**Звіт з лабораторної роботи №3  
з дисципліни “Бази даних та інформаційні системи”**

*Студентки групи МІТ-31  
Баш Анастасії*

Тема: Реалізація CRUD операцій з використанням SQL та забезпечення використання процедур, функцій, тригерів та транзакцій.

Метою даної лабораторної роботи було реалізувати основні операції CRUD (створення, читання, оновлення, видалення) за допомогою мови SQL без використання технології ORM або подібних інструментів. Крім того, ми також забезпечили використання процедур, функцій, тригерів та транзакцій для оптимізації та контролю над операціями бази даних.

Бізнес процес обліку абонементів у спортивному клубі та записів на тренування:

Клієнти реєструються у системі, обирають абонементи та оформлюються членство у спотивному клубі. Члени клубу можуть записатись на тренування, вибравши тренера, дату та час тренування.

**Хід лабораторної роботи:**

1. **Створення процедур**

*Процедура додавання члена клубу*

CREATE OR REPLACE PROCEDURE add\_member(

    p\_name VARCHAR,

    p\_surname VARCHAR,

    p\_phone VARCHAR

)

LANGUAGE plpgsql

AS $$

BEGIN

INSERT INTO "Member" (name, surname, phone)

VALUES (p\_name, p\_surname, p\_phone);

END;

$$;

*Процедура додавання тренера*

CREATE OR REPLACE PROCEDURE add\_trainer(

    p\_name VARCHAR,

    p\_surname VARCHAR,

    p\_phone VARCHAR

)

LANGUAGE plpgsql

AS $$

BEGIN

INSERT INTO "Trainer" (name, surname, phone)

VALUES (p\_name, p\_surname, p\_phone);

END;

$$;

*Процедура додавання типу абонементу*

CREATE OR REPLACE PROCEDURE add\_service (

    p\_name VARCHAR,

    p\_price DECIMAL

)

LANGUAGE plpgsql

AS $$

BEGIN

INSERT INTO "Service" (name, price)

VALUES (p\_name, p\_price);

END;

$$;

*Процедура заміни номеру телефона члена клубу*

CREATE OR REPLACE PROCEDURE update\_phonenumber\_member(

    p\_name VARCHAR,

    p\_surname VARCHAR,

    p\_phone VARCHAR

)

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "Member" SET phone = p\_phone

WHERE name = p\_name AND surname = p\_surname;

END;

$$;

*Процедура зміни ціни абонемента*

CREATE OR REPLACE PROCEDURE update\_service\_price(

    p\_name VARCHAR,

    p\_price DECIMAL

)

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "Service" SET price = p\_price

WHERE name = p\_name;

END;

$$;

*Процедура видалення типу абонементу*

CREATE OR REPLACE PROCEDURE delete\_service(

    p\_name VARCHAR

)

LANGUAGE plpgsql

AS $$

BEGIN

DELETE FROM "Service" WHERE name = p\_name;

END;

$$;

1. **Створення ункцій**

*Функція вираховування середньої тривалості тренування за ID члена клубу*

CREATE OR REPLACE FUNCTION avg\_duration(member\_id INTEGER)

RETURNS DECIMAL

LANGUAGE plpgsql

AS $$

DECLARE

    total\_duration INTERVAL;

    avg\_duration DECIMAL;

    training\_count INTEGER;

BEGIN

    -- отримати загальну тривалість тренувань та кількість тренувань для даного клієнта

    SELECT SUM(t.duration), COUNT(\*)

    INTO total\_duration, training\_count

    FROM "Training" t

    JOIN "Membership" m ON t.membership\_id = m.id

    WHERE m.member\_id = avg\_duration.member\_id;

    -- якщо тренувань немає, повернути 0

    IF training\_count = 0 THEN

        RETURN 0;

    END IF;

    -- розрахувати

    avg\_duration := EXTRACT(EPOCH FROM total\_duration) / training\_count;

    RETURN avg\_duration;

END;

$$;

*Функція, що вираховує кількість абонементів за їх типом*

CREATE OR REPLACE FUNCTION ms\_by\_service(service\_name VARCHAR)

RETURNS INTEGER

LANGUAGE plpgsql

AS $$

DECLARE

    count INTEGER;

BEGIN

    SELECT COUNT(\*) INTO count

    FROM "Membership" m

    JOIN "Service" s ON m.service\_id = s.id

    WHERE s.name = ms\_by\_service.service\_name;

    RETURN count;

END;

$$;

-- перевірка

SELECT ms\_by\_service('Platinum');

1. **Створення транзакцій**

*Транзакція для додавання нового клієнта та оформлення його абонементу*

BEGIN;

INSERT INTO "Member" (name, surname, phone)

