ИИКС

Лабораторная работа №1: «Построение модели данных»

Грущин Илья

Б21-515

2024 г

Задание

1. Выбрать предметную область для дальнейшей работы. Предметная область должна быть составлена из нескольких взаимосвязанных процессов.
2. Проанализировать выбранную предметную область и разработать систему таблиц для хранения необходимой информации в рамках реляционной базы данных.В таблице должно быть 5-8 таблиц;
3. Если модель данных не отвечает требованиям по меньшей мере третьей нормальной формы, нормализовать её. Доказать, что требования третьей нормальной формы соблюдены
4. Построить диаграмму отношений сущностей (Entity Relations Diagram; ERD) для разработанной модели данных;

Реализовать разработанную схему данных средствами языка SQL в системе управления базами данных SQLite3.

На защиту

На защиту: бывает так, что издатель и разработчик - одна и та же компания, бывает, что разработчик находится в собственности издателя, а бывает так, что разработчик и издатель - независимые компании-партнёры. Хуже того, этот расклад имеет свойство меняться с течением времени.

Обновите структуру базы данных, чтобы хранить эти новые сведения

Решение: добавим таблицу

Предметная область: Видеоигры

В данной лабораторной работе рассматривается организация, занимающаяся систематизацией и управлением данными о видеоиграх, их разработчиках, издателях, жанрах, а также пользовательских рейтингах. Примерами могут выступать игровые журналисты или работники ИРИ, определяющие кому выделить денег на новую «Смуту».

Основная деятельность данной организации заключается в следующем:

* Хранение и управление информацией о видеоиграх, включая их название, год выпуска, возрастной рейтинг и пиковое количество игроков.
* Управление данными о разработчиках и издателях видеоигр.
* Классификация видеоигр по жанрам.
* Управление информацией о пользователях, включая их возраст и пол.
* Хранение и управление пользовательскими рейтингами и отзывами на видеоигры.

Спецификация таблиц

**Таблица Developers**

* developer\_id: уникальный идентификатор разработчика.
* name: имя разработчика.
* employee\_count — количество сотрудников.
* games\_released — количество выпущенных игр.

**Таблица Publishers**

* publisher\_id: уникальный идентификатор издателя.
* name: имя издателя.
* games\_published — количество изданных игр.
* average\_annual\_revenue — среднегодовой доход.

**Таблица Genres**

* genre\_id: уникальный идентификатор жанра.
* genre\_name: название жанра.

**Таблица Games**

* game\_id: уникальный идентификатор игры.
* title: название игры.
* release\_year: год выпуска игры.
* developer\_id: ссылка на разработчика игры.
* publisher\_id: ссылка на издателя игры.
* age\_rating: возрастной рейтинг игры.
* peak\_players: пиковое количество игроков в день.

**Таблица GameGenres**

* game\_id: ссылка на игру.
* genre\_id: ссылка на жанр.
* (game\_id, genre\_id): составной первичный ключ, обеспечивающий уникальность комбинации игра-жанр.

**Таблица Users**

* user\_id: уникальный идентификатор пользователя.
* username: имя пользователя.
* email: электронная почта пользователя.
* age: возраст пользователя.
* gender: пол пользователя.

**Таблица Ratings**

* rating\_id: уникальный идентификатор рейтинга.
* game\_id: ссылка на игру.
* user\_id: ссылка на пользователя.
* score: оценка игры.
* review: текст отзыва.
* rating\_date: дата выставления рейтинга.

**Таблица DeveloperPublisherRelationships**

* relationship\_id: уникальный идентификатор отношений.
* developer\_id: ссылка на разработчика.
* publisher\_id: ссылка на издателя.
* relationship\_type: вид отношений между разработчиком и издателем.
* start\_date: дата начала отношений.
* end\_date: дата окончания отношений.

Отношения могут быть 3 видов: independent – значит разработчик сам себе издатель, partner – разработчик и издатель заключили договор об издании игр, owner – издатель владеет разработчиком.

