



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

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

**Лабораторна робота №1
«Ознайомлення з базовими операціями СУБД
PostgreSQL»**

з дисципліни
«Бази даних і засоби управління»

Виконав: студент III курсу
ФПМ групи КВ-94
Анохін Костянтин
Перевірів(ла):

Київ – 2021

Метою роботи є здобуття практичних навичок створення реляційних базданих за допомогою PostgreSQL.

Завдання роботи полягає у наступному:

1. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4;
2. Провести аналіз та опис предметної галузі;
3. Розробити модель «сутність-зв'язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі»;
4. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL та внести декілька рядків даних у кожную з таблиць засобами pgAdmin 4.

Опис обраної предметної галузі:

Сервіс перегляду аніме.

Перелік сутностей з описом їх призначення:

Сутність Anime призначено для ідентифікації аніме серіалу, визначення кількості серій.

Сутність Genre призначено для визначення жанру аніме серіалів.

Сутність User призначено для ідентифікації користувача та надавання йому змоги дивитись та залишати відгуки до аніме.

Сутність Review призначено для залишання відгуків від користувачів до аніме серіалів.

Графічний файл розробленої моделі «сутність-зв'язок»:

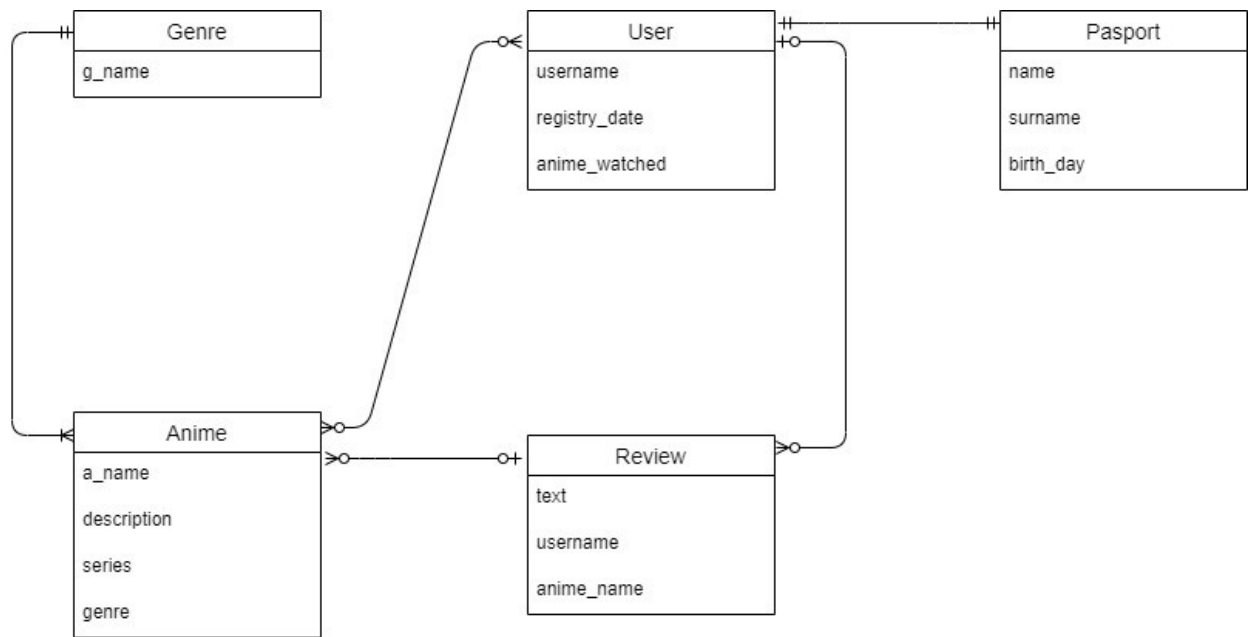


Рисунок 1 – Концептуальна модель

Назва нотації: нотація “Пташиної лапки (Crow’s foot)”

Опис процесу перетворення:

Зв’язок між сутностями Anime-User зумовив появу додаткової сутності **watched**.

Зв’язок між сутностями User-Pasport зумовив появу додаткової сутності **user_pasport**.

