

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-91

Селетков В. Р.

Перевірив:

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 занести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

*Вимоги до ER-моделі:*

1. Сутності моделі предметної галузі мають містити зв’язки типу 1:N або N:M.
2. Кількість сутностей у моделі – 3-4. Кількість атрибутів у кожній сутності: від двох до п’яти.
3. Передбачити наявність зв’язку з атрибутом.

Для побудови ER-діаграм використовувати одну із нотацій: Чена, “Пташиної лапки (Crow’s foot)”, UML.

**Опис предметної галузі**

При проектуванні бази даних «Готель» можна виділити наступні сутності: загальні відомості про готель (hotel), номер (room), категорія (category), постоялець (guest), покоївка (chambermaid).

Атрибути заданих сутностей:

* + - 1. hotel: name, restaurant, city, stars.
      2. room: arrival date, departure date, number.
      3. category: room type, allocation type, eating type, price.
      4. guest: name, surname, patronymic, birthday.
      5. chambermaid: name, phone number, salary.

**Опис зв’язків**

У відповідному готелі розташована значна кількість номерів, тому між сутностями hotel та room існує зв’язок 1 : N.

У деякому готелі може проживати багато постояльців, тому між сутностями hotel та guest існує зв’язок 1 : N.

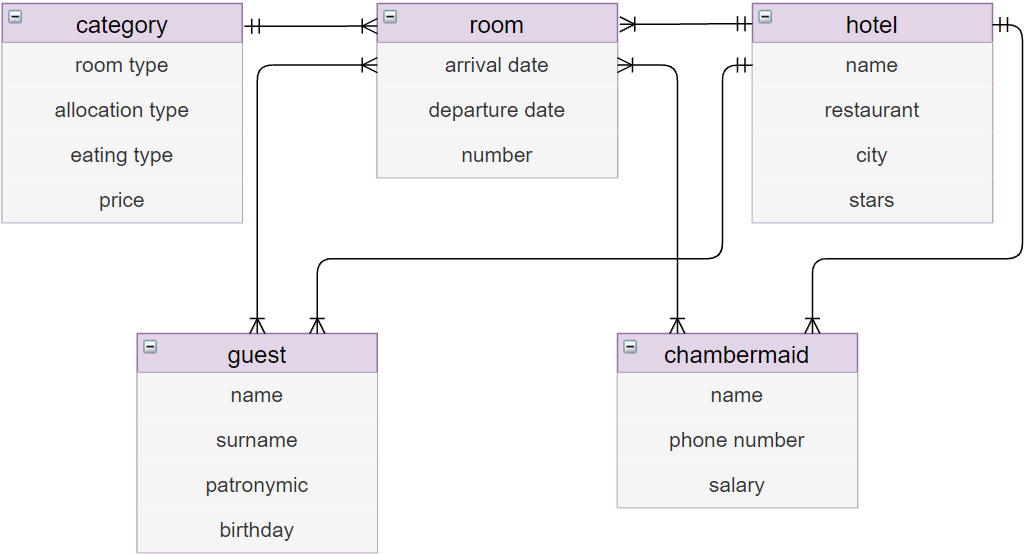
У відповідному готелі повинна працювати велика кількість робітників, тому між сутностями hotel та chambermaid існує зв’язок 1 : N.

Деякій категорії може належати декілька номерів, тому між сутностями category та room існує зв’язок 1 : N.

Один гість міг заселятися в різні номери, також в один номер могли заселятися різні гості, тому між сутностями guest та room існує зв’язок N : M.

Один номер може бути під контролем декількох покоївок, також одна покоївка може прибирати декілька номерів, тому між сутностями room та chambermaid існує зв’язок N : M.

Концептуальна модель «сутність-зв’язок» предметної галузі «Готель» зображена на рисунку 1.

Рисунок 1 - ER-діаграма побудована за нотацією “Пташиної лапки (Crow’s foot)”, задана ER-діаграма була побудована у додатку [draw.io](https://app.diagrams.net/)

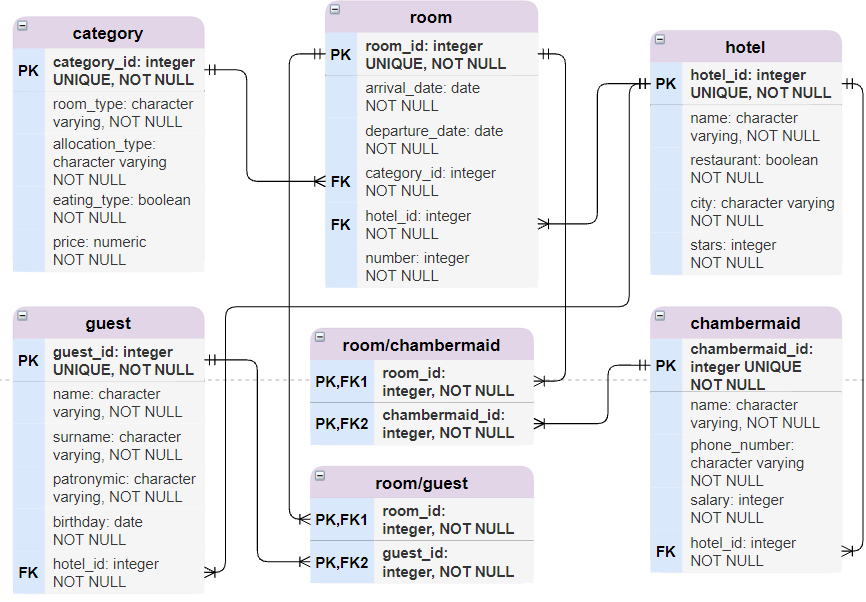
**Перетворення концептуальної моделі у схему баз даних**

Для кожної сутності створюється таблиця. Причому кожному атрибуту сутності відповідає стовпець таблиці. В даній моделі перетворення в схему баз даних відбувалося за такими правилами:

Якщо зв'язок типу 1: N і клас приналежності сутності 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 category та room, hotel та room, hotel та guest, hotel та chambermaid.

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 ключі обох сутностей. Даний зв’язок можна спостерігати між сутностями room та guest, room та chambermaid.

Перетворена концептуальна модель у схему бази даних зображена на рисунку 2.

