МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий
Кафедра Информационных систем и технологий
Специальность 6-05-0611-01 Информационные системы и технологии

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА

по дисциплине «Базы данных»
Тема «Сервис для изучения иностранных языков»

Исполнитель
студент 2 курса 1 группы _____ Авчинник Е.С.
подпись, дата

Руководитель
ассистент _____ Савельева М.Г
должность, учен. степень, ученое звание подпись, дата

Допущен(а) к защите _____ _____
_____ _____
_____ _____
дата, подпись

Курсовой проект защищен с оценкой _____

Руководитель _____ _____ Савельева М.Г.
подпись дата инициалы и фамилия

Минск 2025

Содержание

Введение

Актуальность проекта обусловлена растущей популярностью онлайн-образования и необходимостью создания надежных и функциональных информационных систем для управления образовательным процессом. Важной задачей является разработка базы данных, способной обеспечивать эффективное хранение и обработку данных о пользователях, курсах, уроках и учебных материалах, а также обеспечивать удобный доступ к информации для различных категорий пользователей.

Целью данного курсового проекта является создание базы данных для сервиса по изучению иностранных языков. Онлайн-платформы позволяют пользователям обучаться в удобное время, взаимодействовать с преподавателями и получать качественные образовательные услуги независимо от их местоположения. Однако эффективное управление таким сервисом требует надежной базы данных, способной обеспечивать хранение и обработку больших объемов информации.

Для достижения данной цели необходимо выполнить следующие задачи:

1) определить роли пользователей и их права доступа;
2) реализовать функционал управления курсами, уроками и учебными материалами;
3) организовать хранение мультимедийной информации;
4) обеспечить механизм аутентификации и авторизации пользователей;
5) реализовать импорт и экспорт данных в формате XML;
6) провести тестирование производительности и оптимизацию базы данных.

Основная проблема, которую решает данный проект, заключается в организации структурированной и безопасной системы хранения данных, обеспечивающей удобное управление учебным процессом, доступ к материалам и взаимодействие между пользователями. Также необходимо учитывать требования к масштабируемости базы данных, ее производительности и безопасности, что особенно важно при работе с большими объемами данных и мультимедийной информацией.

Разработка будет осуществляться в среде SQL Server Management Studio с использованием хранимых процедур, импортом и экспортом данных в формате XML, а также применением технологий для хранения мультимедийной информации.

1 Постановка задачи

1.1 Обзор аналогичных решений

Для проекта разработки базы данных сервиса по изучению иностранных языков целесообразно проанализировать существующие платформы, предоставляющие схожие услуги. Первым аналогом может служить Coursera (https://www.coursera.org/), крупная платформа онлайн-обучения, предлагающая курсы на различных языках и по различным дисциплинам[1].

Основные функции, реализуемые в данном аналогичном решении:

1. Хранение информации о курсах: наименование, описание, язык, уровень сложности. А также длительность курса, требования к обучающимся.
2. Управление пользователями: информация о пользователях (имя, контактные данные, предпочтительные языки обучения, прогресс пользователя, завершенные курсы, достижения)
3. Модуль сертификации: выдача сертификатов об успешном окончании курса и регистрация деталей сертификата.
4. Система преподавания: хранение данных о преподавателях: биография, квалификация, отзывы, а также связь преподавателя с курсами.
5. Функции взаимодействия: общение между студентами и преподавателями (форумы, чат) и аналитика и рекомендации на основе интересов пользователя.

Судя по сайту Coursera имеет следующую структуру базы данных.

1. Пользователи (Users):
   1) UserID: уникальный идентификатор пользователя;
   2) Name: имя пользователя;
   3) Email: электронная почта;
   4) Progress: данные о прогрессе пользователя.
2. Курсы (Courses):
   1) CourseID: уникальный идентификатор курса;
   2) Title: название курса;
   3) Description: описание курса;
   4) Language: язык курса;
   5) DifficultyLevel: уровень сложности.
3. Преподаватели (Instructors):
   1) InstructorID: уникальный идентификатор преподавателя;
   2) Name: имя преподавателя;
   3) Bio: биография.
4. Сертификаты (Certificates):
   1) CertificateID: уникальный идентификатор сертификата;
   2) UserID: идентификатор пользователя, получившего сертификат;
   3) CourseID: идентификатор курса, по которому выдан сертификат;
   4) IssueDate: дата выдачи.
5. Общение (Communication): форумы, чаты и обратная связь.
6. Рекомендации (Recommendations): логика построения рекомендаций курсов на основе предпочтений и активности пользователя.

Следующий аналог Skillspace (https://skillspace.ru/) – это образовательная платформа, которая предоставляет инструменты для создания и управления онлайн-курсами[2]. Основные функции платформы включают:

– Конструктор курсов: возможность добавлять лекции с текстом, видео и презентациями.

– Тестирование: создание тестов с автопроверкой, таймерами и статистикой.

– Домашние задания: проверка работ вручную или автоматически, обсуждение и обмен файлами.

– Вебинары и трансляции: проведение онлайн-конференций и вебинаров.

– Мобильное приложение: доступ к обучению с любых устройств.

– Аналитика обучения: отслеживание прогресса учеников и результатов обучения.

– Интеграции: поддержка более 30 интеграций с популярными платформами.

Skillspace также использует реляционную базу данных для управления образовательным процессом. Основные элементы структуры базы данных:

1. Таблица курсов (Courses):
    1) CourseID: уникальный идентификатор курса;
    2) Title: название курса;
    3) Modules: список модулей курса;
    4) Price: стоимость курса.
2. Таблица студентов (Students):
    1) StudentID: уникальный идентификатор студента;
    2) Name: имя студента;
    3) Progress: прогресс обучения.
3. Таблица заданий (Assignments):
    1) AssignmentID: уникальный идентификатор задания;
    2) CourseID: связь с таблицей курсов;
    3) StudentID: связь с таблицей студентов;
    4) Status: статус выполнения задания.
4. Таблица вебинаров (Webinars):
    1) WebinarID: уникальный идентификатор вебинара;
    2) CourseID: связь с таблицей курсов;
    3) Date: дата проведения.

Следующий аналог GeekBrains (https://geekbrains.by/) – это образовательная платформа, специализирующаяся на обучении современным профессиям, особенно в сфере IT[3]. Основные особенности платформы включают:

– Курсы по IT-направлениям: программирование, аналитика, тестирование, маркетинг, управление и дизайн.

– Индивидуальный подход: интерактивные онлайн-уроки с экспертами.

– Готовое портфолио: создание проектов в процессе обучения.

– Программы для детей и подростков: ранний старт в IT и развитие творческих навыков.

– Отзывы и рейтинги: высокие оценки пользователей за качество курсов.

GeekBrains использует реляционную базу данных для управления образовательным контентом и взаимодействием между студентами и преподавателями.

Основные элементы структуры:

1. Таблица пользователей (Users):
   1) UserID: уникальный идентификатор пользователя;
   2) Name: имя пользователя;
   3) Role: роль (студент, преподаватель).
2. Таблица курсов (Courses):
   1) CourseID: уникальный идентификатор курса;
   2) Title: название курса;
   3) Category: категория курса.
3. Таблица уроков (Lessons):
   1) LessonID: уникальный идентификатор урока;
   2) CourseID: связь с таблицей курсов;
   3) Content: содержание урока.
4. Таблица заданий (Assignments):
   1) AssignmentID: уникальный идентификатор задания;
   2) LessonID: связь с таблицей уроков;
   3) StudentID: связь с таблицей пользователей.

1.2 Требования к проекту

Целью данного проекта является разработка базы данных для онлайн-сервиса изучения иностранных языков, обеспечивающей надежное хранение, обработку и доступ к информации о курсах, уроках, пользователях и учебных материалах.

В системе предусмотрены следующие категории пользователей:

1) студент получает доступ к учебным материалам, участвует в онлайн-уроках, выполняет домашние задания, получает сертификаты, просматривает записи уроков, участвует в форумах и конкурсах, общается с преподавателями;

2) преподаватель управляет курсами, проводит онлайн-уроки, проверяет домашние задания, управляет расписанием, участвует в форумах и обсуждениях, взаимодействует со студентами;

3) менеджер управляет курсами и учебными материалами, организует образовательные мероприятия;

4) администратор управляет базой данных, проверяет документы, регулирует права доступа, управляет пользователями и учебными материалами.

База данных должна обеспечивать выполнение следующих функций:

1) хранение информации о пользователях, курсах, уроках и учебных материалах;

2) поддержка механизма аутентификации и авторизации пользователей;

3) возможность просмотра курсов, записей уроков, участия в форумах;

4) поддержка операций сортировки и фильтрации данных;

5) импорт данных из XML и экспорт данных в XML;

6) тестирование производительности базы данных на объемах не менее 100 000 записей;

7) поддержка хранения мультимедийных данных.

Проект предусматривает создание базы данных для сервиса по изучению иностранных языков, реализующей функционал для следующих категорий пользователей: студентов, преподавателей, менеджеров и администраторов.

