МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»**

Кафедра Математического и компьютерного моделирования

Разработка информационной системы «Библиотека»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студентки 5 курса 561 группы

направление 09.03.03 — Прикладная информатика

механико-математического факультета

Волкова Владислава Владимировича

Научный руководитель
доцент, к. ф.-м. н., доцент _____ Е.Ю. Крылова
подпись, дата

Зав. кафедрой
зав. каф., д.ф.-м.н., доцент _____ Ю.А. Блинков
подпись, дата

Саратов 2022

Введение

Базы данных — это комплекс структур, предназначенных для хранения больших размеров информации и программных модулей, исполняющих управление данными, их выборку, сортировку и остальные похожие действия.

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

Каждая запись представляется набором именованных полей, или ячеек, которые могут хранить самую разнообразную информацию. Однотипные поля различных записей образуют столбец таблицы.

Создав одну таблицу, мы уже получаем полноценную базу данных.Впрочем в настоящей жизни структуры баз данных, а соответственно и методы их создания, заметно сложнее.

В информационном обществе преобладает изготовление информационного продукта, а материальный продукт становится более информационно емким. Изменится весь жизненный уклад, система ценностей: возрастает престиж культурного досуга, повышается нужда в знаниях, от человека необходима способность к умственному труду и творчеству. В результате появились противоречия между узкими возможностями человека по восприятию и переработке информации и имеющимися массивами хранящейся и передаваемой информации.

Появилось большое количество лишней информации, в которой временами нелегко найти и выбрать необходимые сведения.

Для решения аналогичных задач используются автоматизированные базы данных. Они стали неотъемлемой частью практически всех компьютерных систем - от отрасли до отдельного предприятия. За последние несколько лет увеличился уровень потребительских качеств систем управления базами данных (СУБД): разновидность поддерживаемых функций, удобный для пользователя интерфейс, сопряжение с программными продуктами, в частности с прочими СУБД, способности для работы в сети и так далее. СУБД дает возможность соединять воедино информацию из самых всевозможных источников (электронные таблицы, иные базы данных) и помогает легко на-

ходить необходимую информацию, донести ее до окружающих с помощью отчетов, графиков или таблиц.

К данному времени накоплен большой опыт в создании БД, специализированных для управления производством, это дает возможность произвести процесс создания БД более эффективным.

Цель данной работы заключается в создании базы данных библиотеки для работы в ней как обычному библиотекарю, так и администратору самой библиотеки. В действительности же, администратор библиотеки отличается от обычного библиотекаря лишь в том, что библиотекарь не имеет права вносить новые издания библиотеки в базу данных. В остальном же их права равны, и база данных является фактически единой как для администратора, так и для библиотекаря.

Большинство людей даже не догадываются, насколько труден и сложен учет книг.

Согласно цели поставим задачи:

- Изучение особенностей кадрового дела
- Разработка схемы БД
- Реализация разработанной схемы в конкретной СУБД
- Создание форм для ввода данных, отчетов, запросов
- Автоматизация работы с созданной БД.

Структура бакалаврской работы.Работа состоит из введения, пяти разделов, заключения, списка литературы и шесть приложений:

- Во введении рассматривается актуальность темы бакалаврской работы, а также ставятся цель и задачи данной работы.
- В первом разделе описываются модели жизненого цикла.
- Во втором разделе описывается предметная область проектируемой системы.
- В третьей главе описывается методология проектирования реляционных баз данных и процесс проектирования базы данных для библиотеки.
- В четвертой главе описан процесс разработки графического интерфейса пользователя информационной системы.
- В пятой главе описан процесс создания связующего кода, для базы данных и сайта.

- В приложениях приведены исходные коды реализации базы данных и графического интерфейса и связующего кода.

Основное содержание работы

Инфологическая модель должна включать такое формализованное описание предметной области, которое легко будет "читаться"не только специалистами по базам данных.

Модель «сущность-связь» называют также «ER-моделью» (essence-сущность, relation-связь)

Диаграмма сущность-связь - инструмент разработки моделей данных, гарантирующий стандартный способ определения данных и отношений между ними. Диаграмма представленна в соответствии с рисунком 1.

