Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА ИУ7 «ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

КУРСОВАЯ РАБОТА

НА ТЕМУ:

Разработка базы данных для онлайн-платформы потокового видео с подпиской

Студент	**ИУ7И-64Б**		
	(группа)	(подпись, дата)	(И.О. Фамилия)
Руководитель курсового проекта			**Гаврилова Ю.М.**
		(подпись, дата)	(И.О. Фамилия)
Консультант			
		(подпись, дата)	(И.О. Фамилия)

2025 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой ИУ7
(индекс)
И. В. Рудаков
(И.О. Фамилия)
(подпись)
28.02.2025 года
(дата)

З А Д А Н И Е
на выполнение курсовой работы

по дисциплине **«Базы данных»**

Студент группы **ИУ7И-64Б Шавиш Тарек**
(Фамилия, имя, отчество)

Тема курсовой работы **Разработка базы данных для онлайн-платформы потокового видео с подпиской**

Направленность КР (учебная, исследовательская, практическая, др.) **учебная**

Источник тематики (кафедра, предприятие, НИР) **кафедра**

Задание

Проанализировать предметную область сервисов потокового видео, а также сформулировать требования и ограничения к разрабатываемой базе данных. Рассмотреть описание пользователей системы и их уровней доступа к базе данных. Спроектировать сущности базы данных и ограничения целостности для сервиса потокового видео, включая пользователей, подписки, фильмы, списки просмотра и отзывы. Разработать ролевую модель на уровне базы данных, определяющую права доступа для разных типов пользователей. Выбрать средства реализации базы данных, включая выбор СУБД для хранения данных сервиса потокового видео. Разработать сущности базы данных с реализацией ограничений целостности. Исследовать влияние структуры базы данных и индексов на производительность запросов в сервисе потокового видео.

Оформление курсовой работы:

Расчетно-пояснительная записка на 18-40 листах формата А4. Презентация к курсовой работе на 6-15 слайдах.

Дата выдачи задания «28» февраля 2025 г.

Руководитель курсовой работы _____ **Гаврилова Ю.М.**
(подпись, дата) (И.О. Фамилия)

Студент _____ _____
(подпись, дата) (И.О. Фамилия)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Видеохостинг - одна из самых востребованных областей в веб-приложениях, предоставляющая пользователям возможность загружать, просматривать и взаимодействовать с видеоконтентом. С появлением технологий потокового видео возникла потребность в платформе, обеспечивающей гибкость в простом управлении контентом, без сложностей, связанных с другим программным обеспечением.

Целью данной курсовой работы является проектирование и разработка базы данных для онлайн-платформы потокового видео с подпиской Для достижения поставленной цели необходимо решить следующие задачи:

— определить необходимый функционал приложения и его архитектуру;

— выделить роли пользователей и формализовать данные;

— проанализировать системы управления базами данных и выбрать подходящую СУБД;

— спроектировать структуру базы данных, определить её сущности и связи;

— реализовать механизмы взаимодействия с базой данных, включая триггеры и индексы для повышения производительности.

1 Аналитический раздел

Аналитический раздел курсовой работы является важным этапом проектирования приложения, поскольку он определяет основные требования к системе и выбирает наиболее подходящую модель базы данных. В данном разделе будут рассмотрены различные типы баз данных, их преимущества и недостатки, а также проведен анализ существующих систем управления базами данных. На основе полученных результатов будет выбрана оптимальная модель базы данных для реализации приложения. Кроме того, в данном разделе будут определены пользователи системы, формализованы данные и представлены сущности, что позволит более точно определить требования к приложению и разработать его структуру.

1.1 Требования к приложению

Приложение должно поддерживать следующий функционал:

- Возможность регистрации и входа в систему.
- Управление подписками (FREE, PLUS, PREMIUM).
- Просмотр каталога видео с возможностью поиска и фильтрации.
- Добавление видео в список просмотра (Watchlist) для пользователей с подпиской PLUS или PREMIUM.
- Возможность оставлять отзывы и рейтинги для пользователей с подпиской PREMIUM.

1.2 Пользователи системы

В системе предусмотрены следующие роли:

- **FREE**: Может просматривать видео.
- **PLUS**: Может использовать Watchlist.
- **PREMIUM**: Может оставлять отзывы и рейтинги.