****Рисунок 2 - Схема бази даних, побудовано у додатку [draw.io](https://app.diagrams.net/)

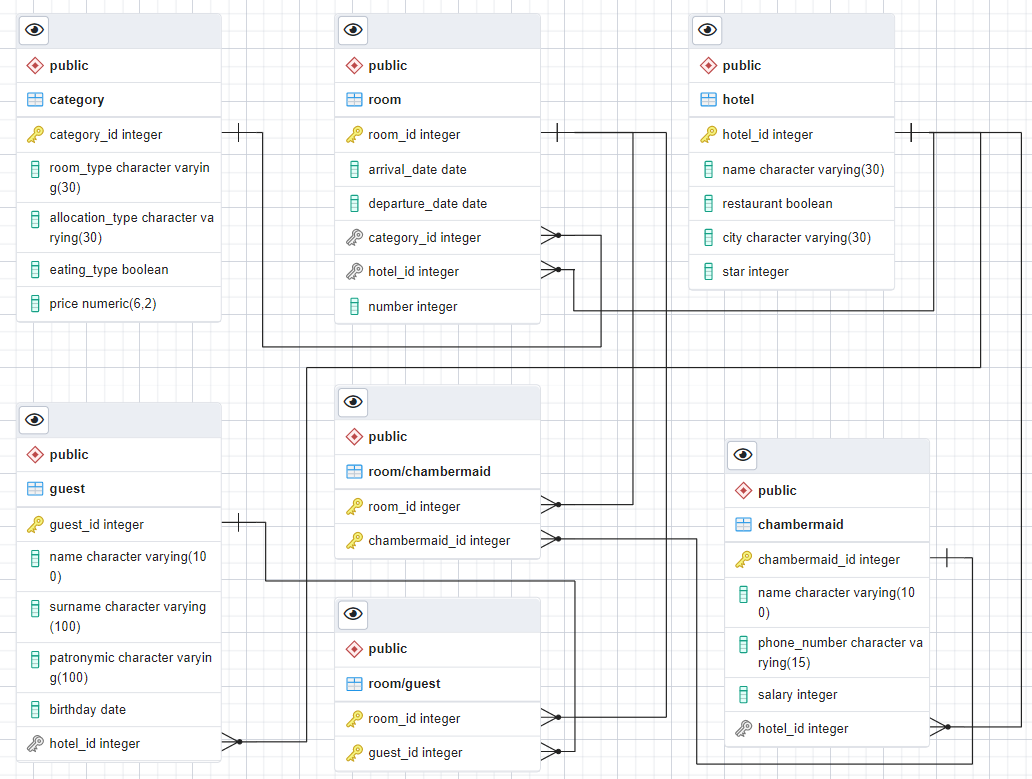
Після розробки моделі предметної галузі «Готель» та перетворення її у схему бази даних, було створено дану базу даних у додатку pgAdmin 4, схема представлена на рисунку 3.

Рисунок 3 - Схема бази даних у додатку pgAdmin 4

**Опис структури бази даних**

Сутність category перетворена у таблицю category, в ній присутній тільки один первинний ключ та відсутні зовнішні ключі. Сутність hotel перетворено у табличку hotel, де теж немає зовнішніх ключів і тільки один первинний ключ. Сутність room перетворена у табличку room, у цієї таблички тільки один первинний ключ та за наявності зв’язків 1 : N присутні також зовнішні ключі: category\_id, hotel\_id. Сутність guest перетворена у табличку guest, в даному випадку знаходиться тільки один первинний ключ та за наявності зв’язку 1 : N ще є зовнішній ключ hotel\_id. Сутність chambermaid перетворена у табличку chambermaid, дана табличка має первинний ключ та зовнішній ключ hotel\_id. Оскільки сутності room та chambermaid, а також сутності room та guest зв’язані багато до багатьох, то це призвело до появи третьої таблички відповідно room/chambermaid та room/guest дані табличка мають два первинних ключі: room\_id, chambermaid\_id та room\_id, guest\_id. Опис даних сутностей та їх атрибутів наведено у таблиці 1.

Таблиця 1 - Опис структури БД.

|  |  |  |
| --- | --- | --- |
| Відношення | Атрибут | Тип атрибуту |
| **hotel –** містить дані про готель | **hotel\_id –** унікальний ідентифікатор  **name –** назва готелю  **restaurant –** наявність ресторану  **city –** місто  **stars –** кількість зірок | **integer** (числовий)  **character varying** (рядок)  **boolean** (булевий)  **character varying** (рядок)  **integer** (числовий) |
| **category –** містить дані про категорію номеру у готелі | **category\_id** – унікальний ідентифікатор  **room\_type –** тип номеру  **allocation\_type –** тип розселення в номері  **eating\_type –** наявність харчування  **price –** ціна | **integer** (числовий)  **character varying** (рядок)  **character varying** (рядок)  **boolean** (булевий)  **numeric** (фіксований) |
| **guest –** містить дані про постояльців готелю | **guest\_id** – унікальний ідентифікатор  **name** – ім’я  **surname –** прізвище  **patronymic –** по батькові  **birthday –** день народження  **hotel\_id –** ідентифікатор готелю | **integer** (числовий)  **character varying** (рядок)  **character varying** (рядок)  **character varying** (рядок)  **date** (дата)  **integer** (числовий) |
| **room** – містить дані щодо номеру | **room\_id -** унікальний ідентифікатор  **arrival\_date –** дата заселення  **departure\_date –** дата виселення  **category\_id** – ідентифікатор категорії  **hotel\_id –** ідентифікатор готелю  **number –** номер кімнати | **integer** (числовий)  **date** (дата)  **date** (дата)  **integer** (числовий)  **integer** (числовий)  **integer** (числовий) |
| **chambermaid –** містить дані про покоївок готелю | **chambermaid\_id –** унікальний ідентифікатор  **name –** ім’я  **phone –** номер телефону  **salary –** заробітня плата  **hotel\_id –** ідентифікатор готелю | **integer** (числовий)  **character varying** (рядок)  **character varying** (рядок)  **integer** (числовий)  **integer** (числовий) |
| **room/**  **chambermaid** -  відношення покоївок до кімнат | **room\_id–** ідентифікатор номера  **chambermaid\_id –** ідентифікатор покоївки | **integer** (числовий)  **integer** (числовий) |
| **room/guest** -  відношення постояльців до кімнат | **room\_id–** ідентифікатор номера  **chambermaid\_id –** ідентифікатор постояльця | **integer** (числовий)  **integer** (числовий) |