Разработка будет осуществляться в среде SQL Server Management Studio с использованием хранимых процедур, импортом и экспортом данных в формате XML, а также применением технологий для хранения мультимедийной информации. Дополнительно будет проведено тестирование производительности системы и оптимизация структуры базы данных для повышения ее эффективности.

1.3 Вывод по разделу

Анализ существующих образовательных платформ, таких как Coursera, Skillspace и GeekBrains, позволил выявить ключевые функциональные элементы баз данных, используемых для управления онлайн-обучением. Эти платформы предоставляют возможности хранения данных о курсах, пользователях, преподавателях, заданиях и сертификатах, а также поддерживают механизмы взаимодействия и персонализированных рекомендаций.

На основе изученных аналогов сформулированы требования к разрабатываемой базе данных онлайн-сервиса изучения иностранных языков. Она должна обеспечивать надежное хранение информации о курсах, уроках и учебных материалах, поддержку аутентификации и авторизации пользователей, механизмы сортировки и фильтрации данных, а также возможность импорта и экспорта данных в XML. Важным аспектом является тестирование производительности при больших объемах данных и поддержка мультимедийного контента.

Разработка базы данных будет осуществляться в SQL Server Management Studio с применением хранимых процедур и технологий хранения мультимедийной информации. Это обеспечит высокую производительность и удобство использования системы, а также создаст эффективную инфраструктуру для онлайн-обучения.

2 Проектирование базы данных

2.1 Определение вариантов использования

Диаграммы вариантов использования (Use Case Diagrams) – это визуальное представление взаимодействия пользователей с системой, показывающее основные функциональные возможности. В моём курсовом проекте есть 4 роли: студент, преподаватель, менеджер и администратор. На рисунке 2.1 представлена диаграмма вариантов использования для роли Студент.

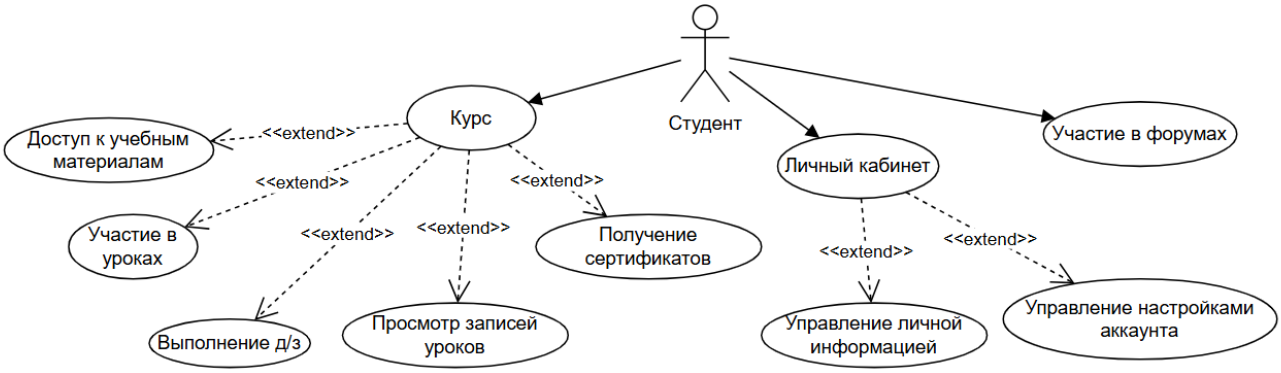

Рисунок 2.1 – Диаграмма вариантов использования для роли Студент

Для того, чтобы создать диаграмму вариантов использования нужно сперва определить акторов – пользователей, взаимодействующих с системой. Затем создать варианты использования – функциональные возможности, которые должны выполняться.

В таблице 2.1 представлены прецеденты роли Студент и их описание.

Таблица 2.1 – Таблица-пояснение для роли Студент

Название прецедента	**Описание прецедента**
Доступ к учебным материалам	Студент может просматривать и использовать все учебные материалы в личном кабинете.
Участие в онлайн-уроках	Студент может присоединяться к живым онлайн-урокам, задавать вопросы и получать ответы в реальном времени.
Выполнение домашних заданий	Студент получает и выполняет домашние задания, которые проверяются и оцениваются преподавателями.
Просмотр записей уроков	Студент может просматривать записи прошедших уроков для повторения материала или восполнения пропущенного.
Участие в форумах и конкурсах	Студент участвует в форумах и конкурсах, делится опытом и получает обратную связь.
Управление личной информацией и настройками аккаунта	Студент обновляет информацию о себе, отслеживает прогресс и управляет настройками аккаунта в личном кабинете.
Получение сертификатов	По завершении курсов студент получает сертификаты, подтверждающие его достижения и уровень владения языком, которые отправляются на электронную почту.

На диаграммах UML для связывания элементов используются различные соединительные линии, которые называются отношениями, которые используется для достижения определённой цели.

На рисунке 2.2 представлена диаграмма вариантов использования для роли Преподаватель.

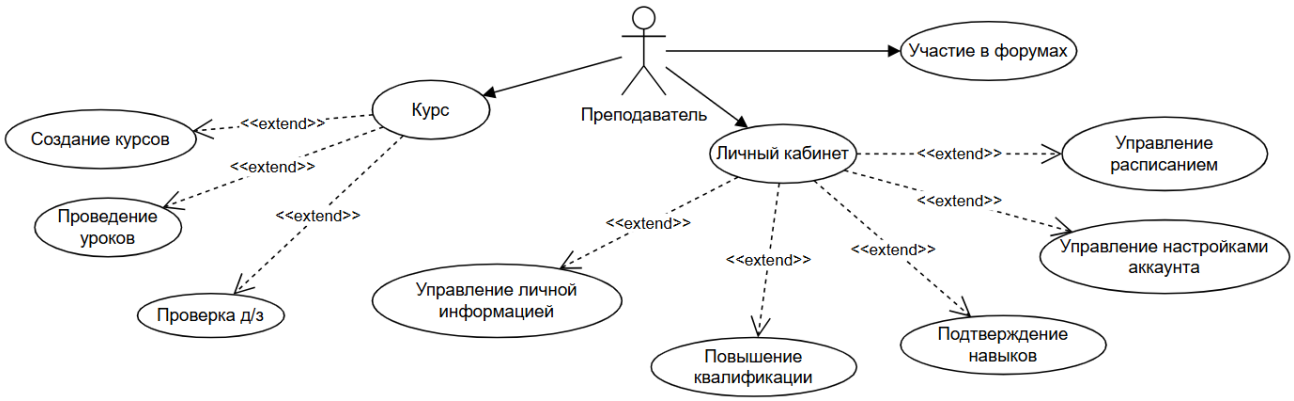

Рисунок 2.2 – Диаграмма вариантов использования для роли Преподаватель

Отношение «ассоциации» — используется для связи между прецедентами и акторами. Оно показывает, что актор взаимодействует с прецедентом. Они изображаются сплошными линиями, соединяющими актера и кейс.

В таблице 2.2 представлены прецеденты роли Преподаватель и их описание.

Таблица 2.2 – Таблица-пояснение для роли Преподаватель

Название прецедента	**Описание прецедента**
Подтверждение навыков	Преподаватель подтверждает свои профессиональные навыки, предоставляя соответствующие документы.
Создание курсов	Преподаватель может создавать новые курсы и добавлять учебные материалы, видеоуроки, тесты и задания.
Проведение онлайн-уроков	Преподаватель проводит онлайн-уроки, взаимодействует в реальном времени, отвечает на вопросы в чате или сообщениях.
Проверка домашних заданий	Преподаватель проверяет выполненные студентами задания, выставляет оценки и предоставляет обратную связь.
Управление расписанием	Преподаватель планирует и управляет своим расписанием уроков, назначает время для консультаций и дополнительных занятий.
Участие в форумах и обсуждениях	Преподаватель участвует в форумах и обсуждениях, отвечает на вопросы студентов и помогает им в освоении материала.
Повышение квалификации	Преподаватель участвует в вебинарах, тренингах и других мероприятиях для повышения квалификации и обмена опытом.
Управление информацией и настройками аккаунта	Преподаватель обновляет информацию о себе и управляет настройками аккаунта в личном кабинете.

В UML-диаграммах вариантов использования <<extend>> обозначает условное расширение сценария.

На рисунке 2.3 представлена диаграмма вариантов использования для роли Менеджер.

