Министерство образования Республики Беларусь

Учреждение образования  
 БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
 ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Рефакторинг и оптимизация программного кода

Отчет

по лабораторной работе №2

на тему:

**РАСЧЕТ МЕТРИК ПРОЕКТА И КОДА В ОБЪЕКТНО-ОРИЕНТИРОВАННЫХ ПРОГРАММНЫХ СИСТЕМАХ КАК ИНСТРУМЕНТ РЕФАКТОРИНГА**

|  |  |  |  |
| --- | --- | --- | --- |
| Проверил | |  | А.В. Шелест |
|  | (подпись) | |  |
| зачтено |  | |  |
|  | (дата защиты) | |  |
|  |  | |  |
| Выполнил |  | | О.Д. Семерник  гр. 114302 |
|  | (подпись) | |  |

Минск, 2024

Ссылка на Github:

При разработке антиплагиатного программного обеспечения для научных статей одной из ключевых задач является создание надёжной и оптимизированной базы данных, которая будет эффективно обрабатывать и хранить большие объёмы текстовой информации. Специфика предметной области требует, чтобы система могла быстро и точно проверять статьи на заимствования, учитывая их структурные особенности, такие как абстракты, разделы, библиографии и ссылки. Это создаёт дополнительные требования к структуре базы данных, поскольку данные должны быть представлены таким образом, чтобы легко анализироваться, извлекаться и сравниваться с уже имеющимися источниками.

Проектирование базы данных для системы антиплагиата требует не только структурирования данных, но и учёта множества технических и функциональных требований. База данных должна поддерживать высокую скорость обработки, обеспечивать масштабируемость и безопасность данных, а также поддерживать различные типы запросов и фильтраций для получения детализированных отчётов. Помимо научных статей, система может обрабатывать и сопоставлять текстовые данные из других источников, включая интернет-ресурсы, научные базы данных, а также внутренние корпоративные хранилища, что предполагает интеграцию с внешними системами.

Проектирование схемы базы данных также должно учитывать удобство последующего анализа данных и поддержку новых функций, таких как автоматическое обновление источников и улучшенные методы идентификации заимствований. Спроектированная структура должна позволять пользователям легко взаимодействовать с системой, отслеживать результаты проверок, получать отчёты и сохранять их для последующего использования.

