью процесса проектирования базы данных является настройка физической модели базы данных. Это позволяет выбрать эффективную структуру базы данных, отвечающую таким требованиям, как производительность, масштабируемость и безопасность данных.

Правила разработки физической модели базы данных могут варьироваться в зависимости от конкретной СУБД и проекта, но в целом следующие правила могут помочь обеспечить эффективность и надежность физической модели базы данных:

1. Использовать правильные типы данных: выбрать наиболее подходящие типы данных для каждого столбца в таблице. Например, использовать тип данных "целое число" для столбца, который содержит целочисленные значения, а не тип данных "текст".
2. Использовать индексы: создать индексы для столбцов, которые будут использоваться в качестве ключей или для поиска данных. Индексы ускоряют выполнение запросов и повышают производительность базы данных.
3. Использовать ограничения целостности данных: определить ограничения целостности данных, чтобы гарантировать правильность и согласованность данных. Например, определить ограничение, которое запрещает вставку дубликатов в столбец, который должен содержать уникальные значения.
4. Нормализовать таблицы: разбить таблицы на отдельные таблицы, чтобы избежать повторения данных и обеспечить согласованность. Нормализация также упрощает обновление и изменение данных.
5. Учитывать производительность: учитывать производительность базы данных при разработке физической модели. Например, избегать создания слишком больших таблиц, которые будут занимать много места на диске и медленно выполняться.
6. Следовать стандартам и правилам СУБД: следовать стандартам и правилам конкретной СУБД, которая используется для разработки физической модели базы данных. Некоторые СУБД могут иметь свои собственные правила и ограничения, которые необходимо учитывать при разработке физической модели.

Для разработки схемы физической модели базы данных использовалось программное решение SAP PowerDesigner (Рисунок 2.3).



**Рисунок 2.3** – Схема физической модели данных

Примечание **–**  Источник: собственная разработка

## 2.4 Нормализация отношений

Система «Домашняя библиотека» используется для отслеживания наличия книг, различных серийных изданий и другой литературы в личной библиотеке. Также она позволяет хранить данные о пользователях библиотеки, в частности: кто, как, когда взял книгу и в каком состоянии вернул.

Приведем примерную таблицу к первой нормальной форме.

Таблица 2.1 **–** Первая нормальная форма

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID пользователя | Имя | Фамилия | Название книги | Автор | Дата выдачи | Дата возврата |

Окончание таблицы 2.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Артем | Василевский | «Котенок по имени Гав» | Г.Б. Остер | 01.01.2001 | 01.02.2001 |
| 1 | Артем | Василевский | «Щеночек по имени Мяу» | О.Г. Бостер | 04.05.2006 | 04.05.2023 |
| 2 | Марина | Жук | «Гадкий утенок» | Ганс Христиан Андерсен | 13.07.2003 | 15.07.2003 |
| 3 | Ирина | Вилеевич | «Лягушка путешественница» | В.М Гаршин | 26.06.2006 | 11.07.2006 |
| 4 | Никита | Легостаев | «Щеночек по имени Мяу» | О.Г. Бостер | 04.05.2006 | 04.05.2023 |
| 5 | Ян | Пожарский | «Муму» | И.С. Тургенев | 19.04.2020 | 19.06.2021 |
| 5 | Ян | Пожарский | «Снежная королева» | Ганс Христиан Андерсен | 26.04.2023 | 26.04.2023 |

Примечание **–**  Источник: собственная разработка

Обеспечим уникальность записей введя в таблицу ключевое поле – «Номер учетной записи».

Приведем примерную таблицу ко второй нормальной форме.

Таблица 2.2 **–** Вторая нормальная форма

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | ID польз. | Имя | Фамилия | Название книги | Автор | Дата выдачи | Дата возврата |
| 0001 | 1 | Артем | Василевский | «Котенок по имени Гав» | Г.Б. Остер | 01.01.2001 | 01.02.2001 |

Окончание таблицы 2.2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0002 | 1 | Артем | Василевский | «Щеночек по имени Мяу» | О.Г. Бостер | 04.05.2006 | 04.05.2023 |
| 0003 | 2 | Марина | Жук | «Гадкий утенок» | Ганс Христиан Андерсен | 13.07.2003 | 15.07.2003 |
| 0004 | 3 | Ирина | Вилеевич | «Лягушка путешественница» | В.М Гаршин | 26.06.2006 | 11.07.2006 |
| 0005 | 4 | Никита | Легостаев | «Щеночек по имени Мяу» | О.Г. Бостер | 04.05.2006 | 04.05.2023 |
| 0006 | 5 | Ян | Пожарский | «Муму» | И.С. Тургенев | 19.04.2020 | 19.06.2021 |
| 0007 | 5 | Ян | Пожарский | «Снежная королева» | Ганс Христиан Андерсен | 26.04.2023 | 26.04.2023 |

Примечание **–**  Источник: собственная разработка

Записи этой таблицы имеют значительное избыточное дублирование данных, так как Автор указывается для каждой книги, ID пользователя указывается для каждого пользователя. Избавиться от дублирования можно разбив таблицу.