VALUES ('Liza', 'Kenedy', '000 000 0000');

INSERT INTO "Membership" (member\_id, service\_id, start\_date, end\_date) VALUES (454, 3, '25/12/2023', '15/12/2024');

COMMIT;

*Транзакція для оновлення номеру телефону клієнта за id*

BEGIN;

UPDATE "Member"

SET phone = '987 654 3266'

WHERE id = 454;

COMMIT;

*Транзакція для видалення тренування за id*

BEGIN;

DELETE FROM "Training" WHERE id = 98;

COMMIT;

1. **Створення тригерів**

*Тригер на видалення абонементу після видалення члена клубу*

CREATE OR REPLACE FUNCTION delete\_related\_member()

RETURNS TRIGGER AS $$

BEGIN

    DELETE FROM "Membership"

    WHERE member\_id = OLD.id;

    RETURN OLD;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_delete\_related\_member

AFTER DELETE ON "Member"

FOR EACH ROW

EXECUTE FUNCTION delete\_related\_member();

1. ***Індекси***

*Індексування полів name та surname таблиці Member*

CREATE INDEX idx\_member\_name\_surname ON "Member" (name, surname);

SELECT id, name, surname, phone

FROM "Member"

WHERE name = 'John' AND surname = 'Doe';

*Індексування дати тренування date таблиці Training*

CREATE INDEX idx\_training\_date ON "Training" (date);

SELECT id, date, time, trainer\_id

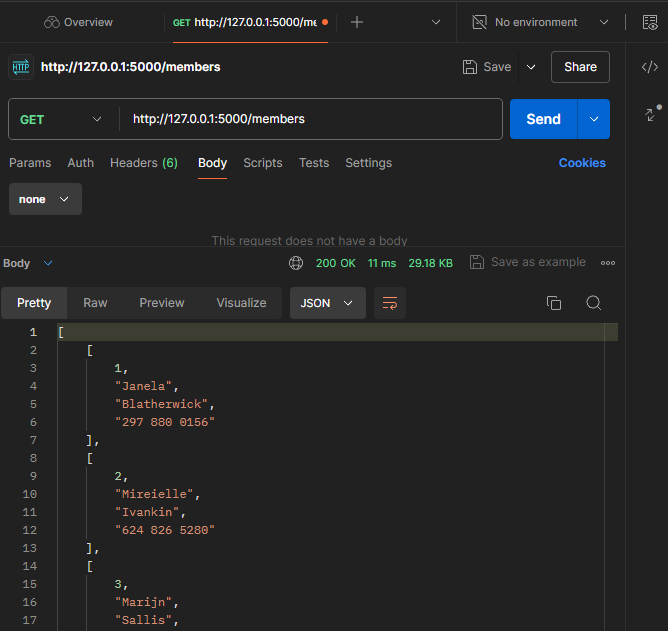
FROM "Training"

WHERE date BETWEEN '2024-01-01' AND '2024-01-31';

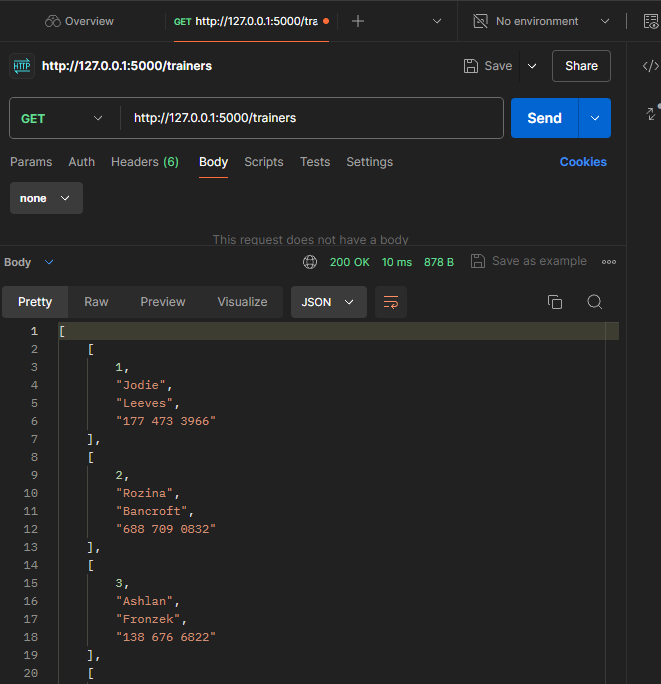
**Виконання CRUD операцій з використанням SQL за допомогою REST сервісу**