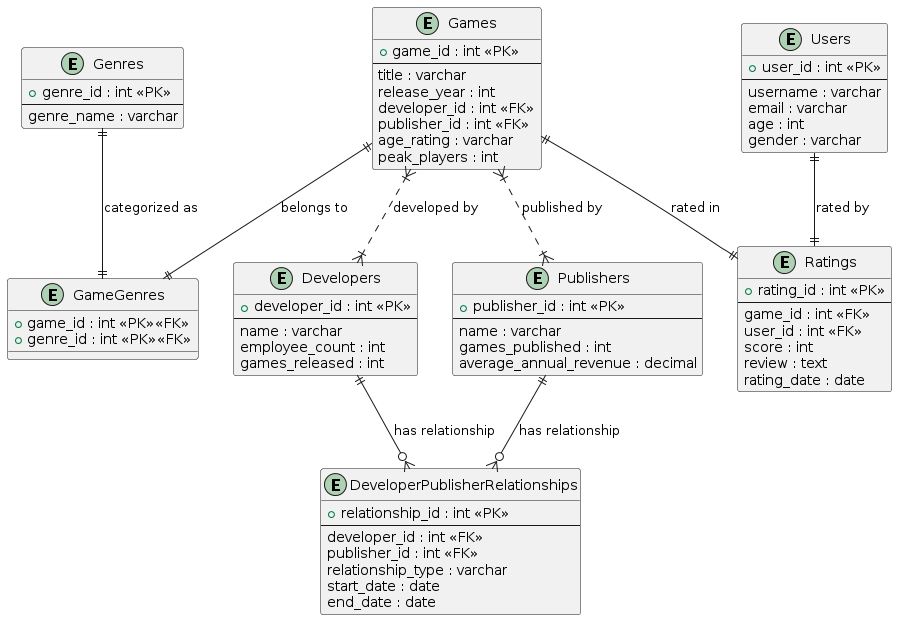
Доказательство соответствия третьей нормальной форме (3NF)

Первая нормальная форма (1NF): Все атрибуты атомарны. Все таблицы содержат только атомарные (неделимые) значения.

Вторая нормальная форма (2NF): Все неключевые атрибуты зависят от всего первичного ключа. Все таблицы либо имеют одинарный первичный ключ, либо составной ключ, как в таблице GameGenres, где атрибуты зависят от обоих ключевых атрибутов.