Приведем примерную таблицу к третьей форме, разбив ее на 14 таблиц и уточнив атрибуты для каждой.

Таблица 2.3 **–** Таблица User

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Имя | Фамилия | Заметки |
| 1 | Артем | Василевский | Задолжал прошлую книгу |
| 2 | Марина | Жук | Порвала корешок |
| 3 | Ирина | Вилеевич | Забыла закладку |

Окончание таблицы 2.3

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | Никита | Легостаев | Хорошо относится к книгам |
| 5 | Ян | Пожарский | Погнул страницу |

Примечание **–**  Источник: собственная разработка

Таблица 2.4 **–** Таблица Genre

|  |  |
| --- | --- |
| ID | Название |
| 0001 | Детектив |
| 0002 | Драма |
| 0003 | Экшен |
| 0004 | Триллер |
| 0005 | Ужасы |
| 0006 | Романтика |

Примечание **–**  Источник: собственная разработка

Таблица 2.5 **–** Таблица Publisher

|  |  |
| --- | --- |
| ID | Название |
| 0001 | «А» |
| 0002 | «Б» |
| 0003 | «В» |
| 0004 | «Д» |
| 0005 | «Ъ» |

Примечание **–**  Источник: собственная разработка

Таблица 2.6 **–** Таблица Author

|  |  |
| --- | --- |
| ID | Псевдоним |
| 0001 | Г.Б. Остер |
| 0002 | Ганс Христиан Андерсен |
| 0003 | В.М Гаршин |
| 0004 | О.Г. Бостер |
| 0005 | И.С. Тургенев |

Примечание **–**  Источник: собственная разработка

Таблица 2.7 **–** Таблица Cover

|  |  |
| --- | --- |
| ID | Тип |
| 1111 | Твердая |
| 1112 | Мягкая |
| 1113 | Интегральная |
| 1114 | Суперобложка |
| 1115 | Хром-эрзац |

Примечание **–**  Источник: собственная разработка

Таблица 2.8 **–** Таблица Book

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Название | ID Автор | Год публикации | ID Издательство | ID Жанр | ID Тип обложки | Доступность | Кол-во страниц |
| 1 | «Котенок по имени Гав» | 0001 | 2001 | 0001 | 0001 | 1111 | + | 20 |
| 2 | «Гадкий утенок» | 0002 | 2002 | 0002 | 0002 | 1112 | + | 30 |
| 3 | «Лягушка путешественница» | 0003 | 2000 | 0003 | 0003 | 1113 | + | 40 |
| 4 | «Щеночек по имени Мяу» | 0004 | 2000 | 0004 | 0004 | 1111 | + | 50 |
| 5 | «Муму» | 0005 | 2000 | 0001 | 0005 | 1114 | + | 60 |
| 6 | «Снежная королева» | 0002 | 2000 | 0005 | 0006 | 1115 | + | 10 |

Примечание **–**  Источник: собственная разработка

Таблица 2.9 **–** Таблица Illustrators

|  |  |
| --- | --- |
| ID | Псевдоним |
| 0101 | Атанасия Ясимович |
| 0102 | Эльмира Юрашкевич |

Примечание **–**  Источник: собственная разработка

Таблица 2.10 **–** Таблица IllustratedBook

|  |  |  |  |
| --- | --- | --- | --- |
| BOOK ID | IL ID | Иллюстратор | Наличие цвета |
| 1 | 11 | 0101 | Да |
| 3 | 33 | 0102 | Нет |

Примечание **–**  Источник: собственная разработка

Таблица 2.11 **–** Таблица Language

|  |  |
| --- | --- |
| ID | Язык |
| 1111 | Датский |
| 1112 | Английский |

Примечание **–**  Источник: собственная разработка

Таблица 2.12 **–** Таблица Translator

|  |  |  |
| --- | --- | --- |
| ID | Имя | Фамилия |
| 0001 | Иван | Иванов |
| 0002 | Петр | Петров |

Примечание **–**  Источник: собственная разработка

Таблица 2.13 **–** Таблица ForeignBook

|  |  |  |  |
| --- | --- | --- | --- |
| BOOK ID | FB ID | ID Переводчика | ID Язык оригинала |
| 2 | 22 | 0001 | 1111 |
| 6 | 66 | 0002 | 1111 |

