НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих комп’ютерних систем**

**Лабораторна робота №1**

з дисципліни **Бази даних та засоби управління**

*на тему: “Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL“*

Виконав:

студент III курсу

групи КВ-02

Вознюк Б. Г.

Перевірив:

Київ - 2022

*Метою роботи* є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

*Завдання* роботи полягає у наступному:

1. Розробити модель «сутність-зв’язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.
3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).
4. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4 та внести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

**Опис предметної області “Відеострімінговий сервіс”**

Обрана предметна область передбачає збереження інформації щодо проведення прямих трансляцій ведучими, а також глядачів, що відслідковують та переглядають даний відеоконтент

**Опис сутностей предметної області**

Для побудови бази даних для предметної області було виділені сутності, зображені на рисунку 1:

1. Трансляція (Stream), з атрибутами код трансляції, назва, кількість глядачів, час та дата проведення. Призначена для відстеження прямих трансляцій, що знаходяться в етері на даний момент.

2. Ведучий трансляції (Streamer), з атрибутами код ведучого, ім’я користувача, дата реєстрації, кількість людей, підписаних на контент а також інформацію, чи є в етері прямо зараз. Призначена для збереження інформації щодо ведучих трансляцій, зареєстрованих на сайті.

3. Глядач (Viewer), з атрибутами код глядача, ім’я користувача, дата реєстрації та кількість придбаних платних підписок на відеоконтент. Призначена для збереження інформації щодо зареєстрованих користувачів - глядачів, та придбаних ними підписок.

**Опис зв’язків між сутностями предметної області**

Сутність “Трансляція” має зв’язок 1:M по відношенню до сутності “Ведучий трансляції”, тому що одна трансляція може бути проведена різними ведучими, але один ведучий, в той самий час, не може вести більш ніж одну пряму трансляцію. Також ця сутність має звязок M:N по відношенню до сутності “Глядач”, тому що одну трансляцію може переглядати безліч глядачів, а один глядач, в свою чергу, може переглядати безліч кількість трансляцій

Сутність “Ведучий трансляції” має звязок M:N по відношенню до сутності “Глядач”, тому що кожен ведучий може мати безліч глядачів, підписаних на контент. Водночас кожен глядач може бути підписаним на контент будь-якої кількості ведучих

**Концептуальна модель області “Відеострімінговий сервіс”**



Рисунок 1 - ER - діаграма, побудована за нотацією Чена (інструмент: draw.io)

**Перетворення концептуальної моделі у логічну схему бази даних**

Cутність “Трансляція” перетворено на таблицю “stream”

Cутність “Ведучий трансляції” перетворено на таблицю “streamer”

Cутність “Глядач” перетворено на таблицю “viewer”

Зв’язок M;N між сутностями “Трансляція” та “Глядач” зумовив появу в ER моделі сутності “Трансляція/глядач”, яку перетворено в таблицю “stream\_viewer” із зовнішніми ключами stream\_id та viewer\_id

Зв’язок M;N між сутностями “Ведучий трансляції” та “Глядач” зумовив появу в ER моделі сутності “Підписка”, яку перетворено в таблицю “subscribtion” із зовнішніми ключами streamer\_id та viewer\_id

**Логічна модель (схема) БД “Відеострімінговий сервіс”**

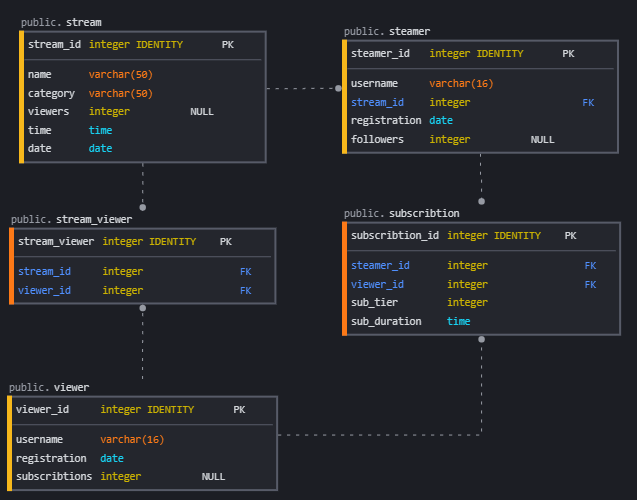


Рисунок 2 - Схема бази даних (інструмент: sqldbm.com)

