МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

******

Лабораторна робота №1

Варіант №10

З дисципліни

“Організація баз даних та знань”

Тема: *«Створення схеми БД»*

*Виконав:*

*Бакош Віктор*

*Прийняв:*

*Польовий В.Є.*

*Львів – 202**3*

Лабораторна робота № 1

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

Кожен пацієнт має ідентифікаційний номер, прізвище, ім’я, по-батькові, адресу, протипоказання на прийом ряду лікарств, характеристику . Кожен з них хворів на певну хворобу протягом деякого періоду часу та мав лікуючого лікаря на цей час. Лікар ставить діагнози підопічному хворому та призначає певні ліки. Лікар характеризується прізвищем, ідентифікаційним номером, спеціалізацією, адресою. Кожен хворий по закінченні періоду хвороби оцінюється по ефективності лікування в 100 бальній шкалі. Деякі ліки є комплексними та складаються з ряду інших ліків. Пацієнту не можна призначати ліки, які є протипоказані йому, або містять протипоказані складові. Лікар не може одночасно мати більше 10 пацієнтів. Пацієнт, який хворіє протягом трьох років неперервно або протягом 7 років сумарно береться на спецоблік.

Система повинна надавати наступні звіти:

* Список лікарів та пацієнтів яких вони лікують на певний момент часу.
* Звіт “Історія хвороби” для певного пацієнта з вказанням хвороб, діагнозів, призначених лікарств.
* Зведена таблиця завантаженості лікарів протягом певного року, де по вертикалі показано список лікарів, по горизонталі - місяці року, а в клітинках кількість хворих.

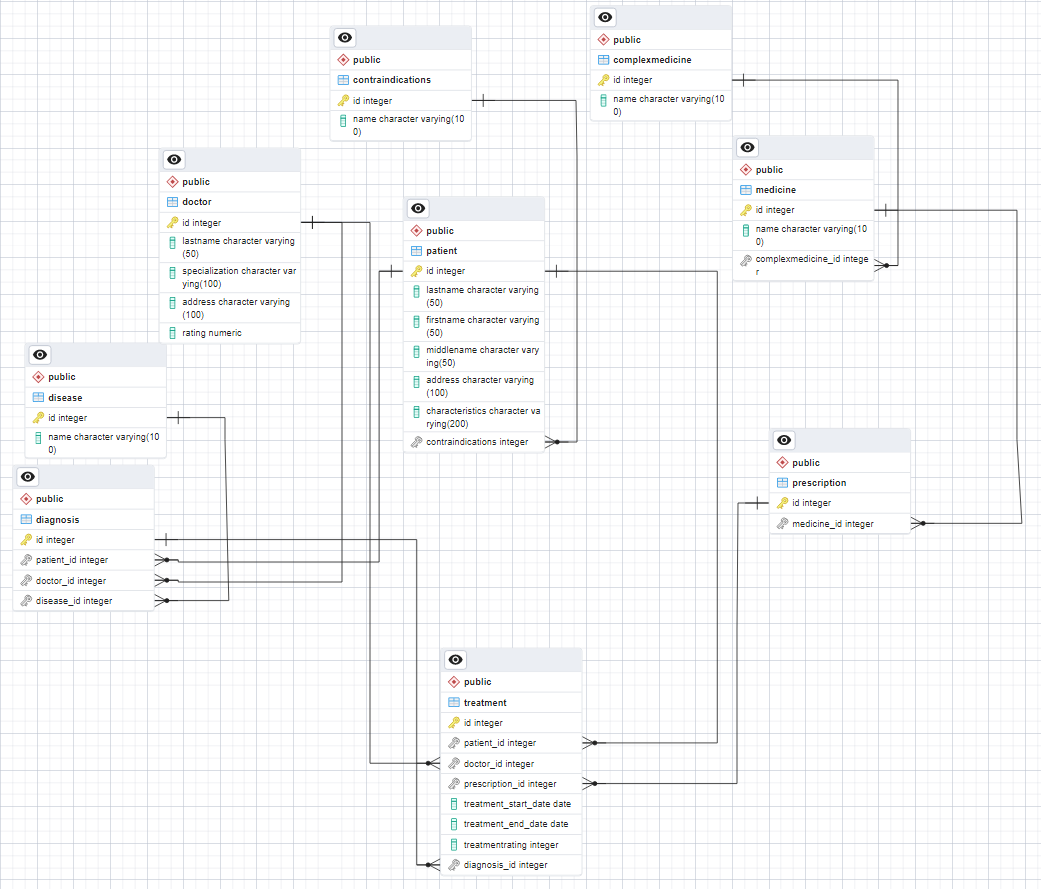
**Завдання до лабораторної роботи:**

**“Створення схеми БД”**

1) Створити концептуальну та логічну модель предметної області згідно з описом, отриманим в викладача (відповідно до обраної предметної області).

2) Створити базу даних: створити таблиці, задати всі потрібні обмеження, створити зв’язки між таблицями. Схема бази даних повинна містити приклади використання сурогатних ключів створених за допомогою послідовностей (sequences). Схема бази даних повинна містити приклади використання всіх можливих способів перевірок обмеження цілісності, зокрема зовнішні ключі, вимоги на унікальність полів, перевірки типу CHECK, перевірки типу NULL/NOT NULL, тощо.

3) Ввести по декілька записів в кожну таблицю, проаналізувати цілісність даних.