Примечание **–**  Источник: собственная разработка

Таблица 2.14 **–** Таблица BookUser

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | US ID | BOOK ID | Дата выдачи | Дата возврата | Заметки |
| 1 | 1111 | 1 | 01.01.2001 | 01.02.2001 | В хорошем состоянии |
| 2 | 2222 | 2 | 13.07.2003 | 15.07.2003 |  |

Примечание **–**  Источник: собственная разработка

Таблица 2.15 **–** Таблица LanguageTransltor

|  |  |
| --- | --- |
| ID Переводчика | ID Языка |
| 0001 | 1111 |
| 0001 | 1112 |
| 0002 | 1111 |

Примечание **–**  Источник: собственная разработка

Таблица 2.16 **–** Таблица GenreBook

|  |  |
| --- | --- |
| ID книги | ID Жанра |
| 1 | 0001 |
| 2 | 0002 |
| 3 | 0003 |
| 4 | 0004 |
| 5 | 0005 |

Примечание **–**  Источник: собственная разработка

## 2.5 Сущности и взаимосвязи

В базе данных для информационной системы «Домашняя библиотека» можно перечислить следующие сущности.

Таблица 2.17 **–** Сущность Book

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| bk\_id (id книги) | int |
| bk\_name (название книги) | text |
| au\_id (id автора) | int |
| pb\_id (id издательства) | int |
| cv\_id (id обложки) | int |
| bk\_pages (количество страниц) | int |
| bk\_publishyear (год издания) | date |
| bk\_availability (доступность) | bool |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут bk\_id.

От сущности Book наследуется еще две сущности, IllustratedBook и ForeignBook. Помимо вышеперечисленных атрибутов данные сущности еще имеют свои собственные атрибуты.

Таблица 2.18 **–** Сущность ForeignBook

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| fb\_id (id иностранной книги) | int |
| lt\_id (id переводчика над книгой с языком оригинала) | int |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут fb\_id.

Таблица 2.19 **–** Сущность IllustratedBook

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| ib\_id (id иллюстрированной книги) | int |
| il\_id (id иллюстратора) | int |
| ib\_colored (наличие цвета) | bool |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут ib\_id.

Таблица 2.20 **–** Сущность User

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| us\_id (id пользователя) | int |
| us\_name (имя) | text |
| us\_lastname (фамилия) | text |
| us\_note (примечание) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут us\_id.

Таблица 2.21 **–** Сущность Author

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| au\_id (id автора) | int |
| au\_pseudonym (псевдоним) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут au\_id.

Таблица 2.22 **–** Сущность Genre

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| gn\_id (id жанра) | int |
| gn\_name (название жанра) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут gn\_id.

Таблица 2.23 **–** Сущность Language

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| ln\_id (id языка) | int |
| ln\_name (название языка) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут ln\_id.

Таблица 2.24 **–** Сущность Cover

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| cv\_id (id обложки) | int |
| cv\_type (тип обложки) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут cv\_id.

Таблица 2.25 **–** Сущность Publisher

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| pb\_id (id издательства) | int |
| pb\_name (название) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут pb\_id.

Таблица 2.26 **–** Сущность Illustrator

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| il\_id (id иллюстратора) | int |
| il\_pseudonym (псевдоним) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут il\_id.

Таблица 2.27 **–** Сущность Translator

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| tr\_id (id переводчика) | int |
| tr\_name (имя) | text |
| tr\_lastname (фамилия) | text |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут tr\_id.

Чтобы показать связь сущностей Book и User была создана промежуточная сущность BookUser которая отображает возможность одного пользователя брать несколько книг и возможность одной книги браться повторно. Данная промежуточная сущность также позволяет контролировать даты возврата и выдачи.

Таблица 2.28 **–** Промежуточная сущность BookUser

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| bu\_id (id записи выдачи) | int |
| bk\_id (id книги) | int |
| us\_id (id пользователя) | int |
| bu\_extraditionDate (дата выдачи) | date |
| bu\_returnDate (дата возврата) | date |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут bu\_id.

Для отображения связи сущностей Language и Translator была создана промежуточная сущность LanguageTranslator которая отображает возможность одного переводчика знать сразу несколько языков.

Таблица 2.29 **–** Промежуточная сущность LanguageTranslator

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| lt\_id (id связанного языка с переводчиком) | int |
| tr\_id (id переводчика) | int |
| ln\_id (id языка) | int |

Примечание **–**  Источник: собственная разработка

В данной сущности первичным ключом является атрибут lt\_id.

Для отображения связи сущностей Book и Genre была создана промежуточная сущность GenreBook которая отображает возможность у одной книги иметь сразу несколько жанров.

