МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«База данных для программного средства «Библиотека» с использованием технологии средства диагностики dbForge Monitor»

Выполнил студент Кравченко Кирилл Владимирович

(Ф.И.О.)

Руководитель проекта ст. п. Блинова Е. A. (учен. степень, звание, должность, Ф.И.О., подпись)

Заведующий кафедрой к.т.н., доц. Смелов В.В.

(учен. степень, звание, должность, Ф.И.О., подпись)

Консультант: ст. п. Блинова Е. A.

(учен. степень, звание, должность, Ф.И.О., подпись)

Нормоконтролер: ст. п. Блинова Е. A.

(учен. степень, звание, должность, Ф.И.О., подпись)

Курсовой проект защищен с оценкой

Минск 2021

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc90986412)

[**1. Постановка задачи** 4](#_Toc90986413)

[**2. Разработка модели** 5](#_Toc90986414)

[**3. Разработка необходимых объектов** 7](#_Toc90986415)

[**3.1.** **Роли и пользователи** 7](#_Toc90986416)

[**3.2. Таблицы** 8](#_Toc90986417)

[**3.3. Представления** 10](#_Toc90986418)

[**3.4. Индексы** 10](#_Toc90986419)

[**3.5. Триггеры** 11](#_Toc90986420)

[**3.6. Хранимые процедуры и функции** 12](#_Toc90986421)

[**4. Описание процедур импорта и экспорта данных** 13](#_Toc90986422)

[**4.1. Импорт данных** 13](#_Toc90986423)

[**5. Тестирование производительности** 14](#_Toc90986424)

[**6.** **Средство диагностики dbForge Monitor** 16](#_Toc90986425)

[**7. Руководство пользователя** 18](#_Toc90986426)

[**ЗАКЛЮЧЕНИЕ** 19](#_Toc90986427)

[**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ** 20](#_Toc90986428)

[**Приложение А** 21](#_Toc90986429)

[**Приложение B** 22](#_Toc90986430)

# **ВВЕДЕНИЕ**

Данный проект представляет собой back-end приложение для контроля над базой данных программного средства «Библиотека» с использованием технологии средства диагностики dbForge Monitor. Приложение реализовано за счёт запросов с помощью набора инструментов для тестирования API (**Postman) + Express.js**. **Назначение** Postman — тестирование отправки запросов с клиента на сервер и получения ответа от сервера.

База данных приложения спроектирована в СУБД Microsoft SQL Server и содержит 2 роли: администратор, студент. Доступ к данным осуществляется через соответствующие процедуры.

Целью проекта является учебно-познавательная деятельность по освоению навыков проектирования базы данных.

# **Постановка задачи**

Целью данной работы является проектирование базы данных для программного средства «Библиотека» с использованием технологии средства диагностики dbForge Monitor. В качестве модели данных следует использовать реляционную модель. Проектирование необходимо произвести таким образом, чтобы конечные данные соответствовали общим требованиям к информации в базе данных [1].

Функционал приложения должен позволять выполнять следующие функции:

Для администратора:

* добавление книги / студента,
* просмотр информации о книгах / студентах,
* изменение информации о книгах / студентах,
* удаление книги / студента.

Для студента:

* бронирование книги,
* возврат книги.

Для реализации функционала необходимо разработать ряд объектов базы данных. К таким объектам относятся:

* роли,
* пользователи,
* таблицы,
* представления,
* хранимые процедуры,
* функции,
* триггеры,
* индексы.

# **Разработка модели**

Задачей курсового проекта является проектирование базы данных для программного средства «Библиотека». Опираясь на тематику, можно выделить основные сущности, которые должны быть реализованы в виде таблиц:

* книга,
* автор,
* жанр,
* студент,
* заказ,
* возврат,
* журнал событий.

Каждая из сущностей c набором ее свойств будет рассмотрена далее.

Книга. Имеет следующие свойства:

* название книги,
* дата публикации книги,
* цена книги,
* количество книг.

Автор. Имеет следующие свойства:

* + - имя автора,
    - фамилия автора.

Жанр. Имеет следующие свойства:

* название жанра.

Студент. Имеет следующие свойства:

* имя студента,
* фамилия студента,
* номер студенческого,
* факультет,
* семестр,
* телефонный номер,
* e-mail,
* дата регистрации,
* дата блокировки,
* статус.

Заказ. Имеет следующие свойства:

* номер студенческого заказчика,
* название книги,
* количество книг,
* дата заказа.

Возврат. Имеет следующие свойства:

* номер студенческого заказчика,
* название книги,
* количество заказанных всего книг,
* количество возвращаемых книг,
* дата возврата.

Запись журнала. Необходима для фиксирования действий администратора. Имеет следующие свойства:

* операция,
* время выполненной операции.

Диаграмма базы данных представлена в приложении А.

# **Разработка необходимых объектов**

Для реализации поставленной задачи были разработаны следующие объекты базы данных:

* роли,
* пользователи,
* таблицы,
* представления,
* индексы,
* триггеры,
* хранимые процедуры,
* функции.

Далее будет приведено описание каждого созданного объекта.

## **Роли и пользователи**

Для разграничения возможностей администратора и клиента базы данных были созданы две роли:

* aдминистратор,
* cтудент.

Для реализации вышеупомянутый ролей были созданы два пользователя:

* library\_admin,
* library\_client.

Администратор имеет доступ к созданию, редактированию и удалению объектов БД.

Скрипт создания приведен ниже.

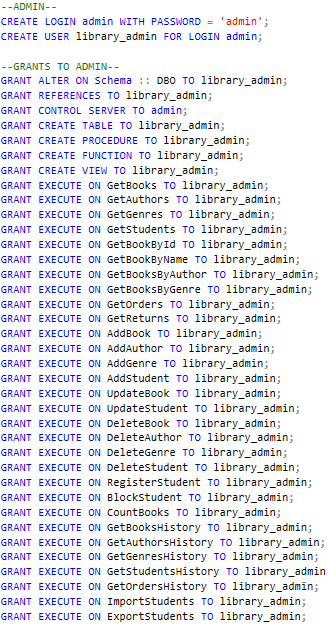


Рисунок 3.1.1 – «Создание роли и пользователя для админа»

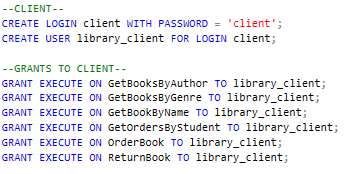


Рисунок 3.1.2 – «Создание роли и пользователя для студента»

## **3.2. Таблицы**

Таблицы являются отражением математических моделей на объекты базы данных. Список сущностей и связанных с ними свойств приведен в главе №2.

