Частное учреждение профессионального образования
«Высшая школа предпринимательства»
(ЧУПО «ВШП»)

КУРСОВОЙ ПРОЕКТ

«Разработка базы данных для видеохостинга»

Выполнил:
студент 3–го курса специальности
09.02.07 «Информационные системы и программирование»
Зеленский Максим Сергеевич
подпись: _____

Проверил:
преподаватель дисциплины,
преподаватель ЧУПО «ВШП»,
к.ф.н. Ткачев П.С.
оценка: _____
подпись: _____

Тверь, 2024 г.

Содержание

Введение

Актуальность темы

Актуальность темы исследования состоит в необходимости приобретения информации о внутреннем устройстве баз данных современных веб сервисов. Одними из самых современных сервисов и платформ с самым обширным и объемным хранилищем данных я считаю платформы для публикации видео контента. Видеохостинги – давно вошли в нашу повседневную жизнь. Кто из нас не тратил пару минут в день на просмотр коротких видео или на поиск видео уроков о том, как приготовить тортик. Но среди всех людей кому как не IT–специалисту придется работать с так называемыми базами данных.

Проблема

Для того чтобы качественно делать свою работу, автомеханик должен знать, как устроена каждая часть автомобиля, так же и в IT. Никто не запрещает, будучи frontend разработчиком не знать, как пишется функционал сайта, как устроена база данных продукта, но, когда каждый член команды разработки понимает, как работает его машина, он может не только увеличить эффективность своей работы, а также уменьшить количество работы для коллектива. Знания, а главное умение ими пользоваться – ключ к успеху. Исходя из этой информации, главной проблематикой данного исследования будет являться: малая осведомленность IT–специалистов о БД, и неумение управлять хранилищами информации.

Объект исследования

Объектом исследования являются процесс разработки базы данных, предназначенный для хранения данных с видеохостинга.

Предмет исследования

Предметом исследования является структура баз данных.

Цель работы

Научится использовать базы данных.

Задачи для достижения поставленной цели

1. Изучить литературу по теме;
2. Разработать базу данных для видеохостинга;

3. Проанализировать собранный материал, полученный опыт и прийти к определенному заключению.

Методы исследования:

- Поиск и систематизация информации;
- Разработка собственной базы данных;
- Описание полученного опыта.

Глава 1

Сёрфинг в интернете

Сёрфинг – это водный вид спорта, в котором человек (сёрфер или сёрфингист) едет на передней или нижней части движущейся волны, которая обычно движется в сторону берега [9]. В данном случае сёрфером являюсь я – пользователь, а волной, на которой пользователь едет является интернет пространство. Таким образом и появилась достаточно популярная фраза «Сёрфинг в интернете», означающая перемещение по всемирной паутине в поиске информации. Следственно – основными источниками найденной информации будут являться электронные ресурсы. Но мало знать где можно вести поиски информации, важно так же знать, как правильно и эффективней всего эту информацию находить. Для понимания дальнейших действий, я разработал план, по которому я производил сбор информации:

1. Что такое база данных
2. Каких видов они бывают
3. Как с ними работать
4. Понятие видеохостинга

Следуя составленному плану, было необходимо ознакомится с базовыми понятиями баз данных.

Что такое база данных?

База данных (БД) – это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе [12]. Это очень обширный и функциональный инструмент для работы с данными, который помогает пользователям легко получать доступ к информации, структурировать ее и обновлять информацию.

Модели данных

Если опираться на классическую теорию баз данных [10], то в ней существует такое понятие как “Модель данных”.

Модель данных – это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Проще говоря модель данных является представлением структуры БД, представлением того, как та или иная база должна выглядеть [4]. Виды БД выделяют внутри классификации по модели данных, которая включает в себя:

- Иерархические;
- Объектно–ориентированные;
- Реляционные;
- Сетевые;
- Функциональные.

Стоит отметить, что классификация по модели данных далеко не единственная. Существует классификация по среде хранения БД, описывающая способ хранения информации. Также БД могут классифицироваться по содержимому, например, могут быть географическими, историческими, научными, мультимедийными. Но затрагивать остальные классификации я не стану, обосновывая это отсутствием необходимости [1].

Перед тем как я начну перечислять характеристики описанных выше видов БД хочу ввести такое понятие как СУБД (система управления базами данных) – это комплекс программно–языковых средств, позволяющих создать базы данных и управлять данными. Иными словами, СУБД – это набор программ, позволяющий организовывать, контролировать и администрировать базы данных [14]. Соответственно под определенный вид БД нужна и своя СУБД. И так можно приступать к разновидностям БД, я дам краткое описание каждого из них:

- **Иерархическая** – первая в списке, БД основанная на древовидной структуре, состоящей из объектов, представленных в виде данных, различных уровней. Между объектами существуют связи, каждый

объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении предка (объект более близкий к корню) к потомку (объект более низкого уровня), при этом возможна ситуация, когда объект–предок имеет несколько потомков, тогда как у объекта–потомка обязателен только один предок. Объекты, имеющие общего предка, называются близнецами. Иерархическая модель одна из самых старых на данный момент [2].

- **Объектно–ориентированная (ООБД)** – вторая в списке, БД в которой данные моделируются в виде объектов, их атрибутов, методов и классов. ООБД обычно рекомендованы для тех случаев, когда требуется высокопроизводительная обработка данных, имеющих сложную структуру [5].
- **Реляционная (РМД)** – следующая в списке, БД построенная на реляционной модели. Реляционная модель данных включает в себя:
    - Структурный аспект – данные в БД представляют собой набор отношений.
    - Аспект целостности – отношения отвечают определенным условиям целостности.
    - Аспект обработки (манипулирования) – РДМ поддерживает операторы манипулирования отношениями.

  Кроме того, в состав реляционной модели данных включают теорию нормализации. Та включает в себя несколько нормальных норм: Нулевая н.ф. (UNF), Первая н.ф. (1NF), Вторая н.ф. (2NF), Третья н.ф (3NF), н.ф. Бойса–Кодда (BCNF), Четвертая н.ф. (4NF), Пятая н.ф. (5NF), Доменно–ключевая н.ф. (DKNF), Шестая н.ф. (6NF) [3]. Определенной к одной из форм характеризуется параметрами каждой уникальной БД. Одна база может принадлежать только к одной из форм [7].
- **Сетевая** – далее в списке, является расширением иерархического подхода. Разница между иерархической моделью данных и сетевой

состоит в том, что в иерархических структурах запись–потомок должна иметь в точности одного предка, а в сетевой структуре данных у потомка может иметься любое число предков [8].

- **Функциональная** – последняя в списке, БД используются для решения аналитических задач, таких как финансовое моделирование и управление производительностью. Функциональная база данных или коротко функциональная модель отличается от реляционной модели [11].

При разработке БД, выбирается та модель данных, которая больше всего подходит под поставленную задачу. На сегодняшний день преимущества реляционного подхода сделали его самой распространенной моделью для баз данных в мире [13].

Так же существует понятие “Сверхбольшая база данных” – это база данных, которая занимает чрезвычайно большой объем на устройстве физического хранения. Термин подразумевает максимально возможные объемы БД, которые определяются последними достижениями в технологиях физического хранения данных и в технологиях программного оперирования данными [1]. Этих знаний достаточно для понимания основ баз данных.

Взаимодействие с базами данных

Как я уже упоминал выше, взаимодействие с самой БД происходит посредством определенной СУБД. Если использовать РМД, то стоит учесть, что все взаимодействия ведутся на языке SQL.

SQL

SQL (Structured Query Language, или язык структурированных запросов) – это декларативный язык программирования (язык запросов), который используют для создания, обработки и хранения данных в реляционных БД [19]. Теперь перейдем к приложениям, в которых необходимо писать запросы на языке SQL. Различные источники утверждают, что топ самых популярных систем управления БД возглавляют Oracle, MySQL, Microsoft SQL Server, PostgreSQL. Именно эти ПО я и

буду рассматривать для работы. Анализ каждой программы я буду проводить исходя из плюсов и минусов каждой из представленных выше ПО.

Oracle

Первым на очереди предстоит разобрать такой инструмент для работы с БД как “Oracle Database” [6].

Плюсы:

1. Высокая производительность – Oracle Database обеспечивает высокую скорость работы и способна обрабатывать большие базы данных;
2. Поддержка нескольких баз данных – Одно из лучших преимуществ Oracle Database – возможность управления несколькими базами данных в рамках одной транзакции;
3. Обновление версий – Oracle своевременно информирует вас о предстоящем большом релизе и потенциальных изменениях, чтобы вы могли подготовиться. Oracle предлагает хорошую обратную совместимость, так что вам не придется переписывать приложение при обновлении СУБД.

