"ПОЛІКЛІНІКА"

Студент групи ПМ-21
Бакош Віктор

СУТЬ ЗАВДАННЯ

Інформаційна система служить для поліклініки, яка обслуговує різні типи пацієнтів. Система повинна проводити облік пацієнтів, історію хвороби кожного з них, вживані медикаменти, лікуючих лікарів.

01 СТВОРЕННЯ СХЕМИ БД

THE PROBLEM?

1) схема даних (малюнок з вказанням типів зв’язків між таблицями та відповідними ключовими полями).
2) SQL-запити для створення та наповнення таблиць.
3) Роз’яснення до схеми даних

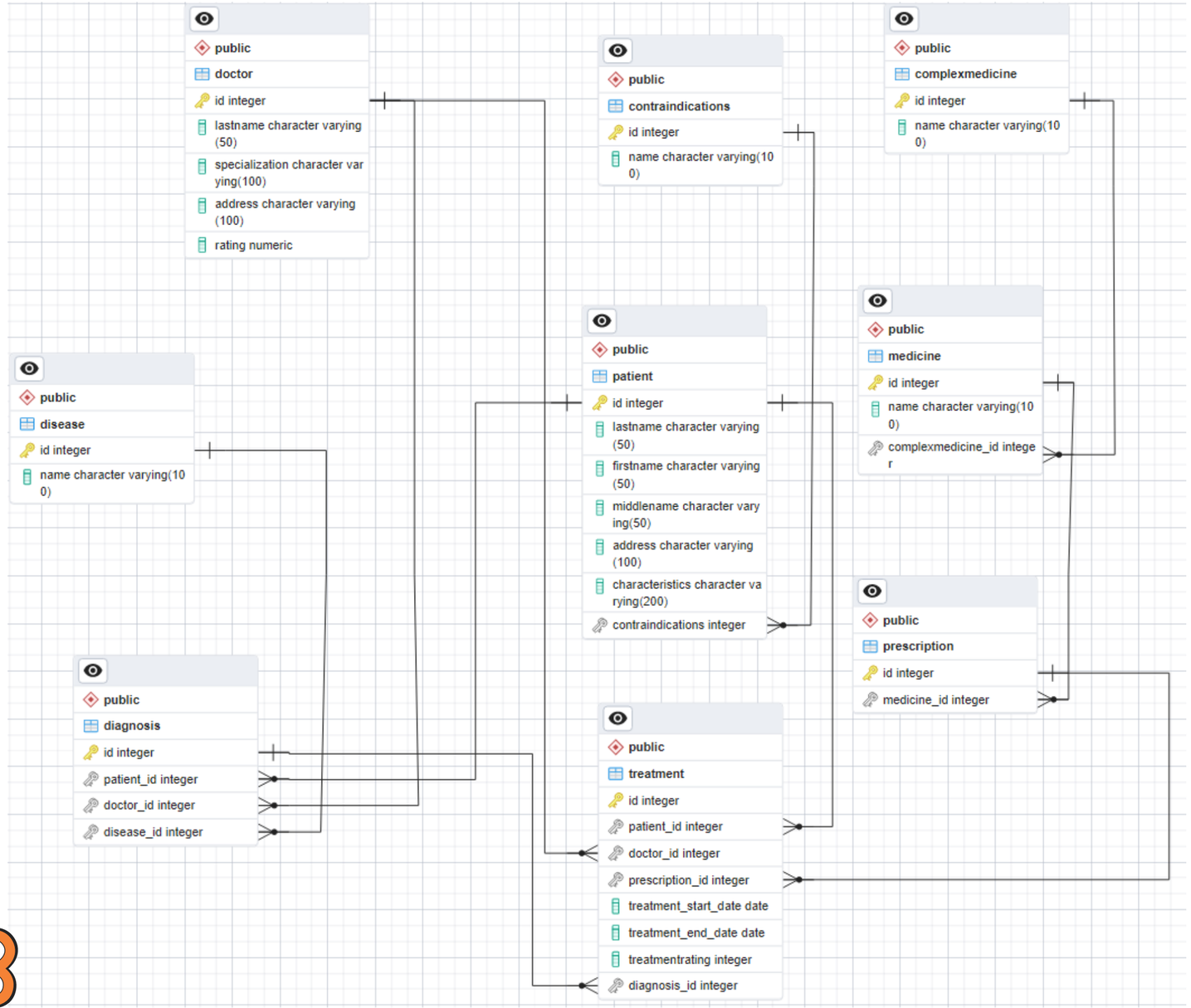


СХЕМА БД

Посилання на роз'яснення схеми БД

(тут)

02 SQL-ЗАПИТИ

Створити запити типу:
SELECT * OR, AND / SELECT * Outer Join / SELECT * Like, Between, In, Exists, All, Any

```sql
SELECT
  p.lastname || ' ' || p.firstname AS patient_name,
  t.treatment_start_date,
  t.treatment_end_date,
  d.lastname AS doctor_name
FROM
  public.patient p
  INNER JOIN public.treatment t ON p.id = t.patient_id
  INNER JOIN public.doctor d ON d.id = t.doctor_id
ORDER BY
  p.lastname, p.firstname, t.treatment_start_date;
```