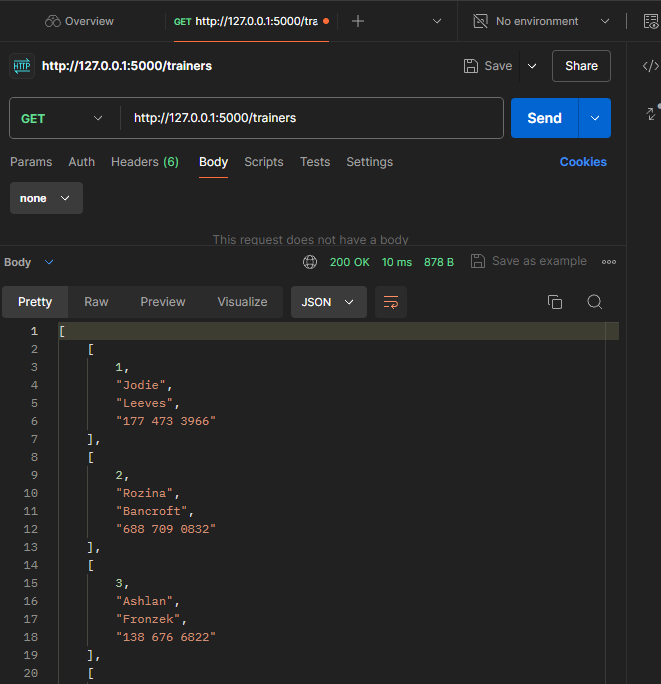
1. **GET**

*Отримання даних таблиці Member:*

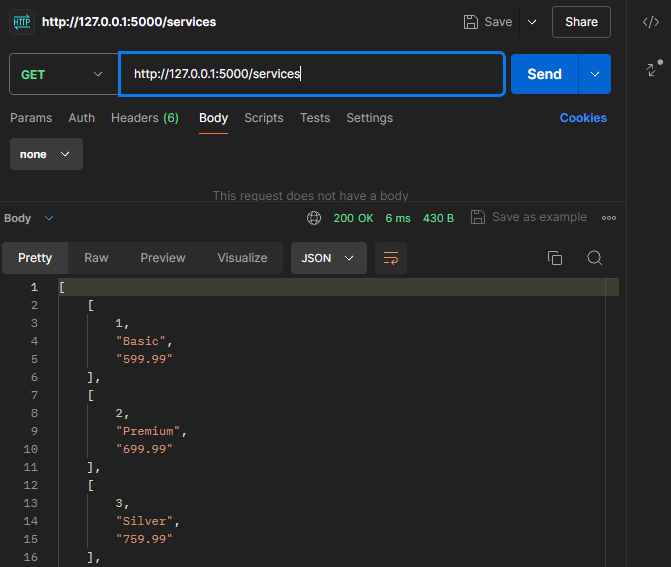


*Таблиці Trainer:*



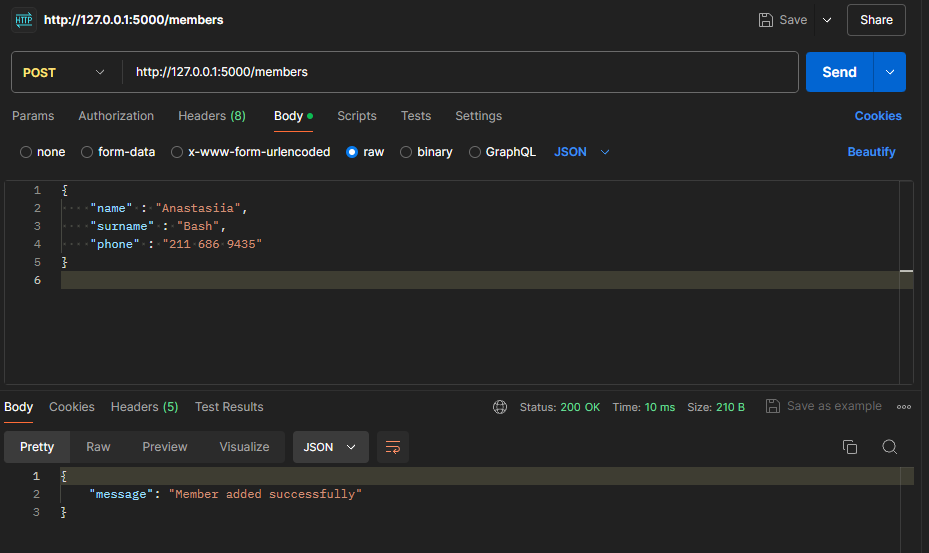


*Таблиці Service:*

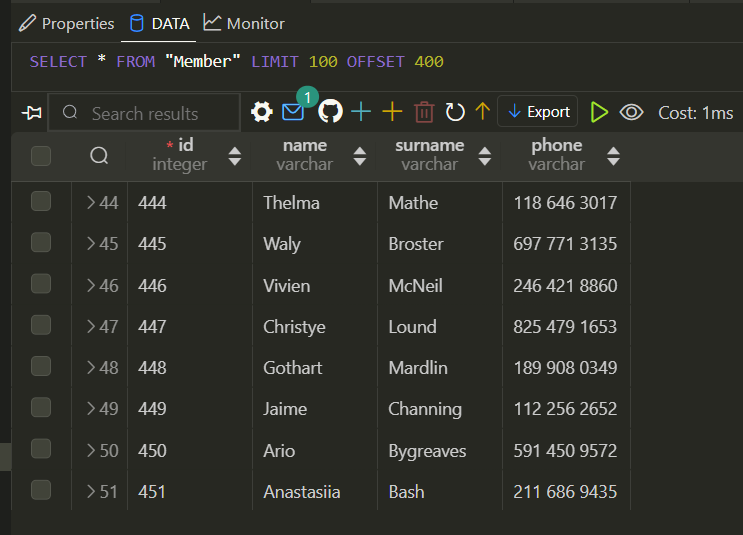


