

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки
Кафедра автоматика та управління в технічних системах

Курсова робота

з дисципліни «Бази даних»

на тему: База даних «Лікувального закладу»

Керівник

Виконавець

ст. Макаров І. С.

«Допущений до захисту»

залікова книжка № ІТ – 0213

гр. ІТ-02

(Особистий підпис керівника)
« » _____ 2020р.

(особистий підпис виконавця)

Захищений з оцінкою

« » _____ 2020р.

(оцінка)

Члени комісії:

(особистий підпис)

(розшифровка підпису)

(особистий підпис)

(розшифровка підпису)

Київ – 2020

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

(назва навчального закладу)

Кафедра АВТОМАТИКИ ТА УПРАВЛІННЯ В ТЕХНІЧНИХ СИСТЕМАХ
Дисципліна «Бази даних – 2. Практичні прийоми створення та супроводження
реляційних баз даних»

Курс 2 Група ІТ-02 Семестр 3

З А В Д А Н Н Я **НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Макаров Ілля Сергійович
(прізвище, ім'я, по батькові)

1 Тема роботи База даних лікувального закладу _____

керівник роботи Ліщук К. І.
(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

2 Строк подання студентом роботи: до 24.12.2021

3 Вихідні дані до роботи: розробити базу даних для підтримки діловодства
організації

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

1) Вивчення літератури

2) Аналіз предметного середовища

3) Побудова ER-діаграми

4) Побудова реляційної схеми

5) Створення бази даних

6) Створення користувачів бази даних

7) Імпорт даних з використанням засобів СУБД

8) Створення запитів до розробленої БД

9) Оптимізація роботи запитів (за необхідності)

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

КАЛЕНДАРНИЙ ПЛАН

№, п/п	Назва етапів виконання курсової роботи	Строк виконання етапів роботи	Підписи або примітки
1.	Отримання завдання	22.09.20	
2.	Титульний аркуш, завдання, анотація	02.10.20	
3.	Вступ, постановка задачі	05.10.20	
4.	Огляд існуючих рішень	07.10.20	
5.	Визначення вимог до проекту бази даних	10.10.20	
6.	Інфологічне проектування	15.10.20	
7.	Визначення вимог до операційного середовища, вибір СУБД	20.10.20	
8.	Даталогічне проектування	25.10.20	
9.	Фізичне проектування	16.11.20	
10.	Діаграма класів та структура застосунку	04.12.20	
11.	Інструкція, висновки, додатки, джерела	10.12.20	
12.	Захист	15.12.20	

Студент _____
(підпис)
батькові)

Макаров І.С
(прізвище, ім'я, по

Керівник _____
(підпис)
батькові)

Ліщук К. І.
(прізвище, ім'я, по

«_____» _____ 20__ р.

ЗМІСТ

ВСТУП

- ТЕОРЕТИЧНА ЧАСТИНА

- о Аналіз предметної області
 - Опис предметної області
 - Опис вхідних даних
 - Опис вихідних даних
- о Проектування бази даних
 - Інфологічна модель бази даних
 - Опис сутностей
 - Опис атрибутів
 - Опис зв'язків
 - ER-діаграма
 - Нормалізація таблиць при проектуванні бази даних.

- ПРАКТИЧНА ЧАСТИНА

- о Створення бази даних за допомогою MS SQL Server
 - Створення бази даних
 - Створення таблиць бази даних
 - Створення діаграми бази даних
 - Заповнення таблиць бази даних даними
 - Створення тригерів
 - Створення представлень
 - Створення функцій
 - DLM запити

■ Результати оптимізації

ВИСНОВКИ

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

ДОДАТКИ

Додаток А. Використані запити

ВСТУП

Останні кілька років чітко показали нам, наскільки системами охорони здоров'я є важливою складовою будь-якої держави. Сучасні можливості інтернету дозволяють нам значно спростити, та покращити організацію існуючої системи лікування та профілактики громадян. Ядром будь-якої системи є її база даних, саме тому, я вирішив розробити та описати базу даних для місцевого лікарняного комплексу.

Чому ця тема важлива? Грамотно організована та повноцінно функціонуюча база даних дозволить нам з вами швидко, а головне комфортно взаємодіяти з системою охорони здоров'я. Мати можливість записатися до лікаря, на медичну процедуру, переглянути свої записи і все це не виходячи з дому, не створюючи натовпу у чергах, що значно знизить вірогідність не захворіти під час візиту до лікарні. Разом із цим, така система значно спростить роботу лікарів, вже не треба писати величезних направлень на процедури, досить просто створити запис у базі, про неї. Лікарі легко та зручно будуть мати можливість передивитися свої записи на сьогодні, а якісна система валідації позбавить нас від “людського фактору” тепер не вийде записатися до лікаря у не робочий час, або якщо на цей час лікар вже має запис.

1. ТЕОРЕТИЧНА ЧАСТИНА

1.1 Аналіз предметної області

1.1.1 Опис предметної області

Інформаційна система відповідає за збереження, захищення та використання даних. А також має API взаємодії користувача з системою. Сама інформаційна база складається з однієї або декількох баз даних.

Предметна область - частина реального світу, що описує інформаційна система в залежності від її призначення.

В даний час усі організації повинні мати доступ до інформації. Це дуже важливо, тому бази даних, які задовольняють потреби зі зберіганням й управлінням даних, мають неабияку цінність, бо допомагають людям в цій справі.

Сфера захисту здоров'я не є виключенням, дослідивши дану сферу я прийшов до висновків, що система має працювати з наступними даними.

