



**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. Ігоря СІКОРСЬКОГО»
ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ
Кафедра системного програмування і спеціалізованих комп'ютерних
систем**

**Лабораторна робота №1
«Проектування бази даних та ознайомлення з базовими операціями
СУБД PostgreSQL»**

**з дисципліни
«Бази даних і засоби управління»**

**Виконав: студент III курсу
ФПМ групи КВ-93
Піскун Андрій
Перевірів:
Павловський В. І.**

Київ – 2021

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 кожен з таблиць засобами pgAdmin 4.

Перелік сутностей з описом їх призначення

Кінотеатр (Cinema) з атрибутами: назва, код, адреса кінотеатру. Призначена для збереження даних про кінотеатри, в яких купуються квитки.

Фільм (Movie) з атрибутами: назва, код, рейтинг. Призначена для збереження даних про фільми, що показують.

Сеанс (Session) з атрибутами: час, код, вартість. Призначена для збереження даних про сеанси фільмів.

Зал (Hall) з атрибутами: код, розмір екрану, кількість місць. Призначена для збереження даних про зали в кінотеатрі.

Місце (Seat) з атрибутами: ряд, код, місце, зайнятість місця. Призначена для збереження даних про місце в залі.

Опис зв'язків

Один кінотеатр може транслювати багато фільмів, фільм може мати декілька сеансів, кінотеатр може мати багато залів, в залі може проводитись декілька сеансів, як і зал має багато місць, тому зв'язки 1:N.

Модель “сутність-зв’язок” предметної галузі кінотеатр

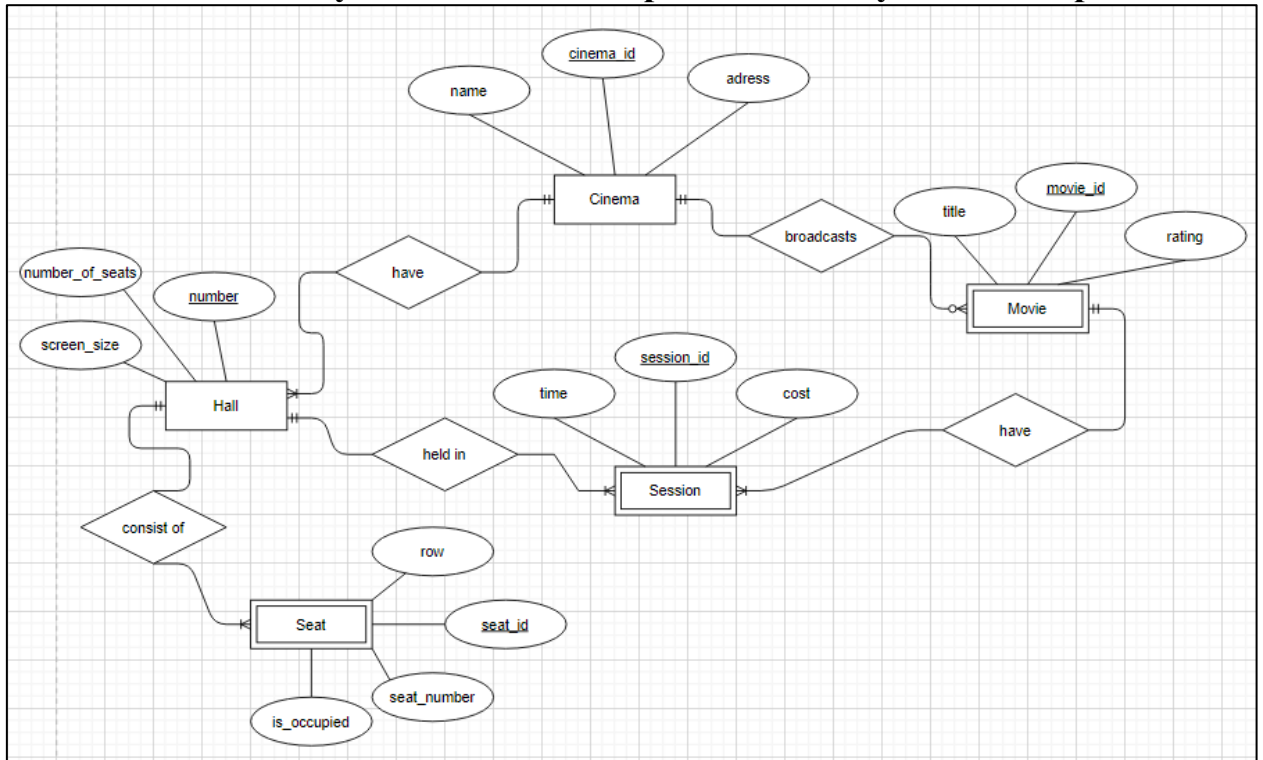


Рисунок 1. ER-діаграма, побудована за нотацією Чена
Побудовано за допомогою веб додатку draw.io

Перетворення моделі у схему баз даних

Сутність ‘Cinema’ перетворено у таблицю ‘Cinema’.

