12. Індексування в PostgreSQL. Пояснення плану виконання запиту

Гаврилов О.В., АІ-243

Вариант 6

Мета: Ознайомити студентів із видами індексів та приципами їх роботи в СУБД PostgreSQL, а також із механізмами, що дозволяють отримати інформацію про виконання запиту за допомогою команди EXPLAIN.

Завдання:

1. Ознайомитися із принципами роботи індексів, їх різновидами в СУБД PostgreSQL, а також із командами, що дозволяють вивчити план виконання запиту;
2. Виконати SQL-оператори для створення індексів до таблиць, подивитися на їх роботу;
3. Навчитися аналізувати план виконання запиту і давати пояснення основним його елементам.

Результат:

Студенти повинні подати SQL-скрипти, що відображають створення індексів до таблиць, а також плани виконання запитів до і після створення індексних структур даних відповідно до завдання та предметної області, їх опис, а також звіт з результатами тестування.

Завдання до лабораторної роботи

1. Створіть запит, який вибирає із таблиці за варіантом предментої області для лабораторних робіт значення за числовим діапазоном (від-до) одного із стовпців таблиці (не з первинним чи зовнішнім ключем). Виконайте команду пояснення плану виконання запиту. Опишіть план виконання запиту.

1) словесна постановка задачі, що вирішується:

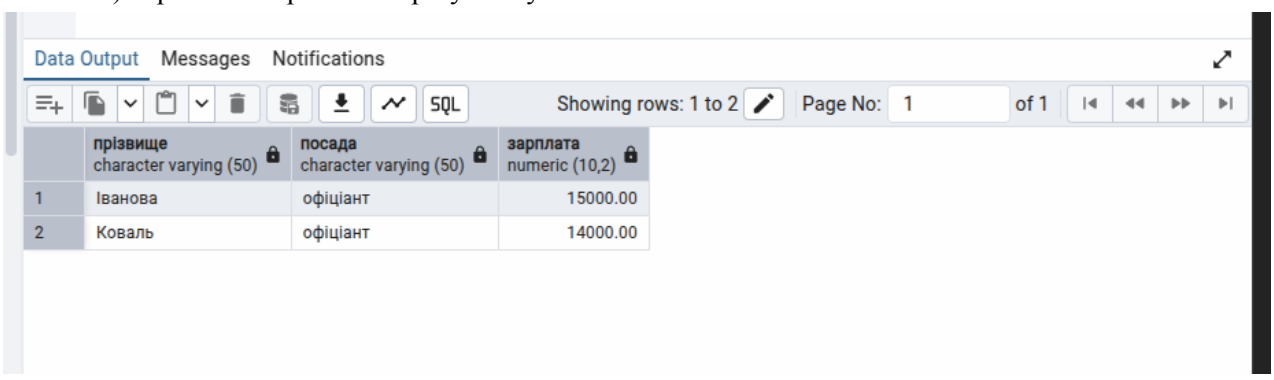
Необхідно вибрати прізвища та посади всього персоналу, чия зарплата знаходиться в діапазоні від 10000.00 до 20000.00 (включно).

2) SQL-код рішення:

```
-- Команда EXPLAIN для аналізу плану
EXPLAIN SELECT прізвище, посада, зарплата
FROM Персонал
WHERE зарплата BETWEEN 10000.00 AND 20000.00;

-- Сам запит
SELECT прізвище, посада, зарплата
FROM Персонал
WHERE зарплата BETWEEN 10000.00 AND 20000.00;
```

3) скриншот отриманого результату:



2. Створіть до стовпця із завдання 1 індекс. Виконайте команду пояснення плану виконання того ж самого запиту, як і в завданні 1. Опишіть зміни, що відбулись.