**Опис об’єктів бази даних у вигляді таблиці**

| Сутність | Атрибут | Тип атрибуту |
| --- | --- | --- |
| **stream** *- містить всі трансляції, що є онлайн в даний момент* | **stream\_id** *- унікальний ідентифікатор трансляції*  **name -** *назва трансляції. Не допускає NULL*  **category -** *назва категорії трансляції. Не допускає NULL*  **viewers -** *кількість глядачів, що дивляться трансляцію*  **time -** *час трансляції. Не допускає NULL*  **date -** *дата проведення трансляції.Не допускає NULL* | **integer** *(числовий)*  **character varying** *(рядок)*  **character varying** *(рядок)*  **integer** *(числовий)*  **time** *(часовий)*  **date** *(дата)* |
| **streamer** *- містить інформацію про всі ведучих, зареєстрованих на відеострімінговому сервісі* | **streamer\_id** *- унікальний ідентифікатор ведучого*  **username -** *унікальне ім’я користувача. Не допускає NULL*  **stream\_id** *- унікальний ідентифікатор трансляції*  **registration -** *даиа реєстрації користувача. Не допускає NULL*  **followers -** *кількість глядачів, що слідкують за контентом* | **integer** *(числовий)*  **character varying** *(рядок)*  **integer** *(числовий*  **date** *(дата)*  **integer** *(числовий)* |
| **viewer** *- містить інформацію про зареєстрованих глядачів* | **viewer\_id** *- унікальний ідентифікатор глядача*  **username -** *унікальне ім’я користувача. Не допускає NULL*  **registration -** *даиа реєстрації користувача. Не допускає NULL*  **subscribtions -** *кількість придбаних платних підписок* | **integer** *(числовий)*  **character varying** *(рядок)*  **date** *(дата)*  **integer** *(числовий)* |
| **stream\_viewer -** *містить інформацію про перебування кожного глядача на кожній з трансляцій, що є онлайн на даний момент* | **stream\_viewer\_id** *- унікальний ідентифікатор комбінації “трансляція - глядач”*  **stream\_id** *- унікальний ідентифікатор трансляції*  **viewer\_id** *- унікальний ідентифікатор глядача* | **integer** *(числовий)*  **integer** *(числовий)*  **integer** *(числовий)* |
| **subscribtion** *- містить усі платні підписки, придбані кожним з глядачів на кожного з ведучих* | **subscribtion\_id** *- унікальний ідентифікатор підписки*  **streamer\_id** *- унікальний ідентифікатор ведучого*  **viewer\_id** *- унікальний ідентифікатор глядача*  **sub\_tier***- рівень підписки. Не допускає NULL*  **sub\_duration** *- тривалість платної підписки* | **integer** *(числовий)*  **integer** *(числовий)*  **integer** *(числовий)*  **integer** *(числовий)*  **time** *(часовий)* |

**Функціональні залежності для кожної таблиці**

STREAM:

stream\_id → name, category, viewers, time, date

stream\_id → name (назва трансляції залежить від її коду)

stream\_id → category (категорія трансляції залежить від її коду)

stream\_id → viewers (кількість глядачів на трансляції залежить від її коду)

stream\_id → time (час трансляції залежить від її коду)

stream\_id → date (дата проведення трансляції залежить від її коду)

STREAMER:

streamer\_id → username, stream\_id, registration, followers

streamer\_id → username (ім’я користувача залежить від коду ведучого)

streamer\_id → stream\_id (код трансляції залежить від коду ведучого)

streamer\_id → registration (дата реєстрації залежить від коду ведучого)

streamer\_id → followers (кількість глядачів, підписаних на контент залежить від коду ведучого)

VIEWER:

viewer\_id → username, registration, subscribtions

viewer\_id → username (ім’я користувача залежить від коду глядача)

viewer\_id → registration (дата реєстрації залежить від коду глядача)

viewer\_id → subscribtions (кількість придбаних глядачів залежить від коду глядача)

