

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 ІІI курсу

ФПМ групи КВ-02

Однораз Андрій Олександрович

Перевірив:

Київ – 2022

*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 4.

*Зміст звіту*

1. Опис проблемного середовища;

2. Концептуальна модель предметної області;

3. Логічна модель (схема) даних БД;4. Склад СУБД PostgreSQL;

5. Список обмежень цілісності в термінах СУБД PostgreSQL;

6. Фізична модель (схема) даних БД в pgAdmin III;

7. Приклад вмісту БД.

**Опис предметної області «Магазин»**

Обрана предметна область передбачає купівлю товарів різної категорії. Товари певних категорій зберігаються у відділах. Замовлення може містити різні товари з різних категорій.

**Опис сутностей предметної області**

1. Товар (goods), з атрибутами: код товару, назва товару, код категорії. Призначена для збереження інформації про товар у магазині;

2. Відділ (department), з атрибутами: код відділу, назва відділу. Призначена для збереження інформації про відділ, що містить товари певної категорії;

3. Замовлення (order), з атрибутами: код замовлення, код товару, код категорії, ім'я замовника. Призначена для збереження інформації про замовлення;

4. Категорія товарів (category), з атрибутами: код категорії, назва категорії, код відділу. Призначена для збереження інформації про певну категорію товарів.

**Опис зв’язків між сутностями предметної області**

Сутність “Товар” має зв’язок М:N по відношенню до сутності “Замовлення” оскільки замовлення може містити декілька товарів, а товар може бути у кількох замовленнях.

Сутність “Категорія товарів” має зв'язок 1:N по відношенню до товару, оскільки товар може відноситися тільки до однієї з категорій, але до однієї категорії входять різні товари. Також ця сутність має зв'язок M:N по відношенню до сутності “Замовлення” оскільки у замовленні можуть бути товари різних категорій, а одна категорія товару може бути присутньою у декількох замовленнях.

Сутність “Відділ” має зв'язок 1:N по відношенню до категорії товарів, оскільки кожна категорія товарів має лише один ввідділ, але в одному відділі можуть бути присутні кілька категорій.

**Концептуальна модель предметної області**

Концептуальна модель наведена на рисунку 1.

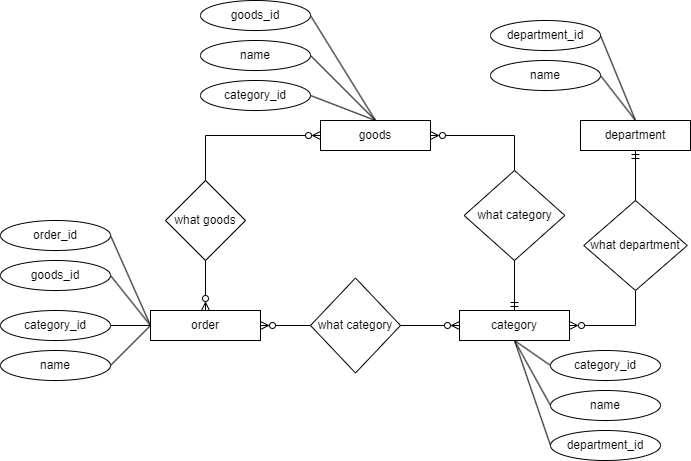


Рисунок 1 - ER-діаграма, побудована за нотацією Чена (інструмент: draw.io)

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

Сутність “Товар” перетворено у таблицю “goods”.

Сутність “Відділ” перетворено у таблицю “department”.

Сутність “Замовлення” перетворено у таблицю “order”.

Зв'язок М:N між сутностями “Товар” та “Замовлення” зумовив появу таблиці “goods/order” з зовнішніми ключами goods\_id та order\_id.

Сутність “Категорія товару” перетворено у таблицю “category”.

Зв'язок М:N між сутностями “Категорія” та “Замовлення” зумовив появу таблиці “category/order” з зовнішніми ключами category\_id та order\_id.

**Логічна модель (схема) БД «Магазин»**

Логічну модель (схему бази даних) наведено на рисунку 2.