Результат виконання лабораторної роботи  
  


SQL запити для створення і заповнення таблиць бази даних

-- Table: public.complexmedicine

-- DROP TABLE IF EXISTS public.complexmedicine;

CREATE TABLE IF NOT EXISTS public.complexmedicine

(

    id integer NOT NULL DEFAULT nextval('complexmedicine\_id\_seq'::regclass),

    name character varying(100) COLLATE pg\_catalog."default",

    CONSTRAINT complexmedicine\_pkey PRIMARY KEY (id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.complexmedicine

    OWNER to postgres;

-- Table: public.contraindications

-- DROP TABLE IF EXISTS public.contraindications;

CREATE TABLE IF NOT EXISTS public.contraindications

(

    id integer NOT NULL DEFAULT nextval('contraindications\_id\_seq'::regclass),

    name character varying(100) COLLATE pg\_catalog."default",

    CONSTRAINT contraindications\_pkey PRIMARY KEY (id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.contraindications

    OWNER to postgres;

GRANT ALL ON TABLE public.contraindications TO postgres;

-- Table: public.diagnosis

-- DROP TABLE IF EXISTS public.diagnosis;

CREATE TABLE IF NOT EXISTS public.diagnosis

(

    id integer NOT NULL DEFAULT nextval('diagnosis\_id\_seq'::regclass),

    patient\_id integer,

    doctor\_id integer,

    disease\_id integer,

    CONSTRAINT diagnosis\_pkey PRIMARY KEY (id),

    CONSTRAINT diagnosis\_disease\_id\_fkey FOREIGN KEY (disease\_id)

        REFERENCES public.disease (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION,

    CONSTRAINT diagnosis\_doctor\_id\_fkey FOREIGN KEY (doctor\_id)

        REFERENCES public.doctor (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION,

    CONSTRAINT diagnosis\_patient\_id\_fkey FOREIGN KEY (patient\_id)

        REFERENCES public.patient (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.diagnosis

    OWNER to postgres;

-- Table: public.disease

-- DROP TABLE IF EXISTS public.disease;

CREATE TABLE IF NOT EXISTS public.disease

(

    id integer NOT NULL DEFAULT nextval('disease\_id\_seq'::regclass),

    name character varying(100) COLLATE pg\_catalog."default",

    CONSTRAINT disease\_pkey PRIMARY KEY (id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.disease

    OWNER to postgres;

-- Table: public.doctor

-- DROP TABLE IF EXISTS public.doctor;

CREATE TABLE IF NOT EXISTS public.doctor

(

    id integer NOT NULL DEFAULT nextval('doctor\_id\_seq'::regclass),

    lastname character varying(50) COLLATE pg\_catalog."default",

    specialization character varying(100) COLLATE pg\_catalog."default",

    address character varying(100) COLLATE pg\_catalog."default",

    rating numeric,

    CONSTRAINT doctor\_pkey PRIMARY KEY (id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.doctor

    OWNER to postgres;

-- Table: public.medicine

-- DROP TABLE IF EXISTS public.medicine;

CREATE TABLE IF NOT EXISTS public.medicine

(

    id integer NOT NULL DEFAULT nextval('medicine\_id\_seq'::regclass),

    name character varying(100) COLLATE pg\_catalog."default",

    complexmedicine\_id integer,

    CONSTRAINT medicine\_pkey PRIMARY KEY (id),

    CONSTRAINT complexmedecine\_fk FOREIGN KEY (complexmedicine\_id)

        REFERENCES public.complexmedicine (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.medicine

    OWNER to postgres;

-- Table: public.patient

-- DROP TABLE IF EXISTS public.patient;

CREATE TABLE IF NOT EXISTS public.patient

(

    id integer NOT NULL DEFAULT nextval('patient\_id\_seq'::regclass),

    lastname character varying(50) COLLATE pg\_catalog."default",

    firstname character varying(50) COLLATE pg\_catalog."default",

    middlename character varying(50) COLLATE pg\_catalog."default",

    address character varying(100) COLLATE pg\_catalog."default",

    characteristics character varying(200) COLLATE pg\_catalog."default",

    contraindications integer,

    CONSTRAINT patient\_pkey PRIMARY KEY (id),

    CONSTRAINT contraindications\_fk FOREIGN KEY (contraindications)

        REFERENCES public.contraindications (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.patient

    OWNER to postgres;

GRANT ALL ON TABLE public.patient TO postgres;

-- Table: public.prescription

-- DROP TABLE IF EXISTS public.prescription;

CREATE TABLE IF NOT EXISTS public.prescription

(

    id integer NOT NULL DEFAULT nextval('prescription\_id\_seq'::regclass),

    medicine\_id integer,

    CONSTRAINT prescription\_pkey PRIMARY KEY (id),

    CONSTRAINT prescription\_medicineid\_fkey FOREIGN KEY (medicine\_id)

        REFERENCES public.medicine (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.prescription

    OWNER to postgres;

-- Table: public.treatment

-- DROP TABLE IF EXISTS public.treatment;

CREATE TABLE IF NOT EXISTS public.treatment

(

    id integer NOT NULL DEFAULT nextval('treatment\_id\_seq'::regclass),

    patient\_id integer,

    doctor\_id integer,

    prescription\_id integer,

    treatment\_start\_date date NOT NULL,

    treatment\_end\_date date NOT NULL,

    treatmentrating integer,

    diagnosis\_id integer,

    CONSTRAINT treatment\_pkey PRIMARY KEY (id),

    CONSTRAINT diagnosis\_fk FOREIGN KEY (diagnosis\_id)

        REFERENCES public.diagnosis (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION,

    CONSTRAINT doctor\_fk FOREIGN KEY (doctor\_id)

        REFERENCES public.doctor (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION,

    CONSTRAINT patient\_fk FOREIGN KEY (patient\_id)

        REFERENCES public.patient (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION,

    CONSTRAINT prescription\_kf FOREIGN KEY (prescription\_id)

        REFERENCES public.prescription (id) MATCH SIMPLE

        ON UPDATE NO ACTION

        ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.treatment

    OWNER to postgres;