STREAM\_VIEWER:

stream\_viewer\_id → stream\_id, viewer\_id

stream\_viewer\_id → stream\_id (код трансляції залежить від коду комбінації “трансляція - глядач”)

stream\_viewer\_id → viewer\_id (код глядача залежить від коду комбінації “трансляція - глядач”)

SUBSCRTIBTION:

subscribtion\_id → streamer\_id, viewer\_id, sub\_name, sub\_duration

subscribtion\_id → streamer\_id (код ведучого залежить від коду платної підписки)

subscribtion\_id → viewer\_id (код глядача залежить від коду платної підписки)

subscribtion\_id → sub\_tier (рівень підписки залежить від коду платної підписки)

subscribtion\_id → sub\_duration (тривалість підписки залежить від коду платної підписки)

**Відповідність схеми БД до третьої нормальної форми**

Схема відповідає 1НФ, тому що:

1. Значення в кожній комірці таблиці має бути атомарним

2. Кожен рядок є унікальним, і не повторюється

Схема відповідає 2НФ, тому що:

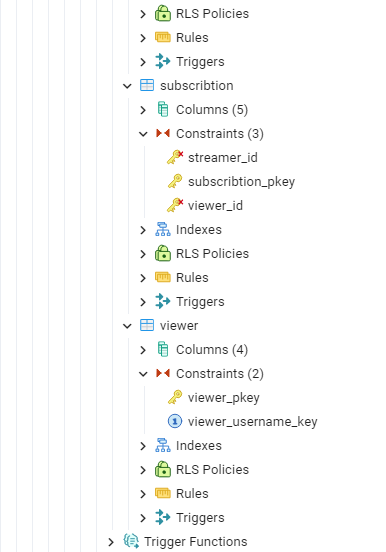
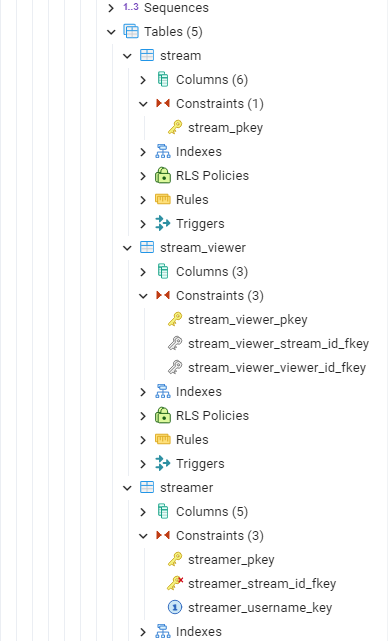
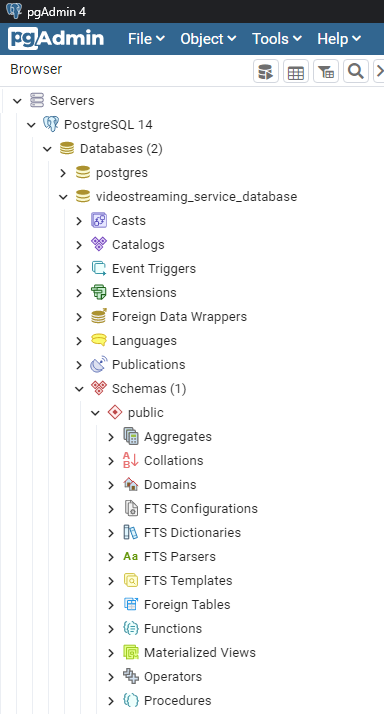
1. Задовольняє умови 1НФ

2. Для кожного неключового атрибута має існувати функціональна залежність від цілого ключа, але не від його частини

Схема відповідає 3НФ, тому що:

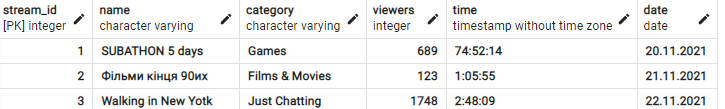
1. Задовольняє умови 2НФ

2. Кожен неключовий атрибут відношення або таблиці не є транзитивно залежним від кожного ключа