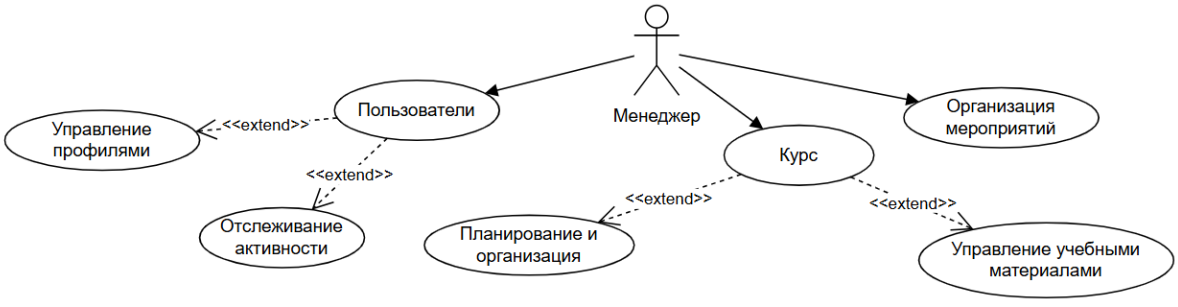

Рисунок 2.3 – Диаграмма вариантов использования для роли Менеджер

Также существует отношение включения, которое обозначается пунктирной стрелкой с <<include>>. Когда мы используем отношение включения, мы

подразумеваем, что составные варианты использования обязательно входят в состав общего варианта использования.

В таблице 2.3 представлены прецеденты роли Менеджер и их описание.

Таблица 2.3 – Таблица-пояснение для роли Менеджер

Название прецедента	**Описание прецедента**
Управление профилями пользователей	Менеджер может добавлять, удалять и редактировать профили пользователей, включая студентов, преподавателей и администраторов.
Отслеживание активности	Менеджер отслеживает активность пользователей на платформе и анализирует статистику посещений, успеваемости студентов и эффективности преподавателей.
Планирование и организация курсов	Менеджер планирует и организовывает новые курсы, назначает преподавателей и следит за их выполнением.
Управление учебными материалами	Менеджер добавляет, редактирует и удаляет учебные материалы, следит за их актуальностью и качеством.
Организация мероприятий	Менеджер организовывает и координирует вебинары, конкурсы и мастер-классы.

Также на диаграммах вариантов использования может использоваться отношение обобщения. Это отношение используется, когда несколько акторов имеют общие характеристики, но один актор является более общим, чем другой. Обозначается стрелкой с пустым треугольником.

На рисунке 2.4 представлена диаграмма для роли Администратор

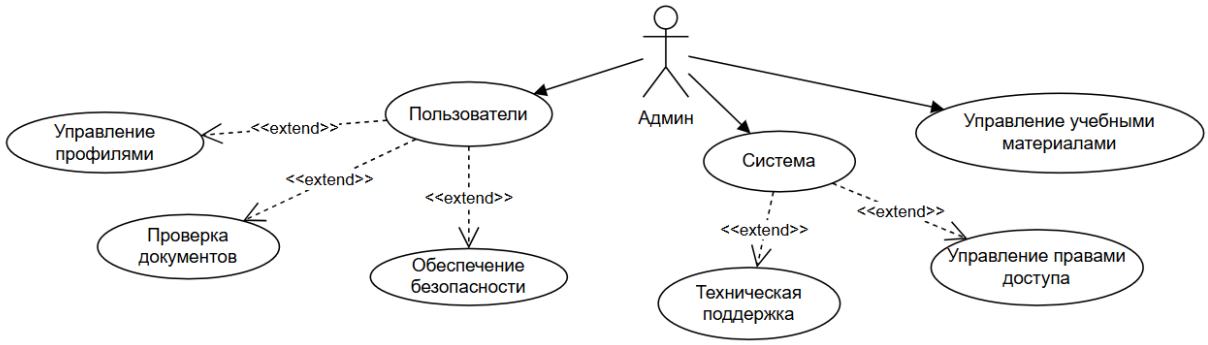

Рисунок 2.4 – Диаграмма вариантов использования для роли Администратор

В таблице 2.4 представлены прецеденты роли Администратор и их описание.

Таблица 2.4 – Таблица-пояснение для роли Administrator

Название прецедента	**Описание прецедента**
Управление профилями	Администратор добавляет, удаляет и редактирует профили всех пользователей.
Проверка документов	Администратор проверяет подлинность документов преподавателей при регистрации.
Обеспечение безопасности	Администратор отвечает за безопасность данных пользователей и предотвращение несанкционированного доступа.
Техническая поддержка	Администратор предоставляет техническую поддержку и решает технические проблемы.
Управление правами доступа	Администратор назначает и изменяет права доступа для различных ролей пользователей.
Управление учебными материалами	Администратор проверяет и утверждает учебные материалы, следит за их качеством.

Диаграммы вариантов использования помогают определить границы системы и функциональность, также позволяют анализировать требования перед разработкой.

2.2 Взаимодействие всех компонентов

В данном разделе на рисунке 2.5 представлена схема взаимодействия всех компонентов базы данных.

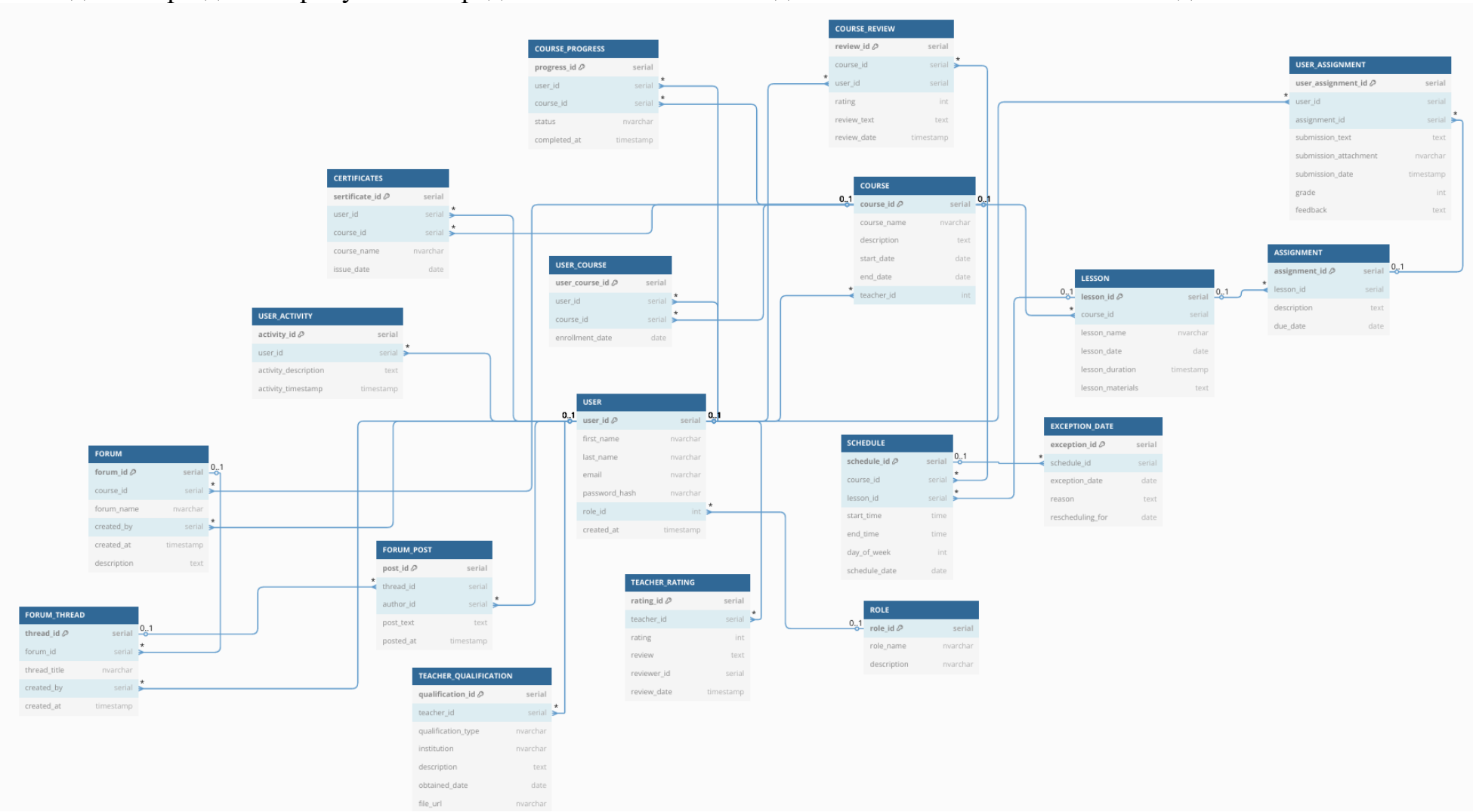

Рисунок 2.5 – Схема базы данных

Диаграмма содержит таблицы, связанные между собой ключами и определяющие структуру базы данных.

2.3 Вывод по разделу

В данном разделе были разработаны диаграммы вариантов использования для всех ролей. Эти диаграммы наглядно показывают функциональные возможности, доступные каждой роли, и их взаимодействие с системой. Также были использованы ключевые отношения UML для точного описания связей. Кроме того, представлена схема базы данных, отражающая структуру и взаимосвязи всех компонентов системы.

3 Разработка объектов базы данных

3.1 Разработка таблиц базы данных

В данном разделе будет рассмотрена разработка таблиц базы данных. Для начала следует создать саму базу данных. В листинге 3.1 представлен SQL-запрос для создания базы данных.

```
use master
create database LINGUABENDER_DB;
```

Листинг 3.1 – SQL-запрос для создания базы данных.

