Лабораторна робота №4

З дисципліни: Бази даних та інформаційні системи
Студента групи МІТ-31: Добровольського Дмитра

Тема: Розширення можливостей PostgreSQL: користувацькі типи, функції та тригери

Мета роботи: Закріпити знання з розширюваності PostgreSQL. Навчитися створювати користувацькі типи даних. Реалізувати власну користувацьку функцію або агрегат. Створити тригери для логування змін у базі даних. Автоматично оновлювати пов’язані таблиці чи заповнювати значення. Оновити діаграму бази даних відповідно до виконаних завдань. Перевірити коректність роботи реалізованих об’єктів через виконання тестових SQL-запитів.

Завдання:

1. Створення користувацького типу даних

   Додаємо ENUM тип для статуса пацієнта, наприклад: 'активний', 'виписаний', 'переведений'.

```
-- Створення ENUM типу для статусу пацієнта
CREATE TYPE patient_status AS ENUM ('активний', 'виписаний', 'переведений');

-- Додавання колонки до таблиці пацієнтів
ALTER TABLE patients ADD COLUMN status patient_status DEFAULT 'активний';
```

   Результат:

	id [PK] integer	full_name character varying (100)	birth_date date	gender character varying (10)	phone character varying (20)	status patient_status
1	2	Олена Петрівна	1985-05-20	жінка	+380631234567	активний
2	3	Петро Сидоренко	1975-12-05	чоловік	+380971234567	активний
3	4	Марія Дубенко	1985-05-20	жінка	+380671112233	активний
4	5	д-р Оксана Бондаренко	1992-08-08	жінка	+380671234888	активний
5	6	Андрій Безприйомний	2000-07-15	чоловік	+380991234321	активний

2. Створення користувацької функції або агрегату

   - Наступним кроком ми реалізуємо функцію для обчислення середньої кількості прийомів на лікаря.

```
-- функція для обчислення середньої кількості прийомів на лікаря.
CREATE OR REPLACE FUNCTION avg_appointments_per_doctor() RETURNS NUMERIC AS $$
DECLARE
    result NUMERIC;
BEGIN
    SELECT COUNT(*) * 1.0 / COUNT(DISTINCT doctor_id) INTO result FROM appointments;
    RETURN result;
END;
$$ LANGUAGE plpgsql;

-- Тестовий запит:
SELECT avg_appointments_per_doctor();
```

   Результат:

	avg_appointments_per_doctor numeric
1	2.0000000000000000

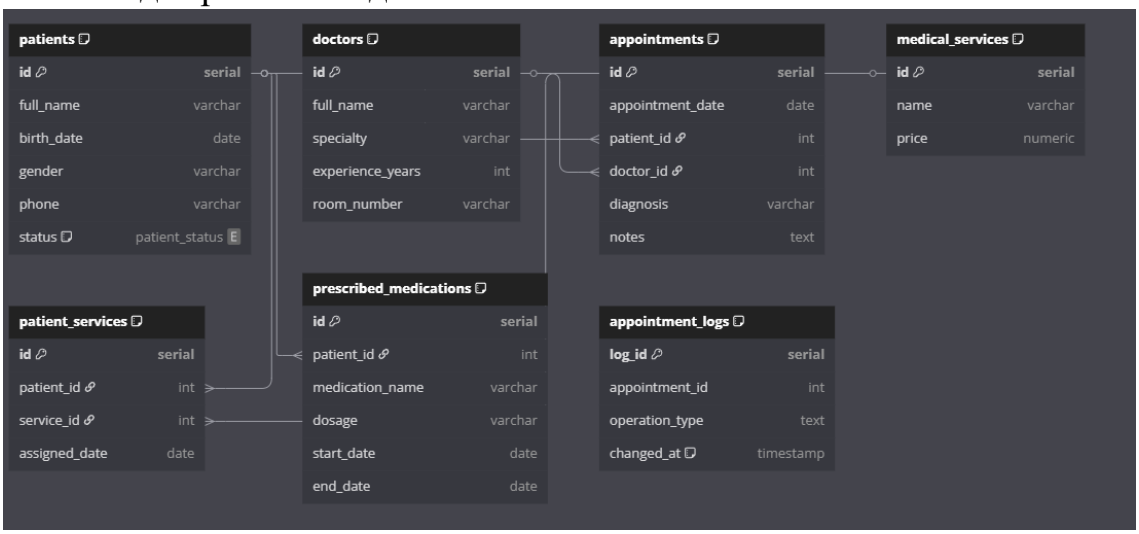
3. Створення тригерів для логування змін та автоматичного оновлення пов’язаних таблиць
   - створюємо лог-таблицю для змін у таблиці appointments

```sql
-- Таблиця для логів
CREATE TABLE appointment_logs (
    log_id SERIAL PRIMARY KEY,
    appointment_id INT,
    operation_type TEXT,
    changed_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Функція логування
CREATE OR REPLACE FUNCTION log_appointment_changes() RETURNS TRIGGER AS $$
BEGIN
    INSERT INTO appointment_logs (appointment_id, operation_type)
    VALUES (
        COALESCE(NEW.id, OLD.id),
        TG_OP
    );
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Тригер
CREATE TRIGGER trg_log_appointment
AFTER INSERT OR UPDATE OR DELETE ON appointments
FOR EACH ROW
EXECUTE FUNCTION log_appointment_changes();
```

4. Оновлення діаграми бази даних



5. Перевірка роботи

```sql
-- Тест 1: Змінимо статус пацієнта
UPDATE patients
SET status = 'виписаний' WHERE id = 1;
```

	id [PK] integer	full_name character varying (100)	birth_date date	gender character varying (10)	phone character varying (20)	status patient_status
1	2	Олена Петрівна	1985-05-20	жінка	+380631234567	активний
2	3	Петро Сидоренко	1975-12-05	чоловік	+380971234567	активний
3	4	Марія Дубенко	1985-05-20	жінка	+380671112233	активний
4	5	д-р Оксана Бондаренко	1992-08-08	жінка	+380671234888	активний
5	6	Андрій Безприйомний	2000-07-15	чоловік	+380991234321	активний
6	1	Іван Іваненко	1990-01-10	чоловік	+380501112233	виписаний

```
-- Тест 2: Додамо новий прийом
INSERT INTO appointments (appointment_date, patient_id, doctor_id, diagnosis, notes)
VALUES (CURRENT_DATE, 1, 1, 'Новий діагноз', 'Примітка до прийому');

select * from appointments;
```

	id [PK] integer	appointment_date date	patient_id integer	doctor_id integer	diagnosis text	notes text
1	1	2025-04-10	1	1	Гіпертонія	Призначено аналіз крові
2	2	2025-04-11	2	2	Застуда	Рекомендовано постільний режим
3	3	2025-04-12	3	3	Мігрень	Призначено обстеження
4	4	2025-04-14	2	1	ГРВІ	Призначено симптоматичне лікування
5	5	2025-04-23	1	2	Головний біль	Рекомендовано обстеження
6	6	2025-04-23	1	1	Новий діагноз	Примітка до прийому

```
-- Тест 3: Перевіримо лог
SELECT * FROM appointment_logs ORDER BY changed_at DESC;
```

	log_id [PK] integer	appointment_id integer	operation_type text	changed_at timestamp without time zone
1	1	6	INSERT	2025-04-23 01:21:50.673318

```
-- Тест 4: Виклик користувацької функції
SELECT avg_appointments_per_doctor();
```

	avg_appointments_per_doctor numeric
1	2.0000000000000000

Висновок:

У цій лабораторній роботі база даних була розширена за допомогою користувацького типу ENUM, функції для аналізу даних та тригера для логування змін. Це підвищило гнучкість, контроль та зручність використання системи.