Сутність ‘Movie’ перетворено у таблицю ‘Movie’, а зв’язок 1:N із сутністю ‘Cinema’ зумовив появу зовнішнього ключа cinema_id.

Сутність ‘Hall’ перетворено у таблицю ‘Hall’, а зв’язок 1:N із сутністю ‘Cinema’ зумовив появу зовнішнього ключа cinema_id.

Сутність ‘Seat’ перетворено у таблицю ‘Seat’, а зв’язок 1:N із сутністю ‘Hall’ зумовив появу зовнішнього ключа number.

Сутність ‘Session’ перетворено у таблицю ‘Session’, зв’язок 1:N із сутністю ‘Movie’ зумовив появу зовнішнього ключа movie_id, зв’язок 1:N із сутністю ‘Hall’ зумовив появу зовнішнього ключа number.

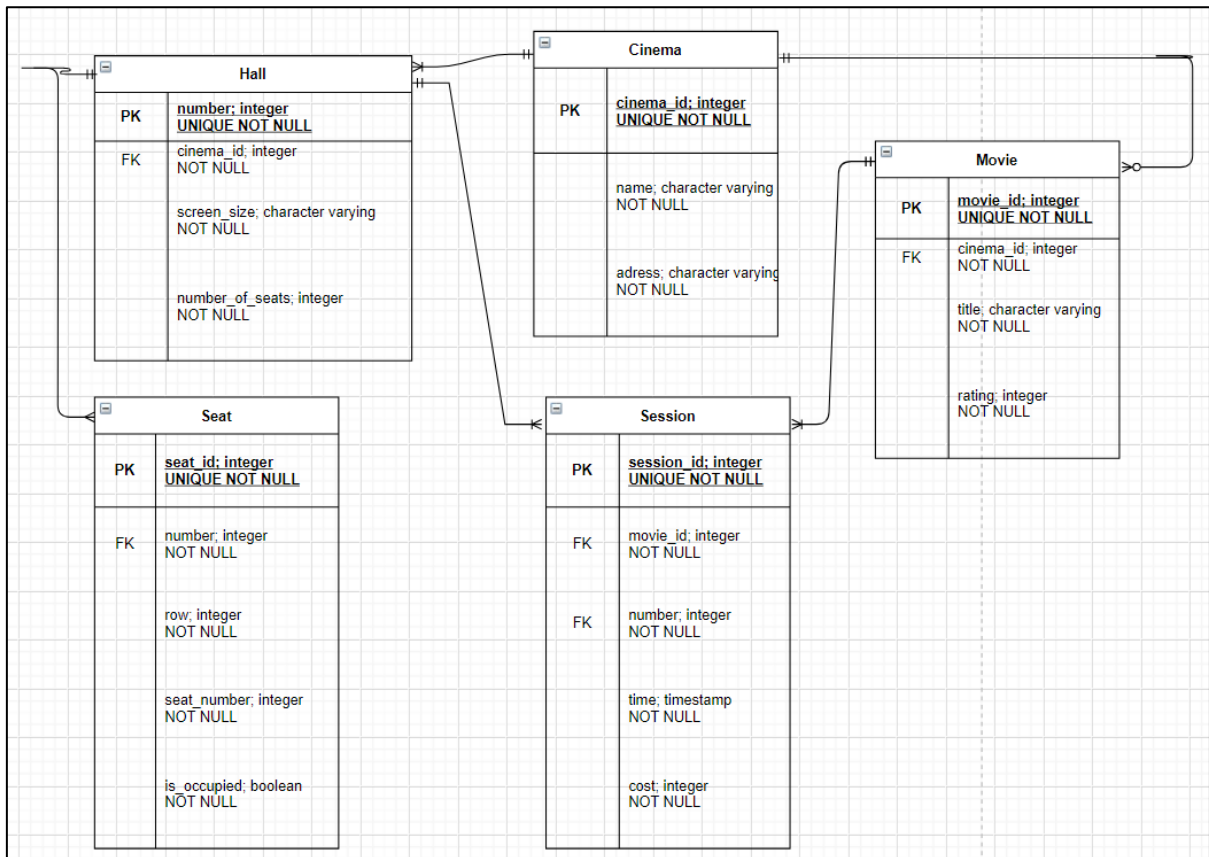


Рисунок 2. Схема бази даних у таблицях

Побудовано за допомогою веб додатку draw.io

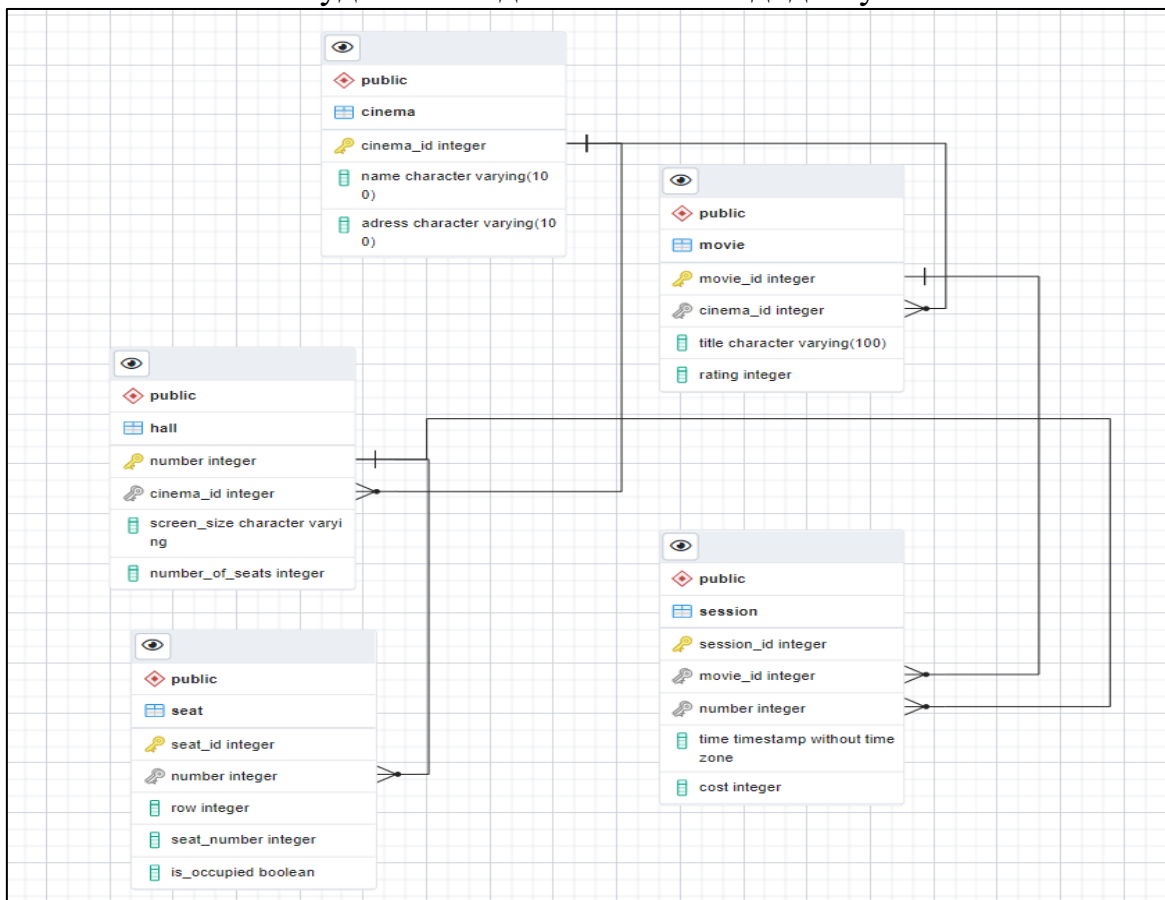


Рисунок 3. Схема бази даних у pgAdmin 4

Опис структури БД

Cinema – містить дані про кінотеатр

- cinema_id (числовий) – унікальний ідентифікатор кінотеатру
- name (текстовий) – назва кінотеатру
- address (текстовий) – адреса кінотеатру

Movie – містить дані про фільм

- movie_id (числовий) – унікальний ідентифікатор фільму
- cinema_id (числовий) – ідентифікатор кінотеатру, в якому транслюється фільм
- title (текстовий) – назва фільму
- rating (числовий) – рейтинг фільму

Hall – містить дані про зал

- number (числовий) – унікальний ідентифікатор залу, а також його номер
- cinema_id (числовий) – ідентифікатор кінотеатру, в якому знаходиться зал
- screen_size (текстовий) – розмір екрану у залі
- number_of_seats (числовий) – кількість місць у залі

