Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ № 10

Проектирование базы данных и ее реализация в среде СУБД PostgreSQL
Тема

Преподаватель _____ Е. П. Моргунов
Подпись, дата Инициалы, Фамилия

Студент КИ19-17/1Б, №031939174 _____ А. К. Никитин
Номер группы, зачетной книжки Подпись, дата Инициалы, Фамилия

Красноярск 2021

1 Краткое описание предметной области

Аниме - японская мультипликация. База данных включает себя таблицы для хранения информации о самих аниме, студиях и режиссерах, снявших их, и пользователе, который хранит у себя список аниме, информацию о которых он может посмотреть.

2 Концептуальное проектирование БД

На рисунке 1 представлена ER-диаграмма концептуальной стадии проектирования БД.

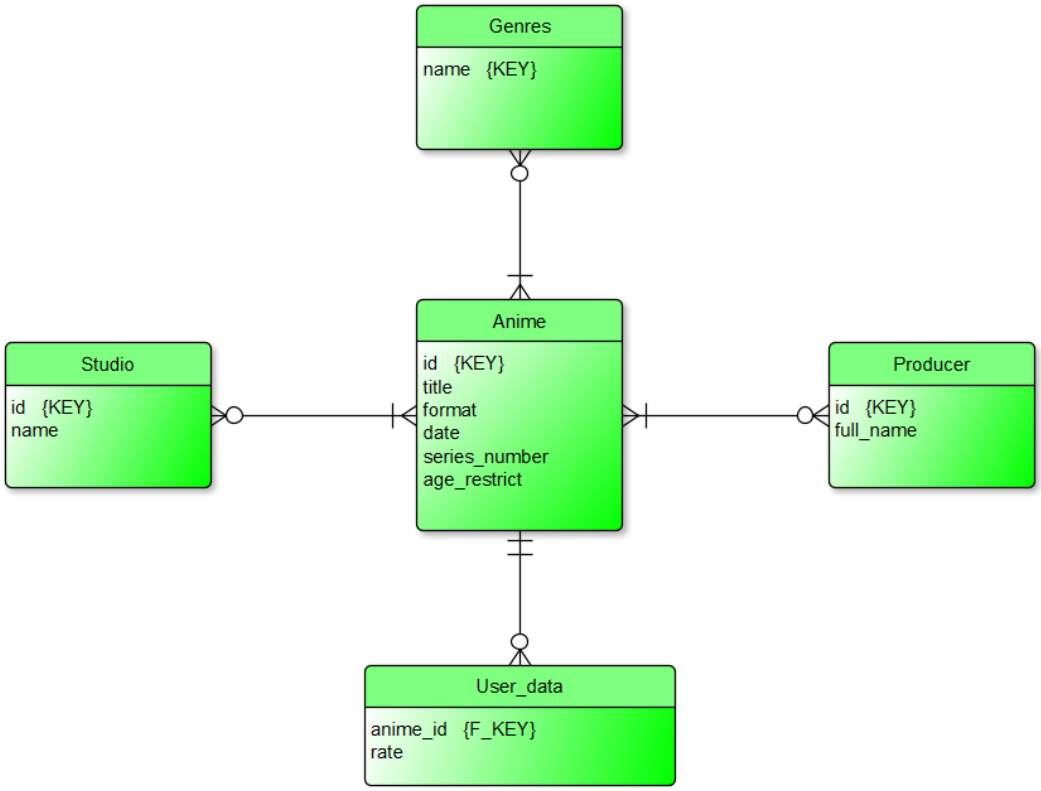

Рисунок 1 – Концептуальная диаграмма базы данных

3 Логическое проектирование БД

На рисунке 1 представлена ER-диаграмма логической стадии проектирования БД.