Таблица 2.30 **–** Промежуточная сущность GenreBook

|  |  |
| --- | --- |
| Название атрибута | Тип атрибута |
| bk\_id (id книги) | int |
| gn\_id (id жанра) | int |

Примечание **–**  Источник: собственная разработка

В данной сущности внешними ключами являются атрибуты bk\_id и gn\_id.

Все связи в базе данных между сущностями покажем в таблице ниже.

Таблица 2.31 **–** Связи между сущностями

|  |  |  |
| --- | --- | --- |
| Первая сущность | Вторая сущность | Тип связи |
| Book | Author | Один-ко-многим (1:n) |
| Book | Publisher | Один-ко-многим (1:n) |
| Book | Cover | Один-ко-многим (1:n) |
| Book | ForeignBook | Один-ко-многим (1:n) |
| Book | IllustratedBook | Один-ко-многим (1:n) |
| Book | GenreBook | Один-ко-многим (1:n) |
| Book | BookUser | Один-ко-многим (1:n) |
| User | BookUser | Один-ко-многим (1:n) |
| Genre | GenreBook | Один-ко-многим (1:n) |
| Language | LanguageTranslator | Один-ко-многим (1:n) |
| Translator | LanguageTranslator | Один-ко-многим (1:n) |
| Illustrator | IllustratedBook | Один-ко-многим (1:n) |
| LanguageTranslator | ForeignBook | Один-ко-многим (1:n) |

Примечание **–**  Источник: собственная разработка

# 3 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ

Для разработки форм, с которыми мог бы взаимодействовать пользователь использовались Windows Forms. Это интерфейс программирования приложений, отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework.

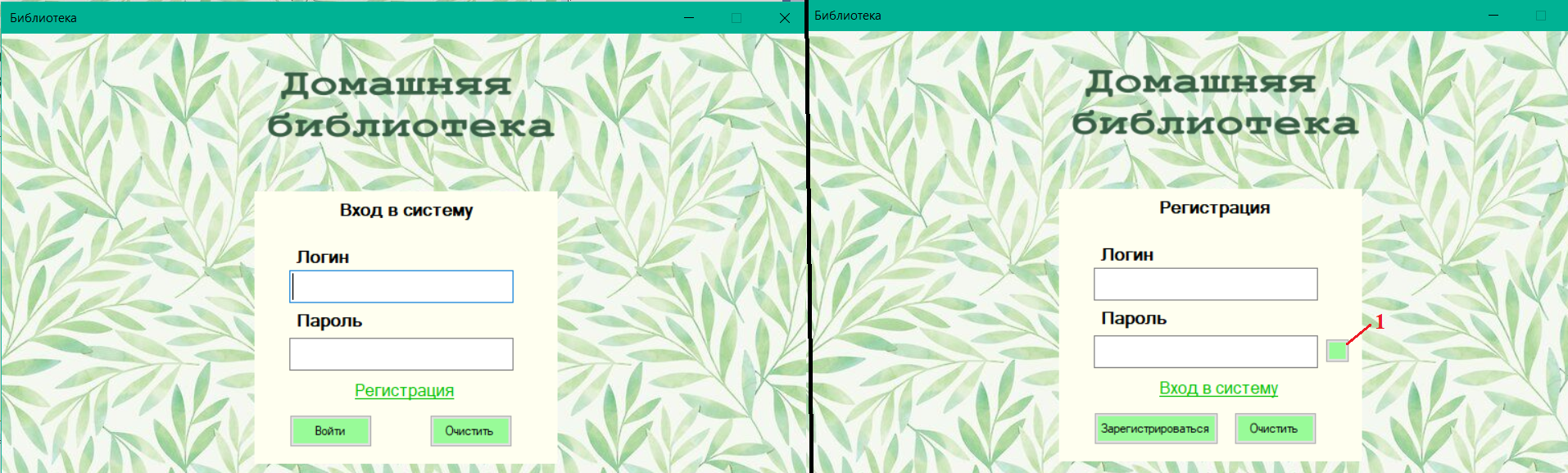
Всего в приложении 18 форм (Таблица 3.1).

**Таблица 3.1 – Формы приложения**

|  |  |
| --- | --- |
| Название | Назначение |
| Main | Родительская форма, в которую помещаются все остальные для корректного отображения |
| Login, SignUp | Формы, на которых содержатся все необходимые элементы управления, с помощью которых пользователь может авторизоваться или зарегистрироваться соответсвенно. |
| Menu | Форма, представляющая собой главное меню, через которое пользователь может получить доступ к просмотру и редактированию информации в базе данных через другие формы |
| Author, Book, Genre, Publisher, User, Accounting | Формы, которые позволяют пользователю просмотреть информацию в соответствующих таблицах базы данных, а также добавить новую запись, изменить запись или удалить ее. |
| ForeignBook, SeeBookInfo | Формы, которые позволяют осуществить просмотр информации, а также поиск и сортировку данных в таблице. |
| AddAccounting, AddAuthor, AddBook, AddGenre, AddPublisher, AddUserForm | Формы, представляющее собой окно для создания новой записи в базе данных. |

Примечание – Источник: собственная разработка

Формы Login и SignUp практически идентичны за исключением нескольких моментов (Рисунок 3.1).