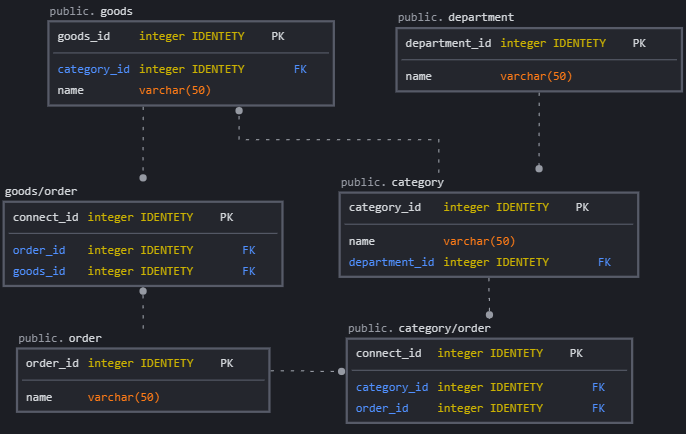


Рисунок 2 - Схема бази даних (інструмент: sqldbm.com)

**Опис об’єктів бази даних у вигляді таблиці**

Таблиця 1 - Опис структури бази даних “Магазин”

| Сутність | Атрибут | Тип атрибуту |
| --- | --- | --- |
| **goods** – містить всі товари що є у наявності | **goods\_id –** унікальний ідентифікатор товару  **name –** назва товару. Не допускає NULL  **category\_id –** унікальний ідентифікатор категорії | **integer** (числовий)  **character varying** (рядок)  **integer** (числовий) |
| **department** – містить інформацію про відділ. | **department\_id –** унікальний ідентифікатор відділу  **name –** назву відділу. Не допускає NULL | **integer** (числовий)  **character varying** (рядок) |
| **order** – містить інформацію про замовлення | **order\_id –** унікальний ідентифікатор замовлення  **name –** ім’я замовника. Не допускає NULL | **integer** (числовий)  **character varying** (рядок) |
| **goods/order** – (додаткова сутність) містить ідентифікатори товару і замовлення | **connect\_id** – унікальний ідентифікатор звязку  **goods\_id –** унікальний ідентифікатор товару  **order\_id –** унікальний ідентифікатор замовлення | **integer** (числовий)  **integer** (числовий) |
| **category** – містить інформацію про категорію товару | **category\_id –** унікальний ідентифікатор категорії  **name –** назва категорії. Не допускає NULL  **department\_id –** унікальний ідентифікатор відділу | **integer** (числовий)  **character varying** (рядок)  **integer** (числовий) |
| **category/order** – (додаткова сутність) містить ідентифікатори категорії і замовлення | **connect\_id** – унікальний ідентифікатор зв'язку  **category\_id –** унікальний ідентифікатор категорії  **order\_id –** унікальний ідентифікатор замовлення | **integer** (числовий)  **integer** (числовий) |

**Функціональні залежності для кожної таблиці**

GOODS:

order\_id → name (назва залежить від коду товару)

DEPARTMENT:

department\_id → name (назва залежить від коду відділу)

ORDER:

order\_id → name (ім’я залежить від коду замовлення)

GOODS/ORDER:

connect\_id → order\_id (код замовлення залежить від ідентифікатора зв'язку)

connect\_id → goods\_id (код товару залежить від ідентифікатора зв'язку)

CATEGORY:

category\_id → name (назва залежить від коду категорії)

CATEGORY/ORDER:

connect\_id → order\_id (код замовлення залежить від ідентифікатора зв'язку)

connect\_id → category\_id (код категорії залежить від ідентифікатора зв'язку)

**Відповідність схеми бази даних нормальним формам:**

Схема відповідає 1НФ, тому що в таблиці немає дубльованих рядків, у кожній комірці зберігається атомарне значення, у кожному стовпчику зберігаються дані одного типу.

Схема відповідає 2НФ, тому що вона відповідає 1НФ, має первинний ключ від якого залежать всі не ключові стовпчики таблиці.

Схема відповідає 3НФ, тому що вона відповідає 2НФ і всі не ключові атрибути таблиці залежать від первинного ключа.