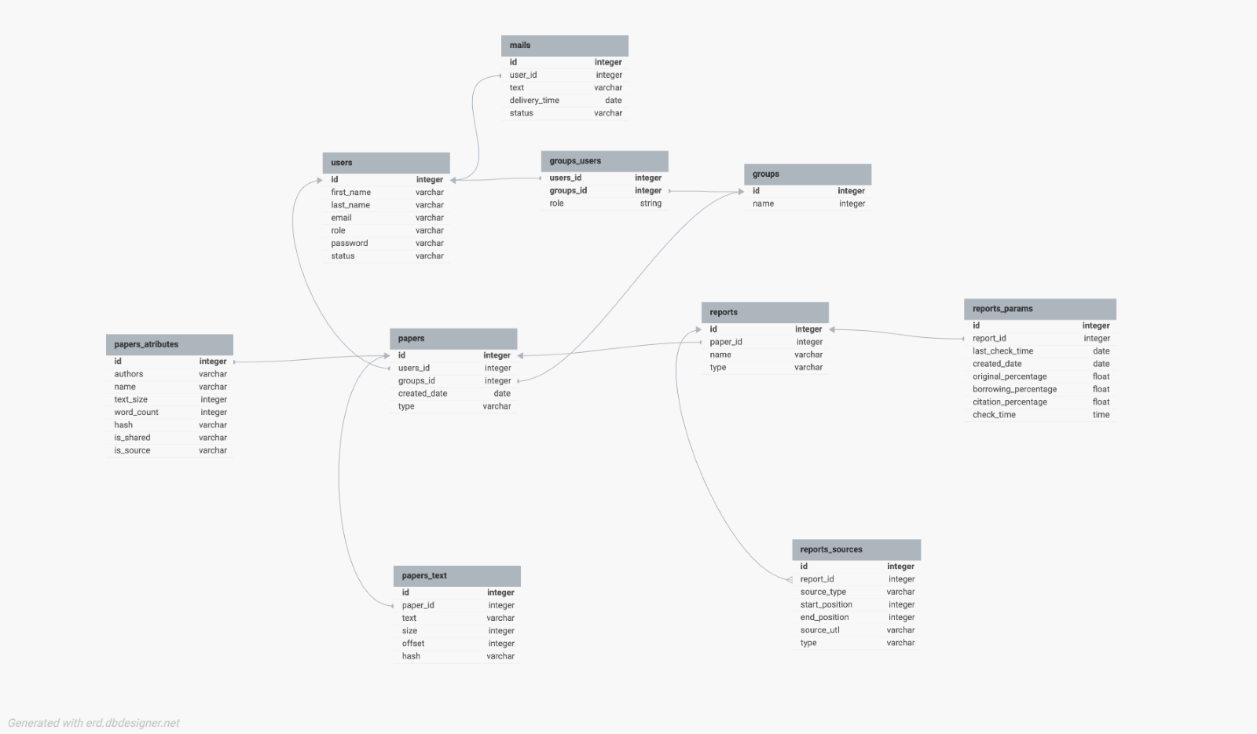


Рисунок 1 – Реляционная схема базы данных

Таблица 1.1 – Описание таблицы *users*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| first\_name | varchar | Имя пользователя |
| last\_name | varchar | Фамилия пользователя |
| email | varchar | Электронная почта пользователя |
| role | varchar | Роль пользователя |
| password | varchar | Пароль пользователя. Хранится в виде hash |
| status | varchar | Статус пользователя |

 Таблица *users* хранит информацию о пользователях системы. Каждому пользователю присваивается уникальный идентификатор *id* (первичный ключ), а также записываются его имя (*first\_name*) и фамилия (*last\_name*). В поле *email* указывается электронная почта, используемая для связи и аутентификации. Поле *role* хранит роль пользователя в системе, определяющую его права доступа. Пароль (*password*) сохраняется в виде хеша для обеспечения безопасности данных. Статус пользователя (*status*) указывает, активен ли аккаунт, заблокирован или находится в другом состоянии.

Таблица 1.2 – Описание таблицы *groups*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| name | varchar | Название группы |

Таблица *groups* используется для хранения информации о группах пользователей. Основным идентификатором каждой группы является поле *id* (первичный ключ). Поле *name* хранит название группы, что позволяет структурировать пользователей, объединяя их по общим признакам или правам в системе.

Таблица 1.3 – Описание таблицы *groups\_users*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| user\_id | integer | Внешний ключ |
| group\_id | integer | Внешний ключ |
| role | varchar | Роль пользователя в группе |

Таблица *groups\_users* реализует связь между таблицами *users* и *groups*, представляя собой связь многие ко многим. Поле *user\_id* выступает внешним ключом, указывающим на уникальный идентификатор пользователя из таблицы *users*, а *group\_id* ссылается на идентификатор группы из таблицы *groups*. Поле *role* определяет роль конкретного пользователя в данной группе, позволяя различать права пользователей внутри одной группы.

Таблица 1.4 – Описание таблицы *mails*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| user\_id | integer | Внешний ключ |
| text | varchar | Текст сообщения |
| delivery\_time | datetime | Время отправки письма |
| status | varchar | Статус письма |

Таблица *mails* предназначена для хранения сообщений, отправляемых пользователям системы. Каждое сообщение идентифицируется уникальным *id* (первичный ключ) и связано с пользователем через поле *user\_id*, являющееся внешним ключом. Поле *text* содержит текст сообщения, а *delivery\_time* фиксирует время отправки. Поле *status* указывает текущий статус письма, например, доставлено или ожидает отправки.

Таблица 1.5 – Описание таблицы *papers*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| user\_id | integer | Внешний ключ |
| groups\_id | integer | Внешний ключ |
| created\_date | datetime | Дата загрузки |
| type | datetime | Тип статьи (личная или групповая) |

Таблица *papers* хранит данные о загруженных статьях. Уникальный идентификатор статьи – поле *id*, а поле *user\_id* связывает статью с пользователем, который её загрузил. *groups\_id* указывает на принадлежность статьи к определённой группе, если она была загружена в групповом контексте. В *created\_date* сохраняется дата загрузки, а *type* указывает тип статьи, например, индивидуальная или групповая.

Таблица 1.6 – Описание таблицы *papers\_atributes*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ. Внешний ключ |
| authors | varchar | Название автора |
| name | varchar | Название статьи |
| word\_count | integer | Количество слов в статье |
| hash | varchar | Хеш-сумма статьи для идентификации повторно загруженных статей |
| is\_shared | varchar | Является ли общей |
| is\_source | varchar | Является ли источником для проверки на заимствования в других статьях |
| text\_size | integer | Размер текста |

  Таблица *papers\_attributes* содержит дополнительные атрибуты статей, такие как *word\_count* (количество слов в статье) и *hash*, представляющий собой хеш-сумму текста для ускорения поиска дублирующих материалов. Поле *is\_shared* указывает, доступна ли статья для общего использования, а *is\_source* определяет, является ли статья источником для проверки других материалов. *text\_size* хранит размер текста в байтах, что может быть полезно для оценки объёма данных.