Data Output | Messages | Notifications

	patient_name text	treatment_start_date date	treatment_end_date date	doctor_name character varying (50)
1	Doe John	2023-01-01	2023-01-31	Smith
2	Doe John	2023-01-01	2023-01-31	Smith
3	Doe John	2023-01-01	2023-01-31	Smith
4	Doe John	2023-01-01	2023-01-31	Smith
5	Doe John	2023-06-01	2023-06-15	Smith
6	Johnson Emily	2023-06-04	2023-06-18	Johnson
7	Smith Jane	2023-02-01	2023-02-28	Smith
8	Smith Jane	2023-02-01	2023-02-28	Smith
9	Smith Jane	2023-02-01	2023-02-28	Smith
10	Smith Jane	2023-02-01	2023-02-28	Johnson
11	Smith Jane	2023-02-01	2023-02-28	Smith
12	Smith Jane	2023-02-01	2023-02-28	Smith
13	Smith Jane	2023-06-02	2023-06-16	Johnson
14	Smith John	2023-06-03	2023-06-17	Smith
15	Williams David	2023-06-05	2023-06-19	Williams

```sql
SELECT *
FROM doctor
WHERE specialization = 'Cardiology' AND (rating > 4.5 OR address = 'New York')
ORDER BY rating DESC;
```

Data Output | Messages | Notifications

	id [PK] integer	lastname character varying (50)	specialization character varying (100)	address character varying (100)	rating numeric
1	3	Smith	Cardiology	New York	26.90000000000000000

```sql
SELECT doctor_id, COUNT(*) AS patient_count
FROM treatment
GROUP BY doctor_id;
```

Data Output | Messages | Notifications

	doctor_id integer	patient_count bigint
1	3	1
2	5	1
3	4	1
4	2	2
5	1	10

ВСІ ЗАПИТИ (ТУТ)

03 ПІДПРОГРАМИ СУБД

Написати процедуру визначення рейтингу кожного лікаря на основі власного алгоритму, який враховував би кількість хворих, тривалість лікування, ефективність лікування, кількість протипоказань, тощо. Написати процедуру, яка б викликала першу процедуру для всіх лікарів

Процедура №1

```sql
-- PROCEDURE: public.calculate_doctor_rating(integer)

-- DROP PROCEDURE IF EXISTS public.calculate_doctor_rating(integer);

CREATE OR REPLACE PROCEDURE public.calculate_doctor_rating(
    IN p_doctor_id integer)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    total_patients INTEGER;
    total_treatment_duration INTERVAL;
    total_effectiveness NUMERIC;
    total_contraindications INTEGER;
BEGIN
    total_patients := 0;
    total_treatment_duration := INTERVAL '0';
    total_effectiveness := 0;
    total_contraindications := 0;

    -- Calculate the total patients, treatment duration, effectiveness, and contraindications for the doctor
    SELECT COUNT(DISTINCT t.patient_id), SUM(t.treatment_end_date - t.treatment_start_date), SUM(t.treatmentrating), COUNT(DISTINCT p.contraindications)
    INTO total_patients, total_treatment_duration, total_effectiveness, total_contraindications
    FROM public.treatment t
    JOIN public.patient p ON t.patient_id = p.id
    WHERE t.doctor_id = p_doctor_id;

    -- Calculate the average treatment duration and effectiveness
    IF total_patients > 0 THEN
        total_treatment_duration := total_treatment_duration / total_patients;
        total_effectiveness := total_effectiveness / total_patients;
    END IF;

    -- Calculate the rating based on the custom algorithm and update the doctor table
    UPDATE public.doctor
    SET rating = (
        total_patients * 0.3 +
        EXTRACT(EPOCH FROM total_treatment_duration) * 0.2 +
        total_effectiveness * 0.3 -
        total_contraindications * 0.2
    )
    WHERE id = p_doctor_id;
END;
$BODY$;
ALTER PROCEDURE public.calculate_doctor_rating(integer)
    OWNER TO postgres;