1.1.2 Опис вхідних даних

При розробці реляційної бази даних «Лікувального закладу» були виділені наступні вхідні дані:

1. Інформація про лікарів
2. Інформація про спеціальності лікарів
3. Інформація про будівлі закладу
4. Інформація про відділення
5. Інформація про розклад лікарів
6. Інформація про наявне обладнання
7. Інформація про медичні процедури
8. Інформація про пацієнтів
9. Інформація про записи до лікарів
10. Інформація про медичні кабінети
11. Інформація про час та тривалість запису/процедури
12. Місце та причина запису до лікаря

1.1.3 Опис вихідних даних

Вихідні дані - повідомлення і результати, які видаються самою системою. Беруться з постійних даних. Вихідними даними для бази даних є вихідні запити(табл.1-10):

Таблиця 1 – Будівлі

Назва поля	Тип поля	Опис поля
id	SERIAL	Id будівлі
address	VARCHAR	Адреса
name	VARCHAR	Назва будівлі

Таблиця 2 – Відділення

Назва поля	Тип поля	Опис поля
id	SERIAL	Id відділення
building_id	INT	Id будівлі, де відділення знаходиться
name	VARCHAR	Назва відділення

Таблиця 3 – Спеціалізації

Назва поля	Тип поля	Опис поля
id	SERIAL	Id спеціалізації
name	VARCHAR	Назва спеціалізації

Таблиця 4 – Кабінети

Назва поля	Тип поля	Опис поля
building_id	INT	Id будівлі
number	INT	Номер кабінету

Таблиця 5 – Обладнення

Назва поля	Тип поля	Опис поля
id	SERIAL	Id обладнання
model	VARCHAR	Модель обладнання
department_id	INT	Id департменту

Таблиця 6 – Пацієнти

Назва поля	Тип поля	Опис поля
id	SERIAL	Id пацієнта
first_name	VARCHAR	Ім'я
last_name	VARCHAR	Фамілія
age	INT	Вік пацієнта
sex	VARCHAR	Пол пацієнта

Таблиця 7 – Медичні Процедури

Назва поля	Тип поля	Опис поля
id	SERIAL	Id процедури
quipment_id	INT	Id обладнання
time	TIMESPAMP	Час процедури
duration_minutes	INT	Тривалість процедури
patient_id	INT	Id пацієнта

Таблиця 8 – Розклади

Назва поля	Тип поля	Опис поля
id	SERIAL	Id розкладу
monday_timetable	VARCHAR	Часи роботи у понеділок
tuesday_timetable	VARCHAR	Часи роботи у вівторок
wednesday_timetable	VARCHAR	Часи роботи у середу
thursday_timetable	VARCHAR	Часи роботи у четверг
friday_timetable	VARCHAR	Часи роботи у п'ятницю

Таблиця 9 – Лікарі

Назва поля	Тип поля	Опис поля
id	SERIAL	Id лікаря
first_name	VARCHAR	Ім'я
last_name	VARCHAR	Фамілія
experience	INT	Кількість років досвіду
specialization_id	INT	Id спеціальності
department_id	INT	Id відділу
timetable_id	INT	Id розкладу

Таблиця 10 – Записи до лікаря

Назва поля	Тип поля	Опис поля
id	SERIAL	Id запису
time	TIMESPAMP	Час запису
reason	VARCHAR	Причина запису
patient_id	INT	Id пацієнта
doctor_id	INT	Id лікаря

building_id	INT	Id будівлі
cabinet_number	INT	Номер кабінету
duration_minutes	INT	Тривалість
is_closed	BOOLEAN	Чи закритий запис

1.2 Проектування бази даних

1.2.1 Інфологічна модель бази даних

Проектувальники інфологічної моделі розглядають як модель сутність - зв'язок”.

Призначення інфографічної моделі полягає у наданні найбільш дійсних способів збору та представлення інформації, що зберігається у базі даних. Ось чому інфологічну модель даних пробують будувати по аналогії з дійсною мовою.

Початковим етап в проектуванні баз даних - це інфологічна модель, яка допомагає в побудові таблиць та зв'язків між ними.

Існують різні способи опису інфологічної моделі, однак на даний момент найбільш популярним з підходів є той, що заснований на діаграмі "сутністьзв'язок". Отже, при створенні інфологічної моделі ми будемо використовувати ER – діаграму. Ця інфологічна модель - це модель предметної області, яка визначає сутності та зв'язки бази даних. Побудова такої моделі є індивідуальною і особливою, тому однієї методики створення інфологічної моделі немає, кожен робить так як хоче, так як саме він бачить цю модель предметної області.

Інфологічна модель – це опис бази даних, яка буде існувати в майбутньому, опис бд поданий за допомогою діаграм, таблиць, дійсної мови. Будувannya цієї

моделі є продовженням аналізу предметної області інфологічна модель є представленням БД з точки зору того, хто розробляє її.

Головними конструктивними елементами інфологічної моделі даних є: сутності, їх атрибутів та типи зв'язків.

1.2.1.1 Опис сутностей

Одним із елементом інфологічної моделі "сутність-зв'язок" є сутності. Сутність - це те, про що накопичується інформація в інформаційній системі і що може бути однозначно унікальне ідентифіковане. При цьому ім'я сутності повинно відображати клас об'єкта або тип об'єкта.

У відповідності з описом предметної області були отримано такі сутності:

“Будівля” — інформація про будівлі

“Кабінет” — інформація про кабінети

“Пацієнт” — інформація про пацієнтів

“Лікар” — інформація про лікарів

“Спеціалізація” — інформація про спеціальності лікарів

“Розклад” — інформація про розклади лікарів

“Обладнання” — інформація про обладнання

“Процедура” — інформація про медичні процедури

“Відділення” — інформація про відділення

“Запис до лікаря” — інформація про записи до лікарів

1.2.1.2 Опис атрибутів

Атрибут - це поіменована характеристика сутності, за допомогою якої моделюється її властивість. Атрибути називають ще інформаційними елементами. У відповідності з описом предметної області були виділені наступні атрибути у кожній сутності:

Лікарі

id
first_name
last_name
Experience
specialization_id
department_id
timetable_id

Пацієнти

id
first_name
last_name
age
Sex

Будівлі

id
address
name

Кабінети

building_id
number

Спеціалізації

id
name

Відділення

id
name
building_id

Обладнання

id
model
department_id

Процедури

id
quipment_id
time
duration_minutes
patient_id

Розклади

id
monday-timetable
tuesday-timetable
wednesday-timetable
thursday-timetable
friday-timetable

Записи до лікарів

id
patient_id
doctor_id
building_id
cabinet_number
time
reason
is_closed
duration_minutes

1.2.1.3 Опис зв'язків

Дві сутності можуть пов'язуватися через зв'язок екземплярів однієї сутності з екземплярами іншої сутності.