1) словесна постановка задачі, що вирішується:

1. Створити індекс на стовпці зарплата таблиці Персонал.
2. Вибрати прізвища та посади персоналу, чия зарплата знаходиться в діапазоні від 10000.00 до 20000.00, використовуючи створений індекс.
3. Проаналізувати новий план виконання запиту та описати зміни.
   Після створення індексу idx_персонал_зарплата, планувальник запитів, ймовірно, перейде від Послідовного сканування (Seq Scan, як було у Завданні 1) до Індексного сканування (Index Scan) або Сканування бітової карти (Bitmap Index Scan).

- Зміна вузла: Замість Seq Scan (читання всієї таблиці) з'явиться вузол, що використовує індекс, наприклад, Index Scan using idx_персонал_зарплата on Персонал.

- Механізм: База даних тепер спочатку звертається до індексу, щоб швидко знайти фізичні адреси (TID) рядків, які відповідають умові зарплата BETWEEN 10000.00 AND 20000.00. Це значно швидше, ніж повне сканування, оскільки індекс є меншим і впорядкованим.

- Зменшення вартості (Cost): Загальна оціночна вартість запиту (cost=...) має значно знизитися порівняно з вартістю Seq Scan із Завдання 1, що вказує на підвищення продуктивності для даного типу пошуку.

2) SQL-код рішення:

```
-- Створення індексу для стовпця "зарплата"
CREATE INDEX idx_персонал_зарплата ON Персонал (зарплата);


-- Команда EXPLAIN для аналізу плану після створення індексу EXPLAIN SELECT
прізвище, посада, зарплата FROM Персонал WHERE зарплата BETWEEN 10000.00 AND
20000.00; -- Сам запит SELECT прізвище, посада, зарплата FROM Персонал WHERE
зарплата BETWEEN 10000.00 AND 20000.00;
```

3) скриншот отриманого результату:

3. Виконайте запит, що вибирає значення із різних 2 стовпців іншої таблиці (наприклад, за цілочисельним і символьним значенням чи іншим). Виконайте команду пояснення плану виконання запиту. Опишіть план виконання запиту.

1) словесна постановка задачі, що вирішується:

Необхідно вибрати дату та загальну суму всіх Замовлень, які були обслуговані офіціантами, що працюють у зміну №1.
Очікується, що планувальник запитів виконає JOIN (ймовірно, Hash Join або Nested Loop

Join) та застосує фільтр:

1. Сканування таблиці Офіціант:

- Якщо на стовпці номер_зміни немає індексу, буде виконано Seq Scan по таблиці Офіціант з фільтром номер_зміни = 1.

2. Об'єднання (Join):

- Після цього (або під час цього, якщо це Nested Loop) буде виконано операцію об'єднання з таблицею Замовлення за стовпцем id_офіціанта. Оскільки id_офіціанта є зовнішнім ключем, він автоматично має індекс (або успадковує його від Персонал.id_персоналу), що сприятиме ефективному об'єднанню (наприклад, через Index Scan на таблиці Замовлення).

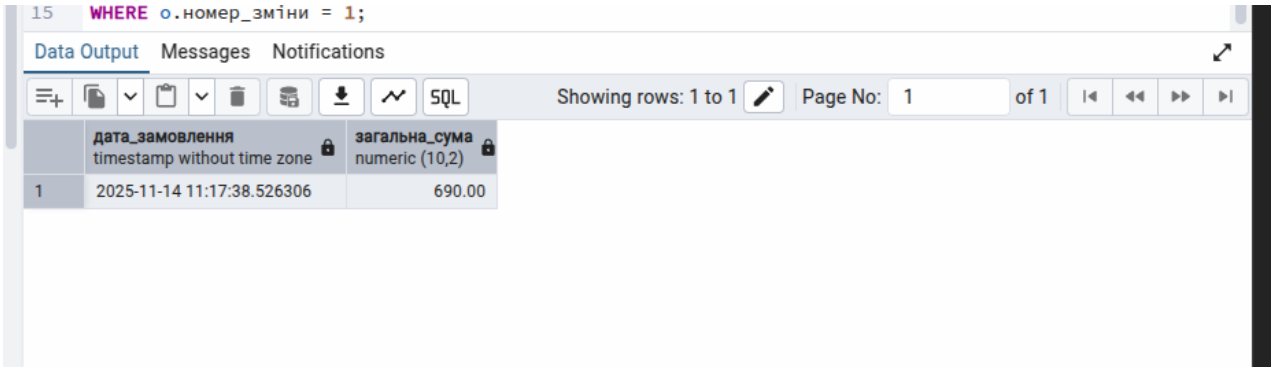
3. Висновок: Загальна ефективність об'єднання залежить від обраного механізму JOIN. Завдяки наявності індексу на ключах, об'єднання буде відносно швидким, але пошук у таблиці Офіціант за неіндексованим стовпцем номер_зміни може уповільнювати процес (хоча для малих таблиць це не критично).