1. **POST (procedures add\_member, add\_trainer, add\_service)**

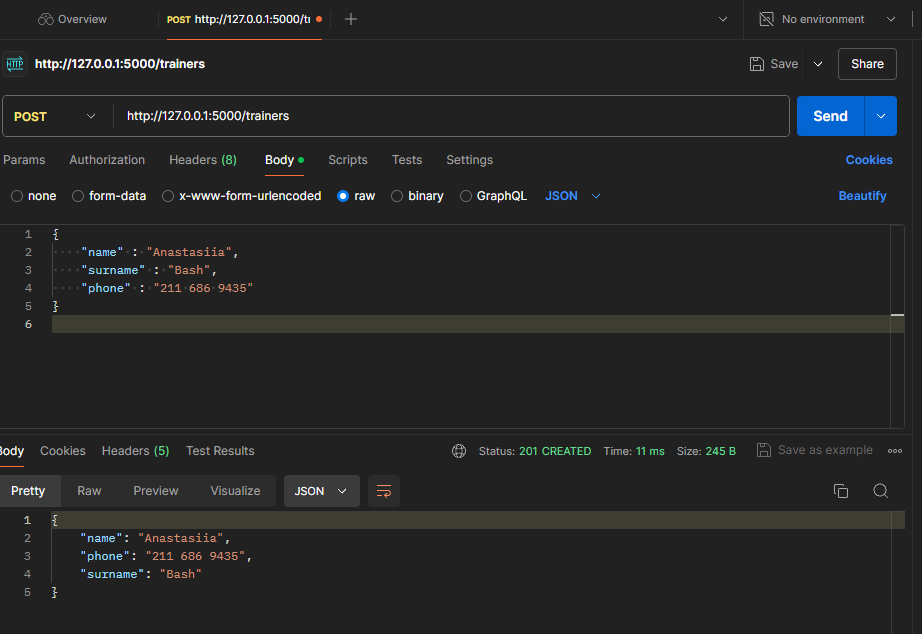
*Додавання члена клубу*



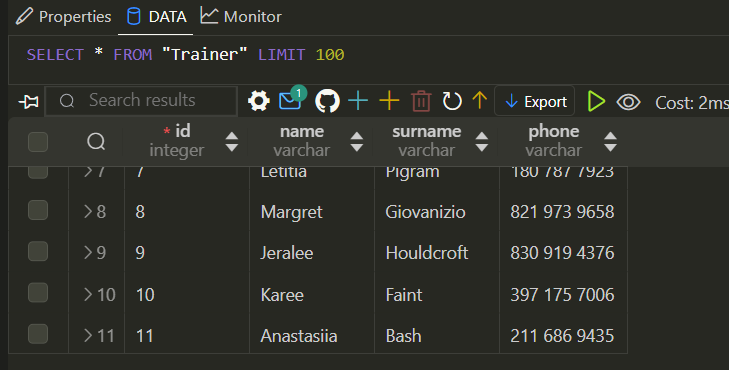
*Результат додавання видно у таблиці*:



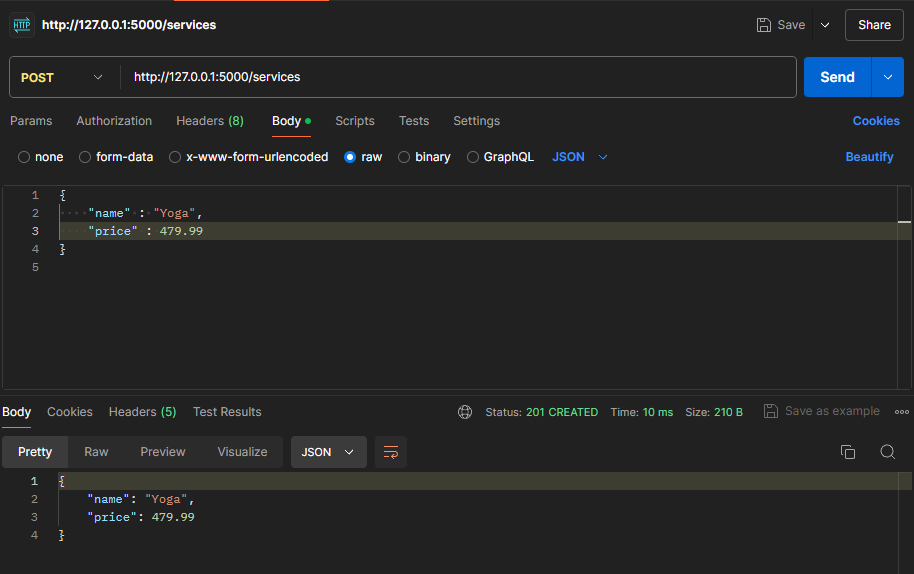
*Додавання тренера:*



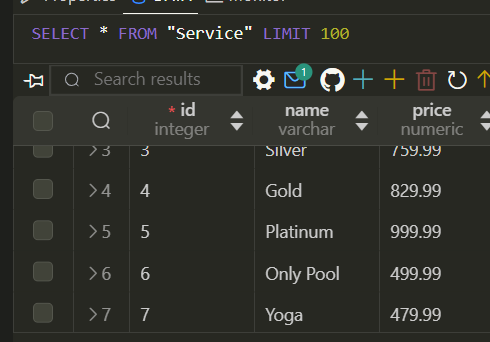
*Результати роботи:*



*Додавання типу абонемента*

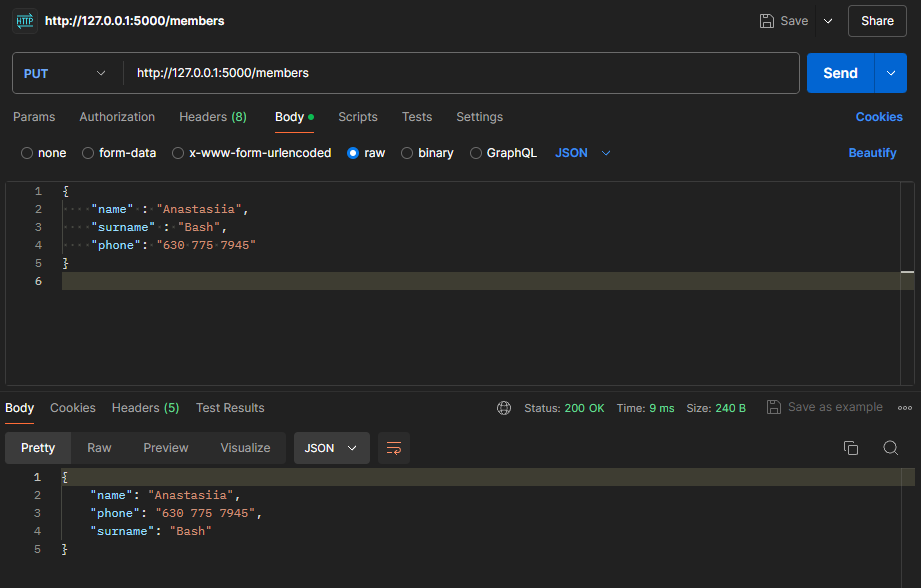


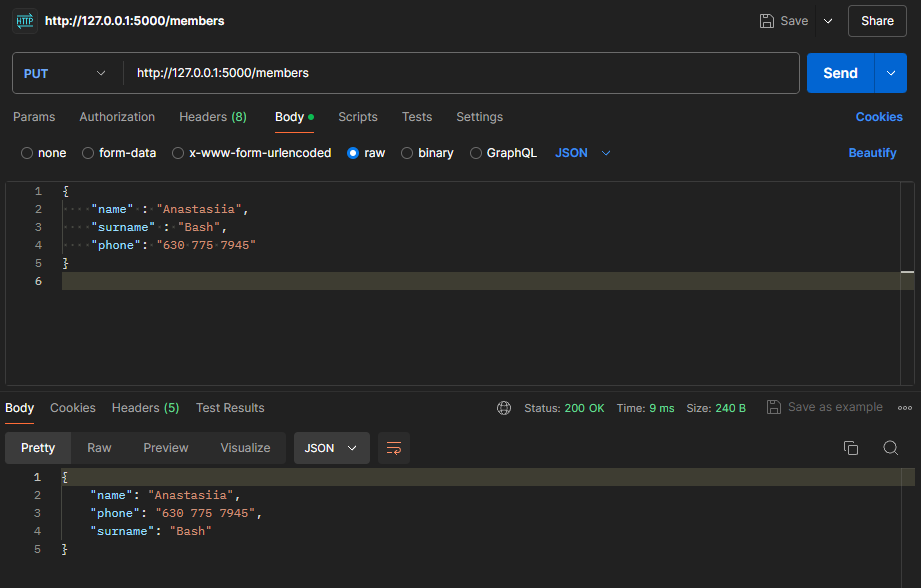
