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 III курсу
ФПМ групи КВ-81
Рожко Д.В.
Перевірив:

Київ – 2020

Завдання

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:M, M:M 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.

Завдання №1

DEFAULT

Data Output | Explain | Messages | Notifications

	person_id [PK] integer	name text	address text	contact_email text	contact_tel_num text
1	1	Steve ...	4244 Benso...	steve@gmail.com	+17632274992
2	2	Paul M...	Huntington ...	paul@xaxaxa.com	+1763336087
3	3	Abraha...	4480 King S...	ab_li@hohoho.com	+17456368450
4	4	dima	uk	dimaaxc	456

INSERT

	person_id [PK] integer	name text	address text	contact_email text	contact_tel_num text
1	1	Steve ...	4244 Benso...	steve@gmail.com	+17632274992
2	2	Paul M...	Huntington ...	paul@xaxaxa.com	+1763336087
3	3	Abraha...	4480 King S...	ab_li@hohoho.com	+17456368450
4	4	dima	uk	dimaaxc	456
5	5	Eg	bbbb	asdad@xvi.com	+10005044

UPDATE

Data Output | Explain | Messages | Notifications

	person_id [PK] integer	name text	address text	contact_email text	contact_tel_num text
1	1	Steve ...	4244 Benso...	steve@gmail.com	+17632274992
2	2	Paul M...	Huntington ...	paul@xaxaxa.com	+1763336087
3	3	Abraha...	4480 King S...	ab_li@hohoho.com	+17456368450
4	4	dima	uk	dimaaxc	456
5	5	Eg	bbbb	supermegahacer86	+10005044

DELETE

	person_id [PK] integer	name text	address text	contact_email text	contact_tel_num text
1	1	Steve ...	4244 Benso...	steve@gmail.com	+17632274992
2	2	Paul M...	Huntington ...	paul@xaxaxa.com	+1763336087
3	3	Abraha...	4480 King S...	ab_li@hohoho.com	+17456368450
4	4	dima	uk	dimaaxc	456

Завдання №2 (ddl/q2.sql)

```
--create table
CREATE TABLE random (
	rand_num	integer		NOT NULL,
	rand_word	varchar(2)	NOT NULL);
--insert data
INSERT INTO random (rand_num, rand_word)
	SELECT trunc(73 + random()*15),
		chr(trunc(73 + random()*15)::int) || chr(trunc(73 + random()*15)::int)
	 FROM generate_series(1,50);
--Btree (default)
CREATE INDEX num ON random(rand_num);
--Btree
CREATE INDEX num_btree ON random USING btree(rand_num);
--GIN
CREATE EXTENSION btree_gin;
CREATE INDEX gin_num ON random USING GIN (rand_num);
```

Переваги B-Tree:

Можуть обробляти всі типи даних, а також можуть використовуватися при отриванні значення NULL. B-Tree призначені для ефективногї обробки кешованих даних, навіть якщо вони кешовані частково.

Переваги GIN:

Ефективні при відображені багатьох значень в один рядок. Добре підходять для індексації значень масивів. А також для здійснення повнотекстового пошуку.

Використання індексів є дуже ситуативним, оскільки їхня актуальність у пришвидшені зберігається виключно при не змінній з часом базі, що в реальних умовах трапляється не часто. Із задачею пришвидшення роботи запитів краще всього справляється оптимізатор вкладений в саму СУБД, який визначає найкращий методи виконання запиту на основі поточного стану бази даних, а також з використанням статистичних даних. А також якщо один запит виконується декілька разів підряд і не було між ними їх оновлення в базі, оптимізатор може і не виконувати цей запит взявши данні з кеш пам’яті. Такий розвиток подій присутній також в базах даних Oracle і т. д.

Завдання №3 (ddl/q3.sql)

```
CREATE FUNCTION add_num(
	num    integer,
	val    integer)
	RETURNS integer
	AS $$
		BEGIN
			UPDATE random
			SET rand_num = rand_num + val
			WHERE rand_num = num;
			COMMIT;
			RETURN random_num;
		END;$$ LANGUAGE plpgsql;



CREATE TRIGGER num_rand
	AFTER UPDATE OF rand_num
		ON random
EXECUTE PROCEDURE add_num(36, 5);
```