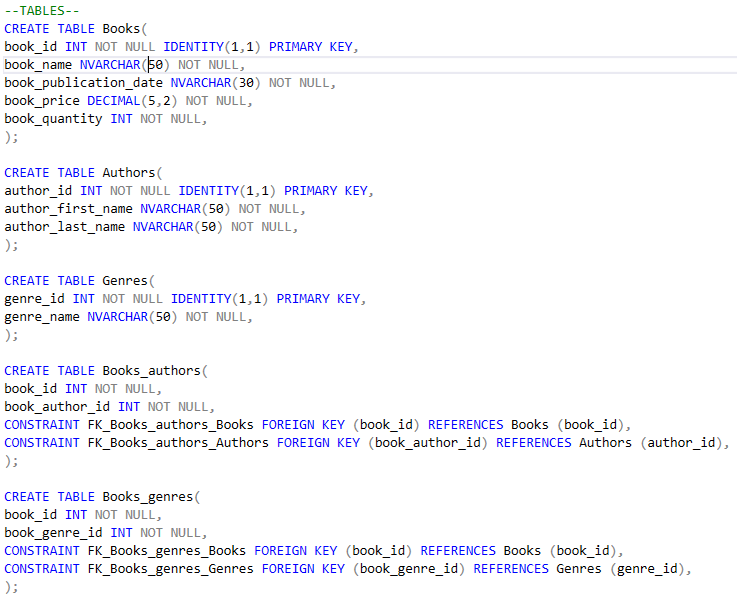
 Листинги скриптов создания таблиц приведены ниже.

Рисунок 3.2.1 – «Создание таблиц для книг, авторов и жанров»

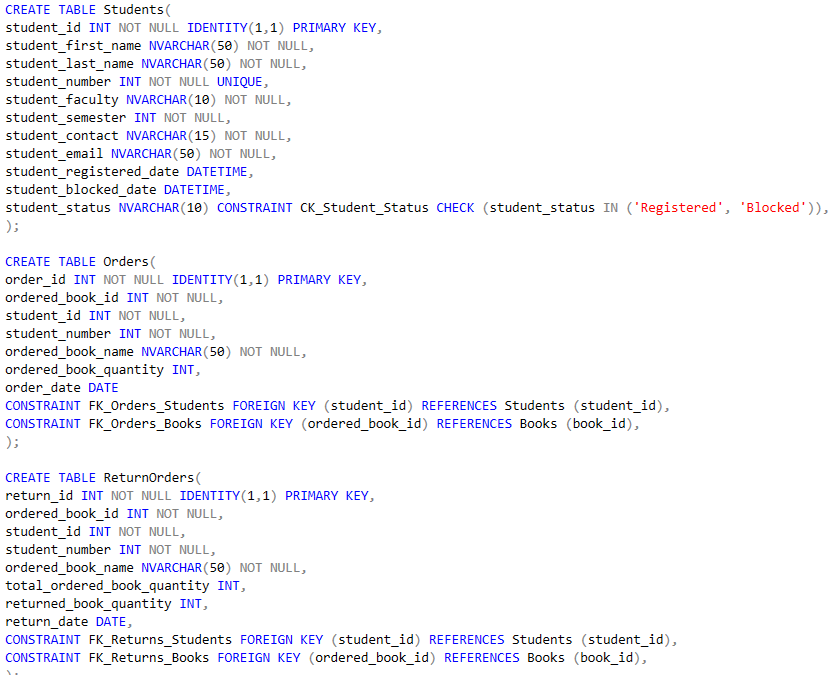


Рисунок 3.2.2 – «Создание таблиц для студентов, заказов и возвратов»

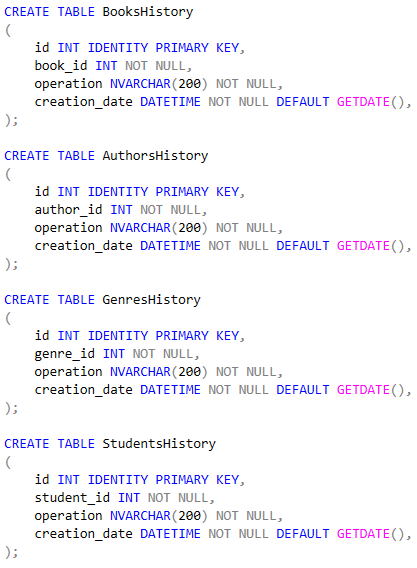


Рисунок 3.2.3 – «Создание таблиц для журнала событий»

## **3.3. Представления**

В результате работы были созданы представления на таблицы книг, авторов, жанров, студентов, заказов и возвратов.

Листинг скриптов создания представлений приведен ниже.

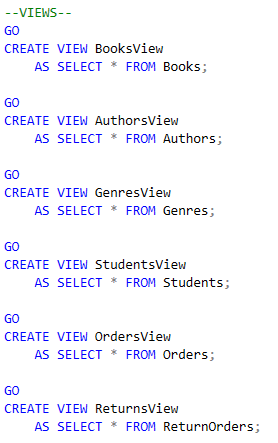


Рисунок 3.3.1 – «Создание представлений»

## **3.4. Индексы**

Индексы необходимы для повышения производительности выполнения поиска данных. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. Для оптимальной производительности запросов индексы обычно создаются на тех столбцах таблицы, которые часто используются в запросах. Однако увеличение числа индексов замедляет операции добавления, обновления, удаления строк таблицы, поскольку при этом приходится обновлять сами индексы [1].

Учитывая вышесказанное, для повышения производительности запросов на получение книги по имени, был создан индекс в таблице «Books». Листинг скриптов приведен ниже.

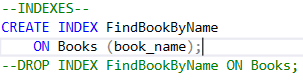


Рисунок 3.4.1 – «Создание индекса»

## **3.5. Триггеры**

Для фиксирования действий администратора в журнал событий были созданы триггеры.

Листинги создания триггеров приведены ниже.

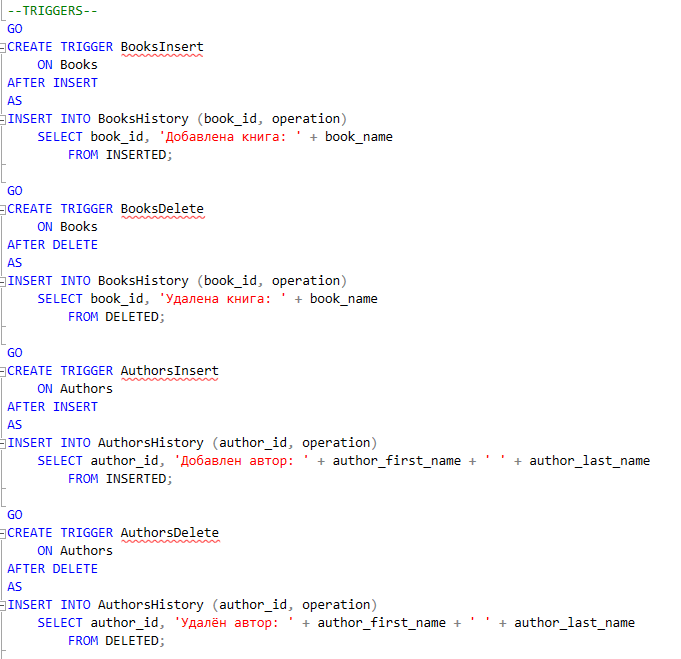


Рисунок 3.5.1 – «Создание триггеров для книг и авторов»