*Результат роботи:*



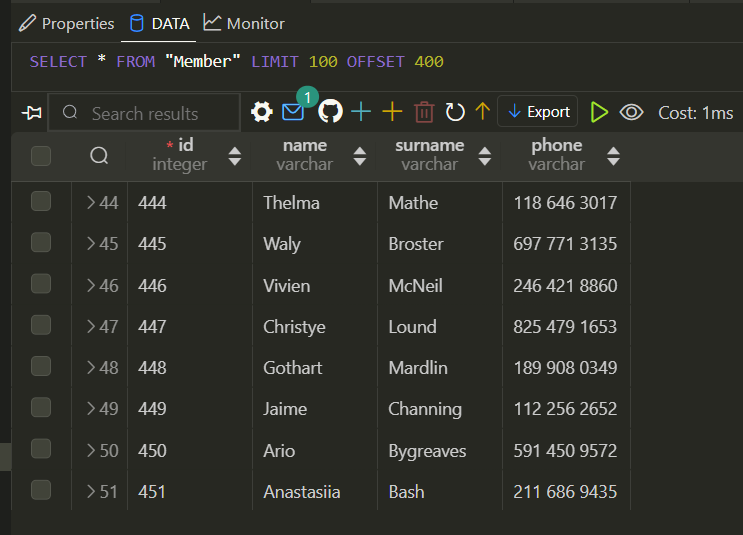
1. **PUT (procedures update\_phonenumber\_member, update\_service\_price)**

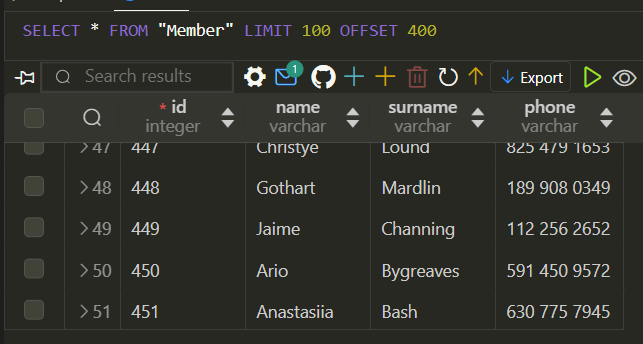
*Зміна номера телефону клієнта за його ID:*