2) SQL-код рішення:

```
-- Команда EXPLAIN для аналізу плану
EXPLAIN SELECT
    z.дата_замовлення,
    z.загальна_сума
FROM Замовлення z
INNER JOIN Офіціант o ON z.id_офіціанта = o.id_офіціанта
WHERE o.номер_зміни = 1;

-- Сам запит
SELECT
    z.дата_замовлення,
    z.загальна_сума
FROM Замовлення z
INNER JOIN Офіціант o ON z.id_офіціанта = o.id_офіціанта
WHERE o.номер_зміни = 1;
```

3) скриншот отриманого результату:



4. Створіть до стовпців із завдання 3 багатостовпчиковий індекс. Виконайте команду пояснення плану виконання того ж самого запиту, як і в завданні 3. Опишіть зміни, що відбулись.

1) словесна постановка задачі, що вирішується:

Створити багатоколонковий індекс на таблиці Офіціант для стовпців номер_зміни та рівень_обслуговування (або лише номер_зміни, якщо ми фокусуємося виключно на оптимізації Завдання 3). Створимо на номер_зміни для оптимізації фільтра.

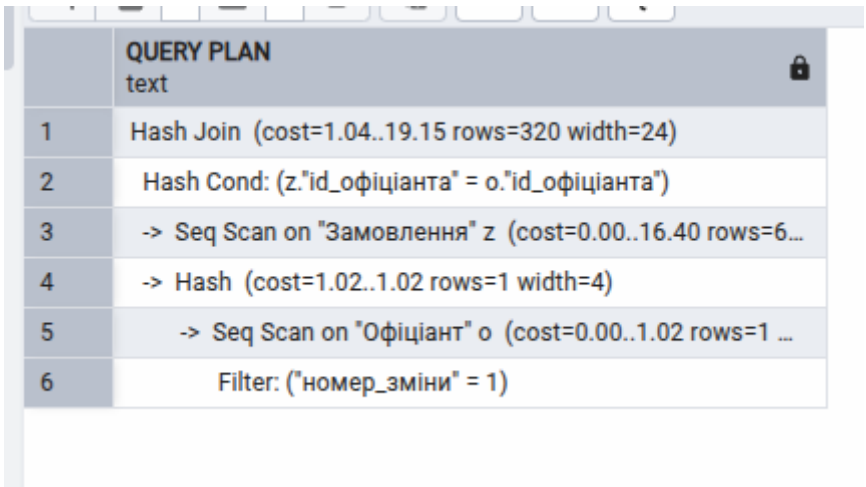
- Повторно виконати та проаналізувати запит із Завдання 3 (вибрати замовлення, обслуговані офіціантами зі зміни №1).

