**Проектирование и разработка базы данных для автоматизированной системы учета "Библиотека"**

**1. Введение**

**1.1. Назначение системы**

Автоматизированная система учета "Библиотека" (далее – Система) предназначена для эффективного управления книжным фондом, учета читателей, регистрации операций по выдаче и возврату книг, а также для формирования отчетности и анализа деятельности библиотеки. Система призвана заменить ручные методы учета, сократить время на выполнение типовых операций и повысить качество обслуживания читателей.

**1.2. Цели проекта**

* Разработать логическую и физическую структуру базы данных (БД), отвечающую требованиям предметной области.
* Обеспечить целостность, непротиворечивость и безопасность хранимых данных.
* Создать набор представлений (views) для упрощения типовых запросов и формирования отчетов.
* Подготовить документацию по структуре БД и основным операциям.
* Предоставить SQL-скрипты для создания структуры БД и заполнения её тестовыми данными.

**1.3. Область применения**

Разработанная база данных может быть использована в малых и средних публичных, учебных или специализированных библиотеках для автоматизации их основной деятельности.

**2. Описание предметной области (Библиотека)**

**2.1. Основные объекты учета:**

* **Книги:** Каждая книга характеризуется названием, автором(ами), ISBN, годом издания, издательством, жанром и количеством страниц. Важно различать книгу как произведение (уникальный ISBN) и физический экземпляр книги (инвентарный номер).
* **Авторы:** Информация об авторах книг (ФИО, даты жизни, биография).
* **Издательства:** Информация об издательствах (название, город).
* **Жанры:** Классификатор жанров литературы.
* **Читатели:** Зарегистрированные пользователи библиотеки, имеющие читательский билет и персональные данные.
* **Экземпляры книг:** Физические копии книг, каждая со своим инвентарным номером, статусом (на полке, выдана, списана) и местоположением.
* **Выдачи/Возвраты:** Факты выдачи экземпляров книг читателям с указанием дат выдачи, предполагаемого и фактического возврата.

**2.2. Основные процессы:**

* Регистрация новых читателей.
* Поступление новых книг и их экземпляров, постановка на учет.
* Выдача книг читателям.
* Возврат книг от читателей.
* Отслеживание задолженностей по книгам.
* Списание утерянных или поврежденных экземпляров.
* Поиск книг в фонде по различным критериям.

**3. Проектирование базы данных**

**3.1. Ключевые сущности и их атрибуты**

На основе анализа предметной области были выделены следующие ключевые сущности, которые будут представлены в виде таблиц в БД:

* authors (Авторы)
* publishers (Издательства)
* genres (Жанры)
* books (Книги как произведения)
* book\_copies (Экземпляры книг)
* readers (Читатели)
* loans (Выдачи книг)
* Вспомогательные таблицы для связей "многие-ко-многим": book\_authors, book\_genres.

**3.2. Концептуальная модель данных (ERD - описание)**

* **Авторы и Книги:** Связь "многие-ко-многим". Один автор может написать много книг, и одна книга может быть написана несколькими авторами. Реализуется через связующую таблицу book\_authors.
* **Жанры и Книги:** Связь "многие-ко-многим". Одна книга может относиться к нескольким жанрам, и один жанр может включать много книг. Реализуется через связующую таблицу book\_genres.
* **Издательства и Книги:** Связь "один-ко-многим". Одно издательство может выпустить много книг, но каждая книга (произведение) издается одним издательством (в рамках данного учета).
* **Книги и Экземпляры книг:** Связь "один-ко-многим". Одна книга (как произведение) может иметь множество физических экземпляров.
* **Читатели и Выдачи:** Связь "один-ко-многим". Один читатель может взять много книг (в разное время или несколько одновременно).
* **Экземпляры книг и Выдачи:** Связь "один-ко-многим". Один конкретный экземпляр книги может быть выдан много раз (последовательно), но в один момент времени он может быть связан только с одной активной выдачей.

**4. Описание структуры базы данных (таблицы и поля)**

**4.1. Таблица authors (Авторы)**

* **Назначение:** Хранение информации об авторах книг.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор автора.
  + first\_name (VARCHAR(100), NOT NULL): Имя автора.
  + last\_name (VARCHAR(100), NOT NULL): Фамилия автора.
  + middle\_name (VARCHAR(100), NULL): Отчество автора.
  + birth\_date (DATE, NULL): Дата рождения.
  + biography (TEXT, NULL): Краткая биография.

**4.2. Таблица publishers (Издательства)**

* **Назначение:** Хранение информации об издательствах.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор издательства.
  + name (VARCHAR(255), NOT NULL, UNIQUE): Название издательства.
  + city (VARCHAR(100), NULL): Город издательства.
  + contact\_info (VARCHAR(255), NULL): Контактная информация.