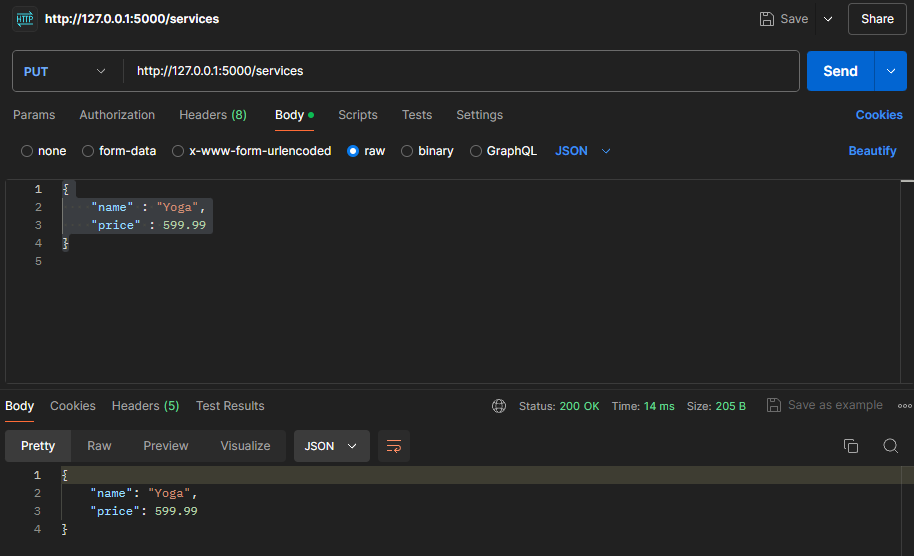


*Результати роботи*

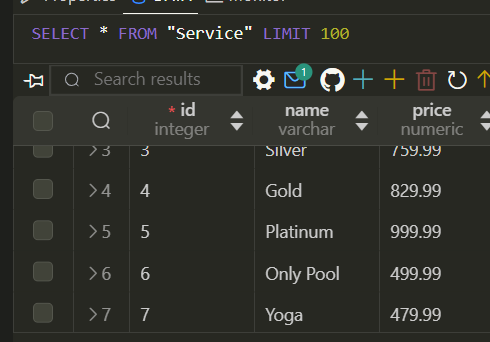


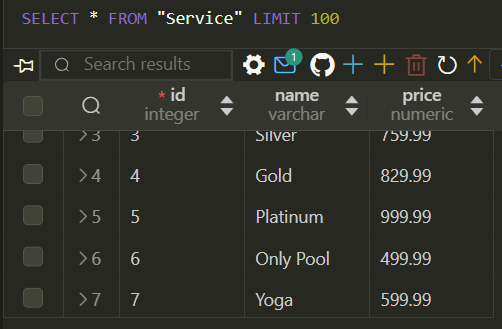


*Зміна ціни на абонемент:*



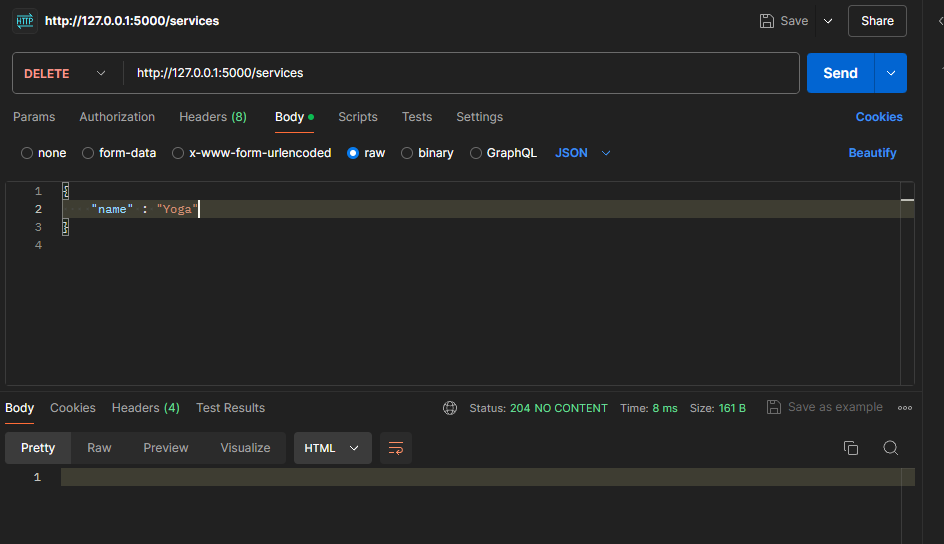
*Результат роботи:*



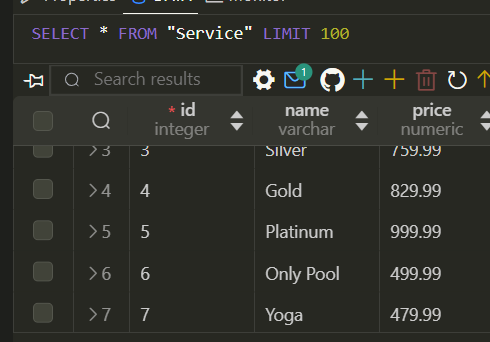


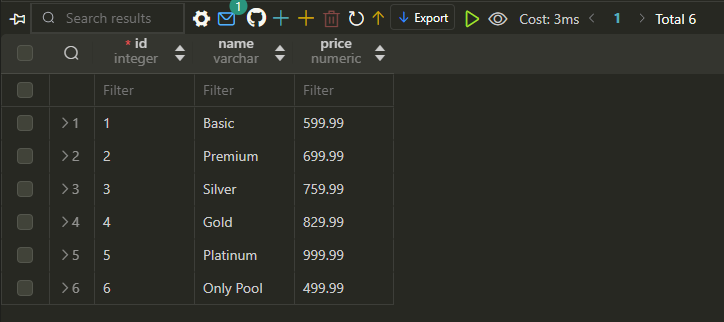
1. **DELETE (procedure delete\_service)**

