Зображення, що містить будівля, ескіз, чорно-білий, панорама

Автоматично згенерований опис

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих комп’ютерних**

**cистем**

**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-12

Манжелій Р. В.

Перевірив:

Павловский В. І.

Київ – 2023

*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 кожну з таблиць засобами pgAdmin.

**Модель «сутність-зв’язок» галузі виробництва**

Обрана предметна галузь передбачає отримання і обробку потрібної інформації щодо відстеження запасних частин для виробництва.

**Сутності**

Згідно цієї області для побудови бази даних було виділено наступні сутності:

**Запасна частина з атрибутами**: код запчастини, код постачальника, назва запчастини, ціна. Призначення: збереження основних даних, що стосуються запасних запчастин.

**Постачальник з атрибутами**: код постачальника, назва компанії, адреса та номер телефону. Призначення: збереження даних щодо постачальників.

**Склад з атрибутами**: номер складу, адрес. Призначення: збереження даних щодо складів.

**Клієнт з атрибутами**: код клієнта, ПІБ, номер телефону. Призначення: збереження інформації стосовно клієнтів.

**Замовлення з атрибутами**: номер замовлення, код запчастин, код клієнта, загальна вартість замовлення, дата замовлення. Призначення: збереження інформації стосовно замовлень запасних частин.

**Опис зв’язків**

Одна компанія може постачати декілька видів запасних частин, а запасна частина може мати одного постачальника. Тому між сутностями Постачальник і Запасна Частина існує зв’язок 1:N.

Запасна частина може зберігатися на різних складах, а на одному складі може зберігатися декілька видів запчастин. Тому між сутностями Запасна частина і Склад існує зв’язок М:N.

Кожен тип запчастин може бути частиною різних замовлень. В одному замовленні може бути декілька типів запчастин. Тому між сутностями Запасна запчастина і Замовлення існує зв’язок М: N.

Клієнт може зробити декілька замовлень, а замовлення може стосуватися тільки одного клієнта. Тому між сутностями Клієнт і Замовлення існує зв’язок 1:N.

Зображення, що містить текст, знімок екрана, Шрифт, Графіка

Автоматично згенерований описРисунок 1 - ER-діаграма, побудована за нотацією Чена

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

**бази даних**

В логічній моделі неможливий безпосередній зв'язок М:N, а в концептуальній моделі він існує між сутностями Запасна частина і Склад, Запасна частина і Замовлення. Для їх представлення було введено допоміжні таблиці Запасна\_частина/Склад та Запасна\_частина/Замовлення.

Назви таблиць і полів було перекладено англійською, для зручності

наведено таблицю нижче.

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| **Spare\_Component –**  *містить дані про запасні запчастини* | **component\_id -** *унікальний ідентифікатор запасної частини*  **provider\_id -** *ідентифікатор постачальника*  **name -** *назва запчастини*  **price -** *ціна запчастини* | **integer** *(числовий)*  **integer** *(числовий)*  **character varying** *(рядок)*  **integer** *(числовий)* |
| **Component/Warehouse –**  *таблиця зв’язку між запчастиною і складом* | **table\_id -** *унікальний ідентифікатор таблиці*  **component \_id -** *ідентифікатор запасної частини*  **warehouse\_id -** *ідентифікатор складу* | **integer** *(числовий)*  **integer** *(числовий)*  **integer** *(числовий)* |
| **Warehouse –** *містить дані про склад* | **warehouse\_id -** *унікальний ідентифікатор складу*  **address -** *адреса складу* | **integer** *(числовий)*  **character varying** *(рядок)* |
| **Provider –**  *містить дані про постачальника* | **provider\_id -** *унікальний ідентифікатор постачальника*  **company\_name** - *назва компанії*  **adress** - *адреса постачальника*  **phone\_number** - *телефонний номер постачальника* | **integer** *(числовий)*  **character varying** *(рядок)*  **character varying** *(рядок)*  **character varying** *(рядок)* |
| **Component/Order –**  *таблиця зв’язку між запчастиною і замовленням* | **comp\_or\_id** *- унікальний ідентифікатор таблиці*  **component\_id -** *ідентифікатор запасної запчастини*  **order\_id -** *ідентифікатор замовлення* | **integer** *(числовий)*  **integer** *(числовий)*  **integer** *(числовий)* |
| **Order –**  *містить дані про замовлення* | **order\_id** *- унікальний ідентифікатор замовлення*  **customer\_id -** *ідентифікатор клієнта*  **total\_cost** – *загальна вартість замовлення*  **order\_date** - *дата замовлення* | **integer** *(числовий)*  **integer** *(числовий)*  **integer** *(числовий)*  **timestamp***(час)* |
| **Сustomer –**  *містить дані про клієнта* | **customer\_id -** *унікальний ідентифікатор клієнта*  **name** *– ПІБ клієнта*    **phone\_number** - *телефонний номер клієнта* | **integer** *(числовий)*  **character varying** *(рядок)*  **character varying** *(рядок)* |