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

**Функціональні залежності для кожної таблиці**

1. **hotel:**
2. hotel\_id -> name (назва готелю залежить від унікального коду готелю)
3. hotel\_id -> restaurant (наявність ресторану залежить від унікального коду готелю)
4. hotel\_id -> city (місто в якому знаходиться готель залежить від унікального коду готелю)
5. hotel\_id -> stars (кількість зірок готелю залежать від унікального коду готелю)
6. **category:**
7. category\_id -> room\_type (тип відповідної кімнати залежить від унікального коду категорії)
8. category\_id -> allocation\_type (тип розселення в кімнаті залежить від унікального коду категорії)
9. category\_id -> eating\_type (тип харчування залежить від унікального коду категорії)
10. category\_id -> price (ціна номеру залежить від типу номера та типу розселення номера, які в свою чергу залежать від унікального коду категорії)
11. **room:**
12. room\_id -> arrival\_date (дата заселення в номер залежить від унікального ідентифікатора номеру)
13. room\_id -> departure\_date (дата виселення з номеру залежить від унікального ідентифікатора номеру)
14. room\_id -> number (номер кімнати залежить від унікального ідентифікатора кімнати)
15. **guest:**
16. guest\_id -> name (ім’я постояльця залежить від унікального ідентифікатора гостя)
17. guest\_id -> surname (прізвище постояльця залежить від унікального ідентифікатора гостя)
18. guest\_id -> patronymic (ім’я по батькові постояльця залежить від унікального ідентифікатора гостя)
19. guest\_id -> birthday (день народження постояльця залежить від унікального ідентифікатора гостя)
20. **chambermaid:**
21. chambermaid\_id -> name (ім’я покоївки залежить від її унікального ідентифікатора)
22. chambermaid\_id -> phone (номер телефону покоївки залежить від її унікального ідентифікатора)
23. chambermaid\_id -> salary (заробітня плата покоївки залежить від її унікального ідентифікатора)

**Відповідність схеми бази даних до третьої нормальної форми**

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.

**SQL текст**

-- Database: hotel\_db

-- DROP DATABASE hotel\_db;

CREATE DATABASE hotel\_db

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'Russian\_Russia.1251'

LC\_CTYPE = 'Russian\_Russia.1251'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1;

-- Table: public.category

-- DROP TABLE public.category;

CREATE TABLE IF NOT EXISTS public.category

(

category\_id integer NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

room\_type character varying(30) COLLATE pg\_catalog."default" NOT NULL,

allocation\_type character varying(30) COLLATE pg\_catalog."default" NOT NULL,

eating\_type boolean NOT NULL,

price numeric(6,2) NOT NULL,

CONSTRAINT category\_pkey PRIMARY KEY (category\_id)

)

TABLESPACE pg\_default;

ALTER TABLE public.category

OWNER to postgres;

-- Table: public.chambermaid

-- DROP TABLE public.chambermaid;

CREATE TABLE IF NOT EXISTS public.chambermaid

(

chambermaid\_id integer NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

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

phone\_number character varying(20) COLLATE pg\_catalog."default" NOT NULL,

salary integer NOT NULL,

hotel\_id integer NOT NULL,

CONSTRAINT chambermaid\_pkey PRIMARY KEY (chambermaid\_id),

CONSTRAINT fk\_chambermaid\_hotel FOREIGN KEY (hotel\_id)

REFERENCES public.hotel (hotel\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.chambermaid

OWNER to postgres;

-- Table: public.guest

-- DROP TABLE public.guest;

CREATE TABLE IF NOT EXISTS public.guest

(

guest\_id integer NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

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

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

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

birthday date NOT NULL,

hotel\_id integer NOT NULL,

CONSTRAINT guest\_pkey PRIMARY KEY (guest\_id),

CONSTRAINT fk\_guest\_hotel FOREIGN KEY (hotel\_id)

REFERENCES public.hotel (hotel\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.guest

OWNER to postgres;

-- Table: public.hotel

-- DROP TABLE public.hotel;

CREATE TABLE IF NOT EXISTS public.hotel

(

hotel\_id integer NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

name character varying(30) COLLATE pg\_catalog."default" NOT NULL,

restaurant boolean NOT NULL,

city character varying(30) COLLATE pg\_catalog."default" NOT NULL,

star integer NOT NULL,

CONSTRAINT hotel\_pkey PRIMARY KEY (hotel\_id)

)

TABLESPACE pg\_default;

ALTER TABLE public.hotel

OWNER to postgres;

-- Table: public.room

-- DROP TABLE public.room;

CREATE TABLE IF NOT EXISTS public.room

(

room\_id integer NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

arrival\_date date NOT NULL,

departure\_date date NOT NULL,

category\_id integer NOT NULL,

hotel\_id integer NOT NULL,

"number" integer NOT NULL,

CONSTRAINT room\_pkey PRIMARY KEY (room\_id),

CONSTRAINT fk\_room\_category FOREIGN KEY (category\_id)

REFERENCES public.category (category\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT fk\_room\_hotel FOREIGN KEY (hotel\_id)

REFERENCES public.hotel (hotel\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.room

OWNER to postgres;

-- Table: public.room/chambermaid

-- DROP TABLE public."room/chambermaid";

CREATE TABLE IF NOT EXISTS public."room/chambermaid"

(

room\_id integer NOT NULL DEFAULT '-1'::integer,

chambermaid\_id integer NOT NULL DEFAULT '-1'::integer,

CONSTRAINT "room/chambermaid\_pkey" PRIMARY KEY (room\_id, chambermaid\_id),

CONSTRAINT fk\_chambermaid\_room FOREIGN KEY (chambermaid\_id)

REFERENCES public.chambermaid (chambermaid\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT fk\_room\_chambermaid FOREIGN KEY (room\_id)

REFERENCES public.room (room\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public."room/chambermaid"

OWNER to postgres;

-- Table: public.room/guest

-- DROP TABLE public."room/guest";

CREATE TABLE IF NOT EXISTS public."room/guest"

(

room\_id integer NOT NULL,

guest\_id integer NOT NULL,

CONSTRAINT "room/guest\_pkey" PRIMARY KEY (room\_id, guest\_id),

CONSTRAINT fk\_guest\_room FOREIGN KEY (guest\_id)

REFERENCES public.guest (guest\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT fk\_room\_guest FOREIGN KEY (room\_id)

REFERENCES public.room (room\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