**Рисунок 3.1 –** Формы Login и SignUp

Примечание – Источник: собственная разработка.

При нажатии на надпись «Регистрация» пользователь перейдет на форму SignUp, а при нажатии на «Вход в систему» – на Login соответственно. Под номером 1 находится кнопка, которая позволяет увидеть пароль.

Форма Menu изображена на рисунке 3.2.

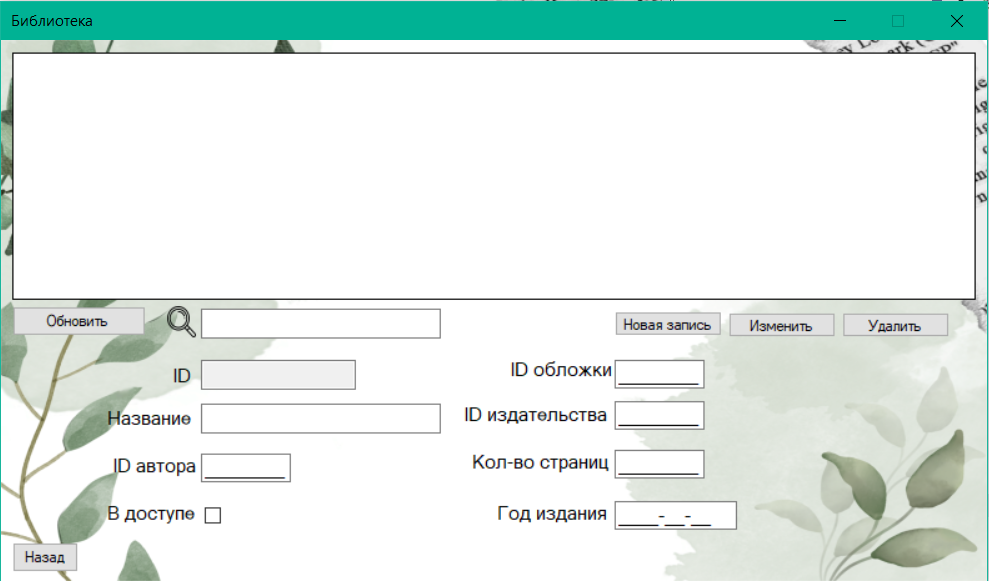
Данная форма позволяет увидеть все таблицы информацию в которых можно просмотреть, либо изменить, для этого нужно просто нажать на кнопку с названием таблицы, которая интересует пользователя. Также есть кнопка выйти, которая позволяет вернуться к форме Login.



**Рисунок 3.2 –** Форма Menu

Примечание – Источник: собственная разработка.

Рассмотрим работу с данными на примере таблицы «Книга» (Рисунок 3.3).

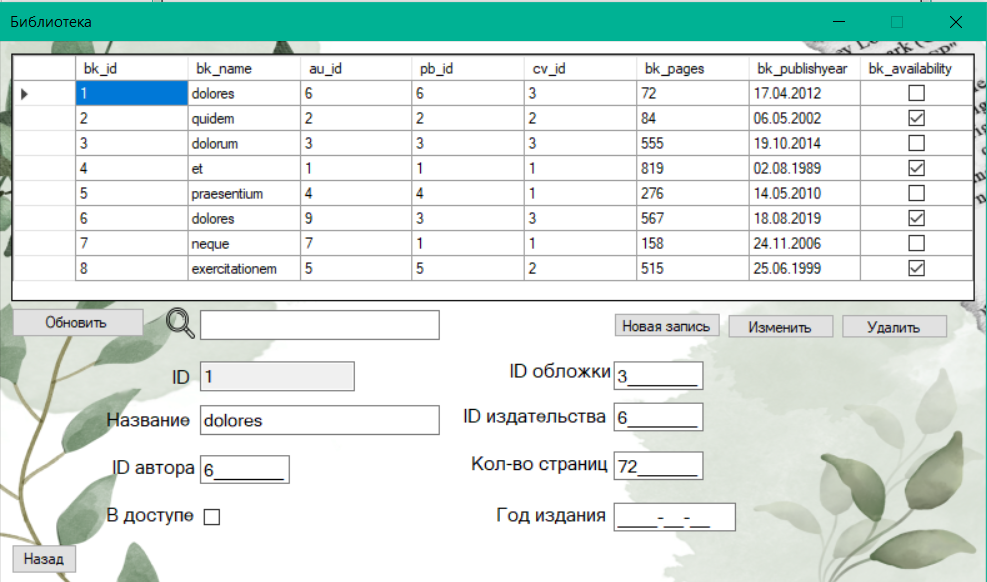


**Рисунок 3.3 –** Форма Book

Примечание – Источник: собственная разработка.

