

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота №3**

з дисципліни

**«Бази даних і засоби управління»**

## Тема: «Засоби оптимізації роботи СУБД PostgreSQL»

Виконав: студент ІІI курсу

ФПМ групи КВ-84

Іванюк В. І.

Перевірив:

Київ – 2020

*Метою роботи* є здобуття практичних навичок використання засобів оптимізації СУБД PostgreSQL.

*Завдання* роботи полягає у наступному:

1. Перетворити модуль “Модель” з шаблону MVC лабораторної роботи №2 у вигляд об’єктно-реляційної проекції (ORM).
2. Створити та проаналізувати різні типи індексів у PostgreSQL.
3. Розробити тригер бази даних PostgreSQL.

*Вимоги до пункту завдання №1*

Для перетворення функцій, що реалізують запити до об’єктної бази даних, необхідно встановити бібліотеку sqlAlchemy, налаштувати програму на роботу з ORM, розробити класи-сутності для об’єктів-сутностей, представлених відповідними таблицями БД та пов’язаних зв’язками 1:М, М:М та 1:1 виконати опис схеми бази даних. Особливу увагу приділити контролю зовнішніх зв’язків між таблицями засобами ORM.

Замінити виклики запитів мовою SQL на відповідні запити засобами SQLAlchemy по роботі з об’єктами. Обов’язковим є реалізація вставки, вилучення та редагування екземплярів класів-сутностей. Розробка запитів на генерацію даних та пошук екземплярів класів-сутностей вітається, але не є обов’язковою.

Інтерфейси функцій (вхідні та вихідні аргументи функцій модуля “Модель”) мають залишитись без змін.

*Вимоги до пункту завдання №2*

Відповідно до варіанту індексування продемонструвати на прикладах запитів SQL SELECT підвищення швидкодії їх виконання з використанням індексів, а також пояснити чому для деяких випадків індексування використовувати недоцільно. При цьому для наочного представлення слід використати функцію генерування рандомізованих даних з лабораторної роботи №2, створивши необхідну кількість тестових даних. Навести 4-5 прикладів запитів SELECT (із виведенням результуючих даних), що містять фільтрацію, агрегатні функції, групування та сортування (у необхідних комбінаціях).

*Вимоги до пункту завдання №3*

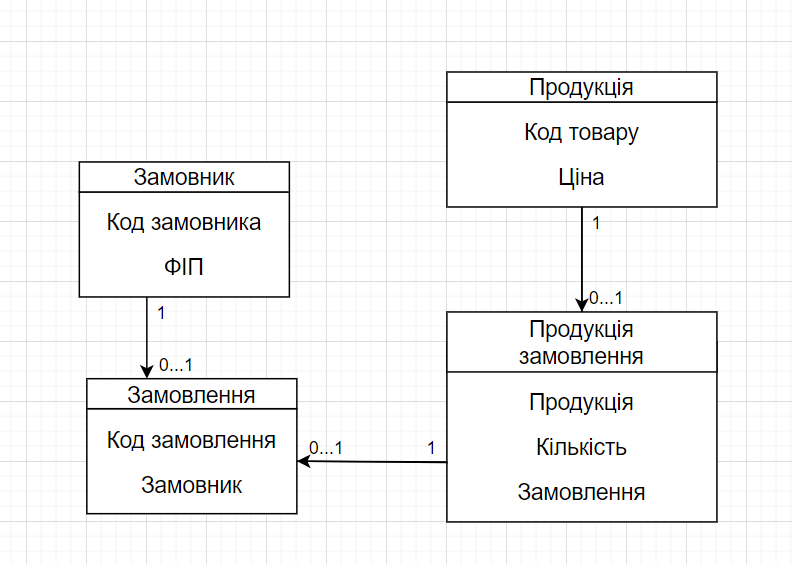
Створити тригер бази даних PostgreSQL відповідно до варіанта. Тригерна функція має включати обробку запису, що модифікується (вставляється або вилучається), умовні оператори, курсорні цикли та обробку виключних ситуацій. Виконати відлагодження тригера при різних вхідних даних, навівши 2-3 приклади його використання.

