Міністерство освіти і науки України

Львівський національний університет імені Івана Франка

Факультет прикладної математики та інформатики

Кафедра програмування

Лабораторна робота №2

**“Транзакції в СКБД PostgreSQL”**

Підготував:

студент групи ПМІ-31

 Процьків Назарій

Львів 2023

**Тема**: Вивчення понять транзакції та управління конкурентним доступом в СКБД PostgreSQL.

**Мета роботи:** Ознайомлення з використанням транзакцій, їх розробкою та застосуванням, рівнями ізоляцій та механізмом управління конкурентним доступом в СКБД PostgreSQL.

Завдання: (Варіант №17)

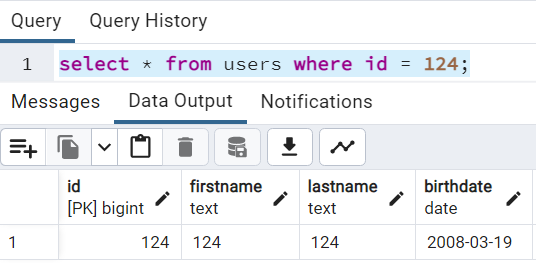
Розробити базу даних для сайту соціальної мережі. Соціальна мережа підтримує реєстрацію користувачів зі збереженням усіх їхніх деталей (ім'я, прізвище, дата народження, місце проживання, телефони, сайти/e-mail/skype іт.д.), місця і періоди перебування, місця і періоди навчання, роботи, служби, приєднані файли (зображення,фільми, аудіо), які можна пов’язувати із місцями з деталей. Крім того, кожен користувач має можливість розміщувати свої повідомлення на власній сторінці, отримувати на повідомлення «лайки» та коментарі, а також додавати інших користувачів в друзі, або в «чорний список». Додатково користувач повинен мати змогу шукати нових друзів за довільними критеріями.

**Хід роботи**

1. Опрацював теоретичні відомості про транзакції, атомарні транзакції, команди, скасування змін, точки збереження, повернення до точок збереження, рівні ізоляції, явні блокування, перевірки цілісності даних, обмеження, блокування та індекси.

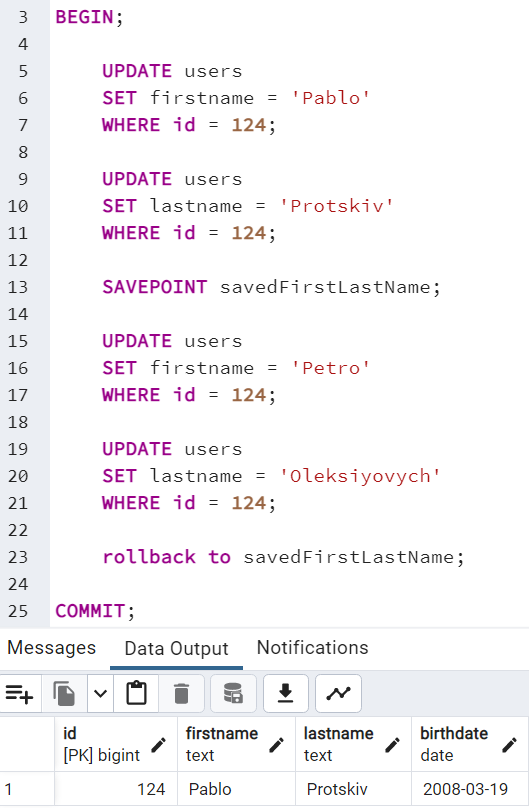
2. Написав елементарну транзакцію з двома операціями, одна з яких скасована з допомогою SAVEPOINT та операції повернення до конкретного моменту ROLLBACK TO (Рівень ізоляції - Read Committed).

Операція зі зміни імені користувача з id 124. Початкові дані:



Транзакція:

Змінив ім’я та прізвище користувача на Pablo Protskiv, зберіг це за допомогою SAVEPOINT та змінив дані ще раз. Зробив відкат до попередньої зміни за допомогою ROLLBACK..



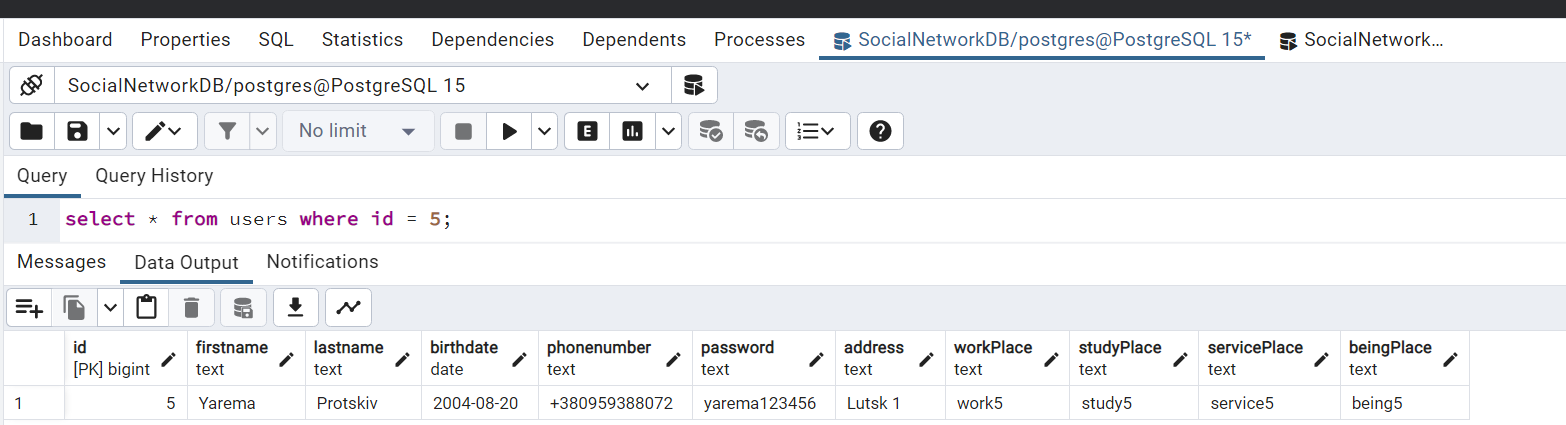
***Проблеми паралельних запитів***

Написав транзакції, щоб показати, як вони допомагають вирішити проблеми під час виконання запитів паралельно.

