

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 і засоби управління**

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

Виконав:

студент ІII курсу

групи КВ-01

Роговий Д.С.

Перевірив:

Павловський В. І.

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. Ознайомитися із інструментарієм PostgreSQL та pgAdmin 4 та занести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

**Опис Предметової галузі**

Для лабораторної роботи я обрав тему – Інтернет-магазин ігор.

При проектуванні бази даних «Магазин» можна виділити наступні сутності:

1. Загальні відомості про видавця гри (publisher);
2. Інформація про власника компанії-видавця (owner);
3. Список всіх тегів для сортування ігр (tag);
4. Загальна інформація про гру(game);

Атрибути заданих сутностей:

1. publisher: name, creation\_year, country;
2. owner: name, age;
3. game: price, release\_date, name;
4. tag: name;

**Опис зв’язків**

У кожного видавця може бути багато ігор, тому між сутностями publisher і game зв’язок 1:N

Кожен власник може володіти лише одним видавцем, тому між сутностями owner і publisher зв’язок 1:1.

У кожної гри може бути багато тегів (жанрів), і кожен тег може відповідати більше ніж одній грі, тому між сутностями game і tag зв’язок N:M

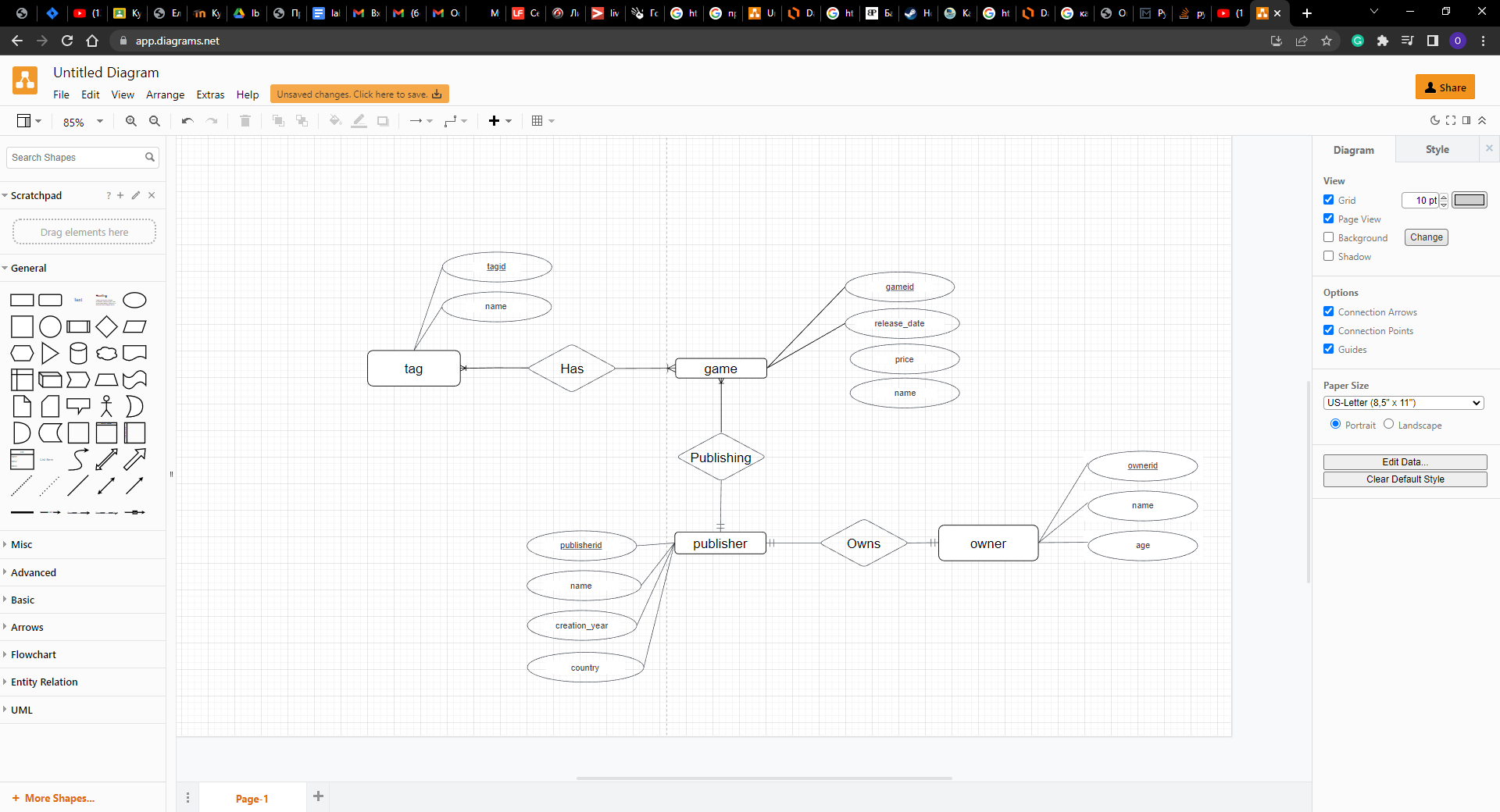


Рисунок 1 - ER-діаграма побудована за нотацією “Crow`s foot”, задана ER-діаграма побудована в додатку [lucid.app](https://lucid.app/)

**Перетворення концептуальної моделі у схему баз даних**

1. Якщо зв'язок типу 1:N і клас приналежності сутності (характер участі сутності у зв’язку, буває обов’язковий і необов’язковий клас) на стороні N є обов'язковим, то необхідно побудувати таблицю для кожної сутності. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Первинний ключ сутності на стороні 1 додається як атрибут в таблицю для сутності на стороні N. Даний зв’язок можна спостерігати між сутностями publisher і game.
2. Якщо зв’язок типу 1:1, то необхідно побудувати таблицю для кожної сутності. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Первинний ключ головної сутності оголошується унікальним(UNIQUE) та додається як атрибут в таблицю залежної сутності.
3. Якщо зв’язок типу N:M, то необхідно побудувати три таблиці - по одній для кожної сутності і одну для зв'язку. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Таблиця для зв'язку серед своїх атрибутів повинна мати зовнішні ключі - обох сутностей. Даний зв’язок можна спостерігати між сутностями game і tag

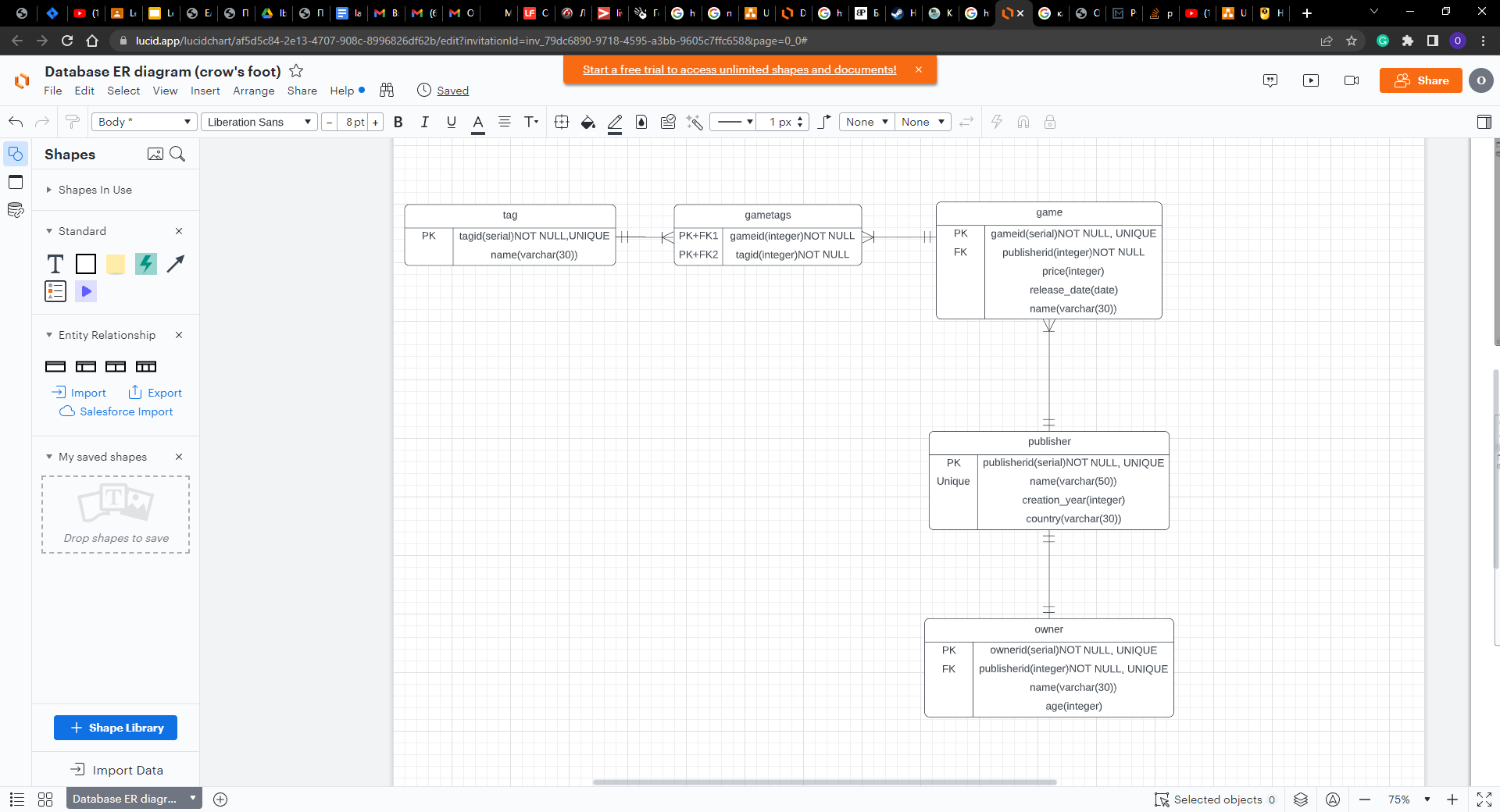


Рисунок 2 - Схема бази даних, побудовано у додатку [lucid.app](https://lucid.app/)