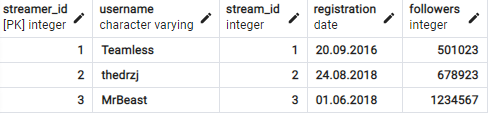
**Фізична модель БД “Відеострімінговий сервіс” у pgAdmin4**

**Фотографії вмісту таблиць**

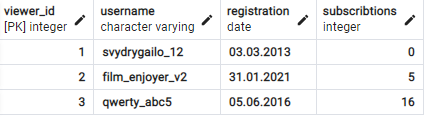
**stream**

****

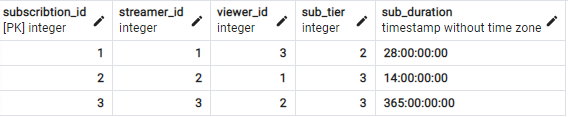
**streamer**

****

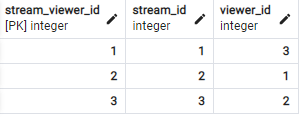
**viewer**

****

**subscribtion**

****

**stream\_viewer**

****

**SQL-текст опису БД “Відеострімінговий сервіс”**

CREATE DATABASE videostreaming\_service\_database

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'Ukrainian\_Ukraine.1251'

LC\_CTYPE = 'Ukrainian\_Ukraine.1251'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;

CREATE SCHEMA IF NOT EXISTS public

AUTHORIZATION postgres;

COMMENT ON SCHEMA public

IS 'standard public schema';

GRANT ALL ON SCHEMA public TO PUBLIC;

GRANT ALL ON SCHEMA public TO postgres;

CREATE TABLE IF NOT EXISTS public.stream

(

stream\_id integer NOT NULL,

name character varying COLLATE pg\_catalog."default" NOT NULL,

category character varying COLLATE pg\_catalog."default" NOT NULL,

viewers integer,

"time" timestamp without time zone NOT NULL,

date date NOT NULL,

CONSTRAINT stream\_pkey PRIMARY KEY (stream\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.stream

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public.stream\_viewer

(

stream\_viewer\_id integer NOT NULL,

stream\_id integer NOT NULL,

viewer\_id integer NOT NULL,

CONSTRAINT stream\_viewer\_pkey PRIMARY KEY (stream\_viewer\_id),

CONSTRAINT stream\_viewer\_stream\_id\_fkey FOREIGN KEY (stream\_id)

REFERENCES public.stream (stream\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT stream\_viewer\_viewer\_id\_fkey FOREIGN KEY (viewer\_id)

REFERENCES public.viewer (viewer\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.stream\_viewer

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public.streamer

(

streamer\_id integer NOT NULL,

username character varying COLLATE pg\_catalog."default" NOT NULL,

stream\_id integer NOT NULL,

registration date NOT NULL,

followers integer,

CONSTRAINT streamer\_pkey PRIMARY KEY (streamer\_id),

CONSTRAINT streamer\_username\_key UNIQUE (username),

CONSTRAINT streamer\_stream\_id\_fkey FOREIGN KEY (stream\_id)

REFERENCES public.stream (stream\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.streamer

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public.subscribtion

(

subscribtion\_id integer NOT NULL,

streamer\_id integer NOT NULL,

viewer\_id integer NOT NULL,

sub\_tier integer NOT NULL,

sub\_duration timestamp without time zone NOT NULL,

CONSTRAINT subscribtion\_pkey PRIMARY KEY (subscribtion\_id),

CONSTRAINT streamer\_id FOREIGN KEY (streamer\_id)

REFERENCES public.streamer (streamer\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID,

CONSTRAINT viewer\_id FOREIGN KEY (viewer\_id)

REFERENCES public.viewer (viewer\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.subscribtion

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public.viewer

(

viewer\_id integer NOT NULL,

username character varying COLLATE pg\_catalog."default" NOT NULL,

registration date NOT NULL,

subscribtions integer,

CONSTRAINT viewer\_pkey PRIMARY KEY (viewer\_id),

CONSTRAINT viewer\_username\_key UNIQUE (username)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.viewer

OWNER to postgres;