A black and white drawing of a building

Description automatically generated

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”*

Виконала:

студентка ІІI курсу

групи КВ-12

Давидюк М.Ю.

Перевірив:

Павловский В. І.

Київ – 2023

**Мета:** здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

**Завдання:**

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 кожну з таблиць засобами pgAdmin 4.

**Опис предметної області**

Система обліку використання енергоресурсів у житлових будинках призначена для комплексного контролю та ефективного управління споживанням енергії в житлових приміщеннях. В даній предметній області враховуються власники будинків, їхні житлові об'єкти та використання різних енергоресурсів у цих будівлях..

**Опис сутностей**

Для побудови бази даних обраної області, були виділені такі сутності:

1. Користувач(User)

Атрибути: ідентифікатор користувача, ім’я, номер телефону.

Призначення: збереження даних щодо користувача.

1. Будинок (House)

Атрибути: ідентифікатор будинку, адреса.

Призначення: збереження даних щодо групи.

1. Енергоресурси (EnergyResource)

Атрибути: ідентифікатор енергоресурсу, назва енергоресурсу, одиниця вимірювання, ідентифікатор будинку.

Призначення: збереження даних щодо наявності енергоресурсів в певному будинку.

1. Використання (Using)

Атрибути: ідентифікатор кожного запису, ідентифікатор енергоресурсу, кількість використаного енергоресурсу, дата.

Призначення: збереження даних щодо використання енергоресурсу.

**Опис зв’язків між сутностями**

Зв’язок «Користувач» - «Будинок» є зв’язком N:M. Один користувач може використовувати енергоресурси в декількох приміщеннях, і в одному будинку енергоресурси можуть використовувати декілька осіб.

Зв’язок «Будинок» - «Енергоресурси» є зв’язком M:N. В одному будинку може використовуватися декілька енергоресурсів і один енергоресурс може використовуватись в декількох будинках.

Зв’язок «Енергоресурси» - «Використання» є зв’язком 1:N. Один енергоресурс можуть використати декілька разів, але записати дані використання енергоресурсу можна лише раз за певний час.

Графічне подання концептуальної моделі «Сутність-зв’язок» зображено на рисунку 1.

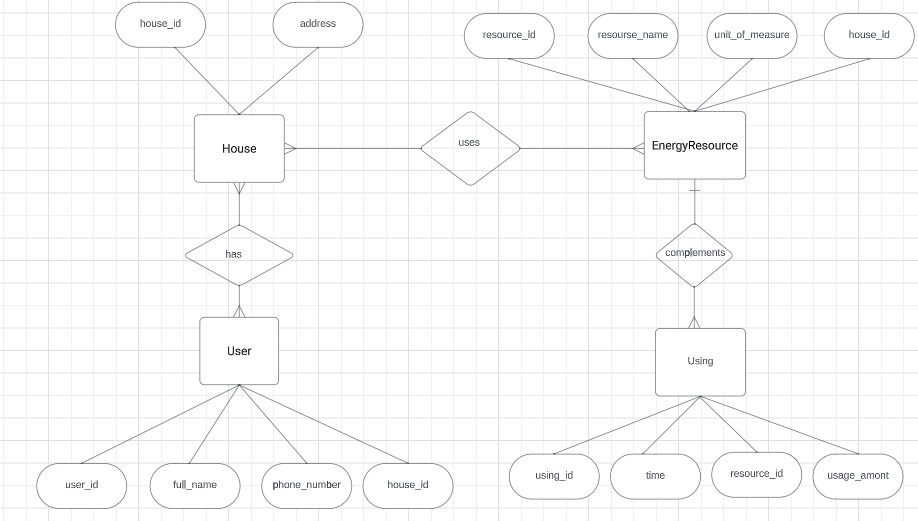


Рисунок 1 – ER-діаграма, побудована за нотацією Чена

**Перетворення концептуальної моделі у логічну модель та схему бази даних**

Сутність User перетворено в таблицю User з первинним ключем user\_id та атрибутами full\_name,phone\_number.

Сутність House перетворено в таблицю House з первинним ключем house\_id та атрибутом address.

Сутність EnergyResource перетворено в таблицю EnergyResource з первинним ключем resource \_id та атрибутами resourse\_name, unit\_of\_measure.

Сутність Using перетворено в таблицю Using з первинним ключем using\_id та атрибутами time, usage\_amont та зовнішнім ключем resource \_id.

Оскільки в логічній моделі безпосередній зв’язок N:M є неможливим, а в концептуальній моделі він існує між сутностями User i House та House та EnergyResource, то для їх реалізації було створено таблиці House\_User та House\_ EnergyResource, з первинним ключем tab1\_id і tab2\_id, та зовнішніми ключами house\_id i user\_id та house\_id і resource\_id.

Графічне подання логічної моделі «Сутність-зв’язок» зображено на рисунку 2.

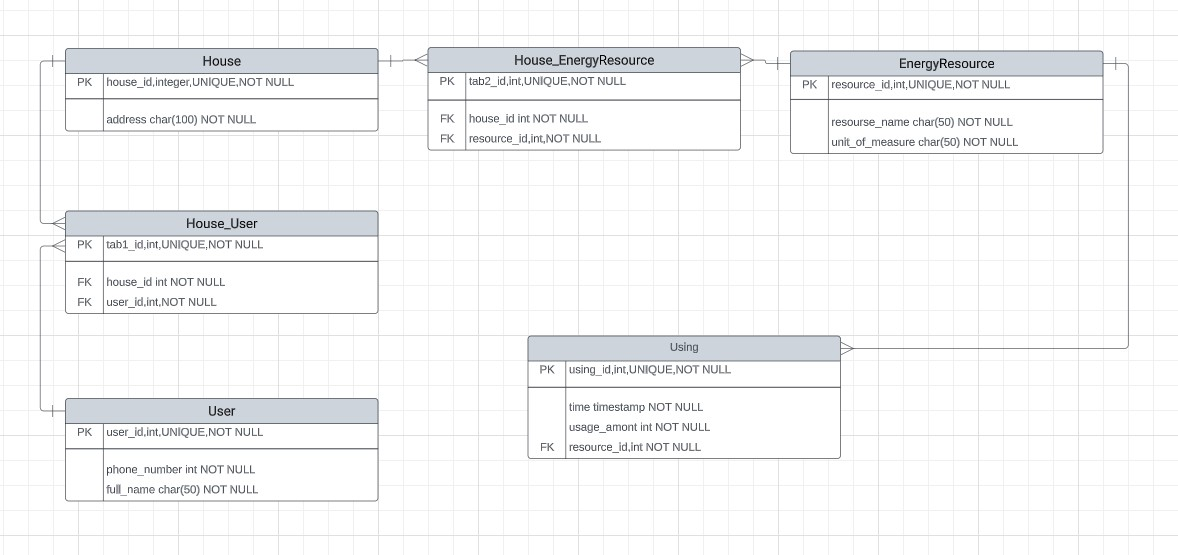


Рисунок 2 – Схема бази даних

Таблиця 1 ілюструє детальний перехід від однієї моделі до іншої.