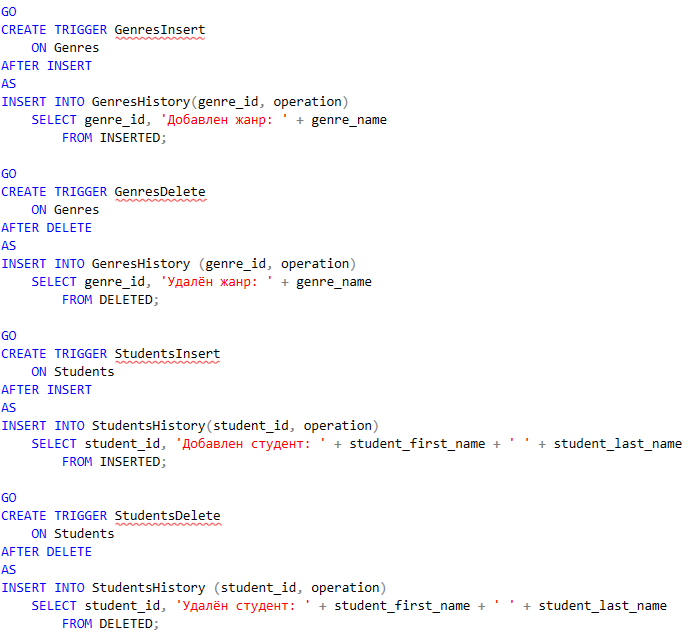


Рисунок 3.5.2 – «Создание триггеров для жанров и студентов»

## **3.6. Хранимые процедуры и функции**

Доступ к данным осуществляется через хранимые процедуры и функции. Хранимая процедура – это блок SQL-команд, который компилируется и хранится на сервере. Процедуры имеют входные и выходные параметры с помощью которых выполняется поиск, обработка и возврат данных.

К админу относятся следующие процедуры и функции:

* Получение всех книг
* Получение всех авторов
* Получение всех жанров
* Получение книг с фильтрацией по автору, жанру или названию
* Получение всех студентов
* Добавление книги
* Добавление автора
* Добавление жанра
* Добавление студента
* Удаление книги
* Удаление автора
* Удаление жанра
* Удаление студента
* Импорт студентов
* Экспорт студентов
* Получение списка заказов
* Получение списка возвратов
* Получение заказов одного студента
* Получение количества книг для каждого автора

К студенту относятся следующие процедуры:

* Получение всех книг
* Получение книг с фильтрацией по автору, жанру или названию
* Заказ книги
* Возврат книги

Листинги хранимых процедур и функций описаны в приложении B.

# **Описание процедур импорта и экспорта данных**

Для импорта и экспорта данных был выбран формат XML и таблица «Students».

Была создана директория xml, где находятся XML-файлы для импорта и экспорта.

## **4.1. Импорт данных**

Листинг процедуры импорта данных приведен ниже.

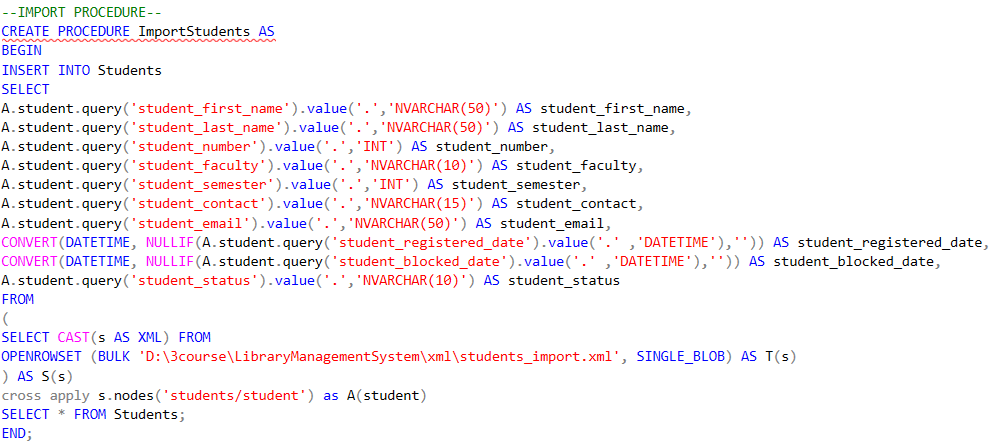


Рисунок 4.1 – «Создание процедуры импорта данных»

**4.2. Экспорт данных**

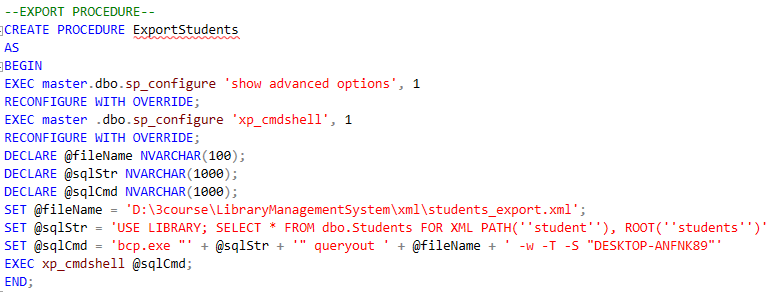
Листинг процедуры экспорта данных приведен ниже.

Рисунок 4.2 – «Создание процедуры экспорта данных»

