

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-02

Шевченко І.І.

Перевірив:

Київ – 2022

*Метою роботи* є здобуття 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 галузь передбачає замовлення і купівлю товарів у різні відділи магазину. Кожен магазин має різну кількість відділів , які потребують різні типи товарів , тому кожен відділ окремо закуповує необхідну кількість товарів. Магазин , відділ , замовлення і товар мають свій унікальний номер.

**Опис сутностей предметної області**

1. Магазин, з атрибутами: “Номер магазину” , “Назва магазину”. Призначено для зберігання інформації про наявні магазини.
2. Відділ, з атрибутами: “Категорія товарів” , “Номер відділу”.

Призначено для зберігання інформації про відділи в конкретному магазині.

1. Замовлення, з атрибутами: “ Номер замовлення”. Містить інформацію про замовлення , що робляться відділом.
2. Товар, за атрибутами: “Номер товару”, “Тип товару”, “ кількість товару”. Сутність призначення для зберігання інформації про товар.

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

1. Сутність “Магазин” має зв’язок 1:N по відношенню до відділів , бо в одному магазині може будет кілька відділів , але не навпаки.
2. Сутність “Відділ” має зв’язок 1:N по відношенню до замовлень , бо кожен відділ може зробити багато різних замовлень, але кожне замовлення відноситься до конкретного відділу.
3. Сутність “Замовлення” має зв’язок М:N по відношенню до товарів, бо багато замовлень можуть мати багато товарів.

**Концептуальна модель предметної області**

**“Магазин”**

Концептуальна модель наведена на рисунку 1.

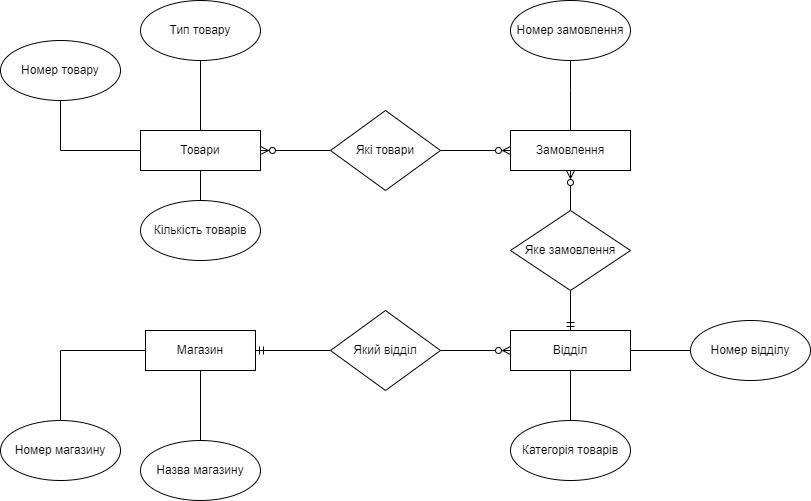


Рисунок 1 - ER-діаграма

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

Сутність “Магазин” перетворено на таблицю “shop” із зовнішніми ключами “shop\_id” і “shop\_name”.

Сутність “Відділ” перетроверно на таблицю “department” із зовнішніми ключами “department\_id” і “department\_type”.

Сутність “Замовлення” перетроверно на таблицю “order" із зовнішнім ключем “order\_id”.

Сутність “Товари” перетроверно на таблицю “products” із

зовнішніми ключами “product\_id”, “product\_type” і “quantity”.

Створено додаткову таблицю «product/order» через зв’язок N:M.