Варіант 13

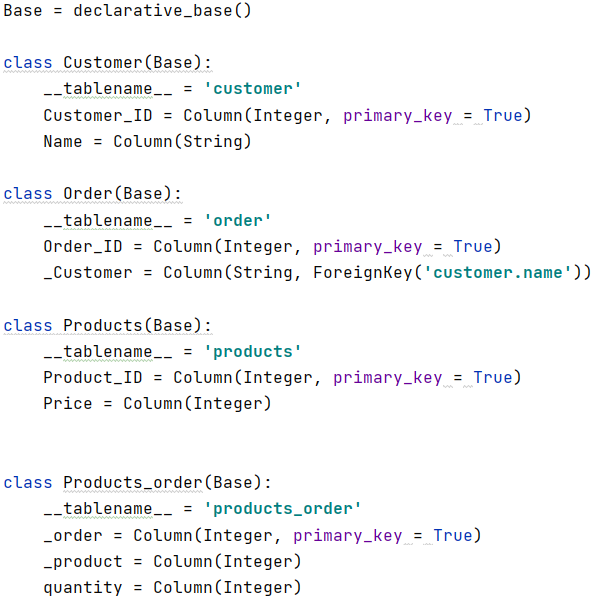


**Пункт №1**

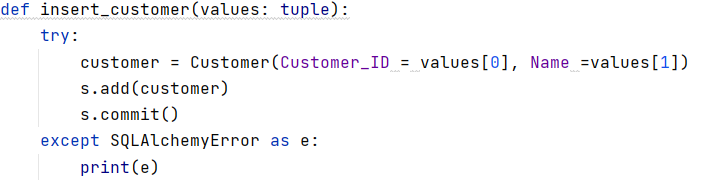
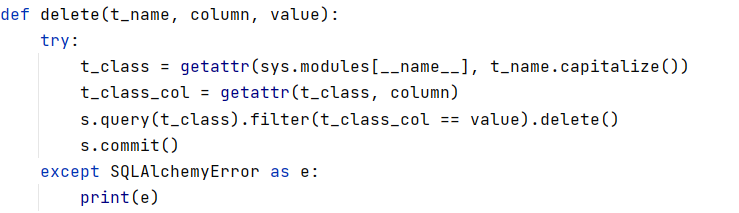
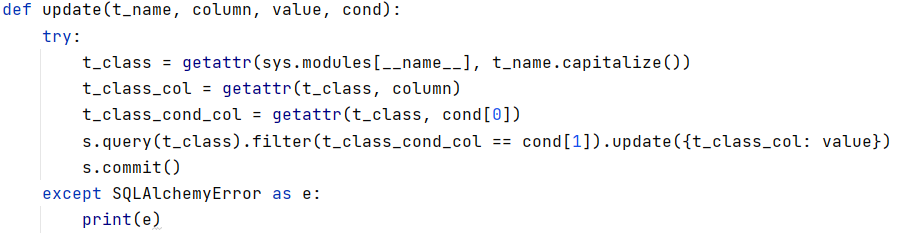
**Схема бази даних у вигляді таблиць та зв'язків між ними**