1. Dirty read - може виникнути, коли перша транзакція зчитує дані, які були змінені, але ще не зафіксовані іншою транзакцією:

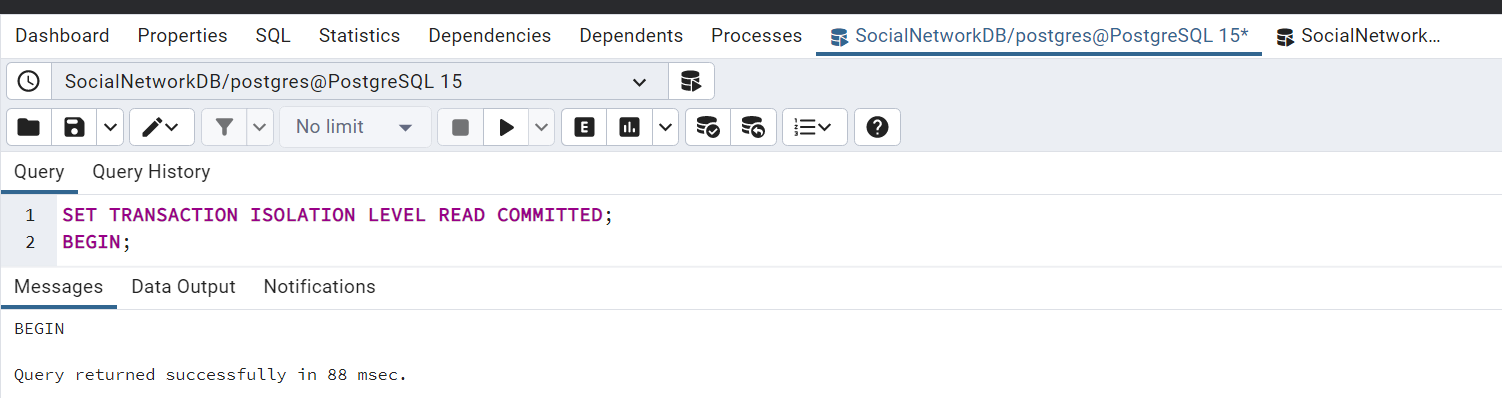
Приклад 1:

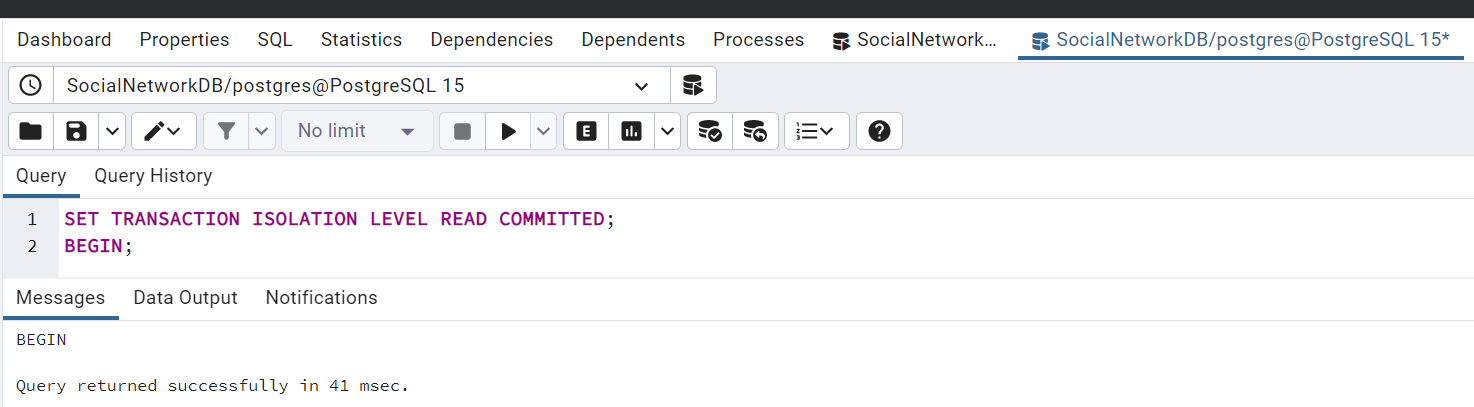
вибір юзера з id = 5:



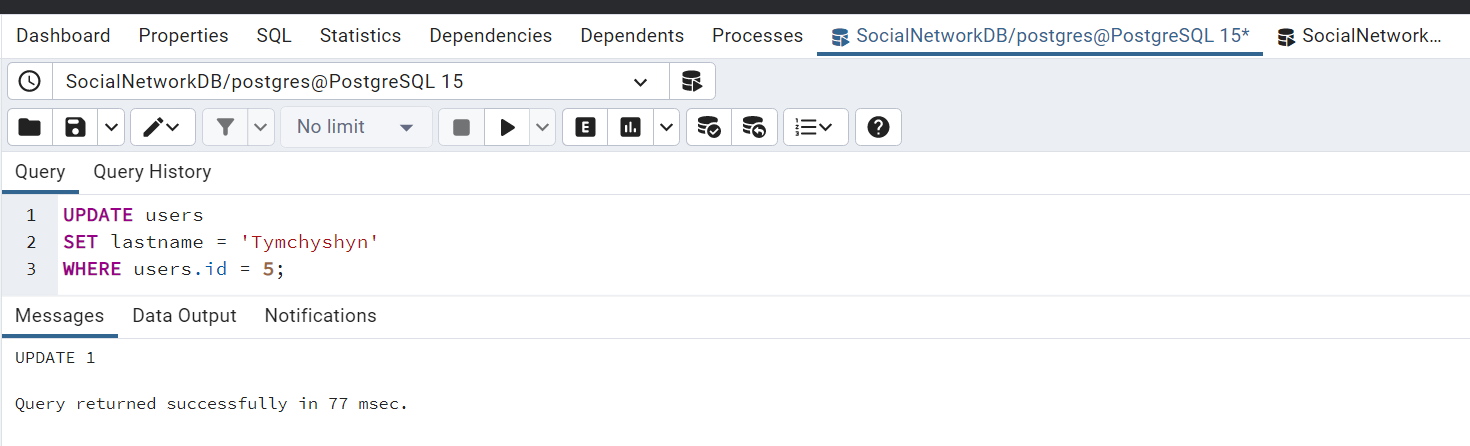
Зараз його прізвище “Protskiv”.