# **Тестирование производительности**

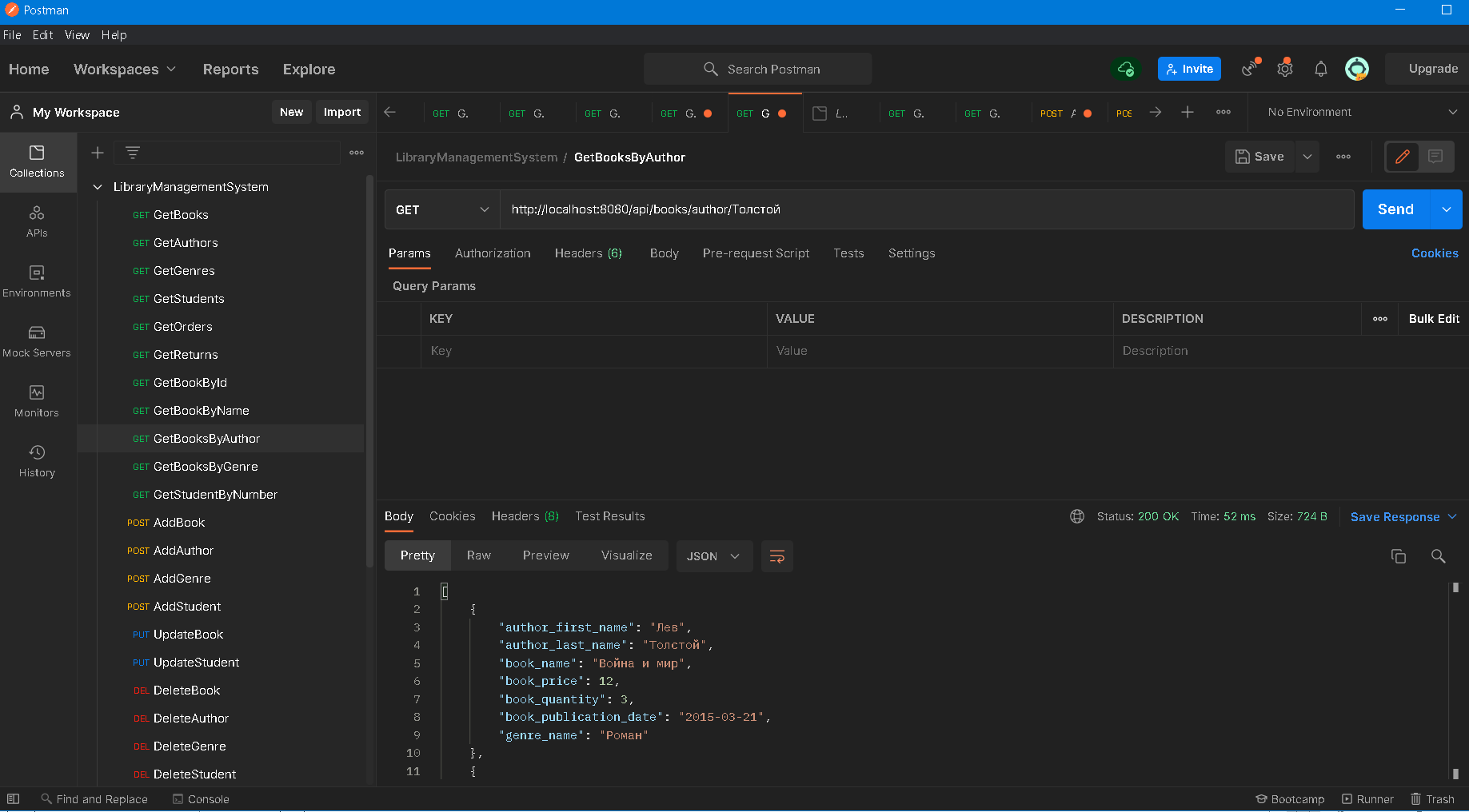
****Тестирование производительности приложения реализовано за счёт запросов к БД с помощью набора инструментов для тестирования API (**Postman) и Express.js**.

Рисунок 5.1 – «Тестирование приложения с помощью Postman»

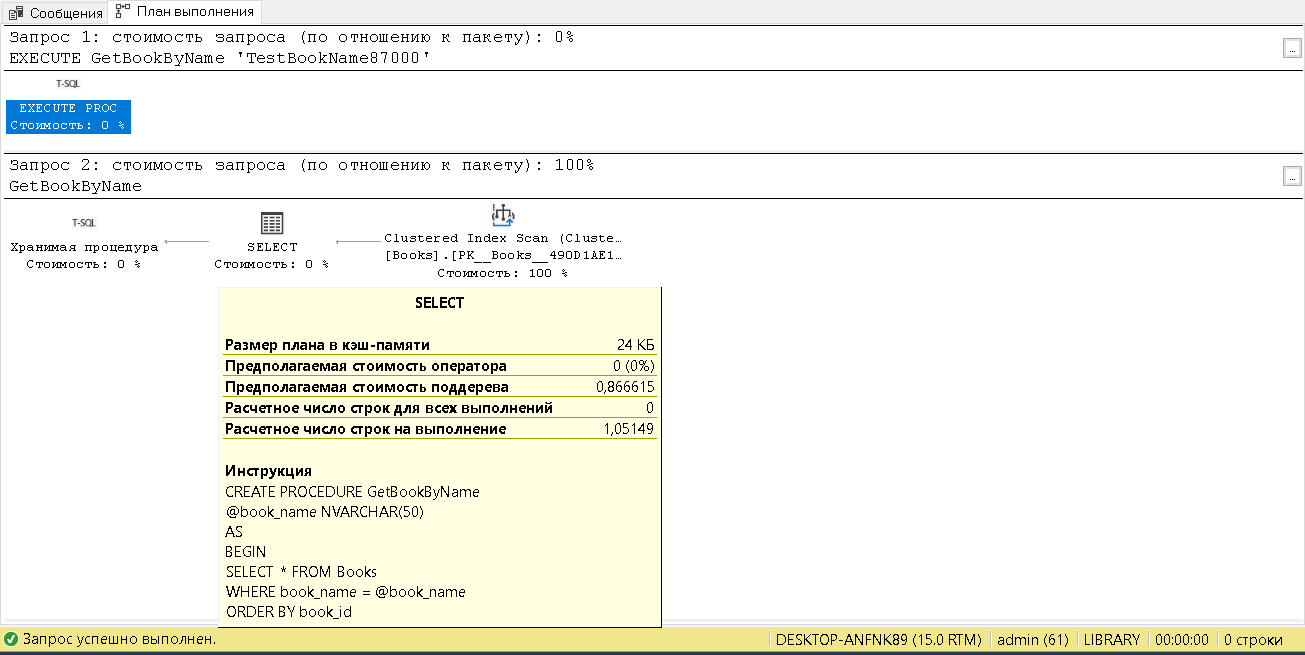
В процессе разработки был создан индекс, а также таблица «Books» была заполнена одной сотней тысяч строк. Для определения качества и необходимости созданных индексов будут выполнены запросы к данной таблице с и без индексов, а потом будет проведено сравнение и анализ полученных результатов. Тесты будут выполнены на стороне конечного клиента.

Рисунок 5.2 – «Тестирование без индекса»