Після розробки моделі предметної галузі “Магазин” та перетворення її у схему бази даних, було створено дану базу даних у додатку pgAdmin 4

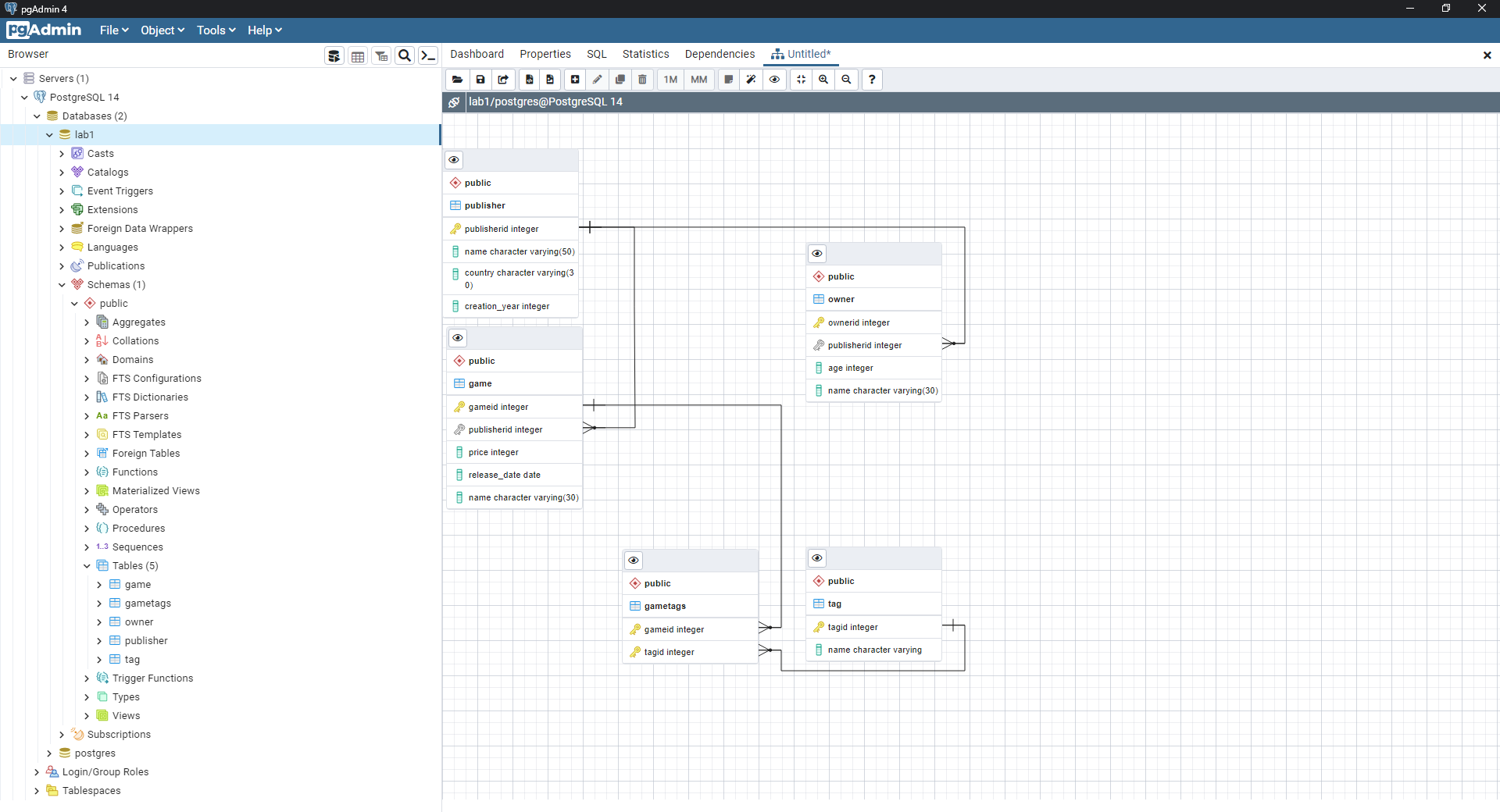


Рисунок 3 – Схема бази даних, сгенерована у pgadming4

**Опис структури бази данних**

|  |  |  |
| --- | --- | --- |
| Відношення | Атрибут | Тип артибуту |
| **publisher** – містить данні про компанію-видавця комп’ютерних ігр | **publisherid** – унікальний ідентефікатор  **name** – назва видавця  **creation\_year** – дата створення видавця  **country** – країна видавця | integer  character varying  integer  character varying |
| **owner** – інформація про власника | **ownerid** **-** унікальний ідентефікатор  **publisherid** – ідентифікатор видавця  **name** – ім’я власника  **age** – вік власника | integer  integer  character varying  integer |
| **game** | **gameid** - унікальний ідентефікатор  **publisherid** – ідентифікатор видавця  **price** **–** ціна гри  **release\_date** – дата виходу гри  **name** – назва гри | integer  integer  integer  date  character varying |
| **tag** | **tagid** - унікальний ідентефікатор  **name** – назва тегу | integer  character varying |
| **gametags** | **gameid** - ідентифікатор гри  **tagid** - ідентифікатор тегу | integer  integer |

**Нормалізація схеми баз даних до третьої нормальної форми**

Функціональні залежності для кожної таблиці

1. publisher

* publisherid -> name (назва залежить від унікального коду видавця)
* publisherid -> country(країна реєстрації залежить від унікального коду видавця)

1. owner

* ownerid -> name (ім’я залежить від унікального коду власника)
* ownerid -> age (вік залежить від унікального коду власника)

1. game

* gameid -> price (ціна залежить від унікального коду гри)
* gameid -> release\_date (дата виходу залежить від унікального коду гри)
* gameid -> name (назва залежить від унікального коду гри)

1. tag

* tagid -> name (назва залежить від унікального коду тегу)

Схема відповідає 1НФ, тому що:

1. В таблиці немає дубльованих рядків.
2. В кожній комірці зберігається атомарне (скалярне) значення.
3. В кожному стовпці зберігаються дані одного типу.