Відкрив дві вкладки в середовищі pgAdmin, в них встановив рівень ізоляції “Read Committed”:

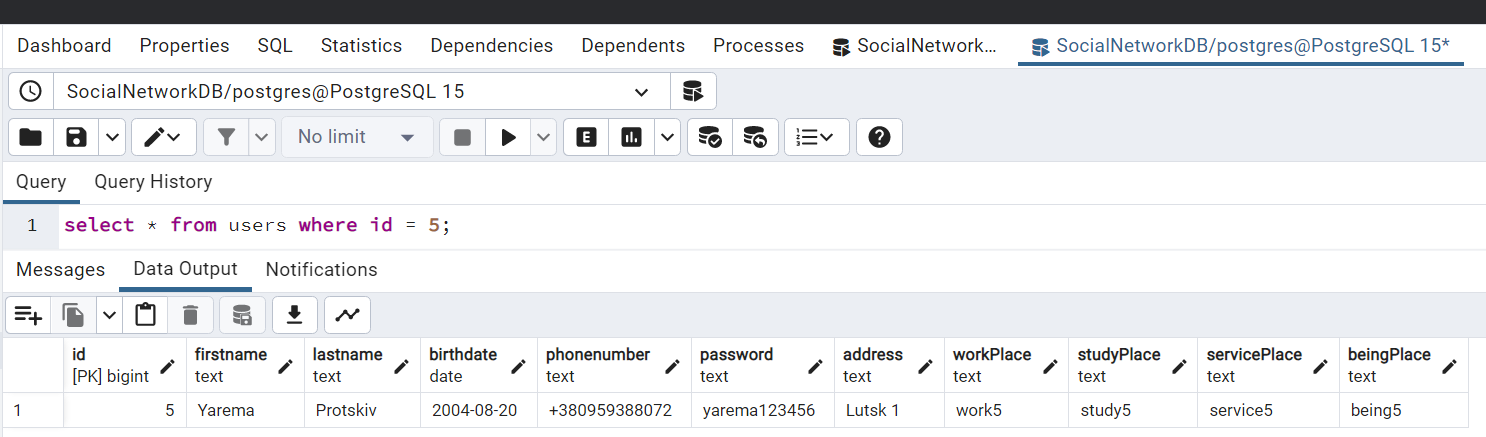




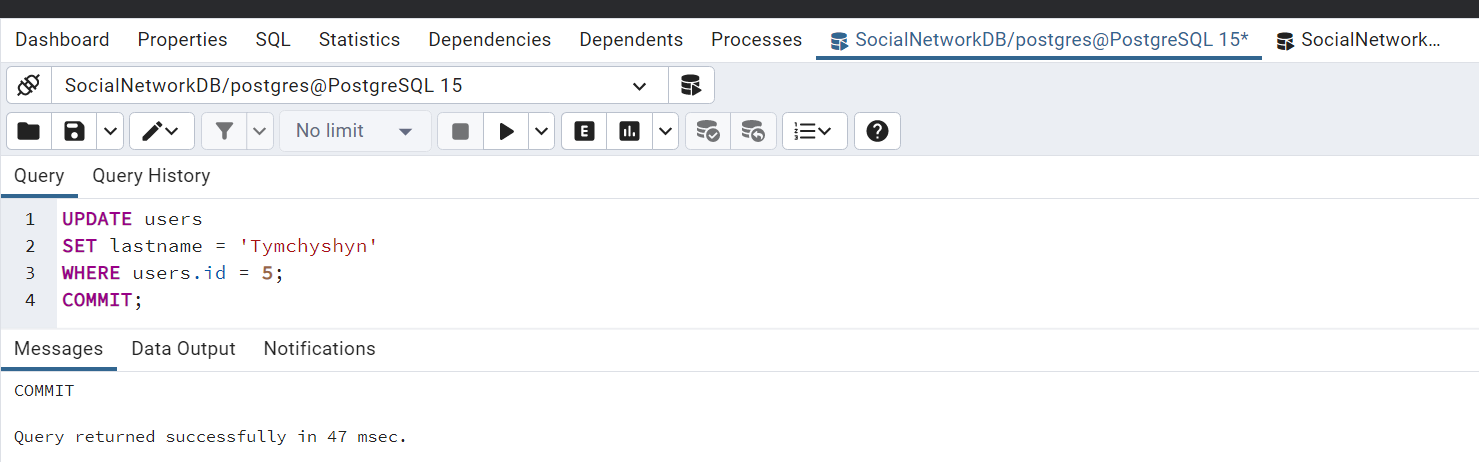
Змінив прізвище цього юзера в першій вкладці без коміту:



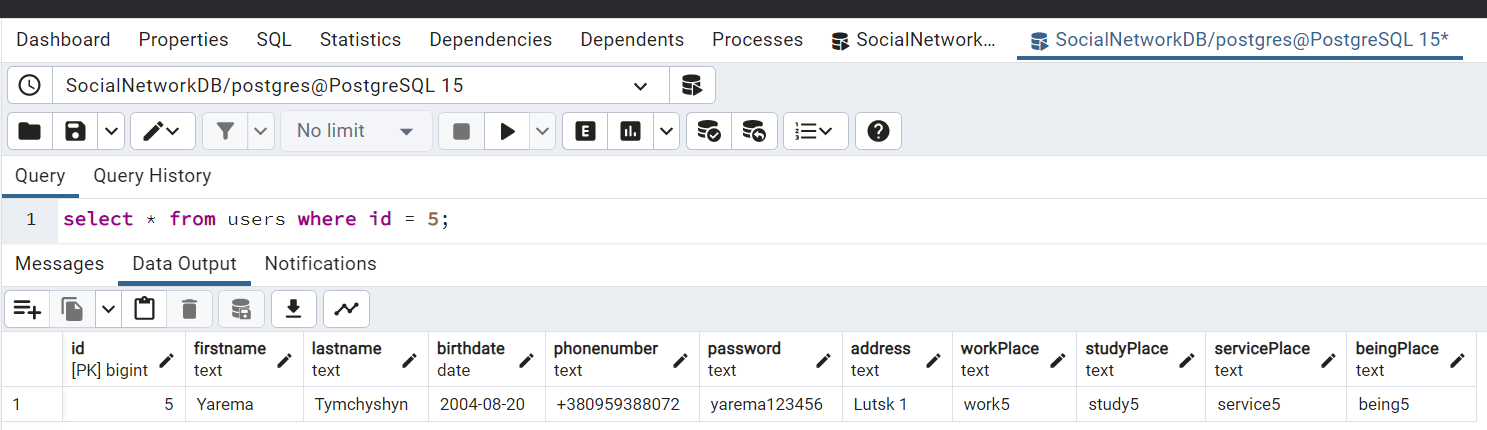
В другій вкладці написав SELECT для вибору даних:



Бачимо, що прізвище користувача з id = 5 не змінилось. Це означає, що друга вкладка не побачила змін. Але якщо дописати COMMIT до першої вкладки



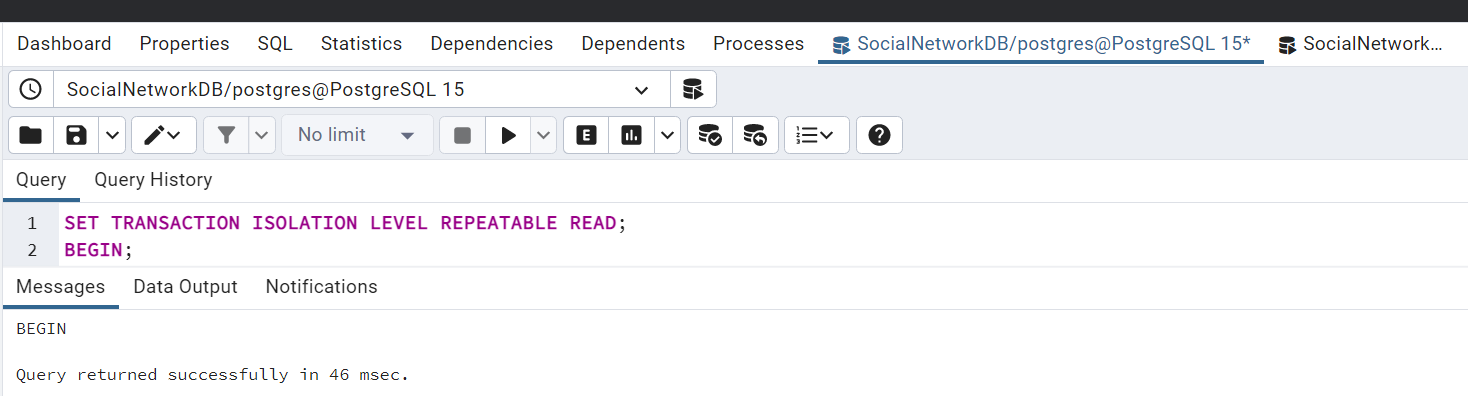
і запустити цей же запит в другій, то зміни вже буде видно:

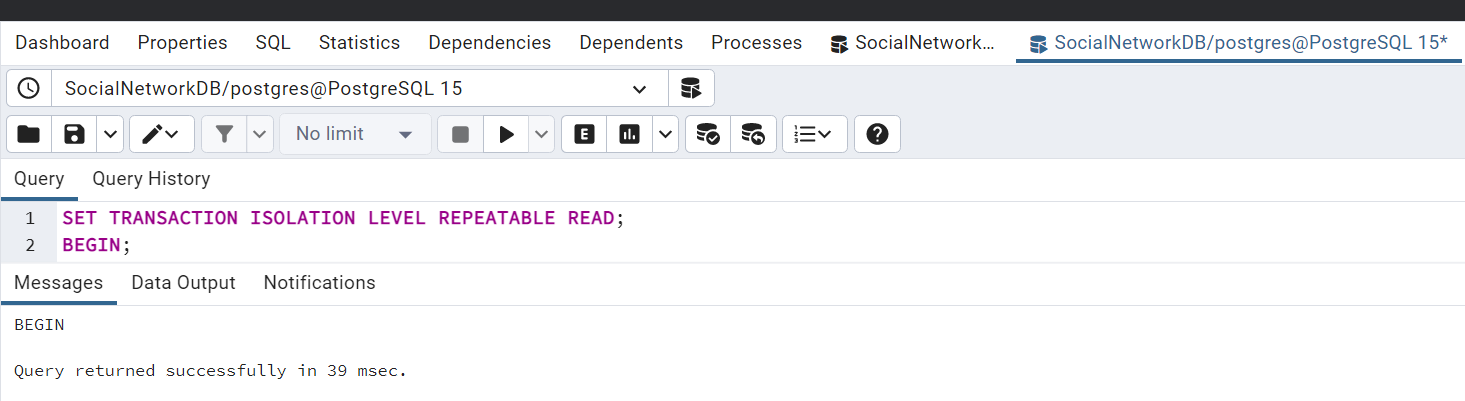


Бачимо, що тепер в користувача з id = 5 прізвище Tymchyshyn.