2) SQL-код рішення:

```
-- Створення багатоколонкового індексу (для демонстрації концепції)
-- Ми створимо індекс на "номер_зміни" та "рівень_обслуговування"
CREATE INDEX idx_офіціант_зміна_рівень ON Офіціант (номер_зміни,
рівень_обслуговування);

-- Команда EXPLAIN для аналізу плану після створення індексу
EXPLAIN SELECT
    z.дата_замовлення,
    z.загальна_сума
FROM Замовлення z
INNER JOIN Офіціант o ON z.id_офіціанта = o.id_офіціанта
WHERE o.номер_зміни = 1;
```

3) скриншот отриманого результату:



	QUERY PLAN text
1	Hash Join (cost=1.04..19.15 rows=320 width=24)
2	Hash Cond: (z."id_офіціанта" = o."id_офіціанта")
3	-> Seq Scan on "Замовлення" z (cost=0.00..16.40 rows=6...
4	-> Hash (cost=1.02..1.02 rows=1 width=4)
5	-> Seq Scan on "Офіціант" o (cost=0.00..1.02 rows=1 ...
6	Filter: ("номер_зміни" = 1)

5. Виконайте запит до таблиці із завдання 3, використовуючи 1 індексований стовпець із завдання 4 та 1 неідексований, поясніть зміни у плані виконання запиту.

1) словесна постановка задачі, що вирішується:

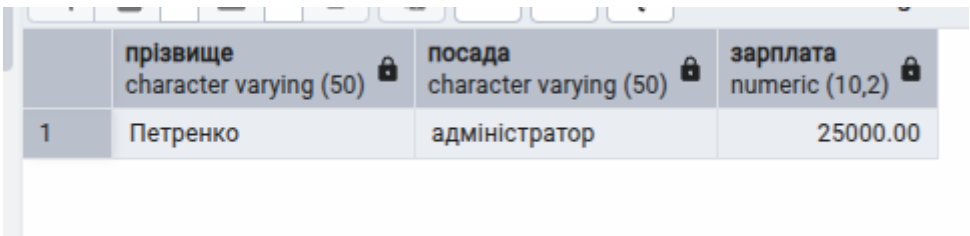
Необхідно вибрати прізвища, посади та зарплати персоналу, чия зарплата більша за 15000.00 (індексований стовпець) І чия посада є 'адміністратор' (неіндексований стовпець).

2) SQL-код рішення:

```
-- Команда EXPLAIN для аналізу плану
EXPLAIN SELECT прізвище, посада, зарплата
FROM Персонал
WHERE зарплата > 15000.00 AND посада = 'адміністратор';

-- Сам запит
SELECT прізвище, посада, зарплата
FROM Персонал
WHERE зарплата > 15000.00 AND посада = 'адміністратор';
```

3) скриншот отриманого результату:



	прізвище character varying (50)	посада character varying (50)	зарплата numeric (10,2)
1	Петренко	адміністратор	25000.00

6. Створіть індекс, що індексуватиме *частину* значень стовпчика третьої таблиці (не

повторюючи таблиці із попередніх завдань). Виконайте пояснення плану виконання запиту, що використовує цей індекс.

1) словесна постановка задачі, що вирішується:

Створити частковий індекс на таблиці Страва для стовпця ціна, який індексує лише ті страви, які недоступні (доступність = false).

- Виконати та проаналізувати запит, що вибирає всі недоступні страви з ціною, більшою за 100.00.

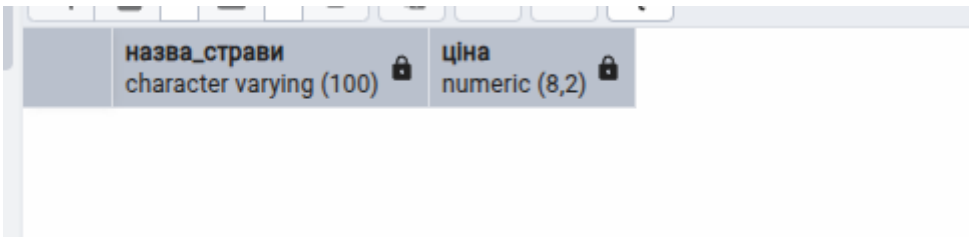
2) SQL-код рішення:

```
-- Створення часткового індексу: індексує ціну лише для недоступних страв
CREATE INDEX idx_страва_недоступна_ціна ON Страва (ціна)
WHERE доступність = false;


-- Команда EXPLAIN для аналізу плану
EXPLAIN SELECT назва_страви, ціна
FROM Страва
WHERE доступність = false AND ціна > 100.00;

-- Сам запит
SELECT назва_страви, ціна
FROM Страва
WHERE доступність = false AND ціна > 100.00;
```