```

Процедура №2

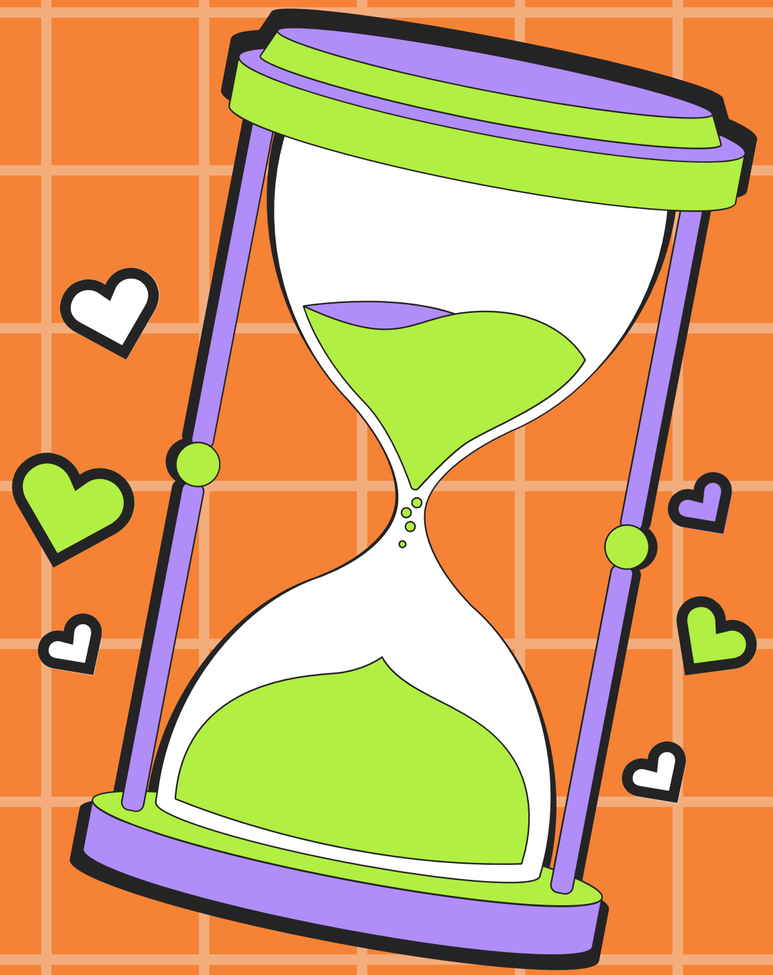
```sql
-- PROCEDURE: public.calculate_all_doctor_ratings()

-- DROP PROCEDURE IF EXISTS public.calculate_all_doctor_ratings();

CREATE OR REPLACE PROCEDURE public.calculate_all_doctor_ratings(
    )
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    doctor_id INTEGER;
BEGIN
    -- Loop through each doctor and calculate the rating
    FOR doctor_id IN SELECT id FROM public.doctor
    LOOP
        call calculate_doctor_rating(doctor_id);
    END LOOP;

    -- Output a message indicating the completion of the calculation
    RAISE NOTICE 'Ratings calculation completed for all doctors.';
END;
$BODY$;
ALTER PROCEDURE public.calculate_all_doctor_ratings()
    OWNER TO postgres;

```

КОД НА GITHUB

[ТУТ]

04 ТРИГЕРИ В СУБД

Пацієнту не можна призначати лікарства, які є протипоказані йому, або містять протипоказані складові. Лікар не може одночасно мати більше 10 пацієнтів.



```sql
-- FUNCTION: public.check_doctor_patient_limit()

-- DROP FUNCTION IF EXISTS public.check_doctor_patient_limit();

CREATE OR REPLACE FUNCTION public.check_doctor_patient_limit()
    RETURNS trigger
    LANGUAGE 'plpgsql'
    COST 100
    VOLATILE NOT LEAKPROOF
AS $BODY$
DECLARE
  patient_count INTEGER;
BEGIN
  -- GET COUNT OF PATIENTS OF THE DOCTOR
  SELECT COUNT(*) INTO patient_count
  FROM public.treatment
  WHERE doctor_id = NEW.doctor_id;

  -- CHECK IF IT'S MEET CONDITION
  IF patient_count >= 10 THEN
    RAISE EXCEPTION 'The doctor already has the maximum number of patients';
  END IF;

  RETURN NEW;
END;
$BODY$;

ALTER FUNCTION public.check_doctor_patient_limit()
    OWNER TO postgres;
-- Trigger: check_doctor_patient_limit_trigger

-- DROP TRIGGER IF EXISTS check_doctor_patient_limit_trigger ON public.treatment;

CREATE TRIGGER check_doctor_patient_limit_trigger
    BEFORE INSERT
    ON public.treatment
    FOR EACH ROW
    EXECUTE FUNCTION public.check_doctor_patient_limit();
```

GIT HUB (ТУТ)

```
ERROR: ПОМИЛКА:  The doctor already has the maximum number of patients
CONTEXT:  Функція PL/pgSQL check_doctor_patient_limit() рядок 12 в RAISE


SQL state: P0001
```

1) Створити 3-4 різнотипних користувача БД
2) Надати створеним користувачам привілеї відповідно до їх типових задач.
3) Створити 2-3 типові ролі для користувачів БД
4) Надати необхідні привілеї створеним ролям.
5) Призначити користувачам ролі.
6) Відкликати у користувача привілей
7) Відкликати роль у користувача.
8) Видалити роль. Видалити користувача.

Звіт тут!

05 АДМІНІСТРУВАННЯ БД

THANK YOU!