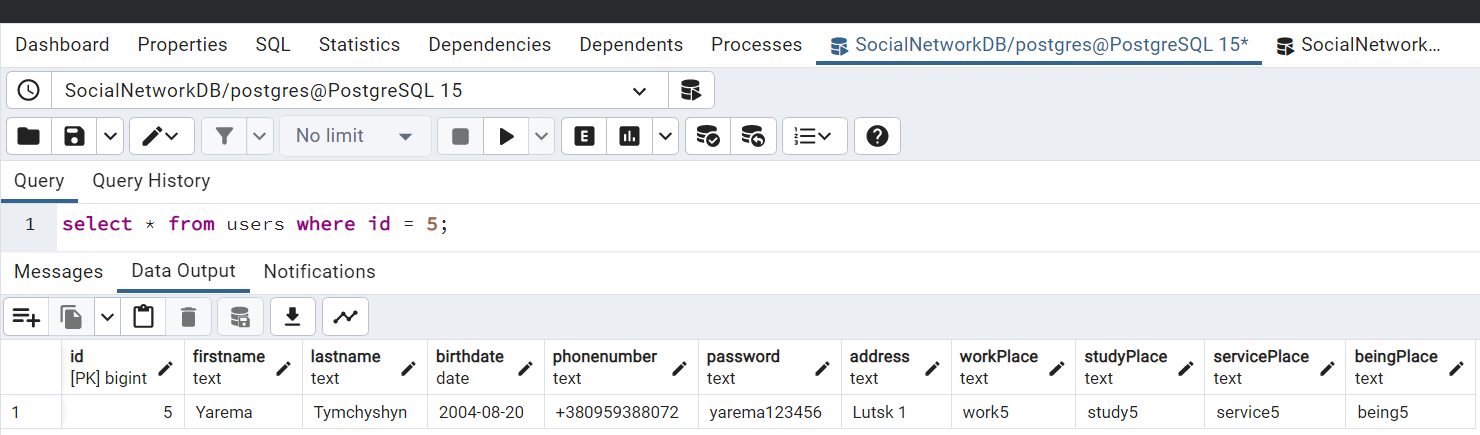
1. Nonrepeatable Read - коли одна і та ж транзакція отримує різні результати:

Встановив в обох вкладках рівень ізольованості “Repeatable Read”:



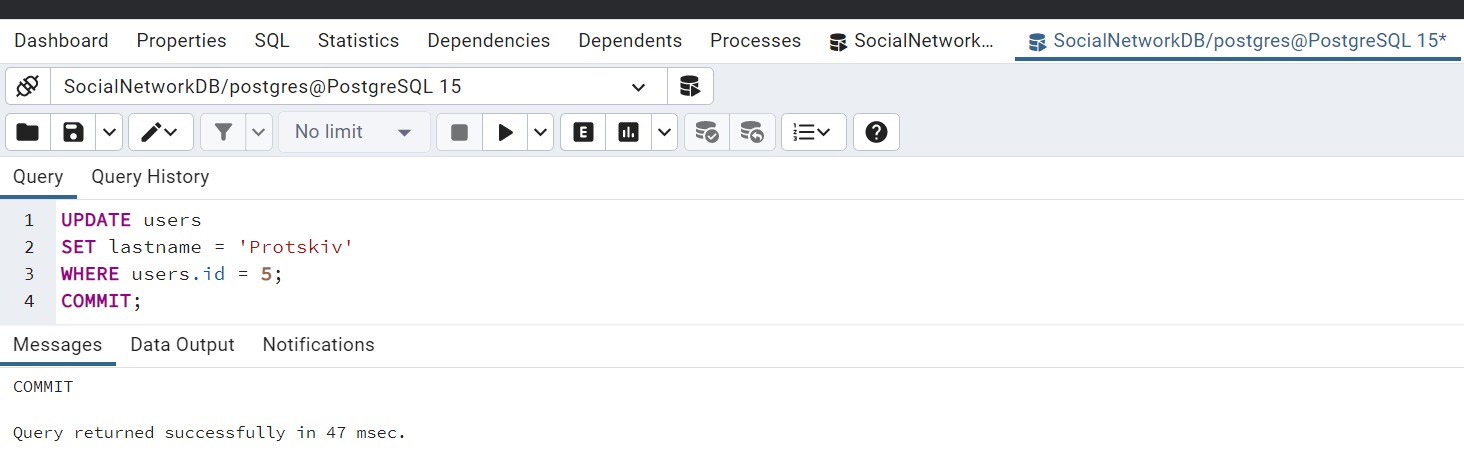


В першій вкладці знову написав SELECT:

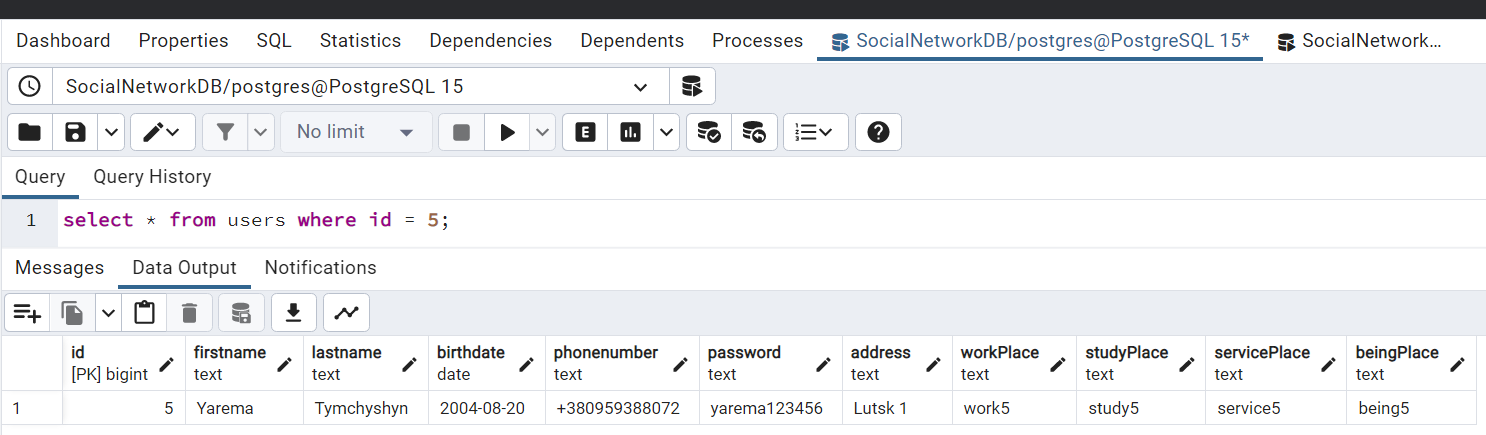


Бачимо, що на початку прізвище користувача - Тимчишин.