Далее созданы 18 таблиц базы данных LINGUABENDER_DB. Детальное описание всех таблиц базы данных с описанием полей приведено в приложении А. SQL-запрос для создания таблиц представлен в листинге 2.2.

```
use LINGUABENDER_DB
CREATE TABLE [USER] (
  [user_id] int PRIMARY KEY,
  [first_name] nvarchar(50),
  [last_name] nvarchar(50),
  [email] nvarchar(50) UNIQUE,
  [password_hash] nvarchar(50),
  [role_id] int,
  [created_at] timestamp
)
GO
```

Листинг 3.2 – SQL-запрос для создания таблицы USER

SQL-запросы для создания таблиц представлен в приложении Б.

3.2 Разработка схем базы данных

Схема − логическая структура, которая представляет собой набор таблиц, индексов, представлений, функций и других объектов, организованных в единую группу. В рамках курсового проекта была создана одна общая схема SCHEMA1. Код создания схемы приведен в листинге 3.3.

```
use LINGUABENDER_DB
CREATE SCHEMA SCHEMA1;
```

Листинг 3.3 – SQL-запрос для создания схемы

Схемы группируют таблицы, процедуры, функции и другие объекты, упрощая администрирование, разграничение доступа и поддержку безопасности.

3.3 Разработка процедур базы данных

В рамках данного проекта были разработаны хранимые процедуры для управления учебным процессом, учетными записями пользователей и администрированием. Они обеспечивают ключевые функции, необходимые для

корректной работы платформы. Процедура для управления личной информацией и настройками аккаунта представлен в листинге 3.4

```
USE LINGUABENDER_DB
GO
CREATE PROCEDURE SP_UPDATE_USER_INFO
@user_id int, @first_name nvarchar(50), @last_name nvarchar(50),
@email nvarchar(50)
AS
BEGIN
UPDATE [USER]
SET first_name = @first_name,
     last_name = @last_name,
     email = @email
WHERE user_id = @user_id;
end;
```

Листинг 3.4 – SQL-запрос для создания процедуры для управления личной информацией

Все остальные процедуры, созданные в рамках данного курсового проекта представлены в приложении В.

3.4 Разработка функций базы данных

Функции не могут изменять данные, но удобны для расчетов и фильтрации. В данном курсовом проекте были созданы функции для получения аналитических данных и взаимодействия с базой. Пример создания одной из функций представлен в листинге 3.5.

```
CREATE FUNCTION FN_USER_ACTIVITY_BY_DATE (@user_id int,
@activity_date date)
RETURNS TABLE
AS
RETURN (
SELECT * FROM USER_ACTIVITY
WHERE user_id = @user_id AND CAST(activity_datetime AS DATE) =
@activity_date
);
GO
```

Листинг 3.5 – SQL-запрос для создания функции получения активности по дате

Функции позволяют получать статистику по преподавателям, студентам, курсам и активности пользователей. SQL-запросы для создания остальных функций приведены в приложении Г.

3.5 Разработка представлений базы данных

Представления (VIEW) позволяют упрощать доступ к данным, формируя удобные виртуальные таблицы для анализа и отчетов. SQL-запрос для создания

представления «Топ 5 преподавателей по среднему рейтингу» представлен в листинге 3.6.

```
CREATE VIEW V_USER_ACTIVITY_SUM AS
SELECT       u.user_id,       u.first_name,       u.last_name,
COUNT(ua.activity_id) AS activity_count
FROM [USER] u
LEFT JOIN USER_ACTIVITY ua ON u.user_id = ua.user_id
GROUP BY u.user_id, u.first_name, u.last_name
GO
```

Листинг 3.6 – SQL-запрос для создания представления «Топ 5 преподавателей по среднему рейтингу»

В рамках данного курсового проекта созданы представления для просмотра учебных материалов студентами, отслеживания рейтингов преподавателей, а также просмотра расписания преподавателей, который представлены в приложении Д.

3.6 Разработка триггеров базы данных

Триггеры в базе данных позволяют автоматизировать выполнение действий при изменении данных, обеспечивая целостность и безопасность. В рамках проекта были созданы триггеры для управления учебным процессом и проверки данных.

Триггер TR_INIT_COURSE_PROGRESS автоматически добавляет запись о прогрессе студента при его зачислении на курс, устанавливая статус «Не начат».

Триггер TR_VALIDATE_COURSE_DATES предотвращает создание курса, если его дата начала не предшествует дате окончания. Если условие нарушено, триггер генерирует ошибку и не позволяет внести некорректные данные.

Триггер TR_CHECK_EMAIL_UNIQ гарантирует, что при создании учетной записи email пользователя остается уникальным. Если такой email уже существует, триггер прерывает выполнение операции и возвращает ошибку.

3.7 Роли и пользователи

В проекте определены роли и пользователи, обеспечивающие безопасность и разграничение прав доступа. Созданы логины для студентов, преподавателей, менеджеров и администраторов, после чего каждому логину назначен пользователь базы данных. Для удобства управления правами были введены роли, к которым добавлены соответствующие пользователи. Листинг этих SQL-запросов представлен в приложении Е.

3.8 Вывод по разделу

Разработанная база данных включает процедуры, функции, представления, триггеры и систему пользователей. Процедуры автоматизируют операции, функции помогают анализировать данные, представления упрощают доступ к информации, триггеры обеспечивают целостность, а роли и пользователи гарантируют безопасность. Все компоненты делают работу с базой удобной и структурированной.

4 Описание процедур экспорта и импорта данных

Экспорт и импорт данных в формате XML позволяют передавать информацию между базой данных и внешними системами. Этот процесс упрощает интеграцию и хранение данных в структурированном виде.

4.1 Процедура импорта данных из XML-файла

Импорт XML-данных выполняется с помощью OPENXML, позволяющего извлекать данные из загруженного XML-документа. SQL-запрос для импорта данных в таблицу FORUM_POST представлен в листинге 4.1.

```
DECLARE @xmlData XML =
'<ForumPosts>
    <Post>
        <thread_id>10</thread_id>
        <author_id>23</author_id>
        <post_text>Пример поста</post_text>
        <posted_at>2025-04-28T12:00:00</posted_at>
    </Post>
    <Post>
        <thread_id>15</thread_id>
        <author_id>37</author_id>
        <post_text>Еще один пост</post_text>
        <posted_at>2025-04-28T12:30:00</posted_at>
    </Post>
</ForumPosts>';
DECLARE @xmlHandle INT;
EXEC sp_xml_preparedocument @xmlHandle OUTPUT, @xmlData;
INSERT INTO dbo.FORUM_POST (thread_id, author_id, post_text,
posted_at)
SELECT
    thread_id, author_id, post_text, posted_at
FROM OPENXML(@xmlHandle, '/ForumPosts/Post', 2)
WITH (
    thread_id INT 'thread_id',
    author_id INT 'author_id',
    post_text NVARCHAR(MAX) 'post_text',
    posted_at DATETIME 'posted_at'
);
EXEC sp_xml_removedocument @xmlHandle;
```

Листинг 4.1 – SQL-запрос для импорта данных в таблицу FORUM_POST

Вначале XML-файл загружается в переменную, после чего создается его обработчик с sp_xml_preparedocument. Затем данные извлекаются и вставляются в таблицу. Завершающим этапом является удаление обработчика с помощью sp_xml_removedocument, чтобы освободить память.

4.1 Процедура экспорта данных из XML-файла

Экспорт данных организован с помощью FOR XML.

SQL-запрос для экспорта данных из таблицы FORUM_POST представлен в листинге 4.2

```
SELECT post_id, thread_id, author_id, post_text, posted_at
FROM dbo.FORUM_POST
ORDER BY post_id
FOR XML PATH('Post'), ROOT('ForumPosts'), ELEMENTS;
```

Листинг 4.2 – SQL-запрос для экспорта данных из таблицы FORUM_POST

Запрос выбирает данные из таблицы, упаковывая их в XML-структуру с PATH, а корневой элемент обозначается ROOT. В результате формируется XML-документ, который можно использовать для передачи или хранения данных.

4.1 Вывод по разделу

Реализованные процедуры обеспечивают удобный обмен данными между системой и внешними ресурсами. Импорт позволяет загружать записи из XML в базу, а экспорт формирует XML-файл на основе данных из таблицы. Эти механизмы упрощают интеграцию и автоматизируют обработку данных, обеспечивая их структурированное хранение.

5 Тестирование производительности базы данных

5.1 Заполнение таблицы и её оптимизация

Тестирование производительности базы данных является важным этапом проверки ее эффективности. В данном проекте проведено тестирование фильтрации данных в таблице ROLE, чтобы оценить влияние индексации на скорость выполнения запросов.