**Класи в ORM :**



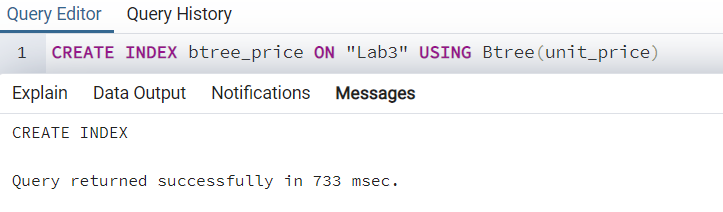
**Приклади запитів :**

1. **Вставки**
2. **Видалення**
3. **Редагування**

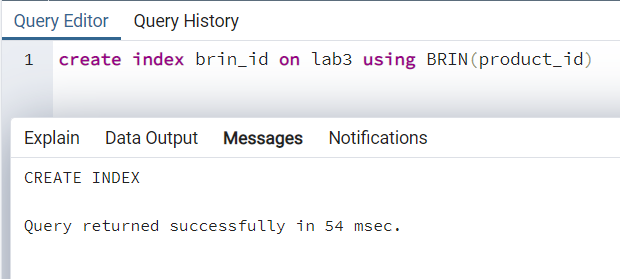
**Пункт №2**

**Створення індексу**

**Btree:**

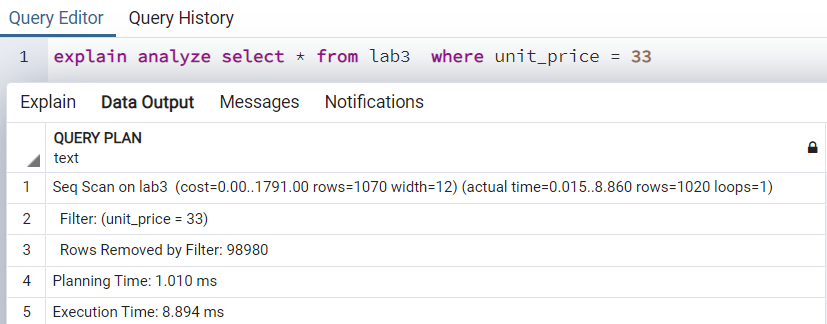


