

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»**

Виконав: студент ІII курсу

ФПМ групи КВ-83

Трофімцов Дмитро

Київ – 2020

**Лабораторна робота № 3.**

Засоби оптимізації роботи СУБД PostgreSQL

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

Завдання роботи полягає у наступному:

Перетворити модуль “Модель” з шаблону MVC лабораторної роботи №2 у вигляд об’єктно-реляційної проекції (ORM).

Створити та проаналізувати різні типи індексів у PostgreSQL.

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.

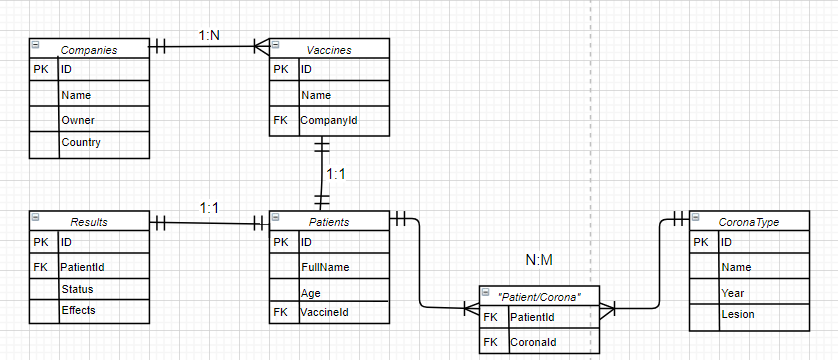
Вимоги до інструментарію

1. Бібліотека для реалізації ORM - [SQLAlchemy для Python](https://www.sqlalchemy.org/) або інша з подібною функціональністю.
2. Середовище для відлагодження SQL-запитів до бази даних – pgAdmin 4.
3. СУБД - PostgreSQL 11-12.

Варіант 26 (номер залікової книжки 8126)

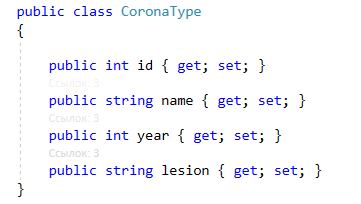
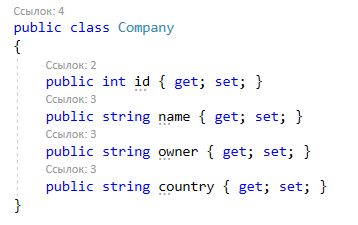
**Завдання 1**

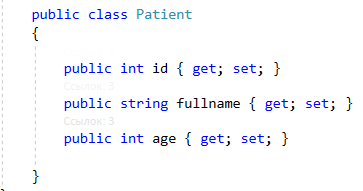
Логічна схема бази даних “Лікарня”