На первом этапе таблица была заполнена 100000 записями с тестовыми значениями. Это позволило имитировать реальную нагрузку на систему. Была выполнена фильтрация записей, где role_name начинался с «Роль 9999%». На рисунке 5.1 приведен SQL-запрос заполнения таблицы строками и результат выполнения фильтрации таблицы.

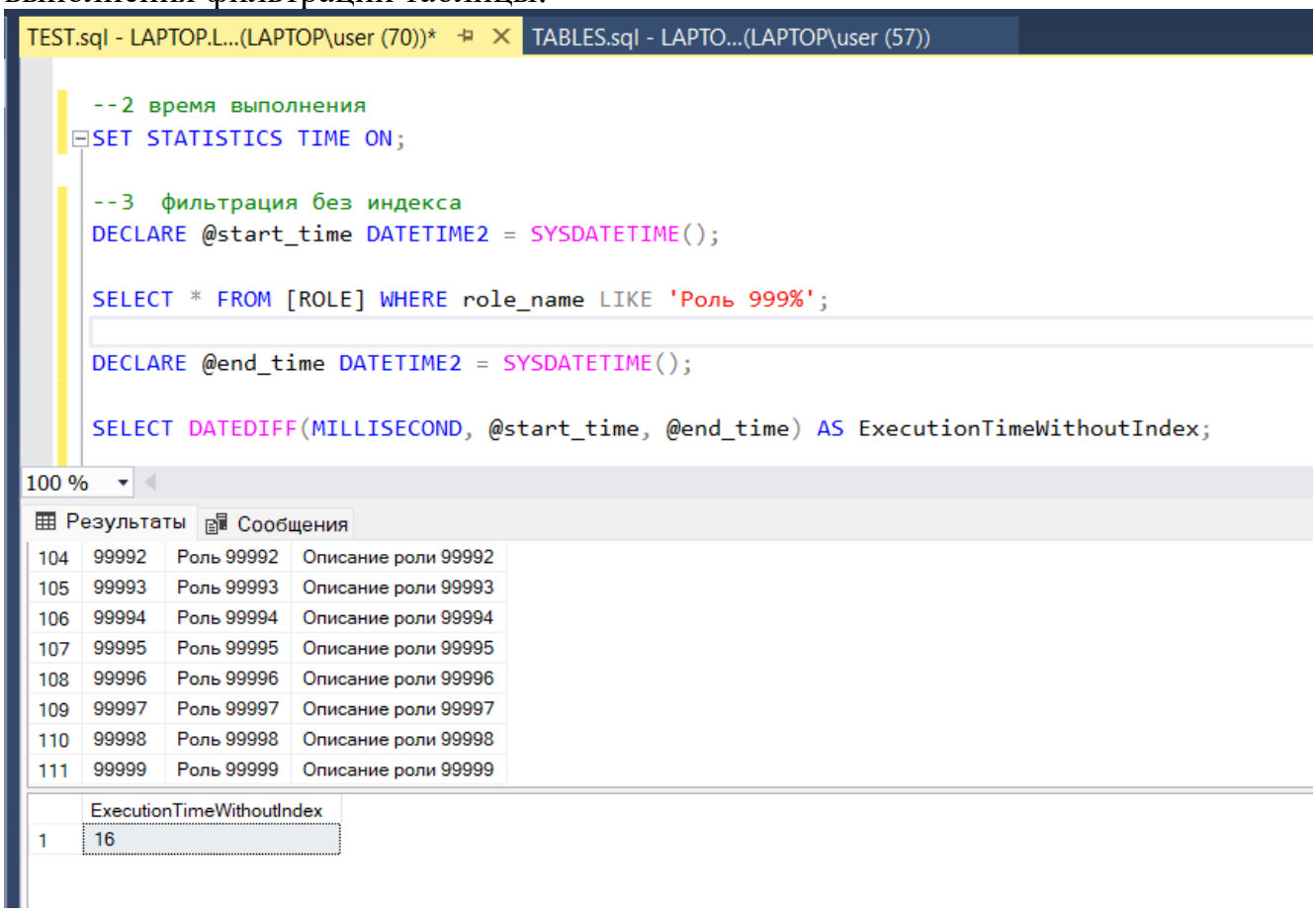

Рисунок 5.1 – Выполнение запроса без индекса

Запрос выполнялся без индекса, что привело к полному сканированию таблицы и затраченному времени 16 миллисекунд.

Чтобы ускорить выполнение запроса, был создан индекс IDX_ROLE_NAME на поле role_name, позволяющий базе данных использовать более быстрый способ поиска данных. Результат запроса уже после создания индекса представлен на рисунке 5.2.

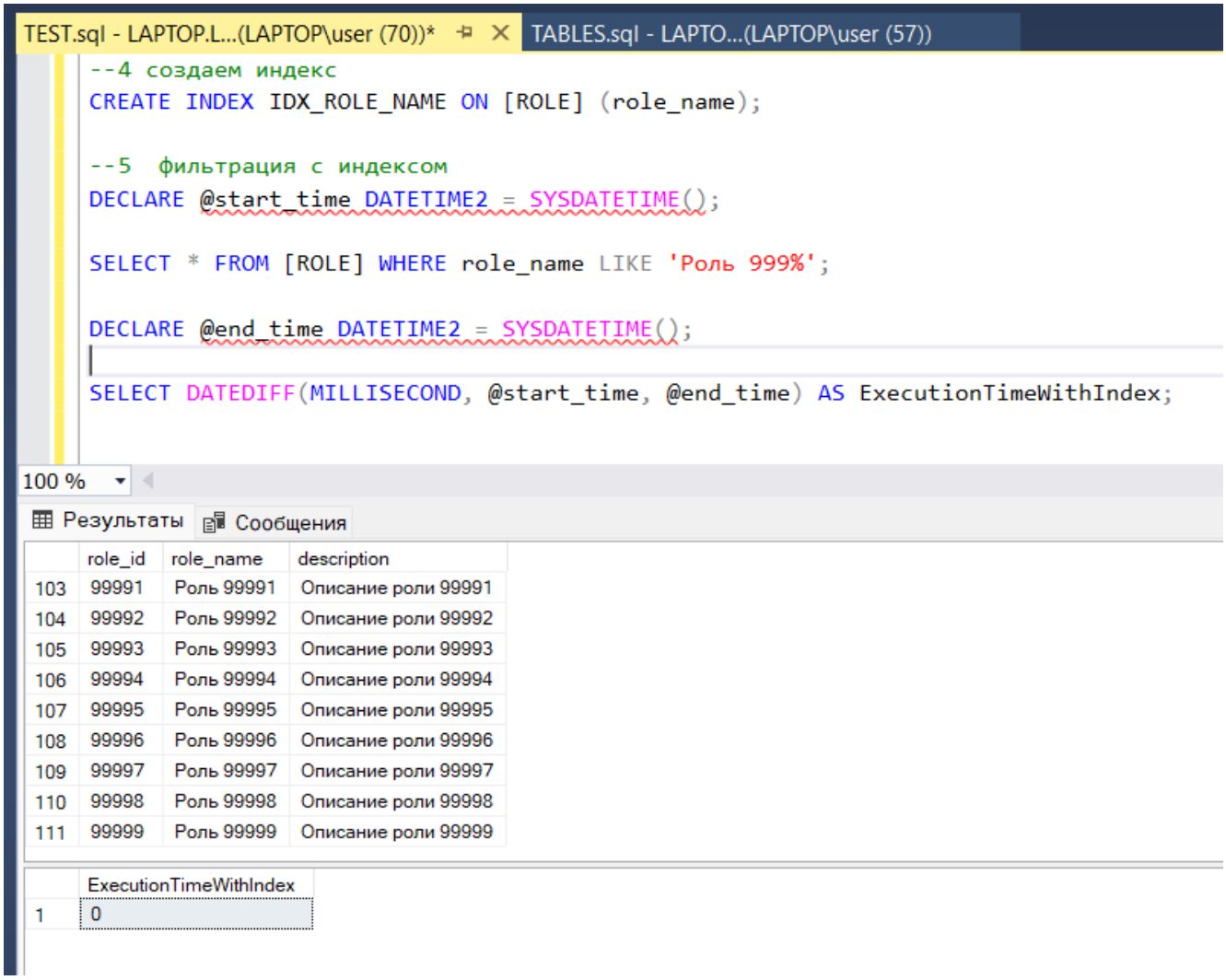

Рисунок 5.2 – Запрос после создания индекса

Как видно на рисунке 5.2, после добавления индекса фильтрация выполняется быстрее, а именно 0 миллисекунд вместо 16-ти, так как SQL Server теперь использует структурированный поиск вместо полного сканирования таблицы.

5.2 Вывод по разделу

Результаты тестирования показывают, что индексирование значительно улучшает производительность поиска данных. Без индекса SQL Server просматривает всю таблицу, тогда как индекс позволяет находить нужные записи быстрее, снижая нагрузку на систему.

Оптимизация запросов через индексацию является важным инструментом для повышения эффективности работы базы данных, особенно при обработке больших объемов информации.