**BRIN:**

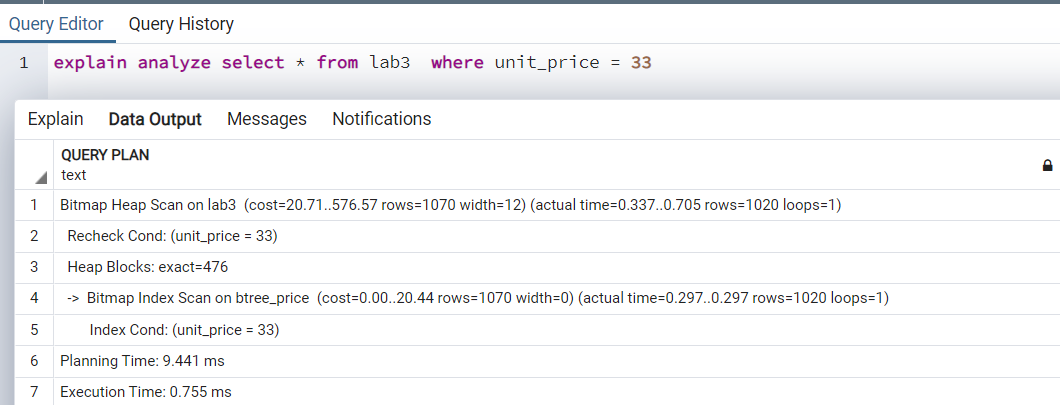


**Приклад фільтрації даних**

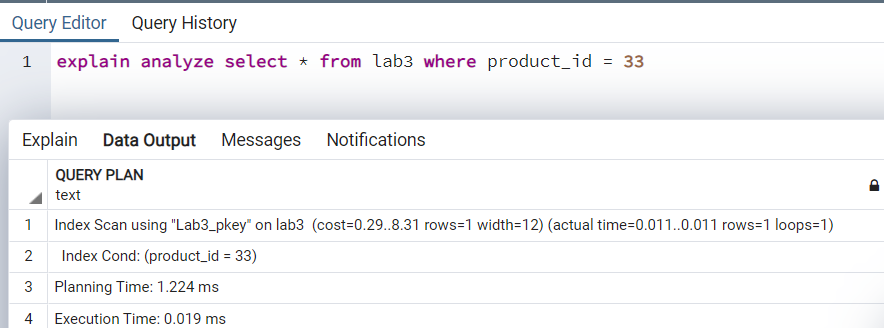
Без індексації



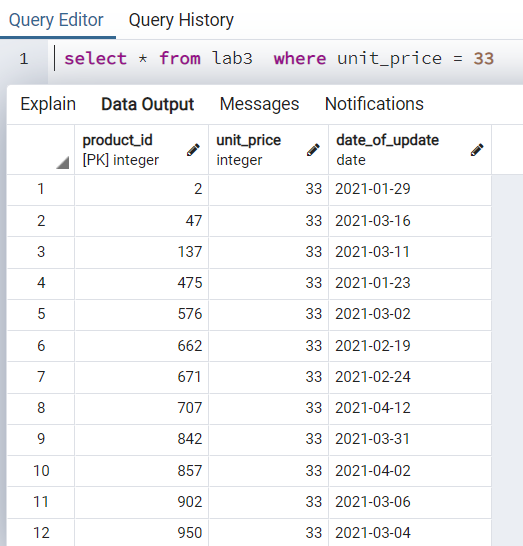
Btree



BRIN

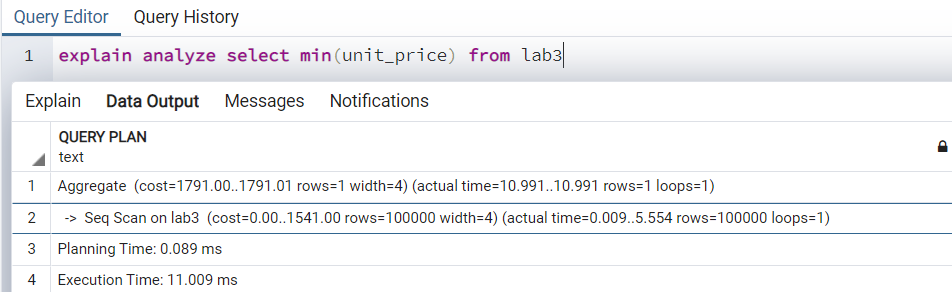


Результат

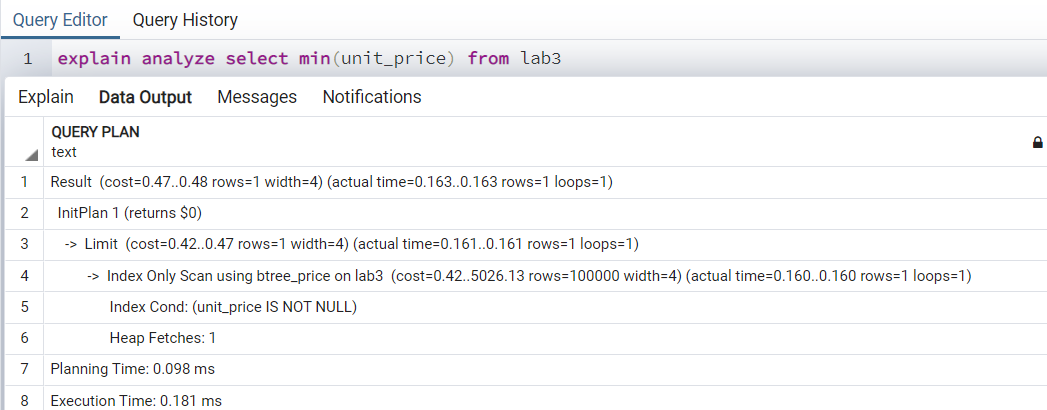