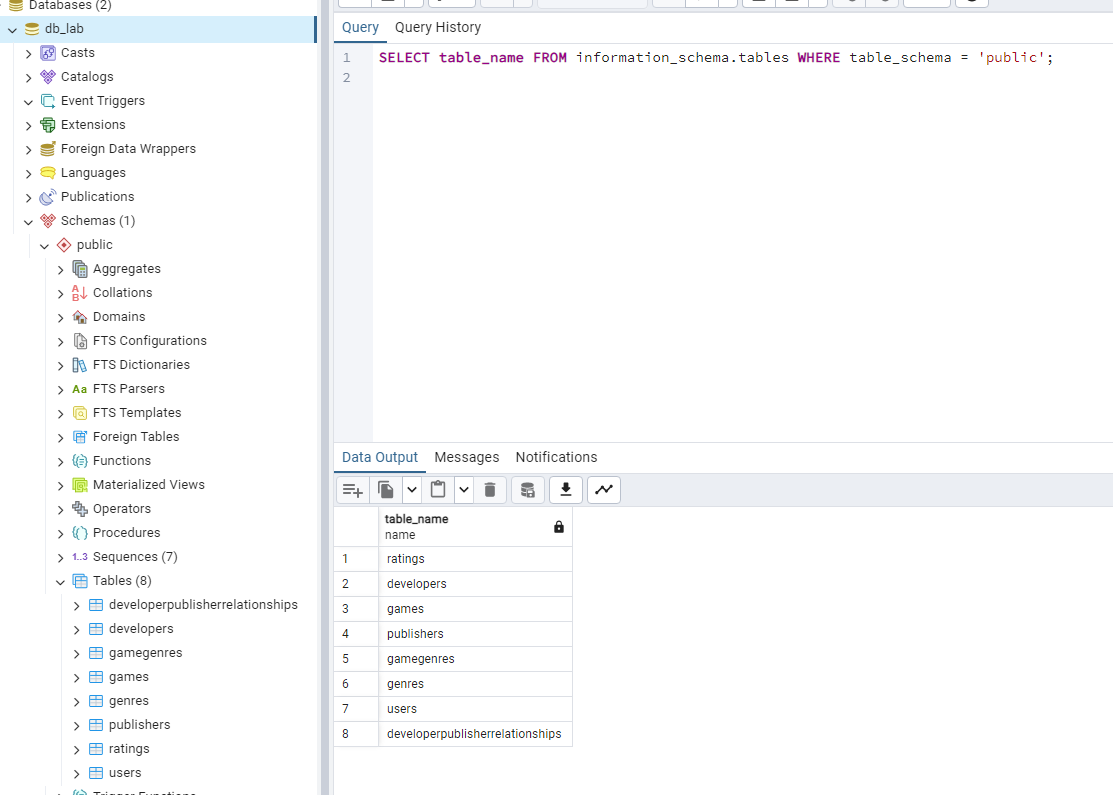
Третья нормальная форма (3NF): Все неключевые атрибуты не зависят транзитивно от первичного ключа. В наших таблицах нет атрибутов, которые зависят от других неключевых атрибутов.