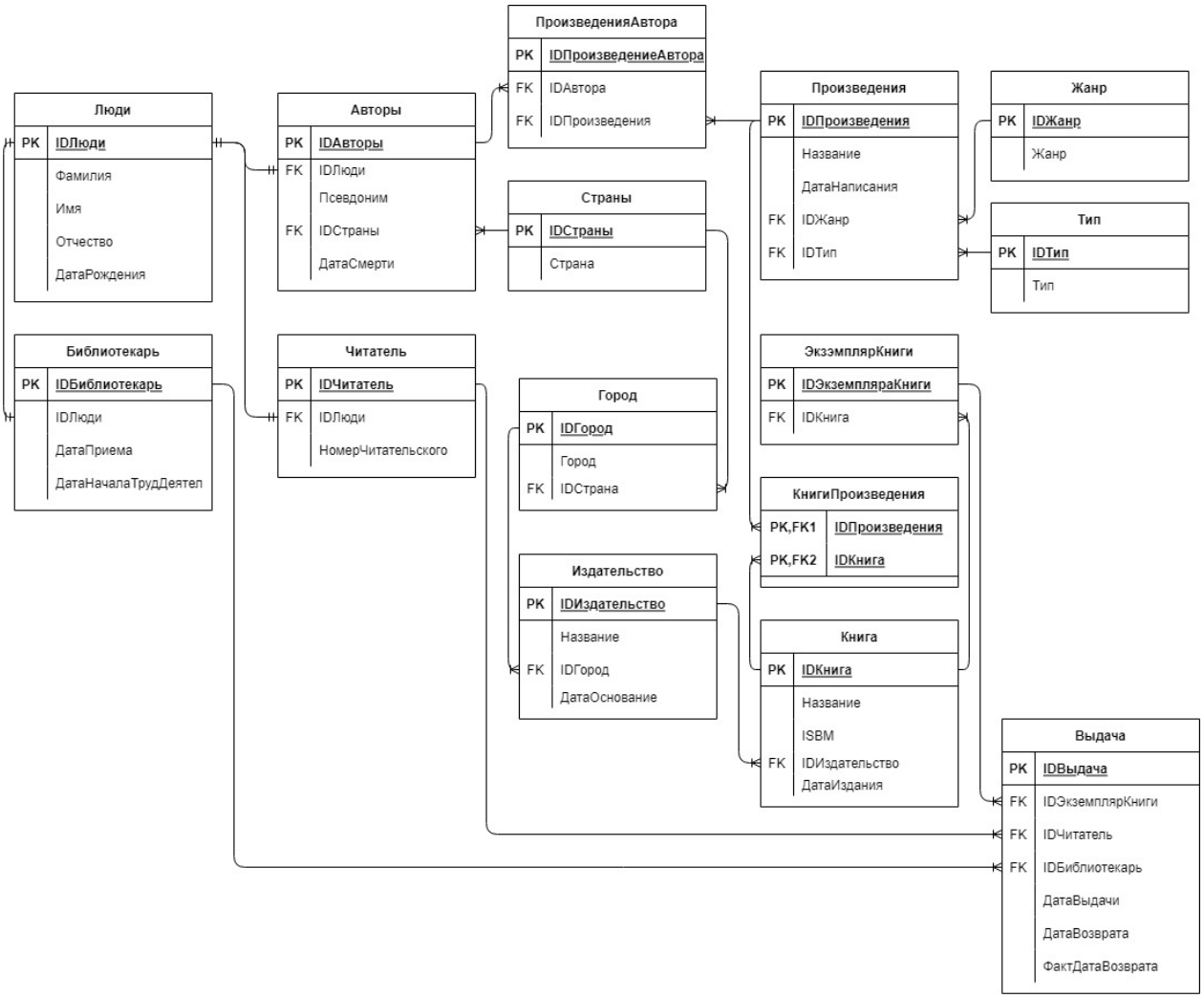

Рисунок 1: Диаграмма сущность-связь

Создадим базу данных Biblioteka написав код.

```
DROP DATABASE IF EXISTS Biblioteka;
CREATE DATABASE Biblioteka;
USE Biblioteka;
```

Листинг 1: Создание базы данных

Теперь создадим первую таблицу. В данной таблице будет хранится ID каждого человека и информация о них.

```
DROP TABLE IF EXISTS People;
CREATE TABLE People (
  IDPeople Int PRIMARY KEY,
  Surname VarChar(20) NOT NULL,
  FirstName VarCHar(20) NOT NULL,
  Surname2 VarChar(20),
  DateOfBirth Date not null);
```