Основна вимога бази даних - вміти знаходити одну сутність за значеннями інших, для чого необхідно встановити зв'язок між ними.

Так як часто в базах даних створюють більше 50 сутностей, то між цими сутностями може бути дуже багато зв'язків. Складність інфологічної моделей визначається наявністю великої кількості зв'язків.

За допомогою зв'язків показують відношення між сутністю та атрибутами.

Зв'язкам можна надати імена для зручності. Термін зв'язок використовується для позначення типу зв'язку і може відображати одноразовість чи багаторазовість.

За направленістю розрізняють такі зв'язки:

1) Однонаправлений зв'язок поділяється на:

- о однозначний, якщо один екземпляр однієї сутності відповідає точно одному екземпляру іншої сутності, зворотного зв'язку не надається;
- о багатозначний, якщо екземпляр сутності відповідає декільком екземплярам іншої сутності, зворотного зв'язку теж не надається.

2) Двонаправлений зв'язок поділяється на:

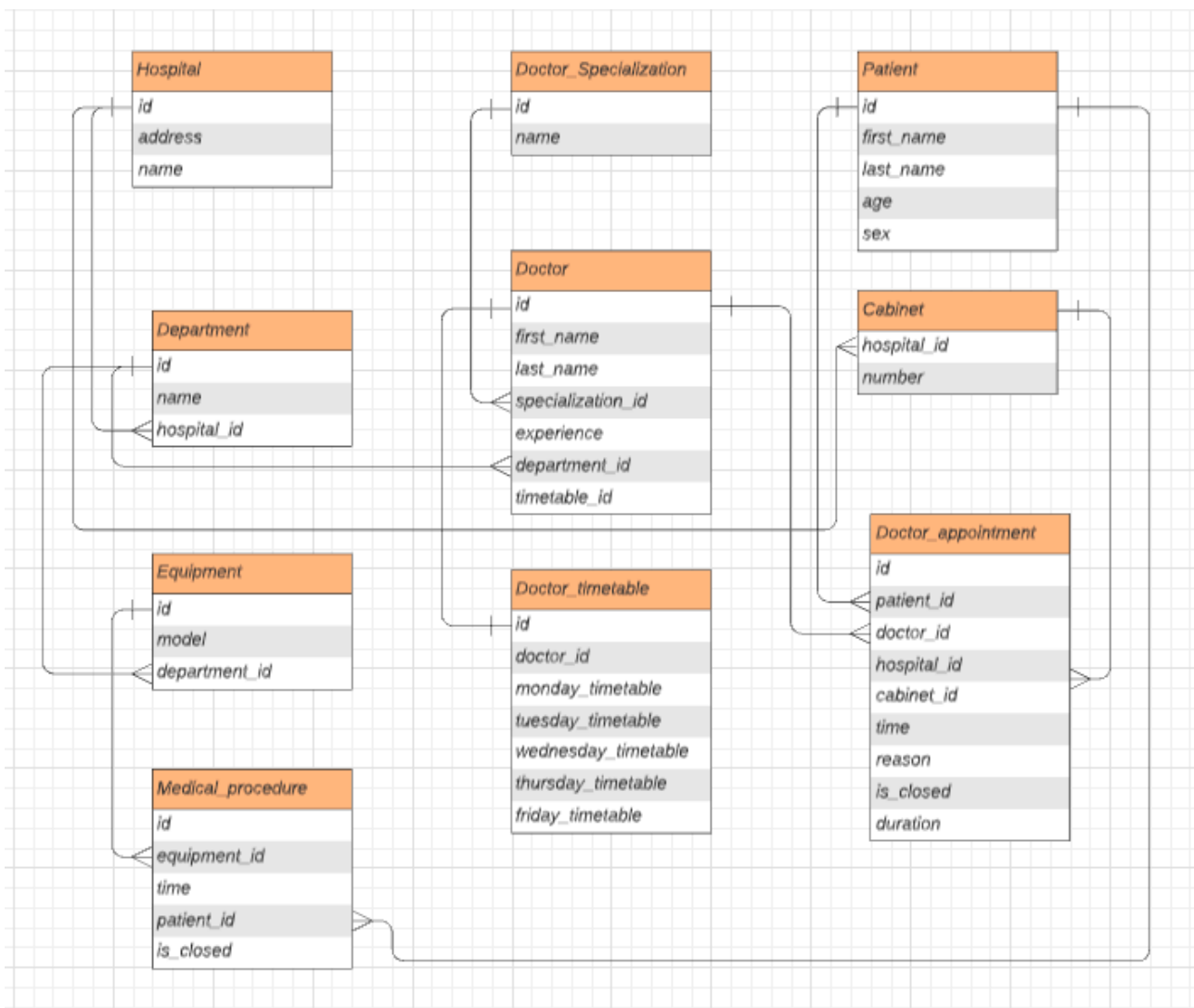
однозначний "один до одного" (1:1) одному екземпляру сутності однієї відповідає лише одному екземпляру другої сутності (навпаки теж працює);

двозначні: "один до багатьох" (1: N) один екземпляр сутності відповідає декільком екземплярам іншої сутності, а кілька екземплярів іншої сутності відповідають лише одному екземпляру сутності "багато до багатьох" (M: N) певна кількість екземплярів однієї сутності відповідає певній кількості екземплярів іншої сутності (навпаки теж працює).

В курсовій роботі були використані наступні типи зв'язків між таблицями:

Номер зв'язку	Головна таблиця	Дочірня таблиця	Тип зв'язку
1	department	building	1:M
2	equipment	department	1:M
3	medical_procedure	equipment	1:M
4	medical_procedure	patient	1:M
5	doctor	doctor_specialization	1:M
6	doctor	department	1:M
7	doctor	timetable	1:M
8	doctor_appointment	patient	1:M
9	doctor_appointment	doctor	1:M
10	doctor_appointment	cabinet	1:M

1.2.1.4 ER-діаграма



На рисунку представлена ER-діаграма бази даних, на якій відображені всі сутності, їх атрибути та зв'язки між сутностями:

1.2.2 Нормалізація таблиць при проектуванні бази даних

Нормалізація – це формальна процедура, у ході якої атрибути даних групуються в таблиці, а таблиці групуються в базу даних.