Форма содержит кнопки «Обновить», «Новая запись», «Изменить», «Удалить», которые позволяют отобразить/обновить данные таблицы, открыть форму для создания новой записи, сохранить изменения и удалить данные из таблицы соответственно. Кнопка «Назад» дает возможность вернуться к форме Menu.

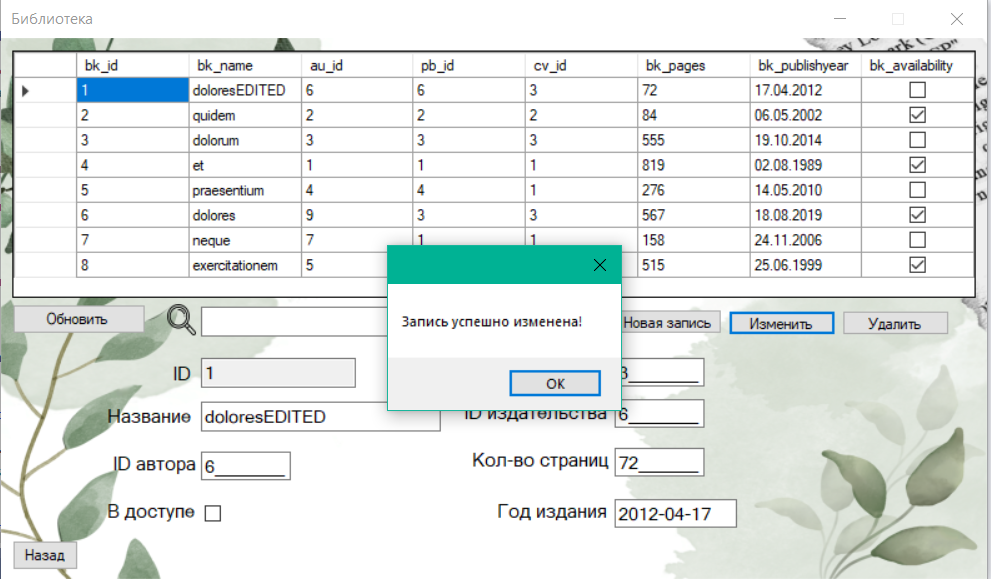
После нажатия на кнопку «Обновить» данные таблицы отобразятся и при выборе определенного элемента, его данные перенесутся в текстовые поля ниже, что необходимо для осуществления редактирования данных (Рисунок 3.4).



**Рисунок 3.4 –** Отображение данных в форме Book

Примечание – Источник: собственная разработка.

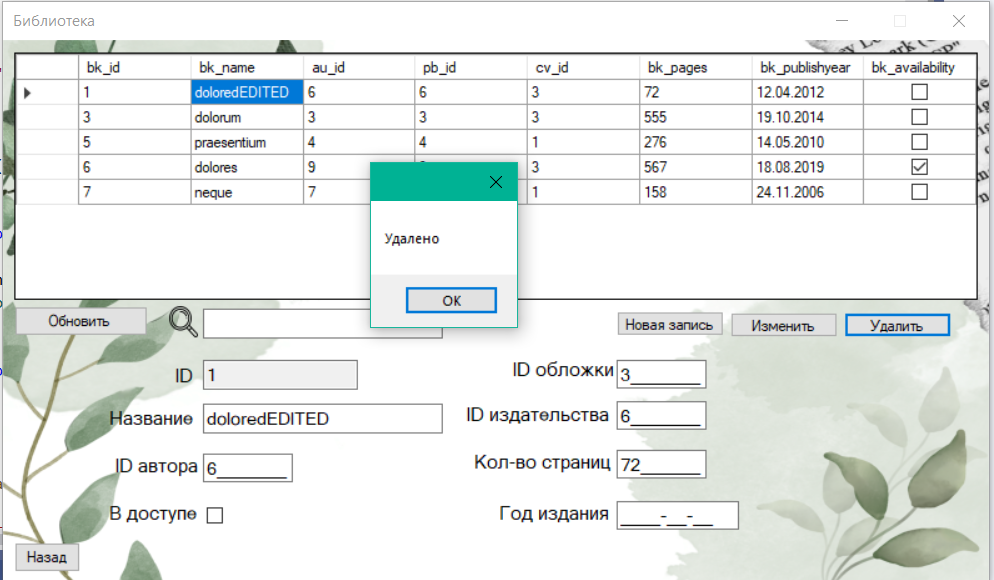
Для демонстрации функции изменения данных перепишем название книги на «doloresEDITED» и нажмем на кнопку «Изменить». Если все данные введены правильно, то пользователь увидит окно, которое подтвердит успешность операции (Рисунок 3.5).