**Логічна модель (схема) БД « Сервіс продажу залізничних квитків »**

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

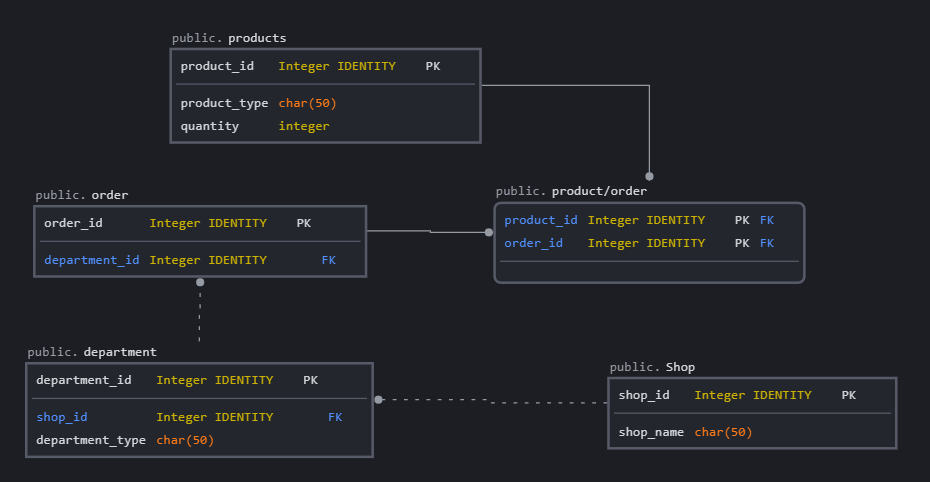


Рисунок 2 - Схема бази даних

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

| **Сутність** | **Атрибут** | **Тип атрибуту** |
| --- | --- | --- |
| **Shop** - містить інформацію про магазин | **Shop\_id** - унікальний ідентифікатор магазину  **Shop\_name** - назва магазину | **Integer***(числовий)*  **Char** *(символьний)* |
| **Department** - містить інформацію про відділи магазина | **department\_id** - унікальний ідентифікатор відділу  **shop\_id** - унікальний ідентифікатор магазину  **department\_type** - тип відділу (які товари в ньому знаходятся) | **Integer***(числовий)*  **Integer***(числовий)*  **Char** *(символьний)* |
| **Order** - містить інформацію про замовлення , що робляться відділом | **order\_id** - унікальний ідентифікатор замовлення  **department\_id** - унікальний ідентифікатор відділу | **Integer***(числовий)*  **Integer***(числовий)* |
| **Products** - містить інформацію про товари | **product\_id** - унікальний ідентифікатор товару  **quantity** - кількість товару  **product\_type** - тип товару | **Integer***(числовий)*  **Integer***(числовий)*  **Char** *(символьний)* |
| **Product/order** - проміжна таблиця, містить , що містить id замовлень та товарів | **product\_id** - унікальний ідентифікатор товару  **order\_id** - унікальний ідентифікатор замовлення | **Integer***(числовий)*  **Integer***(числовий)* |

**Фізична модель БД «Магазин»**

**у pgAdmin4**

SHOP:

Shop\_id → shop\_name ( Назва магазину залежить від його коду)

Department:

department\_id → shop\_id, department\_type

department\_id → department\_type ( тип відділу залежить від його коду)

department\_id → shop\_id ( код магазину залежить від коду відділу)

Order:

order\_id → department\_id ( код відділу залежить від коду замовлення)

Products:

product\_id → product\_type , quantity

Product\_id → product\_type ( тип продукта залежить від його коду)

product\_id → quantity (кількість товарів залежить від його коду)

Product/order:

Product\_id → order\_id

order\_id → product\_id

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

Схема відповідає 1НФ, тому що:

* Значення в кожній комірці є атомарними
* Кожний рядок є унікальним

Схема відповідає 2НФ, тому що:

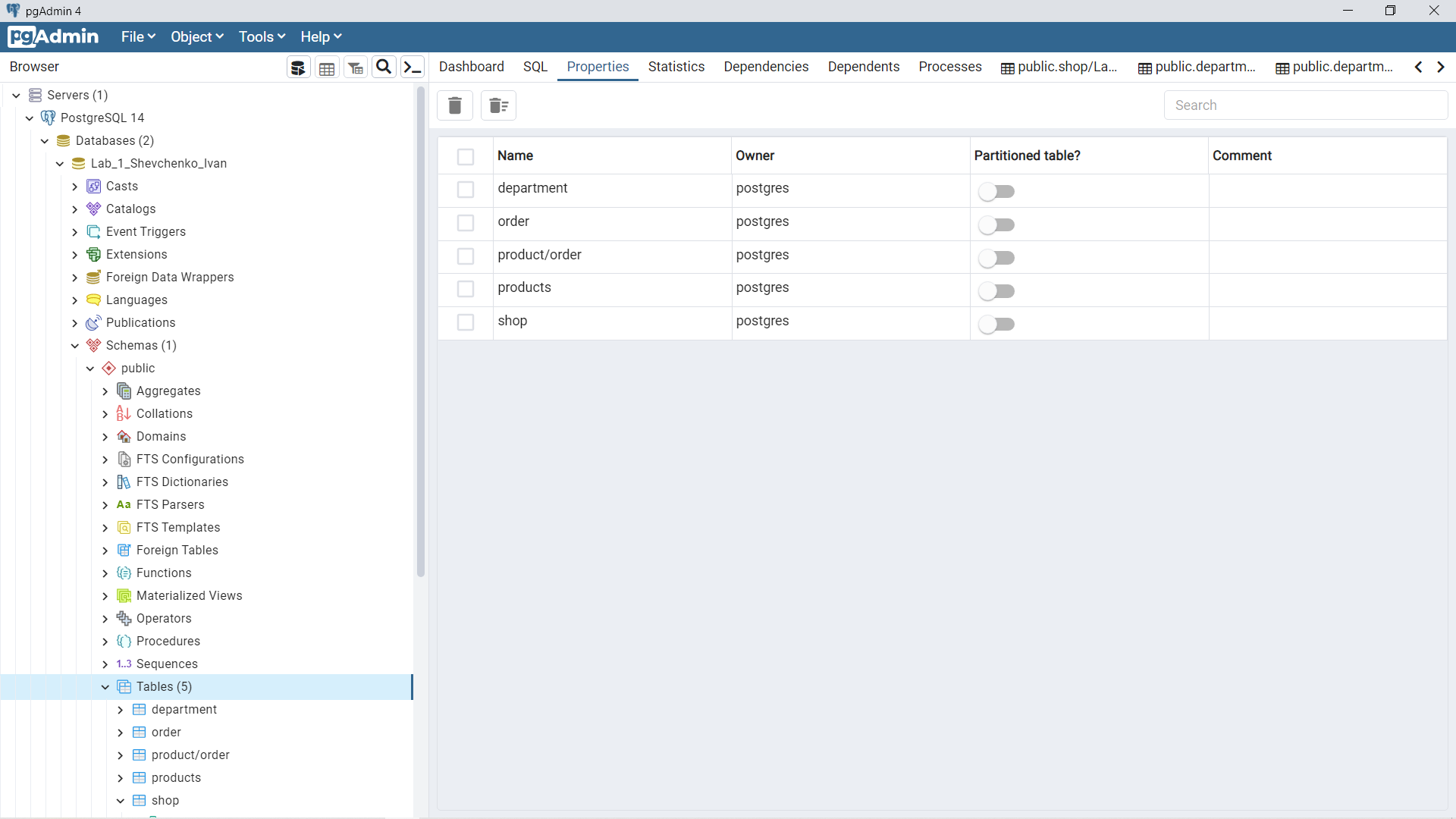
* Дотримується вимог 2НФ
* Неключовий атрибут в списку функціональних залежностей залежить від усього переліку ключа

