Проект

**Дзестелова Хетага Артуровича**

по теме

**Контентная платформа**

# Описание проекта

Основными сущностями в проекте выступают пользователь (User), у которого зафиксируем стандартную информацию для авторизации и публикация (Publication), при этом на платформе реализовано два типа публикаций - статьи (Article) и видео (Videos).

Пользователь может оставлять комментарии, выступать в качестве владельца канала (Channel), подписываться на канал (Subscription), оставлять в закладках публикации (Bookmark), при этом закладка может быть как просмотрена так и нет. У канала и публикации существуют категории (Category). При этом у канала может существовать несколько публикаций.

На публикацию пользователь получает позитив (взаимодействие с публикацией, например, лайк, дизлайк, полное прочитывание), для этого введем тип позитива (для начала ограничимся like, dislike, view).

Будем считать что наша база данных будет использоваться в рекомендательном сервисе, который рекомендует новые публикации на основе позитивов пользователя (т. е. публикация может быть показана только по какому-то позитиву; считаем что изначально даже у новых пользователей есть какие-то заранее зафиксированные позитивы). В перспективе возможно использовать модели машинного обучения и эмбеддинг публикаций - но вероятно целесообразно не хранить всю эту информацию в базе данных, а использовать другие механизмы.

# Построение отношений

В результате предварительного проектирования были выделены следующие отношения:

* Users - пользователь в системе.
* Publications - обобщенный тип публикации, который является либо публикацией типа “Видео”, либо публикацией типа “Статья”.
* Channels - канал с публикациями.
  + Считаем что у канала должен быть как один владелец и/или несколько менеджеров (в первом приближении считаем менеджера и владельца одной ролью).
* ChannelManagers - менеджерские права на канал.
* Positives - взаимодействие пользователя с публикацией.
  + Несмотря на название “Positive” взаимодействие не обязательно должно быть “хорошим”, учитываем факт любого взаимодействия.
  + Для начала положим три типа - [‘like’, ‘dislike’, ‘view’], где ‘view’ - глубокий просмотр (например, можно считать что пользователь кликнул на рекомендацию и начал её “просмотр”).
* Comments - комментарий от пользователя к публикации.
* Categories - тематическая категория (например, “Кино” или “Спорт”).
* ChannelCategories - категория канала.
* PublicationCategories - категория публикации.
* Subscriptions - подписка пользователя на канал.
* Bookmarks - закладка пользователя на публикацию.

## Отношение Users

Атрибуты:

* UserId - суррогатный идентификатор
* UserName - имя пользователя в форме ФИ
* PasswordHash - засоленный пароль, используется для авторизации
* PhoneNumber - номер телефона, используется для авторизации
* RegistrationDate - дата и время регистрации

Функциональные зависимости:

* UserId → UserName, PasswordHash, PhoneNumber, RegistrationDate
* PhoneNumber → UserId

Ключи:

* UserId
* PhoneNumber

Указанное отношение находится в 1НФ (есть ключ, считаем все атрибуты атомарными, нет повторяющихся групп атрибутов), 2НФ (нет составных ключей), 3НФ, т. к. все неключевые атрибуты непосредственно зависят от ключа**.** И т. к. все ключи простые, то по теореме Дейта-Фейгина 1 отношение находится в 5НФ.

## Отношение Publications

Для того чтобы реализовать несколько типов публикаций “Video” и “Article” будем использовать широкую таблицу. В данном случае откажемся от нормализации и представлении наследования через таблицы классов для производительности загрузки одновременно всех типов публикаци. Отмечу, что кол-во новых возможных типов публикаций весьма ограничено и добавляться они будут крайне редко. В связи с этим считаю в данном случае решение оправданным.

Атрибуты:

* PublicationId - суррогатный идентификаторPp
* Titile - заголовок публикации
* PublicationDate - дата и время публикации
* ChannelId - канал, сделавший публикацию
* Description - описание публикации типа видео
* HostingUrl - ссылка на стриминг публикации типа видео
* Content - текст публикации типа статья

Функциональные зависимости:

* PublicationId -> Titile, PublicationDate, ChannelId

Ключи:

* PublicationId

## Отношение Channels

Атрибуты:

* ChannelId - суррогатный идентификатор
* ChannelName - имя канала
* RegistrationDate - дата и время открытия канала
* OwnerId - обязательный владелец

Функциональные зависимости:

* ChannelId -> ChannelName, RegistrationDate, OwnerId

Ключи:

* ChannelId

## Отношение ChannelManagers

Атрибуты:

* ChannelId - идентификатор канала
* UserId - идентификатор пользователя

Функциональные зависимости:

* -

Ключи:

* (ChannelId, UserId)

## Отношение Positives

Атрибуты:

* PositiveId - суррогатный идентификатор
* PositiveType - тип взаимодействия
* UserId - пользователь, который осуществил взаимодействие
* PublicatioId - публикация, с которой взаимодействовал пользователь
* Date - дата и время взаимодействия