Минусы:

1. Сложность – Одним из крупных недостатков Oracle Database является ее сложность. Oracle не рекомендуется для использования без достаточных технических знаний и навыков работы с Oracle Database. Oracle также не подойдет для тех, кто ищет простую в использовании и основных функциях базу данных;
2. Oracle является платным ПО, по сравнению с конкурентами, его цена выше средней по рынку. К тому же, стоимость лицензии Oracle может изменяться со временем в зависимости от различных факторов, таких как изменения в политике компании, обновления;
3. Сложность управления – Oracle обычно требует больше усилий для управления некоторыми операциями. Oracle Database оправдана только при работе с большими базами данных.

MySQL

Следующим вариантом является “MySQL Workbench”, разрабатываемый компанией Oracle, имеющий две версии. Первая “MySQL Enterprise Edition”, является платной вариацией и распространена меньше чем вторая вариация под названием “ MySQL Community Edition”, будучи бесплатной версией с открытым исходным кодом, является самой популярной СУБД в мире [17].

Плюсы:

1. **Безопасность** – MySQL имеет в своем функционале различные функции безопасности, включая возможность установки привилегий пользователя, шифрование данных и аутентификацию, чтобы обезопасить информацию, хранящуюся в базах данных;
2. **Производительность** – MySQL обладает хорошей производительностью и быстродействием благодаря оптимизированным алгоритмам выполнения запросов. Умеет работать с большими объемами данных с минимальными задержками;
3. **Масштабируемость** – MySQL может использоваться как для небольших веб-приложений, так и для серьезных корпоративных систем. СУБД предлагает различные методы масштабирования, включая горизонтальное и вертикальное масштабирование, что дает возможность расширять базы данных при возникновении необходимости. Это позволяет обрабатывать большое количество одновременных запросов и поддерживать беспрепятственный доступ к данным для пользователей;
4. **Доступность** – рассматриваемый продукт имеет открытый исходный код и может быть использован бесплатно. Это делает его доступным для большого круга разработчиков и пользователей;
5. **Развитое сообщество** – MySQL имеет большое сообщество разработчиков и пользователей, что дает возможность доступа к различным ресурсам и помощи при возникновении проблем.

Минусы

1. **Отказоустойчивость** – MySQL может иметь проблемы с производительностью при обработке больших объемов данных или, когда

требуется обработка сложных запросов или, подключение большого количества клиентов. Это может привести к замедлению работы или даже отказу в обслуживании;

2. **Масштабируемость** – MySQL может столкнутся с ограничением масштабируемости при работе с большим количеством клиентов. Тогда отказоустойчивость и возможность горизонтального масштабирования могут быть ограничены;
3. **Сложность** – MySQL может быть сложным для администрирования, особенно для новичков. Настройка и оптимизация параметров конфигурации может быть сложной задачей, требующей глубоких знаний и опыта;
4. **Отсутствие обновлений** – MySQL достаточно редко обновляется, привнося лишь небольшие изменения. Глобальных изменений не было с выхода последней версии в 2016 году [16]. Ситуацию спасает широко развитое сообщество и прочие плюсы данного ПО.

Microsoft SQL Server

Следующим продуктом на досмотр выходит программа “SQL Server”, разрабатываемая компанией Microsoft [15].

Плюсы:

1. **Масштабирование** – взаимодействовать с ней можно на портативных ПК или мощной мультипроцессорной технике. Процессор может одновременно обрабатывать большой объем запросов;
2. **Размер страниц** – до 8 кб, поэтому данные извлекаются быстро, подробную и сложную информацию хранить удобнее. Возможно сложно понять в чём тут плюс, но в больших компаниях постоянно следят за цифровым весом итогового продукта и SQL Server даёт возможность экономить пространство на цифровых носителях;
3. **Автоматизированное администрирование** – такие процессы как управление блокировками, памятью, редактура размеров файлов. У системы продуманы настройки, можно создать профили пользователей. Это облегчает освоение для новичков;

4. **Поддержка Microsoft** – Поддержка работы с другими решениями Майкрософт.

Минусы:

1. **Кроссплатформенность** – программу можно установить только на операционные системы от Microsoft, т.е. Windows;
2. **Доступность** – рассматриваемый продукт, является платным решением, заставляя выбрать другое ПО свободным разработчикам и маленьким компаниям.

PostgreSQL

Последним пунктом в списке остался запоминающийся Слоник, программа PostgreSQL, основанная Майклом Стоунбрейкером, разработка над которой ведётся командой PostgreSQL [18].

Плюсы:

1. **Объектно-реляционная модель** – PostgreSQL — объектно-реляционная СУБД. Это значит, что она поддерживает и объектный, и реляционный подход. Это делает данное ПО более гибким для разработки по сравнению с остальными;
2. **Поддержка множества типов данных** – Еще одна особенность PostgreSQL — поддержка большого количества типов записи информации. Это не только стандартные целочисленные значения, числа с плавающей точкой, строки и булевы значения («да/нет»), но и денежный, геометрический, перечисляемый, бинарный и другие типы;
3. **Работа с большими объемами** – В большинстве СУБД, рассчитанных на средние и небольшие проекты, есть ограничения по объему базы и количеству записей в ней. В PostgreSQL ограничений нет;
4. **Поддержка сложных запросов** – PostgreSQL работает со сложными, составными запросами. Система справляется с задачами разбора и выполнения трудоемких операций, которые подразумевают и чтение, и запись, и валидацию одновременно. Она медленнее аналогов, если речь заходит только о чтении, но в других аспектах превосходит конкурентов;

5. **Написание функций на нескольких языках** – В PostgreSQL можно писать собственные функции — пользовательские блоки кода, которые выполняют те или иные действия. Эта возможность есть практически в любых СУБД, но PostgreSQL поддерживает больше языков, чем аналоги. Кроме стандартного SQL, в PostgreSQL можно писать на C и C++, Java, Python, PHP, Lua и Ruby;
6. **Одновременная модификация базы** – Важная особенность PostgreSQL — возможность одновременного доступа к базе с нескольких устройств. В СУБД реализована клиент-серверная архитектура, когда база данных хранится на сервере, а доступ к ней осуществляется с клиентских компьютеров;
7. **Доступность** – PostgreSQL — ПО с открытым исходным кодом, которое распространяется по свободной лицензии. Это означает, что любой разработчик может посмотреть, как написана система, или предложить для нее свои правки. СУБД разрабатывается сообществом энтузиастов и в определенной степени никому не принадлежит, а значит, ее можно свободно и без ограничений использовать в своих проектах;
8. **Кроссплатформенность** – Чаще всего PostgreSQL используют на серверах с операционными системами семейства Linux, но СУБД поддерживает и другие ОС. Ее можно установить в системы на базе Windows, BSD, macOS и Solaris.

Минусы:

1. **Сложности при настройке** – У PostgreSQL очень обширный набор возможностей. Очевидно, что такое разнообразие функций влечёт за собой множество настроек, что может вызвать затруднения у новичков. Корректная настройка базы данных требует глубокого понимания архитектуры и параметров;
2. **Повышенное потребление ресурсов** – В сравнении с некоторыми другими СУБД, PostgreSQL может потреблять больше ресурсов (включая оперативную память и процессорное время). Это особенно заметно при работе с большими объёмами данных и выполнении сложных запросов;

3. **Отсутствие некоторых функций** – По сравнению с определенными коммерческими аналогами PostgreSQL может немного уступать в функциональности.

Стоит отметить, что перечисленные недостатки в основном применимы к конкретным сценариям использования. В общем плане PostgreSQL остаётся одной из самых мощных и популярных открытых СУБД.

На этом описание базовых понятий, принципов и аспектов работы с БД, а также различных возможностей разработки БД я считаю завершённым. Далее следует разобраться с понятием “Видеохостинг”.

Введение в видеохостинги

Видеохостинг (от лат. video и англ. hosting) – веб-сервис, позволяющий загружать и просматривать видео в браузере, например, через специальный проигрыватель. При этом большинство подобных сервисов не предоставляют видео, следуя таким образом принципу «контент генерирует пользователь» («User-generated content»). Самым лучшим и популярным примером видеохостинга, является “YouTube” – веб-сервис, разрабатываемый компанией Google. Изначально YT разрабатывался командой в которой состояли: Стив Чен, Чад Хёрли, Джавед Карим. Я предполагаю, что YouTube стал первым видеохостингом в мире, хотя я не нашёл информации опровергающей это высказывание, я так же не нашёл информации подтверждающей его. Будем считать, что YT был первым, а если это не является правдой, то он точно являлся толчком в мире веб-сервисов для загрузки и просмотра видео контента. Основан самый популярный видеохостинг в 2005 году, и с тех пор подобных веб приложений стало огромное множество.