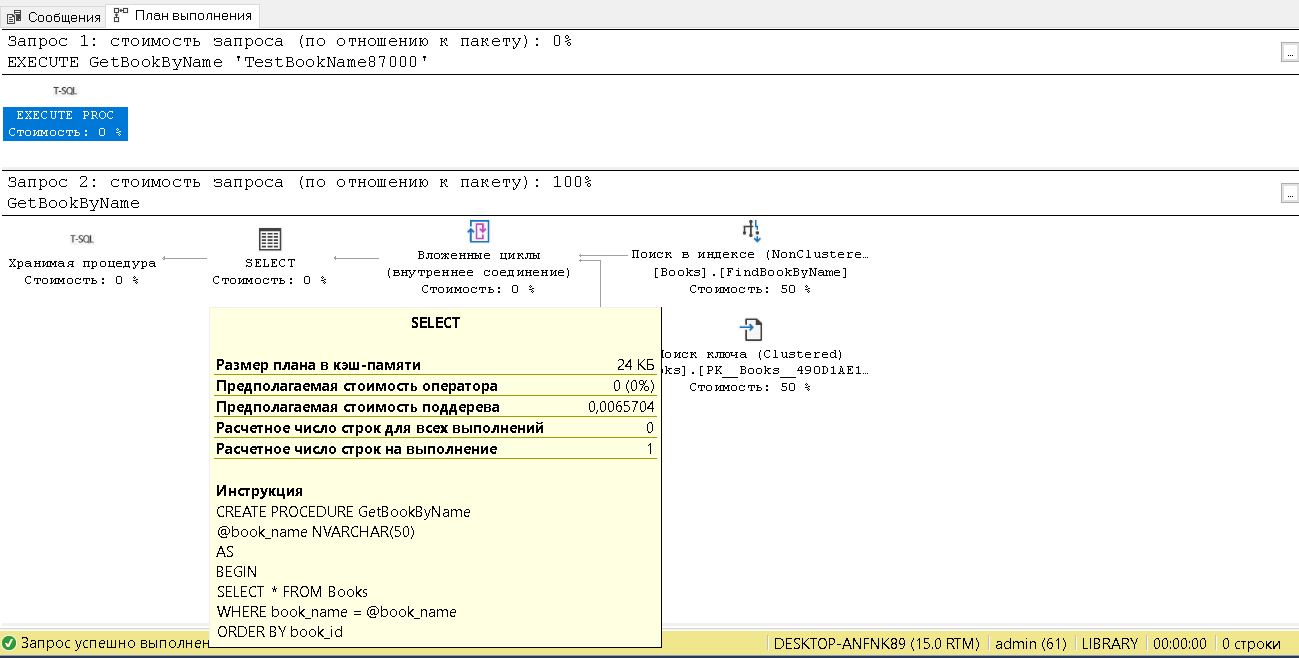


Рисунок 5.3 – «Тестирование с индексом»

Исходя из полученных результатов можно подвести итог, что без индексов запросы выполняются намного дольше, нежели если индексы есть. Данный результат считаю приемлемым, а задачу по оптимизации выполненной.

## **Средство диагностики dbForge Monitor**

В качестве дополнительной технологии было выбрано использование технологии средства диагностики dbForge Monitor.

dbForge Monitor [2] - бесплатная надстройка для Microsoft SQL Server Management Studio, которая позволяет отслеживать и анализировать производительность SQL-сервера. Инструмент включает в себя разнообразные инструменты для диагностики SQL Server, включая совокупность показателей производительности SQL, позволяющих увидеть полное и четкое представление обо всех процессах, происходящих на сервере, выявить наиболее дорогие запросы, задачи ожидания и узкие места производительности. С помощью dbForge Monitor вы сэкономите время на анализ производительности SQL Server.

Панель инструментов:

На вкладке «Обзор» этого инструмента вы можете увидеть, что на самом деле происходит на вашем SQL Server с разных ракурсов.

Богатая коллекция показателей для универсального мониторинга производительности SQL Server включает в себя:

* Использование процессора
* Утилизация памяти
* Деятельность диска
* Чтение и запись задержек
* Задержка ввода-вывода.

Ввод / вывод данных:

С вкладкой данных ввода-вывода инструмента вы можете легко просмотреть статистику операций чтения и записи для каждого файла базы данных.

В частности, вы можете отслеживать:

* Общий объем чтения и записи данных
* Количество операций ввода-вывода
* Задержка чтения / записи.

Ожидание статистики:

На вкладке «Подождите» вы можете определить ресурсы, которые замедляют работу сервера. Он отображает список ожидающих задач, встречающихся в потоках выполнения, потому что ресурсы, необходимые для выполнения запроса, были заняты или недоступны.

Лучшие запросы:

dbForge Monitor включает анализатор производительности SQL-запросов, который позволяет обнаруживать и оптимизировать самые дорогие запросы, которые перегружают систему.

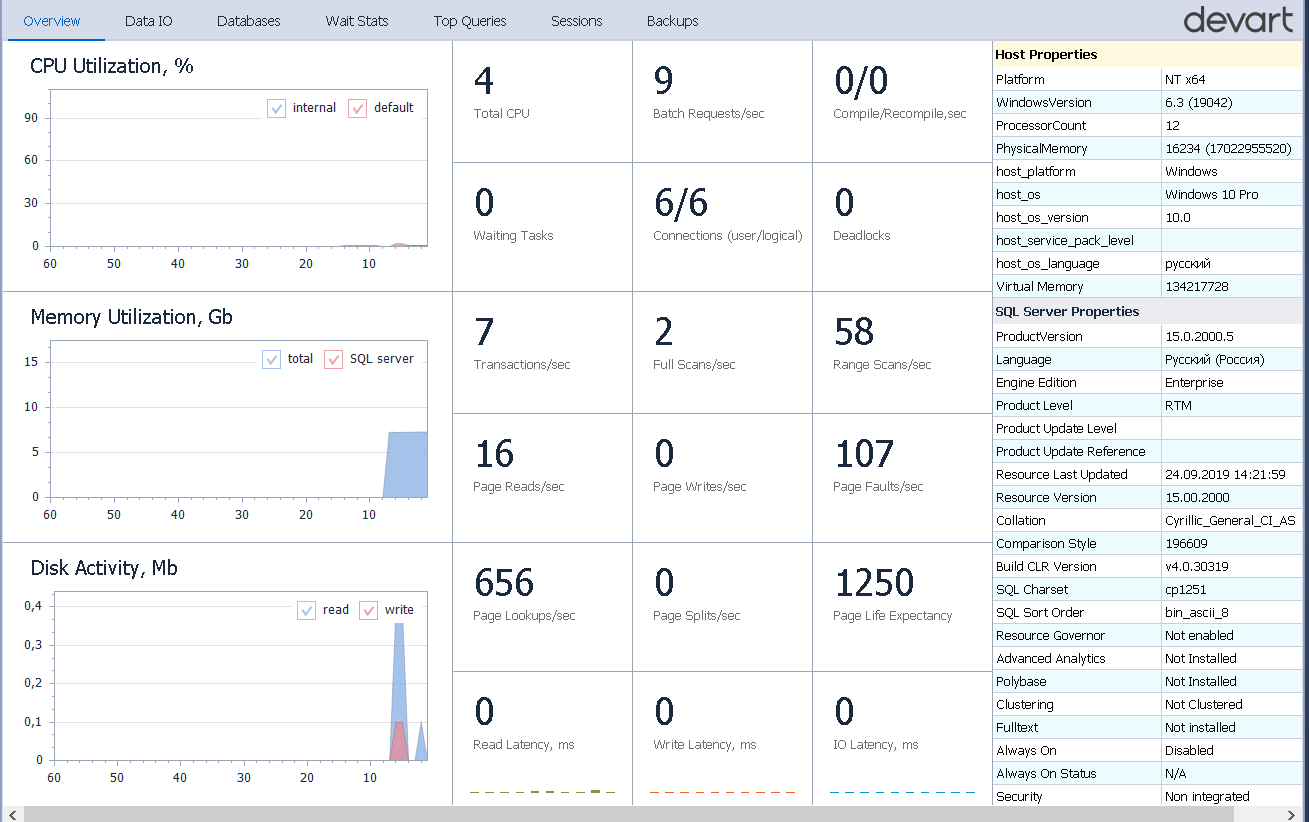


Рисунок 6.1 – «Вкладка обзор»