Нормалізація бази даних – це одна з важливих вимог БД. Нормалізація потрібна для усунення надмірності даних, тобто видалення повторення даних в різних рядках однієї таблиці.

Перший крок до нормалізації бази даних - це позбавлення від повторень. Другий крок до нормалізації бази даних – усі поля повинні бути неділимими. Тобто треба розділити усі поля, які можна, щоб в подальшому можна було легше шукати потрібну інформацію і працювати з БД.

Третій крок до нормалізації БД – не повинно бути полів, які можна знайти за допомогою інших полів.

Четвертий крок до нормалізації БД – не потрібні поля в БД, які позначають різні види одного і того ж.

Кожна наступна нормальна форма має кращі властивості, ніж попередня форма. У теорії реляційних БД виділяють п'ять нормальних форми і нормальну форму Бойса-Кодда. Ми будемо розглядати тільки три нормальні форми.

§ Перша нормальна форма вимагає, щоб кожне поле таблиці бази даних було неділимим і не містило повторень.

§ Друга нормальна форма вимагає, щоб всі атрибути таблиці залежали від первинного ключа.

§ Третя нормальна форма вимагає, щоб в таблиці не малося транзитивних залежностей між атрибутами таблиці.

Вигляд після нормалізації таблиць:

patient	
id	integer
cabinet_number	integer
building_id	integer
first_name	varchar(255)
last_name	varchar(255)
age	integer
sex	varchar(10)

doctor_appointment	
id	integer
patient_id	integer
doctor_id	integer
building_id	integer
cabinet_number	integer
time	timestamp
reason	text
is_closed	boolean
duration_minutes	integer

equipment	
id	integer
model	varchar(255)
department_id	integer

department	
id	integer
name	varchar(255)
building_id	integer

medical_procedure	
id	integer
equipment_id	integer
time	timestamp
duration_minutes	integer
patient_id	integer

doctor	
id	integer
first_name	varchar(255)
last_name	varchar(255)
experience	integer
specialization_id	integer
department_id	integer
timetable_id	integer

doctor_timetable	
id	integer
monday_timetable	varchar(255)
tuesday_timetable	varchar(255)
wednesday_timetable	varchar(255)
thursday_timetable	varchar(255)
friday_timetable	varchar(255)

cabinet	
building_id	integer
number	integer

doctor_specialization	
id	integer
name	varchar(255)

building	
id	integer
address	varchar(255)
name	varchar(255)

2 ПРАКТИЧНА ЧАСТИНА

Весь код може бути знайдений тут: <https://github.com/kinfi4/Hospital-database-system>

2.1 Створення бази даних за допомогою *DataGrip*

2.1.1 Створення бази даних:

```
kinfi4@kinfi4-HP-EliteBook-850-G5 ~-> sudo -u postgres psql
[sudo] password for kinfi4:
psql (14.0 (Ubuntu 14.0-1.pgdg20.04+1))
Type "help" for help.

postgres=# CREATE DATABASE hospital;
```

Також створимо користувачів для роботи з нашою БД.

Перший адмін, має всі права на БД

```
CREATE USER hospital_admin WITH ENCRYPTED PASSWORD 'admin-password';
GRANT ALL PRIVILEGES ON DATABASE hospital TO hospital_admin;
```

Користувач пацієнту має права на створення запису до лікаря, та перегляду своїх записів, процедур, розкладу лікарів

```
CREATE USER hospital_patient WITH ENCRYPTED PASSWORD 'patient-password';
GRANT INSERT ON TABLE doctor_appointment TO hospital_patient;
GRANT SELECT ON TABLE medical_procedure, doctor_appointment, doctor_timetable TO hospital_patient;
```

Користувач лікаря може створювати, переглядати будь-які записи, та редагувати їх, та переглядати розклад.

```
CREATE USER hospital_doctor WITH ENCRYPTED PASSWORD 'doctor-password';
GRANT INSERT, SELECT, UPDATE ON TABLE patient, doctor_appointment, medical_procedure TO hospital_doctor;
```

```
GRANT SELECT ON TABLE doctor_timetable TO hospital_doctor;
```