6 Описание технологии и её применение в базе данных

6.1 Технология разработка баз данных для хранения мультимедийной информации

Современные базы данных поддерживают хранение мультимедийной информации, включая изображения, документы и другие файлы. Для этого используются поля типа NVARCHAR(MAX), VARBINARY(MAX), а также специализированные файловые хранилища.

Возможны два подхода хранения:

1. Хранение файлов в базе (VARBINARY(MAX)), когда документ сохраняется прямо в таблице.
2. Ссылочное хранение (NVARCHAR(MAX)), где вместо файлов записываются пути к ним на сервере.

В данном проекте используется подход ссылочного хранения мультимедийных данных. Вместо загрузки файлов непосредственно в базу данных они размещаются на сервере, а в таблицах хранятся пути к этим файлам.

В проекте реализовано хранение файлов в базе данных для подтверждения квалификации преподавателей (дипломов) и загрузки домашних заданий. Для подтверждения квалификации преподавателей ссылки на дипломы и сертификаты записываются в соответствующие поля. Аналогично, файлы с выполненными домашними заданиями студентов сохраняются на внешнем сервере, а их расположение фиксируется в базе.

Использование ссылок вместо бинарного хранения (VARBINARY(MAX)) имеет ряд преимуществ:

1. Снижение нагрузки на базу данных – файлы не занимают место в таблицах, что повышает производительность.
2. Гибкость управления данными – файлы могут храниться на локальном сервере или в облачном хранилище, упрощая доступ и резервное копирование.
3. Упрощение масштабирования – ссылки позволяют легко обновлять файлы без необходимости изменения записей в БД.

6.2 Вывод по разделу

Используемая технология позволяет надежно хранить мультимедийную информацию, обеспечивая удобный доступ к документам преподавателей и заданиям студентов. Это упрощает управление учебным процессом и делает данные доступными внутри системы.

7 Руководство пользователя

В ходе курсового проекта разработано Python-приложение «LINGUABENDER» – система управления обучением для языковой школы с разграничением прав по ролям: студент, преподаватель, менеджер и администратор.

Для начала работы нужно запустить приложение. Главное окно имеет интуитивный интерфейс: слева расположена панель навигации с пятью разделами («Студент», «Преподаватель», «Менеджер», «Администратор», «Функции»), справа – рабочая область с соответствующими инструментами.

Система «LINGUABENDER» предлагает студентам удобный двусторонний интерфейс, представленный на рисунке 7.1.

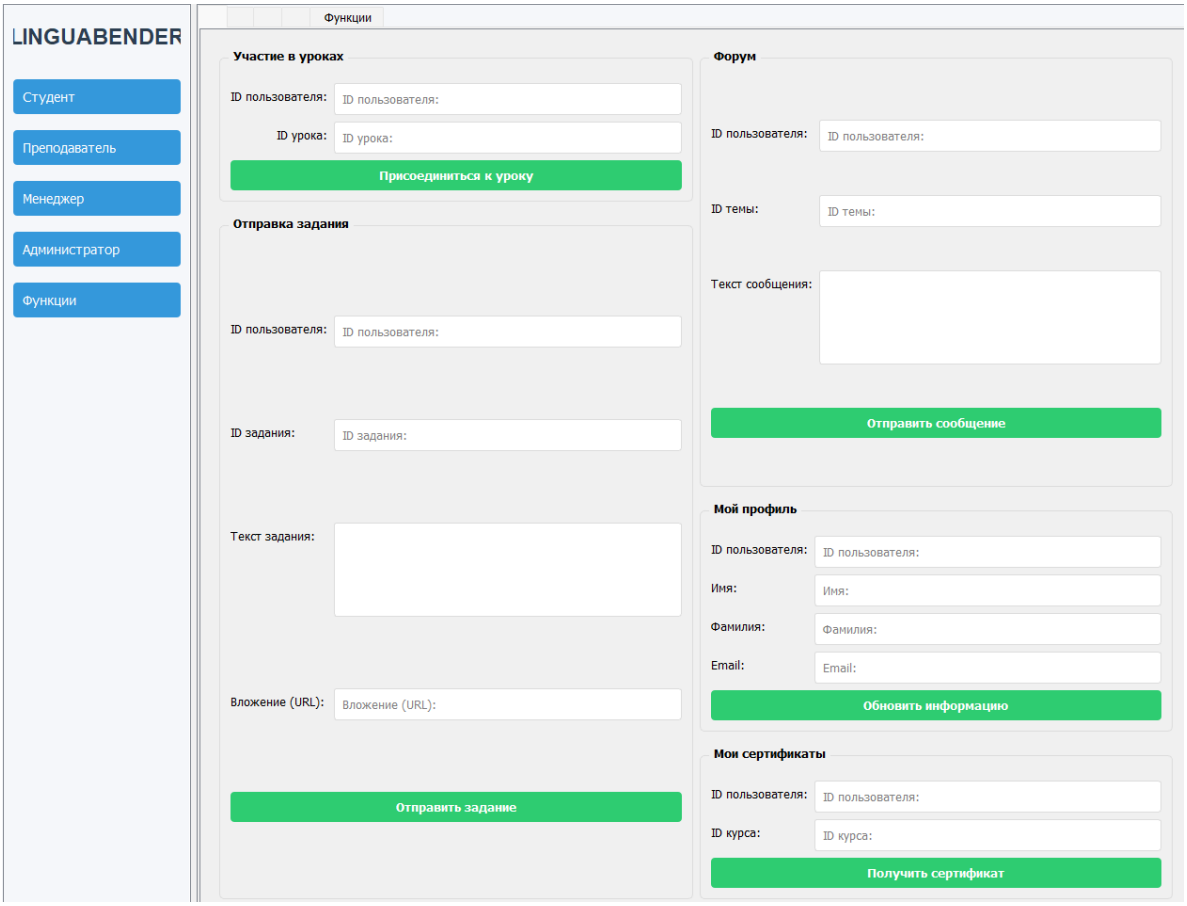

Рисунок 7.1 – Вид главного экрана приложения

В левой части экрана есть инструменты для учебной деятельности: подключение к урокам и отправка выполненных работ. Правая сторона интерфейса представляет собой персональное пространство, где учащиеся могут общаться на форуме, обновлять свои контактные данные в профиле и получать цифровые сертификаты по окончании курсов.

Преподаватели могут разрабатывать новые курсы, детально прописывая программу и временные рамки обучения, а также планировать отдельные занятия с указанием необходимых материалов. Преподавательский модуль, представленный на рисунке 7.2, предлагает комплекс инструментов для организации учебного процесса.

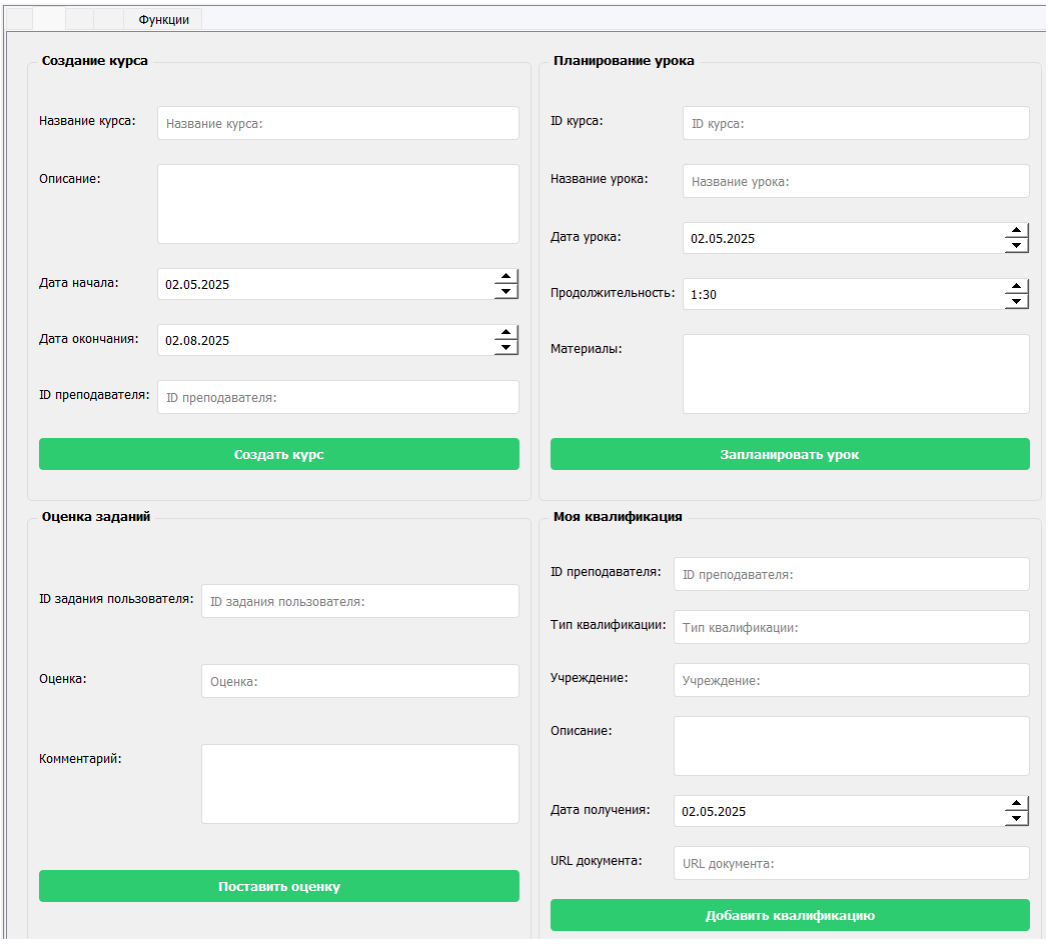

Рисунок 7.2 – Интерфейс преподавателя

Особое внимание уделено системе проверки работ: преподаватели имеют возможность не только выставлять оценки, но и давать комментарии к выполненным заданиям. Дополнительный функционал позволяет подтверждать свою квалификацию путем загрузки соответствующих документов.