В топ самых популярных веб-сервисов входят следующие решения: YouTube, Twitch, Vimeo, Dailymotion, Google Drive. У каждого из перечисленных приложений своя политика и принципы, но суть одинакова: пользователь регистрируется, тем самым создав свой канал, выгружает на него собственные или не очень видео, те в свою очередь загружаются в БД, после чего к просмотру видео открывается доступ и другим пользователям. Не стоит забывать и про взаимодействие пользователей между собой, проявляющееся в виде комментирования видео, возможности ставить положительные или отрицательные оценки, с английского их называют “Like” и “Dislike”, а также возможность подписки на определённый канал, для того чтобы отслеживать информацию о новых видео на канале.

Так же существуют решения, позволяющие вести поток в реальном времени, называемый прямым эфиром. Данную возможность сервисов для видеохостинга я не рассматриваю. Данной информации достаточно для начала работы над собственной БД для видеохостинга.

Глава 2

Начало работы

В рамках курсового исследовательского проекта у меня есть технические требования к моей базе данных. В рамках моей дисциплины обязательно использование как минимум одной ER-диаграммы, отображающей все сущности и связи разрабатываемой БД.

Разрабатываемая БД так же должна включать:

● минимум пять связанных таблиц

● минимум две отдельные роли (хотя бы одну кроме роли администратора)

● минимум пять типовых запросов к БД, отражающих соответствующие бизнес-процессы сферы, для которой она разрабатывается

● минимум одну транзакцию

● минимум три локальных переменных с заданным типом данных

● минимум одно условие

● минимум одну хранимую процедуру

● минимум одно представление

● минимум один триггер

● минимум одну пользовательскую функцию

● минимум один обработчик исключений

Все перечисленные функциональные объекты БД должны быть представлены с описанием принципа работы и назначения.

Кроме того, ваша БД должна находиться в одной из перечисленных нормальных форм:

● 3NF (третья нормальная форма)

● BCNF (нормальная форма Бойса-Кодда)

● 4NF (четвертая нормальная форма).

Модель данных

Разработку БД я начну с выбора модели данных. В первой главе я уже перечислял все модели данных. В них были: Иерархическая, Объектно-

ориентированная, Реляционная, Сетевая и Функциональная модели, их можно подразделить на две категории “Реляционная” и так называемая “No-SQL” модель. Виды, подходящие под вторую категорию, все кроме реляционных и объектно-ориентированных, мне не подходят. Обоснован этот выбор сложностью освоения моделей в категории “No-SQL”, а также и не пригодность для разработки видеохостинга. Из оставшихся вариантов, наиболее подходящим является реляционный подход к разработке, из-за его структуры и относительно низкой систематизации, СУБД сделает все за меня. А что на счет СУБД?

Система управления базами данных

Выбор будет состоять из списка самых популярных ПО, в который входят: Oracle Database, MySQL, Microsoft SQL Server и PostgreSQL. Из всех решений, я сразу отбрасываю платные варианты, необходимость в их функционале у меня отсутствует, так же, как и средства на покупку данных ПО. Исходя из ранее сказанного выбор остается между MySQL и PostgreSQL. Взвесив все за и против, взгляд сразу упал на MySQL. У этой СУБД достаточно обширное сообщество, следовательно, если я столкнусь какой-либо проблемой, вероятность существования ответа на неё, будет достаточно высока. Так же имеет значение и скорость работы ПО. PostgreSQL довольно медлительна, в то время как MySQL работает достаточно быстро и отлично подходит для разработки БД под веб-сайты.

Типы данных

- Int / Integer – Числовой тип данных предоставляет целые числа от – 2147483648 до 2147483647;
- Varchar(x) – Текстовый тип данных предоставляющий возможность указать длину от – 0 до 65,535 символов;
- Date – Хранит значение даты в виде ГГГГ-ММ-ДД. Допустимый диапазон от 1000-01-01 до 9999-12-31;
- Datetime – Хранит значение даты и времени в виде ГГГГ-ММ-ДД, ЧЧ:ММ:СС. Допустимый диапазон от 1000-01-01 00:00:00 до 9999-12-31 23:59:59;
- Text – Содержит текстовые строки. Максимальная длинна 65,535;

- Enum – тип данных содержащий набор констант. Константы указываются вручную, и атрибут использующий данный тип данных не может равняться значениям вне константы.

Проектировка базы данных

MySQL предоставляет обширный каталог инструментов. Одним из таких является возможность создавать ER-диаграммы прямо внутри ПО. Используя данный инструмент, я буду создавать таблички для определения взаимодействия таблиц между собой.

Всего в моей БД планируется 8 табличек: users, channels, videos, subscribers, playlists, video_has_playlists, comments, likes

1. Первое с чего стоит начать это таблица users (пользователи). Данная табличка включает в себя данные пользователя при регистрации. Это будет основой для всей БД.

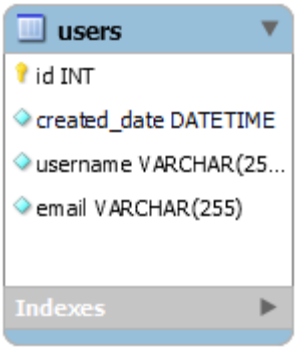

Рис.1

1.1. В первой сточке описан атрибут id, использующий тип данных Integer. Так же данный атрибут использует следующие параметры: Not Null – это означает что id не может быть равен нулю, и всегда должен быть заполнен; Auto_increment – это означает что id будет автоматически инкрементироваться, т.е. увеличивать собственное значение при добавлении новой записи в эту табличку; Primary Key – это означает что атрибут id является уникальным во всей таблице;

1.2. Атрибут created_date, использующий тип данных Datetime. Так же данный атрибут использует следующие параметры: Not Null;

1.3. Атрибут username, использующий тип данных Varchar (255). Так же данный атрибут использует следующие параметры: Not Null;

1.4. Атрибут email, использующий тип данных Varchar (255). Так же данный атрибут использует следующие параметры: Not Null.

2. Второй на очереди стоит таблица channels (каналлы). Данная табличка включает в себя все данные пользовательского канала.

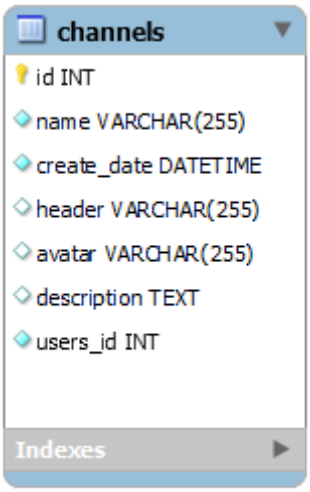

Рис.2

2.1. Атрибут id в этой табличке существует с теми же параметрами что и в предыдущей таблице

2.2. Атрибут name, использующий тип данных Varchar (255). Так же данный атрибут использует дополнительные параметры: Not Null;

2.3. Атрибут create_date, использующий тип данных Datetime. Так же данный атрибут использует дополнительные параметры: Not Null;

2.4. Атрибут header, использующий тип данных Varchar (255). Данный атрибут не использует дополнительные параметры;

2.5. Атрибут avatar, использующий тип данных Varchar (255). Данный атрибут не использует дополнительные параметры;

2.6. Атрибут description, использующий тип данных Text. Данный атрибут не использует дополнительные параметры;

2.7. Атрибут users_id, использующий тип данных Integer. Данный атрибут использует дополнительные параметры: Not Null; Foreign Key (users_id) References users (id) – это значит, что эта таблица связана с таблицей users посредством первичного ключа users.id.

3. Третьей таблицей является videos (видео). В это таблице хранятся данные каждого отдельного видео.

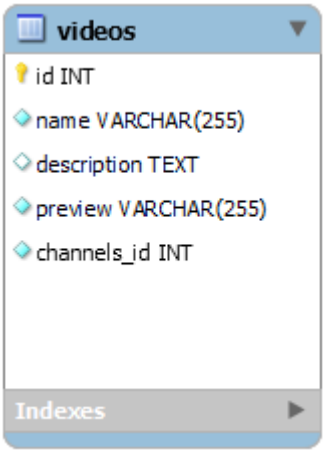

Рис.3

3.1. Атрибут id в этой табличке существует с теми же параметрами что и в остальных таблицах;

3.2. Атрибут name в этой табличке существует с теми же параметрами что и в остальных таблицах;

3.3. Атрибут description в этой табличке существует с теми же параметрами что и в остальных таблицах;

3.4. Атрибут preview в этой табличке существует, используя все параметры, используемые атрибутом name;

3.5. Атрибут channels_id, использующий тип данных Integer. Данный атрибут использует дополнительные параметры: Not Null; Foreign Key (channels_id) References channels (id) – это значит, что эта таблица связана с таблицей channels посредством первичного ключа channels.id.