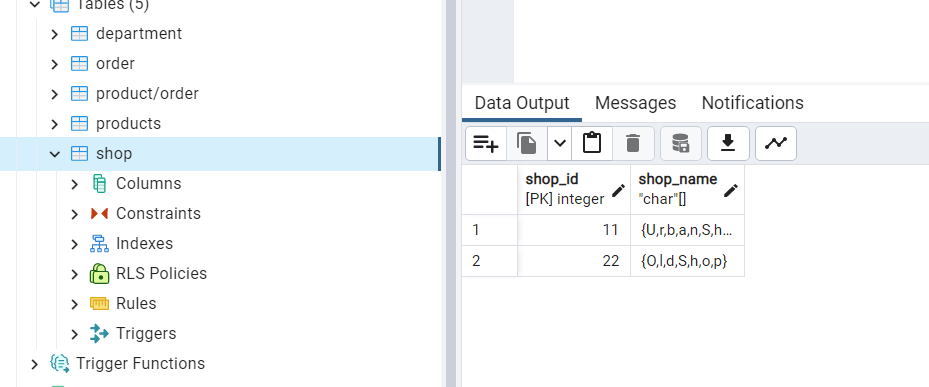
Схема відповідає 3НФ, тому що:

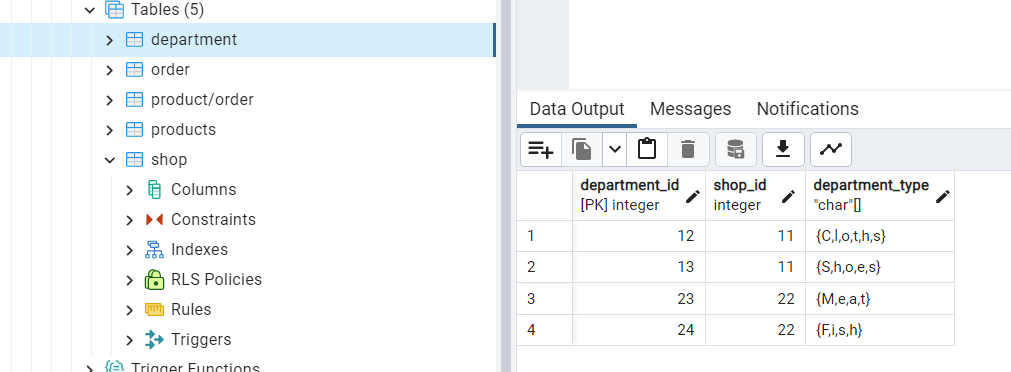
* Відповідає 2НФ
* Кожен неключовий атрибут не є транзитивно залежним від кожного кандидатного ключа