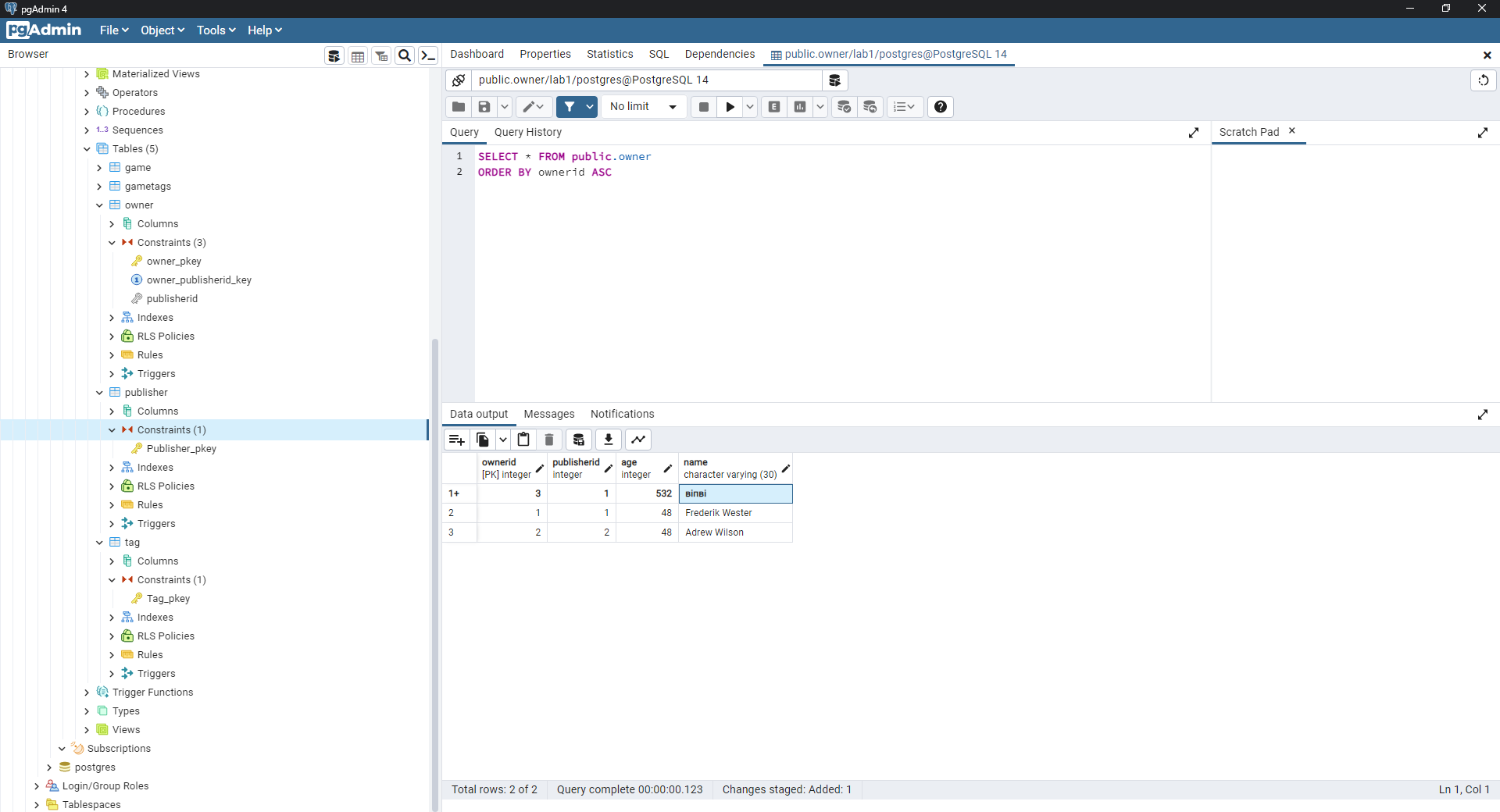
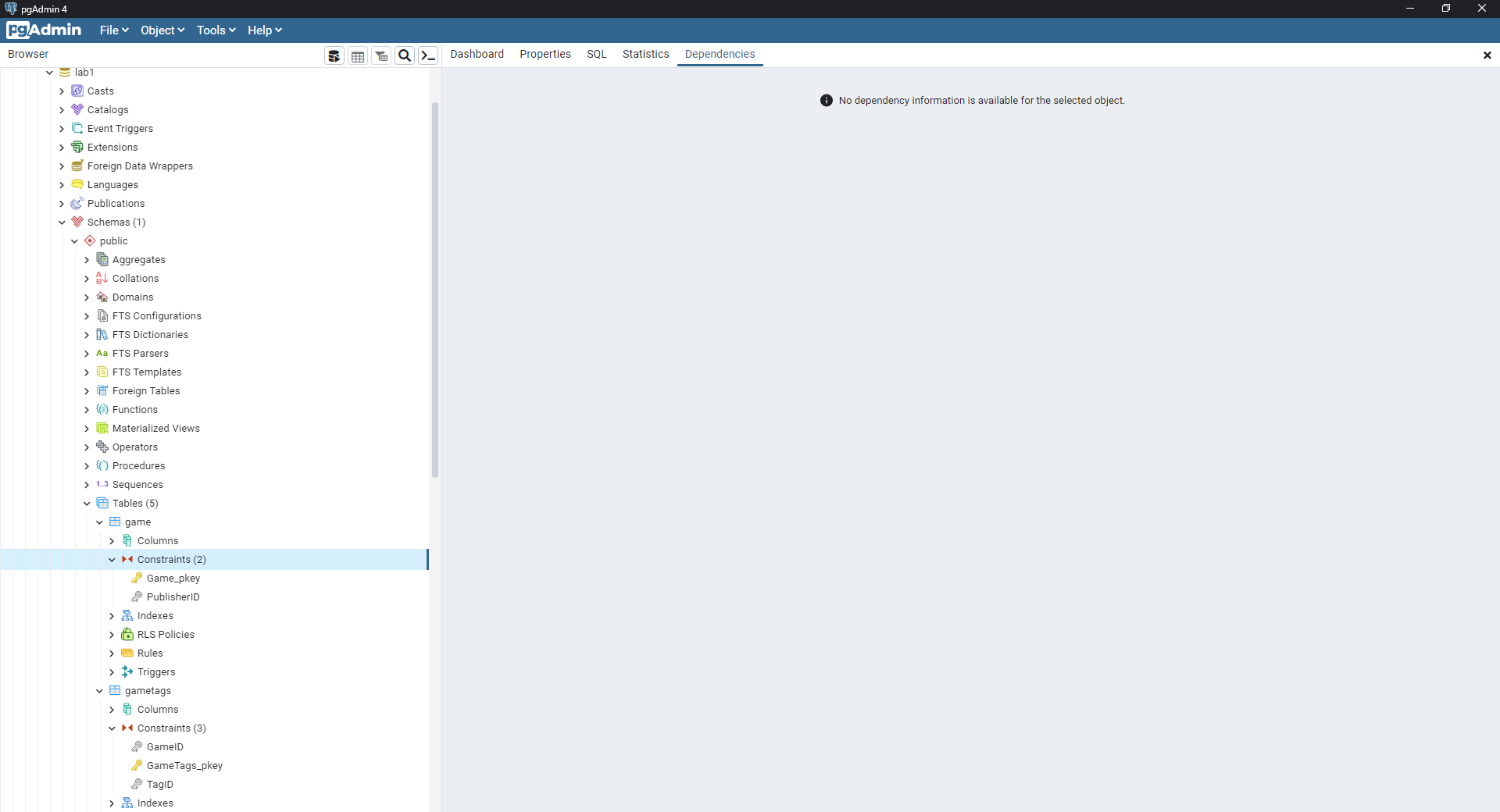
Схема відповідає 2НФ, тому що:

1. Вона відповідає 1НФ.
2. Має первинний ключ, а всі не ключові стовпці таблиці залежать від первинного ключа.

Схема відповідає 3НФ, тому що:

1. Вона відповідає 2НФ.
2. Всі не ключові атрибути таблиці залежать винятково від усього первинного ключа, а не його частини. Тобто кожен неключовий атрибут нетранзитивно (без посередника) залежить від первинного ключа

**Фізична модель бд у pgadmin4**



**SQL TEXT**

-- Database: lab1

-- DROP DATABASE IF EXISTS lab1;

**CREATE** **DATABASE** lab1

**WITH**

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'English\_United States.1252'

LC\_CTYPE = 'English\_United States.1252'

**TABLESPACE** = pg\_default

**CONNECTION** **LIMIT** = -1

IS\_TEMPLATE = **False**;

-- Table: public.game

-- DROP TABLE IF EXISTS public.game;

**CREATE** **TABLE** **IF** **NOT** **EXISTS** **public**.game

(

gameid **integer** **NOT** **NULL** **GENERATED** **BY** **DEFAULT** **AS** **IDENTITY** ( **INCREMENT** 1 **START** 1 **MINVALUE** 1 **MAXVALUE** 2147483647 **CACHE** 1 ),

publisherid **integer** **NOT** **NULL**,

price **integer**,

release\_date **date**,

name **character** **varying**(30) **COLLATE** pg\_catalog."default",

**CONSTRAINT** "Game\_pkey" **PRIMARY** **KEY** (gameid),

**CONSTRAINT** "PublisherID" **FOREIGN** **KEY** (publisherid)

**REFERENCES** **public**.publisher (publisherid) **MATCH** SIMPLE

**ON** **UPDATE** **NO** **ACTION**

**ON** **DELETE** **NO** **ACTION**

)

**TABLESPACE** pg\_default;

**ALTER** **TABLE** **IF** **EXISTS** **public**.game

OWNER **to** postgres;

-- Table: public.gametags

-- DROP TABLE IF EXISTS public.gametags;

**CREATE** **TABLE** **IF** **NOT** **EXISTS** **public**.gametags

(

gameid **integer** **NOT** **NULL**,

tagid **integer** **NOT** **NULL**,

**CONSTRAINT** "GameTags\_pkey" **PRIMARY** **KEY** (gameid, tagid),

**CONSTRAINT** "GameID" **FOREIGN** **KEY** (gameid)

**REFERENCES** **public**.game (gameid) **MATCH** SIMPLE

**ON** **UPDATE** **NO** **ACTION**

**ON** **DELETE** **NO** **ACTION**,

**CONSTRAINT** "TagID" **FOREIGN** **KEY** (tagid)

**REFERENCES** **public**.tag (tagid) **MATCH** SIMPLE

**ON** **UPDATE** **NO** **ACTION**

**ON** **DELETE** **NO** **ACTION**

)

**TABLESPACE** pg\_default;

**ALTER** **TABLE** **IF** **EXISTS** **public**.gametags

OWNER **to** postgres;

-- Table: public.owner

-- DROP TABLE IF EXISTS public.owner;

**CREATE** **TABLE** **IF** **NOT** **EXISTS** **public**.owner

(

ownerid **integer** **NOT** **NULL** **GENERATED** **BY** **DEFAULT** **AS** **IDENTITY** ( **INCREMENT** 1 **START** 1 **MINVALUE** 1 **MAXVALUE** 2147483647 **CACHE** 1 ),

publisherid **integer** **NOT** **NULL**,

age **integer**,

name **character** **varying**(30) **COLLATE** pg\_catalog."default",

**CONSTRAINT** owner\_pkey **PRIMARY** **KEY** (ownerid),

**CONSTRAINT** owner\_publisherid\_key **UNIQUE** (publisherid),

**CONSTRAINT** publisherid **FOREIGN** **KEY** (publisherid)

**REFERENCES** **public**.publisher (publisherid) **MATCH** SIMPLE

**ON** **UPDATE** **NO** **ACTION**

**ON** **DELETE** **NO** **ACTION**

)

**TABLESPACE** pg\_default;

**ALTER** **TABLE** **IF** **EXISTS** **public**.owner

OWNER **to** postgres;

-- Table: public.publisher

-- DROP TABLE IF EXISTS public.publisher;

**CREATE** **TABLE** **IF** **NOT** **EXISTS** **public**.publisher

(

publisherid **integer** **NOT** **NULL** **GENERATED** **BY** **DEFAULT** **AS** **IDENTITY** ( **INCREMENT** 1 **START** 1 **MINVALUE** 1 **MAXVALUE** 2147483647 **CACHE** 1 ),

name **character** **varying**(50) **COLLATE** pg\_catalog."default",

country **character** **varying**(30) **COLLATE** pg\_catalog."default",

creation\_year **integer**,

**CONSTRAINT** "Publisher\_pkey" **PRIMARY** **KEY** (publisherid)

)

**TABLESPACE** pg\_default;

**ALTER** **TABLE** **IF** **EXISTS** **public**.publisher

OWNER **to** postgres;

-- Table: public.tag

-- DROP TABLE IF EXISTS public.tag;