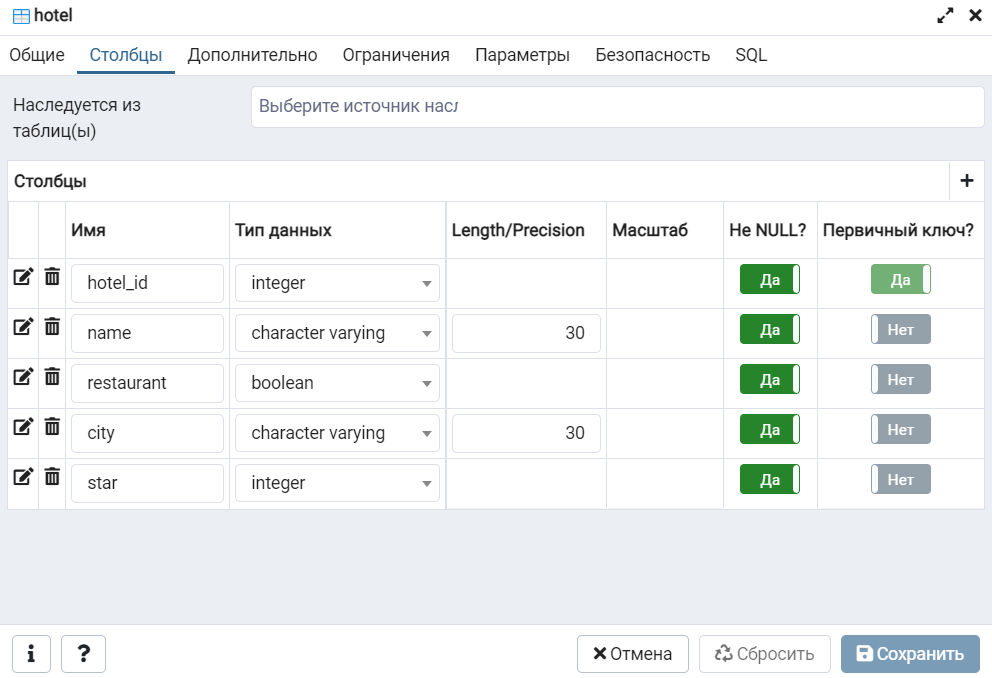
)

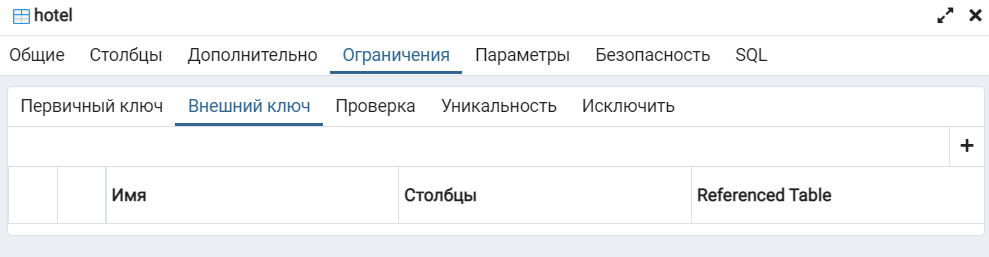
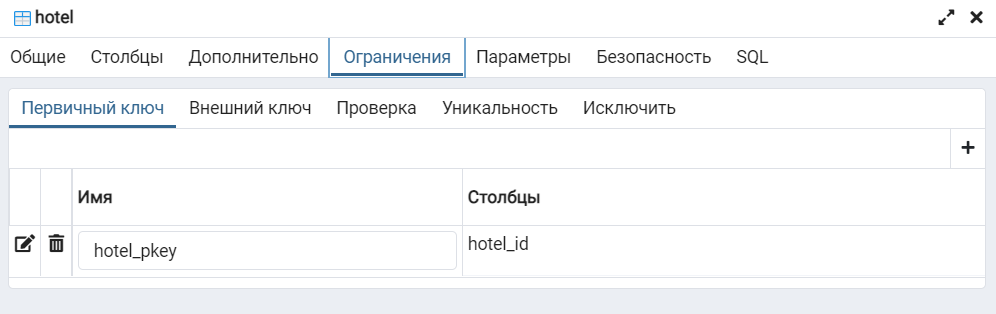
TABLESPACE pg\_default;

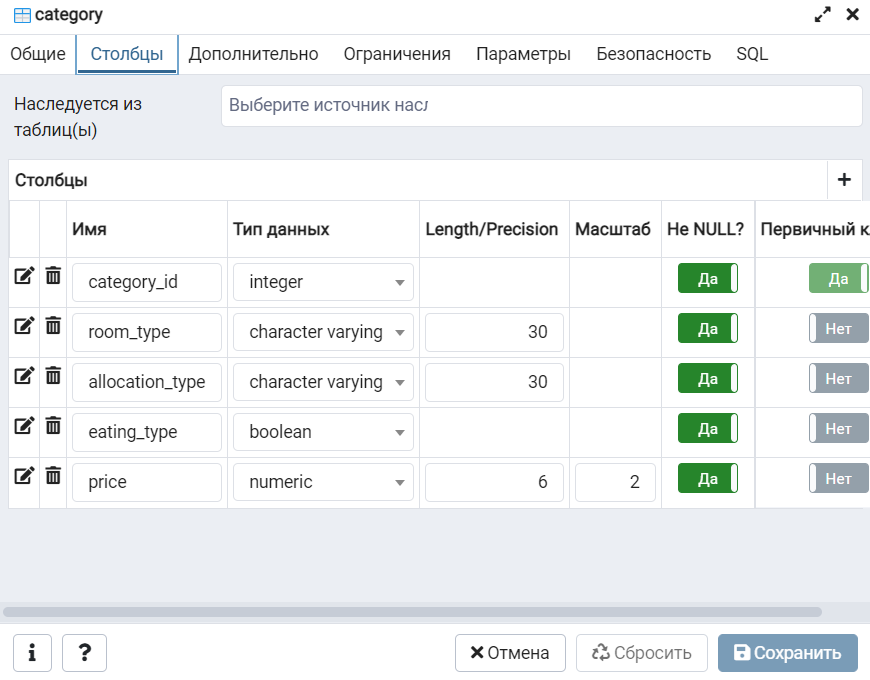
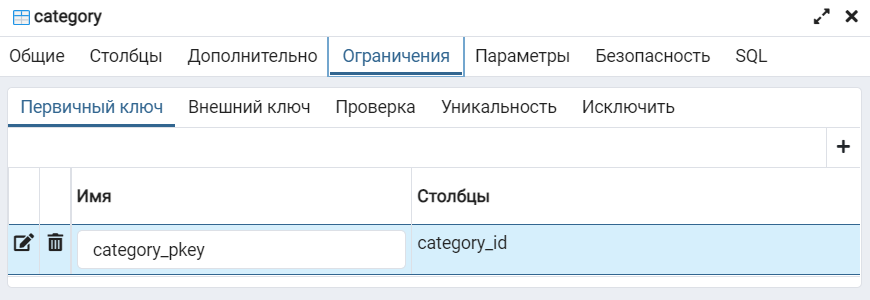
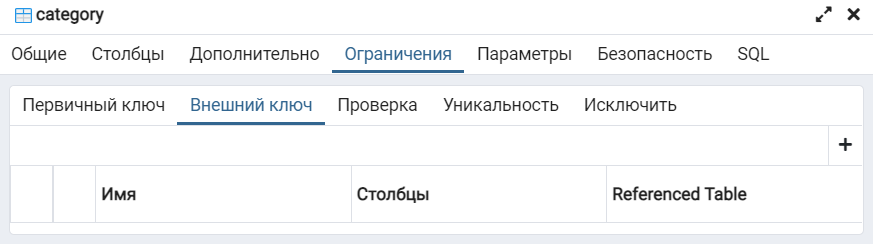
ALTER TABLE public."room/guest"