Листинг 2: Таблица люди

Теперь напишем вторую таблицу, читатель. В данной таблице будет хранится информация о людях, которая будет подтягиваться из нашей первой таблицы и дополняться читательским билетом и каждому читателю будет присваиваться свой ID.

```
DROP TABLE IF EXISTS Reader;
CREATE TABLE Reader (
  IDReader Int PRIMARY KEY,
  IDPeople INT NOT NULL REFERENCES People(IDPeople) ON UPDATE CASCADE ON
   DELETE CASCADE UNIQUE,
  ReaderNumber VarCHar(15) UNIQUE);
```

Листинг 3: Таблица читатели

Аналогичным способом напишем слудующую таблицу библиотекарь. В данной таблице будет хранится информация о людях из таблицы люди и дополняться данными о дате приема на работу и дате фактическго выхода на работу.

```
DROP TABLE IF EXISTS Librarian;
CREATE TABLE Librarian (
  IDLibrarian INT PRIMARY KEY,
  IDPeople INT NOT NULL REFERENCES People(IDPeople) ON UPDATE CASCADE ON
   DELETE CASCADE UNIQUE,
  DateOfReceipt Date NOT Null CHECK (DateOfReceipt<=convert (date,
   CURRENT_TIMESTAMP)) DEFAULT CURRENT_TIMESTAMP,
  EmploymentDate date Not NULL,
```

```
  CHECK (EmploymentDate>=DateOfReceipt));
```

Листинг 4: Таблица библиотекари

Создадим таблицу страны, в которой будут хранится список стран и каждой стране будет присвоен свой ID.

```
DROP TABLE IF EXISTS Country;
CREATE TABLE Country (
  IDCountry Int PRIMARY KEY,
  Country VarChar(15) NOT NULL UNIQUE);
```

Листинг 5: Таблица страны

Аналогичным способом создадим таблицу города, в которой будет хранится список городов и страна в которой он находится, будет подтягиваться из таблицы страны.

```
DROP TABLE IF EXISTS Town;
CREATE TABLE Town (
  IDCity int PRIMARY KEY,
  City varchar(30) not null,
  IDCountry int Not null REFERENCES Country(IDCountry),
  UNIQUE (IDCountry, City));
```

Листинг 6: Таблица города

Теперь создадим таблицу авторы. В данной даблице будет подтягиваться информация о человеке из таблицы люди и страны из таблицы страны. Так же будет дополняться информацие о псевдоними автора и его дате смерти.

```
DROP TABLE IF EXISTS Author;
CREATE TABLE Author (
  IDAuthor Int PRIMARY KEY,
  IDPeople INT NOT NULL REFERENCES People(IDPeople) ON UPDATE CASCADE ON
   DELETE CASCADE UNIQUE,
  Alias VArChar(20),
  IDCountry INT REFERENCES Country(IDCountry) ON UPDATE CASCADE ON DELETE
   CASCADE,
  DateOfDeath Date);
```

Листинг 7: Таблица авторы

Теперь создадим таблицы жанров и стилей. В даблице жанры будут хранится жанры, а в табиц стили, стили.

```
DROP TABLE IF EXISTS Genre;
CREATE TABLE Genre (
```

```
  IDGenre INT PRIMARY KEY,
  Genre varchar(25) Not Null UNIQUE);
```

Листинг 8: Таблица жанров

```
DROP TABLE IF EXISTS Style;
CREATE TABLE Style (
  IDType INT PRIMARY KEY,
  Style varchar(30) Not Null UNIQUE);
```

Листинг 9: Таблица стилей

Далее создим таблицу произведений в которую будет подтягиваться информация о жанрах и стилях из прошлых таблиц.Так же дополняться информацией о название произведения и дате его написания.

```
DROP TABLE IF EXISTS Artworks;
CREATE TABLE Artworks (
  IDArtworks Int PRIMARY KEY,
  NameArtworks VARCHar(150) Not Null DEFAULT 'Без Названия',
  DateOfWriting Date CHECK (DateOfWriting<=convert (date, CURRENT_TIMESTAMP)
   ),
  IDGenre int Not Null REFERENCES Genre(IDGenre),
  IDType int Not Null REFERENCES Style(IDType));
```