**CREATE** **TABLE** **IF** **NOT** **EXISTS** **public**.tag

(

tagid **integer** **NOT** **NULL** **GENERATED** **BY** **DEFAULT** **AS** **IDENTITY** ( **INCREMENT** 1 **START** 1 **MINVALUE** 1 **MAXVALUE** 2147483647 **CACHE** 1 ),

name **character** **varying** **COLLATE** pg\_catalog."default",

**CONSTRAINT** "Tag\_pkey" **PRIMARY** **KEY** (tagid)

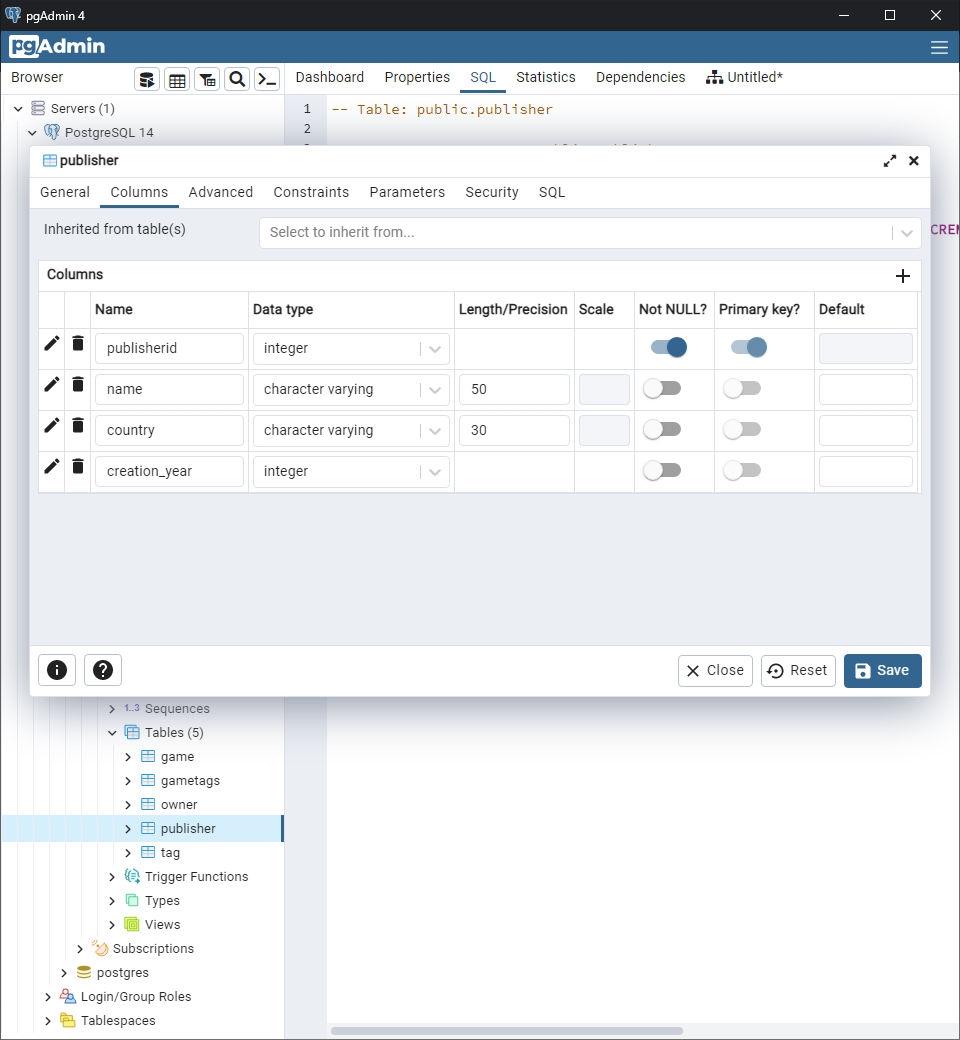
)

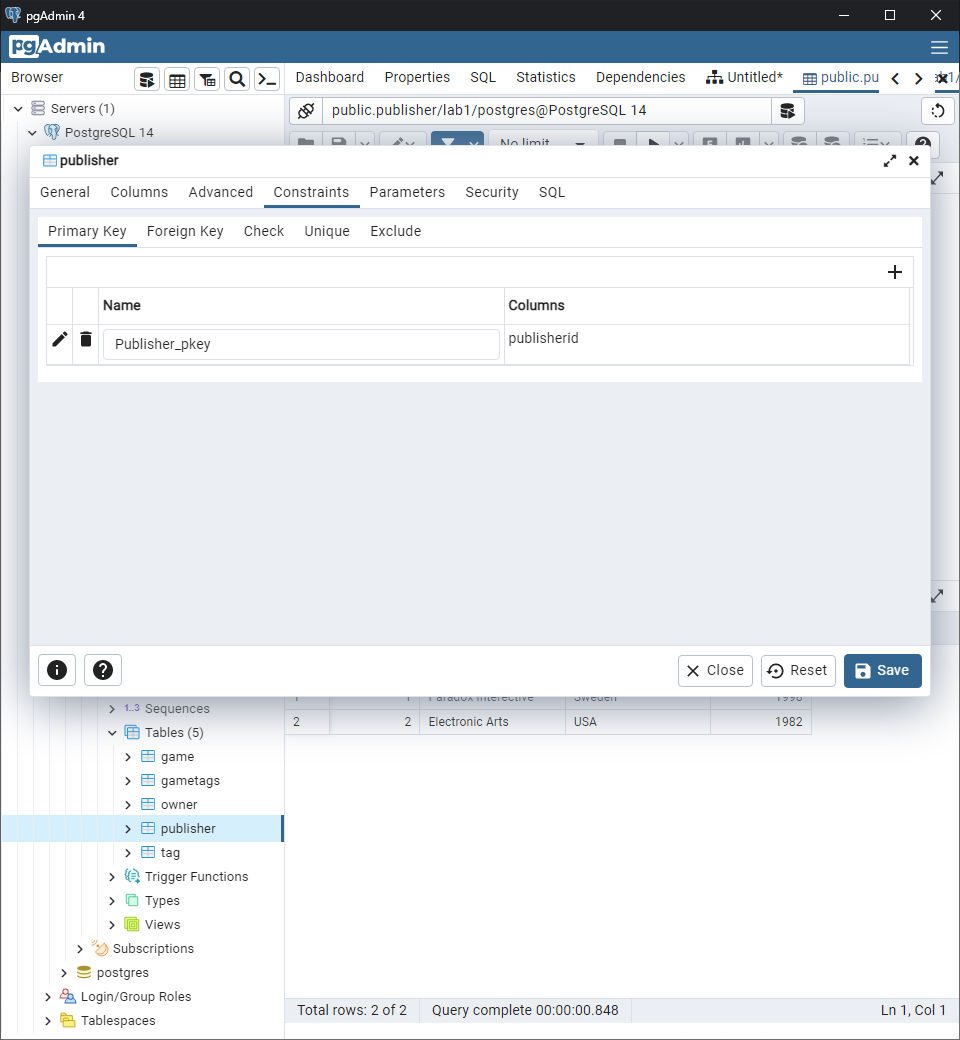
**TABLESPACE** pg\_default;

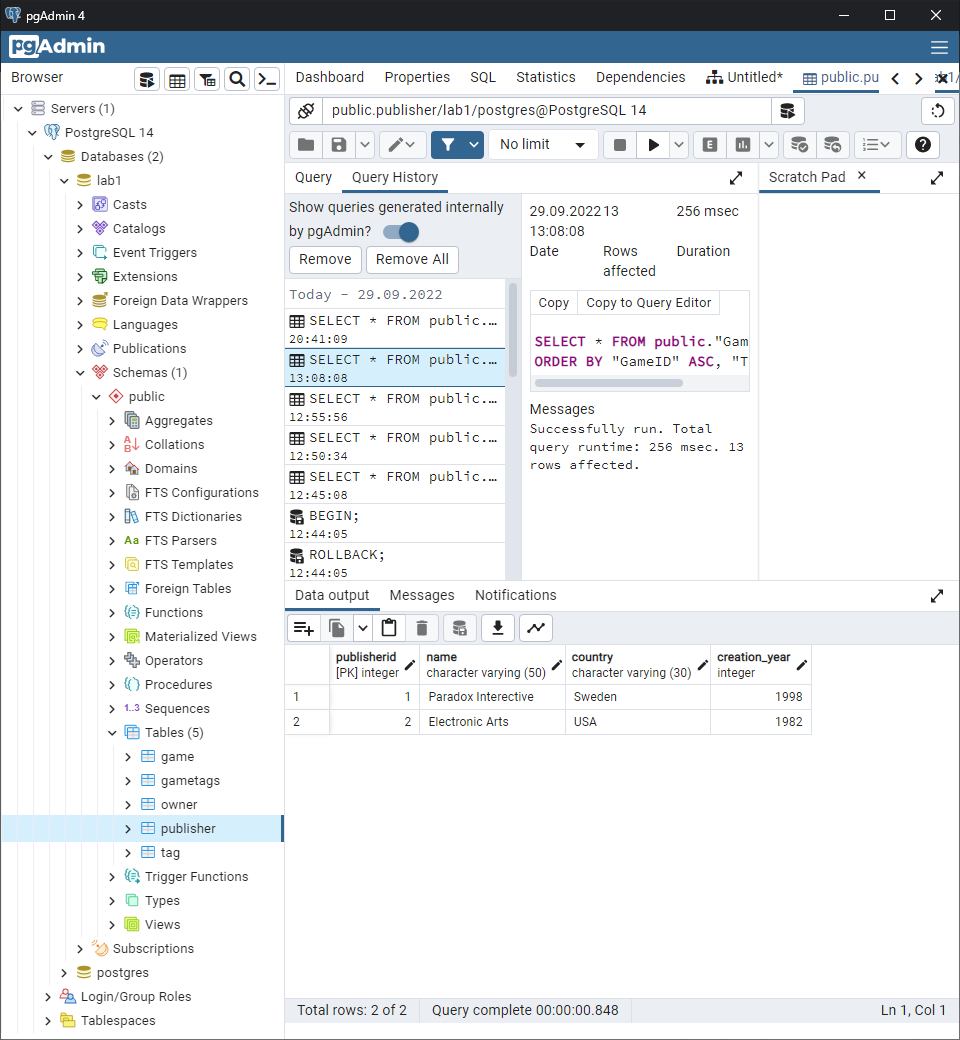
**ALTER** **TABLE** **IF** **EXISTS** **public**.tag

OWNER **to** postgres;