Таблиця 1 – Опис об’єктів бази даних

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| User – містить дані про студентів | user\_id – унікальний ідентифікатор користувача | *integer* (число) |
| full\_name – ім’я користувача | *character varying* (рядок) |
| phone\_number – номер телефону | *integer* (число) |
| House– містить дані про будинк | house\_id – унікальний ідентифікатор будинку | *integer* (число) |
| address – адрес | *character varying* (рядок) |
| EnergyResource – містить дані про енергоресурси. | resource \_id – унікальний ідентифікатор енергоресурсу | *integer* (число) |
| resourse\_name – назва енергоресурсу | *character varying* (рядок) |
| unit\_of\_measure – одиниця вимірювання | *character varying* (рядок) |
| Using – містить дані про використання енергоресурсів. | using\_id – унікальний ідентифікатор використання енергоресурсів. | *integer* (число) |
| time –дата, час | *timestamp*(дата та час) |
| usage\_amont – використано енергоресурсу. | *integer* (число) |
| resource\_id – ідентифікатор енергоресурсу | *integer* (число) |
| House\_User– містить інформацію про відповідність користувача та будинку. | *tab1\_id* – унікальний ідентифікатор відповідності | *integer* (число) |
| user\_id – ідентифікатор користувача | *integer* (число) |
| house\_id *–* ідентифікатор будинку | *integer* (число) |
| House\_EnergyResource – містить інформацію про відповідність будинку та енергоресурсу. | *tab2\_id* – унікальний ідентифікатор відповідності | *integer* (число) |
| house\_id *–* ідентифікатор будинку | *integer* (число) |
| resource\_id –ідентифікатор енергоресурсу | *integer* (число) |

**Функціональні залежності для кожної таблиці**

1. User:

- user\_id -> full\_name, phone\_number

- full\_name, phone\_number -> user\_id

2. House:

- house\_id -> address

- address -> house\_id

3. EnergyResource:

- resource \_id -> resourse\_name, unit\_of\_measure

- resourse\_name, unit\_of\_measure -> resource \_id

4. Using:

- using\_id -> time, usage\_amont, resource\_id

- time-> using\_id, usage\_amont, resource\_id

- resource\_id-> using\_id, usage\_amont, time

5. House\_User:

- tab1\_id-> house\_id, user\_id

- house\_id -> tab1\_id, user\_id

- user\_id -> tab1\_id, house\_id

6. House\_EnergyResource:

- tab2\_id -> resource \_id, house\_id

- resource \_id -> tab2\_id, house\_id

- house\_id -> tab2\_id, resource \_id

Ці функціональні залежності вказують на те, які атрибути в кожній таблиці визначаються від інших атрибутів. Це важливо для нормалізації та управління базою даних.

Транзитивні функціональні залежності виникають, коли один атрибут функціонально визначає інший через інший атрибут. Іншими словами, якщо A визначає B, а B визначає C, то ми можемо сказати, що A транзитивно визначає C. Дані таблиці не мають транзитивних функціональних залежностей.

**Відповідність схеми нормальним формам**

1. Щоб задовільнити умови 1НФ кожен атрибут в таблиці має бути атомарним, тобто:

* Кожна клітинка містить єдине значення;
* Кожен запис є унікальним.

Дана схема відповідає 1НФ.

1. Щоб схема відповідала 2НФ повинні виконуватись умови:

* Схема перебуває в 1НФ;
* Кожний неключовий атрибут функціонально залежить від цілого ключа.

У даній схемі кожна таблиця має власний унікальний ідентифікатор (ключ). Кожний атрибут у кожній таблиці залежить від цього унікального ідентифікатора. Тобто, схема також в НФ2.

1. Щоб схема відповідала 3НФ повинні виконуватись умови:

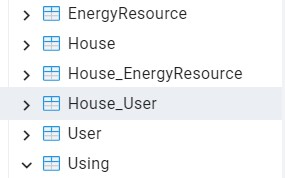
* Схема перебуває в 2НФ;
* Кожен не первинний атрибут має бути не транзитивно залежним від кожного ключа.

Оскільки дана схема в НФ2 та неключові атрибути не транзитивно залежать від інших неключових атрибутів, схема також в НФ3.

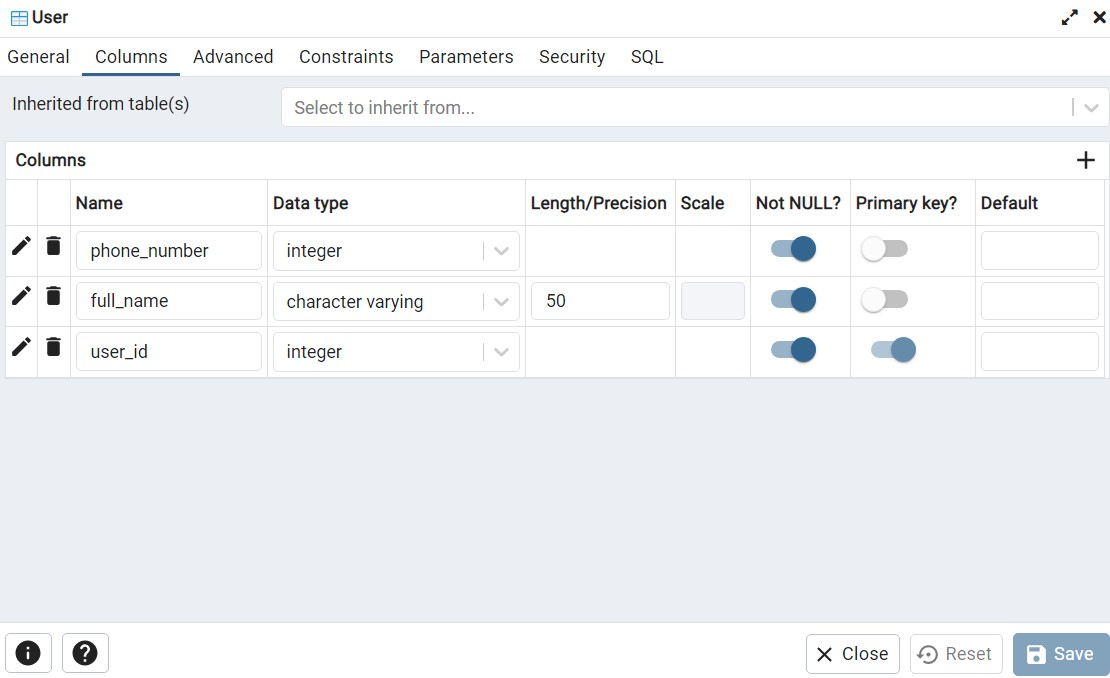
Висновок:

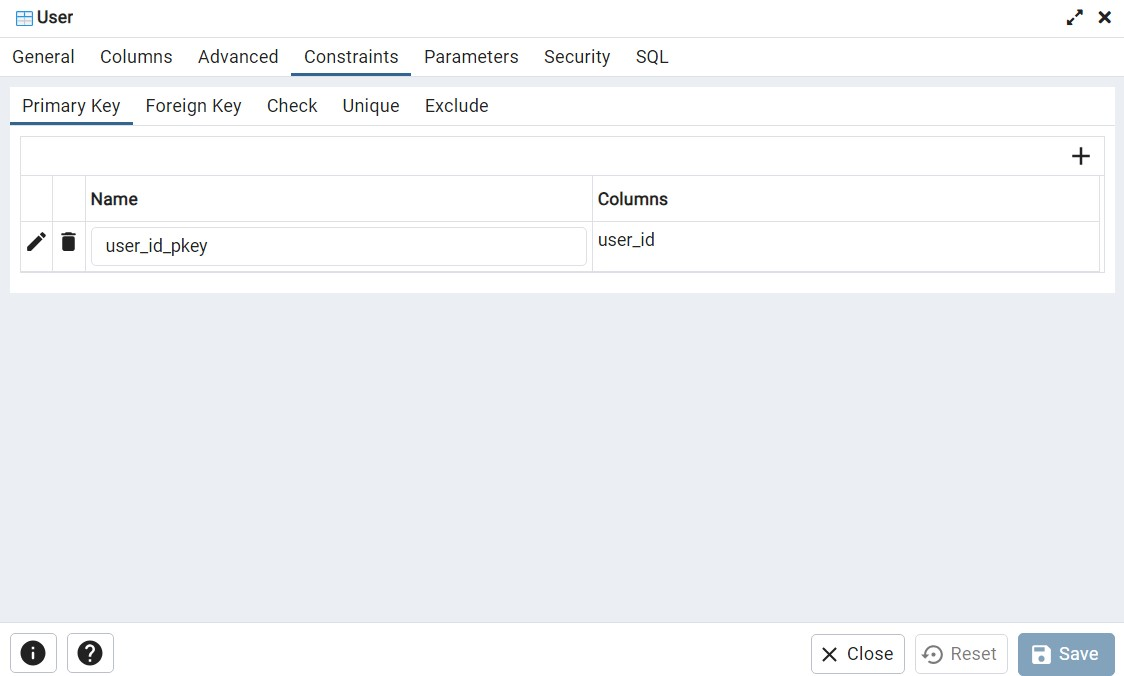
Схема бази даних відповідає нормальним формам НФ1, НФ2 та НФ3. Вона добре структурована і нормалізована, що сприяє ефективному та надійному зберіганню та обробці даних.