**Рисунок 3.5 –** Подтверждение изменения информации

Примечание – Источник: собственная разработка.

А теперь удалим запись, нажав на кнопку «Удалить». При успешном выполнении функции пользователь получит соответствующее сообщение (Рисунок 3.6).

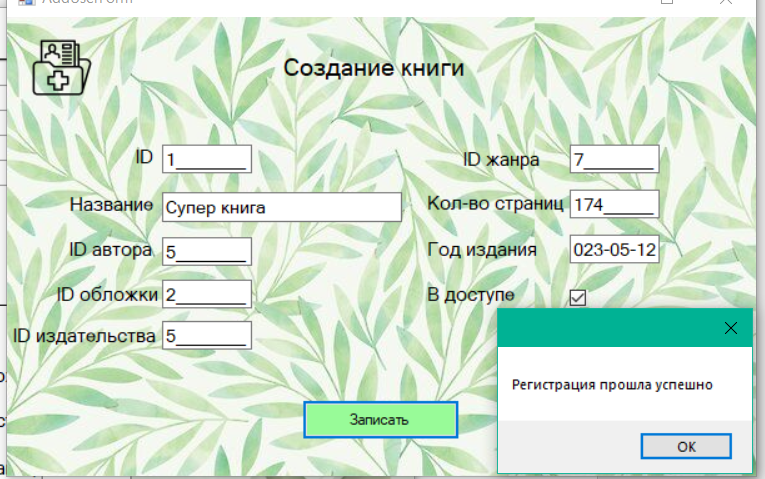


**Рисунок 3.6 –** Подтверждение изменения информации

Примечание – Источник: собственная разработка.

Рассмотрим создание новой записи, для этого необходимо нажать на кнопку «Новая запись». Откроется форма для создания новой записи. В данном случае это форма AddBook.

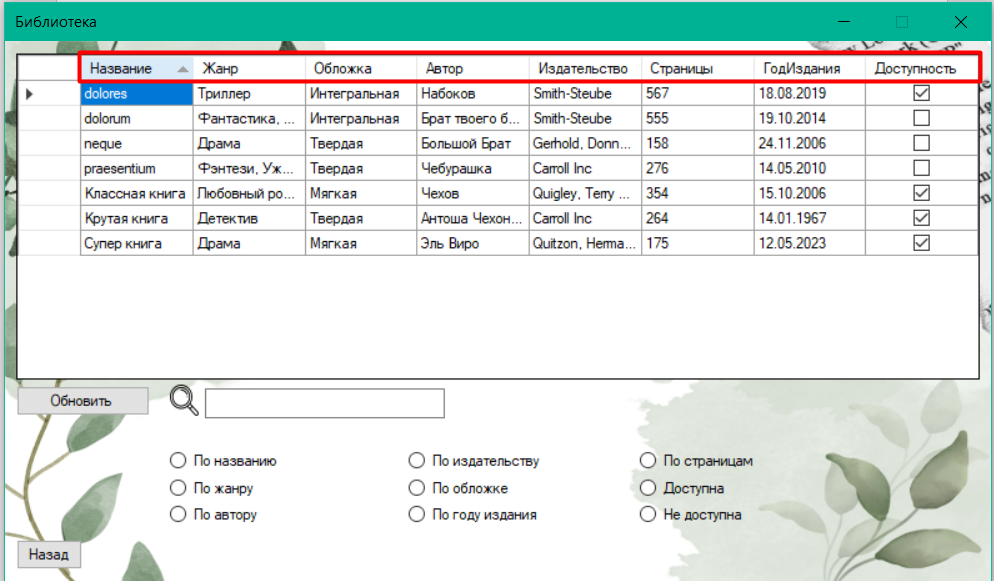
Заполнив все данные, для произведения записи необходимо нажать кнопку «Записать» после чего пользователь получит сообщение о результате выполненного действия (Рисунок 3.7).



**Рисунок 3.7 –** Создание новой записи

Примечание – Источник: собственная разработка.