4. Следующей таблицей будет таблица playlists (плейлисты). Данная табличка хранит данные о всех плейлистах пользователя между таблицами существует еще одна таблица, которая связывает оба объекта. Такой тип связи называется многие ко многим (смотреть рис.4).

4.1. Атрибут id в этой табличке существует с теми же параметрами что и в остальных таблицах;

4.2. Атрибут name в этой табличке существует с теми же параметрами что и в остальных таблицах;

4.3. Атрибут description в этой табличке существует с теми же параметрами что и в остальных таблицах;

4.4. Атрибут availability, использующую тип данных Enum (‘public’, ‘link’, ‘closed’) – это значит, что атрибут может бать равен только одному из параметров, перечисленных внутри;

4.5. Атрибут channels_id, использующий тип данных Integer. Данный атрибут использует дополнительные параметры: Not Null; Foreign Key (channels_id) References channels (id) – это значит, что эта таблица связана с таблицей channels посредством первичного ключа channels.id.

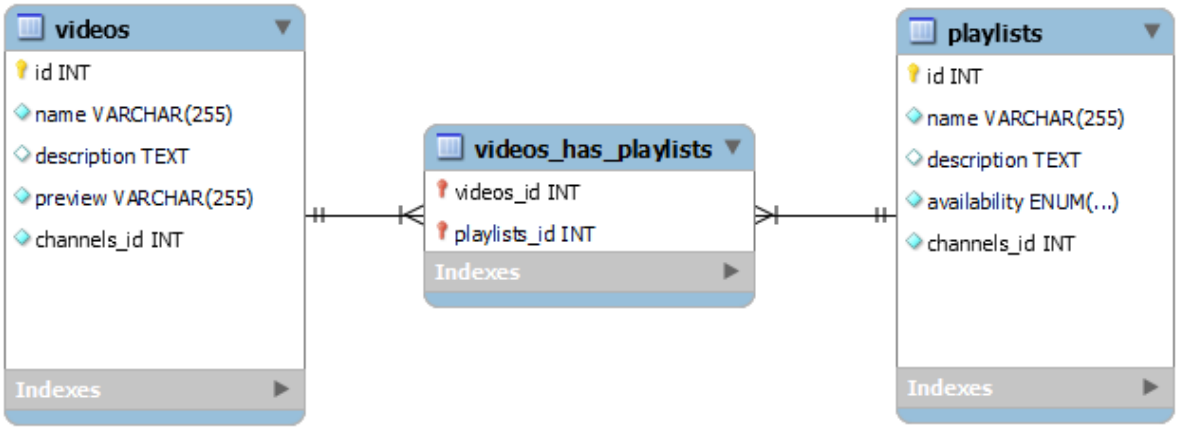

Рис.4

5. Дальше по списку идёт таблица subscribers (подписчики). В это табличке хранятся данные подписок на каждый канал. Эта таблица связывает два объекта.

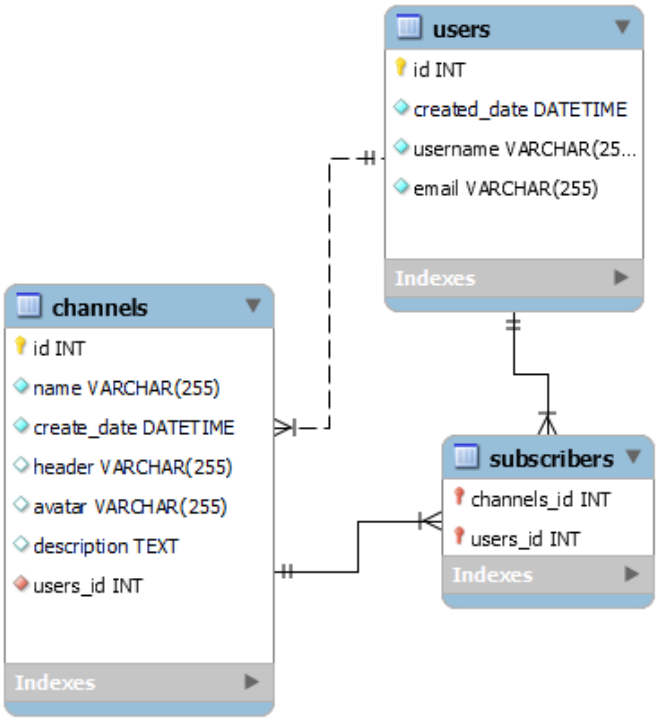

Рис.5

6. В шестом пункте будут содержатся сразу две таблички, относящиеся как к channels, так и к videos – это таблички likes и comments. Они содержат в себе информацию о лайках и комментариях на определённых видео от определённого канала (смотреть рис.6).
   6.1. Атрибут id в двух табличках существует с теми же параметрами что и в остальных таблицах;
   6.2. Атрибут create_date в этой табличке существует с теми же параметрами что и в остальных таблицах;
   6.3. Атрибут create_date в этой табличке существует с теми же параметрами что и в остальных таблицах;
   6.4. Атрибут videos_id в двух таблицах, использующий тип данных Integer. Так же данный атрибут использует следующие дополнительные параметры: Not Null; Foreign Key (videos_id) References videos (id);

6.5. Атрибут channels_id в двух таблицах, использующий тип данных Integer. Так же данный атрибут использует следующие дополнительные параметры: Not Null; Foreign Key (channels_id) References channels (id);

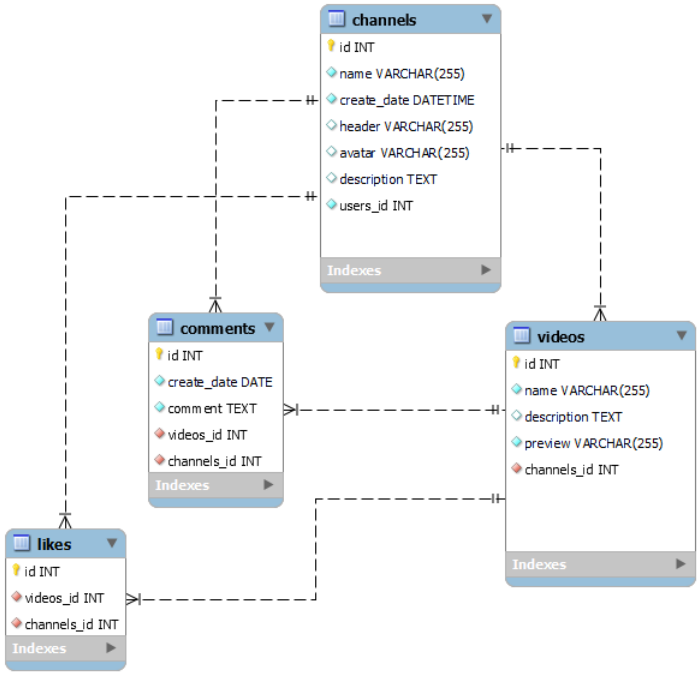

Рис.6

После создания всех таблиц и подключения связей между ними у меня получилась полная ER-диаграмма, состоящая из 8 таблиц, связанных между собой. (Смотреть рис.7)