3) скриншот отриманого результату:



7. Додайте до таблиці із першого завдання 5 рядків, виконайте команду реіндексування.

1) словесна постановка задачі, що вирішується:

Вставити 5 нових рядків у таблицю Персонал.

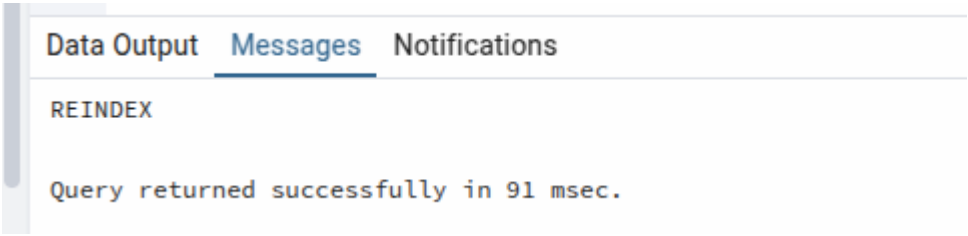
- Виконати команду реіндексації для всіх індексів таблиці Персонал (включаючи idx_персонал_зарплата).

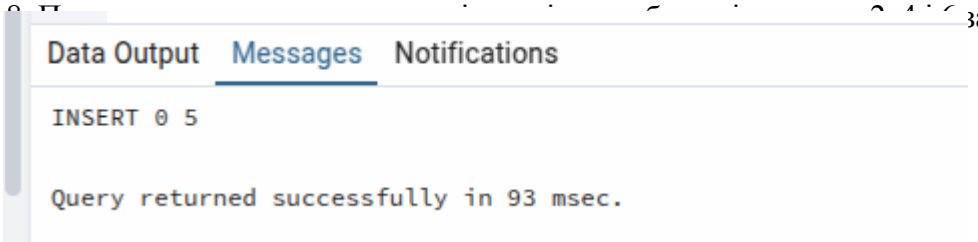
2) SQL-код рішення:

```
INSERT INTO Персонал (ім'я, прізвище, посада, логін, пароль, зарплата) VALUES
('Катерина', 'Лисиця', 'офіціант', 'kate_l', 'pass_k', 16500.00),
('Віктор', 'Крук', 'кухар', 'victor_k', 'pass_v', 32000.00),
('Назар', 'Орел', 'адміністратор', 'nazar_o', 'pass_n', 26000.00),
('Софія', 'Зайцева', 'офіціант', 'sofia_z', 'pass_s', 15500.00),
('Денис', 'Ведмідь', 'кухар', 'denis_v', 'pass_d', 29000.00);


REINDEX TABLE Персонал;
```

3) скриншот отриманого результату:





за допомогою представлення pg_indexes або `\d`.