**Приклад з агрегатними функціями**

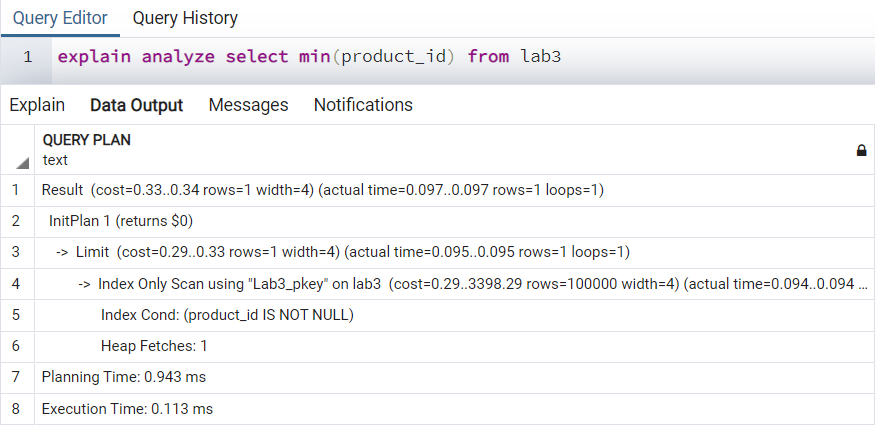
Без індексації



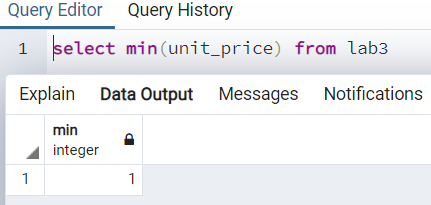
Btree



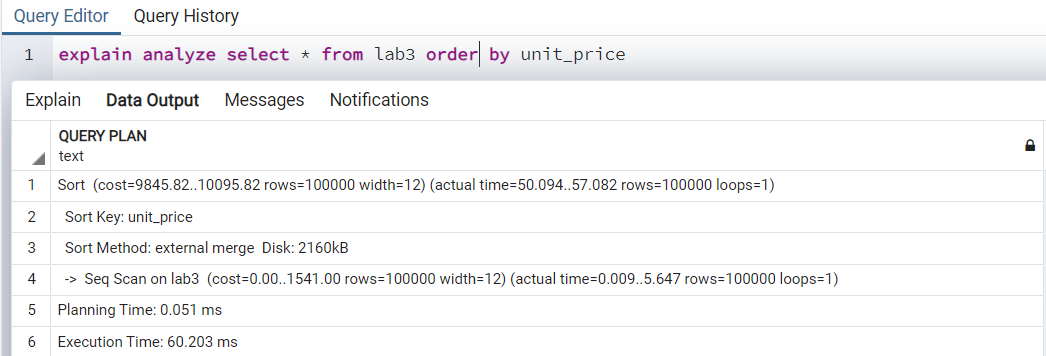
BRIN



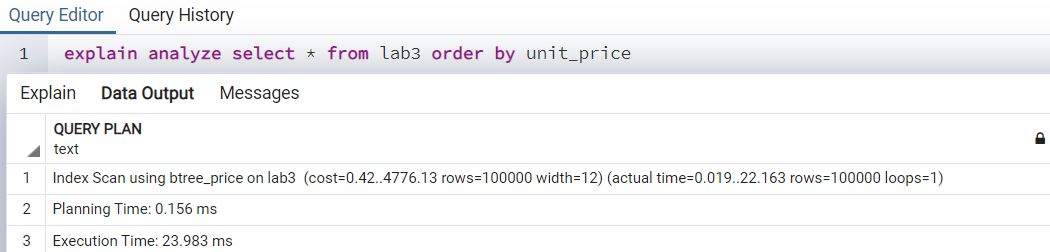
Результат



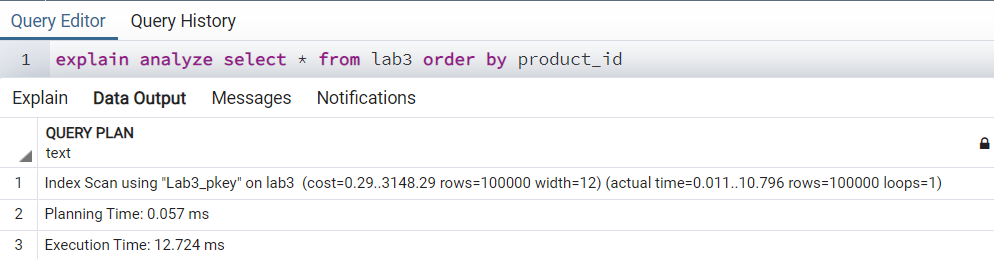