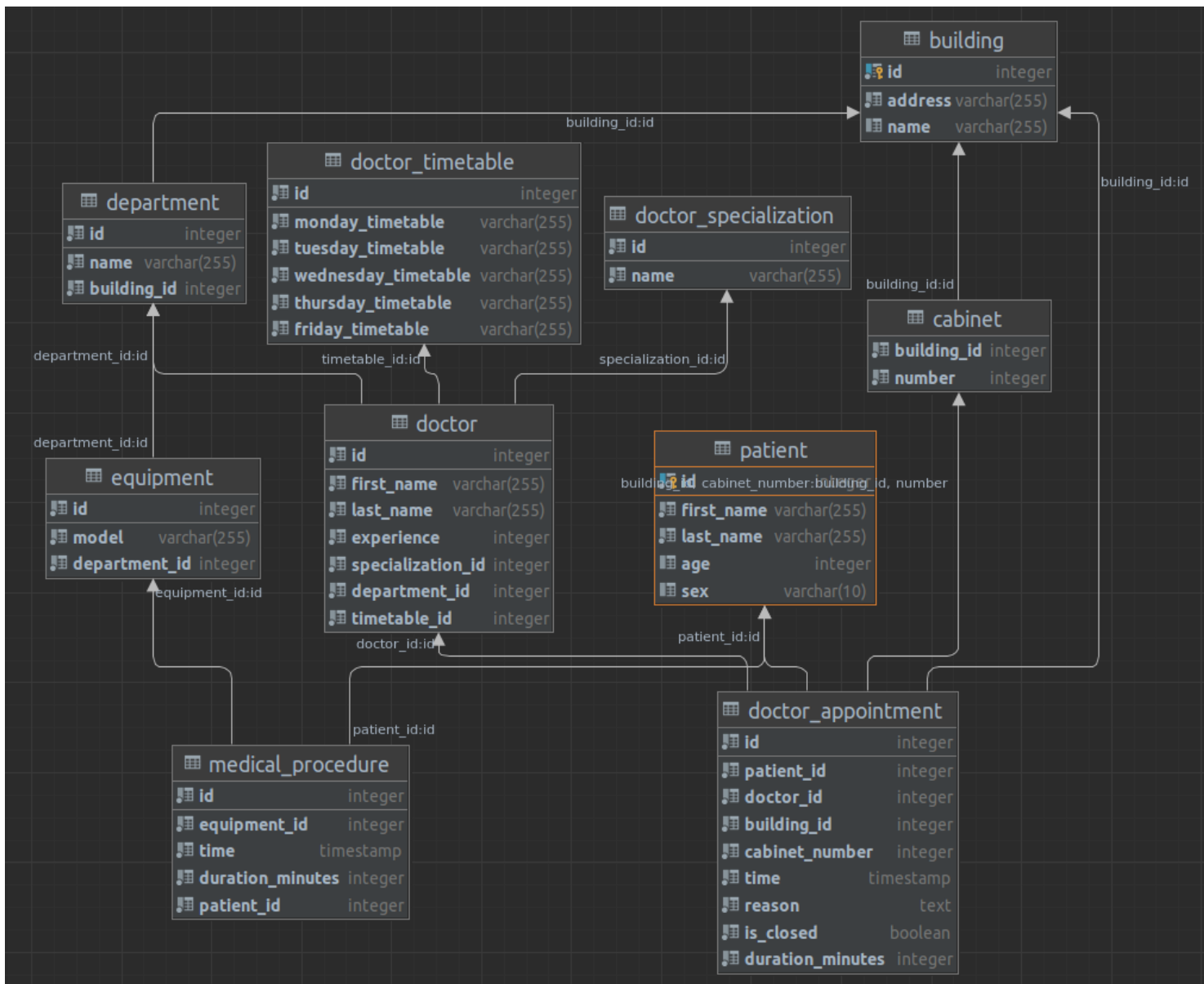
2.1.2 Створення таблиць бази даних

```
CREATE TABLE IF NOT EXISTS Building (  
    id SERIAL PRIMARY KEY,  
    address VARCHAR(255) NOT NULL,  
    name VARCHAR(255) NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Department (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(255) NOT NULL,  
    building_id INT NOT NULL,  
  
    CONSTRAINT fk_on_building  
        FOREIGN KEY (building_id) REFERENCES Building (id) ON DELETE CASCADE ON UPDATE CASCADE  
);  
  
CREATE TABLE IF NOT EXISTS Equipment (  
    id SERIAL PRIMARY KEY,  
    model VARCHAR(255) NOT NULL,  
    department_id INT NULL,  
  
    CONSTRAINT fk_on_department  
        FOREIGN KEY (department_id) REFERENCES department (id) ON DELETE CASCADE ON UPDATE CASCADE  
);  
  
CREATE TABLE IF NOT EXISTS Doctor_specialization (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(255) NOT NULL  
);
```

Інші таблиці представлено в додатках або ось <https://github.com/kinfi4/Hospital-database-system/blob/master/source/create-tables.sql>.

2.1.3 Створення діаграми бази даних

У розділі діаграм в утиліті *DataGrip* створюємо нову діаграму, в яку додаємо зі списку одинадцять таблиць із даної предметної області та перевіряємо зв'язки між таблицями:



2.1.4 Заповнення таблиць бази даних даними

Дані для заповнення таблиць можна знайти тут:

<https://github.com/kinfi4/Hospital-database-system/tree/master/source/data>

2.1.5 Створення тригерів

В рамках роботи було створення три тригери: перевірка чи запис до лікаря попадає у розклад даного лікаря, перевірка, що запис до лікаря можливий (кабінет для запису вільний, у лікаря нема іншого запису на цей час) та аналогічна перевірка можливості проведення медичної процедури.

Код тригерів можна переглянути тут: <https://github.com/kinfi4/Hospital-database-system/tree/master/source/triggers> або в додатках.

```
hospital@system:public> INSERT INTO doctor_appointment (id, patient_id, doctor_id, building_id, cabinet_number, time, reason, is_closed, duration_minutes) VALUES
(100, 1, 35, 1, 15, '2021-12-28 06:09:42.747000', 'some reason', false, 23)
[2021-12-22 15:01:05] [P0001] ERROR: Patient: 1 cant have an appointment with doctor: 35 at 2021-12-28 06:09:42.747 because doctor is not working at that time.s
[2021-12-22 15:01:05] Where: PL/pgSQL function raise_error(character varying) line 3 at RAISE
[2021-12-22 15:01:05] SQL statement "SELECT raise_error(error_message)"
[2021-12-22 15:01:05] PL/pgSQL function check_appointment_time_func() line 37 at SQL statement
```

```
hospital@system:public> INSERT INTO doctor_appointment (id, patient_id, doctor_id, building_id, cabinet_number, time, reason, is_closed, duration_minutes) VALUES
(100, 1, 7, 3, 149, '2021-11-30 15:18:51.593000', 'some reason', false, 23)
[2021-12-22 15:02:36] [P0001] ERROR: Patient: 1 cant have an appointment with doctor: 7 at 2021-11-30 15:18:51.593 because cabinet is busy at that moment or doctor or patient is having another
appointment at that times
[2021-12-22 15:02:36] Where: PL/pgSQL function raise_error(character varying) line 3 at RAISE
[2021-12-22 15:02:36] SQL statement "SELECT raise_error(error_message)"
[2021-12-22 15:02:36] PL/pgSQL function check_uniques_of_appointment_func() line 25 at SQL statement
```

2.1.6 Створення представлень

Маємо два представлення.