*Видалення типу абонементу:*



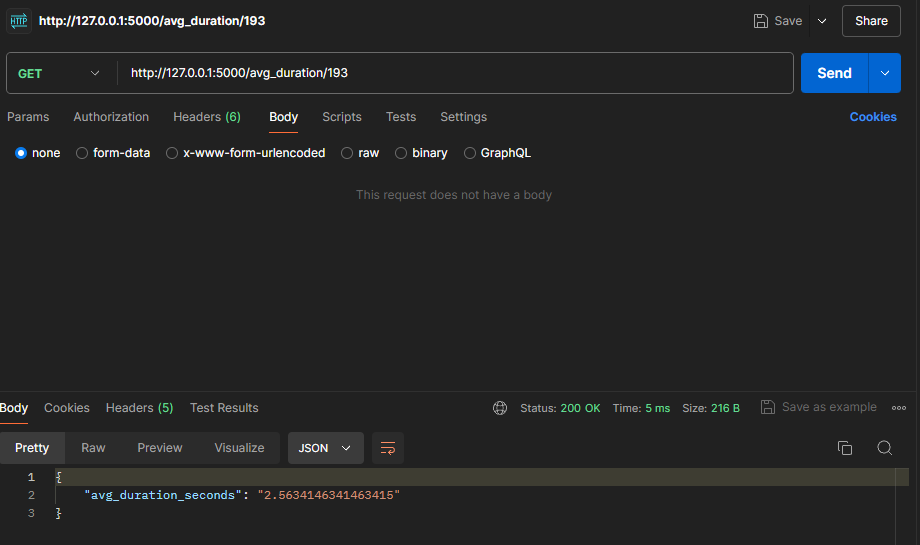
*Результати роботи:*



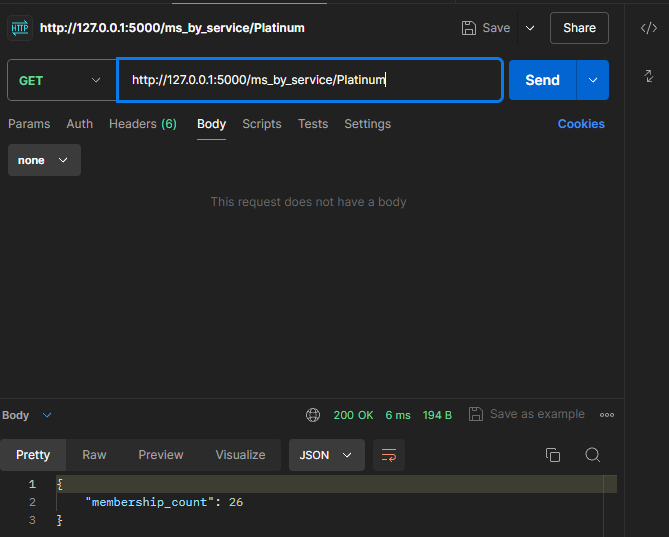


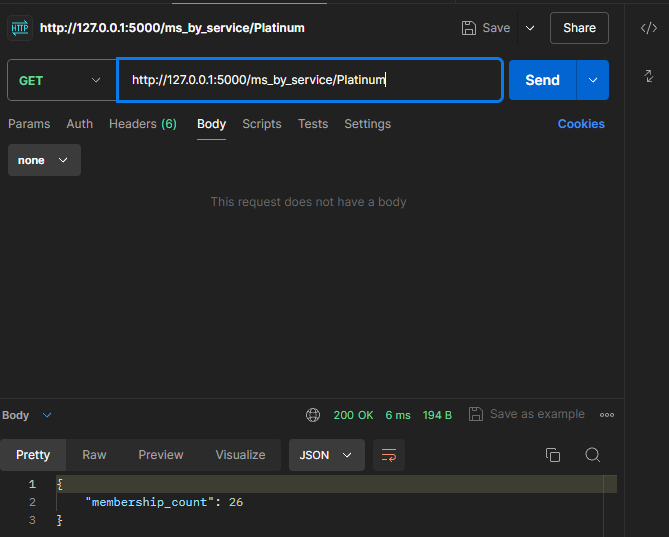
1. **GET (functions)**

*Вирахування середньjї тривалості тренування клієнта за його ID:*



*Підрахунок кількості абонементів за його типом:*





**Висновки:**

У результаті лабораторної роботи було успішно реалізовано CRUD операції з використанням мови SQL. Також було забезпечено використання процедур, функцій, тригерів та транзакцій для оптимізації та забезпечення цілісності даних у базі.

*Додатки:*

1. Procedures – .sql файл зі створенням процедур для лабороторної роботи
2. Functions - .sql файл зі створенням функцій для лабороторної роботи
3. Transaction - .sql файл зі створенням транзакцій для БД.
4. Trigers - . sql файл зі створенням тригерів для БД.
5. Index - .sql файл зі створенням індексів та їх перевіркою для БД.
6. app.py – файл з реалізацією фласкового додатку для роботи з БД.