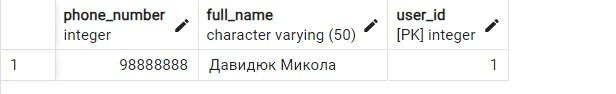
**Таблиці бази даних у pgAdmin4**



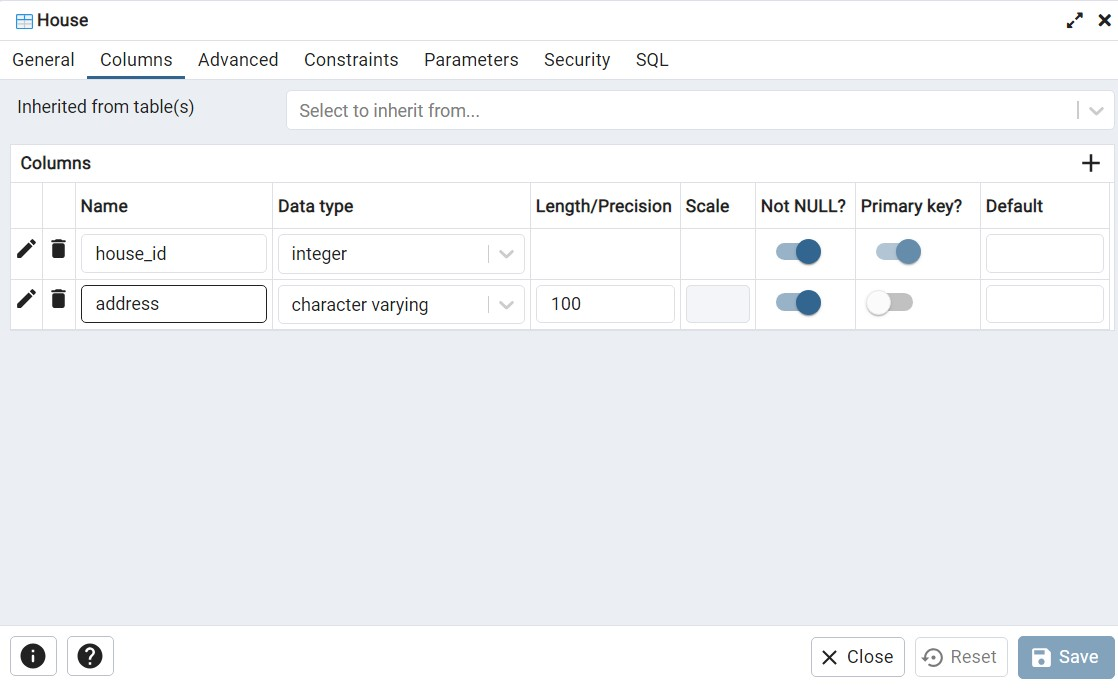
User

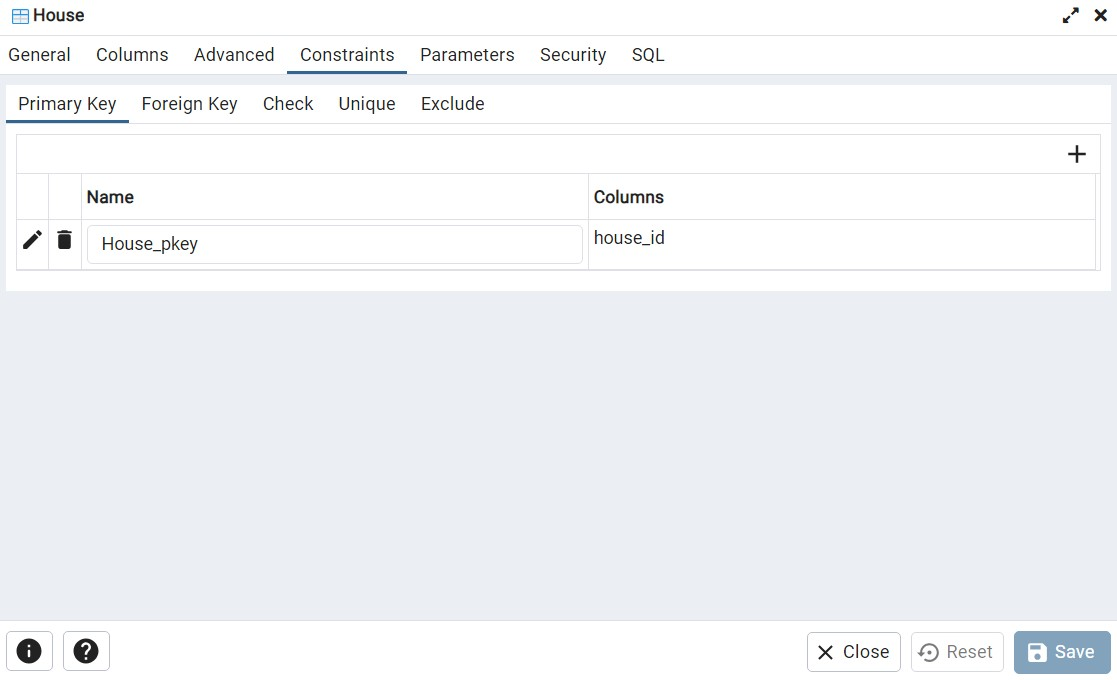


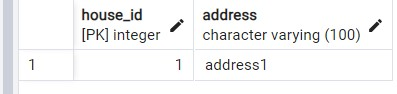




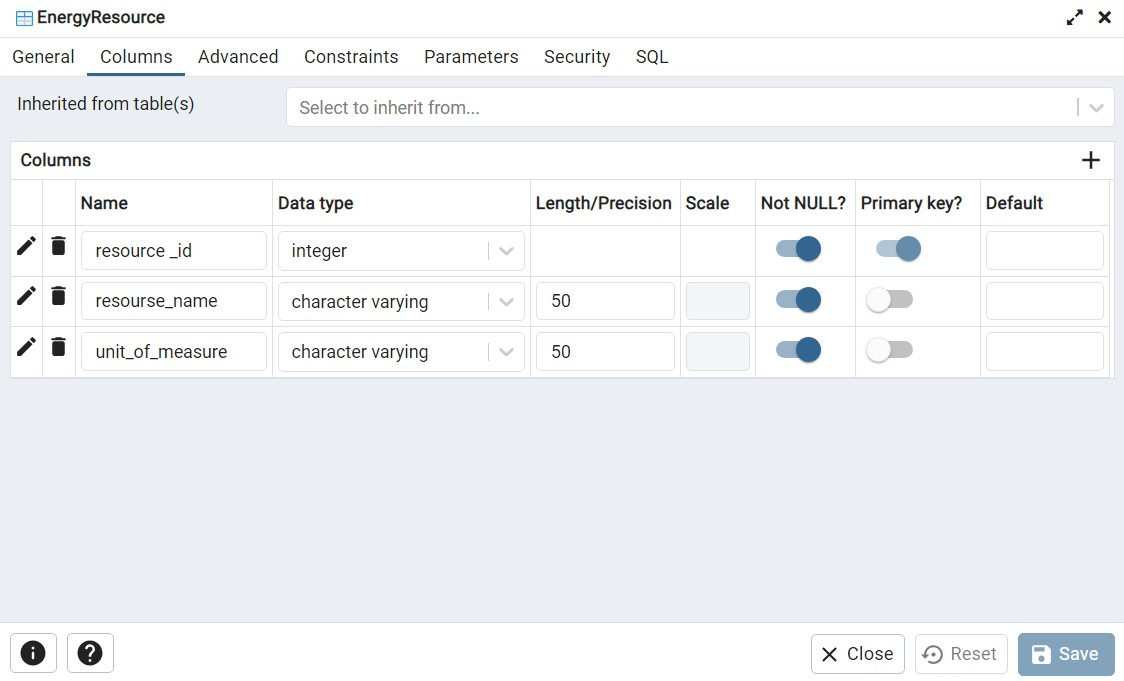
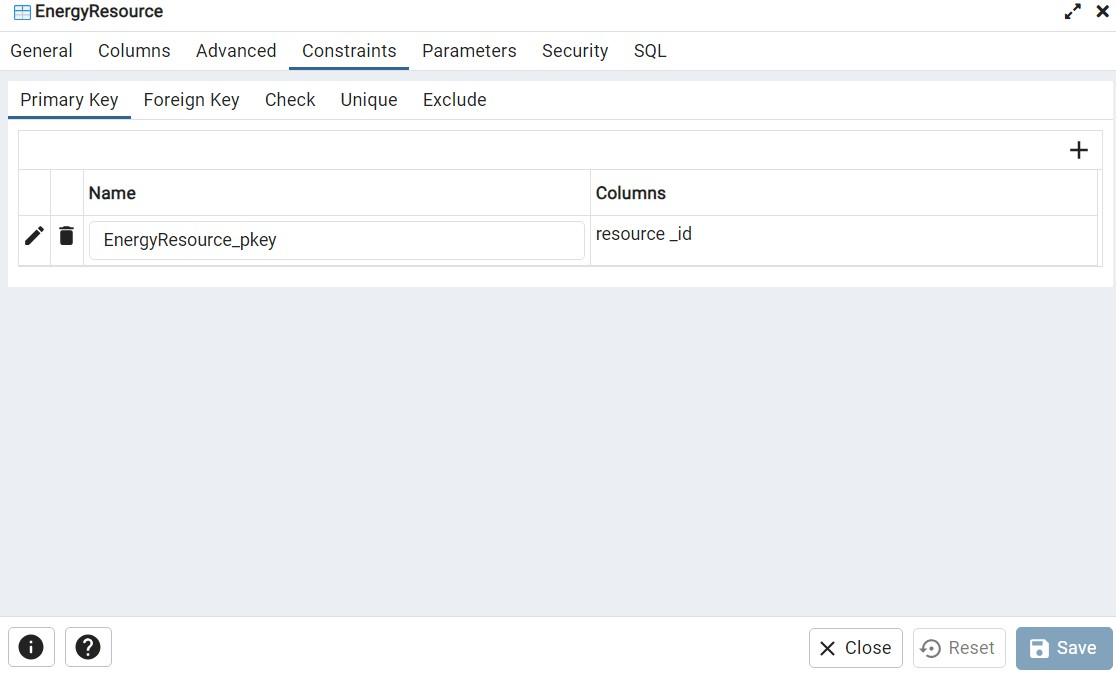
House