Административно-управленческий блок системы включает два взаимосвязанных уровня. Менеджерский инструментарий сосредоточен на поддержании актуальности учебного контента и организации дополнительных образовательных активностей, что позволяет гибко реагировать на изменения в учебном процессе. Административный уровень обеспечивает контроль достоверности преподавательской квалификации и безопасное распределение прав доступа через многоуровневую систему ролей.

7.1 Вывод по разделу

Система «LINGUABENDER» реализует эффективную модель управления учебным процессом через четкое распределение ролей. Каждый участник получает специализированный инструментарий: студенты – для обучения, преподаватели – для организации занятий, менеджеры – для администрирования, администраторы – для контроля системы. Такой подход обеспечивает слаженную работу всех звеньев образовательного процесса, гарантируя прозрачность и удобство взаимодействия.

Заключение

Разработанная база данных для онлайн-платформы по изучению иностранных языков успешно решает задачи, поставленные в начале проекта. В ней реализована система ролей пользователей, что позволяет четко разграничить права доступа и обеспечить безопасность данных. Организован функционал управления курсами, уроками и учебными материалами, упрощающий процесс администрирования образовательного контента.

Для хранения мультимедийной информации используется удобное ссылочное хранение, что минимизирует нагрузку на базу данных и обеспечивает быстрый доступ к файлам. Внедрен механизм аутентификации и авторизации, позволяющий управлять учетными записями пользователей. Реализован импорт и экспорт данных в формате XML, что упрощает взаимодействие с внешними системами и автоматизирует обработку данных.

База данных прошла тестирование производительности и оптимизацию, что гарантирует ее стабильную работу с большими объемами информации. В целом проект обеспечивает эффективное хранение, обработку и управление учебными процессами, отвечая требованиям масштабируемости, надежности и безопасности.

Список использованных источников

1. Аналог «Coursera» [Электронный ресурс]. – Режим доступа: https://www.coursera.org/. – Дата доступа: 06.03.2025.
2. Аналог «Skillspace» [Электронный ресурс]. – Режим доступа: https://skillspace.ru/ . – Дата доступа: 06.03.2025.
3. Аналог «GeekBrains» [Электронный ресурс]. – Режим доступа: https://geekbrains.by/.– Дата доступа: 06.03.2025.

Приложение А

Таблица А.1 – USER

Поле таблицы	Тип поля	Ограничения	Назначение поля
user_id	Int	IDENTITY(1,1) PRIMARY KEY	Уникальный идентификатор пользователя
first_name	Nvarchar(50)		Имя пользователя
last_name	Nvarchar(50)		Фамилия пользователя
Email	Nvarchar(50)	UNIQUE	Электронная почта пользователя
password_hash	Nvarchar(50)		Хеш пароля пользователя для личного кабинета
role_id	Int		Идентификатор роли пользователя
created_at	datetime		Дата регистрации

Таблица А.2 – USER_ACTIVITY

Поле таблицы	Тип поля	Ограничения	Назначение поля
activity_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор активности пользователя
user_id	Int		Идентификатор пользователя, активность которого фиксируется
activity_description	Nvarchar(50)		Описание активности
activity_timestamp	datetime		Дата и время фиксации активности

Таблица А.3 – COURSE

Поле таблицы	Тип поля	Ограничения	Назначение поля
course_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор курса
course_name	Nvarchar(50)		Название курса
description	Nvarchar(50)		Описание курса
start_date	Date		Дата начала курса
end_date	Time		Дата окончания курса
teacher_id	Nvarchar(50)		Идентификатор преподавателя курса

Таблица А.4 – USER_COURSE

Поле таблицы	Тип поля	Ограничения	Назначение поля
user_course_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор таблицы user_course
user_id	Int		Идентификатор пользователя
course_id	Int		Идентификатор курса
enrollment_date	date		Дата начала курса

Таблица А.5 – LESSON

Поле таблицы	Тип поля	Ограничения	Назначение поля
lesson_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор урока
course_id	Int		Идентификатор курса
lesson_name	Nvarchar(50)		Название урока
lesson_date	Date		Дата урока
lesson_duration	Time		Продолжительность урока по времени
lesson_materials	Nvarchar(50)		Материалы для урока

Таблица А.6 – SCHEDULE

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
schedule_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор расписания
course_id	Int		Идентификатор курса
lesson_id	Int		Идентификатор урока
start_time	Time		Начало урока
end_time	Time		Окончание урока
day_of_week	Int		День недели
schedule_date_	date		Дата

Таблица А.7 – EXCEPTION_DATE

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
exception_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор отмены занятия
schedule_id	Int		Идентификатор расписания
exception_date	Date		Дата отмены занятия
reason	Nvarchar(100)		Причина отмены занятия
rescheduling_for	date		Занятие перенесено на эту дату

Таблица А.8 – ASSIGNMENT

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
assignment_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор д/з
lesson_id	Int		Идентификатор урока
Description	Nvarchar(200)		Описание д/з
due_date	date		Дедлайн сдачи д/з

Таблица А.9 – USER_ASSIGNMENT

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
user_assignment_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор выполненного д/з
user_id	Int		Идентификатор пользователя (студента), который выполнил д/з
assignment_id	Int		Идентификатор заданного д/з
submission_text	Nvarchar(200)		Текст выполненного д/з
attachment	Nvarchar(max)		Файл с прикрепленным выполненным д/з
submission_date	Datetime		Дата отправки д/з
grade	Int		Оценка
feedback	Nvarchar(200)		Фидбек от преподавателя

Таблица А.10 – COURSE_PROGRESS

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
progress_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор прогресса
user_id	Int		Идентификатор пользователя, проходящего курс (студента)
course_id	Int		Идентификатор проходимого курса
status	Nvarchar(50)		Статус прохождения
completed_at	datetime		Дата завершения курса

Таблица А.11 – CERTIFICATES

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
sertificate_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор сертификата
user_id	Int		Идентификатор пользователя (студента), получившего сертификат
course_id	Int		Идентификатор пройденного курса
course_name	Nvarchar(50)		Название курса
issue_date	date		Дата завершения

Таблица А.12 – COURSE_REVIEW

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Review_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор отзыва
Course_id	Int		Идентификатор курса, на который оставили отзыв
User_id	Int		Идентификатор пользователя, который оставил отзыв
Rating	Int		Рейтинг отзыва от 0 до 5
Review_text	Nvarchar(200)		Текст отзыва
Review_date	datetime		Дата отзыва

Таблица А.13 – FORUM

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Forum_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор форума
Course_id	Int		Идентификатор курса, с которым связан форум
Forum_name	Nvarchar(50)		Название форума
Created_by	Int		Пользователь, создавший форум
Created_at	Datetime		Дата и время создания форума
description	Nvarchar(200)		Описание форума

Таблица А.14 – FORUM_THREAD

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Thread_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор темы форума
Forum_id	Int		Идентификатор форума по этой теме
Thread_title	Nvarchar(50)		Заголовок темы
Created_by	Int		Пользователь, создавший тему
Created_at	datetime		Дата и время создания темы на форуме

Таблица А.15 – FORUM_POST

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Post_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор поста
Thread_id	Int		Идентификатор темы, на которой опубликован пост
Author_id	Nvarchar(50)		Идентификатор автора поста
Post_text	Int		Текст поста
Posted_at	datetime		Дата и время публикации поста

Таблица А.16 – TEACHER_QUALIFICATION

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Qualification_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор квалификации
Teacher_id	Int		Идентификатор пользователя
Qualification_type	Nvarchar(50)		Тип квалификации
institution	Nvarchar(50)		Учреждение образования, выдавшего документ об образовании
Description	Nvarchar(200)		Описание документа
Obtained_date	Date		Дата получения документа о квалификации
File_url	Nvarchar(500)		url прикрепленного файла с подтверждением квалификации