Схема бази даних у графічному вигляді:

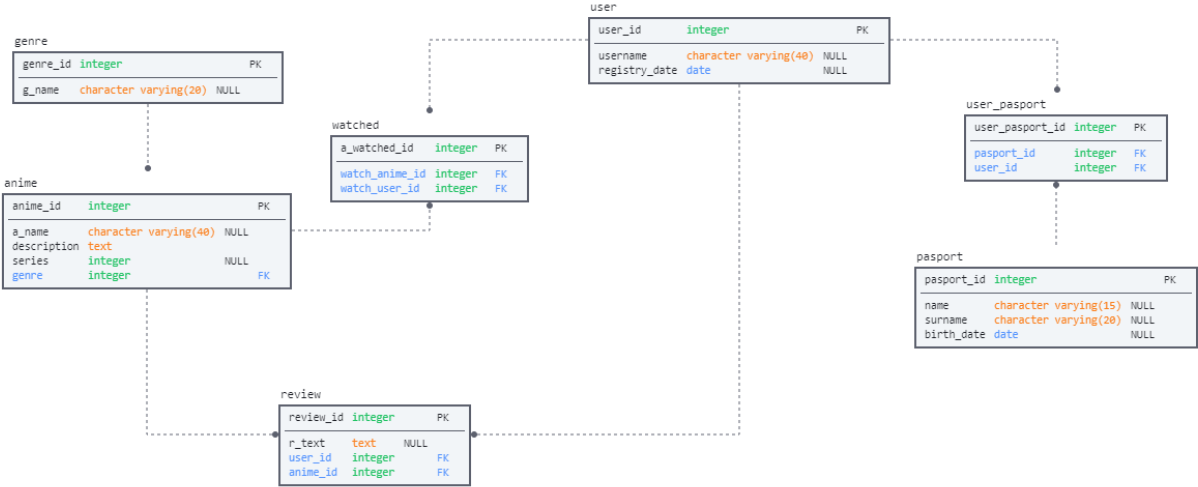


Рисунок 2 – Логічна модель

Структура БД «Сервіс перегляду аніме серіалів»

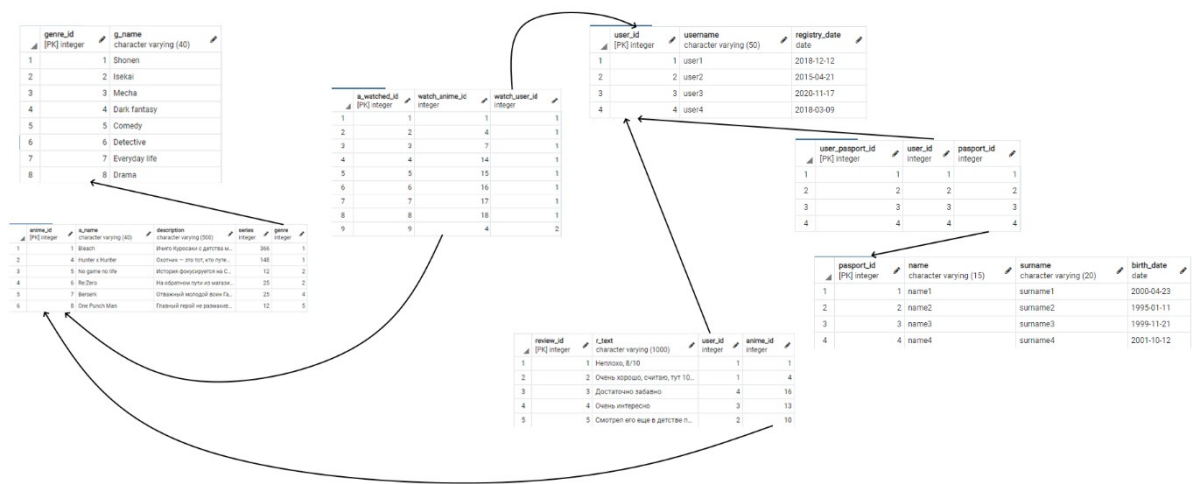


Рисунок 3 - структура БД «Сервіс перегляду аніме серіалів»

Опис структури БД «Сервіс перегляду аніме серіалів»