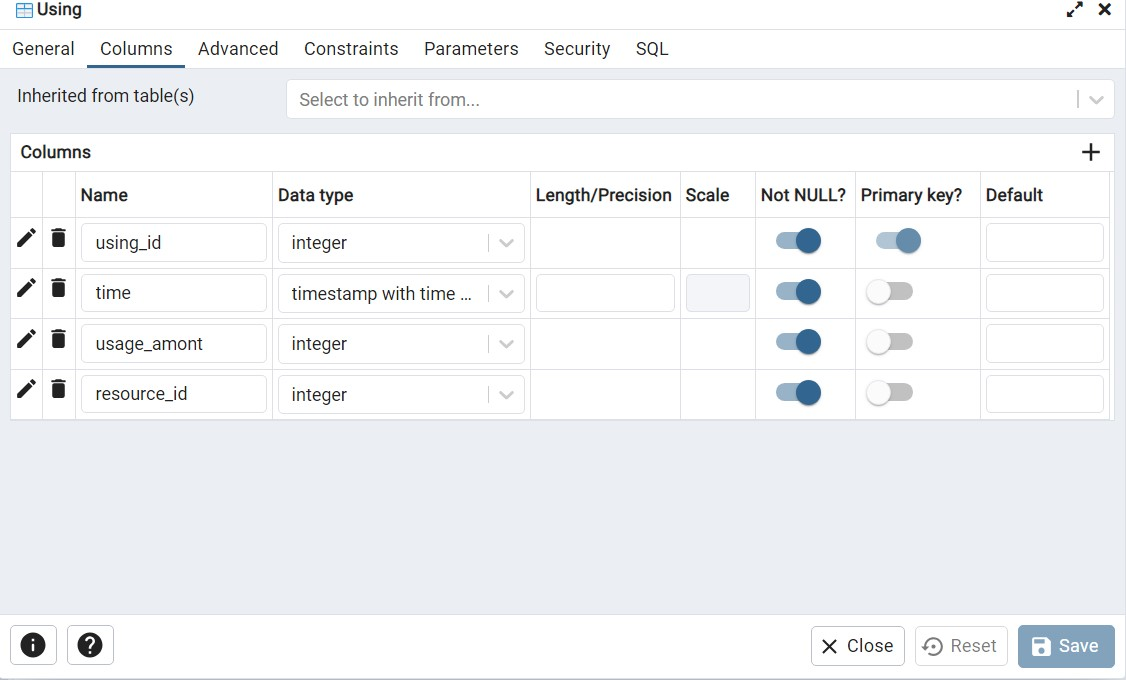
EnergyResource

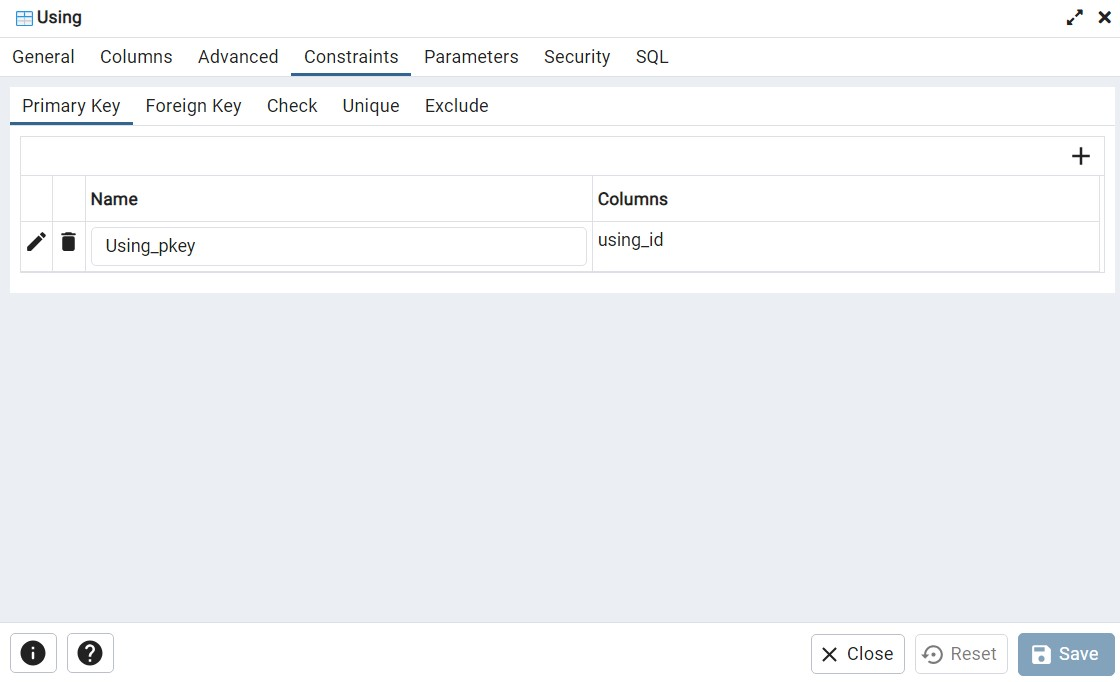
 

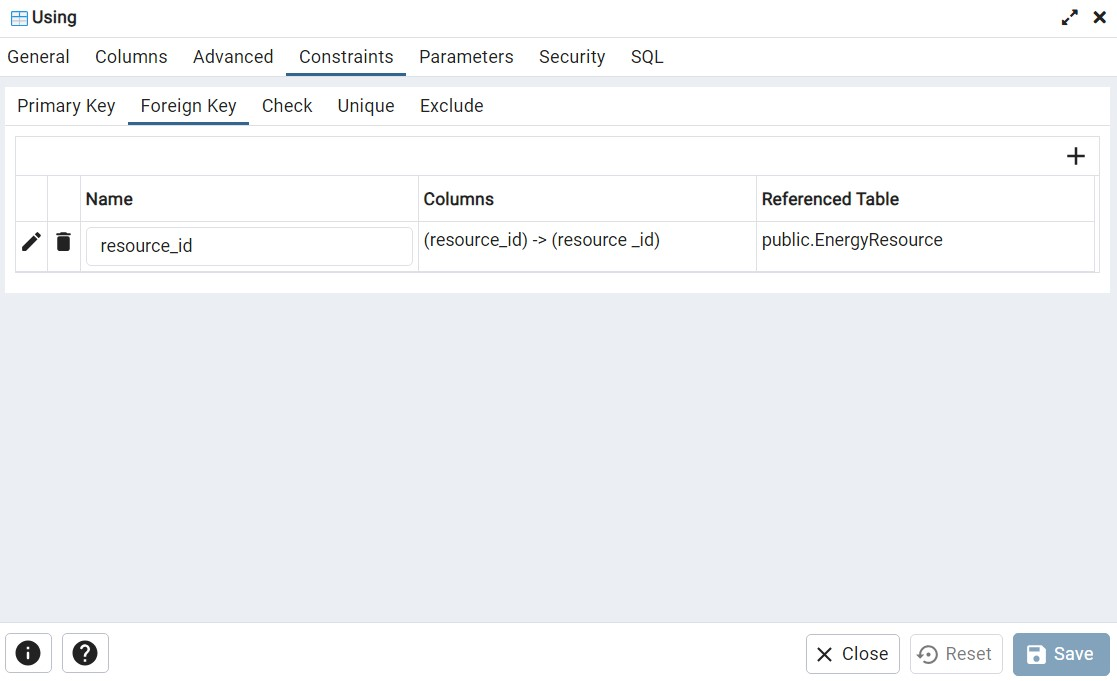
**A screenshot of a computer