**Таблиці бази даних у pgAdmin4**

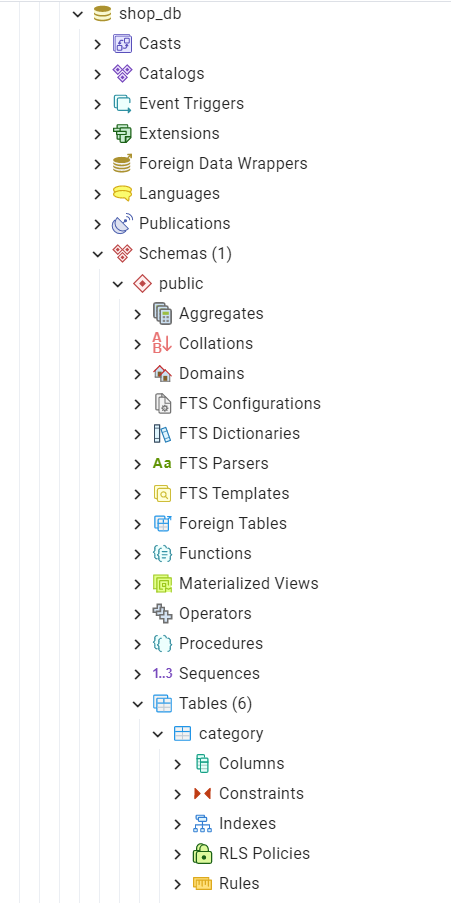
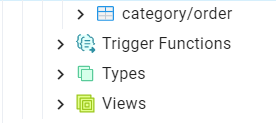
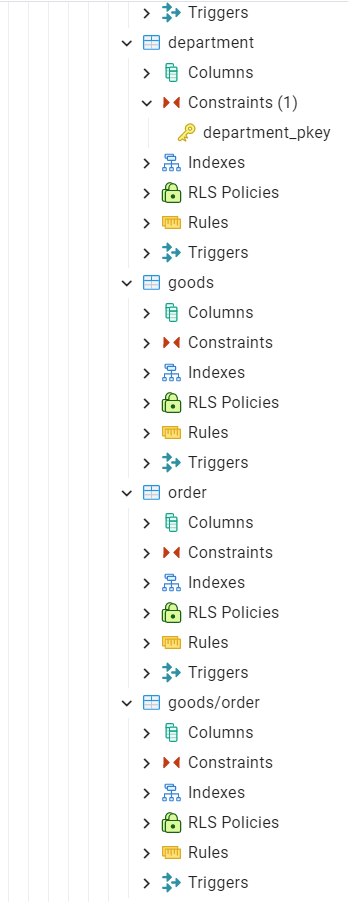