**4.3. Таблица genres (Жанры)**

* **Назначение:** Справочник жанров литературы.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор жанра.
  + name (VARCHAR(100), NOT NULL, UNIQUE): Название жанра.
  + description (TEXT, NULL): Описание жанра.

**4.4. Таблица books (Книги - как произведения)**

* **Назначение:** Хранение информации о книгах как о литературных произведениях.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор книги.
  + title (VARCHAR(255), NOT NULL): Название книги.
  + isbn (VARCHAR(20), NOT NULL, UNIQUE): Международный стандартный книжный номер.
  + publication\_year (YEAR, NULL): Год издания.
  + publisher\_id (INT UNSIGNED, NULL, FK -> publishers.id): ID издательства.
  + pages\_count (INT UNSIGNED, NULL): Количество страниц.
  + description (TEXT, NULL): Краткое описание/аннотация.

**4.5. Таблица book\_authors (Авторы книг - связующая)**

* **Назначение:** Реализация связи "многие-ко-многим" между книгами и авторами.
* **Поля:**
  + book\_id (INT UNSIGNED, PK, FK -> books.id): ID книги.
  + author\_id (INT UNSIGNED, PK, FK -> authors.id): ID автора.

**4.6. Таблица book\_genres (Жанры книг - связующая)**

* **Назначение:** Реализация связи "многие-ко-многим" между книгами и жанрами.
* **Поля:**
  + book\_id (INT UNSIGNED, PK, FK -> books.id): ID книги.
  + genre\_id (INT UNSIGNED, PK, FK -> genres.id): ID жанра.

**4.7. Таблица book\_copies (Экземпляры книг)**

* **Назначение:** Хранение информации о конкретных физических экземплярах книг.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор экземпляра.
  + book\_id (INT UNSIGNED, NOT NULL, FK -> books.id): ID книги, к которой относится экземпляр.
  + inventory\_number (VARCHAR(50), NOT NULL, UNIQUE): Инвентарный номер экземпляра.
  + status (ENUM('available', 'on\_loan', 'damaged', 'lost', 'under\_repair'), NOT NULL, DEFAULT 'available'): Статус экземпляра.
  + acquisition\_date (DATE, NOT NULL): Дата поступления экземпляра.
  + location (VARCHAR(100), NULL): Местоположение в библиотеке.

**4.8. Таблица readers (Читатели)**

* **Назначение:** Хранение информации о зарегистрированных читателях.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор читателя.
  + library\_card\_number (VARCHAR(20), NOT NULL, UNIQUE): Номер читательского билета.
  + first\_name (VARCHAR(100), NOT NULL): Имя читателя.
  + last\_name (VARCHAR(100), NOT NULL): Фамилия читателя.
  + middle\_name (VARCHAR(100), NULL): Отчество читателя.
  + birth\_date (DATE, NOT NULL): Дата рождения.
  + address (VARCHAR(255), NULL): Адрес проживания.
  + phone\_number (VARCHAR(20), NULL): Контактный телефон.
  + email (VARCHAR(100), NULL, UNIQUE): Адрес электронной почты.
  + registration\_date (DATE, NOT NULL): Дата регистрации.
  + notes (TEXT, NULL): Заметки о читателе.

**4.9. Таблица loans (Выдачи книг)**

* **Назначение:** Регистрация фактов выдачи книг читателям.
* **Поля:**
  + id (INT UNSIGNED, PK, AI): Уникальный идентификатор выдачи.
  + book\_copy\_id (INT UNSIGNED, NOT NULL, FK -> book\_copies.id): ID выданного экземпляра.
  + reader\_id (INT UNSIGNED, NOT NULL, FK -> readers.id): ID читателя.
  + loan\_date (DATETIME, NOT NULL, DEFAULT CURRENT\_TIMESTAMP): Дата и время выдачи.
  + due\_date (DATE, NOT NULL): Ожидаемая дата возврата.
  + return\_date (DATETIME, NULL): Фактическая дата и время возврата.
  + fine\_amount (DECIMAL(10,2), NULL, DEFAULT 0.00): Сумма штрафа за просрочку.
  + notes (TEXT, NULL): Заметки по выдаче.

**5. Представления (Views)**

**5.1. view\_full\_book\_info**

* **Назначение:** Предоставляет полную информацию о книге, включая название, ISBN, год издания, издательство, список авторов (через запятую) и список жанров (через запятую).
* **Используемые таблицы:** books, publishers, authors, genres, book\_authors, book\_genres.

**5.2. view\_reader\_active\_loans**

* **Назначение:** Отображает информацию о книгах, которые в данный момент находятся на руках у читателей (не возвращены). Включает данные о читателе, названии книги, инвентарном номере, дате выдачи и сроке возврата.
* **Используемые таблицы:** loans, readers, book\_copies, books.