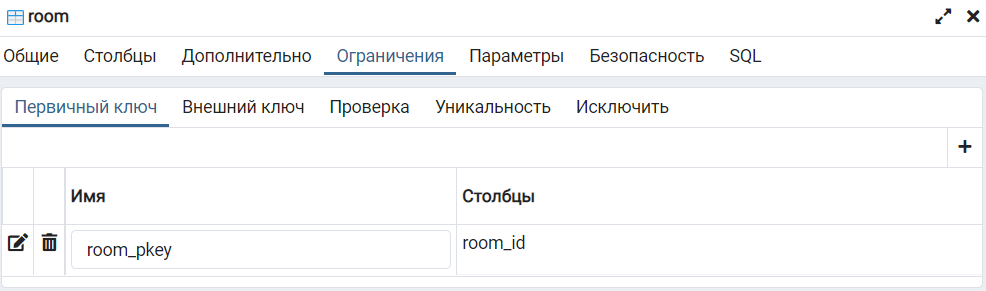
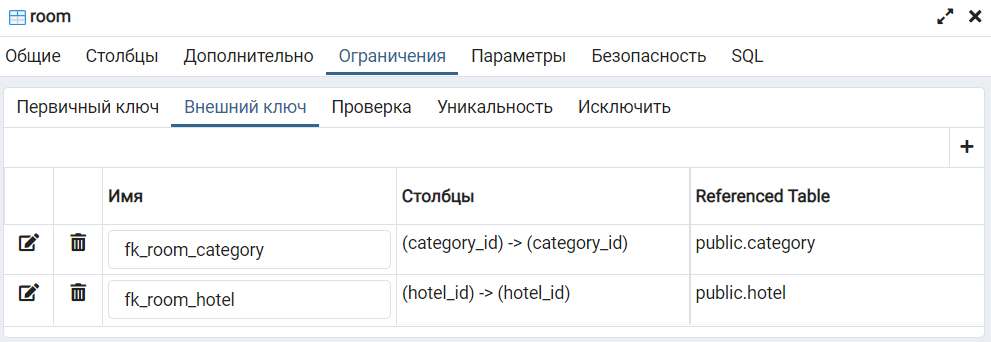
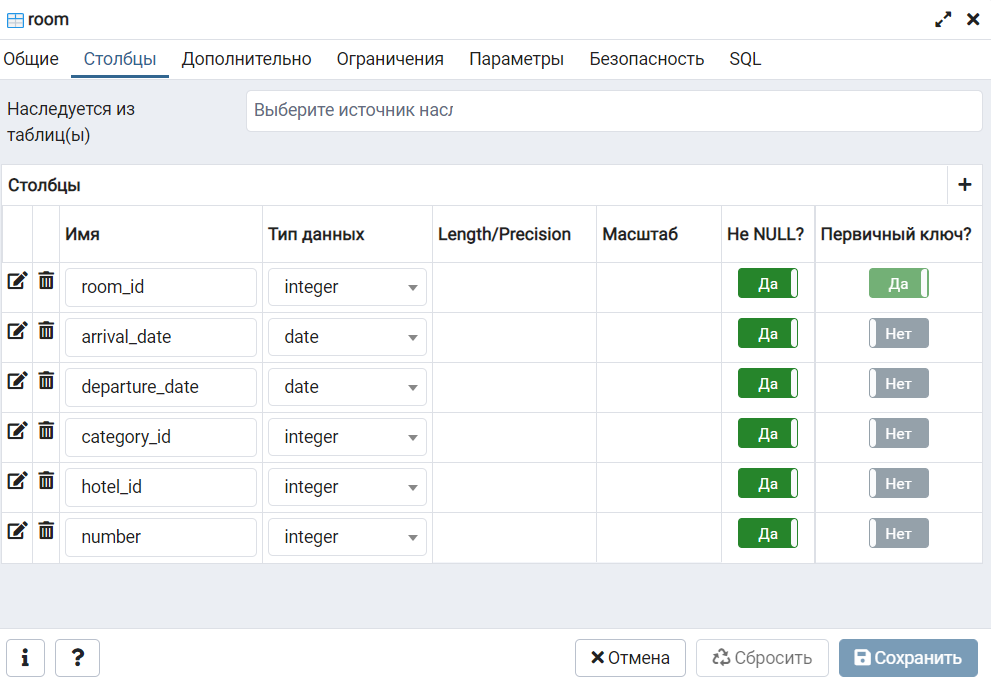
OWNER to postgres;