ВІДНОШЕННЯ	АТРИБУТ	ТИП
Genre Вміщує інформацію про жанр	genre_id(PK) – унікальний ID жанру в БД g_name – назва жанру	Числовий Текст(20)
Anime Вміщує інформацію про аніме	anime_id(PK) – унікальний ID аніме в БД a_name – назва аніме description – опис аніме series – кількість серій аніме genre(FK) – жанр даного аніме	Числовий Текстовий(40) Текстовий(1000) Числовий Числовий
User Вміщує інформацію про користувача	user_id(PK) – унікальний ID користувача в БД username – нікнейм користувача registry_date - дата реєстрації користувача user_pasport_id(FK) – ID паспорта користувача	Числовий Текстовий(40) Дата Числовий
Watched Вміщує інформацію про переглянуті користувачами аніме	a_watched_id(PK) – унікальний ID перегляду watch_anime_id(FK) – ID переглянутого аніме watch_user_id(FK) – ID користувача, що переглянув аніме	Числовий Числовий Числовий
Review Вміщує інформацію про відгуки до аніме	review_id(PK) – унікальний ID відгуку r_text – текст відгуку user_id(FK) – ID користувача, що залишив відгук anime_id(FK) – ID аніме, до якого залишили відгук	Числовий Текстовий(1000) Числовий Числовий
Pasport Вміщує інформацію про паспорт користувача	passport_id(PK) – унікальний ID паспорта name – ім'я користувача surname – прізвище користувача birth_date – дата народження користувача	Числовий Текстовий(15) Текстовий(20) Дата
User_pasport Зв'язує таблиці User та Pasport	user_pasport_id(PK) – ідентифікатор запису passport_id(FK) – ідентифікатор паспорта user_id(FK) – ідентифікатор користувача	Числовий Числовий Числовий

Пояснення щодо відповідності схеми бази даних нормальним формам:

Схеми бази даних відповідають вимогам 1НФ тому що дані в схемі атомарні тобто лише 1 елемент в кожній комірці.

Схеми бази даних відповідають вимогам 2НФ тому що відповідають вимогам 1НФ та не мають потенціальних ключів, які складаються з декількох атрибутів.

Схеми бази даних відповідають вимогам 3НФ тому що відповідають 2НФ та відсутні транзитивні функціональні залежності неключових атрибутів від ключових.

Таблица anime:

Servers (1)

PostgreSQL 12

Databases (2)

anime

Casts

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Schemas (1)

public

Collations

Domains

FTS Configurations

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Procedures

1.3 Sequences

Tables (5)

anime

genre

review

user

watched

</

Таблица genre:

Servers (1)

- PostgreSQL 12
 - Databases (2)
 - anime
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Schemas (1)
 - public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Procedures
 - Sequences
 - Tables (5)
 - anime
 - genre
 - review
 - user
 - watched

```
1 -- Table: public.genre
2
3 -- DROP TABLE public.genre;
4
5 CREATE TABLE public.genre
6 (
7     genre_id integer NOT NULL DEFAULT nextval('genre_genre_id_seq'::regclass),
8     g_name character varying(40) COLLATE pg_catalog."default" NOT NULL,
9     CONSTRAINT genre_pkey PRIMARY KEY (genre_id)
10 )
11
12 TABLESPACE pg_default;
13
14 ALTER TABLE public.genre
15     OWNER to postgres;
```

- FTS Parsers
- FTS Templates
- Foreign Tables
- Functions
- Materialized Views
- Procedures
- Sequences
- Tables (5)
 - anime
 - genre
 - review
 - user
 - watched
- Trigger Functions

Data Output		Explain	Messages	Notifications
	genre_id [PK] integer		g_name character varying (40)	
1		1	Shonen	
2		2	Isekai	
3		3	Mecha	
4		4	Dark fantasy	
5		5	Comedy	
6		6	Detective	
7		7	Everyday life	
8		8	Drama	

Таблица review:

Servers (1)

- PostgreSQL 12
 - Databases (2)
 - anime
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Schemas (1)
 - public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Procedures
 - Sequences
 - Tables (5)
 - anime
 - genre
 - review
 - user
 - watched
 - Trigger Functions

```
1 -- Table: public.review
2
3 -- DROP TABLE public.review;
4
5 CREATE TABLE public.review
6 (
7     review_id integer NOT NULL DEFAULT nextval('review_review_id_seq'::regclass),
8     r_text character varying(1000) COLLATE pg_catalog."default" NOT NULL,
9     user_id integer NOT NULL,
10    anime_id integer NOT NULL,
11    CONSTRAINT review_pkey PRIMARY KEY (review_id),
12    CONSTRAINT fk_anime FOREIGN KEY (anime_id)
13        REFERENCES public.anime (anime_id) MATCH SIMPLE
14        ON UPDATE NO ACTION
15        ON DELETE NO ACTION,
16    CONSTRAINT fk_user FOREIGN KEY (user_id)
17        REFERENCES public."user" (user_id) MATCH SIMPLE
18        ON UPDATE NO ACTION
19        ON DELETE NO ACTION
20 )
21
22 TABLESPACE pg_default;
23
24 ALTER TABLE public.review
25     OWNER to postgres;
```