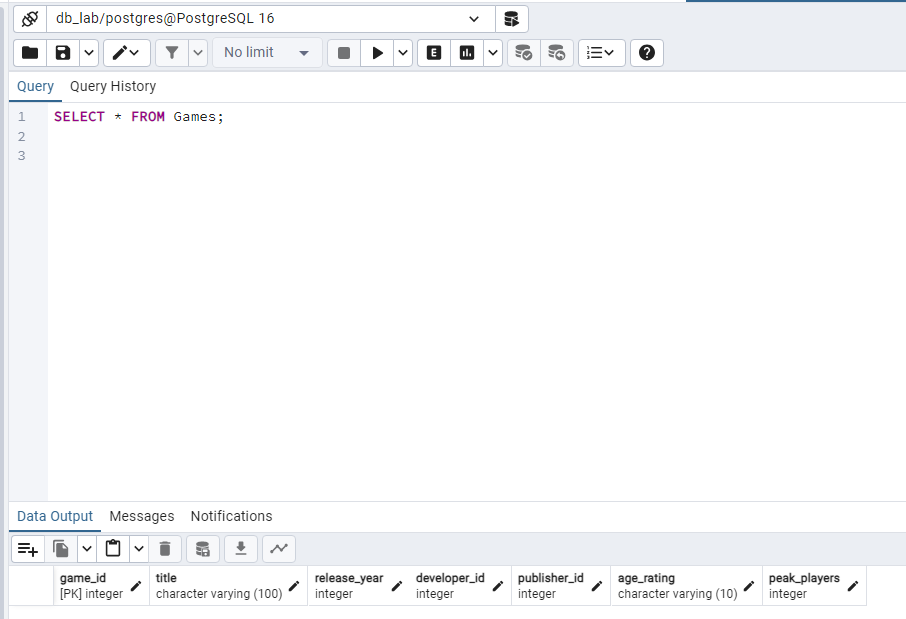
ERD



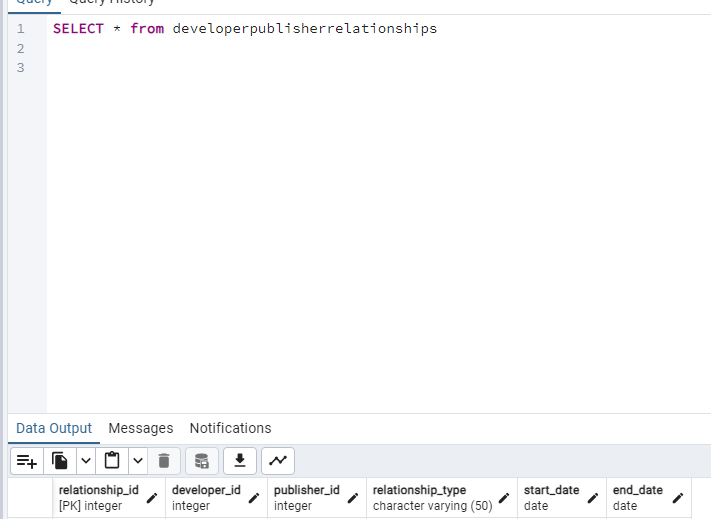
Заключение

В ходе данной лабораторной работы была разработана реляционная база данных для хранения информации о видеоиграх, их разработчиках, издателях, жанрах и пользовательских рейтингах. Создано семь таблиц, каждая из которых соответствует требованиям третьей нормальной формы. С помощью диаграммы отношений сущностей (ERD) была визуализирована структура базы данных и взаимосвязи между таблицами. Также был разработан SQL-сценарий (см. Приложение) для создания таблиц. Результат – указанный сценарий, диаграмма отношений сущностей созданная в PostgreSQL база данных (см. рисунки ниже)





При выполнении задания на защиту добавлена таблица с данными о взаимоотношениях студии-разработчика и издателя:



Приложение

Файл create.sql, реализующий создание таблицы.

CREATE TABLE Developers (

    developer\_id SERIAL PRIMARY KEY,

    name VARCHAR(100) NOT NULL,

    employee\_count INT,

    games\_released INT

);

CREATE TABLE Publishers (

    publisher\_id SERIAL PRIMARY KEY,

    name VARCHAR(100) NOT NULL,

    games\_published INT,

    average\_annual\_revenue DECIMAL(15, 2)

);

CREATE TABLE Genres (

    genre\_id SERIAL PRIMARY KEY,

    genre\_name VARCHAR(100) NOT NULL

);

CREATE TABLE Games (

    game\_id SERIAL PRIMARY KEY,

    title VARCHAR(100) NOT NULL,

    release\_year INT,

    developer\_id INT REFERENCES Developers(developer\_id),

    publisher\_id INT REFERENCES Publishers(publisher\_id),

    age\_rating VARCHAR(10),

    peak\_players INT

);

CREATE TABLE GameGenres (

    game\_id INT,

    genre\_id INT,

    PRIMARY KEY (game\_id, genre\_id),

    FOREIGN KEY (game\_id) REFERENCES Games(game\_id),

    FOREIGN KEY (genre\_id) REFERENCES Genres(genre\_id)

);

CREATE TABLE Users (

    user\_id SERIAL PRIMARY KEY,

    username VARCHAR(100) NOT NULL,

    email VARCHAR(100) UNIQUE NOT NULL,

    age INT,

    gender VARCHAR(10)

);

CREATE TABLE Ratings (

    rating\_id SERIAL PRIMARY KEY,

    game\_id INT REFERENCES Games(game\_id),

    user\_id INT REFERENCES Users(user\_id),

    score INT,

    review TEXT,

    rating\_date DATE

);

CREATE TABLE DeveloperPublisherRelationships (

    relationship\_id SERIAL PRIMARY KEY,

    developer\_id INT NOT NULL,

    publisher\_id INT NOT NULL,

    relationship\_type VARCHAR(50),

    start\_date DATE,

    end\_date DATE,

    FOREIGN KEY (developer\_id) REFERENCES Developers(developer\_id),

    FOREIGN KEY (publisher\_id) REFERENCES Publishers(publisher\_id)

);