Лабораторна робота 3: Маніпулювання даними SQL (OLTP)

У цій лабораторній роботі ви протестуєте базу даних, яку створили в лабораторних роботах 1–2, виконуючи запити в стилі транзакцій у PostgreSQL (за допомогою pgAdmin). Ми зосереджуємося на запитах OLTP (онлайн-обробка транзакцій) – великій кількості коротких транзакцій, які отримують або оновлюють окремі записи. На практиці це означає написання ручних SQL-інструкцій для вибору даних та вставки, оновлення або видалення рядків. Це імітує реальні операції (наприклад, отримання реєстрацій студентів на курси або обробку замовлень електронної комерції) та перевіряє, чи ваша схема та дані поводяться належним чином.

Цілі

- Написати запити **SELECT** для отримання даних (включаючи фільтрацію за допомогою `WHERE` та вибір певних стовпців).
- Практикувати використання оператори **INSERT** для додавання нових рядків до таблиць.
- Практикувати використання оператора **UPDATE** для зміни існуючих рядків (використовуючи `SET` та `WHERE`).
- Практикувати використання оператори **DELETE** для безпечного видалення рядків (за допомогою `WHERE`).
- Вивчити основні операції маніпулювання даними (DML) у PostgreSQL та спостерігайте за їхнім впливом.

Інструкції

Пишіть та виконуйте оператори `SELECT` для отримання даних з таблиць. Наприклад, щоб переглянути всі записи в таблиці, ви можете використовувати `SELECT * FROM Students;`. Для фільтрації результатів використовуйте речення `WHERE`, наприклад, `SELECT name, email FROM Students WHERE major = 'CS';`. Включайте лише потрібні стовпці – наприклад, `SELECT id, name FROM Courses;` замість `SELECT *`. Тестуйте різні запити: у схемі **студент-курс** ви можете об'єднувати таблиці або фільтрувати за ідентифікатором студента чи кодом курсу; у схемі **електронної комерції** спробуйте вибрати замовлення на суму понад певну суму або клієнтів з певного міста. Пам'ятайте, що `SELECT` отримує дані з таблиць (або представлень).

Напишіть оператори `INSERT INTO`, щоб додати нові рядки. Для кожної таблиці, яку потрібно змінити, вкажіть стовпці та значення. Наприклад:

```
INSERT INTO Students (id, name, major) VALUES (1001, 'Alice Smith', 'Biology');
```

Виконайте запит SELECT, щоб перевірити, чи з'явився новий рядок. (Якщо ви пропустили список стовпців, переконайтеся, що значення відповідають порядку стовпців таблиці) Оператор INSERT використовується для додавання нових рядків до таблиці.

Використайте оператор UPDATE для зміни існуючих даних. Використовуйте SET для зміни одного або кількох стовпців і завжди додавайте речення WHERE, щоб обмежити, які рядки змінюються. Наприклад:

```
UPDATE Students
SET email = 'alice.smith@univ.edu', graduation_year = 2025
WHERE id = 1001;
```

Після UPDATE виконайте команду SELECT, щоб перевірити зміни. Примітка: оператор UPDATE змінює дані. Без належного WHERE ви можете ненавмисно змінити всі рядки, тому завжди вибирайте конкретні записи.

Напишіть оператори DELETE для видалення рядків. Знову ж таки використовуйте речення WHERE. Приклади:

```
DELETE FROM Enrollments
WHERE student_id = 1001 AND course_id = 'CS101';
```

Перевірте за допомогою SELECT, що рядок зник. (Якщо ви забудете WHERE, вся таблиця буде очищена) Оператор DELETE видаляє рядки з таблиці; без WHERE **всі рядки будуть видалені**.

Результати

— SQL-скрипт(и) у звіті з усіма вашими операторами CREATE, UPDATE/SET, DELETE, SELECT/WHERE по порядку. Ви можете додавати коментарі до SQL-файлу для ідентифікації таблиць.

— Короткий письмовий звіт, що підсумовує вашу остаточну схему. Перелічіть кожну таблицю з її стовпцями та ключами, а також поясніть будь-які важливі обмеження або припущення.

— Доказ того, що таблиці були заповнені (це може бути частина SQL-файлу або скріншоти результатів запиту). Переконайтеся, що кожна таблиця містить щонайменше 3–5 рядків.

— Ви можете додати знімки екрана pgAdmin, що показують ваші таблиці, інструмент запитів або результати SELECT *, якщо це корисно (особливо для розділу Оцінка або якщо самі SQL-файли є неоднозначними).

Поради

Завжди додавайте речення WHERE до операторів UPDATE та DELETE, якщо ви дійсно не маєте наміру впливати на кожен рядок. Наприклад, DELETE FROM Products; без WHERE видаляє всі дані таблиці.

Спочатку протестуйте за допомогою SELECT. Перш ніж запускати оновлення/видалення, спробуйте виконати `SELECT` з тим самим `WHERE`, щоб побачити, які рядки будуть вплинуті. Це допомагає виявити помилки.

За потреби створіть резервні копії даних. Для великих або критично важливих операцій розгляньте можливість копіювання таблиць (наприклад, через `SELECT INTO backup_table`) перед видаленням або оновленням багатьох рядків.

Критерії оцінювання

Ваша лабораторна робота буде оцінюватися за такими критеріями:
- **Правильність:** синтаксис SQL є коректним, а запити виконуються без помилок.
- **Ефективність:** запити дають правильні результати (вони відповідають вимогам завдання та фактично отримують/змінюють потрібні записи).
- **Безпека та найкращі практики:** правильне використання речень `WHERE`, транзакцій та перевірки результатів.
- **Чіткість та документація:** ваші коментарі або короткий виклад чітко описують, що робить кожен запит, а ваші результати організовані відповідно до запиту.

До кінця лабораторної роботи 3 ви потренуєтеся з основними операціями SQL DML у PostgreSQL та побачите, як вони впливають на вашу базу даних. Ще раз перевірте результат кожного запиту, перш ніж рухатися далі.

Додаткові ресурси:

Лекція 4 – DML basics

https://github.com/ZheniaTrochun/db-intro-course/tree/master/lectures/4%20-%20DML%20basics

Що таке OLTP та OLAP і яка між ними різниця? - Запитай SQL

https://stackoverflow.com/questions/21900185/what-are-oltp-and-olap-and-what-is-the-difference-between-them

Шпаргалка SQL для команд SELECT, INSERT, DELETE та UPDATE

https://www.mssqltips.com/sqlservertip/7511/sql-cheat-sheet-for-select-insert-delete-and-update-commands/

Підручник з SQL DELETE: Безпечне видалення даних | Outerbase

https://www.outerbase.com/learn/sql-delete/