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Procedures

Sequences

Tables (5)

- anime
- genre
- review
- user
- watched

Data Output	Explain	Messages	Notifications
review_id [PK] integer	r_text character varying (1000)	user_id integer	anime_id integer
1	1 Неплохо, 8/10	1	1
2	2 Очень хорошо, считаю, тут 10...	1	4
3	3 Достаточно забавно	4	16
4	4 Очень интересно	3	13
5	5 Смотрел его еще в детстве п...	2	10

Таблица user:

Browser

Servers (1)

PostgreSQL 12

Databases (2)

anime

Casts

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Schemas (1)

public

Collations

Domains

FTS Configurations

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Procedures

Sequences

Tables (7)

anime

genre

passport

review

user

user_pasport

watched

Dashboard

Properties

SQL

Statistics

Dependencies

Dependents

public.pasport/...

1

-- Table: public.user

2

3

-- DROP TABLE public."user";

4

5

CREATE TABLE public."user"

6

(

7

user_id integer NOT NULL DEFAULT nextval('user_user_id_seq'::regclass),

8

username character varying(50) COLLATE pg_catalog."default" NOT NULL,

9

registry_date date NOT NULL,

10

CONSTRAINT user_pkey PRIMARY KEY (user_id)

11

)

12

13

TABLESPACE pg_default;

14

15

ALTER TABLE public."user"

16

OWNER to postgres;

Tables (7)

anime

genre

passport

review

user

user_pasport

watched

Data Output

Explain

Messages

Notifications

	user_id [PK] integer	username character varying (50)	registry_date date
1	1	user1	2018-12-12
2	2	user2	2015-04-21
3	3	user3	2020-11-17
4	4	user4	2018-03-09

Таблица watched:

Browser

Servers (1)

- PostgreSQL 12
 - Databases (2)
 - anime
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Schemas (1)
 - public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Procedures
 - Sequences
 - Tables (5)
 - anime
 - genre
 - review
 - user
 - watched

Dashboard

Properties

SQL

Statistics

Dependencies

Dependents

public.anime/a...

public.user/ani...

```
1 -- Table: public.watched
2
3 -- DROP TABLE public.watched;
4
5 CREATE TABLE public.watched
6 (
7     a_watched_id integer NOT NULL DEFAULT nextval('watched_a_watched_id_seq'::regclass),
8     watch_anime_id integer NOT NULL,
9     watch_user_id integer NOT NULL,
10    CONSTRAINT watched_pkey PRIMARY KEY (a_watched_id),
11    CONSTRAINT fk_anime FOREIGN KEY (watch_anime_id)
12        REFERENCES public.anime (anime_id) MATCH SIMPLE
13        ON UPDATE NO ACTION
14        ON DELETE NO ACTION,
15    CONSTRAINT fk_user FOREIGN KEY (watch_user_id)
16        REFERENCES public."user" (user_id) MATCH SIMPLE
17        ON UPDATE NO ACTION
18        ON DELETE NO ACTION
19 )
20
21 TABLESPACE pg_default;
22
23 ALTER TABLE public.watched
24     OWNER to postgres;
```

EXTENSIONS

Foreign Data Wrappers

Languages

Schemas (1)

- public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Procedures
 - Sequences
 - Tables (5)
 - anime
 - genre
 - review
 - user
 - watched
 - Trigger Functions
 - Types
 - Views





























Data Output

Explain

Messages

Notifications

	a_watched_id [PK] integer	watch_anime_id integer	watch_user_id integer
1	1	1	1
2	2	4	1
3	3	7	1
4	4	14	1
5	5	15	1
6	6	16	1
7	7	17	1
8	8	18	1
9	9	4	2
10	10	5	2
11	11	6	2
12	12	7	2
13	13	8	2
14	14	9	2
15	15	10	2
16	16	11	2
17	17	12	2
18	18	13	2
19	19	14	2