Далі в другій вкладці змінив його на Процьків і зберіг дані комітом.



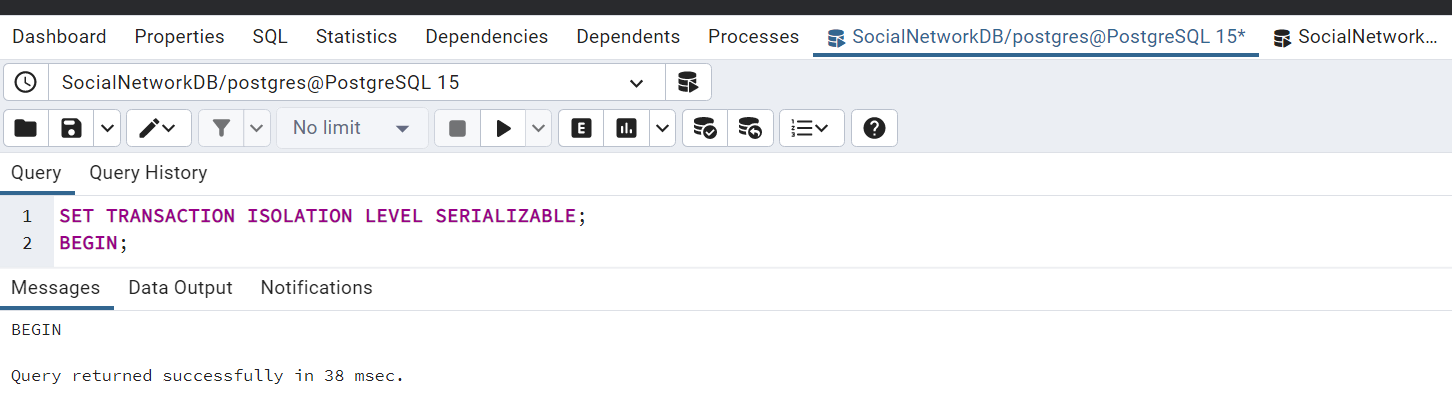
Тепер знову пишу той же SELECT в першій вкладці

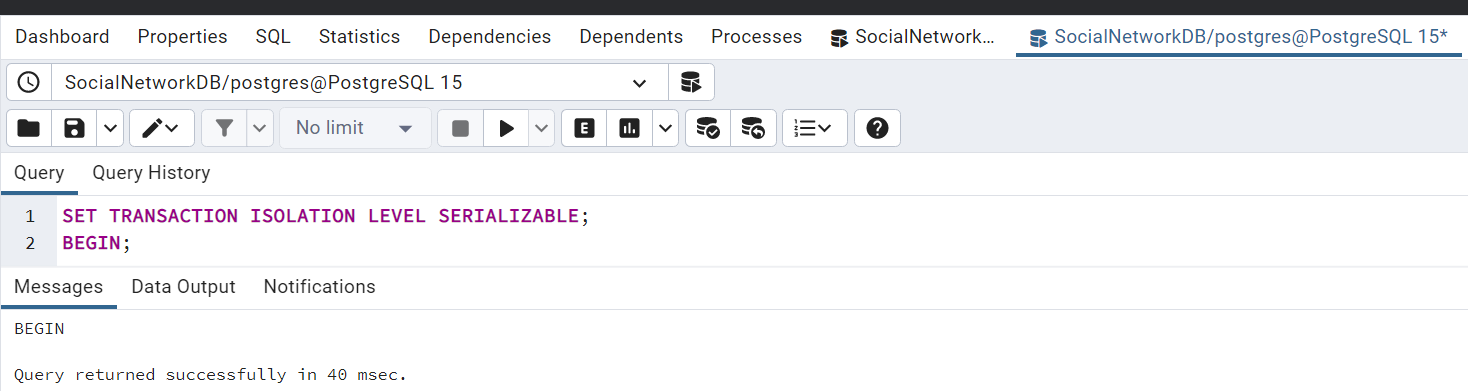


Побачив, що прізвище користувача так і не змінилось, це означає, що рівень ізоляції “Repeatable Read” повертає той самий результат, навіть якщо якісь інші транзакції його змінили, тобто він запобігає “Non-repeatable Read”.

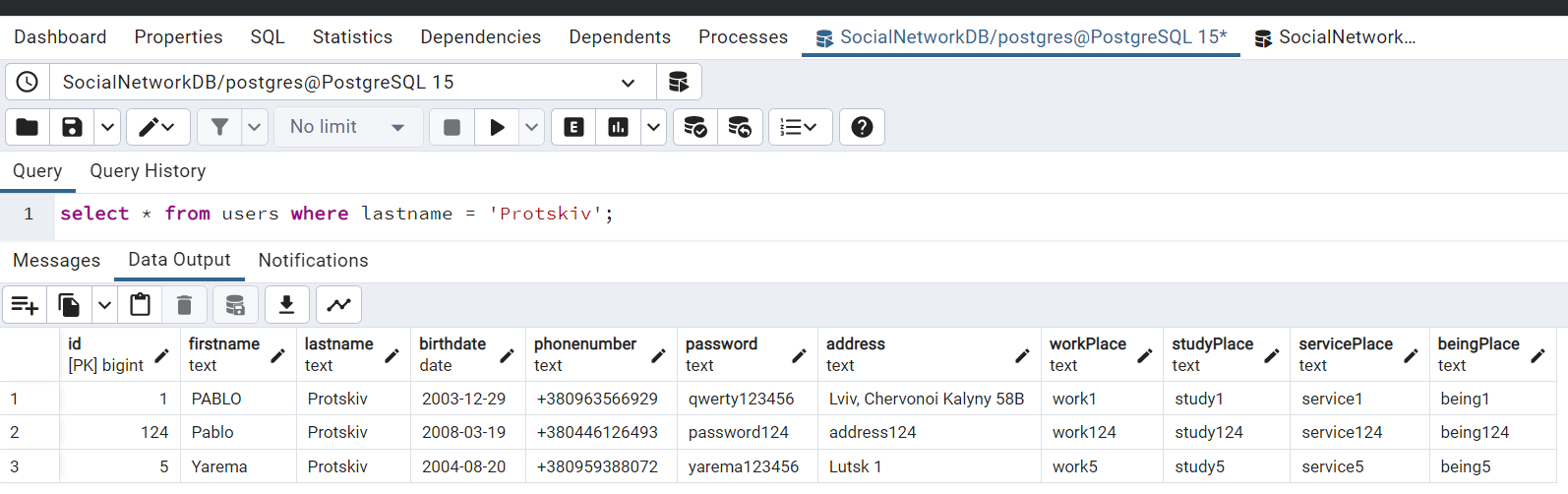
1. Phantom read - виникає, коли транзакція читає рядки, які задовольняють умову, а інша транзакція в цей же момент вставляє або видаляє рядки, які також задовольняють цю умову.