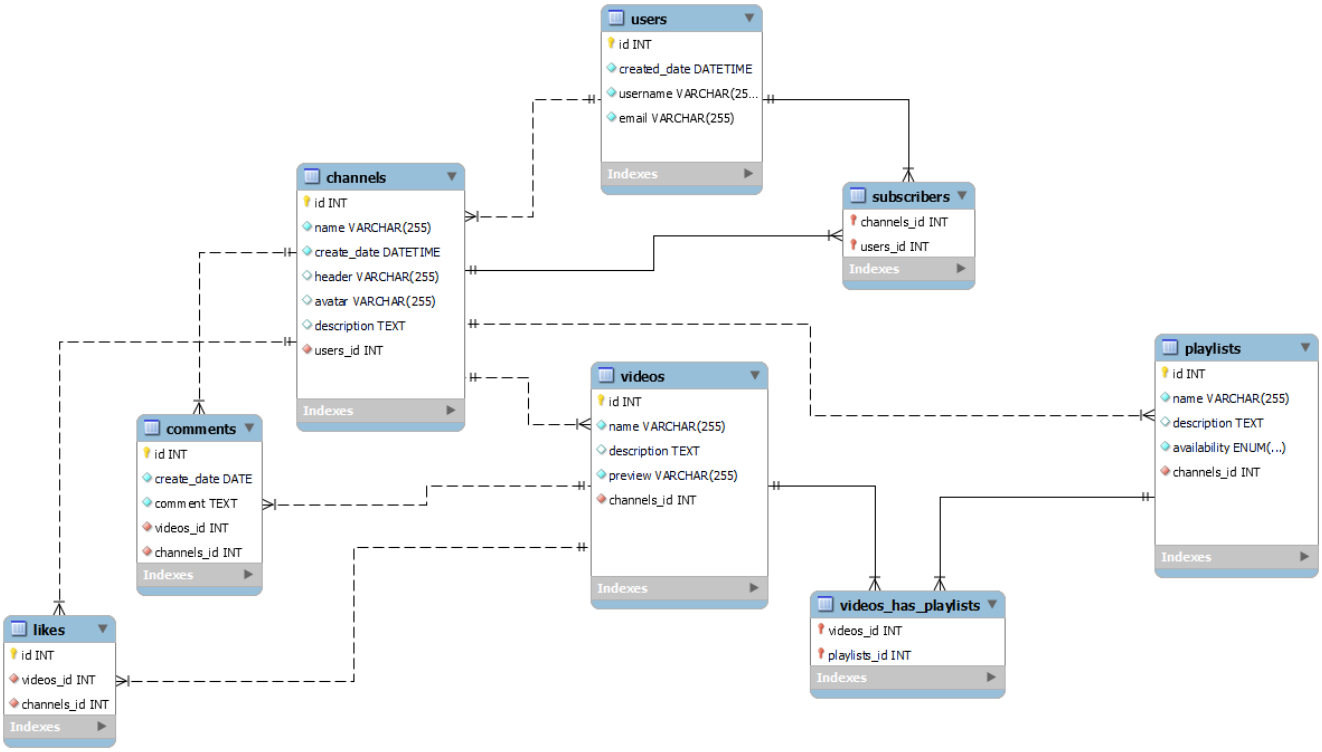

Рис.7

Хранимые процедуры

Хранимая процедура — это объект базы данных, представляющий собой набор SQL-инструкций, который компилируется один раз и хранится на сервере. MySQL поддерживает использование хранимых процедур, которые могут вызываться по требованию приложениями, управляемыми данными. Каждая хранимая процедура является именованным объектом базы данных, которая содержит процедурный код, состоящий из одного или более операторов SQL. Когда приложение вызывает хранимую процедуру, MySQL выполняет эти операторы и возвращает результаты в приложение. В моей БД прописана 1 хранимая процедура под названием “DeleteUnwantedComments”, содержащая в себе транзакцию и условие

Транзакция — группа последовательных операций с базой данных, которая представляет собой логическую единицу работы с данными.

В системе управления реляционными базами данных условия используются в разделах Where и Having операторов Select, Update, Delete, чтобы ограничить подмножество значений, с которыми они работают.

Рассмотрим работу процедуры DeleteUnwantedComments (Приложение 1). После запуска процедуры, программа потребует ввести значение под названием “UnwantedText”. После того как некий текст был получен, будет идти сравнение unwantedText и поля таблицы comments.comment. После произойдет проверка с условием, если количество комментариев, в которых совпало значение, больше чем 0, то данные записи внутри таблички comments удаляются, если же совпадений найдено не было, то СУБД выведет на сообщение с текстом “No unwanted text found” – “Не желательного текста найдено не было”. Таким образом модераторы могут удалять комментарии с нежелательным содержанием со всех видео. Так же по мимо текста в запрос данных процедуре можно ввести url ссылку после чего все комментарии с не нужным медиа контентом будут удалены.

Триггеры

Триггеры в SQL - это специальные хранимые процедуры, которые автоматически выполняются при возникновении определенного события в базе данных, такого как вставка, обновление или удаление данных из таблицы. В моей БД прописан 1 триггер, который взаимодействует с именами каналов. Внутри триггера прописываются запрещенные имена каналов. Если любой юзер создаст канал с названием или изменит название канала на слово из запрещенного списка, то канал и все связанные с ним записи автоматически удалятся.

Сам триггер называется “DeleteBadnameChannels”, его код выглядит следующим образом

```
--Создание триггера
CREATE TRIGGER `DeleteBadnameChannels`
BEFORE DELETE ON `channels`
FOR EACH ROW
BEGIN
--Проверка на совпадение имен канала с запрещенными словами
    IF OLD.name LIKE '%nigger%' OR OLD.name LIKE '%bitch%' OR OLD.name LIKE '%fuck%' THEN
--Удаления данных
        DELETE FROM videos_has_playlists
        WHERE videos_id IN (SELECT id FROM videos WHERE channels_id = OLD.id);
        DELETE FROM comments WHERE channels_id = OLD.id;
        DELETE FROM likes WHERE channels_id = OLD.id;
        DELETE FROM subscribers WHERE channels_id = OLD.id;
        DELETE FROM playlists WHERE channels_id = OLD.id;
        DELETE FROM videos WHERE channels_id = OLD.id;
    END IF;
END$$
```

Представление

Представление — это виртуальная таблица, содержимое которой определяется запросом. Как и таблица, представление состоит из ряда именованных столбцов и строк данных. Пока представление не будет проиндексировано, оно не существует в базе данных как хранимая совокупность значений. В моей БД предусмотрено одно представление под названием “ChannelStatistics”. Оно создано для просмотра глобальной статистики со всех каналов.

```
--Создание представления
CREATE VIEW ChannelStatistics AS
SELECT
    channels.id AS channel_id,
    channels.name AS channel_name,
    channels.create_date,
    COUNT(DISTINCT videos.id) AS total_videos,
    COUNT(DISTINCT comments.id) AS total_comments,
    COUNT(DISTINCT likes.id) AS total_likes
FROM
    channels
    LEFT JOIN videos ON channels.id = videos.channels_id
    LEFT JOIN comments ON channels.id = comments.channels_id
    LEFT JOIN likes ON channels.id = likes.channels_id
GROUP BY
    channels.id, channels.name, channels.create_date;
```

Максимально простое и удобное представление. Отзывает необходимость каждый раз писать запросы для просмотра общей статистики.

Пользовательская функция

Функция пользователя представляет собой группу произвольных операторов SQL, предназначенных для выполнения некоторой задачи. Эти функции не поставляются из коробки и обычно создаются для обработки специфичных сценариев. Можно выполнять операции select, insert, update, delete внутри функции.

В моей базе данных прописана 1 пользовательская функция под названием GetChannelStats, которая запрашивает id канала по которому нужно собрать общее количество лайков видео и комментариев на канале.

```
--Создание пользовательской функции
CREATE FUNCTION GetChannelStats(channel_id INT)
RETURNS JSON
DETERMINISTIC
BEGIN
--Создание переменных
    DECLARE total_videos INT;
    DECLARE total_comments INT;
    DECLARE total_likes INT;
    DECLARE result JSON;

    SELECT COUNT(*) INTO total_videos
    FROM videos
    WHERE channels_id = channel_id;

    SELECT COUNT(*) INTO total_comments
    FROM comments
    WHERE channels_id = channel_id;

    SELECT COUNT(*) INTO total_likes
    FROM likes
    WHERE channels_id = channel_id;

    SET result = JSON_OBJECT(
        'total_videos', total_videos,
        'total_comments', total_comments,
        'total_likes', total_likes
    );

    RETURN result;
END //
```

Типовые запросы

Типовые запросы призваны отобразить внутренние бизнес процессы.

1. Вывести комментарии и их даты с канала “Channel 2”

```
SELECT comments.create_date, comments.comment
FROM comments
JOIN channels ON comments.channels_id = channels.id
WHERE channels.name LIKE 'Channel 2';
```

2. Вывести имя плейлистов и количество видео в них с канала “Channel 1” где доступность равняется публичному

```
SELECT playlists.name, COUNT(videos_has_playlists.videos_id) AS
videos
FROM playlists
JOIN videos_has_playlists ON videos_has_playlists.playlists_id =
playlists.id
JOIN channels ON playlists.channels_id = channels.id
WHERE channels.name LIKE 'Channel 1' and playlists.availability =
'public'
GROUP BY playlists.name;
```

3. Вывести количество пользователей чьи аккаунты было созданы в 2023 году

```
SELECT COUNT(users.id) AS 'Users'
FROM users
WHERE users.created_date <= '2023-01-01';
```

4. Вывести лайки с каждого канала и отсортировать по наибольшему количеству

```
SELECT channels.name, COUNT(likes.id) AS 'Video_likes'
FROM likes
JOIN channels ON likes.channels_id = channels.id
JOIN videos ON likes.videos_id = videos.id
GROUP BY channels.name
ORDER BY 'Video_likes' ASC;
```

5. Вывести имена людей создавших свой канал

```
SELECT users.username, channels.name
FROM users
JOIN channels ON channels.users_id = users.id
WHERE channels.id > 0 || channels.id != 0;
```

Роли

Роли работают как шаблоны, помогающие предоставлять права пользователям баз данных. Поддерживаются следующие роли пользователей баз данных: Чтение и запись (используется по умолчанию), только чтение и только запись.