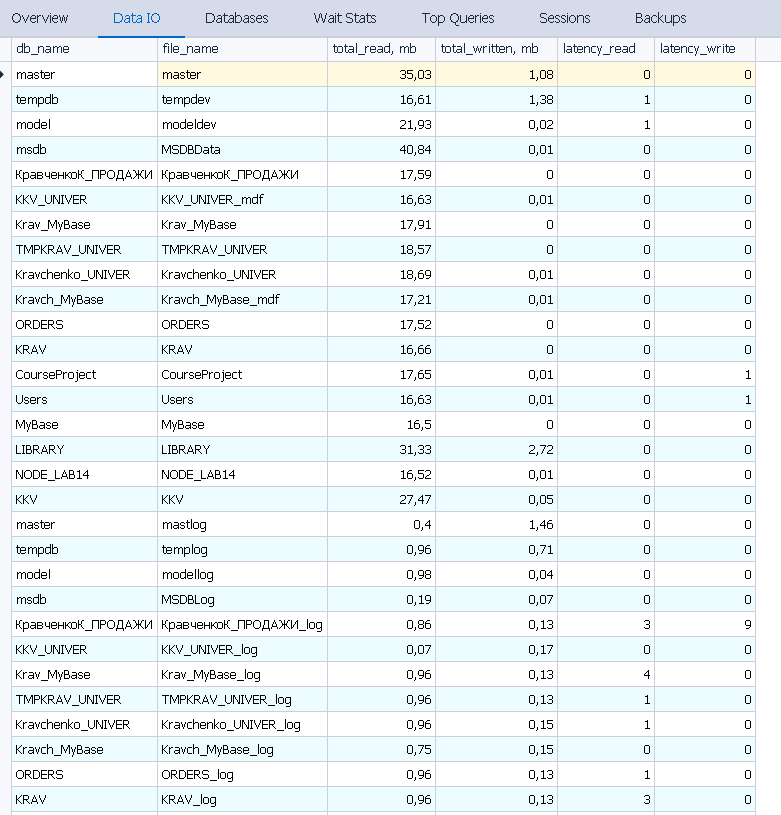
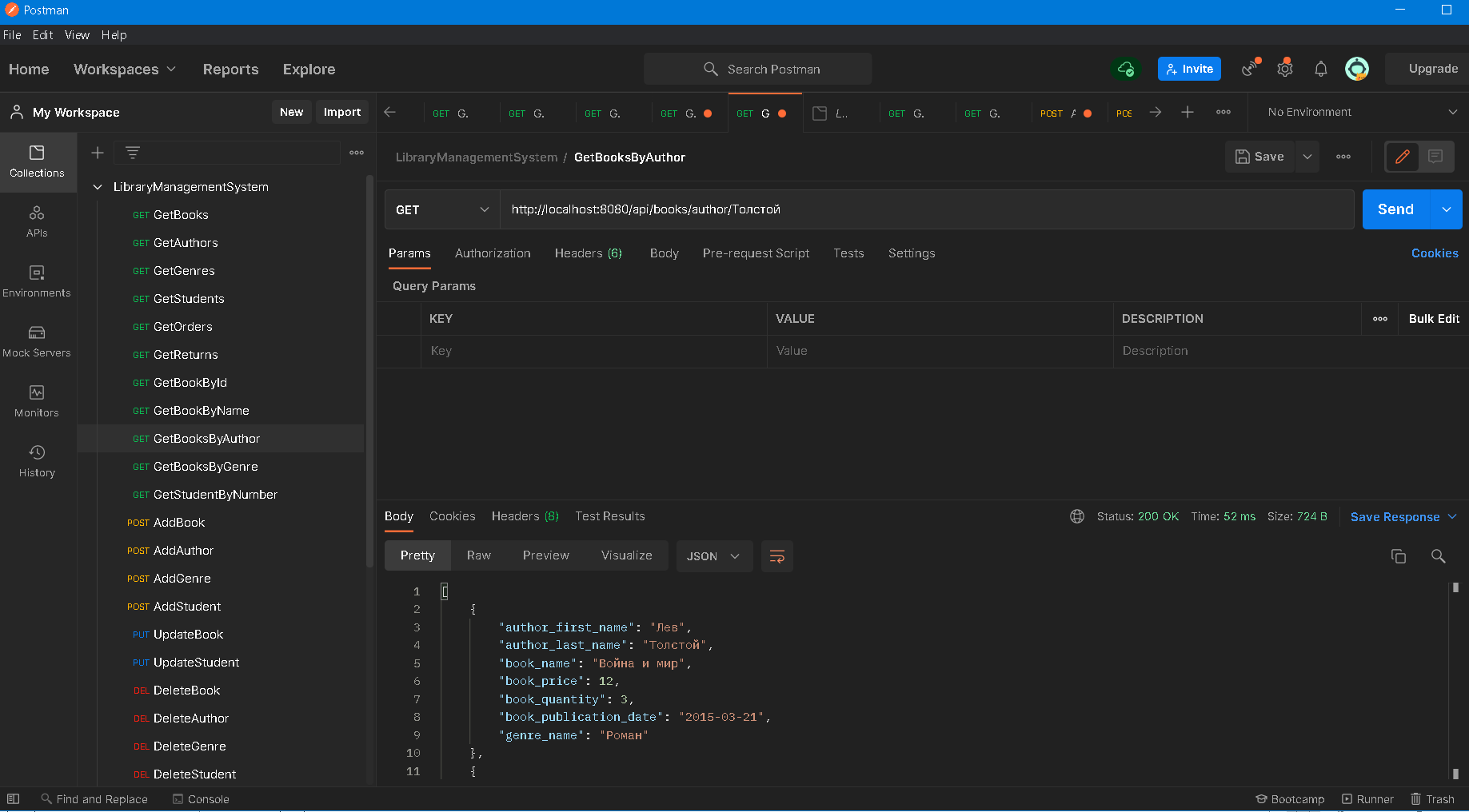


Рисунок 6.2 – «Вкладка ввод-вывод данных»

# **Руководство пользователя**

Интерфейс приложения представляет собой стандартный набор инструментов **Postman для отправки запросов к серверу и получению ответов в виде данных БД**.

Рисунки 7.1 – «Интерфейс Postman»

**ЗАКЛЮЧЕНИЕ**

В данном курсовом проекте были освоены навыки проектирования базы данных, а также спроектирована база данных для программного средства «Библиотека» с использованием технологии средства диагностики dbForge Monitor.

В ходе работы выполнены следующие задачи:

* разработана модель данных;
* разработаны необходимые объекты базы данных;
* разработаны процедуры экспорта и импорта данных;
* произведено тестирование приложения;
* применена технология средства диагностики.

В рамках технологии средства диагностики dbForge Monitor был проведён анализ и диагностика производительности SQL-сервера.