В обох вкладках встановив рівень ізоляції “Serializable”:



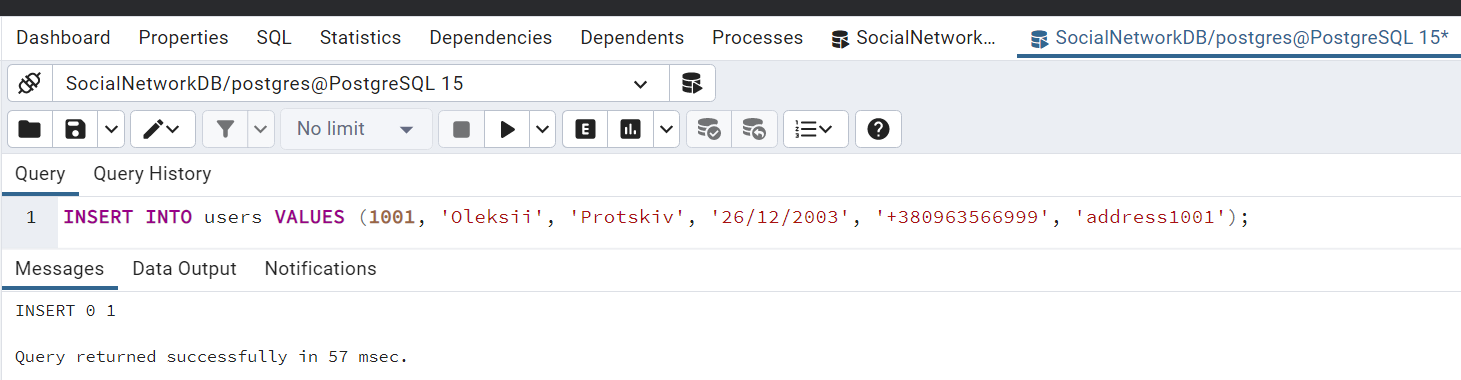


В першій вкладці виконав SELECT з такою умовою, яку задовольняє кілька рядків:

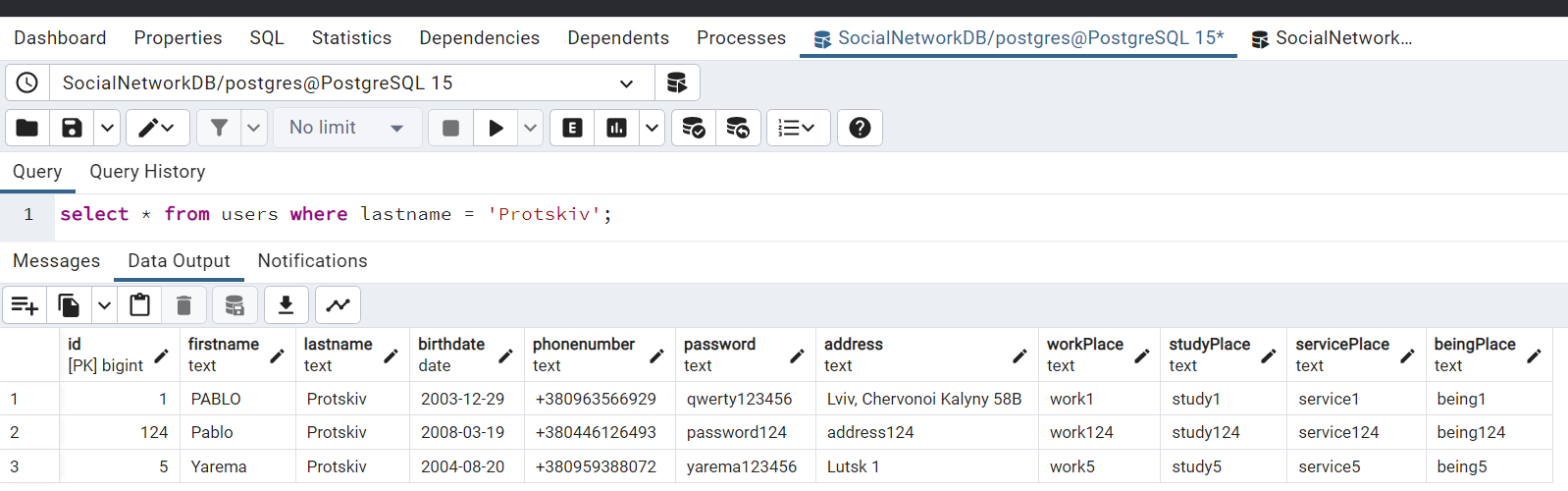


Троє користувачів з прізвищем “Процьків”.