Листинг 10: таблица произведений

Создадим таблицу произведения автора. В которой будет подтягиваться информация о авторе из таблицы авторы и информация о произведение из таблицы произведений.

```
DROP TABLE IF EXISTS ArtworksAuthor;
CREATE TABLE ArtworksAuthor (
  IDArtworksAuthor INT PRIMARY KEY,
  IDAuthor INT NOT NULL REFERENCES Author(IDAuthor),
  IDArtworks INT Not Null REFERENCES Artworks(IDArtworks),
  UNIQUE (IDAuthor, IDArtworks));
```

Листинг 11: Таблица произведений авторов

Теперь создадим таблицу издательства. В которой будет хранится информация о название издательства, годе его основания и подтягиваться информация о городе в котором находится издательство.

```
DROP TABLE IF EXISTS Publisher;
CREATE TABLE Publisher (
  IDPublisher INT PRIMARY KEY,
```

```
  NamePublisher varchar(20) not null,
  IDCity int not null REFERENCES Town(IDCity),
  FoundationDate date CHECK (FoundationDate<=convert (date,
   CURRENT_TIMESTAMP)),
  UNIQUE (NamePublisher, IDCity));
```

Листинг 12: Таблица издательства

Аналогичным способом создаем таблицу книги. В данной таблице будет хранится информация о названии книги, ISBN, дате ее публикации и будет подтягиваться информация о издательстве книги.

```
DROP TABLE IF EXISTS Book;
CREATE Table Book (
  IDBook INT PRIMARY KEY,
  NameBook varchar(150) not null,
  ISBN varchar(13) UNIQUE,
  IDPublisher INT not null REFERENCES Publisher(IDPublisher),
  DateOfPublish date not null);
```

Листинг 13: Таблица книги

Создаем таблицу произведение в книги, в данной таблице будет хранится информация о книги, которая подтягивается из таблицы книги и информация из таблицы произведения из таблицы произведения.

```
DROP TABLE IF EXISTS BookArtworks;
CREATE TABLE BookArtworks (
  IDBook Int Not null REFERENCES Book(IDBook),
  IDArtworks INT Not null REFERENCES Artworks(IDArtworks),
  PRIMARY KEY (IDBook, IDArtworks));
```

Листинг 14: Таблица произведение в книги

Далее создаем таблицу экземпляр книги, в которой будет хранится информация о книги из таблицы книги.

```
DROP TABLE IF EXISTS CopyBook;
CREATE TABLE CopyBook (
  IDCopyBook Int PRIMARY KEY,
  IDBook int not null REFERENCES Book(IDBook));
```

Листинг 15: Таблица экземпляр книги

Создаем нашу последнюю таблицу выдачи книг. В данной таблице будет хранится информация о экземпляри книги, читателе, библиотекаре из прошлых таблиц и дополняться информацией о дате выдачи, дате возврата книги и фактической дате возврата.

```sql
DROP TABLE IF EXISTS Delivery;
CREATE TABLE Delivery (
  IDDelivery INT PRIMARY KEY,
  IDCopyBook INT Not null REFERENCES CopyBook(IDCopyBook),
  IDReader Int Not Null REFERENCES Reader(IDReader),
  IDLibrarian INT Not null REFERENCES Librarian(IDLibrarian),
  DateOfDelivery Date not null DEFAULT CURRENT_TIMESTAMP CHECK (
   DateOfDelivery<=convert (date, CURRENT_TIMESTAMP)),
  ReturnDate date not null DEFAULT (CURRENT_TIMESTAMP+30),
  FactReturnDate date,
  CHECK (FactReturnDate>=DateOfDelivery),
  CHECK (ReturnDate>=DateOfDelivery));
```

Листинг 16: Таблица экземпляр книги

Для удобной работы с базой данных библиотеки, создадим простой сайт. Библиотекарь заходит на домен web-сервиса, попадает на главный экран. За отображение данной страницы отвечает index.php