Таблица 1.7 – Описание таблицы *papers\_text*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| paper\_id | integer | Внешний ключ |
| text | varchar | Часть текста статьи |
| size | integer | Размер текста |
| offset | integer | Отступ от начала статьи |
| hash | varchar | Хеш-сумма статьи для быстрого поиска частей статьи при проверке |

Таблица *papers\_text* содержит текстовые данные статей. Поле *paper\_id* связывает текст с определённой статьёй из таблицы *papers*. Поле *text* содержит блок текста статьи, а *offset* указывает смещение от начала текста, позволяя хранить длинные тексты частями. Поле *hash* содержит хеш-сумму каждого блока текста для ускоренной проверки на совпадения.

Таблица 1.8 – Описание таблицы *reports*

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля |  | Тип поля | Назначение |
| id |  | integer | Первичный ключ |
| paper\_id |  | integer | Внешний ключ |
| name |  | varchar | Название отчета |
| type |  | varchar | Тип отчета |

Таблица *reports* содержит информацию о сформированных отчётах по проверке статей. Поле *id* является первичным ключом, идентифицирующим каждый отчёт, а *paper\_id* выступает внешним ключом, связывающим отчёт с проверяемой статьёй. Поле *name* хранит название отчёта, а *type* указывает его тип, например, отчёт о заимствованиях или цитировании.

Таблица 1.9 – Описание таблицы *reports\_params*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| report\_id | integer | Внешний ключ |
| last\_check\_date | varchar | Дата последней проверки |
| created\_date | varchar | Дата создания отчета |
| original\_ percentage | float | Процент оригинальности |
| borrowing\_ percentage | float | Процент заимствования |
| citation\_percentage | float | Процент цитирования |
| check\_time | datetime | Время проверки |

Таблица *reports\_params* предназначена для хранения параметров каждого отчёта, таких как процент оригинальности (*original\_percentage*), процент заимствований (*borrowing\_percentage*) и процент цитирований (*citation\_percentage*). Поле *report\_id* служит внешним ключом, связывая параметры с конкретным отчётом. В таблице также хранятся даты создания (*created\_date*) и последней проверки (*last\_check\_date*), а также время проверки (*check\_time*), что позволяет отслеживать историю проверок и их результаты.

Таблица 1.10 – Описание таблицы *reports\_sources*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип поля | Назначение |
| id | integer | Первичный ключ |
| report\_id | integer | Внешний ключ |
| type | varchar | Тип (цитирование или заимствование) |
| source\_type | varchar | Тип источника (Интернет или локальная база) |
| start\_position | integer | Начало совпадения |
| end\_position | integer | Конец совпадения |
| source\_url | varchar | Ссылка на источник |

Таблица *reports\_sources* содержит данные об источниках, на которые ссылается проверяемая статья. Поле *report\_id* является внешним ключом и связывает источник с конкретным отчётом. Поля *type* и *source\_type* определяют, является ли источник заимствованием или цитированием, а также его происхождение (интернет или локальная база). Поля *start\_position* и *end\_position* указывают позиции совпадения в тексте статьи, а *source\_url* содержит ссылку на источник, что позволяет пользователю просмотреть оригинал текста.

Данная схема базы данных находится в третьей нормальной форме (3НФ), поскольку каждая таблица имеет первичный ключ, позволяющий уникально идентифицировать данные. Это гарантирует, что любые данные в таблице можно найти однозначно, используя заданный ключ, что исключает проблемы дублирования информации и упрощает поиск.

Все атрибуты в таблицах атомарные, что означает отсутствие необходимости в преобразованиях при вставке или извлечении данных. Это также устраняет вероятность дублирования данных, поскольку каждый атрибут содержит минимально неделимое значение. Благодаря этому данные хранятся в ясной и удобной для обработки форме, что способствует поддержанию целостности информации в системе.

Кроме того, в структуре базы данных отсутствуют повторяющиеся группы данных и атрибуты с одинаковым смыслом. Это гарантирует, что каждый столбец уникален по своему значению и назначению, что упрощает понимание структуры базы данных и снижает вероятность логических ошибок, связанных с обработкой данных. Отсутствие избыточных данных делает модель более оптимальной и устойчивой к изменениям.

Также все неключевые атрибуты таблиц непосредственно зависят от первичного ключа и не имеют «неявных» зависимостей друг от друга. Благодаря этому данные в базе не подвержены аномалиям при вставке, удалении или обновлении, что означает, что изменение одних данных не повлияет на целостность других. Таким образом, данная схема базы данных в 3НФ способствует поддержанию целостности данных и оптимизирует работу с ними.