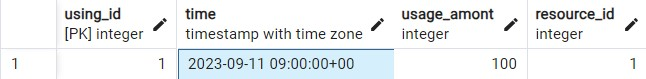
Description automatically generated**

Using

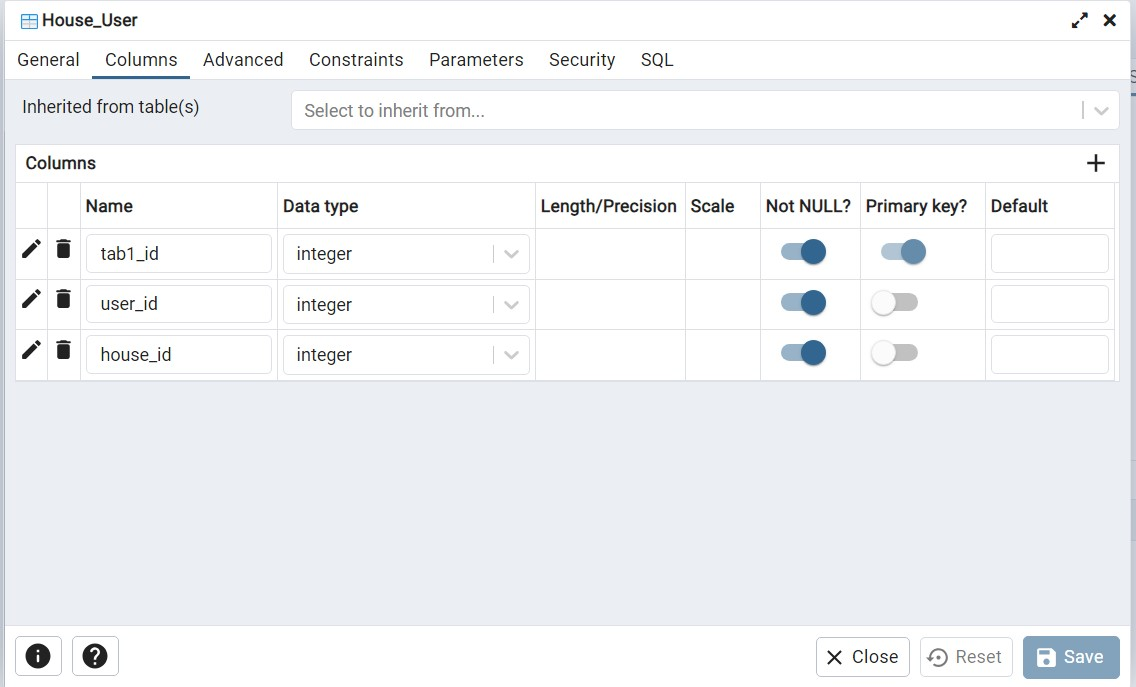


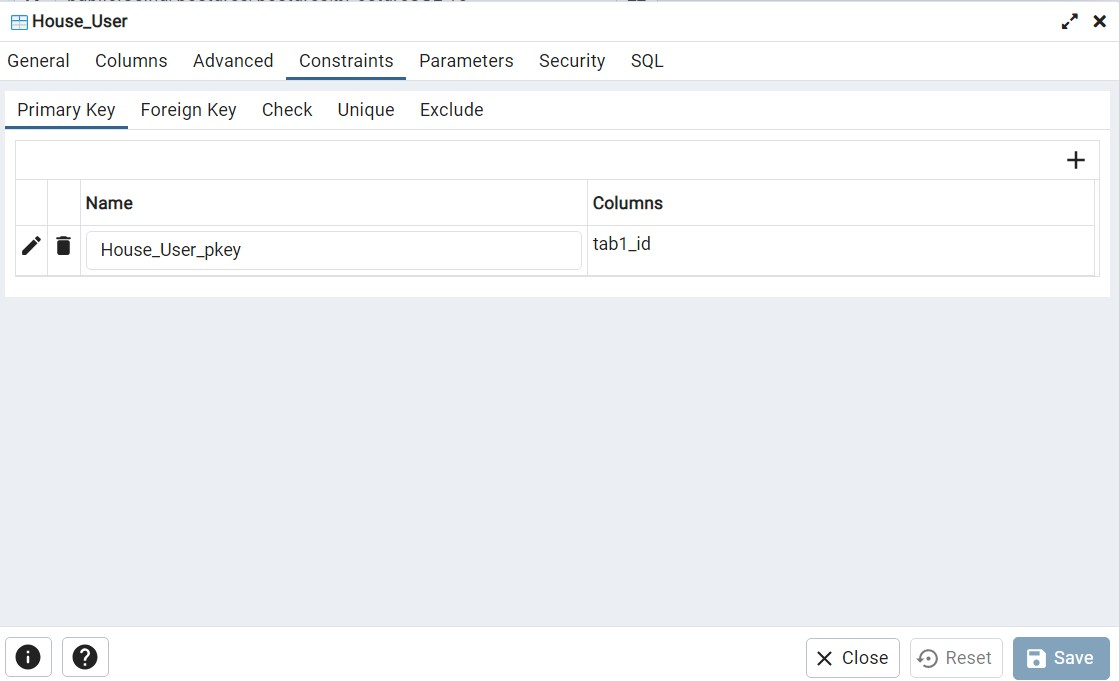