- >  Extensions
- >  Foreign Data Wrappers
- >  Languages
- ▼  Schemas (1)
 - ▼  public
 - >  Collations
 - >  Domains
 - >  FTS Configurations
 - >  FTS Dictionaries
 - >  FTS Parsers
 - >  FTS Templates
 - >  Foreign Tables
 - >  Functions
 - >  Materialized Views
 - >  Procedures
 - >  Sequences
 - ▼  Tables (5)
 - >  anime
 - >  genre
 - >  review
 - >  user
 - >  watched
 - >  Trigger Functions
 - >  Types
 - >  Views
 - >  postgres
 - >  Login/Group Roles
 - >  Tablespaces

	Data Output	Explain	Messages	Notifications
	a_watched_id [PK] integer		watch_anime_id integer	watch_user_id integer
11	11		6	2
12	12		7	2
13	13		8	2
14	14		9	2
15	15		10	2
16	16		11	2
17	17		12	2
18	18		13	2
19	19		14	2
20	20		15	2
21	21		1	3
22	22		4	3
23	23		7	3
24	24		13	3
25	25		1	4
26	26		8	4
27	27		9	4
28	28		10	4
29	29		11	4
30	30		12	4
31	31		13	4
32	32		14	4
33	33		15	4
34	34		16	4
35	35		17	4
36	36		18	4

Таблица passport:

Browser

Servers (1)

PostgreSQL 12

Databases (2)

anime

Casts

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Schemas (1)

public

Collations

Domains

FTS Configurations

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Procedures

1.3 Sequences

Tables (6)

anime

genre

passport

review

Dashboard

Properties

SQL

Statistics

Dependencies

Dependents

1

-- Table: public.passport

2

3

-- DROP TABLE public.passport;

4

5

CREATE TABLE public.passport

6

(

7

passport_id integer NOT NULL DEFAULT nextval('passport_pasport_id_seq'::regclass),

8

name character varying(15) COLLATE pg_catalog."default" NOT NULL,

9

surname character varying(20) COLLATE pg_catalog."default" NOT NULL,

10

birth_date date NOT NULL,

11

CONSTRAINT passport_pkey PRIMARY KEY (passport_id)

12

)

13

14

TABLESPACE pg_default;

15

16

ALTER TABLE public.passport

17

OWNER to postgres;

1.3 Sequences

Tables (6)

anime

genre

passport

review

user

watched

Data Output

Explain

Messages

Notifications

	passport_id [PK] integer	name character varying (15)	surname character varying (20)	birth_date date
1	1	name1	surname1	2000-04-23
2	2	name2	surname2	1995-01-11
3	3	name3	surname3	1999-11-21
4	4	name4	surname4	2001-10-12

Таблица user_pasport:

Browsers (1)

- PostgreSQL 12
 - Databases (2)
 - anime
 - Cast
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Schemas (1)
 - public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Procedures
 - Sequences
 - Tables (7)
 - anime
 - genre
 - passport
 - review
 - user
 - user_pasport
 - watched

Dashboard

Properties

SQL

Statistics

Dependencies

Dependents

public.user_pas...

```
1 -- Table: public.user_pasport
2
3 -- DROP TABLE public.user_pasport;
4
5 CREATE TABLE public.user_pasport
6 (
7     user_pasport_id integer NOT NULL DEFAULT nextval('user_pasport_user_pasport_id_seq'::regclass),
8     user_id integer NOT NULL,
9     passport_id integer NOT NULL,
10    CONSTRAINT user_pasport_pkey PRIMARY KEY (user_pasport_id),
11    CONSTRAINT u_pasport UNIQUE (passport_id),
12    CONSTRAINT u_user UNIQUE (user_id),
13    CONSTRAINT fk_pasport FOREIGN KEY (passport_id)
14        REFERENCES public.passport (passport_id) MATCH SIMPLE
15        ON UPDATE NO ACTION
16        ON DELETE NO ACTION
17        NOT VALID,
18    CONSTRAINT fk_user FOREIGN KEY (user_id)
19        REFERENCES public."user" (user_id) MATCH SIMPLE
20        ON UPDATE NO ACTION
21        ON DELETE NO ACTION
22        NOT VALID
23 )
24
25 TABLESPACE pg_default;
26
27 ALTER TABLE public.user_pasport
28     OWNER to postgres;
```

Tables (7)

- anime
- genre
- passport
- review
- user
- user_pasport
- watched

	Data Output	Explain	Messages	Notifications
	user_pasport_id [PK] integer	user_id integer	passport_id integer	
1	1	1	1	
2	2	2	2	
3	3	3	3	
4	4	4	4	