Seat – містить дані про місце в залі

- seat_id (числовий) – унікальний ідентифікатор місця
- number (числовий) – ідентифікатор залу, в якому знаходиться місце
- row (числовий) – номер ряду
- seat_number (числовий) – номер місця в ряду
- is_occupied (двійковий) – інформація про те, чи зайняте місце

Session – містить дані про сеанс фільму

- session_id (числовий) – унікальний ідентифікатор сеансу
- movie_id (числовий) – ідентифікатор фільму, сеанс якого проводиться
- number (числовий) – ідентифікатор залу, в якому проводиться сеанс
- time (мітка часу) – час, о котрій проводиться сеанс
- cost (числовий) – вартість сеансу

Функціональні залежності

Cinema:

cinema_id -> name, address

cinema_id -> name (назва пов'язана з ідентифікатором)

cinema_id -> address (адреса пов'язана з ідентифікатором)

name -> address

address -> name

cinema_id -> name -> address (включає ключ)
cinema_id -> address -> name (включає ключ)

Movie:

movie_id -> title, rating
movie_id -> title
movie_id -> rating
title -> rating
movie_id -> title -> rating (включає ключ)

Hall:

number -> screen_size, number_of_seats
number -> screen_size
number -> number_of_seats

Seat:

seat_id -> row, seat_number, is_occupied
seat_id -> row
seat_id -> seat_number
seat_id -> is_occupied
seat_number -> is_occupied
seat_id -> seat_number -> is_occupied (includes PK)

Session:

session_id -> time, cost
session_id -> time
session_id -> cost

Схема відповідає 1НФ, тому що:

- Атрибути мають унікальні значення
- В кожному стовпці зберігаються дані одного типу.
- В кожній комірці зберігається атомарне (скалярне) значення.

Схема відповідає 2НФ, тому що:

- Відповідає 1НФ
- Кожен не ключовий атрибут функціонально залежний від ключового

Схема відповідає 3НФ, тому що:

- Відповідає 3НФ
- Залежності в одній таблиці тільки від основного ключа

SQL-текст опису БД

BEGIN;

```
CREATE TABLE IF NOT EXISTS public.cinema  
(  
    cinema_id integer NOT NULL,
```

```
name character varying(100) NOT NULL,  
adress character varying(100) NOT NULL,  
PRIMARY KEY (cinema_id)  
);
```

```
CREATE TABLE IF NOT EXISTS public.hall  
(  
    "number" integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1  
START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),  
    cinema_id integer NOT NULL,  
    screen_size character varying NOT NULL,  
    number_of_seats integer NOT NULL,  
    PRIMARY KEY ("number")  
);
```

```
CREATE TABLE IF NOT EXISTS public.movie  
(  
    movie_id integer NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT  
1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),  
    cinema_id integer NOT NULL,  
    title character varying(100) NOT NULL,  
    rating integer NOT NULL,  
    PRIMARY KEY (movie_id)  
);
```

```
CREATE TABLE IF NOT EXISTS public.seat  
(  
    seat_id integer NOT NULL,  
    "number" integer NOT NULL,  
    "row" integer NOT NULL,  
    seat_number integer NOT NULL,  
    is_occupied boolean NOT NULL,  
    PRIMARY KEY (seat_id)  
);
```

```
CREATE TABLE IF NOT EXISTS public.session  
(  
    session_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1  
START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),  
    movie_id integer NOT NULL,  
    "number" integer NOT NULL,  
    "time" timestamp without time zone NOT NULL,  
    cost integer NOT NULL,  
    PRIMARY KEY (session_id)  
);
```

```
ALTER TABLE public.hall  
    ADD FOREIGN KEY (cinema_id)  
    REFERENCES public.cinema (cinema_id)  
    NOT VALID;
```

```
ALTER TABLE public.movie
  ADD FOREIGN KEY (cinema_id)
  REFERENCES public.cinema (cinema_id)
  NOT VALID;
```

```
ALTER TABLE public.seat
  ADD FOREIGN KEY ("number")
  REFERENCES public.hall ("number")
  NOT VALID;
```

```
ALTER TABLE public.session
  ADD FOREIGN KEY (movie_id)
  REFERENCES public.movie (movie_id)
  NOT VALID;
```

```
ALTER TABLE public.session
  ADD FOREIGN KEY ("number")
  REFERENCES public.hall ("number")
  NOT VALID;
```

```
END;
```