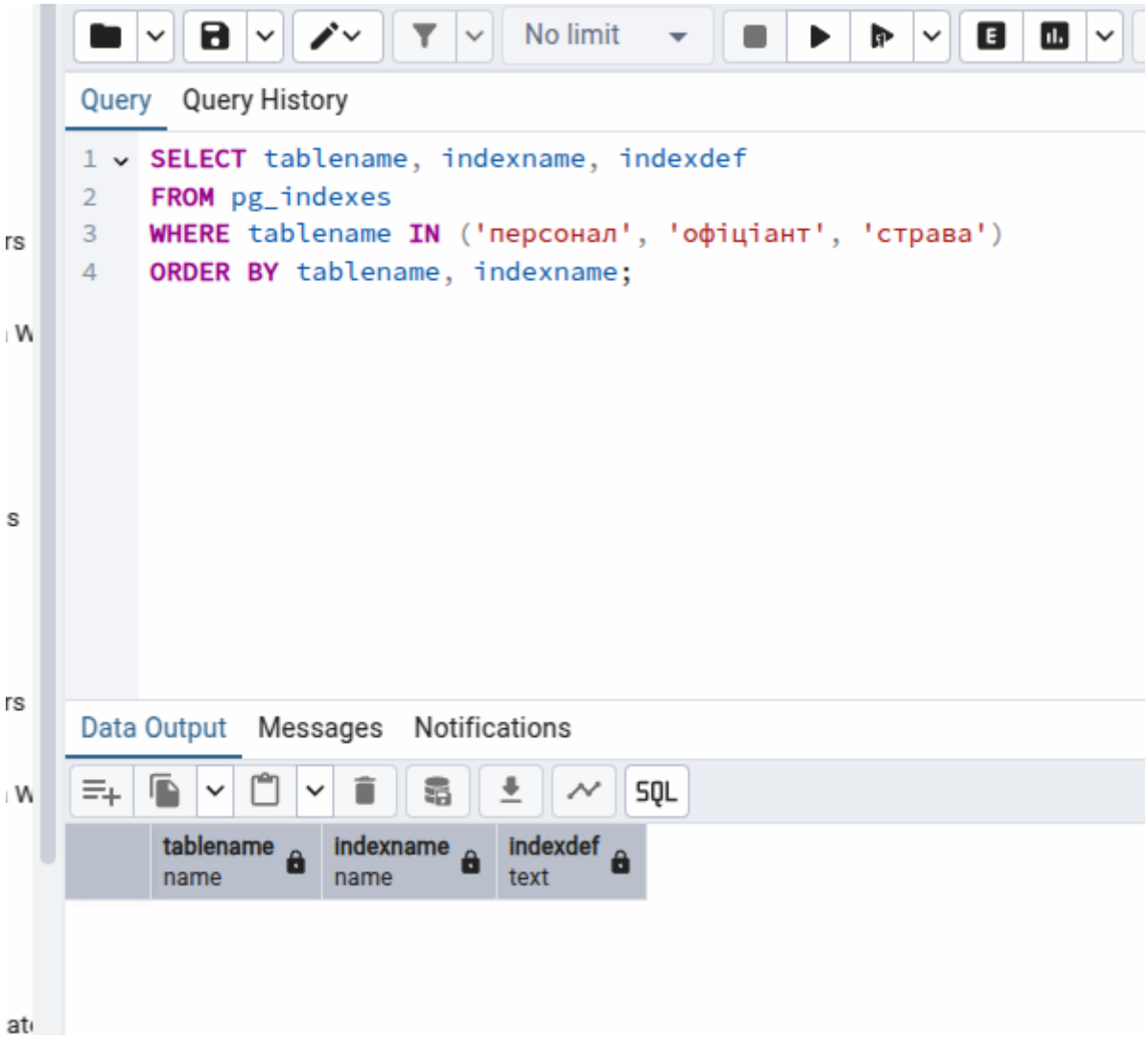
1) словесна постановка задачі, що вирішується:

Отримати повний список індексів, що належать таблицям Персонал, Офіціант та Страва, використовуючи системне представлення pg_indexes, та переконатися, що всі індекси, створені протягом лабораторної роботи (Завдання 2, 4, 6), існують.

2) SQL-код рішення:

```
SELECT tablename, indexname, indexdef
FROM pg_indexes
WHERE tablename IN ('персонал', 'офіціант', 'страва')
ORDER BY tablename, indexname;
```

3) скриншот отриманого результату:



Звіт до лабораторної роботи 12 можна здати онлайн на сайті ДО edu.op.edu.ua до початку вашого заняття.