­­­

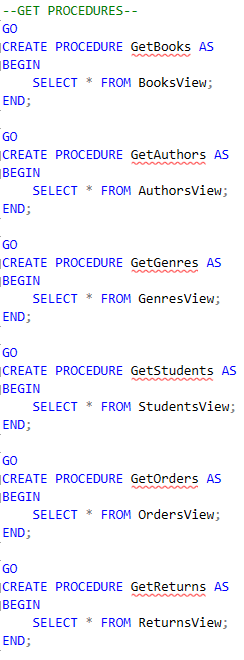
**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

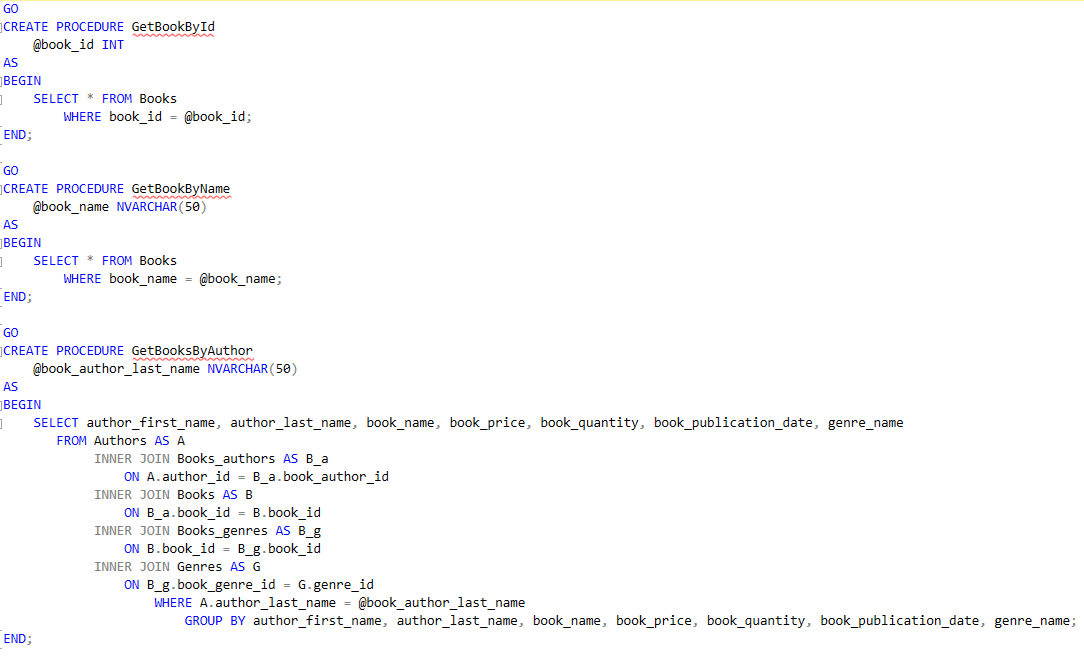
[1] metanit.com [Электронный ресурс] / Режим доступа: https://metanit.com Дата доступа: 10.12.2021

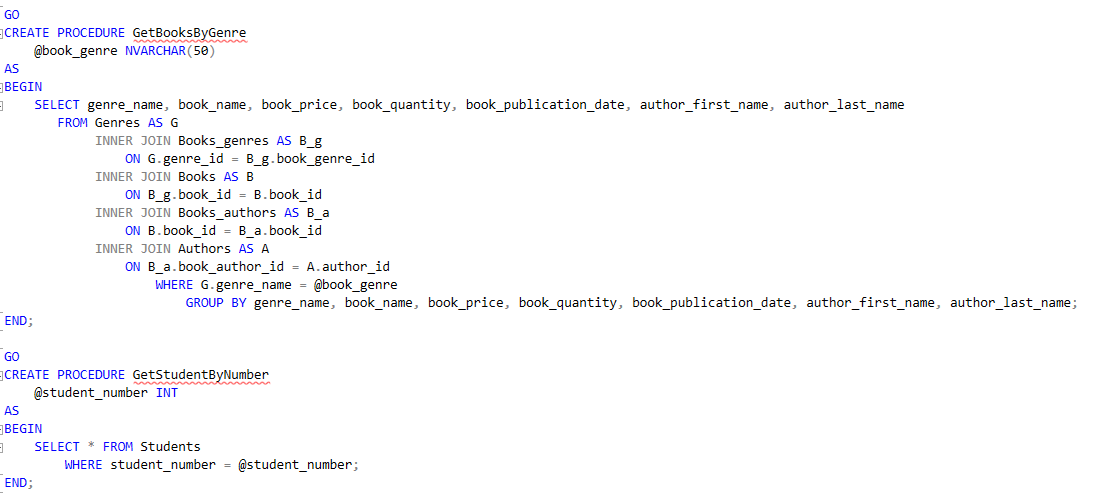
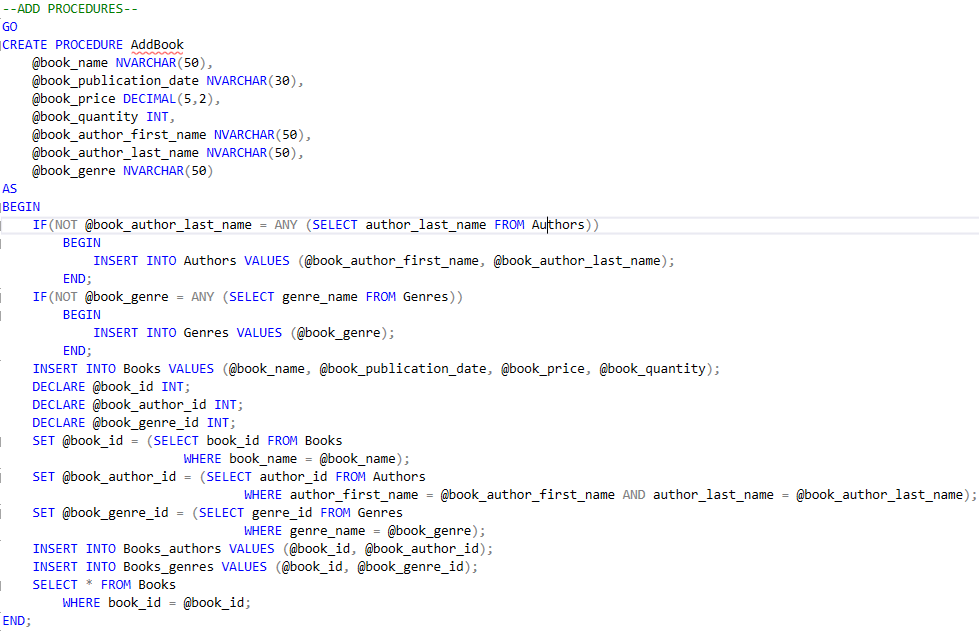
[2] softoware.org [Электронный ресурс] / Режим доступа: https://softoware.org Дата доступа 10.12.2021