Рисунок 3 - Схема бази даних у pgAdmin 4

**Фотографії вмісту таблиць**

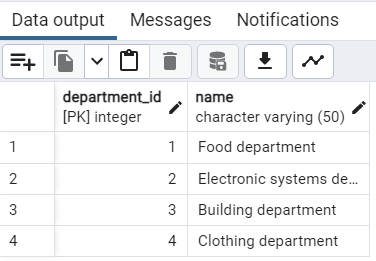


Рисунок 4.1 - Дані таблиці Department

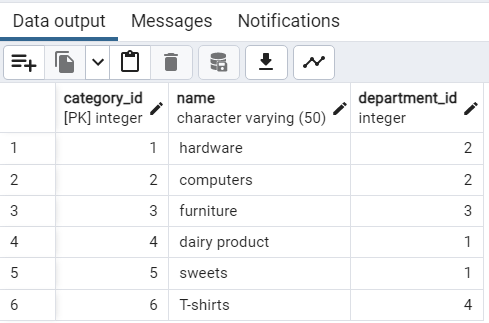


Рисунок 4.2 - Дані таблиці Category

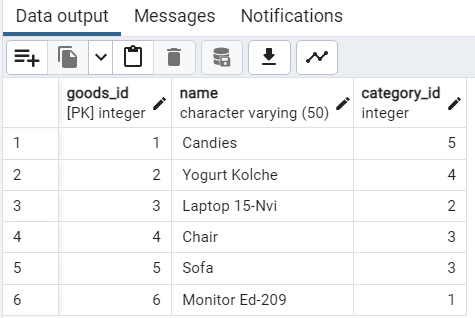


Рисунок 4.3 - Дані таблиці Goods

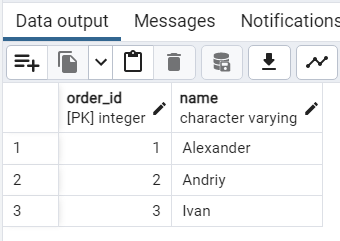


Рисунок 4.4 - Дані таблиці Order

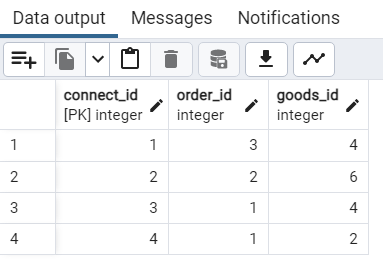


Рисунок 4.5 - Дані таблиці goods\_order

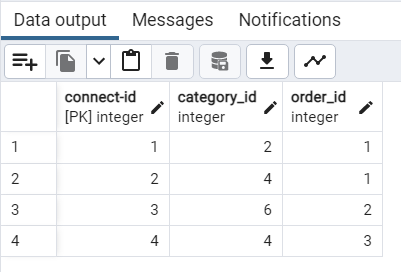


Рисунок 4.6 - Дані таблиці category\_order

**SQL-текст опису БД**

*-- Database: shop\_db*

*-- DROP DATABASE IF EXISTS shop\_db;*

CREATE DATABASE shop\_db

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'Ukrainian\_Ukraine.1251'

LC\_CTYPE = 'Ukrainian\_Ukraine.1251'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1

IS\_TEMPLATE = FALSE;

*-- Table: public.category*

*-- DROP TABLE IF EXISTS public.category;*

CREATE TABLE IF NOT EXISTS PUBLIC.category

(

category\_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,

department\_id INTEGER NOT NULL,

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

CONSTRAINT category\_department FOREIGN KEY (department\_id)

REFERENCES PUBLIC.department (department\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS PUBLIC.category

OWNER TO postgres;

*-- Table: public.department*

*-- DROP TABLE IF EXISTS public.department;*

CREATE TABLE IF NOT EXISTS PUBLIC.department

(

department\_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,

CONSTRAINT department\_pkey PRIMARY KEY (department\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS PUBLIC.department

OWNER TO postgres;

*-- Table: public.goods*

*-- DROP TABLE IF EXISTS public.goods;*

CREATE TABLE IF NOT EXISTS PUBLIC.goods

(

goods\_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,

category\_id INTEGER NOT NULL,

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

CONSTRAINT goods\_category FOREIGN KEY (category\_id)

REFERENCES PUBLIC.category (category\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS PUBLIC.goods

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 ),

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

CONSTRAINT order\_pkey PRIMARY KEY (order\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS PUBLIC."order"

OWNER TO postgres;

*-- Table: public.goods/order*

*-- DROP TABLE IF EXISTS public."goods/order";*

CREATE TABLE IF NOT EXISTS PUBLIC."goods/order"

(

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

order\_id INTEGER NOT NULL,

goods\_id INTEGER NOT NULL,

CONSTRAINT "goods/order\_pkey" PRIMARY KEY (connect\_id),

CONSTRAINT "goods/order\_goods" FOREIGN KEY (goods\_id)

REFERENCES PUBLIC.goods (goods\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "goods/order\_order" 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."goods/order"

OWNER TO postgres;

*-- Table: public.category/order*

*-- DROP TABLE IF EXISTS public."category/order";*

CREATE TABLE IF NOT EXISTS PUBLIC."category/order"

(

"connect-id" INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

category\_id INTEGER NOT NULL,

order\_id INTEGER NOT NULL,

CONSTRAINT "category/order\_pkey" PRIMARY KEY ("connect-id"),

CONSTRAINT "category/order\_category" FOREIGN KEY (category\_id)

REFERENCES PUBLIC.category (category\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "category/order\_order" 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."category/order"

OWNER TO postgres;