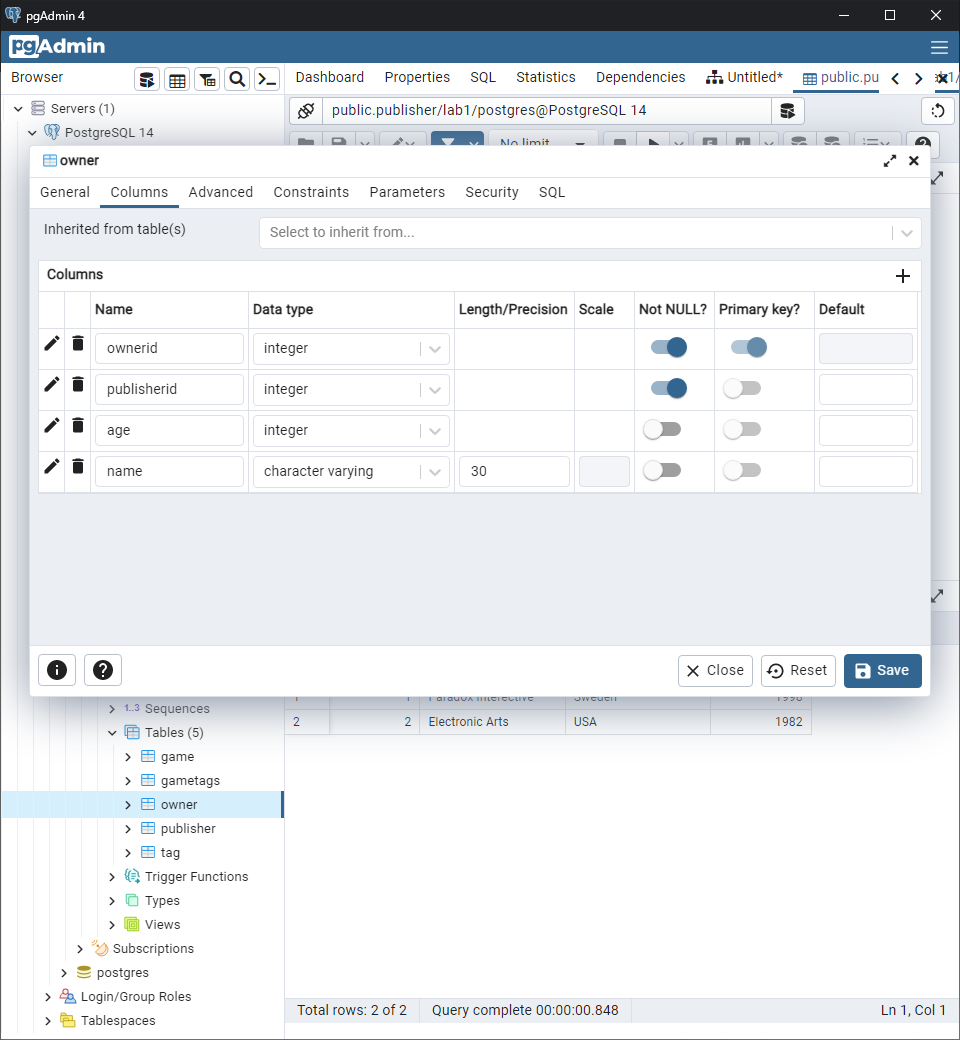
**Publisher:**

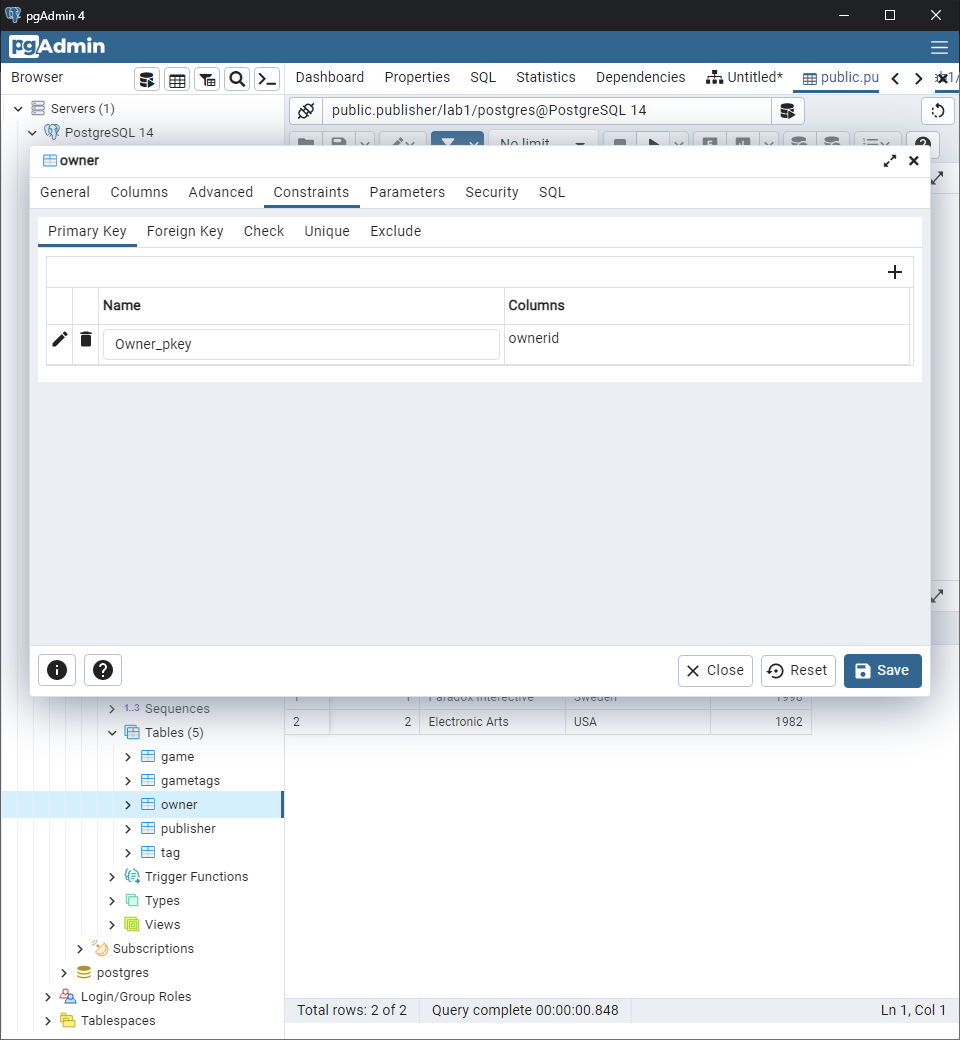


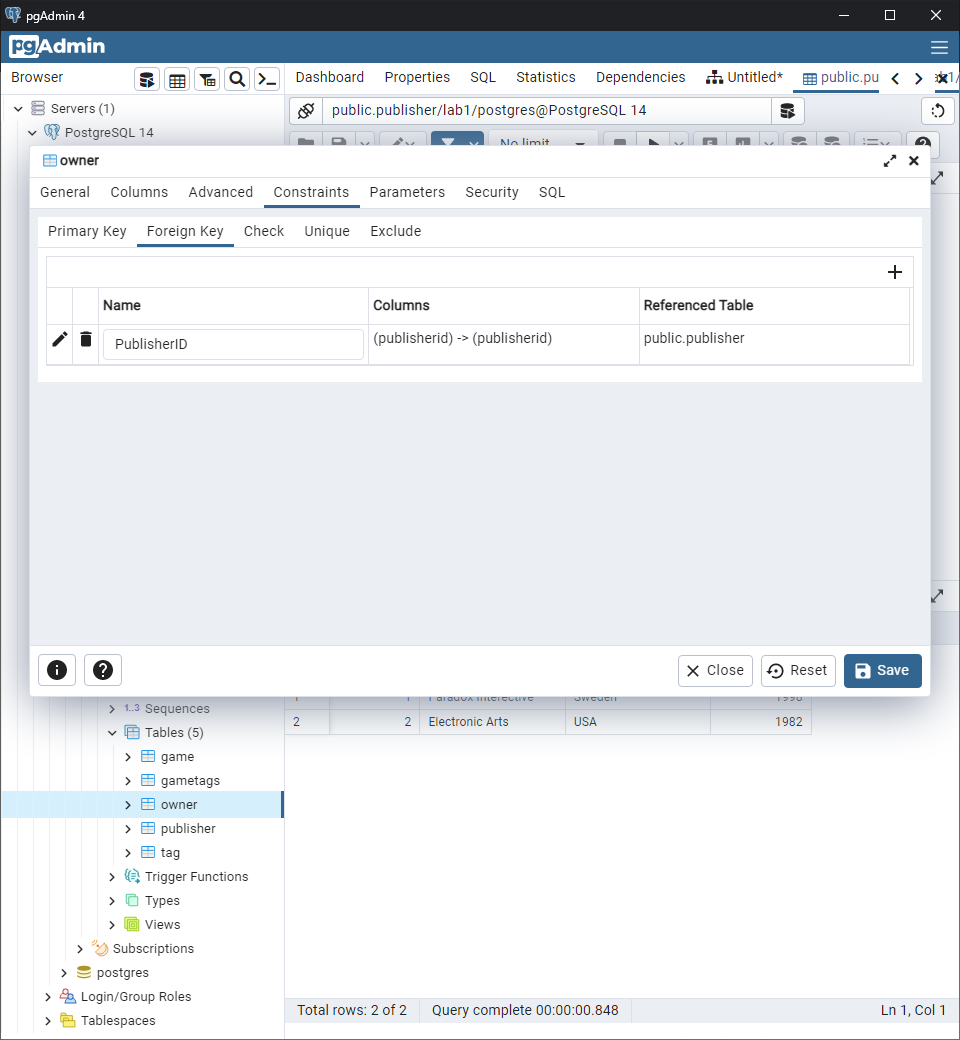