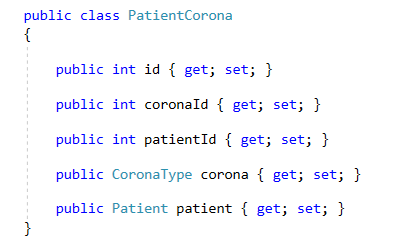


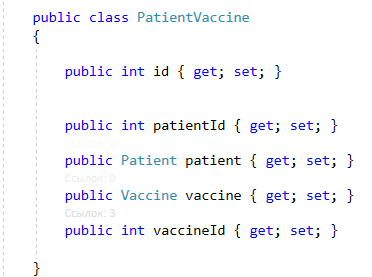
Моделі ORM

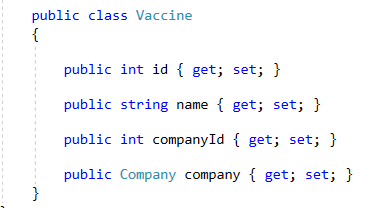
Приклади запитів у вигляді ORM





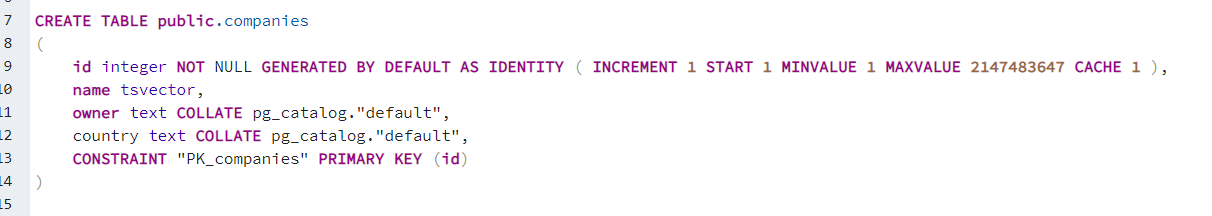


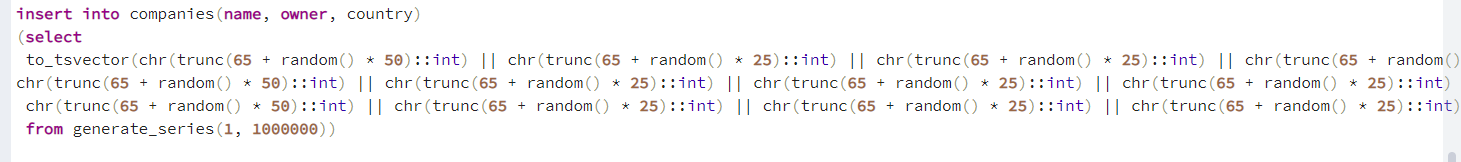






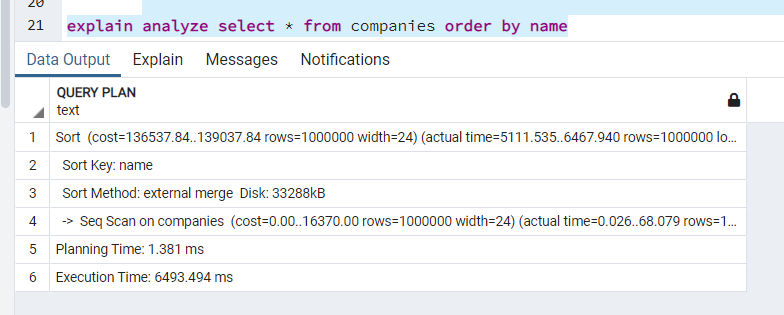
Також програма містить так званий ‘клас контексту’, який відповідає за створення бази даних. В ньому ''регіструються'' всі моделі таблиць бази даних. Відношення створюються автоматично.

**Завдання 2**



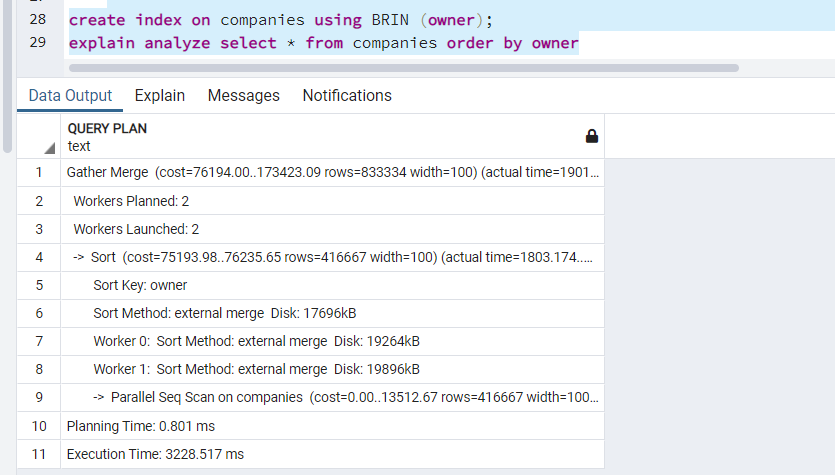
Перш за все я змінив тип даних name на tsvector для того, щоб можна було працювати з індексом GIN. Потім я згенерував випадкові дані в таблицю. В обидва поля – name і owner генеруються рядки однакової довжини, тільки в name цей рядок переводиться в тим даних tsvector.

Без використання індекса



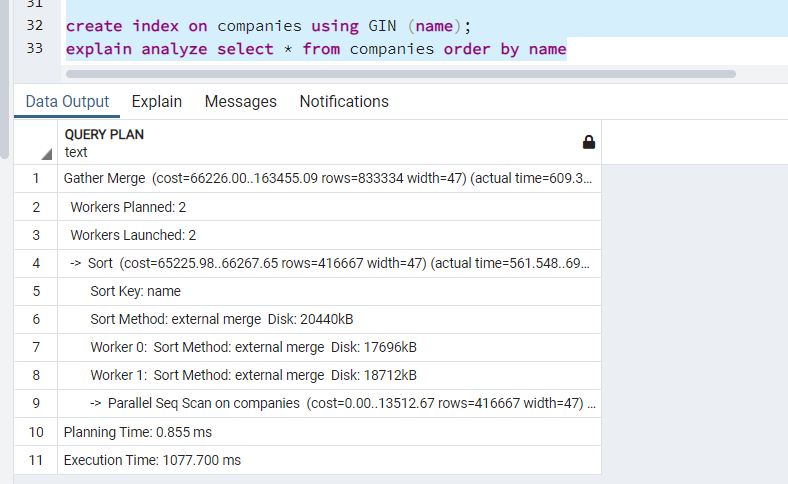
Сортування без індексу відбувається дуже повільно.