**Приклад з сортуванням**

Без індексації 

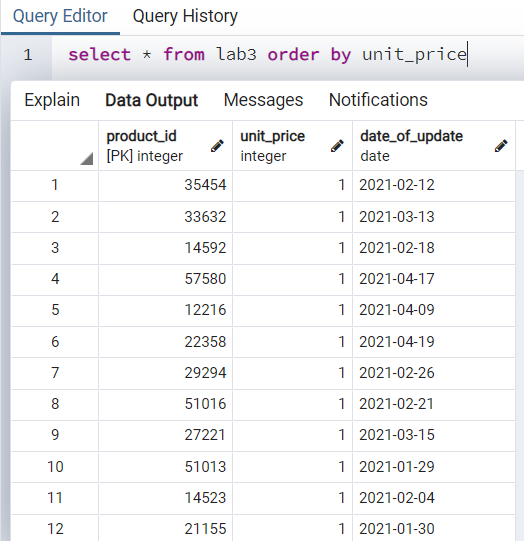
Btree



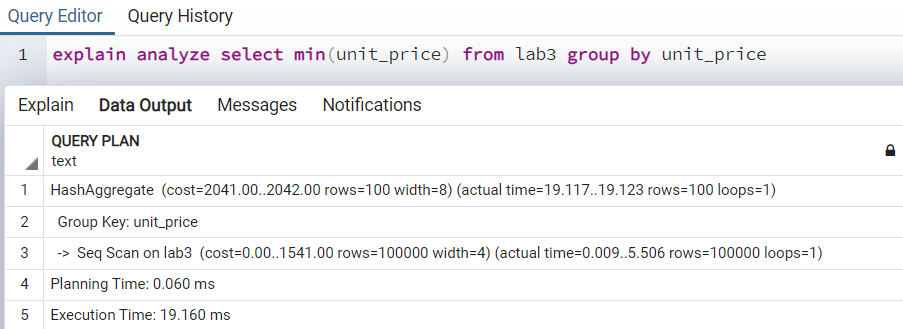
BRIN



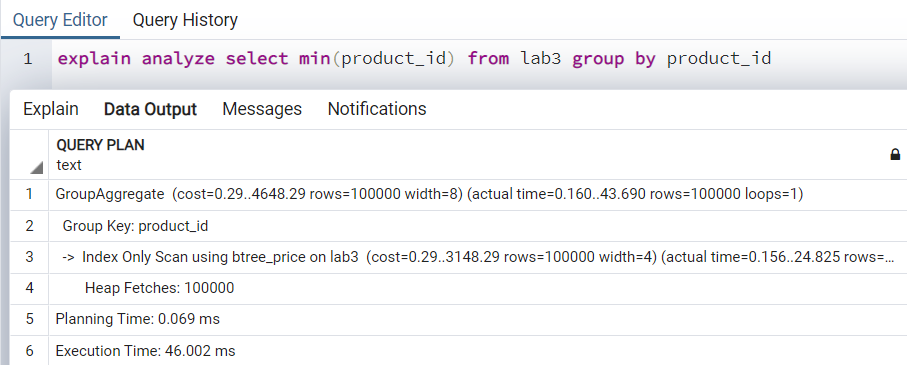
Результат



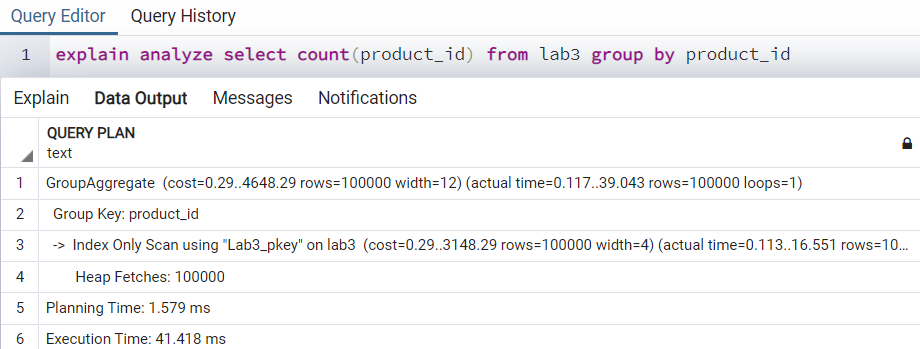
**Приклад з агрегатними функціями та сортуванням**