# **Приложение А**

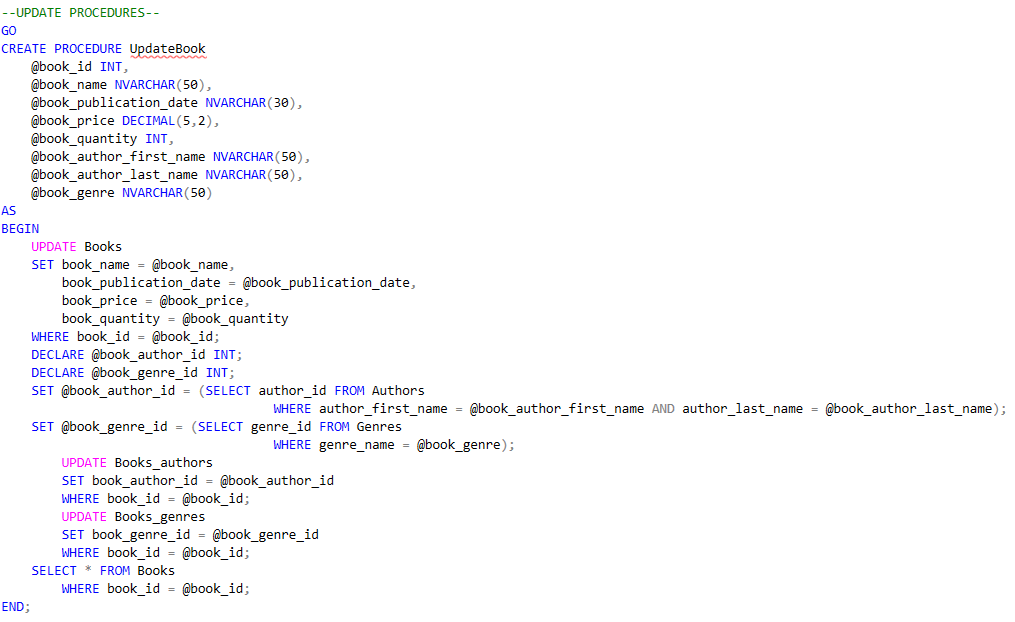
# **Приложение B**

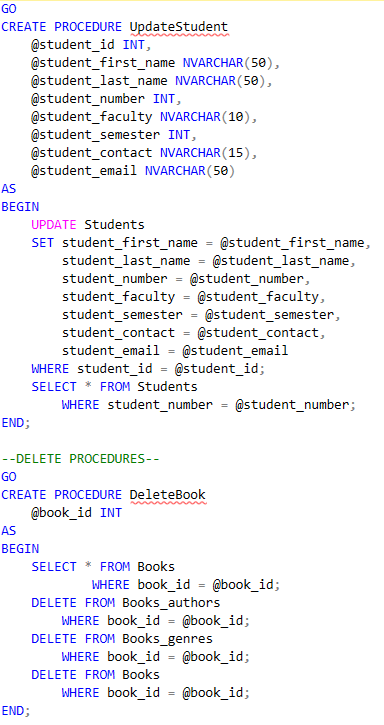


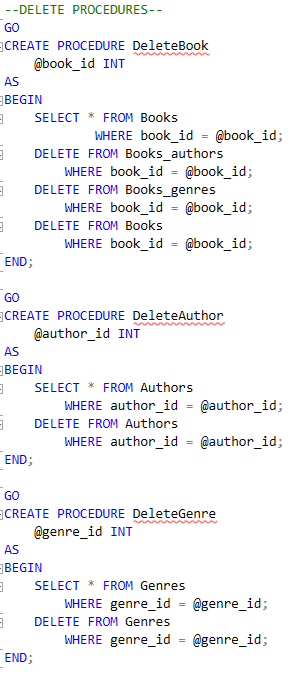


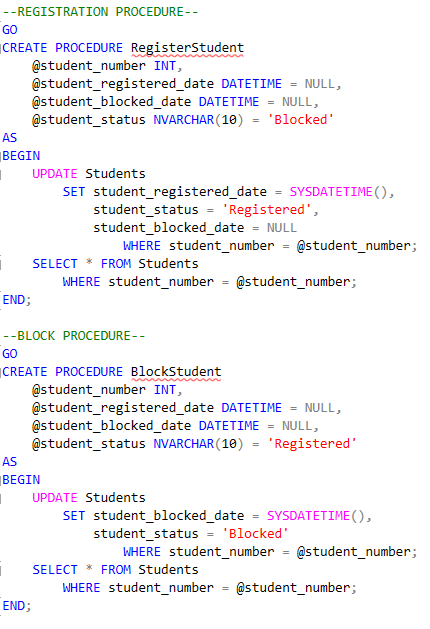


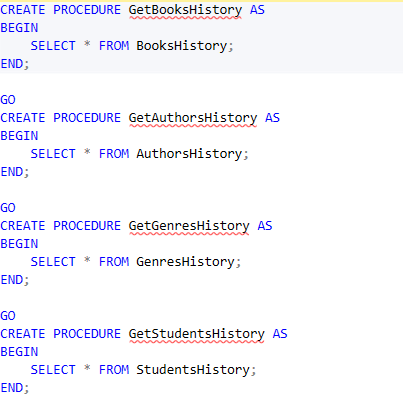


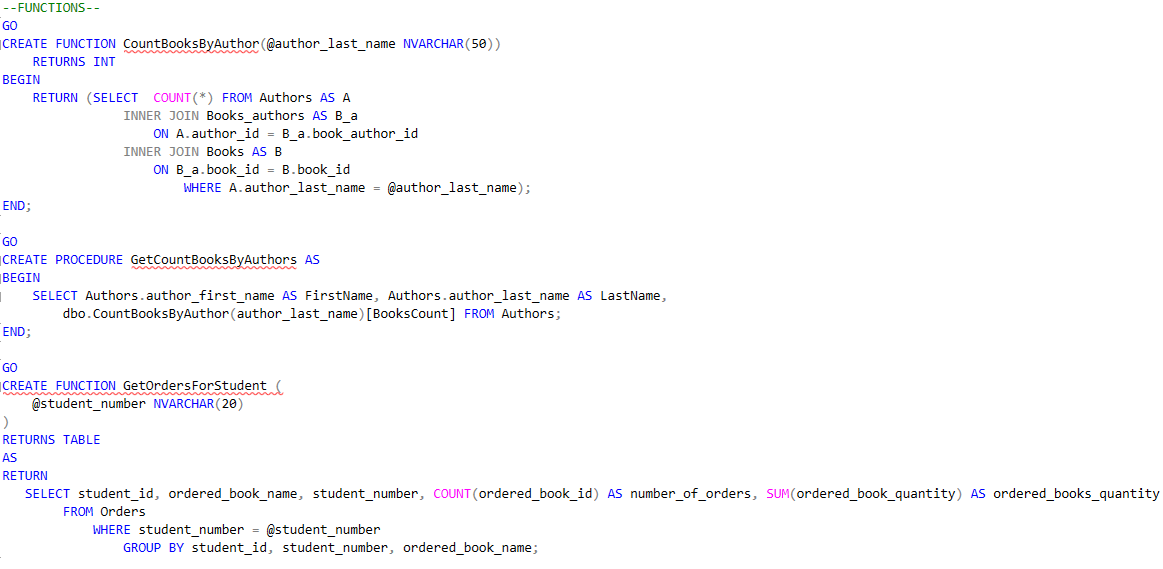














`