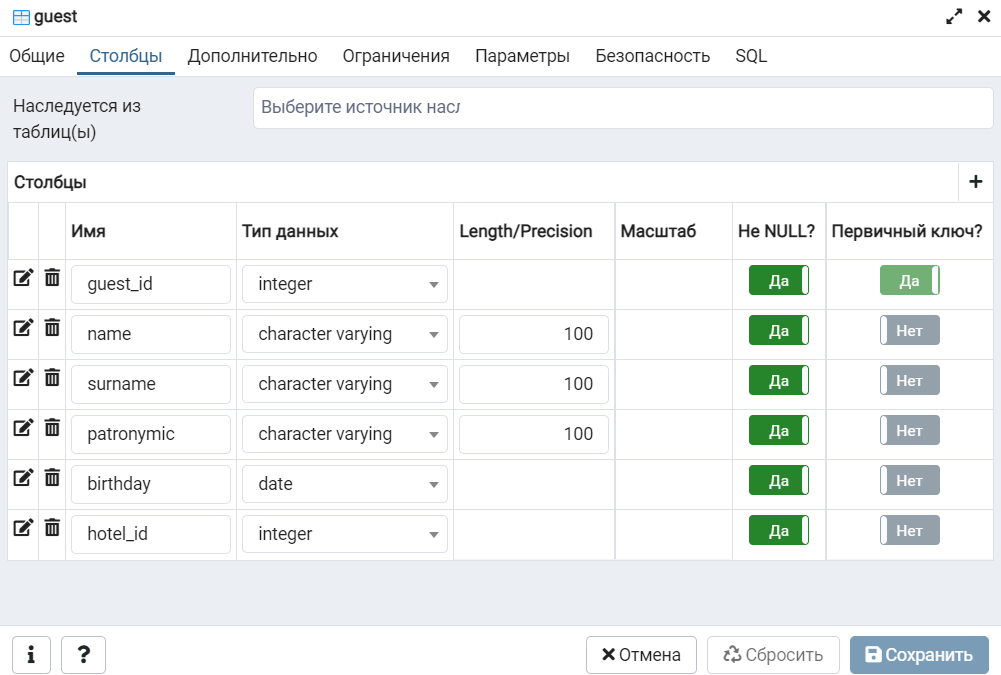
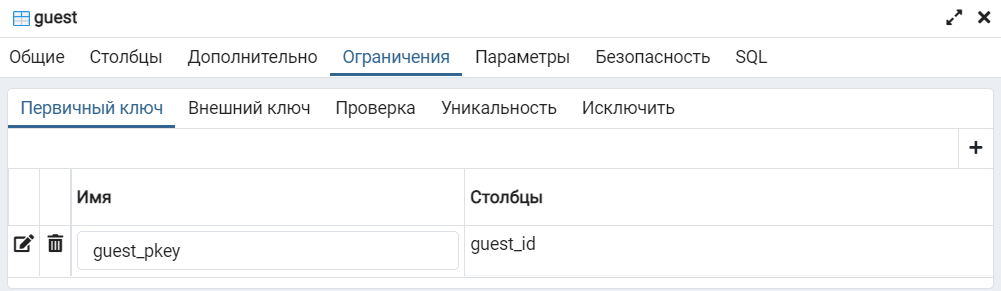
**Таблиці бази даних у pgAdmin 4**

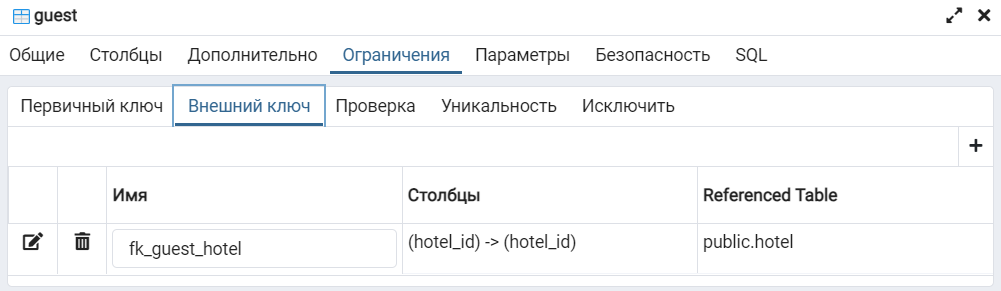
**hotel**

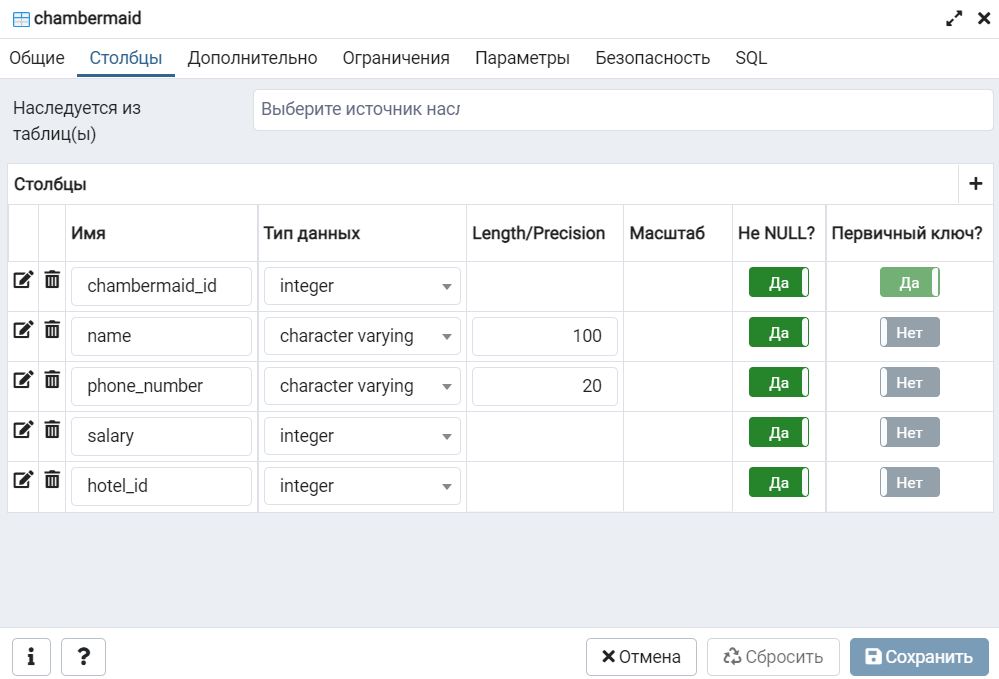
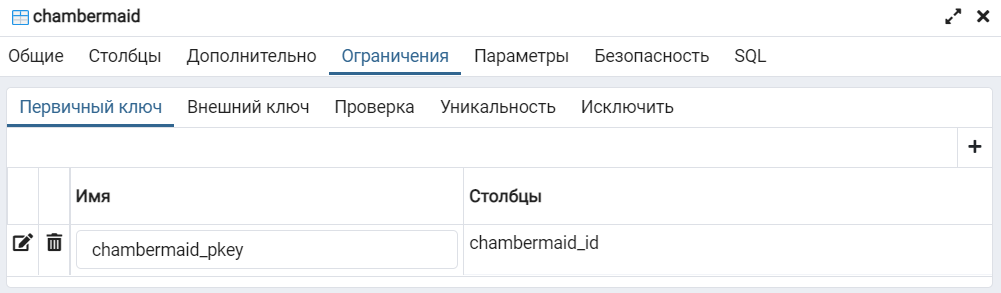