Представлення `get_full_information_about_appointments` показує повну інформацію про записи до лікарів.

	patient_name	doctor_name	building_name	cabinet_number	reason	appointment_time	duration_minutes
1	Dulce Beniamino	Brana Brittani	National Hospital of Chernigiv	15	Lorem Ipsum is simply dummy text of the printing and typesetting industry. ...	2021-12-28 06:09:42.747000	35
2	Gabi Burch	Tobe Jagir	Building named by Makarov	149	the industry's standard dummy text ever since the 1500s, when an unknown	2021-11-30 15:18:51.593000	10
3	Helena Dorine	Jasmina Hermes	Building named by Makarov	169	Contrary to popular belief, Lorem Ipsum is not simply random	2021-11-27 10:48:45.495000	25
4	Sybilie Lacombe	Dulcinea Burkle	Kievan building 1	233	a piece of classical Latin literature from 45 BC, making it over 2000 years...	2022-01-25 10:27:49.203000	25
5	Kore Guthrie	Mariela Catie	National Hospital of Chernigiv	14	more obscure Latin words, consectetur, from a Lorem Ipsum passage, and goi...	2021-12-08 12:13:43.481000	15
6	Sheelagh Regan	Averyl Alice	National Cancer Building	36	There are many variations of passages of Lorem Ipsum available, but the maj...	2021-12-25 19:16:34.088000	15
7	Tamgrah Presber	Florencia Helve	Kievan building 1	167	through the cites of the word in classical literature, discovered	2022-01-03 02:55:01.033000	10
8	Robbi Masao	Sadie Elbertina	Building named by Makarov	241	going to use a passage of Lorem Ipsum, you need to be sure there isn't	2021-12-12 07:08:09.295000	15
9	Adore Azeria	Belinda Hertzfeld	National Cancer Building	345	anything embarrassing hidden in the middle of text. All the Lorem Ipsum	2021-12-02 05:34:17.283000	20
10	Christal Graig	Tobe Jagir	National Hospital of Chernigiv	121	generators on the Internet tend to repeat predefined chunks as necessary	2022-01-15 12:51:04.883000	20
11	Cissiee Sinogold	Olwen Francyne	Kievan building 1	167	Nunc aliquet magna vel velit rhoncus tincidunt.	2021-12-17 13:21:40.978000	25
12	Hildegard Zetta	Evaleen Emanuel	National Cancer Building	345	Lorem ipsum dolor sit amet, consectetur adipiscing elit	2021-12-15 17:21:11.346000	30
13	Shannah Astra	Roz Ivmann	Kievan building 1	167	Curabitur imperdiet velit ut urna facilisis dionissim	2022-01-09 20:27:35.473000	20

Представлення `get_full_information_about_medical_procedures` показує повну інформацію про процедури.

	patient_full_name	model	time	duration_minutes
1	Dyann Philipp	Ингалятор	2021-12-23 23:10:13.090000	20
2	Corene Wind	Дефибрилятор	2022-01-05 00:54:02.545000	30
3	Flory Ailyn	Дефибрилятор	2022-01-08 13:36:02.240000	20
4	Donnie Dulciana	Дефибрилятор	2022-01-10 21:42:13.708000	25
5	Gabi Burch	Томограф	2022-01-29 14:25:56.988000	15
6	Evita Craggie	Томограф	2021-12-22 13:02:19.575000	35
7	Gui Jacobah	Томограф	2022-01-26 07:25:02.308000	25
8	Shannah Astra	Томограф	2022-01-22 10:31:32.812000	15
9	Moyna Lanita	Томограф	2022-01-30 20:49:06.095000	10

2.1.7 Створення функцій

В рамках роботи я створив 8 функцій, код яких можна знайти тут: <https://github.com/kinfi4/Hospital-database-system/tree/master/source/functions> або в додатках, тут я зачеплю лише кілька найцікавіших.

- *get_all_available_doctors_at_specific_time(for_time TIMESPAMP)* — повертає список всіх лікарів, що вільні в указаний час від записів, та цей час співпадає з їх розкладом
- *get_free_cabinets(building_id INT, for_time TIMESTAMP, duration_minutes INT)* — повертає список вільних кабінетів
- *get_doctor_timetable(doc_id INT, day_of_week INT)* — повертає розклад лікаря для даного дня
- *create_appointment(doctor_id INT, patient_id INT, appointment_time TIMESTAMP, appointment_reason TEXT, appointment_duration_minutes INT)* — знаходить вільних кабінет і створює запис до лікаря
- *close_appointment(appointment_id INT)* — закриває запис

2.2.1 DML запити

В рамках роботи було розроблено 20 DML запитів, код яких можна знайти ось тут: <https://github.com/kinfi4/Hospital-database-system/tree/master/source/queries> або в додатку.

Загалом, моя порада вам, переглядати запити на github, це якось зручніше на мою думку, там є і опис запиту і сам запит, тут просто наведу вивід.

РЕЗУЛЬТАТИ:

count-specializations.sql

Кількість лікарів для кожної наявної спеціальності

	specialization_name	number_of_specialists
1	Педиатр	10
2	Медсестра	9
3	Онколог	8
4	Дерматолог	8
5	Семейный врач	7
6	Терапевт	6
7	Хирург	2

count-specializations-departments.sql

Кількість лікарів для кожного відділення, кожної спеціальності

	department_name	specialization_name	number_of_doctors
1	Общее	Педиатр	4
2	Травматология	Медсестра	4
3	Общее	Семейный врач	4
4	Общее	Терапевт	3
5	Онкологическое	Дерматолог	3
6	Интенсивной терапии	Дерматолог	3
7	Травматология	Онколог	2
8	Онкологическое	Онколог	2
9	Интенсивной терапии	Педиатр	2
10	Педиатрия	Педиатр	2

doctor-appointments-ordered-by-number-of-appointments.sql

Лікар, його спеціальність, та кількість записів до цього лікаря, у відсортованому по кількості записів вигляді.