В другій вкладці додав в таблицю рядок, який описує ще одного користувача з прізвищем “Процьків”:



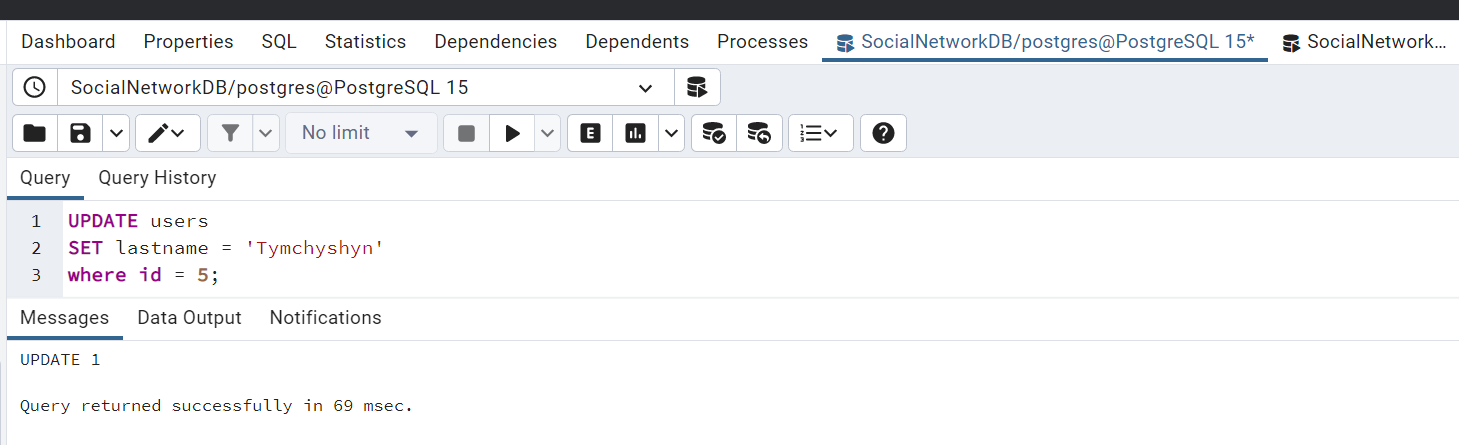
Знову пишу той же SELECT в першій вкладці:



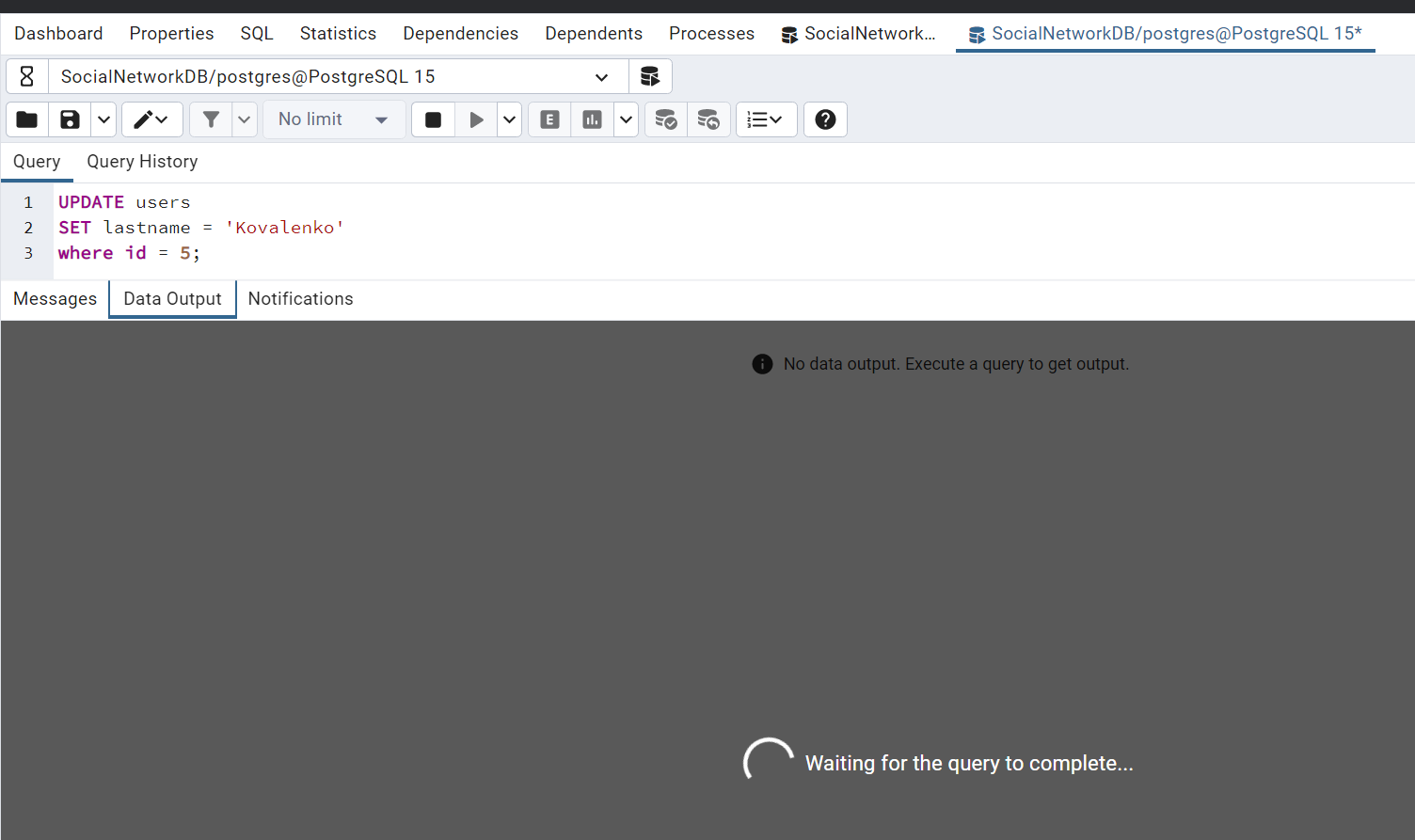
Як було три кортежі даних, так і лишилося, бо рівень ізольованості “Serializable” ніколи не допускає “Phantom read”.

1. Serialization anomaly - коли транзакції, які виглядають серіалізованими, повертають результати, неможливі під час послідовного виконання тих самих транзакцій:

В першій вкладці оновлюю кортеж користувача з id = 5:



В другій вкладці написав схожий запит, який змінює цей же кортеж:



Цей запит не виконався, бо рівень ізоляції “Serializable” не допускає “Serialization anomaly”.

**Висновок:** під час виконання лабораторної роботиознайомився з використанням транзакцій, їх розробкою та застосуванням, рівнями ізоляцій та механізмом управління конкурентним доступом в СКБД PostgreSQL.