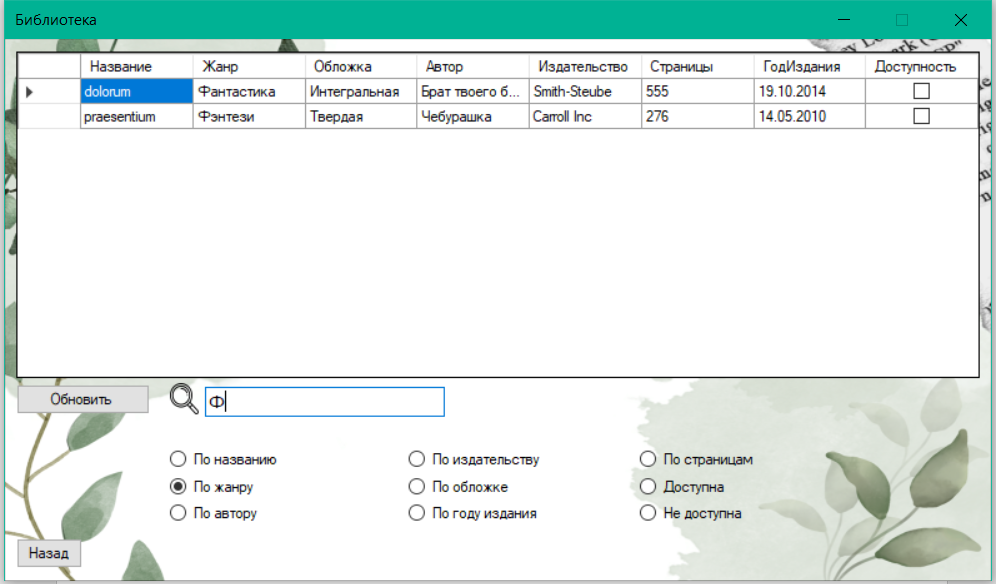
Осуществление поиска и сортировки данных рассмотрим на примере формы SeeBookInfo. Чтобы отсортировать элементы по определенной колонке по убыванию, нужно просто нажать на называние столбца (Рисунок 3.8), для сортировки по возрастанию – нажать еще раз.



**Рисунок 3.8 –** Сортировка данных

Примечание – Источник: собственная разработка.

Для осуществления поиска на данной форме пользователь сначала должен выбрать по какому столбцу будет производиться поиск, а затем ввести искомые значения в поле ввода. Таблица сразу же выдаст подходящие элементы, если такие имеются (Рисунок 3.9)



**Рисунок 3.9 –** Поиск по полям

Примечание – Источник: собственная разработка.

# ЗАКЛЮЧЕНИЕ

Автоматизированная информационная система для личной библиотеки может быть чрезвычайно полезна для людей, которым необходимо следить за своими книгами и другими материалами. Это может помочь им легко организовывать свою библиотеку, быстро получать доступ к книгам и отслеживать то, что они прочитали или взяли напрокат. Кроме того, автоматизированная информационная система также может предоставить ценную информацию о том, какие типы книг читают больше всего, и может помочь определить тенденции в предпочтениях чтения. Эта информация может быть использована для подбора лучших коллекций и рекомендаций читателям.

Для разработки были использованы такие программные продукты как MySQL, PowerDesigner. Они значительно облегчили процесс разработки информационной системы. При проектировании и разработке программы получены практические навыки в работе с классами и Windows-формами, закреплены теоретические знания по объектно-ориентированному программированию. Программа спроектирована и реализована в соответствии с основными принципами и техническим заданием и является интуитивно понятной для любого пользователя.

Таким образом, поставленные цели и задачи курсовой работы достигнуты и решены. Однако в дальнейшем автоматизированная информационная система может быть усовершенствована за счёт расширения функционала редактирования записей в базе данных, перемещения приложения в облачный сервис и добавления новых видов литературы, не ограничиваясь книгами.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. MySQL [Электронный ресурс / MySQL – Режим доступа: <https://www.mysql.com/>. – Дата доступа / 12.05.2023]

2. Проектирование базы данных [Электронный ресурс] / BRSU. – Режим доступа: http://lab314.brsu.by/roleg/BD\_TiG/theory/access04.htm/. - Дата доступа: 25.04.2023.

3. Разработка базы данных [Электронный ресурс] / ArtWell. – Режим доступа: https://www.artwell.ru/services/razrabotka-baz-dannykh//. - Дата доступа: 13.05.2023.

4. Справочник по C# [Электронный ресурс] / MSDN. – Режим доступа: https://msdn.microsoft.com/ru-ru/. - Дата доступа: 20.04.2023.

5. Физическая модель базы данных [Электронный ресурс] / VITMO. – Режим доступа: https://neerc.ifmo.ru/wiki/index.php?title=Физическая\_модель\_базы\_данных#:~:text=Определение%3A,СУБД%20для%20создания%20базы%20данных./. - Дата доступа: 07.05.2023.