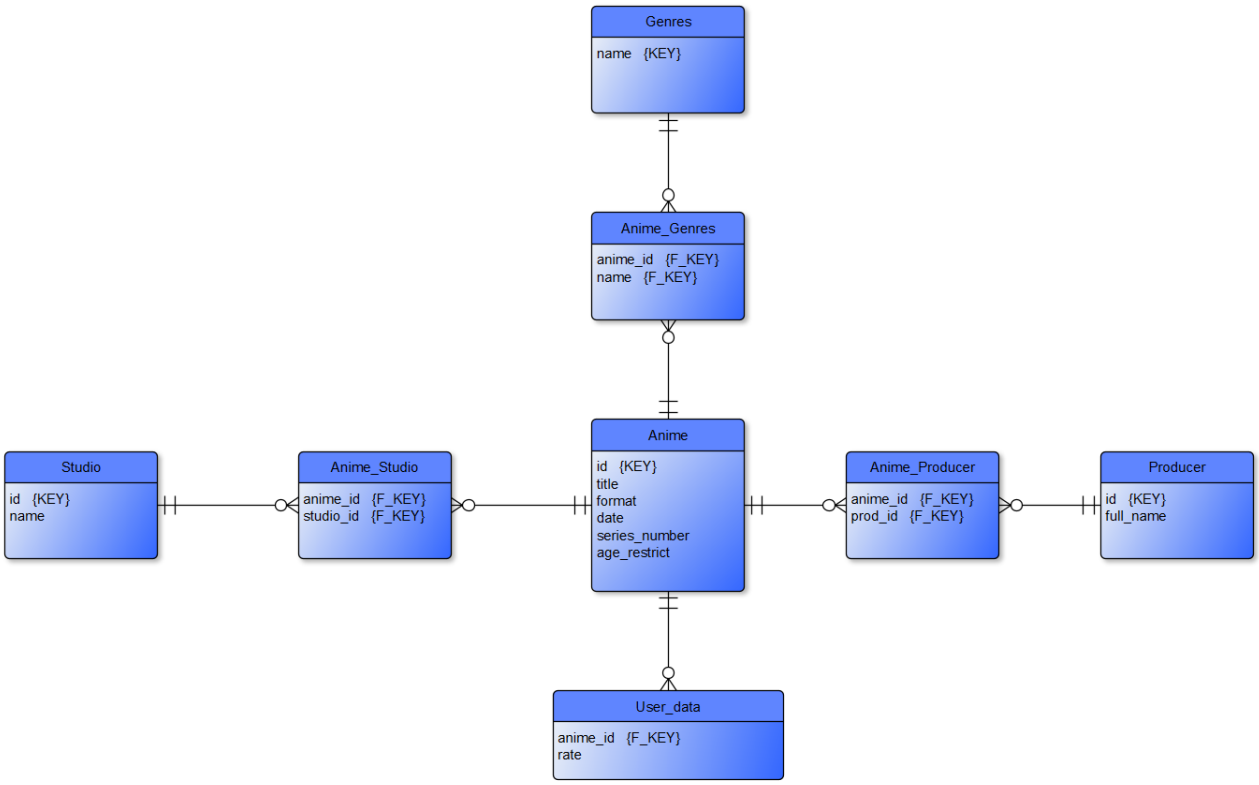

Рисунок 2 – Логическая диаграмма базы данных

4 Физическое проектирование БД

На листингах ниже представлены SQL-команды физического проектирования БД.

Листинг 1 – Создание таблиц с базами данных

```
BEGIN;

CREATE TABLE IF NOT EXISTS Anime
(
id SERIAL PRIMARY KEY,
title text NOT NULL,
format anime_format,
year int NOT NULL,
season seasons,
status status NOT NULL,
age_restrict age_restriction,
series_number int,
CONSTRAINT series_number_check CHECK ( series_number > 0 ),
CONSTRAINT date_check CHECK ( year > 1958 AND ( status = 'Анонсирован' OR year <=
EXTRACT( YEAR FROM DATE 'today'::date ) ) )
```

```
);

CREATE TABLE IF NOT EXISTS Studio
(
id SERIAL PRIMARY KEY,
name text NOT NULL
);

CREATE TABLE IF NOT EXISTS Producer
(
id SERIAL PRIMARY KEY,
full_name text NOT NULL,
CONSTRAINT full_name_check CHECK ( full_name ~* '^^[А-ЯЁ][а-яё]+ [А-ЯЁ][а-яё]+$'
)
);

CREATE TABLE IF NOT EXISTS Genres
(
name text NOT NULL PRIMARY KEY
);

-- Создание таблиц связей

CREATE TABLE IF NOT EXISTS Anime_Studio
(
anime_id int REFERENCES Anime ( id ),
studio_id int REFERENCES Studio ( id )
);

CREATE TABLE IF NOT EXISTS Anime_Producer
(
anime_id int REFERENCES Anime ( id ),
prod_id int REFERENCES Producer ( id )
);

CREATE TABLE IF NOT EXISTS Anime_Genres
(
anime_id int REFERENCES Anime ( id ),
genre text REFERENCES Genres ( name )
);

-- Создание таблицы пользователя
```

```
CREATE TABLE IF NOT EXISTS User_data
(
anime_id int REFERENCES Anime ( id ) PRIMARY KEY,
rate int,
CONSTRAINT series_number_check CHECK ( rate >= 1 AND rate <= 10)
);

END;
```