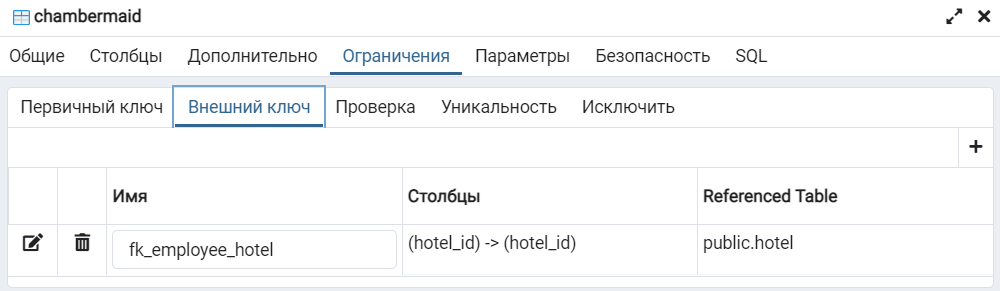
**category**

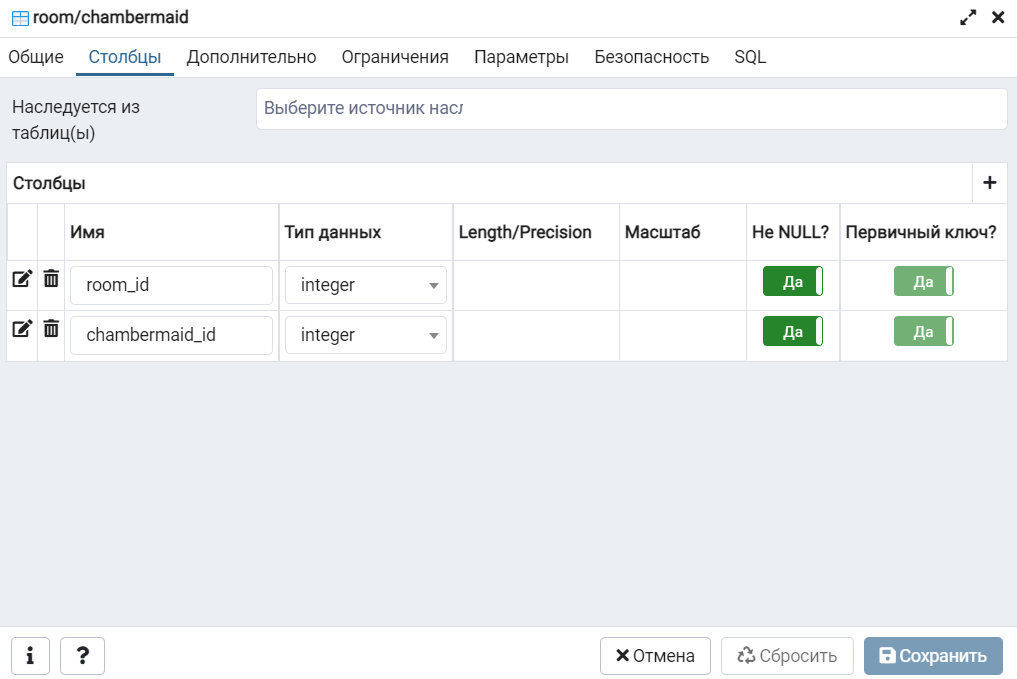
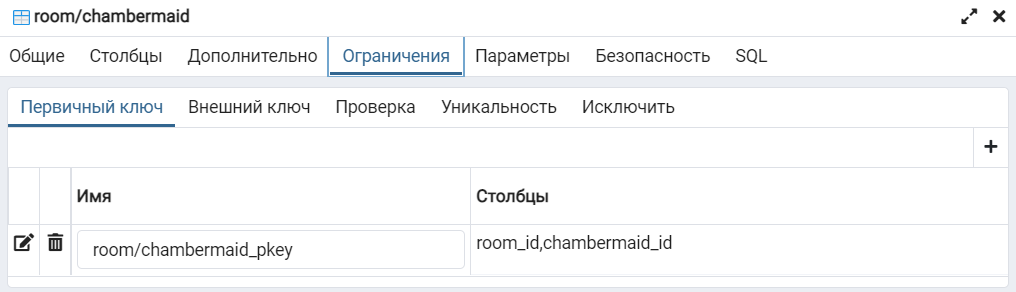
******room**