Копії екрану з pgAdmin4

Cinema

The screenshot displays the pgAdmin4 interface for the 'cinema' table. The left sidebar shows the database structure, including the 'cinema' table with columns 'cinema_id', 'name', and 'adress'. The main pane shows the 'Data Output' tab with a table of data. A modal window for editing the table is also visible, showing the column definitions.

cinema_id [PK] integer	name character varying (100)	adress character varying (100)
1	Poposha	Lykyshkina 9
2	Kokosha	Voluna 222
3	Friend	Bogdana 21

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
cinema_id	integer			Yes	Yes
name	character varying	100		Yes	No
adress	character varying	100		Yes	No

Movie

movie

Columns (4)

movie_id

cinema_id

title

rating

Constraints (2)

cinema_id

movie_pkey

Indexes

RLS Policies

Rules

1 SELECT * FROM public.movie

2 ORDER BY movie_id ASC

Data Output

Explain

Messages

Notifications

	movie_id	cinema_id	title	rating
	[PK] integer	integer	character varying (100)	integer
1	0	0	Harry Potter	9
2	1	0	Operation 'I'	10
3	2	1	Ivan Vasylovich changes profession	8
4	3	2	Love and pigeons	5

movie

General

Columns

Advanced

Constraints

Parameters

Security

SQL

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
	movie_id	integer			<div>Yes</div>	<div>Yes</div>
	cinema_id	integer			<div>Yes</div>	<div>No</div>
	title	character varying	100		<div>Yes</div>	<div>No</div>
	rating	integer			<div>Yes</div>	<div>No</div>

i

?

Cancel

Reset

Save

movie

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

	Name	Columns	Referenced Table
	cinema_id	(cinema_id) -> (cinema_id)	public.cinema

Hall

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Columns' tab for the 'hall' table is selected, displaying the column definitions: 'number' (integer, PK), 'cinema_id' (integer), 'screen_size' (character varying), and 'number_of_seats' (integer). The 'Data Output' tab is also visible, showing the first four rows of data. The 'Columns' tab is active, showing the column names, data types, and the 'Yes/No' status for each column.

	number [PK] integer	cinema_id integer	screen_size character varying	number_of_seats integer
1	1	1	240x420	40
2	2	0	120x920	10
3	3	2	190x900	90
4	4	0	1920x1080	2

hall

General

Columns

Advanced

Constraints

Parameters

Security

SQL



Primary Key

Foreign Key

Check

Unique

Exclude

		Name	Columns	Referenced Table
		<input type="text" value="cinema_id"/>	(cinema_id) -> (cinema_id)	public.cinema

i

?

Cancel

Reset

Save

Seat

The screenshot displays the PostgreSQL pgAdmin interface. On the left, the 'Columns (5)' list for the 'seat' table is visible, including 'seat_id', 'number', 'row', 'seat_number', and 'is_occupied'. The right pane shows the 'Data Output' tab, displaying a table with 6 rows of data. The bottom pane shows the 'Columns' tab of the 'seat' table definition dialog, where each column has a data type and a 'Yes/No' checkbox for a constraint.

seat_id	number	row	seat_number	is_occupied
1	19	1	19	true
2	22	2	2	false
3	11	1	12	false
4	212	2	21	false
5	231	2	33	true
6	321	3	4	false

Column	Data Type	Constraint
seat_id	integer	Yes
number	integer	No
row	integer	No
seat_number	integer	No
is_occupied	boolean	No

seat

General

Columns

Advanced

Constraints

Parameters

Security

SQL



Primary Key



Foreign Key


Check


Unique


Exclude

		Name	Columns	Referenced Table
		<input type="text" value="number"/>	(number) -> (number)	public.hall



 Cancel

 Reset

 Save

Session

session

Columns (5)

session_id

movie_id

number

time

cost

Constraints (3)

movie_id

number

session_pkey

Indexes

```
1 SELECT * FROM public.session
2 ORDER BY session_id ASC
```

Data Output

Explain

Messages

Notifications

	session_id [PK] integer	movie_id integer	number integer	time timestamp without time zone	cost integer
1	22	1	1	12:20	180
2	41	1	2	23:30	999
3	56	2	1	10:30	100
4	99	2	2	18:00	500

session

General

Columns

Advanced

Constraints

Parameters

Security

SQL

	session_id	integer			Yes	Yes
	movie_id	integer			Yes	No
	number	integer			Yes	No
	time	timestamp without time zone			Yes	No
	cost	integer			Yes	No

i

?

Cancel

Reset

Save

session

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

Name

Columns

Referenced Table

		movie_id	(movie_id) -> (movie_id)	public.movie
		number	(number) -> (number)	public.hall

i

?

Cancel

Reset

Save