Функциональные зависимости:

* (PositiveId, PositiveType) → Date
* PositiveId → UserId, PublicationId

Ключи:

* (PositiveId, PositiveType)

Отношение находится в 1НФ (есть ключ, атрибуты атомарные, нет повторяющихся групп атрибутов). Но отношение нарушает 2НФ, т. к. существуют неключевые атрибуты, которые функционально зависят от части ключа (UserId). **Декомпозируем по PositiveId → UserId, PublicationId и получим два отношения:**

## Отношение Positives

Атрибуты:

* PositiveId - суррогатный идентификатор
* PositiveType - тип взаимодействия
* UserId - пользователь, который осуществил взаимодействие
* PublicatioId - публикация, с которой взаимодействовал пользователь
* Date - дата и время взаимодействия

Функциональные зависимости:

* PositiveId → PositiveType, UserId, PublicatioId, Date

Ключи:

* PositiveId

После декомпозиции получаем 2НФ. Указанное отношение находится в 3НФ, т. к. все неключевые атрибуты непосредственно зависят от ключа**.** Находится в НФБК (в ФЗ PositiveId → PositiveType, UserId, PublicatioId, Date PositiveId является ключем → является надключем)Т. к. есть простой ключ и НФБК, то по теореме Дейте-Фейгина 2 отношение находится в 4НФ. И т. к. все ключи простые и 3НФ, то по теореме Дейта-Фейгина 1 отношение находится в 5НФ.

## Отношение PositiveTypes

Атрибуты:

* PositiveId - идентификатор взаимодействия
* PositiveType - тип взаимодействия

Функциональные зависимости:

* -

Ключи:

* (PositiveId, PositiveType)

## Отношение Comments

Атрибуты:

* CommentId - идентификатор канала
* Content - текст комментария
* UserId - пользователь, который оставил комментарий
* PublicationId - публикация, к которой был оставлен комментарий
* Date - дата публикации

Функциональные зависимости:

* CommentId → Content, UserId, PublicationId, Date

Ключи:

* CommentId

## Отношение Categories

Атрибуты:

* CategoryId - суррогатный идентификатор
* CategoryName - наименование категории (например, “Кино”)

Функциональные зависимости:

* CategoryId -> CategoryName

Ключи:

* CategoryId

## Отношение ChannelCategories

Атрибуты:

* CategoryId - идентификатор категории
* ChannelId - идентификатор канала

Функциональные зависимости:

* -

Ключи:

* (CategoryId, ChannelId)

## Отношение PublicationCategories

Атрибуты:

* CategoryId - идентификатор категории
* PublicationId - идентификатор публикации

Функциональные зависимости:

* -

Ключи:

* (CategoryId, PublicationId)

## Отношение Subscriptions

Атрибуты:

* ChannelId - канал, на который пользователь подписался
* UserId - пользователь, осуществивший подписку
* Date - дата и время подписки

Функциональные зависимости:

* ChannelId, UserId → Date

Ключи:

* (ChannelId, UserId)

## Отношение Bookmarks

Атрибуты:

* PublicationId - публикация, которую пользователь сохранил в закладку
* UserId - пользователь, сохранивший закладку
* Date - дата и время подписки
* Seen - отметил ли пользователь закладку просмотренной

Функциональные зависимости:

* PublicationId, UserId → Date, Seen

Ключи:

* (PublicationId, UserId)

Указанное отношение находится в 1НФ (есть ключ, все атрибуты атомарные, нет повторяющихся групп атрибутов), 2НФ (нет зависимости неключевых атрибутов от части ключа), 3НФ (все неключевые атрибуты непосредственно зависят от ключа), НФБК (в единственной ФЗ PublicationId, UserId является ключем → является надключем). Все атрибуты зависят от ключ, одна ФЗ (МЗ) → 4НФ, 5НФ.