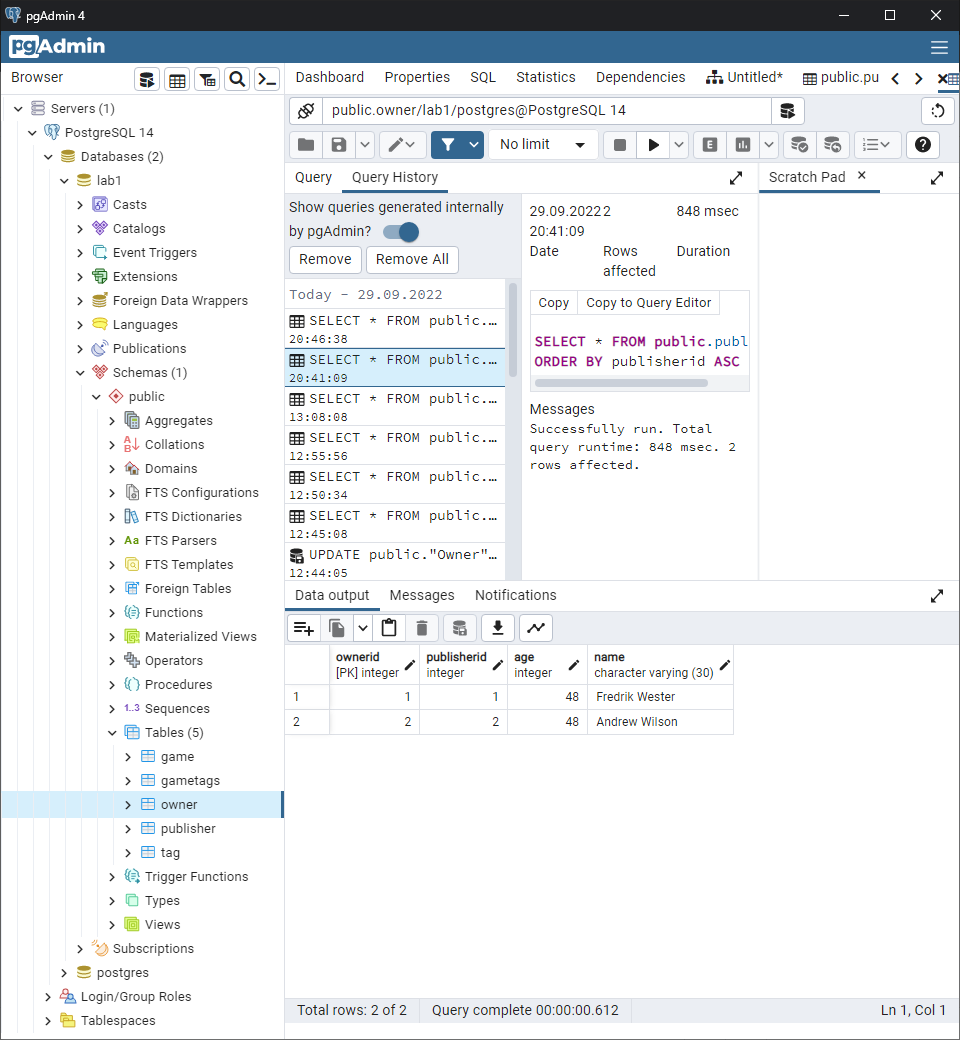


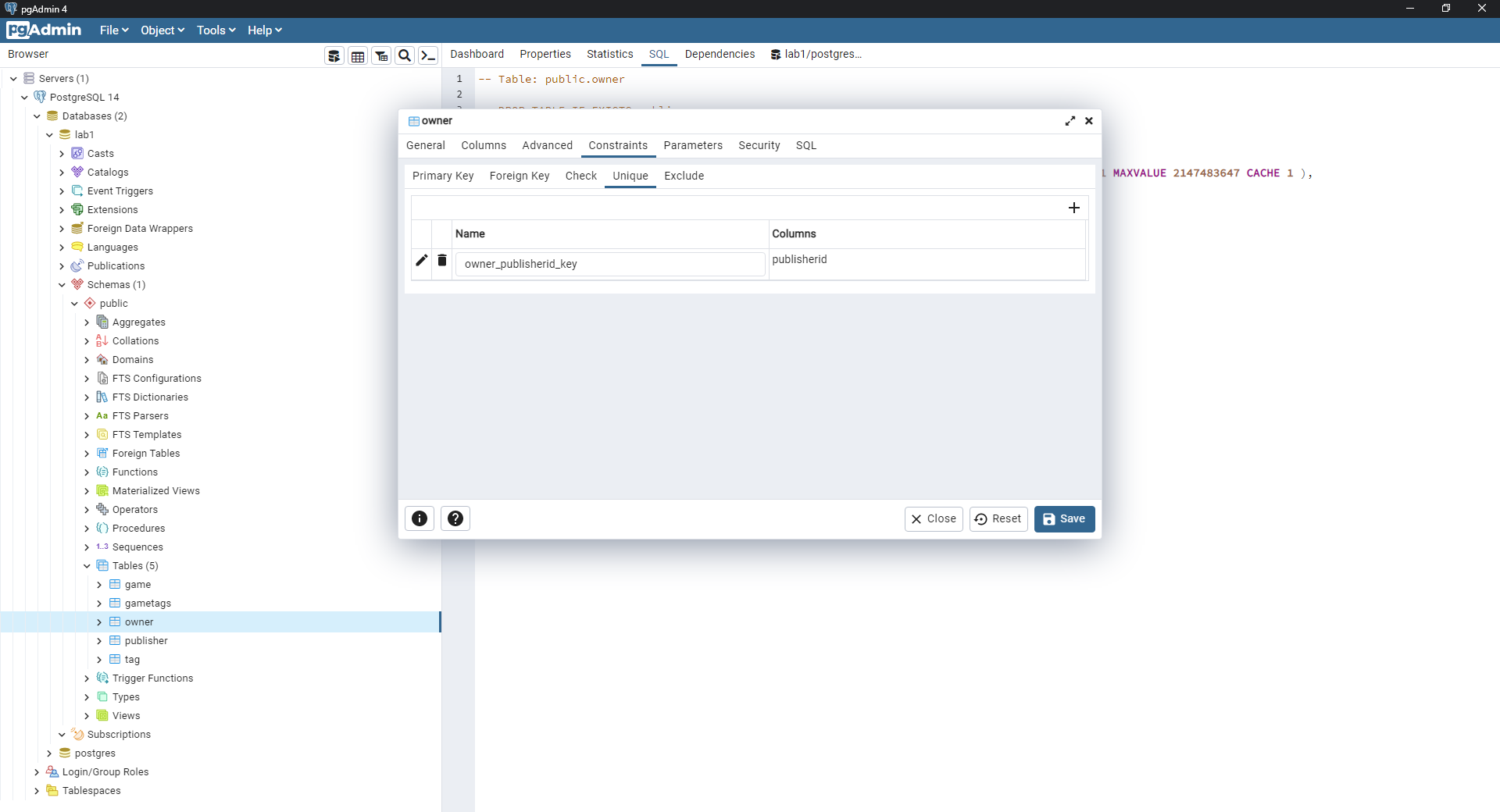
**Owner:**



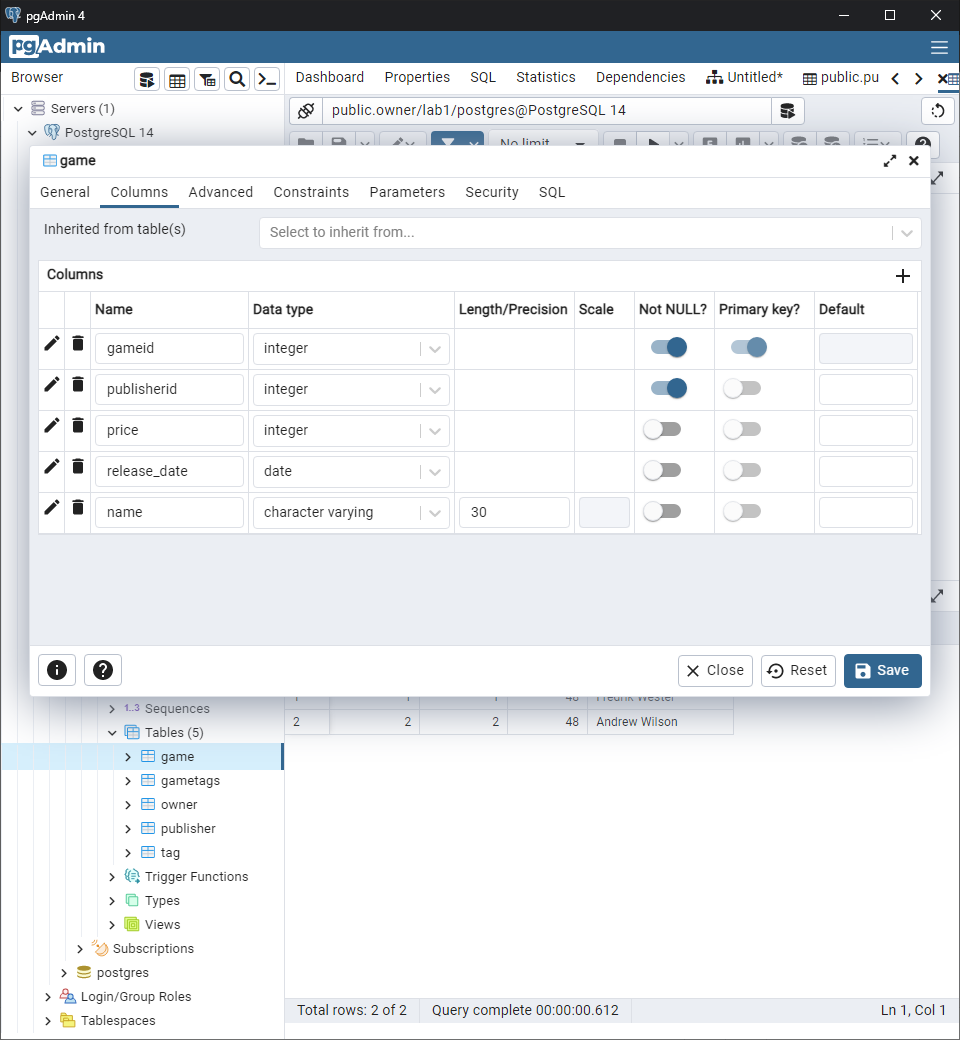