Листинг 2 – Заполнение информации главных таблиц

```
BEGIN;

INSERT INTO Studio ( name ) VALUES
('Sunrise'), ('J.C.Staff'), ('Kyoto Animation'), ('A-1 Pictures'), ('Gainax'),
('Trigger'), ('David Production'), ('Studio Pierrot'), ('Shaft'), ('CoMix Wave
Films');

INSERT INTO Producer ( full_name ) VALUES
('Шиничиро Ватанабе'), ('Симбо Акиюки'), ('Синкай Макото'), ('Ямада Наоко'),
('Цуда Наокацу'), ('Имайси Хироюки');

INSERT INTO Genres VALUES
('Альтернативная история'), ('Альтернативная реальность'), ('Ангелы'),
('Андроиды'), ('Антивойна'), ('Антиутопия'), ('Баскетбол'), ('Безумие'),
('Бисёнэн'), ('Боевые искусства'), ('Божества'), ('Вампиры'), ('Ведьмы'),
('Вестерн'), ('Виртуальная реальность'), ('Военная тематика'), ('Война'),
('Воры'), ('Гарем'), ('Гарем (для девочек)'), ('Гендерная интрига'), ('Демоны'),
('Детектив'), ('Дзёсэй'), ('Драконы'), ('Драма'), ('Зомби'), ('Игры'),
('Инопланетные расы'), ('Искусственный интеллект'), ('Искусство'),
('Исторический'), ('Исэкай'), ('Киберпанк'), ('Киборги'), ('Комедия'),
('Космос'), ('Кулинария'), ('Лоликон'), ('Любовный треугольник'), ('Магия'),
('Манга'), ('Мафия/Якудза'), ('Махо-сёдзё'), ('Машины'), ('Меха'), ('Мистика'),
('Музыка'), ('Не японское'), ('Нелинейный сюжет'), ('Ниндзя'), ('Охотники за
головами'), ('Параллельные миры'), ('Пародия'), ('Перестрелки'), ('Пилотируемые
роботы'), ('Пираты'), ('Повседневность'), ('Политика'), ('Полицейские'),
('Полулюди'), ('Постапокалиптика'), ('Преступность'), ('Призраки'),
('Приключения'), ('Прокси бои'), ('Психология'), ('Путешествия во времени'),
('Романтика'), ('Русалки'), ('Русские в аниме'), ('Самураи'),
('Сверхъестественное'), ('Сёдзё'), ('Сёдзё-ай'), ('Сёнэн'), ('Сёнэн-ай'),
('Силовые костюмы'), ('Современное фэнтези'), ('Спорт'), ('Сражения на мечах'),
('Стимпанк'), ('Суккубы'), ('Суперспособности'), ('Сэйнэн'), ('Тайный заговор'),
('Темное фэнтези'), ('Темные эльфы'), ('Террористы'), ('Трансформеры'),
```

```
('Триллер'), ('Убийцы'), ('Ужасы'), ('Фантастика'), ('Феи'), ('Фэнтези'),
('Школьная жизнь'), ('Экшен'), ('Эльфы'), ('Эротика'), ('Этти');

INSERT INTO Anime (title, format, year, season, status, age_restrict,
series_number) VALUES
('K-ON!', 'Сериал', 2009, 'Весна', 'Вышел', 'PG-13', 13),
('Невероятное приключение ДжоДжо: Золотой Ветер', 'Сериал', 2018, 'Осень',
'Вышел', 'R+', 39),
('Невероятное приключение ДжоДжо: Каменный океан', 'Сериал', 2021, NULL,
'Анонсирован', 'R+', NULL),
('Твое имя', 'Фильм', 2016, 'Лето', 'Вышел', 'PG-13', 1),
('Гуррен-Лаганн', 'Сериал', 2007, 'Весна', 'Вышел', 'PG-13', 26),
('Ковбой Бибоп', 'Сериал', 1998, 'Весна', 'Вышел', 'R+', 26);

INSERT INTO User_data VALUES
(1, 10), (2, 8), (4, 4);

END;
```