На рисунке 1 представлена диаграмма вариантов использования приложения.

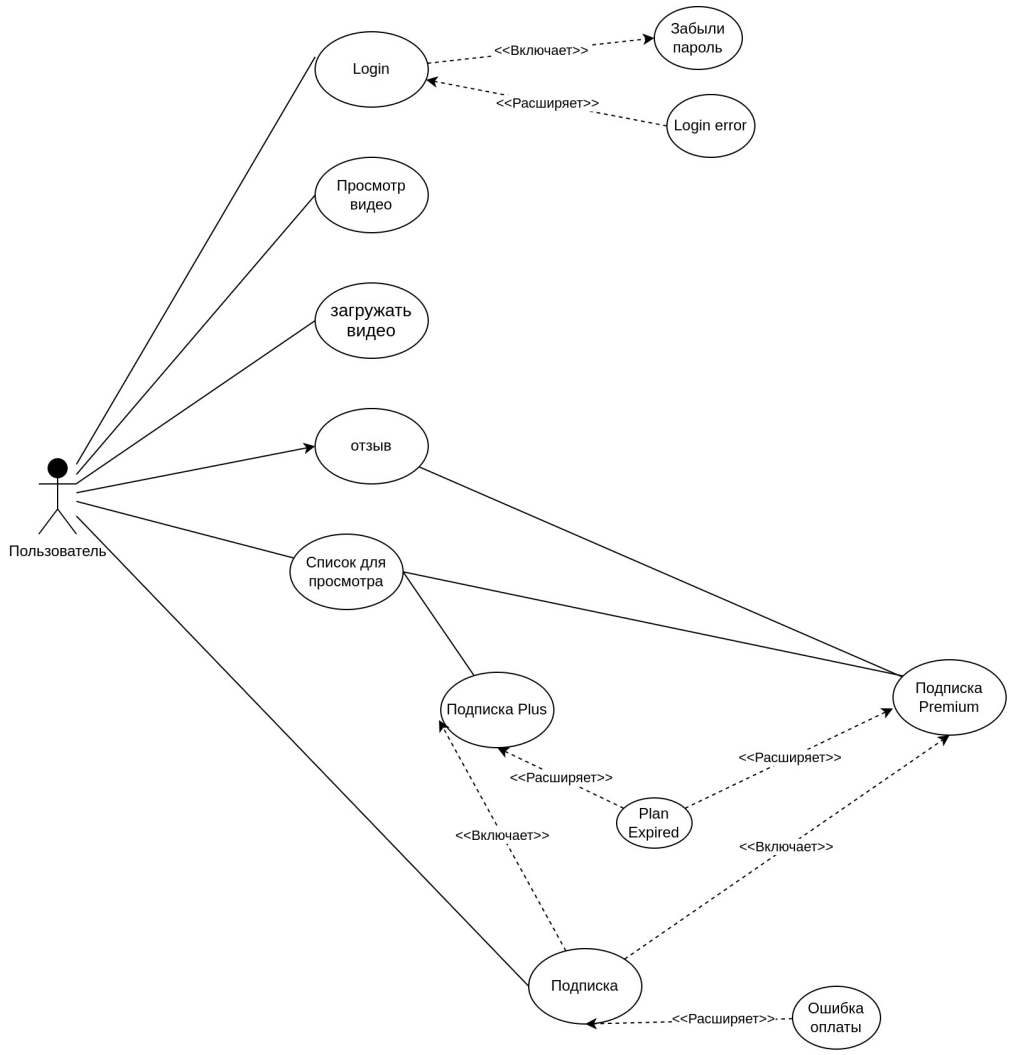

Рис. 1: Диаграмма вариантов использования

1.3 Формализация данных

База данных должна хранить информацию о следующих сущностях:

- **Пользователь**: ID, имя, email, пароль, роль.
- **Подписка**: ID, тип (FREE/PLUS/PREMIUM), цена, дата начала, дата окончания.
- **Видео**: ID, заголовок, описание, путь к файлу, дата создания, ID загрузившего пользователя.
- **Список просмотра (Watchlist)**: ID, пользователь, видео.
- **Отзывы (Rate)**: ID, комментарий, оценка, дата создания, пользователь, видео.
- **Оплата (Payment)**: ID, сумма платежа, дата платежа, пользователь.