В моей БД по мимо встроенных ролей есть три собственные роли.

Роль admin_ru_server

```
CREATE USER ‘admin_ru_server’@’%’ IDENTIFIED BY ‘admin_ru_server’;
GRANT ALL PRIVILEGES ON *.* TO admin_ru_server’@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

Роль developer

```
CREATE USER ‘developer’@’%’ IDENTIFIED BY ‘developer’;
GRANT SELECT, INSERT, UPDATE, DELETE ON *.* TO 'developer'@'%';
FLUSH PRIVILEGES;
```

Роль moderator

```
CREATE USER ‘moderator’@’localhost’ IDENTIFIED BY ‘moderator’;
GRANT SELECT, INSERT, UPDATE ON ‘channels’ TO ‘moderator’@’localhost’;
FLUSH PRIVILEGES;
```

Заключение

В ходе выполнения курсовой работы на тему "Разработка базы данных для видеохостинга" была достигнута основная цель - приобретение навыков использования баз данных. В процессе работы были решены следующие задачи:

- Изучить литературу по теме: Проведён анализ литературы и различных источников по теме. Было проведено углубление в тему Баз Данных
- Разработать базу данных для видеохостинга: Разработана собственная БД, при использовании реляционных баз данных и MySQL в качестве СУБД.
- Проанализировать собранный материал, полученный опыт и прийти к определенному заключению: Проведя тщательную работу надо изучением темы и разработки собственной БД, приведя свои умозаключения к определённой форме, я могу сказать, что работа позволила получить ценные практические навыки в области проектирования, реализации и оптимизации баз данных, полученные знания и опыт могут быть использованы для решения более сложных задач в будущем и внесут значительный вклад в моё будущее.

Список источников

1. База данных – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/База_данных
2. Иерархическая модель данных – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Иерархическая_модель_данных
3. Кафедра ИТ – it.vshp.online – [Электронный ресурс] – режим доступа: https://it.vshp.online
4. Модель данных – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Модель_данных
5. Объектно–ориентированная база данных – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/ https://ru.wikipedia.org/wiki/Объектно–ориентированная_база_данных
6. Плюсы и минусы Oracle Database – UniwexSoft – [Электронный ресурс] – режим доступа: https://blog.uniwex.io/plyusy-i-minusy-oracle-database/
7. Реляционная модель данных – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Реляционная_модель_данных
8. Сетевая модель данных – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Сетевая_модель_данных
9. Серфинг – Wikipedia [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Серфинг
10. Теория баз данных – Backend interview – [Электронный ресурс] – режим доступа: https://backendinterview.ru/db/dBTheory/index.html
11. Функциональные базы данных – Cyclowiki – [Электронный ресурс] – режим доступа: https://cyclowiki.org/wiki/Функциональные_базы_данных
12. Что такое база данных – Oracle [Электронный ресурс] – режим доступа: https://www.oracle.com/cis/database/what–is–database/
13. Что такое реляционная база данных – Oracle – [Электронный ресурс] – режим доступа: https://www.oracle.com/cis/database/what–is–a–relational–database/#:~:text=Реляционные%20базы%20данных%20появились%20в,для%20баз%20данных%20в%20мире

14. Что такое СУБД? Наиболее популярные СУБД – Nic.ru [Электронный ресурс] – режим доступа: https://www.nic.ru/help/chto–takoe–subd_8580.html
15. Что такое SQL Server: плюсы и минусы использования – Muzeon – [Электронный ресурс] – режим доступа: https://muzeon.ru/medicina/2912-chto-takoe-sql-server-plyusy-i-minusy-ispolzovaniya.html
16. MySQL – Wikipedia – [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/MySQL
17. MySQL Преимущества и недостатки – Servergate – [Электронный ресурс] – режим доступа: https://servergate.ru/articles/mysql-preimushchestva-i-nedostatki/
18. PostgreSQL: что это за СУБД, основы и преимущества – Skillfactory Media – [Электронный ресурс] – режим доступа: https://blog.skillfactory.ru/glossary/postgresql/
19. SQL: что это, в каких базах его используют и как работать с языком программирования – Skillbox Media – [Электронный ресурс] – режим доступа: https://skillbox.ru/media/code/chto-takoe-sql-kak-ustroen-zachem-nuzhen-i-kak-s-nim-rabotat/

Приложение 1

Хранимая процедура для удаления нужного комментария

```
--Создание процедуры
CREATE PROCEDURE DeleteUnwantedComments(IN unwanted_text TEXT)
BEGIN
    DECLARE rows_count INT DEFAULT 0;

    SELECT COUNT(*) INTO rows_count
    FROM comments
    WHERE comments.comment LIKE CONCAT('%', unwanted_text, '%');

    IF rows_count > 0 THEN
        START TRANSACTION;
               DELETE FROM comments
               WHERE     comments.comment     LIKE     CONCAT('%',
unwanted_text, '%');
        COMMIT;
    ELSE
        SELECT 'No unwanted comments found' AS message;
    END IF;
END //