**5.3. view\_available\_book\_copies**

* **Назначение:** Показывает список всех доступных для выдачи экземпляров книг с указанием названия книги, авторов, жанров, инвентарного номера и местоположения.
* **Используемые таблицы:** book\_copies, books, publishers, view\_full\_book\_info (для удобства получения агрегированных данных об авторах/жанрах).

**5.4. view\_overdue\_loans**

* **Назначение:** Отображает список просроченных выдач, т.е. книг, не возвращенных в срок. Включает информацию о читателе (включая контакты), книге, дате выдачи, сроке возврата и количестве дней просрочки.
* **Используемые таблицы:** loans, readers, book\_copies, books.

**6. Типовые операции и сценарии использования**

**6.1. Регистрация нового читателя**

* **Описание:** Добавление информации о новом читателе в систему.
* **Затрагиваемые таблицы:** readers (INSERT).
* **Данные для ввода:** ФИО, дата рождения, адрес, телефон, email, генерация номера читательского билета.

**6.2. Добавление новой книги (произведения) и ее авторов/жанров**

* **Описание:** Внесение информации о новом литературном произведении.
* **Затрагиваемые таблицы:** books (INSERT), authors (INSERT, если автор новый, или SELECT существующего), book\_authors (INSERT), genres (INSERT, если жанр новый, или SELECT существующего), book\_genres (INSERT).
* **Данные для ввода:** Название, ISBN, год издания, издательство, страницы, описание, ФИО авторов, названия жанров.

**6.3. Поступление экземпляров книг**

* **Описание:** Регистрация поступления физических экземпляров уже существующих в каталоге книг.
* **Затрагиваемые таблицы:** book\_copies (INSERT).
* **Данные для ввода:** ID книги (из books), инвентарный номер, дата поступления, местоположение, начальный статус 'available'.

**6.4. Выдача книги читателю**

* **Описание:** Регистрация факта выдачи экземпляра книги читателю.
* **Действия:**
  1. Найти доступный экземпляр книги (book\_copies.status = 'available').
  2. Создать запись о выдаче в loans (указать book\_copy\_id, reader\_id, loan\_date, рассчитать due\_date).
  3. Изменить статус экземпляра в book\_copies на 'on\_loan'.
* **Затрагиваемые таблицы:** loans (INSERT), book\_copies (UPDATE).

**6.5. Возврат книги читателем**

* **Описание:** Регистрация факта возврата книги.
* **Действия:**
  1. Найти активную запись о выдаче для данного экземпляра (loans.return\_date IS NULL).
  2. Обновить loans.return\_date на текущую дату/время.
  3. Проверить на просрочку (сравнить return\_date и due\_date), при необходимости рассчитать и записать fine\_amount.
  4. Изменить статус экземпляра в book\_copies на 'available' (или 'damaged', если книга повреждена).
* **Затрагиваемые таблицы:** loans (UPDATE), book\_copies (UPDATE).

**6.6. Поиск книг**

* **Описание:** Поиск книг в каталоге по различным критериям.
* **Запросы к:** view\_full\_book\_info, view\_available\_book\_copies с использованием WHERE по названию, автору, ISBN, жанру.

**6.7. Списание экземпляра книги**

* **Описание:** Изменение статуса экземпляра на 'lost' или 'damaged' (если не подлежит восстановлению).
* **Затрагиваемые таблицы:** book\_copies (UPDATE).

**7. Пример заполнения данными (SQL-скрипт)**

Примеры заполнения таблиц тестовыми данными приведены в SQL-скрипте, прилагаемом к проекту (library\_db\_project.sql). Эти данные иллюстрируют структуру таблиц и взаимосвязи между ними, а также позволяют протестировать основные запросы и представления.

**8. SQL-скрипт для создания БД и представлений**

Полный SQL-скрипт для создания всех таблиц, связей, индексов и представлений, а также для вставки начальных данных, находится в файле library\_db\_project.sql.

**9. Возможные пути развития системы**

* **Управление пользователями и ролями:** Разделение прав доступа (администратор, библиотекарь).
* **Система бронирования книг:** Возможность для читателей зарезервировать книгу.
* **Автоматический расчет штрафов:** Более сложная логика расчета пени за просрочку.
* **Модуль отчетности:** Генерация различных статистических отчетов (популярные книги, активность читателей, задолженности и т.д.).
* **Интеграция со сканером штрих-кодов:** Для быстрого поиска книг и читательских билетов.
* **API для внешних интеграций:** Например, для веб-сайта библиотеки.
* **История изменений статусов экземпляров:** Для более детального аудита.

**10. Заключение**

Разработанная структура базы данных "Библиотека" обеспечивает основу для создания полнофункциональной автоматизированной системы учета. Она учитывает основные потребности библиотечной деятельности, позволяет хранить данные в структурированном виде, обеспечивает их целостность и предоставляет возможности для анализа и отчетности через представления. Дальнейшее развитие системы может включать расширение функционала и интеграцию с другими информационными системами.