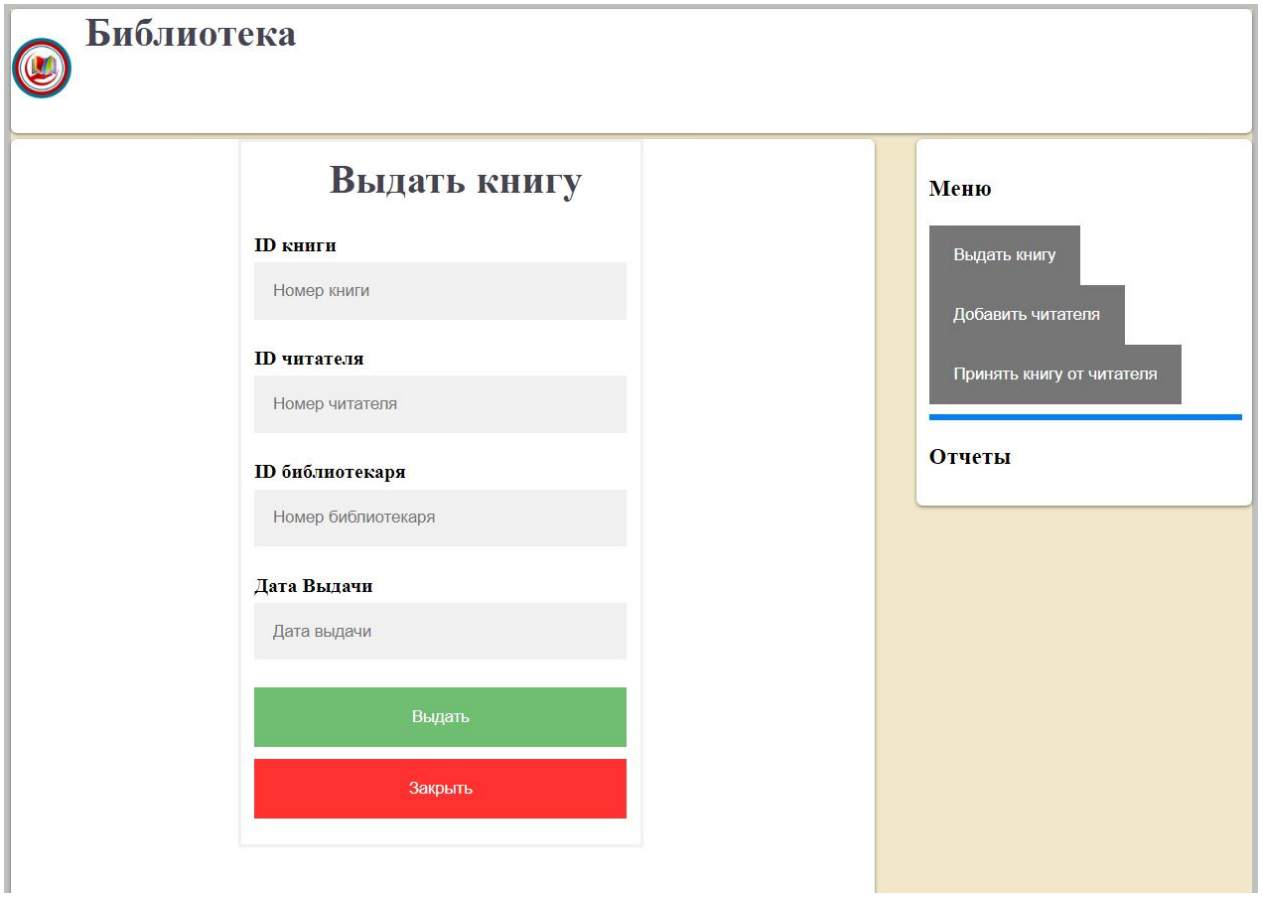

Рисунок 2: Главная страница сайта

Библиотекарь на данной странице может выдать книгу с помощью кнопки, для этого был написан следующий код в соответствии с рисунком (2):

```
<button class="open-button" onclick="openForm()">Выдать книгу</button>

<div class="form-popup" id="myForm">
  <form action="action.php" METHOD=POST class="form-container">
    <h1>Выдать книгу</h1>
    <label for="IDCopyBook"><b>ID книги</b></label>
    <input type="text" placeholder="Номер книги" name="IDCopyBook" required>
    <label for="IDReader"><b>ID читателя</b></label>
    <input type="text" placeholder="Номер читателя" name="IDReader" required>
    <label for="IDLibrarian"><b>ID библиотекаря</b></label>
    <input type="text" placeholder="Номер библиотекаря" name="IDLibrarian"
   required>
    <label for="DateOfDelivery"><b>Дата Выдачи</b></label>
    <input type="text" placeholder="Дата выдачи" name="DateOfDelivery"
   required>
    <label for="ReturnDate"><b>Дата Возврата</b></label>
    <input type="text" placeholder="Дата возврата" name="ReturnDate" required>
    <button type="submit" class="btn">Выдать</button>
    <button type="button" class="btn cancel" onclick="closeForm()">Закрыть</
   button>
  </form>
</div>
```