DELIMITER ;
```

Приложение 2

Ссылка для ручного ввода

```
https://github.com/BaldurusExspa/VSHP_DataBase
```

Уважаемый пользователь!
Обращаем ваше внимание, что система Антиплагиус отвечает на вопрос, является тот или иной фрагмент текста заимствованным или нет. Ответ на вопрос, является ли заимствованный фрагмент именно плагиатом, а не законной цитатой, система оставляет на ваше усмотрение.

Отчет о проверке № 8896678

Дата выгрузки: 2024-06-19 23:08:22
Пользователь: deswasdtail285.288.098@gmail.com, ID: 8896678

Отчет предоставлен сервисом «Антиплагиат»
на сайте antiplagius.ru/

Информация о документе

№ документа: 8896678
Имя исходного файла: Зеленский М - курсовой проект.pdf
Размер файла: 0.68 МБ
Размер текста: 35058
Слов в тексте: 5171
Число предложений: 385

Информация об отчете

Дата: 2024-06-19 23:08:22 - Последний готовый отчет
Оценка оригинальности: 73%
Заимствования: 27%

73.91% 26.09%

Источники:

Доля в тексте	Ссылка
63.20%	https://en.ppt-online.org/1277187
63.20%	https://ppt-online.org/1277187
56.05%	https://blog.skillfactory.ru/glossary/postgresql/
34.30%	https://blog.uniwex.io/plyusy-i-minusy-oracle-database/
31.60%	https://lms.kgeu.ru/mod/book/view.php?id=77952&chapterid=3854
23.90%	https://iaassaaspaas.ru/subd/postgresql
21.40%	https://ru.wikipedia.org/wiki/%D0%98%D0%B5%D1%80%D0%B0%D1%80%D1%...
19.00%	http://www.energyed.ru/Inform/PCCh07
17.90%	https://www.klyaksa.net/htm/exam/answers/a21.htm
14.60%	https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C...
12.30%	https://infourok.ru/baza-dannyh-ierarhicheskaya-model-bazy-danny...
11.40%	https://infopedia.su/19x550c.html
11.25%	https://skillbox.ru/media/code/postgresql-vsye-chto-nuzhno-znat-...
9.60%	https://otherreferats.allbest.ru/programming/00334784_0.html
9.30%	https://studfile.net/preview/1671852/page:25/
9.20%	https://spravochnick.ru/lektoriy/baza-dannyh-relyacionnaya-bd-kl...

Доля в тексте	Ссылка
7.80%	http://www.programmer-lib.ru/delphi_page.php?id=9
7.60%	https://fileenergy.com/prochee/mysql-populyarnaya-sistema-upravl...
7.50%	https://webdesigngroup.ru/2023/11/03/chto-takoe-mysql-i-zachem-o...

Информация о документе:

Частное учреждение профессионального образования "Высшая школа предпринимательства" (ЧУПО "ВШП") КУРСОВОЙ ПРОЕКТ "Разработка базы данных для видеохостинга" Выполнил: студент 3-го курса специальности 09.02.07 "Информационные системы и программирование" Зеленский Максим Сергеевич подпись: _____ Проверил: преподаватель дисциплины, преподаватель ЧУПО "ВШП", к.ф.н. Ткачев П.С. оценка: _____ подпись: _____ Тверь, 2024 г. Содержание Введение 3 Глава 1 5 Сёрфинг в интернете 5 Что такое база данных? 6 Модели данных 7 Взаимодействие с базами данных 9 Введение в видеохостинги 16 Глава 2 17 Начало работы 17 Проектировка базы данных 20 Хранимые процедуры 27 Триггеры 28 Представление 29 Пользовательская функция 30 Типовые запросы 31 Роли 32 Заключение 33 Список источников 34 Приложение 1 36 Введение Актуальность темы Актуальность темы исследования состоит в необходимости приобретения информации о внутреннем устройстве баз данных современных веб сервисов. Одними из самых современных сервисов и платформ с самым обширным и объемным хранилищем данных я считаю платформы для публикации видео контента. Видеохостинги - давно вошли в нашу повседневную жизнь. Кто из нас не тратил пару минут в день на просмотр коротких видео или на поиск видео уроков о том, как приготовить тортик. Но среди всех людей кому как не IT-специалисту придется работать с так называемыми базами данных. Проблема Для того чтобы качественно делать свою работу, автомеханик должен знать, как устроена каждая часть автомобиля, так же и в IT. Никто не запрещает, будучи frontend разработчиком не знать, как пишется функционал сайта, как устроена база данных продукта, но, когда каждый член команды разработки понимает, как работает его машина, он может не только увеличить эффективность своей работы, а также уменьшить количество работы для коллектива. Знания, а главное умение ими пользоваться - ключ к успеху. Исходя из этой информации, главной проблематикой данного исследования будет являться: малая осведомленность IT- специалистов о БД, и неумение управлять хранилищами информации. Объект исследования Объектом исследования являются процесс разработки базы данных, предназначенный для хранения данных с видеохостинга. Предмет исследования Предметом исследования является структура баз данных. Цель работы Научится использовать базы данных. Задачи для достижения поставленной цели 1. Изучить литературу по теме; 2. Разработать базу данных для видеохостинга; 3. Проанализировать собранный материал, полученный опыт и прийти к определенному заключению. Методы исследования: • Поиск и систематизация информации; • Разработка собственной базы данных; • Описание полученного опыта. Глава 1 Сёрфинг в интернете Сёрфинг - это водный вид спорта, в котором человек (сёрфер или сёрфингист) едет на передней или нижней части движущейся волны, которая обычно движется в сторону берега [9]. В данном случае сёрфером являюсь я - пользователь, а волной, на которой пользователь едет является интернет пространство. Таким образом и появилась достаточно популярная фраза "Сёрфинг в интернете", означающая перемещение по всемирной паутине в поиске информации. Следственно - основными источниками найденной информации будут являться электронные ресурсы. Но мало знать где можно вести поиски информации, важно так же знать, как правильно и эффективней всего эту информацию находить. Для понимания дальнейших действий, я разработал план, по которому я производил сбор информации: 1. Что такое база данных 2. Каких видов они бывают 3. Как с ними работать 4. Понятие видеохостинга Следуя составленному плану, было необходимо ознакомится с базовыми понятиями баз данных. Что такое база данных? База данных (БД) - это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе [12]. Это очень обширный и функциональный инструмент для работы с данными, который помогает пользователям легко получать доступ к информации, структурировать ее и обновлять информацию. Модели данных Если опираться на классическую теорию баз данных [10], то в ней существует такое понятие как "Модель данных". Модель данных - это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Проще говоря модель данных является представлением структуры БД, представлением того, как та или иная база должна выглядеть [4]. Виды БД выделяют внутри классификации по модели данных, которая включает в себя: • Иерархические; • Объектно-ориентированные; • Реляционные; • Сетевые; • Функциональные. Стоит отметить, что классификация по модели данных далеко не единственная. Существует классификация по среде хранения БД, описывающая способ хранения информации. Также БД могут классифицироваться по содержимому, например, могут быть географическими, историческими, научными, мультимедийными. Но затрагивать остальные классификации я не стану, обосновывая это отсутствием необходимости [1]. Перед тем как я начну перечислять характеристики описанных выше видов БД хочу ввести такое понятие как СУБД (система управления базами данных) - это комплекс программно-языковых средств, позволяющих создать базы данных и управлять данными. Иными словами, СУБД - это набор программ, позволяющий организовывать, контролировать и администрировать базы данных [14]. Соответственно под определенный вид БД нужна и своя СУБД. И так можно приступать к разновидностям БД, я дам краткое описание каждого из них: • Иерархическая - первая в списке, БД основанная на древовидной структуре, состоящей из объектов, представленных в виде данных, различных уровней. Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие

объекты находятся в отношении предка (объект более близкий к корню) к потомку (объект более низкого уровня), при этом возможна ситуация, когда объект-предок имеет несколько потомков, тогда как у объекта-потомка обязателен только один предок. Объекты, имеющие общего предка, называются близнецами. Иерархическая модель одна из самых старых на данный момент [2]. • Объектно-ориентированная (ООБД) - вторая в списке, БД в которой данные моделируются в виде объектов, их атрибутов, методов и классов. ООБД обычно рекомендованы для тех случаев, когда требуется высокопроизводительная обработка данных, имеющих сложную структуру [5]. • Реляционная (РМД) - следующая в списке, БД построенная на реляционной модели. Реляционная модель данных включает в себя: о Структурный аспект - данные в БД представляют собой набор отношений. о Аспект целостности - отношения отвечают определенным условиям целостности. о Аспект обработки (манипулирования) - РДМ поддерживает операторы манипулирования отношениями. Кроме того, в состав реляционной модели данных включают теорию нормализации. Та включает в себя несколько нормальных норм: Нулевая н.ф. (UNF), Первая н.ф. (1NF), Вторая н.ф. (2NF), Третья н.ф (3NF), н.ф. Бойса-Кодда (BCNF), Четвертая н.ф. (4NF), Пятая н.ф. (5NF), Доменно-ключевая н.ф. (DKNF), Шестая н.ф. (6NF) [3]. Определенной к одной из форм характеризуется параметрами каждой уникальной БД. Одна база может принадлежать только к одной из форм [7]. • Сетевая - далее в списке, является расширением иерархического подхода. Разница между иерархической моделью данных и сетевой состоит в том, что в иерархических структурах запись-потомок должна иметь в точности одного предка, а в сетевой структуре данных у потомка может иметься любое число предков [8]. • Функциональная - последняя в списке, БД используются для решения аналитических задач, таких как финансовое моделирование и управление производительностью. Функциональная база данных или коротко функциональная модель отличается от реляционной модели [11]. При разработке БД, выбирается та модель данных, которая больше всего подходит под поставленную задачу. На сегодняшний день преимущества реляционного подхода сделали его самой распространенной моделью для баз данных в мире [13]. Так же существует понятие "Сверхбольшая база данных" - это база данных, которая занимает чрезвычайно большой объем на устройстве физического хранения. Термин подразумевает максимально возможные объемы БД, которые определяются последними достижениями в технологиях физического хранения данных и в технологиях программного оперирования данными [1]. Этих знаний достаточно для понимания основ баз данных. Взаимодействие с базами данных Как я уже упоминал выше, взаимодействие с самой БД происходит посредством определенной СУБД. Если использовать РМД, то стоит учесть, что все взаимодействия ведутся на языке SQL. SQL SQL (Structured Query Language, или язык структурированных запросов) - это декларативный язык программирования (язык запросов), который используют для создания, обработки и хранения данных в