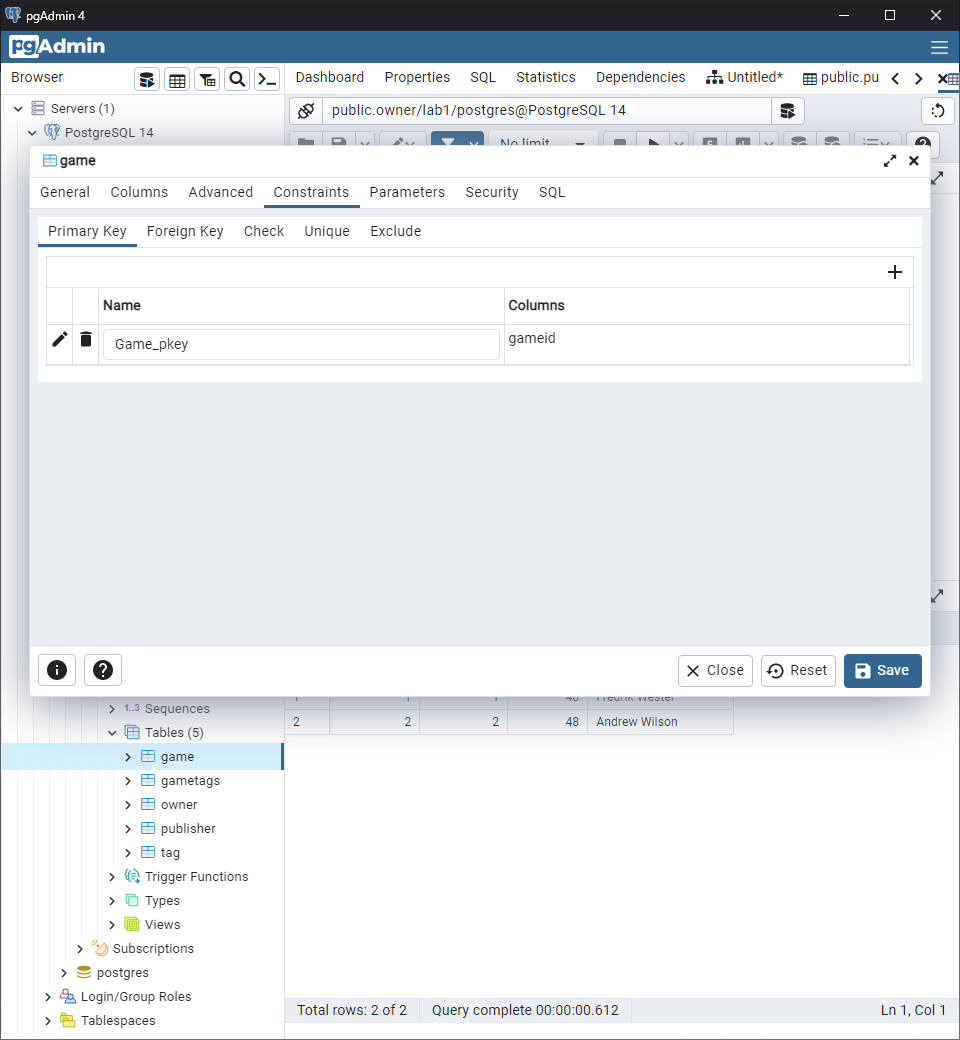


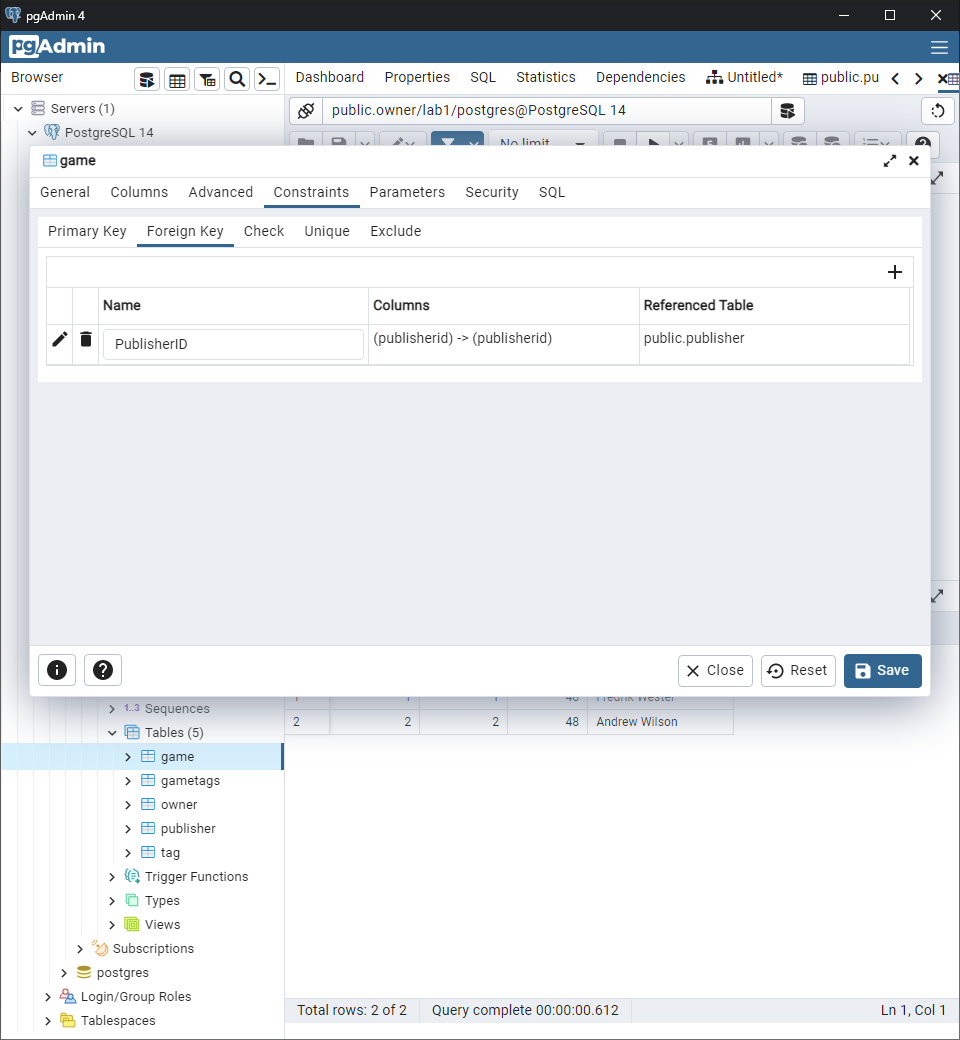


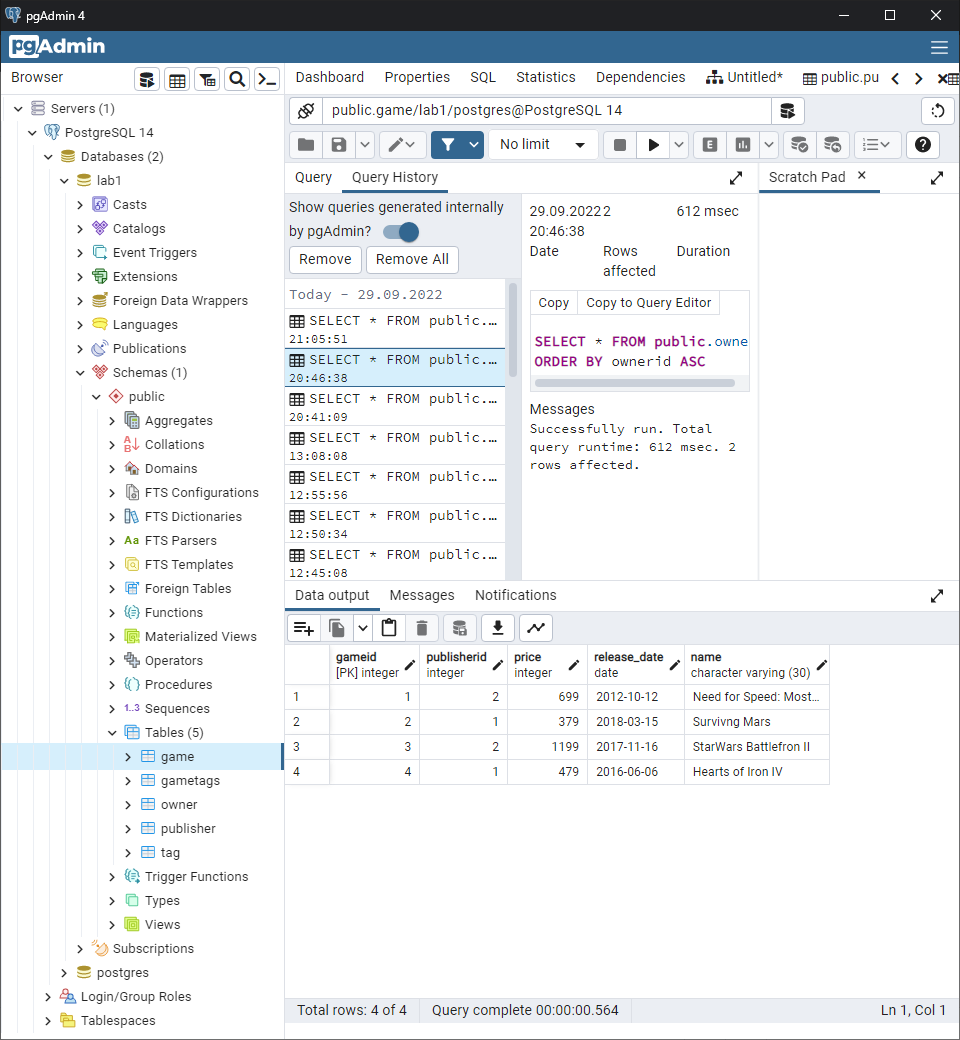


**Game:**