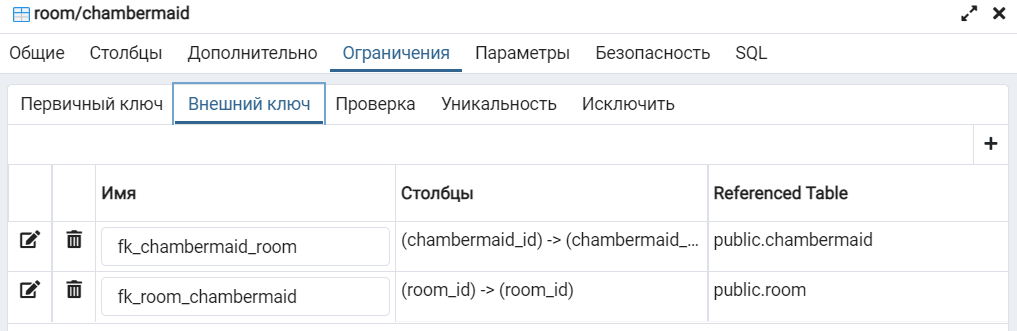
**guest**

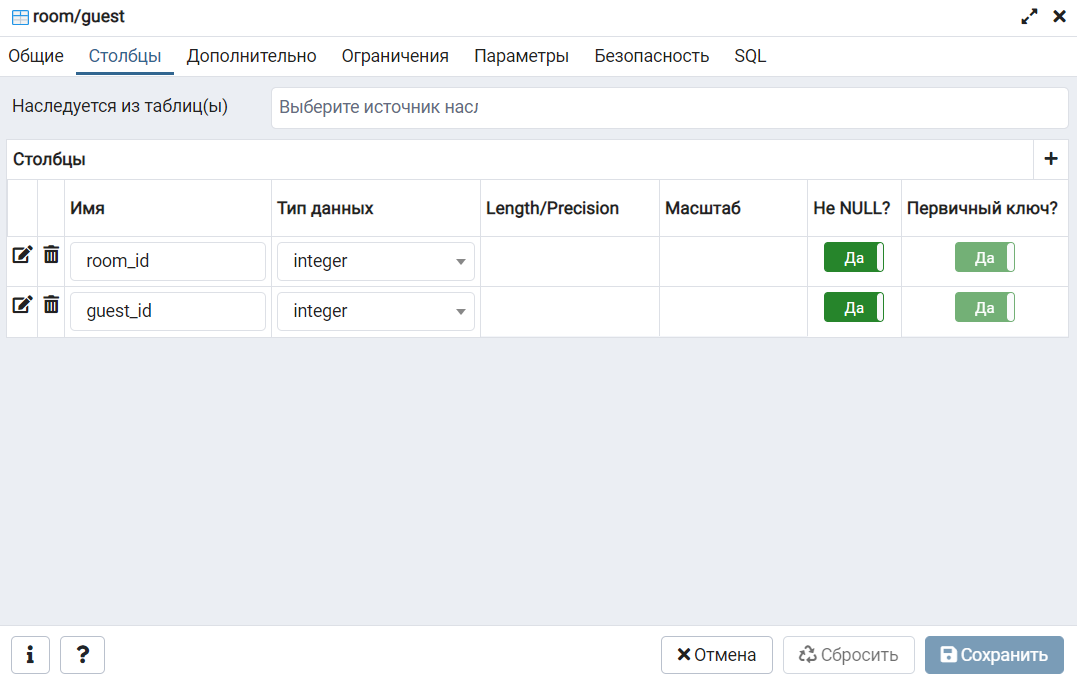
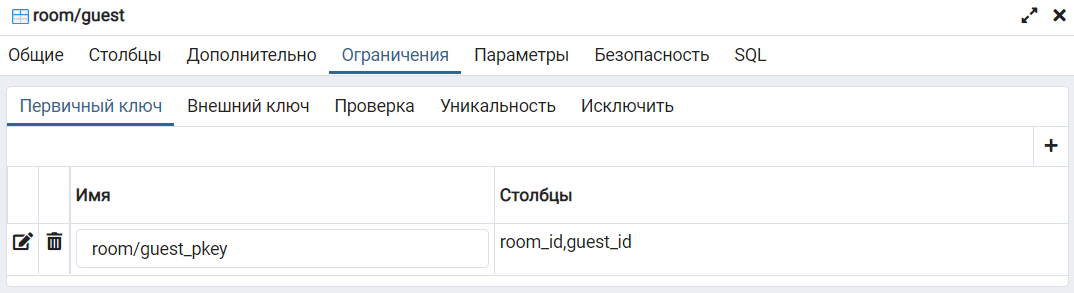


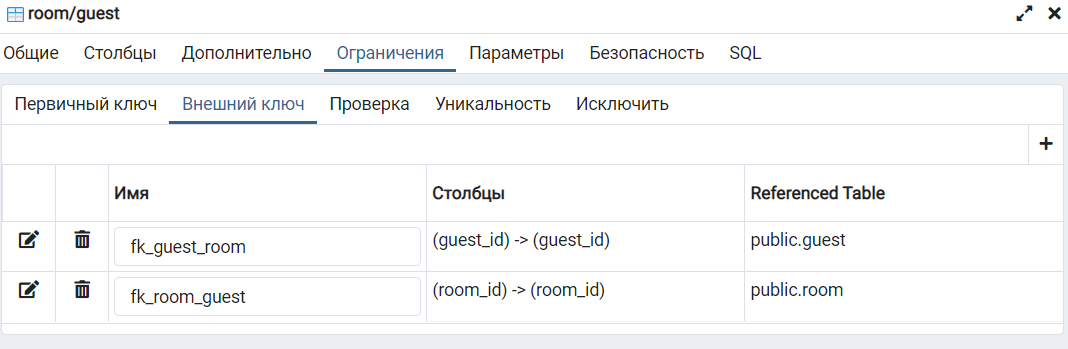
**chambermaid**

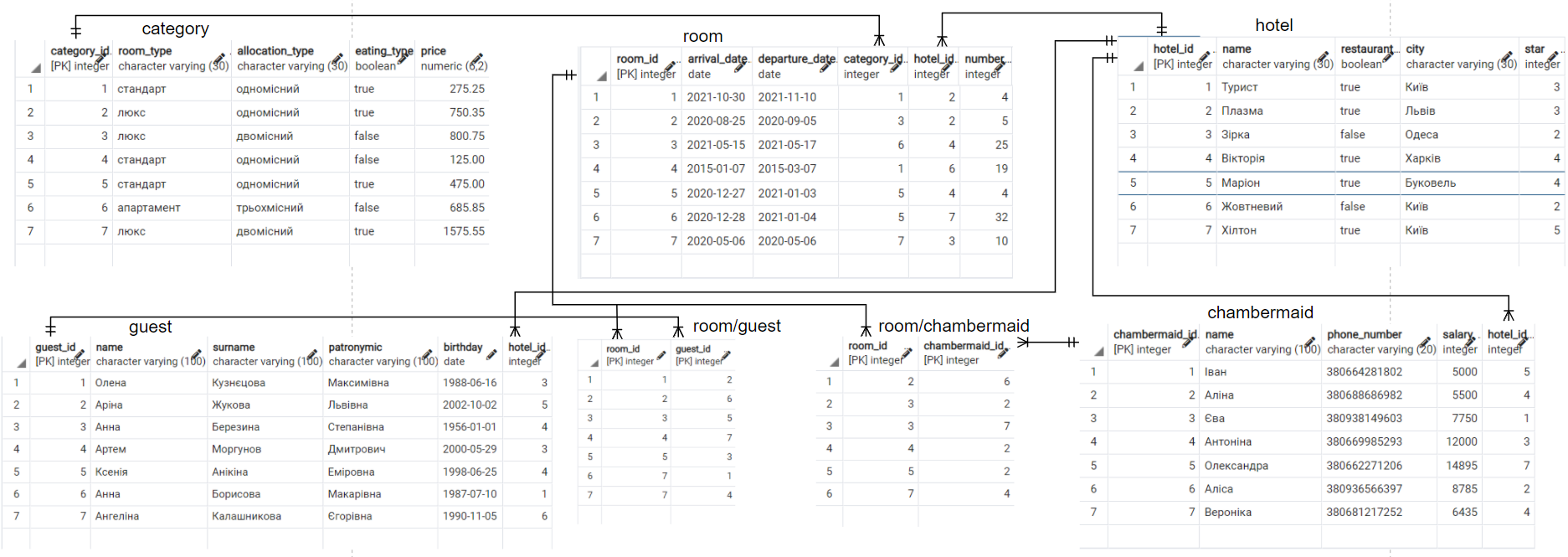


**room/chambermaid**



**room/guest**



**Вміст таблиць бази даних**