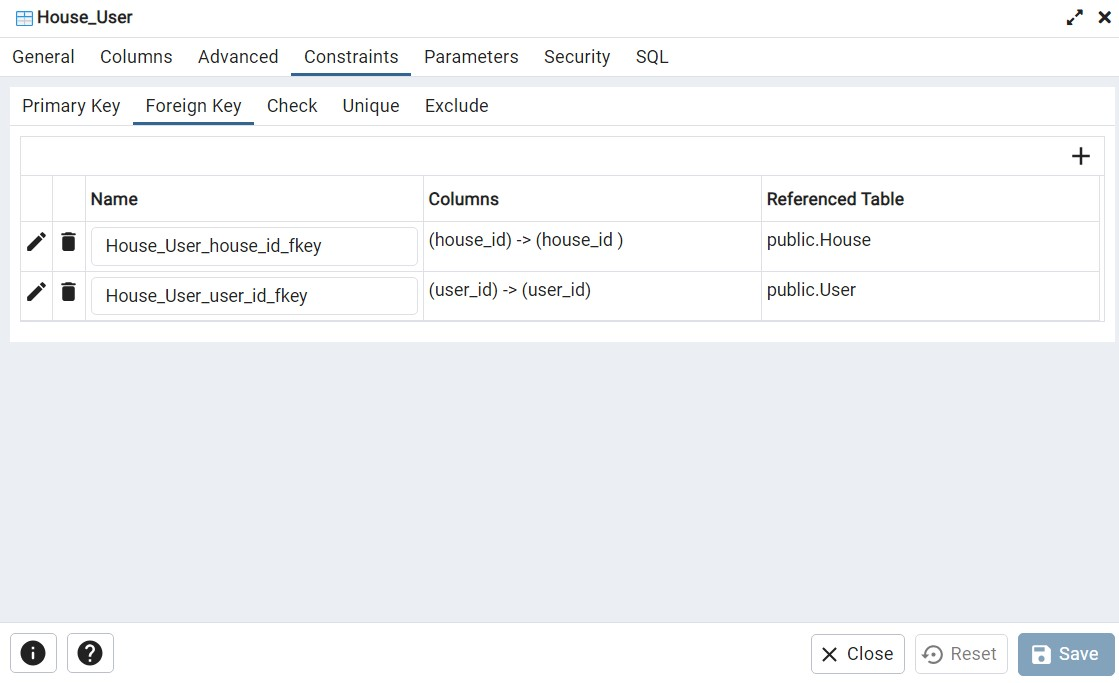
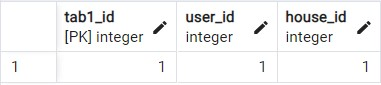




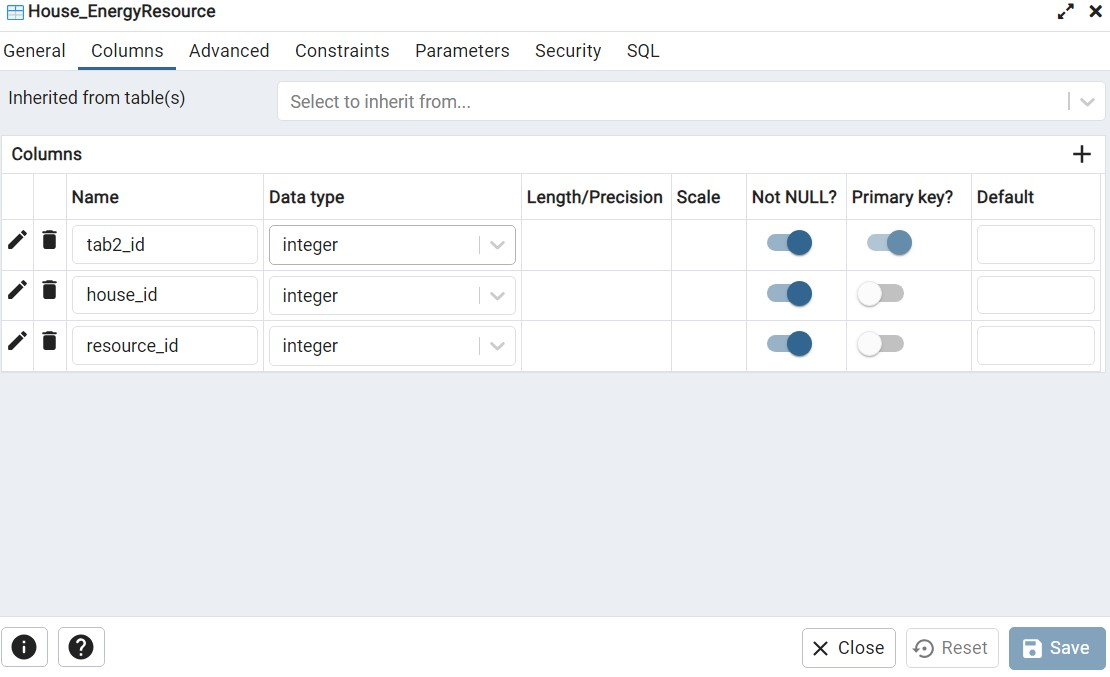
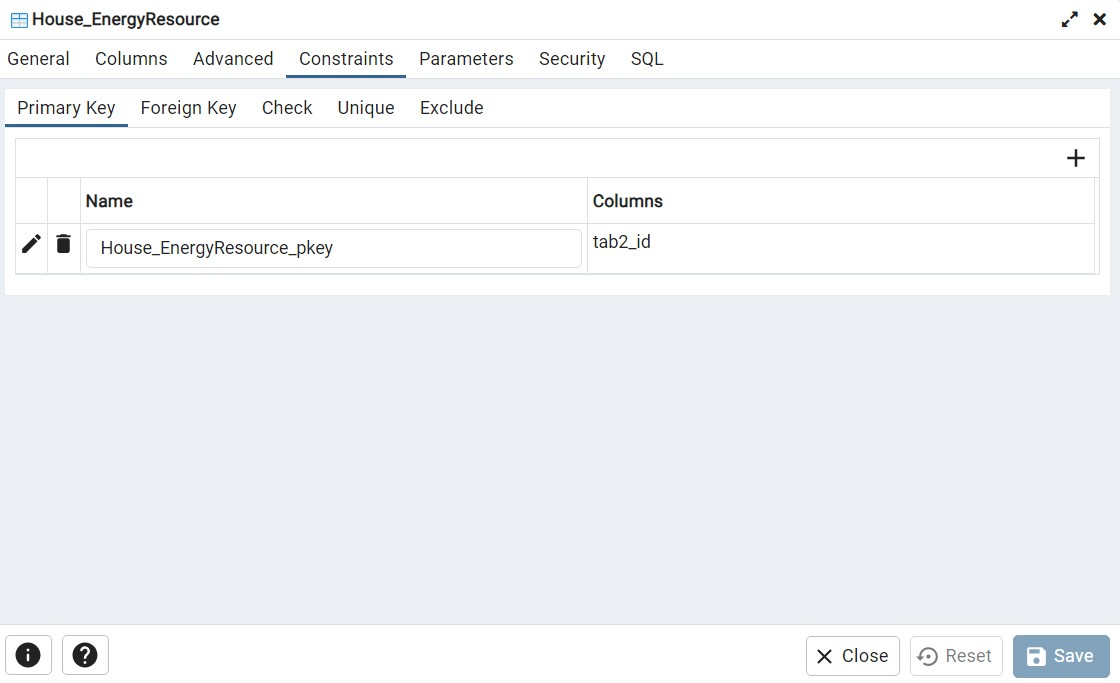
House\_User