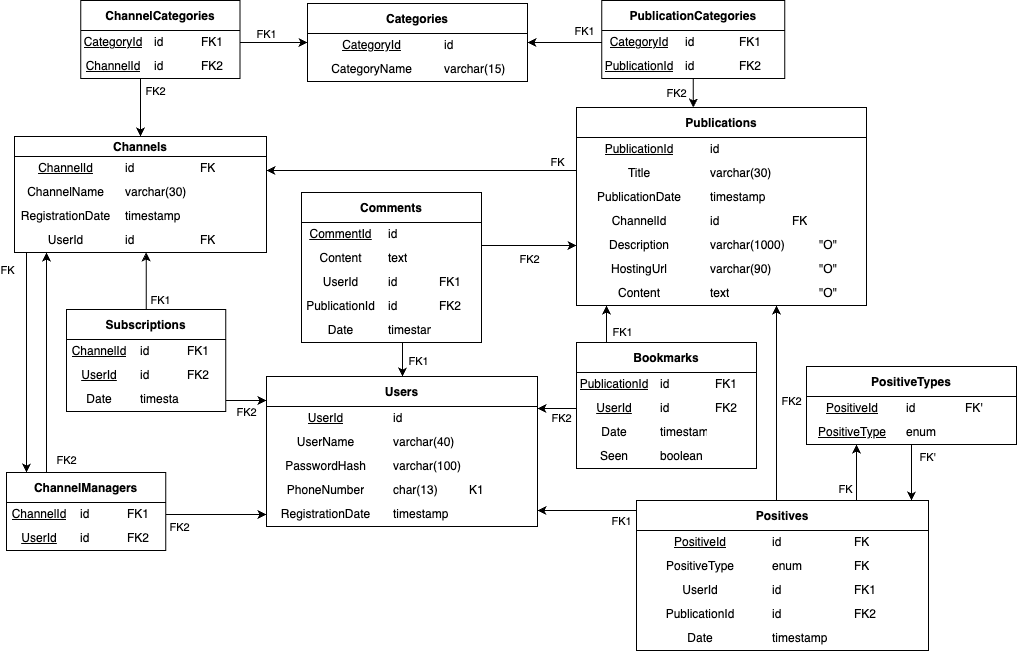
# 

# Считаем по-умолчанию все атрибуты **обязательными** в обеих моделя. В модели сущность-связь во всех ассоциациях и их связях используется **Chen-like** интерпретация. Для **наглядности** в модели сущность-связь выделил Video и Article в слабые сущности. Замечу, что в физической модели принято решение об использовании **широкой таблицы** для Publication. (Для справки: корректнее было бы отразить это решение в модели сущность-связь в виде добавления соответствующих атрибутов в сущность Publication без слабых сущностей)

# Модель сущность-связь ([ссылка](https://drive.google.com/drive/folders/1eUrxtCkc-MnnFBWyhsyA6B2D4mufombU))

# 

# Физическая модель ([ссылка](https://drive.google.com/drive/folders/1eUrxtCkc-MnnFBWyhsyA6B2D4mufombU))



При построении физической модели использовалось следующее отображение доменов в типы:

| Домен | Тип |
| --- | --- |
| id | id |
| channelName | varchar(30) |
| userName | varchar(40) |
| passwordHash | varchar(100) |
| phone | char(13) |
| categoryName | varchar(15) |
| descriptionText | varchar(1000) |
| url | varchar(90) |
| positiveType | ENUM ('view', 'dislike', 'like')\* |
| titleText | varchar(30) |
| text | text |
| date | timestamp |

\* - PostgeSQL 8.7. Enumerated Types

# Определения схемы

Для реализации проекта использовалась СУБД PostgreSQL (ver. 16.1). Определения таблиц и их индексов приведено в файле ddl.sql (Приложение 1).

# Тестовые данные

Скрипт для добавления тестовых данных приведен в файле insert.sql (Приложение 2).

# 

# Запросы на получение данных

В рамках проекта были реализованы следующие запросы:

* RegisterUser(user\_name, pasword, phone\_number, registration\_date) -- Зарегистрировать пользователя и засолить пароль
* AuthUser(phone\_number, password) -- Аутентифицировать пользователя по телефону и паролю.
* GetChannelsIds(channel\_name) -- Узнать каналы с указанным названием.
* GetChannelManagers(channel\_id) -- Получить всех менеджеров канала.
* GetManagedChannels(user\_id) -- Узнать, какими каналами владеет пользователь.
* GetChannelCategories(channel\_id) -- Получить категории канала.
* GetAllChannelPublicationsCategories(channel\_id) -- Получить категории всех публикаций канала.
* GetUserSubscriptions(user\_id) -- Получить подписки определенного пользователя.
* GetSubscriptionsCount(channel\_id) -- Получить количество подписчиков канала.
* GetUnsubscribedUsersComments(publication\_id) -- Получить все комментарии к публикации от пользователей, которые не подписаны на канал.
* GetSubscribedUsersWithBookmarkComments() -- Получить всех пользователей, которые подписаны.
* GetMostPositivityUsers(channel\_id) -- Получить топ-3 пользователей с наибольшим количеством позитивов для указанного канала.

Запросы на получение данных и вспомогательные представления приведены в файле select.sql (Приложение 3).

# Запросы на изменение данных

В рамках проекта были реализованы следующие запросы:

* DeleteCategoryById(category\_id) -- удаляет категорию и отвязывает её от всех связанных с ней каналов и публикаций
* UpdatePublicationDescriptionById(publication\_id, new\_description) -- обновляет описание публикации
* AddPositiveToPublication(user\_id, publication\_id, positive\_type) -- добавляет позитив по публикации, юзеру и типу

Запросы на изменение данных, хранимые процедуры и триггеры приведены в файле update.sql (Приложение 4).