Зображення, що містить знімок екрана, текст, дизайн

Автоматично згенерований опис

Рисунок 2 - Схема бази даних

Схема відповідає 1НФ, тому що:

1. Кожен атрибут у кожному рядку таблиці атомарний.
2. Всі значення в кожному стовпці одного типу.

Схема відповідає 2НФ, тому що:

1. Виконується 1НФ
2. Кожен неключовий атрибут має функціонально залежати від усього ключа, а не від його частини (у нашому випадку всі первинні ключі складаються з одного атрибута, що автоматично забезпечує умову 2НФ).

Схема відповідає 3НФ, тому що:

1. Виконується 2НФ
2. Всі неключові атрибути відношення нетранзитивно залежать від кожного ключа кандидата (тобто взаємозалежність між неключовими атрибутами відсутня).

Зображення, що містить знімок екрана, схема

Автоматично згенерований опис

Рисунок 3 - Схема бази даних у pgAdmin 4

**Таблиці бази даних у pgAdmin4**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Customer**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис

Зображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, знімок екрана

Автоматично згенерований опис У цієї таблиці немає зовнішніх ключів (FK)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Order**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить знімок екрана, текст, програмне забезпечення

Автоматично згенерований описЗображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, знімок екрана

Автоматично згенерований описЗображення, що містить текст, програмне забезпечення, знімок екрана

Автоматично згенерований опис

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Component/Order**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить знімок екрана, текст, Мультимедійне програмне забезпечення, програмне забезпечення

Автоматично згенерований описЗображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, знімок екрана

Автоматично згенерований описЗображення, що містить текст, знімок екрана, програмне забезпечення, Шрифт

Автоматично згенерований опис

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Provider**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить знімок екрана, текст

Автоматично згенерований опис

Зображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, знімок екрана

Автоматично згенерований опис У цієї таблиці немає зовнішніх ключів (FK)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Spare\_Component**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить знімок екрана, текст, програмне забезпечення, Мультимедійне програмне забезпечення

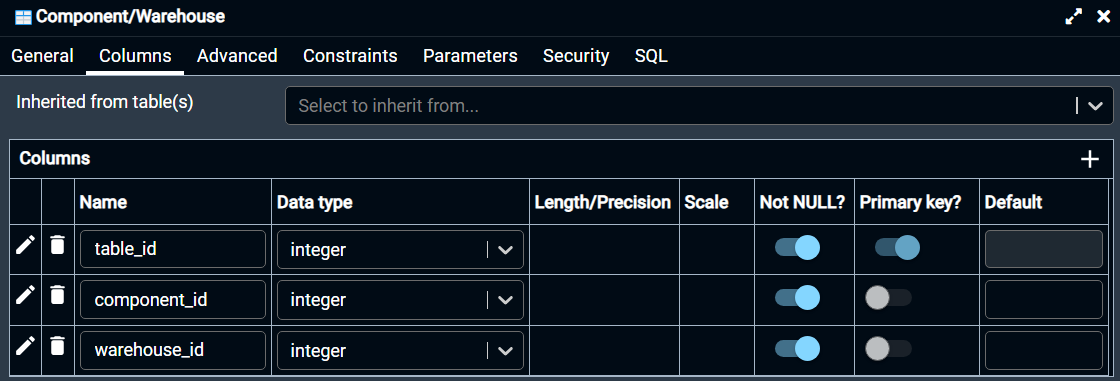
Автоматично згенерований опис

Зображення, що містить текст, програмне забезпечення, знімок екрана

Автоматично згенерований описЗображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, знімок екрана