Без індексації 

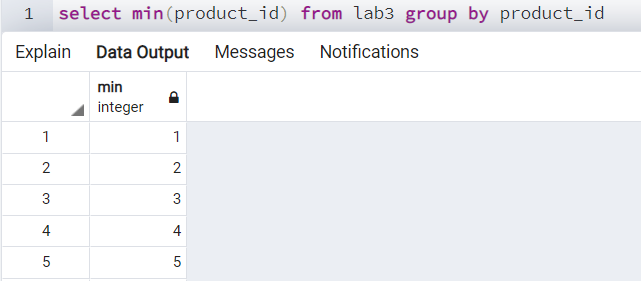
Btree



BRIN



Результат

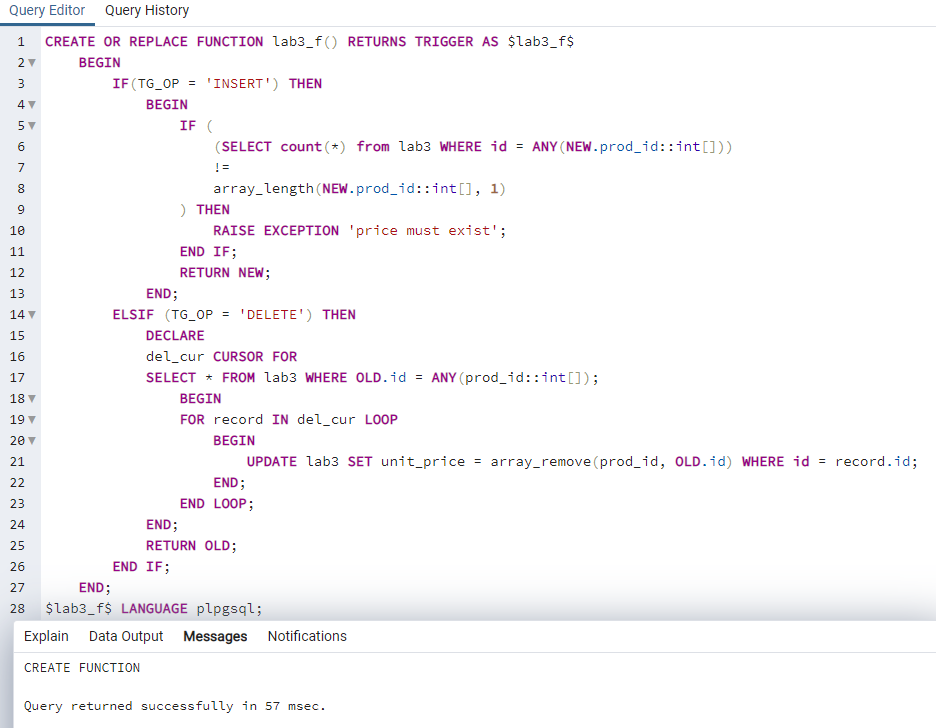


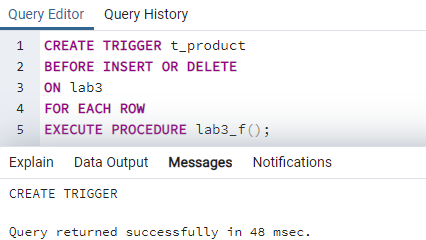
Висновки :

1. При фільтрації обидва індекси чудово виконують свою роботу в порівняні з послідовним пошуком при відсутності індексів, хоча BRIN спрацював швидше.
2. При використанні агрегатних функцій, використання індексів залежить від агрегатної функції. Наприклад, використання функції min дає змогу використовувати Btree індекс та збільшує швидкість пошуку оскільки необхідно лише дійти до крайнього лівого «листка». Використання функції count не дає змогу використати Btree індекс тому що в результаті маємо лише один запис. Але при цьому індекс BRIN може працювати при використанні цієї функції.
3. При сортуванні запити з індексами працюють ефективніше ніж без них. BRIN спрацював швидше, ніж Btree.
4. При сортуванні та використанні агрегатних функцій обидва індекси виявились неефективними та спрацювали повільніше аніж запит без індеків.

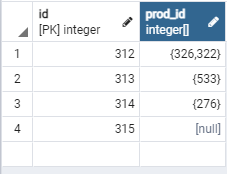
Пункт №3

Команди, що ініціюють виконання тригера, код тригера:

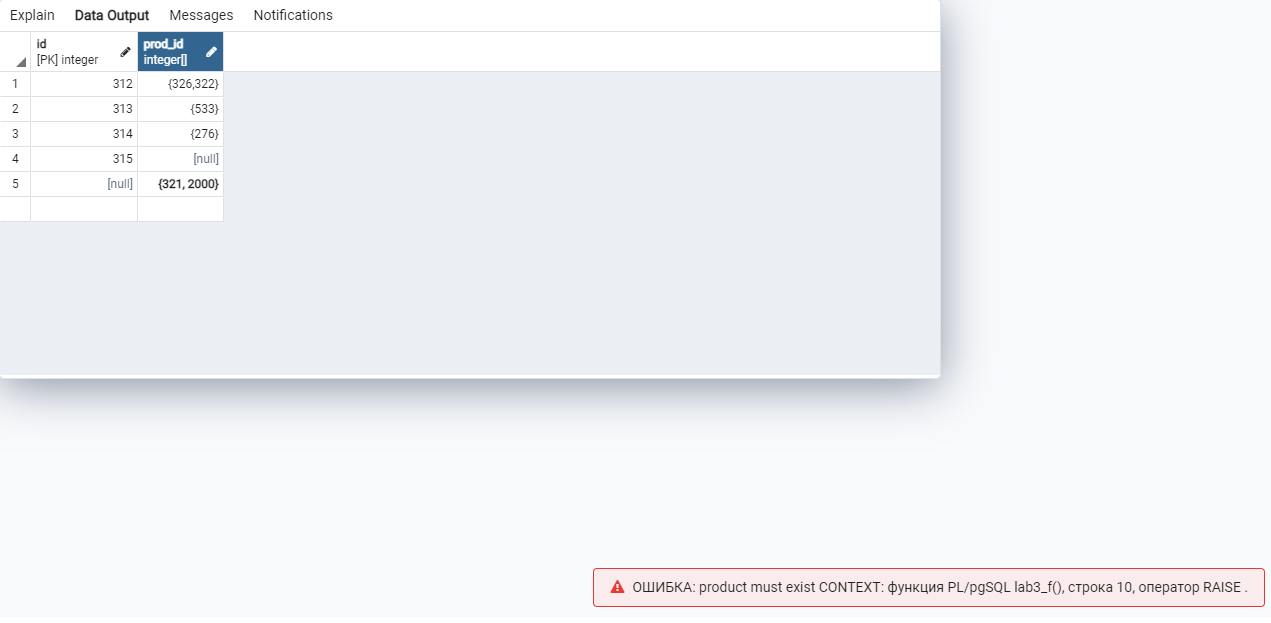


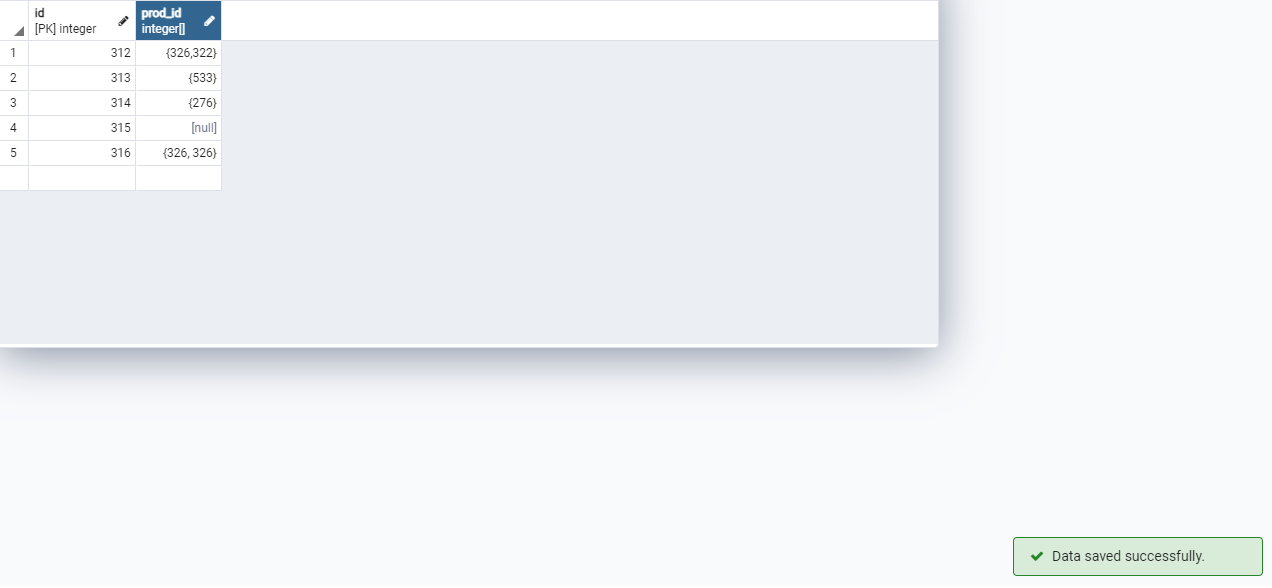


Таблиця :



1. Якщо додається новий запис до таблиці, то усі id що містяться у prod\_id мають існувати.





2. При видаленні запису, записи що містили у prod\_id id видаляємого запису, змінять вмість prod\_id на такий самий масив, але вже без id видаляємого запису