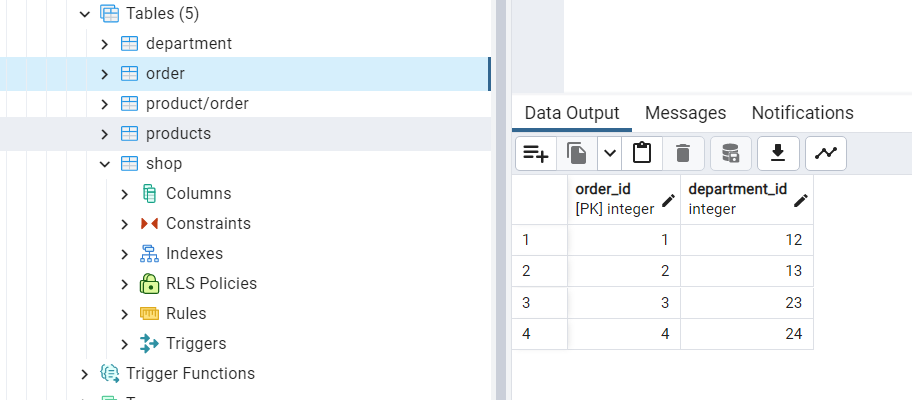
**Фізична модель БД «Магазин»**

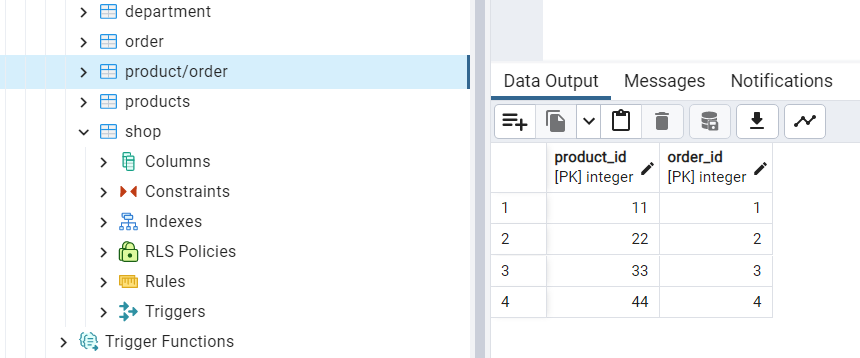
**у pgAdmin4**

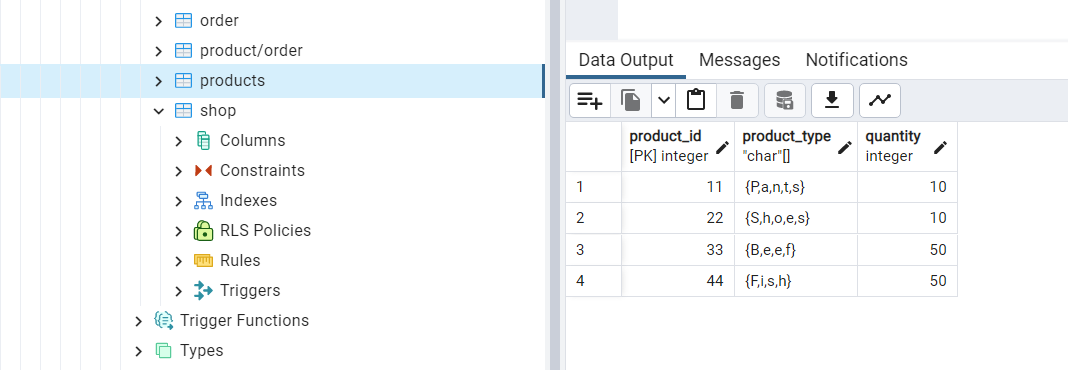
****

****

****

****

****

****

**SQL-текст опису БД «Магазин»**

CREATE DATABASE "Lab\_1\_Shevchenko\_Ivan"

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'Russian\_Russia.1251'

LC\_CTYPE = 'Russian\_Russia.1251'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;

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 ),

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

department\_type "char"[],

CONSTRAINT department\_pk\_id PRIMARY KEY (department\_id),

CONSTRAINT shop\_fk\_id FOREIGN KEY (shop\_id)

REFERENCES public.shop (shop\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.department

OWNER to postgres;

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 ),

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

CONSTRAINT order\_pk\_id PRIMARY KEY (order\_id),

CONSTRAINT department\_fk\_id 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."order"

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public."product/order"

(

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

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

CONSTRAINT "product/order\_pk\_id" PRIMARY KEY (product\_id, order\_id),

CONSTRAINT "order/product\_fk\_id" FOREIGN KEY (order\_id)

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

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "product/order\_fk\_id" FOREIGN KEY (product\_id)

REFERENCES public.products (product\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."product/order"

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public.products

(

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

product\_type "char"[],

quantity integer,

CONSTRAINT product\_pk\_id PRIMARY KEY (product\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.products

OWNER to postgres;

CREATE TABLE IF NOT EXISTS public.shop

(

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

shop\_name "char"[],

CONSTRAINT shop\_pk\_id PRIMARY KEY (shop\_id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.shop

OWNER to postgres;