	id	full_name	specialization_name	appointment_time	cabinet_number	number_of_appointments
1	25	Olwen Francyne	Семейный врач	2022-01-05 07:22:43.258000	121	2
2	25	Olwen Francyne	Семейный врач	2021-12-17 13:21:40.978000	167	2
3	18	Roz Lyman	Медсестра	2022-01-09 20:27:35.473000	167	2
4	18	Roz Lyman	Медсестра	2022-01-20 17:50:31.927000	83	2
5	7	Tobe Jagir	Семейный врач	2022-01-15 12:51:04.883000	121	2
6	7	Tobe Jagir	Семейный врач	2021-11-30 15:18:51.593000	149	2
7	45	Ardys Callista	Медсестра	2021-12-19 17:04:34.439000	161	1
8	43	Averyl Alice	Дерматолог	2021-12-25 19:16:34.088000	36	1
9	21	Belinda Hertzfeld	Семейный врач	2021-12-02 05:34:17.283000	345	1
10	35	Braná Brittani	Педиатр	2021-12-28 06:09:42.747000	15	1
11	37	Caressa Bevin	Хирург	2022-01-10 22:43:44.979000	148	1
12	44	Dulcinea Burkle	Медсестра	2022-01-25 10:27:49.203000	233	1
13	26	EvaLeen Emanuel	Педиатр	2021-12-15 17:21:11.346000	345	1
14	46	Florencia Helve	Хирург	2022-01-03 02:55:01.033000	167	1

doctor-appointments-ordered-by-number-windows-func.sql

Аналогічний запит, але вже з використанням WINDOWS FUNCTIONS

	id	full_name	appointment_time	cabinet_number	number_of_appointments
1	25	Olwen Francyne	2021-12-17 13:21:40.978000	167	2
2	25	Olwen Francyne	2022-01-05 07:22:43.258000	121	2
3	18	Roz Lymann	2022-01-20 17:50:31.927000	83	2
4	18	Roz Lymann	2022-01-09 20:27:35.473000	167	2
5	7	Tobe Jagir	2022-01-15 12:51:04.883000	121	2
6	7	Tobe Jagir	2021-11-30 15:18:51.593000	149	2
7	45	Ardys Callista	2021-12-19 17:04:34.439000	161	1
8	43	Averyl Alice	2021-12-25 19:16:34.088000	36	1
9	21	Belinda Hertzfeld	2021-12-02 05:34:17.283000	345	1
10	35	Brana Brittani	2021-12-28 06:09:42.747000	15	1
11	37	Caressa Bevin	2022-01-10 22:43:44.979000	148	1
12	44	Dulcinea Burkle	2022-01-25 10:27:49.203000	233	1
13	26	Evaleen Emanuel	2021-12-15 17:21:11.346000	345	1

get-all-available-doctors.sql

Повертає всіх вільних лікарів для 2022-01-25 10:27

	full_name	doctor_timetable	name
1	Cam Palocz	08.00:15.15	Онколог
2	Tobe Jagir	10.20:17.45	Семейный врач
3	Marcy Ingra	09.30:18.00	Педиатр
4	Olivette Dalli	10.20:17.45	Дерматолог
5	Gilda Read	10.20:17.45	Медсестра
6	Brynna Faust	08.00:15.15	Онколог
7	Heida Barrus	10.20:17.45	Онколог
8	Ardenia Mandler	09.30:18.00	Медсестра
9	Brana Brittani	10.20:17.45	Педиатр
10	Mary Stuart	09.30:18.00	Педиатр

get-all-surgeon-with-appointments.sql

Знайти всі записи до хірургів

	first_name	name	time	reason
1	Florencia	Хирург	2022-01-03 02:55:01.033000	through the cites of the word in classical literature, discovered
2	Caressa	Хирург	2022-01-10 22:43:44.979000	Nunc tristique lectus in ullamcorper lacinia

get-avg-patient-age-by-specialist.sql

Середній вік пацієнтів для кожної спеціальності

	specialization_name	number_of_appointments	average_age
1	Семейный врач	9	39.22
2	Педиатр	2	34.5
3	Хирург	2	34
4	Медсестра	6	45.17
5	Дерматолог	1	34

get-buildings-departments.sql

Отримання всіх відділень, для кожної будівлі лікарні

	building_name	address	department_name
1	Building named by Makarov	Kiev: Victory st 12	Общее
2	Building named by Makarov	Kiev: Victory st 12	Педиатрия
3	Building named by Makarov	Kiev: Victory st 12	Травматология
4	Kievan building 1	Kiev: Shevchenko st 5	Интенсивной терапии
5	Kievan building 1	Kiev: Shevchenko st 5	Общее
6	Kievan building 1	Kiev: Shevchenko st 5	Педиатрия
7	National Cancer Building	Lviv: Alley of Bandera 12	Онкологическое
8	National Hostital of Chernigiv	Chernigiv: Main st. 142	Интенсивной терапии
9	National Hostital of Chernigiv	Chernigiv: Main st. 142	Общее
10	National Hostital of Chernigiv	Chernigiv: Main st. 142	Онкологическое
11	National Hostital of Chernigiv	Chernigiv: Main st. 142	Травматология

get-difference-between-procedure-and-appointments-per-department.sql

Отримати різницю між кількістю процедур і записів до лікарів, для кожного відділення.

	h.name	department.name	difference_between_procedures_and_appointments
1	National Hostital of Chernigiv	Общее	-4
2	Building named by Makarov	Педиатрия	-3
3	Building named by Makarov	Травматология	-1
4	Kievan building 1	Педиатрия	-1
5	Kievan building 1	Общее	1
6	Building named by Makarov	Общее	1
7	National Cancer Building	Онкологическое	3
8	National Hostital of Chernigiv	Онкологическое	4
9	Kievan building 1	Интенсивной терапии	5

get-doctor-appointments.sql

Записи до лікаря Mariele Catie

doctor_name	patient_name	appointment_time	reason	duration_minutes
1 Mariele	Kore	2021-12-08 12:13:43.481000	more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going	15

get-doctor-ranks-by-hospital.sql

Отримати ранги лікарів, за кількістю записів до них, для кожної будівлі

	building_name	full_name	rank	number_of_appointments
1	Building named by Makarov	Roz Lymann	1	2
2	Building named by Makarov	Sadie Elbertina	2	1
3	Building named by Makarov	Jasmina Hermes	2	1
4	Building named by Makarov	Evaleen Emanuel	2	1
5	Building named by Makarov	Belinda Hertzfeld	2	1
6	Building named by Makarov	Lorie Orpah	2	1
7	Building named by Makarov	Caressa Bevin	2	1
8	Building named by Makarov	Dulcinea Burkle	2	1
9	Kievan building 1	Olwen Francyne	1	2
10	Kievan building 1	Mariele Catie	2	1
11	Kievan building 1	Arden Gallardo	3	1