Таблица А.17 – TEACHER_RATING

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Reting_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор рейтинга преподавателя
Teacher_id	Int		Идентификатор преподавателя
Rating	Int		Рейтинг преподавателя
Review	Nvarchar(200)		Отзыв о преподавателе
Reviewer_id	Int		Идентификатор пользователя, оставившего отзыв
Review_date	datetime		Дата и время, когда отзыв был оставлен

Таблица А.18 – ROLE

Поле таблицы	**Тип поля**	**Ограничения**	**Назначение поля**
Role_id	Int	IDENTITY(1,1) PRIMARY KEY	Идентификатор рейтинга преподавателя
role_name	Nvarchar(50)		Идентификатор преподавателя
description	Nvarchar(200)		Рейтинг преподавателя

Приложение Б

```
use LINGUABENDER_DB
CREATE TABLE [USER] (
  [user_id] int IDENTITY(1,1) PRIMARY KEY,
  [first_name] nvarchar(50),
  [last_name] nvarchar(50),
  [email] nvarchar(50) UNIQUE,
  [password_hash] nvarchar(50),
  [role_id] int,
  [created_at] datetime
)
GO
CREATE TABLE [ROLE] (
  [role_id] int IDENTITY(1,1) PRIMARY KEY,
  [role_name] nvarchar(50),
  [description] nvarchar(200)
)
GO
CREATE TABLE [COURSE] (
  [course_id] int IDENTITY(1,1) PRIMARY KEY,
  [course_name] nvarchar(50),
  [description] nvarchar(200),
  [start_date] date,
  [end_date] date,
  [teacher_id] int
)
GO
CREATE TABLE [LESSON] (
  [lesson_id] int IDENTITY(1,1) PRIMARY KEY,
  [course_id] int,
  [lesson_name] nvarchar(50),
  [lesson_date] date,
  [lesson_duration] time,
  [lesson_materials] nvarchar(200)
)
GO
CREATE TABLE [USER_COURSE] (
  [user_course_id] int IDENTITY(1,1) PRIMARY KEY,
  [user_id] int,
  [course_id] int,
  [enrollment_date] date
)
GO
CREATE TABLE [CERTIFICATES] (
  [sertificate_id] int IDENTITY(1,1) PRIMARY KEY,
  [user_id] int,
  [course_id] int,
  [course_name] nvarchar(50),
  [issue_date] date
)
```

Листинг Б – Листинг SQL-запросов для создания таблиц

Приложение В

```
USE LINGUABENDER_DB
GO
CREATE PROCEDURE SP_JOIN_LESSON
@user_id INT, @lesson_id INT
AS
BEGIN
IF EXISTS(
SELECT 1 FROM USER_COURSE uc
JOIN LESSON l ON uc.course_id = l.course_id
WHERE uc.user_id = @user_id AND l.lesson_id = @lesson_id
)
BEGIN
INSERT INTO USER_ACTIVITY (user_id, activity_description,
activity_datetime)
VALUES(@user_id, 'Присоединился к уроку', CURRENT_TIMESTAMP);
END
ELSE
BEGIN
RAISERROR('Пользователь не зачислен на курс этого урока.', 16, 1);
END
END;

--Выполнение д/з
USE LINGUABENDER_DB
GO
CREATE PROCEDURE SP_SUBMIT_ASSIGNMENT
@user_id int, @assignment_id int, @submission_text nvarchar(200),
@submission_attachment nvarchar(200)
AS
BEGIN
INSERT INTO USER_ASSIGNMENT(user_id, assignment_id,
submission_text, submission_attachment, submission_date)
VALUES (@user_id, @assignment_id, @submission_text,
@submission_attachment, CURRENT_TIMESTAMP);
END;

--Участие в форумах
USE LINGUABENDER_DB
GO
CREATE PROCEDURE SP_POST_FORUM_MESSAGE
@user_id int, @thread_id int, @post_text nvarchar(200)
AS
BEGIN
INSERT INTO FORUM_POST (thread_id, author_id, post_text, posted_at)
VALUES (@thread_id, @user_id, @post_text, CURRENT_TIMESTAMP);
END;
```

Листинг В – Листинг SQL-запросов для создания процедур базы данных

Приложение Г

```
--Получение всех форумов, где пользователь участвовал
CREATE FUNCTION FN_USER_FORUMS (@user_id int)
RETURNS TABLE
AS
RETURN (
SELECT DISTINCT f.forum_id, f.forum_name, f.description
FROM FORUM f
JOIN FORUM_THREAD ft ON f.forum_id = ft.forum_id
JOIN FORUM_POST fp ON ft.thread_id = fp.thread_id
WHERE fp.author_id = @user_id
);
GO

--Проверка, завершил ли пользователь курс
CREATE FUNCTION FN_COURSE_COMPLETED (@user_id int, @course_id int)
RETURNS BIT
AS
BEGIN
DECLARE @status nvarchar(50);
SELECT @status = status FROM COURSE_PROGRESS
WHERE user_id = @user_id AND course_id = @course_id;

RETURN CASE WHEN @status = 'Завершен' THEN 1 ELSE 0 END;
END;
GO
--Получение всех домашних заданий студента по курсу
CREATE FUNCTION FN_STUD_ASSIGNMNET_BY_COURSE (@user_id int,
@course_id int)
RETURNS TABLE
AS
RETURN (
SELECT
a.assignment_id,
a.description,
a.due_date,
ua. submission_date,
ua.grade
FROM USER_ASSIGNMENT ua
JOIN ASSIGNMENT a ON ua.assignment_id = a.assignment_id
JOIN LESSON l ON a.lesson_id = l.lesson_id
WHERE ua.user_id = @user_id AND l.course_id = @course_id
);
GO
```

Листинг Г – Листинг SQL-запросов для создания функций базы данных

Приложение Д

```
--Просмотр учебных материалов
CREATE VIEW V_STUDENT_MATERIALS AS
SELECT u.user_id, c.course_name, l.lesson_name, l.lesson_materials
FROM [USER] u
JOIN USER_COURSE uc ON u.user_id = uc.user_id
JOIN COURSE c ON uc.course_id = c.course_id
JOIN LESSON l ON c.course_id = l.course_id
WHERE u.role_id = (SELECT role_id FROM [ROLE] WHERE role_name =
'Студент');
--Просмотр записей уроков
CREATE VIEW V_LESSON_RECORDS AS
SELECT u.user_id, l.lesson_name, l.lesson_materials
FROM [USER] u
JOIN USER_COURSE uc ON u.user_id = uc.user_id
JOIN COURSE c ON uc.course_id = c.course_id
JOIN LESSON l ON c.course_id = l.course_id
WHERE u.role_id = (SELECT role_id FROM [ROLE] WHERE role_name =
'Студент');
--Отслеживание активности--
CREATE VIEW V_USER_ACTIVITY_SUM AS
SELECT u.user_id, u.first_name, u.last_name, COUNT(ua.activity_id)
AS activity_count
FROM [USER] u
LEFT JOIN USER_ACTIVITY ua ON u.user_id = ua.user_id
GROUP BY u.user_id, u.first_name, u.last_name
GO
--Топ 5 преподавателей по среднему рейтингу--
CREATE VIEW V_TOP_TEACHERS_BY_RATING AS
SELECT TOP 5
u.user_id, u.first_name, u.last_name,
AVG(tr.rating) AS average_rating
from TEACHER_RATING tr
JOIN [USER] u ON tr.teacher_id = u.user_id
GROUP BY u.user_id, u.first_name, u.last_name
ORDER BY average_rating desc;
Go
--детализированный прогресс студентов по курсам--
CREATE VIEW V_STUD_PROGRESS_DETAILS AS
SELECT
    u.user_id,
    u.first_name,
    u.last_name,
    c.course_name,
    cp.status,
    cp.completed_at
FROM COURSE_PROGRESS cp
JOIN [USER] u ON cp.user_id = u.user_id
JOIN COURSE c ON cp.course_id = c.course_id;
```

Листинг Д – Листинг SQL-запросов для создания представлений базы данных

Приложение Е

```
CREATE LOGIN student_login WITH PASSWORD = 'student';
CREATE LOGIN teacher_login WITH PASSWORD = 'teacher';
CREATE LOGIN manager_login WITH PASSWORD = 'manager';
CREATE LOGIN admin_login WITH PASSWORD = 'admin';

--создание пользователей
USE LINGUABENDER_DB;

CREATE USER student_user FOR LOGIN student_login;
CREATE USER teacher_user FOR LOGIN teacher_login;
CREATE USER manager_user FOR LOGIN manager_login;
CREATE USER admin_user FOR LOGIN admin_login;

--создание ролей
CREATE ROLE student_role;
CREATE ROLE teacher_role;
CREATE ROLE manager_role;
CREATE ROLE admin_role;

--назначение пользователей на роли
ALTER ROLE student_role ADD MEMBER student_user;
ALTER ROLE teacher_role ADD MEMBER teacher_user;
ALTER ROLE manager_role ADD MEMBER manager_user;
ALTER ROLE admin_role ADD MEMBER admin_user;
```

Листинг Е – Листинг SQL-запросов для создания пользователей и ролей базы данных