На рисунке 2 представлена диаграмма сущностей и связей (ERD).

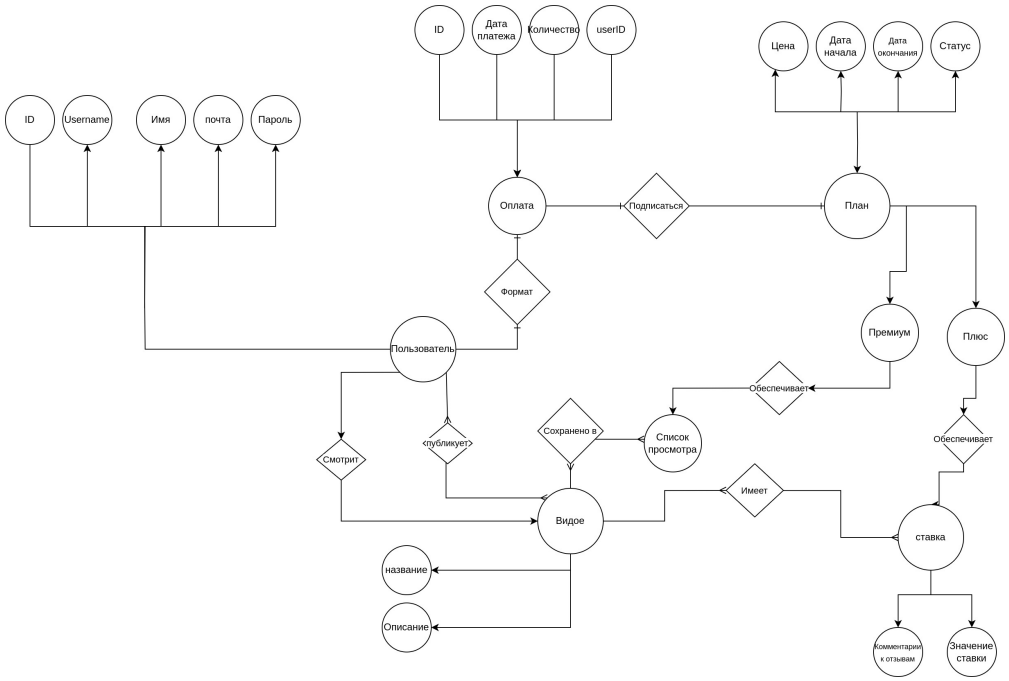

Рис. 2: Диаграмма сущностей и связей в нотации Чена

1.4 Анализ моделей баз данных

Одним из ключевых этапов разработки приложения является выбор модели базы данных, которая будет использоваться для хранения и управления данными. В данном проекте данные включают текстовую информацию о пользователях, подписках, видео, отзывах и списках просмотра. Для хранения видеоконтента используется файловая система, а метаданные (например, пути к файлам, названия, описания) сохраняются в базе данных PostgreSQL. Далее будет проведен анализ различных моделей баз данных, их особенностей и преимуществ, чтобы обосновать выбор реляционной модели для данного проекта.

1.4.1 Реляционная модель

В данном проекте используется реляционная модель базы данных, реализованная с помощью PostgreSQL. В этой модели данные организованы в виде таблиц, состоящих из строк и столбцов. Каждая таблица представляет собой отдельную сущность, такую как пользователи, видео или подписки, а строки содержат информацию о конкретных экземплярах этих сущностей.

Ключевым принципом реляционной модели является использование первичных и внешних ключей для связи между таблицами. Например, информация о пользователях связана с их подписками через уникальный идентификатор (`user_id`), а видео связаны с отзывами через `video_id`. Это позволяет эффективно организовать структуру данных и избежать дублирования информации.

Для улучшения производительности в проекте были созданы индексы для часто используемых столбцов, таких как `user_id`, `video_id` и `created_at`.

1.4.2 Модель ключ–значение

Модель ключ–значение — это способ организации данных, в котором информация хранится в виде пар "ключ–значение". Ключ — это уникальный идентификатор, который используется для доступа к соответствующему значению. Значение может быть любого типа данных, включая строки, числа или объекты.