BRIN індекс



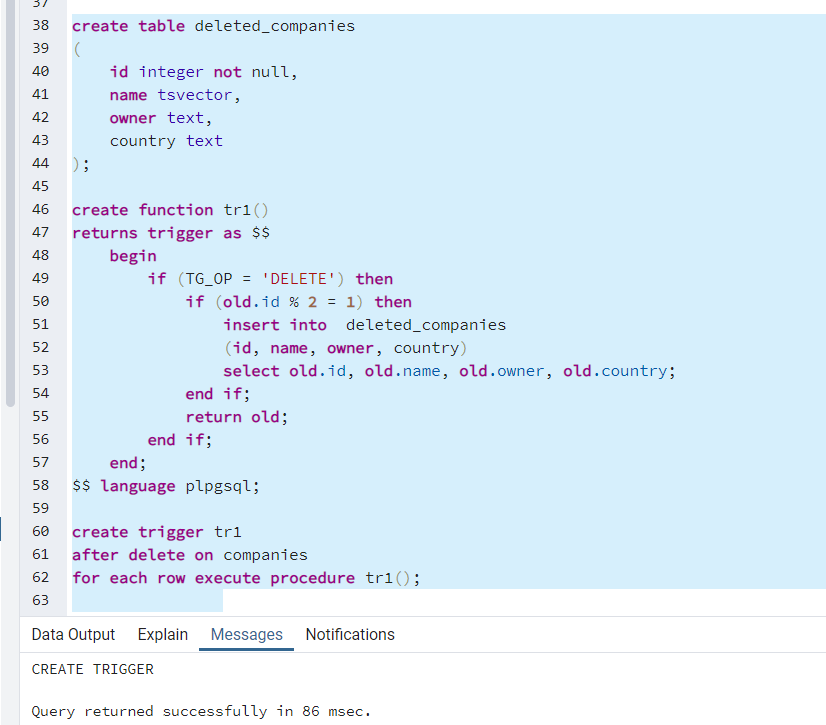
Як видно, з BRIN індексом execution time значно зменшився (вдвічі). Це працює тому, що даний індекс поділяє дані на секції для того, щоб при пошуку не проходити по уже пройдених секціях.

GIN індекс

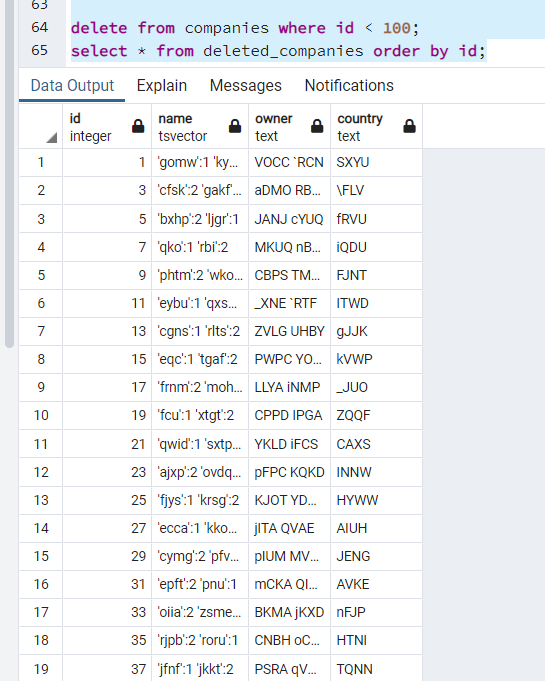


GIN індекс зменшив execution time у 6 разів, що дуже хороший результат. Даний індекс створений для повнотекстового пошуку, тому дуже добре оптимізований для великих рядків.

**Завдання 3**

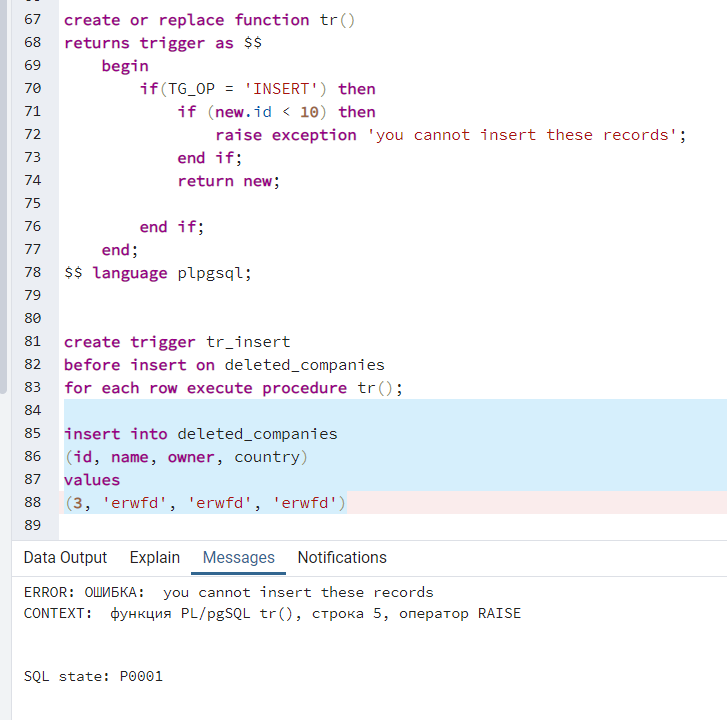
Створю тригер при видаленні 

Перевірю його роботу:



Тригер працює правильно.

Створю тригер при вставці і перевірю:



Все працює правильно.