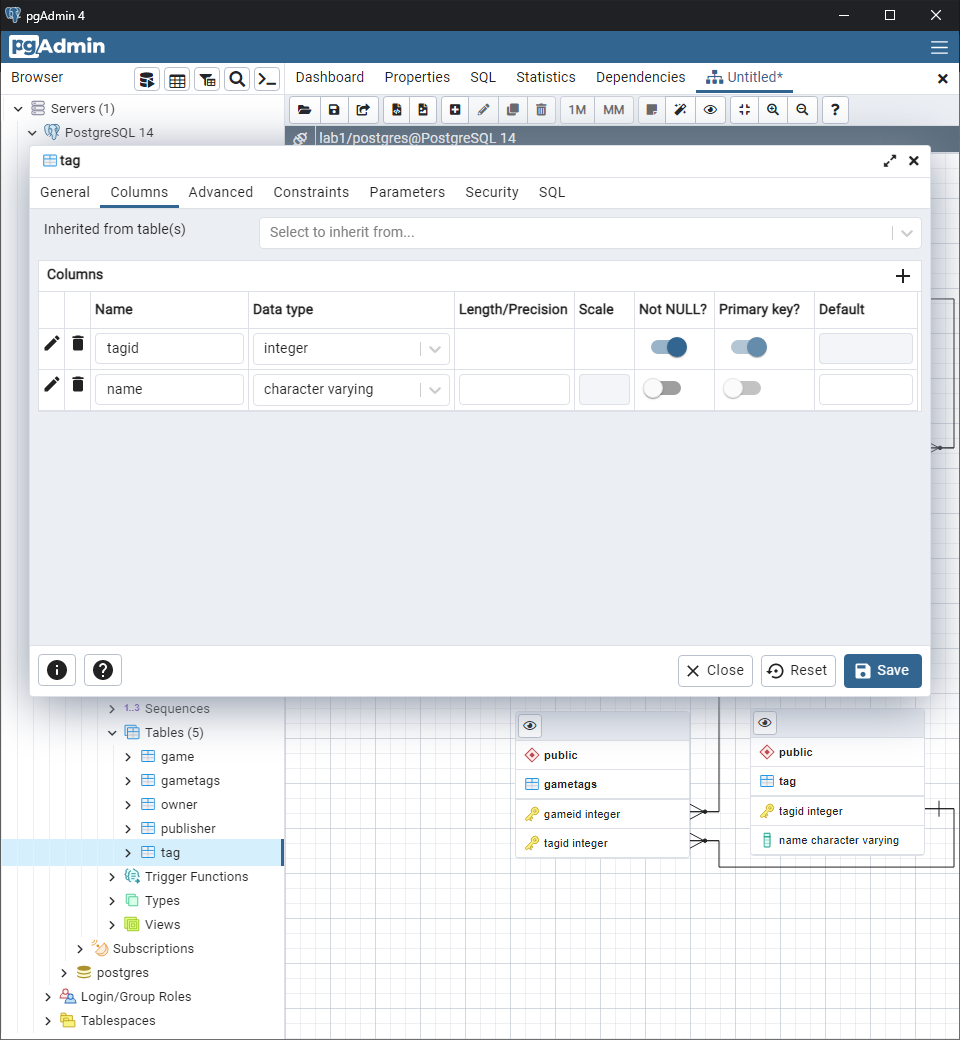


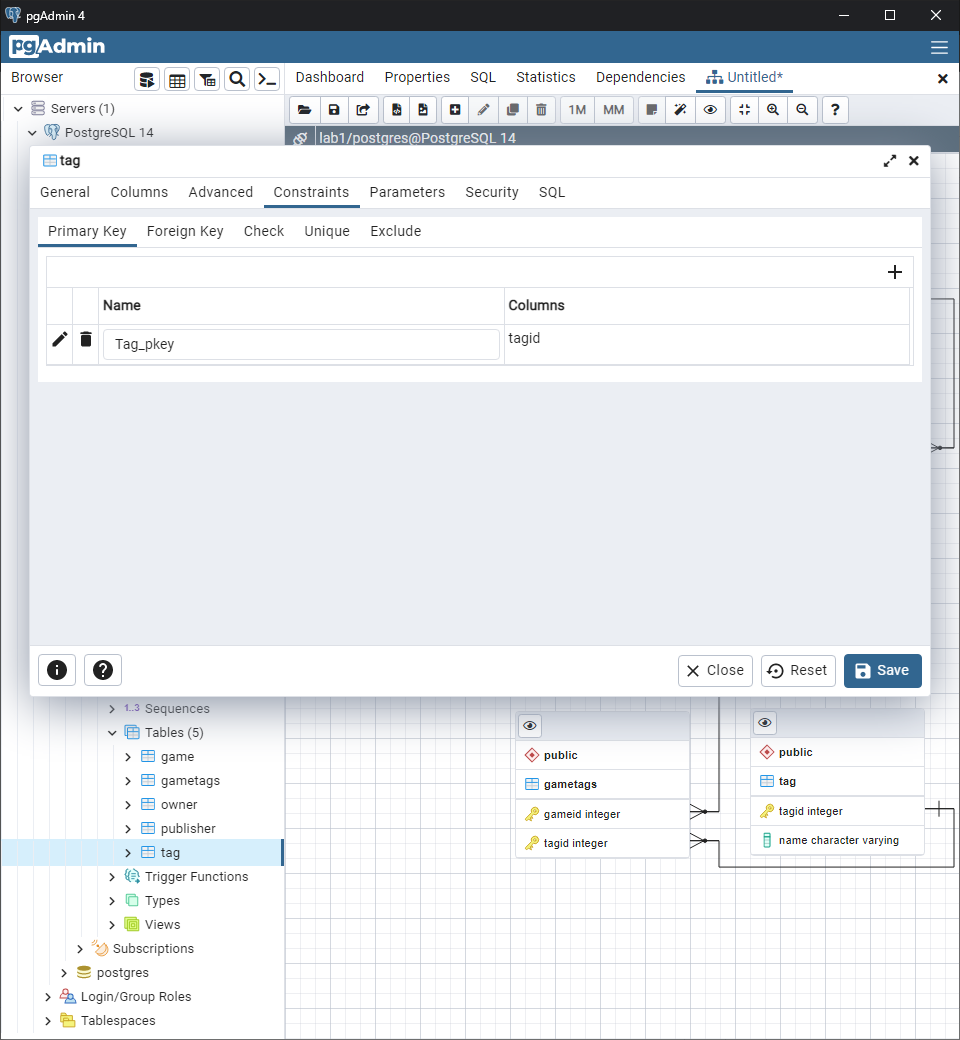


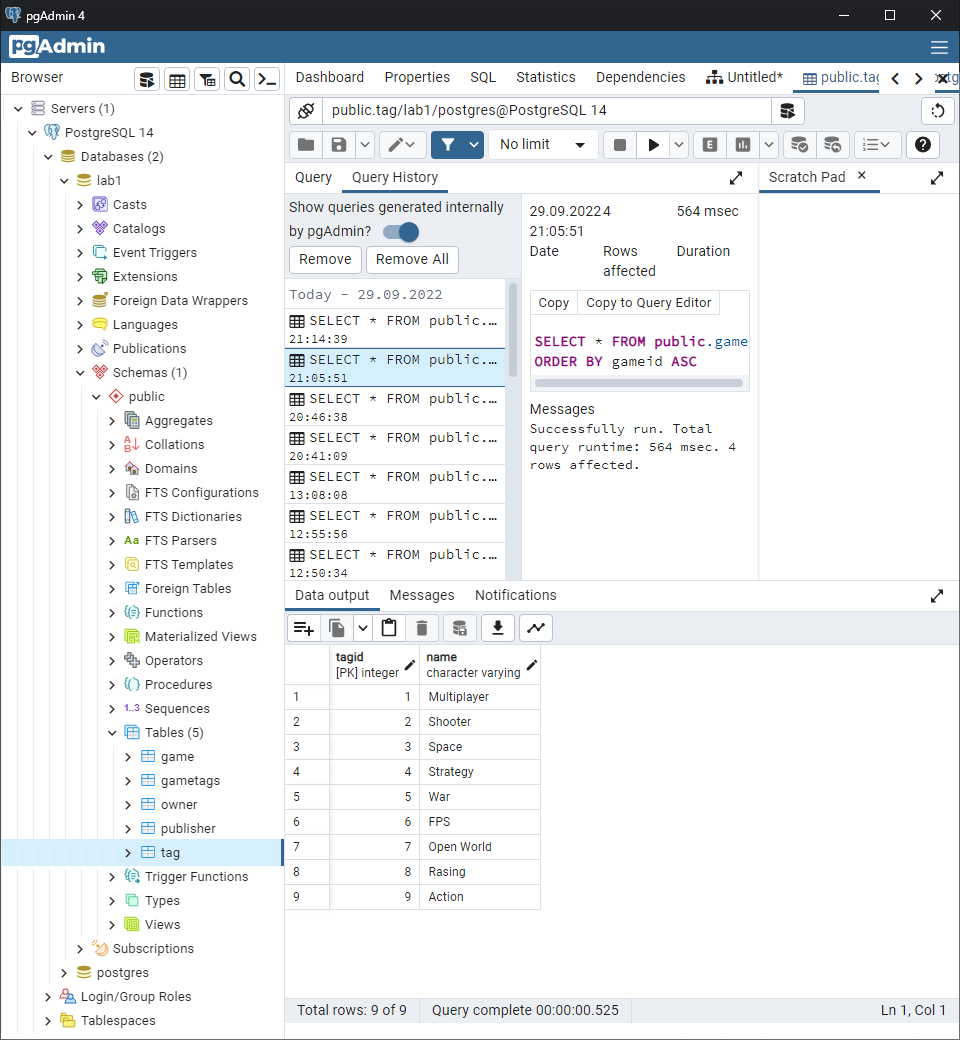




**Tag:**







**Gametags:**