реляционных БД [19]. Теперь перейдем к приложениям, в которых необходимо писать запросы на языке SQL. Различные источники утверждают, что топ самых популярных систем управления БД возглавляют Oracle, MySQL, Microsoft SQL Server, PostgreSQL. Именно эти ПО я и буду рассматривать для работы. Анализ каждой программы я буду проводить исходя из плюсов и минусов каждой из представленных выше ПО. Oracle Первым на очереди предстоит разобрать такой инструмент для работы с БД как "Oracle Database" [6]. Плюсы: 1. Высокая производительность - Oracle Database обеспечивает высокую скорость работы и способна обрабатывать большие базы данных; 2. Поддержка нескольких баз данных - Одно из лучших преимуществ Oracle Database - возможность управления несколькими базами данных в рамках одной транзакции; 3. Обновление версий - Oracle своевременно информирует вас о предстоящем большом релизе и потенциальных изменениях, чтобы вы могли подготовиться. Oracle предлагает хорошую обратную совместимость, так что вам не придется переписывать приложение при обновлении СУБД. Минусы: 1. Сложность - Одним из крупных недостатков Oracle Database является ее сложность. Oracle не рекомендуется для использования без достаточных технических знаний и навыков работы с Oracle Database. Oracle также не подойдет для тех, кто ищет простую в использовании и основных функциях базу данных; 2. Oracle является платным ПО, по сравнению с конкурентами, его цена выше средней по рынку. К тому же, стоимость лицензии Oracle может изменяться со временем в зависимости от различных факторов, таких как изменения в политике компании, обновления; 3. Сложность управления - Oracle обычно требует больше усилий для управления некоторыми операциями. Oracle Database оправдана только при работе с большими базами данных. MySQL Следующим вариантом является "MySQL Workbench", разрабатываемый компанией Oracle, имеющий две версии. Первая "MySQL Enterprise Edition", является платной вариацией и распространена меньше чем вторая вариация под названием " MySQL Community Edition", будучи бесплатной версией с открытым исходным кодом, является самой популярной СУБД в мире [17]. Плюсы: 1. Безопасность - MySQL имеет в своем функционале различные функции безопасности, включая возможность установки привилегий пользователя, шифрование данных и аутентификацию, чтобы обезопасить информацию, хранящуюся в базах данных; 2. Производительность - MySQL обладает хорошей производительностью и быстродействием благодаря оптимизированным алгоритмам выполнения запросов. Умеет работать с большими объемами данных с минимальными задержками; 3. Масштабируемость - MySQL может использоваться как для небольших веб-приложений, так и для серьезных корпоративных систем. СУБД предлагает различные методы масштабирования, включая горизонтальное и вертикальное масштабирование, что дает возможность расширять базы данных при возникновении необходимости. Это позволяет обрабатывать большое количество одновременных запросов и поддерживать беспрепятственный доступ к данным для пользователей; 4. Доступность - рассматриваемый продукт имеет открытый исходный код и может быть использован бесплатно. Это делает его доступным для большого круга разработчиков и пользователей; 5. Развитое сообщество - MySQL имеет большое сообщество разработчиков и пользователей, что дает возможность доступа к различным ресурсам и помощи при возникновении проблем. Минусы 1. Отказоустойчивость - MySQL может иметь проблемы с производительностью при обработке больших объемов данных или, когда требуется обработка сложных запросов или, подключение большого количества клиентов. Это может привести к замедлению работы или даже отказу в обслуживании; 2. Масштабируемость - MySQL может столкнутся с ограничением масштабируемости при работе с большим количеством клиентов. Тогда отказоустойчивость и возможность горизонтального масштабирования могут быть ограничены; 3. Сложность - MySQL может быть сложным для администрирования, особенно для новичков. Настройка и

оптимизация параметров конфигурации может быть сложной задачей, требующей глубоких знаний и опыта; 4. Отсутствие обновлений - MySQL достаточно редко обновляется, привнося лишь небольшие изменения. Глобальных изменений не было с выхода последней версии в 2016 году [16]. Ситуацию спасает широко развитое сообщество и прочие плюсы данного ПО. Microsoft SQL Server Следующим продуктом на досмотр выходит программа "SQL Server", разрабатываемая компанией Microsoft [15]. Плюсы: 1. Масштабирование - взаимодействовать с ней можно на портативных ПК или мощной мультипроцессорной технике. Процессор может одновременно обрабатывать большой объем запросов; 2. Размер страниц - до 8 кб, поэтому данные извлекаются быстро, подробную и сложную информацию хранить удобнее. Возможно сложно понять в чём тут плюс, но в больших компаниях постоянно следят за цифровым весом итогового продукта и SQL Server даёт возможность экономить пространство на цифровых носителях; 3. Автоматизированное администрирование - такие процессы как управление блокировками, памятью, редактура размеров файлов. У системы продуманы настройки, можно создать профили пользователей. Это облегчает освоение для новичков; 4. Поддержка Microsoft - Поддержка работы с другими решениями Майкрософт. Минусы: 1. Кроссплатформенность - программу можно установить только на операционные системы от Microsoft, т.е. Windows; 2. Доступность - рассматриваемый продукт, является платным решением, заставляя выбрать другое ПО свободным разработчикам и маленьким компаниям. PostgreSQL Последним пунктом в списке остался запоминающийся Слоник, программа PostgreSQL, основанная Майклом Стоунбрейкером, разработка над которой ведётся командой PostgreSQL [18]. Плюсы: 1. Объектно-реляционная модель - PostgreSQL - объектно-реляционная СУБД. Это значит, что она поддерживает и объектный, и реляционный подход. Это делает данное ПО более гибким для разработки по сравнению с остальными; 2. Поддержка множества типов данных - Еще одна особенность PostgreSQL - поддержка большого количества типов записи информации. Это не только стандартные целочисленные значения, числа с плавающей точкой, строки и булевы значения ("да/нет"), но и денежный, геометрический, перечисляемый, бинарный и другие типы; 3. Работа с большими объемами - В большинстве СУБД, рассчитанных на средние и небольшие проекты, есть ограничения по объему базы и количеству записей в ней. В PostgreSQL ограничений нет; 4. Поддержка сложных запросов - PostgreSQL работает со сложными, составными запросами. Система справляется с задачами разбора и выполнения трудоемких операций, которые подразумевают и чтение, и запись, и валидацию одновременно. Она медленнее аналогов, если речь заходит только о чтении, но в других аспектах превосходит конкурентов; 5. Написание функций на нескольких языках - В PostgreSQL можно писать собственные функции - пользовательские блоки кода, которые выполняют те или иные действия. Эта возможность есть практически в любых СУБД, но PostgreSQL поддерживает больше языков, чем аналоги. Кроме стандартного SQL, в PostgreSQL можно писать на C и C++, Java, Python, PHP, Lua и Ruby; 6. Одновременная модификация базы - Важная особенность PostgreSQL - возможность одновременного доступа к базе с нескольких устройств. В СУБД реализована клиент-серверная архитектура, когда база данных хранится на сервере, а доступ к ней осуществляется с клиентских компьютеров; 7. Доступность - PostgreSQL - ПО с открытым исходным кодом, которое распространяется по свободной лицензии. Это означает, что любой разработчик может посмотреть, как написана система, или предложить для нее свои правки. СУБД разрабатывается сообществом энтузиастов и в определенной степени никому не принадлежит, а значит, ее можно свободно и без ограничений использовать в своих проектах; 8. Кроссплатформенность - Чаще всего PostgreSQL используют на серверах с операционными системами семейства Linux, но СУБД поддерживает и другие ОС. Ее можно установить в системы на базе Windows, BSD, macOS и Solaris. Минусы: 1. Сложности при настройке - У PostgreSQL очень обширный набор возможностей. Очевидно, что такое разнообразие функций влечёт за собой множество настроек, что может вызвать затруднения у новичков. Корректная настройка базы данных требует глубокого понимания архитектуры и параметров; 2. Повышенное потребление ресурсов - В сравнении с некоторыми другими СУБД, PostgreSQL может потреблять больше ресурсов (включая оперативную память и процессорное время). Это особенно заметно при работе с большими объёмами данных и выполнении сложных запросов; 3. Отсутствие некоторых функций - По сравнению с определенными коммерческими аналогами PostgreSQL может немного уступать в функциональности. Стоит отметить, что перечисленные недостатки в основном применимы к конкретным сценариям использования. В общем плане PostgreSQL остаётся одной из самых мощных и популярных открытых СУБД. На этом описание базовых понятий, принципов и аспектов работы с БД, а также различных возможностей разработки БД я считаю завершённым. Далее следует разобраться с понятием "Видеохостинг". Введение в видеохостинги Видеохостинг (от лат. video и англ. hosting) - веб-сервис, позволяющий загружать и просматривать видео в браузере, например, через специальный проигрыватель. При этом большинство подобных сервисов не предоставляют видео, следуя таким образом принципу "контент генерирует пользователь" ("Usergenerated content"). Самым лучшим и популярным примером видеохостинга, является "YouTube" - веб-сервис, разрабатываемый компанией Google. Изначально YT разрабатывался командой в которой состояли: Стив Чен, Чад Хёрли, Джавед Карим. Я предполагаю, что YouTube стал первым видеохостингом в мире, хотя я не нашёл информации опровергающей это высказывание, я так же не нашёл информации подтверждающей его. Будем считать, что YT был первым, а если это не является правдой, то он точно являлся толчком в мире веб-сервисов для загрузки и просмотра видео контента. Основан самый популярный видеохостинг в 2005 году, и с тех пор подобных веб приложений стало огромное множество. В топ самых популярных веб-сервисов входят следующие решения: YouTube, Twitch, Vimeo, Dailymotion, Google Drive. У каждого из перечисленных приложений своя политика и принципы, но суть одинакова: пользователь регистрируется, тем самым создав свой канал, выгружает на него собственные или не очень видео, те в свою очередь загружаются в БД, после чего к просмотру видео открывается доступ и другим пользователям. Не стоит забывать и про взаимодействие пользователей между собой, проявляющееся в виде комментирования видео, возможности ставить положительные или отрицательные оценки, с английского их называют "Like" и "Dislike", а также возможность подписки на