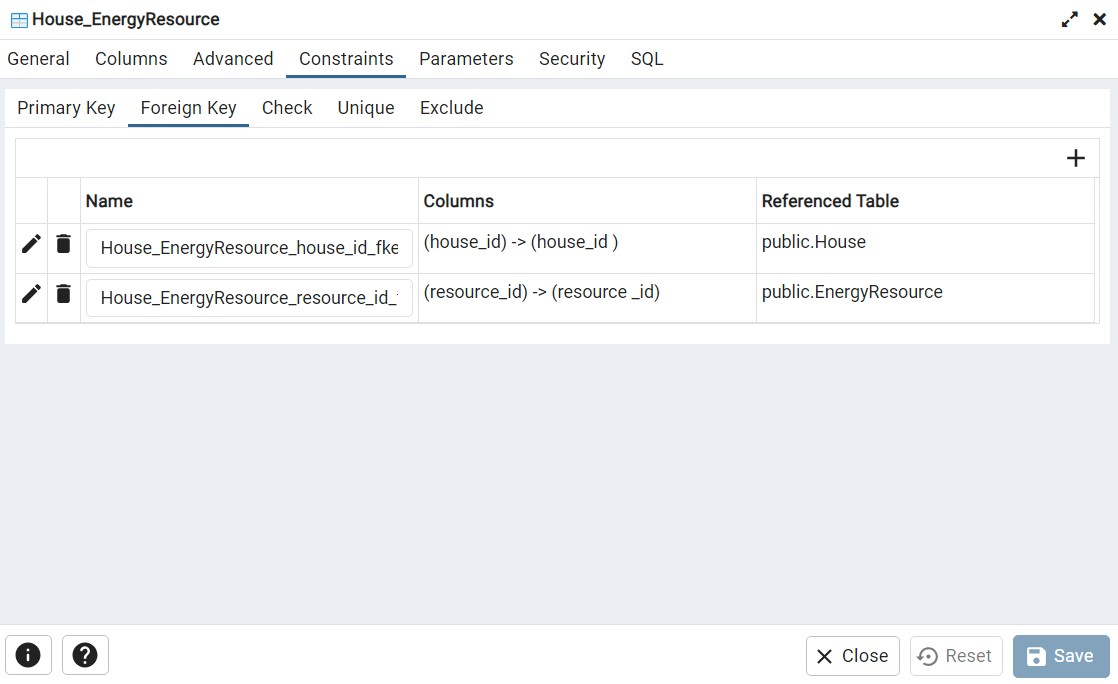


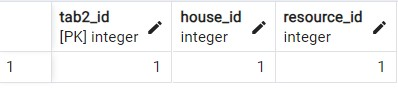


House\_EnergyResource





**Таблиці в коді SQL**

-- Table: public.User

-- DROP TABLE IF EXISTS public."User";

CREATE TABLE IF NOT EXISTS public."User"

(

phone\_number integer NOT NULL,

full\_name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

user\_id integer NOT NULL,

CONSTRAINT user\_id\_pkey PRIMARY KEY (user\_id)

INCLUDE(user\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."User"

OWNER to postgres;

-- Table: Attandance.Group

-- DROP TABLE IF EXISTS "Attandance"."Group";

CREATE TABLE IF NOT EXISTS "Attandance"."Group"

(

group\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

course integer NOT NULL,

number\_of\_students integer NOT NULL,

CONSTRAINT group\_id\_prkey PRIMARY KEY (group\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS "Attandance"."Group"

OWNER to postgres;

-- Table: public.House

-- DROP TABLE IF EXISTS public."House";

CREATE TABLE IF NOT EXISTS public."House"

(

"house\_id " integer NOT NULL,

"address " character varying(100) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT "House\_pkey" PRIMARY KEY ("house\_id ")

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."House"

OWNER to postgres;

-- Table: public.EnergyResource

-- DROP TABLE IF EXISTS public."EnergyResource";

CREATE TABLE IF NOT EXISTS public."EnergyResource"

(

"resource \_id" integer NOT NULL,

resourse\_name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

"unit\_of\_measure " character varying(50) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT "EnergyResource\_pkey" PRIMARY KEY ("resource \_id")

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."EnergyResource"

OWNER to postgres;

-- Table: public.Using

-- DROP TABLE IF EXISTS public."Using";

CREATE TABLE IF NOT EXISTS public."Using"

(

using\_id integer NOT NULL,

"time " timestamp with time zone NOT NULL,

usage\_amont integer NOT NULL,

resource\_id integer NOT NULL,

CONSTRAINT "Using\_pkey" PRIMARY KEY (using\_id),

CONSTRAINT resource\_id FOREIGN KEY (resource\_id)

REFERENCES public."EnergyResource" ("resource \_id") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Using"

OWNER to postgres;

-- Table: public.House\_User

-- DROP TABLE IF EXISTS public."House\_User";

CREATE TABLE IF NOT EXISTS public."House\_User"

(

tab1\_id integer NOT NULL,

user\_id integer NOT NULL,

house\_id integer NOT NULL,

CONSTRAINT "House\_User\_pkey" PRIMARY KEY (tab1\_id),

CONSTRAINT "House\_User\_house\_id\_fkey" FOREIGN KEY (house\_id)

REFERENCES public."House" ("house\_id ") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID,

CONSTRAINT "House\_User\_user\_id\_fkey" FOREIGN KEY (user\_id)

REFERENCES public."User" (user\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."House\_User"

OWNER to postgres;

-- Table: public.House\_EnergyResource

-- DROP TABLE IF EXISTS public."House\_EnergyResource";

CREATE TABLE IF NOT EXISTS public."House\_EnergyResource"

(

tab2\_id integer NOT NULL,

house\_id integer NOT NULL,

resource\_id integer NOT NULL,

CONSTRAINT "House\_EnergyResource\_pkey" PRIMARY KEY (tab2\_id),

CONSTRAINT "House\_EnergyResource\_house\_id\_fkey" FOREIGN KEY (house\_id)

REFERENCES public."House" ("house\_id ") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "House\_EnergyResource\_resource\_id\_fkey" FOREIGN KEY (resource\_id)

REFERENCES public."EnergyResource" ("resource \_id") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."House\_EnergyResource"

OWNER to postgres;

**Висновок**

У ході виконання лабораторної роботи була розроблена база даних для системи відвідуваності занять студентами. База даних включає такі сутності: " User", "House", " EnergyResource", "Using", " House\_ User " та " House \_ EnergyResource ". Кожна з цих сутностей має відповідні атрибути, які дозволяють зберігати та керувати інформацією про студентів, групи, відвідування, викладачів та заняття.

ER-діаграма бази даних була підготовлена для візуального відображення взаємозв'язків між сутностями. Нотація Чена використана для позначення зв'язків та атрибутів.

Схема бази даних пройшла аналіз на відповідність нормальним формам НФ1, НФ2 та НФ3. Всі таблиці відповідають цим нормальним формам, що гарантує надійне та ефективне зберігання даних.

Крім того, була надана оновлена версія схеми бази даних, яка включає в себе таблицю "Teacher\_Discipline" для відображення взаємозв'язків між викладачами та заняттями.

На останньому етапі було надано копії екранів з pgAdmin4, що демонструють властивості стовпців та обмеження, а також вміст таблиць бази даних у PostgreSQL.

У цілому, розроблена база даних відповідає поставленим завданням та вимогам.