Автоматично згенерований опис

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Component/Warehouse**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить текст, Шрифт, ряд, знімок екрана

Автоматично згенерований описЗображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, Шрифт

Автоматично згенерований опис

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Warehouse**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зображення, що містить знімок екрана, текст

Автоматично згенерований опис

Зображення, що містить текст, програмне забезпечення, Шрифт, знімок екрана

Автоматично згенерований опис У цієї таблиці немає зовнішніх ключів (FK)

**Таблиці заповнені даними (уривки знімків екрану pgAdmin4)**

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Зображення, що містить текст, Шрифт, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис

Зображення, що містить текст, програмне забезпечення, Мультимедійне програмне забезпечення, Шрифт

Автоматично згенерований опис

Зображення, що містить текст, Шрифт, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис

**Таблиці в коді SQL**

-- Table: public.Customer

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

CREATE TABLE IF NOT EXISTS public."Customer"

(

customer\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

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

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

CONSTRAINT "Customer\_pkey" PRIMARY KEY (customer\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Customer"

OWNER to postgres;

-- Table: public.Order

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

CREATE TABLE IF NOT EXISTS public."Order"

(

order\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

customer\_id integer NOT NULL,

total\_cost numeric NOT NULL,

order\_date timestamp with time zone NOT NULL,

CONSTRAINT "Order\_pkey" PRIMARY KEY (order\_id),

CONSTRAINT "Order\_customer\_id\_fkey" FOREIGN KEY (customer\_id)

REFERENCES public."Customer" (customer\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Order"

OWNER to postgres;

-- Table: public.Component/Order

-- DROP TABLE IF EXISTS public."Component/Order";

CREATE TABLE IF NOT EXISTS public."Component/Order"

(

comp\_or\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

component\_id integer NOT NULL,

order\_id integer NOT NULL,

CONSTRAINT "Component/Order\_pkey" PRIMARY KEY (comp\_or\_id),

CONSTRAINT "Component/Order\_component\_id\_fkey" FOREIGN KEY (component\_id)

REFERENCES public."Spare\_Component" (component\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "Component/Order\_order\_id\_fkey" FOREIGN KEY (order\_id)

REFERENCES public."Order" (order\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Component/Order"

OWNER to postgres;

-- Table: public.Provider

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

CREATE TABLE IF NOT EXISTS public."Provider"

(

"provider\_id " integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

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

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

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

CONSTRAINT "Provider\_pkey" PRIMARY KEY ("provider\_id ")

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Provider"

OWNER to postgres;

-- Table: public.Spare\_Component

-- DROP TABLE IF EXISTS public."Spare\_Component";

CREATE TABLE IF NOT EXISTS public."Spare\_Component"

(

component\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

"provider\_id " integer NOT NULL,

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

price numeric NOT NULL,

CONSTRAINT "Spare\_Component\_pkey" PRIMARY KEY (component\_id),

CONSTRAINT "Spare\_Component\_provider\_id \_fkey" FOREIGN KEY ("provider\_id ")

REFERENCES public."Provider" ("provider\_id ") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Spare\_Component"

OWNER to postgres;

-- Table: public.Component/Warehouse

-- DROP TABLE IF EXISTS public."Component/Warehouse";

CREATE TABLE IF NOT EXISTS public."Component/Warehouse"

(

table\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

component\_id integer NOT NULL,

warehouse\_id integer NOT NULL,

CONSTRAINT "Component/Warehouse\_pkey" PRIMARY KEY (table\_id),

CONSTRAINT "Component/Warehouse\_component\_id\_fkey" FOREIGN KEY (component\_id)

REFERENCES public."Spare\_Component" (component\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "Component/Warehouse\_warehouse\_id\_fkey" FOREIGN KEY (warehouse\_id)

REFERENCES public."Warehouse" (warehouse\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Component/Warehouse"

OWNER to postgres;

-- Table: public.Warehouse

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

CREATE TABLE IF NOT EXISTS public."Warehouse"

(

warehouse\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

adress character varying COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT "Warehouse\_pkey" PRIMARY KEY (warehouse\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Warehouse"

OWNER to postgres;