Листинг 17: Кнопка выдачи книги

Для того что кнопка работа на этой же странице, она была сделана ввиде формы, которая всплывает при нажатии и исчезает если ее закрыть. Для ее работы необходимо добавить JavaScript.

```
function openForm() {
      document.getElementById("myForm").style.display = "block"; }
    function closeForm() {
      document.getElementById("myForm").style.display = "none"; }

```

Листинг 18: Кнопка выдачи книги JavaScript

Для взаимодействия нашей базы данных и сайта, нам необходимо написать код PHP, который будет обрабатывать полученную информацию. Информацию он будет получать с сайта из формы, которую мы заполняем и отправлять в базу данных.

Получать данный из формы, которую мы заполнили на сайте он будет с помощь кода

```
$IDCopyBook = $_POST['IDCopyBook'];
$IDReader = $_POST['IDReader'];
```

```
$IDLibrarian = $_POST['IDLibrarian'];
$DateOfDelivery = $_POST['DateOfDelivery'];
$ReturnDate = $_POST['ReturnDate'];
```

Листинг 19: Код PHP для получение информации из формы

Получив информацию из формы, ему необходимо выполнить подключение к базе данных, чтобы внести полученную информацию в таблицу. Для подключения к базе данных используется следующий запрос.

```
$Host = "localhost";
$User = "admin";
$Password = "";
$DBName = "biblioteka";
$TableName = "delivery";
$Link = mysqli_connect ($Host, $User, $Password, $DBName);
```

Листинг 20: Код PHP для подключения к базе данных

Подключившись к базе данных нужно составить запрос, с помощью которого будет добавляться полученная информация из формы в таблицу базы данных. IDCopyBook, IDReader, IDLibrarian, DateOfDelivery и ReturnDate это данные которые мы получаем из формы.

```
$Query = "INSERT into $TableName values ('$id', '$IDCopyBook', '$IDReader',
    '$IDLibrarian', '$DateOfDelivery', '$ReturnDate','')";
```

Листинг 21: Код PHP для добавления данных в базу данных

В данном запросе id является следующей пустой строкой в таблице, в которую будет вноситься информация. Для получения следующей пустой строки был использован следующий код:

```
$result = $Link->query("SELECT COUNT(*) FROM $TableName");
$row = $result->fetch_row();
$id = $row[0];
```

Листинг 22: Код PHP для подсчета строк в таблице

Теперь наш код PHP готов, он получает данные из формы, подключается к базе данных и записывает полученную информацию. Для проверки добавим условия, при котором если данный запрос смог подключиться к базе данных, тогда будет выводиться сообщение "Подключился к базе данных".

```
if ($Link = mysqli_connect ($Host, $User, $Password, $DBName)) {
   print("Подключился к базе данных<P>");};
```

Листинг 23: Код PHP для проверки подключения к базе данных

Так же необходимо добавить код для проверки, добавленна ли полученная информация из формы в базу данных.

```
if (mysqli_query($Link, $Query)) {
print ("Добавлено в базу данных!<BR>");
} else {
print ("НЕ добавлено в базу данных!<BR>");}
```

Листинг 24: Код PHP для проверки добавления данных

После выполнения всех поставленных задач, подключения к базе данных и добавления в нее данных, необходимо отключиться от нее. Для этого мы используем следующий код.

```
if (mysqli_close ($Link)){
    echo "<meta http-equiv=refresh content=5;URL=index.html>";}
```

Листинг 25: Код PHP для отключения от базы данных

Чтобы отключения не происходило моментально и мы могли увидеть результат выполнения данного кода необходимо поставить задержку. Для этого в данный код также добавим условие, при котором он будет перенаправлять нас на главную страницу, спустя 5 секунд.

Заключение

В ходе работы был проведен анализ предметной области, исследованы требования к информационной системе, проведено сравнение СУБД, рассмотрены и применены на практике средства проектирования реляционных баз данных, а также спроектирована сама система.

На практике способы разработки web-интерфейса были реализованы с помощью программного кода, написанного на языке HTML. Система является работоспособной, простой в эксплуатации и имеет возможность для расширения.

Проведенные в работе исследования позволили более углубленно изучить процесс проектирования информационных систем, реляционных баз данных и разработки web-интерфейса, а также повысить навыки решения практических задач как теоретическими, так и практическими способами.