Листинг 3 – Заполнение информации таблиц связей

```
BEGIN;

INSERT INTO Anime_Studio VALUES
(1, 3), (2, 7), (3, 7), (4, 10), (5, 5), (6, 1);

INSERT INTO Anime_Producer VALUES
(1, 4), (2, 5), (3, 5), (4, 3), (5, 6), (6, 1);

INSERT INTO Anime_Genres VALUES
(1, 'Сэйнэн'), (1, 'Комедия'), (1, 'Искусство'), (1, 'Музыка'), (1,
'Повседневность'), (1, 'Школьная жизнь'),
(2, 'Сёнэн'), (2, 'Приключения'), (2, 'Экшен'), (2, 'Прокси бои'), (2,
'Суперспособности'),
(3, 'Сёнэн'), (3, 'Приключения'), (3, 'Экшен'),
(4, 'Драма'), (4, 'Романтика'), (4, 'Школьная жизнь'), (4, 'Сверхъестественное'),
(5, 'Сёнэн'), (5, 'Драма'), (5, 'Комедия'), (5, 'Меха'), (5, 'Приключения'), (5,
'Фантастика'), (5, 'Космос'),
(6, 'Сэйнэн'), (6, 'Драма'), (6, 'Комедия'), (6, 'Охотники за головами'), (6,
'Фантастика'), (6, 'Космос');

END;
```

Листинг 4 – Создание представлений

```
BEGIN;

CREATE MATERIALIZED VIEW IF NOT EXISTS Anime_data AS

WITH anime_plus_studios_column AS (
SELECT anime_id, ARRAY_AGG(studio_id) AS studio_ids, title, format, season, year,
status, series_number, age_restrict, ARRAY_AGG(Studio.name) AS studio
FROM Anime_Studio
LEFT JOIN Anime on Anime.id = anime_id
LEFT JOIN Studio on Studio.id = studio_id
GROUP BY anime_id, title, format, season, year, status, series_number,
age_restrict
),

producers_column AS (
SELECT anime_id, ARRAY_AGG(prod_id) AS prod_ids, ARRAY_AGG(Producer.full_name) AS
producer
FROM Anime_Producer
LEFT JOIN Anime on Anime.id = anime_id
LEFT JOIN Producer on Producer.id = prod_id
GROUP BY anime_id
),

genres_column AS (
SELECT anime_id, ARRAY_AGG(Genres.name) AS genres
FROM anime_genres
LEFT JOIN Anime on Anime.id = anime_id
LEFT JOIN Genres on Genres.name = genre
GROUP BY anime_id
)

SELECT src.anime_id, studio_ids, prod_ids, src.title, src.format, src.season,
src.year, src.studio, prod.producer, src.status, src.series_number,
src.age_restrict, gen.genres
FROM anime_plus_studios_column AS src
LEFT JOIN producers_column AS prod ON prod.anime_id = src.anime_id
LEFT JOIN genres_column AS gen ON gen.anime_id = src.anime_id;

-- Представление без системных данных

CREATE OR REPLACE VIEW anime_list AS
```

```
SELECT title, format, season, year, status, studio, series_number, age_restrict,
genres
FROM anime_data;

-- Представление пользователя

CREATE OR REPLACE VIEW user_list AS
SELECT title, format, season, year, status, studio, series_number, age_restrict,
genres, rate
FROM anime_data
RIGHT JOIN user_data on anime_data.anime_id = user_data.anime_id;

END;
```

Листинг 5 – Создание функций и триггера

```
BEGIN;

CREATE OR REPLACE FUNCTION get_status_by_id( id int, OUT status status )
AS $$
 SELECT status
 FROM anime_data
 WHERE id = anime_id;
$$ LANGUAGE SQL;

CREATE OR REPLACE FUNCTION check_announced_rate() RETURNS TRIGGER AS $$
BEGIN
    IF ( TG_OP = 'INSERT' OR TG_OP = 'UPDATE' ) AND ( get_status_by_id (
NEW.anime_id ) = 'Анонсирован' ) THEN
        IF ( NEW.rate IS NOT NULL ) THEN
                RAISE NOTICE 'Попытка оценить невышедшее аниме!';
                RETURN OLD;
            END IF;
        RETURN NEW;
    END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER rate_trigger BEFORE INSERT OR UPDATE ON user_data
    FOR EACH ROW EXECUTE FUNCTION check_announced_rate();

END;
```

5 Описание нормальности форм таблиц

Все таблицы находятся в первой нормальной форме, так как столбцы уникальны и не содержат повторяющихся атрибутов.

Все таблицы находятся во второй нормальной форме, так как все столбцы таблицы зависят от определенного ключа либо являются ключом.

Все таблицы находятся в третьей нормальной форме, так как все столбцы зависят только от ключей.

Таблицы Anime, Studio, Genres, Producer и User_data находятся в четвертой нормальной форме, так как связи многие-ко-многим в них устранены добавлением специальных таблиц.