Преимущества модели ключ–значение:

- Высокая скорость доступа к данным благодаря простой структуре.
- Подходит для хранения небольших фрагментов информации.

Недостатки модели ключ–значение:

- Не подходит для сложных запросов и аналитики данных.
- Может возникнуть проблема с производительностью при работе с большими объемами данных.

В данном проекте модель ключ–значение не используется, так как она не подходит для хранения структурированных данных, таких как информация о пользователях и видео.

1.4.3 Объектно–ориентированная модель

Объектно–ориентированная модель базы данных — это способ организации данных, в котором информация представляется в виде объектов, имеющих свойства и методы. Каждый объект представляет отдельную сущность, например, пользователя или видео, а свойства объекта содержат информацию о нем.

Преимущества объектно–ориентированной модели:

- Более естественное описание данных: Объекты более интуитивно понятны для разработчиков, работающих с объектно–ориентированными языками программирования.
- Гибкость: Позволяет управлять данными на более высоком уровне абстракции.

Недостатки объектно–ориентированной модели:

- Сложность интеграции: Может быть сложно интегрировать с другими системами, использующими реляционные модели.
- Производительность: При работе с большими объемами данных может потребоваться дополнительная оптимизация.

В данном проекте объектно–ориентированная модель не используется, так как реляционная модель PostgreSQL лучше подходит для структурированных данных и обеспечивает высокую производительность.

1.5 Обзор существующих баз данных

Одним из важнейших этапов разработки приложения является выбор системы управления базами данных (СУБД), которая будет использоваться для хранения и управления данными. В данном разделе будут рассмотрены основные варианты СУБД, их преимущества и недостатки, а также обоснован выбор PostgreSQL для реализации проекта.

1.5.1 Документо-ориентированные базы данных

Документо-ориентированные базы данных относятся к категории нереляционных (NoSQL) систем хранения данных. Они используют структуру документов, обычно в формате JSON или BSON, что позволяет хранить гибко структурированную информацию без необходимости строгой схемы. Такой подход особенно удобен при работе с разнородными или изменяющимися данными. Примерами таких СУБД являются MongoDB и CouchDB. Эти базы данных хорошо масштабируются и позволяют быстро выполнять операции чтения и записи, но менее подходят для сложных связей между сущностями.

1.5.2 Графовые базы данных

Графовые базы данных представляют информацию в виде узлов (объектов) и рёбер (связей). Они особенно эффективны в приложениях, где основное внимание уделяется связям между элементами, например, в социальных сетях, рекомендательных системах или маршрутизации. Графовые СУБД, такие как Neo4j, позволяют выполнять быстрые запросы на поиск путей и связей между объектами. Однако они менее универсальны при работе с данными, которые удобно хранить в табличной структуре.

1.5.3 Реляционные базы данных

Реляционные базы данных основаны на модели, в которой данные представлены в виде таблиц с чётко определёнными схемами. Каждая таблица содержит строки (записи) и столбцы (поля), а связи между таблицами устанавливаются через первичные и внешние ключи. SQL используется для управления и обработки данных.

Преимущества выбора реляционной базы данных для проекта:

— Строгая структура данных обеспечивает целостность и согласованность информации.

— Удобное определение связей между сущностями, таких как пользователи, видео, подписки и комментарии.

— Высокая надёжность и зрелость технологий — реляционные СУБД широко используются и хорошо поддерживаются.

— Поддержка сложных запросов, агрегаций и транзакций, что критично для систем, работающих с большим количеством данных.

1.6 Вывод

В рамках аналитического раздела курсовой работы был проведен детальный анализ требований к приложению, определены его основные функциональные возможности и выделены роли пользователей. Рассмотрены различные модели баз данных, включая реляционную, ключ-значение и объектно-ориентированную, а также обоснован выбор реляционной модели на основе PostgreSQL для реализации проекта.

Кроме того, был проведен обзор существующих систем управления базами данных, и на основании анализа их возможностей PostgreSQL был выбран в качестве основной СУБД, учитывая его надежность, масштабируемость и поддержку сложных SQL-запросов. Важной частью исследования стало изучение методов повышения производительности базы данных, таких как использование индексов, триггеров и представлений.

Таким образом, проведенный анализ и выбор архитектурных решений сформировали прочную основу для последующего этапа проектирования и разработки приложения.