get-doctor-timetable.sql

Отримати розклад лікаря **Ada Rossner** на сьогодні, я це запускав у субботу, тому його тут нема.

```
get_doctor_timetable
1 No timetable
```

get-doctors-list-timetable-available.sql

Знайти всіх вільних лікарів на 2022-04-05T08:15:43 у **National Hostital of Chernigiv**

	full_name	doctor_timetable	name	address
1	Cam Palocz	08.00:15.15	Онкологическое	Chernigiv: Main st. 142
2	Brynna Faust	08.00:15.15	Травматология	Chernigiv: Main st. 142
3	Raf Dituri	08.00:15.15	Интенсивной терапии	Chernigiv: Main st. 142
4	Melisent Even	08.00:15.15	Общее	Chernigiv: Main st. 142

get-doctors-todays-timetables.sql

Отримати розклад всіх лікарів на сьогодні

	full_name	get_doctor_timetable
1	Cam Palocz	no timetable, its weekend
2	Catrina Shanley	no timetable, its weekend
3	Ada Rossner	no timetable, its weekend
4	Giustina Travax	no timetable, its weekend
5	Olwen Claudine	no timetable, its weekend
6	Josephine Zina	no timetable, its weekend
7	Tobe Jagir	no timetable, its weekend
8	Roslyn Belanger	no timetable, its weekend
9	Sadie Elbertina	no timetable, its weekend
10	Marcy Ingra	no timetable, its weekend

get-equipments-procedures.sql

Отримати процедури для відділення Інтенсивной терапии та обладнання Ингалятор

	model	department_name	patinet_first_name	patient_last_name	starting_time	duration_minutes
1	Ингалятор	Интенсивной терапии	Dyann	Philipp	2021-12-23 23:10	20
2	Ингалятор	Интенсивной терапии	Elena	Robertson	2021-12-01 19:09	25
3	Ингалятор	Интенсивной терапии	Aeriela	Kaete	2021-11-30 02:16	20

get-free-cabinets.sql

Знайти всі вільні кабінети на 2022-01-25 10:27 для building named by Makarov

	free_cabinet
1	83
2	148
3	149
4	161
5	169
6	241

get-hospital-doctors.sql

Знайти лікарів для National Hostital of Chernigiv

	building.name	department.name	first_name	last_name	experience
1	National Hostital of Chernigiv	Онкологическое	Ardenia	Mandler	9
2	National Hostital of Chernigiv	Общее	Brana	Brittani	1
3	National Hostital of Chernigiv	Травматология	Brynna	Faust	19
4	National Hostital of Chernigiv	Онкологическое	Cam	Palocz	19
5	National Hostital of Chernigiv	Онкологическое	Florencia	Helve	1
6	National Hostital of Chernigiv	Онкологическое	Gilda	Read	1
7	National Hostital of Chernigiv	Общее	Harrietta	Voletta	9
8	National Hostital of Chernigiv	Интенсивной терапии	Heida	Barrus	11
9	National Hostital of Chernigiv	Онкологическое	Marcy	Ingra	2
10	National Hostital of Chernigiv	Травматология	Mary	Stuart	4
11	National Hostital of Chernigiv	Общее	Melisent	Even	14
12	National Hostital of Chernigiv	Интенсивной терапии	Olivette	Dalli	15

get-patient-appointments.sql

Знайти інформацію про запис пацієнта Christal Graig

	doctor_name	patient_name	appointment_time
1	Tobe	Christal	2022-01-15 12:51:04.883000

get-patient-procedures.sql

Знайти інформацію про процедури, що будуть у майбутньому

	full_patient_name	procedure_time	duration_minutes	equipment_model	department_name
1	Moyna Lanita	2022-01-30 20:49:06.095000	10	Томограф	Онкологическое
2	Gabi Burch	2022-01-29 14:25:56.988000	15	Томограф	Онкологическое
3	Johna Iphlgenia	2022-01-24 19:05:33.500000	30	Стетоскоп	Общее
4	Clary Philoo	2022-01-12 19:35:16.252000	15	Ортопедический стол	Травматология
5	Donnie Dulciana	2022-01-10 21:42:13.708000	25	Дефибриллятор	Интенсивной терапии
6	Flory Ailyn	2022-01-08 13:36:02.240000	20	Дефибриллятор	Интенсивной терапии
7	Tamqrah Presber	2021-12-31 23:19:59.462000	35	Стетоскоп	Общее
8	Dyann Philipp	2021-12-23 23:10:13.090000	20	Ингалятор	Интенсивной терапии
9	Evita Craggie	2021-12-22 13:02:19.575000	35	Томограф	Онкологическое
10	Lovella Larochele	2021-12-17 12:12:42.381000	20	Дефибриллятор	Интенсивной терапии
11	Tracey Jotham	2021-12-02 14:56:21.766000	15	Аппарат для интраоперационной реинфузии крови	Травматология
12	Elena Robertson	2021-12-01 19:09:24.011000	25	Ингалятор	Интенсивной терапии
13	Rochette Adrienne	2021-11-26 19:41:03.152000	30	Ортопедический стол	Травматология

sum-departments-experience.sql

Знайти суммарний досвід лікарів для кожної будівлі, для кожного відділення

	building_name	department_name	summaric_experience
1	Building named by Makarov	Травматология	83
2	Kievan building 1	Интенсивной терапии	65
3	Building named by Makarov	Общее	52
4	National Cancer Building	Онкологическое	49
5	National Hostital of Chernigiv	Онкологическое	41
6	Building named by Makarov	Педиатрия	38
7	Kievan building 1	Общее	35
8	National Hostital of Chernigiv	Интенсивной терапии	30
9	Kievan building 1	Педиатрия	30
10	National Hostital of Chernigiv	Общее	26
11	National Hostital of Chernigiv	Травматология	23

2.2.2 Результати оптимізації

Взагалі людство винайшло чимало технік оптимізації для СУБД, тут і partitioning, clustering, index, sharding ітд, взагалі, якщо ви дійсно читаете це, можете мене подушити цим на захисті. Окремо скажу, що всі техніки оптимізації просто змушують нашу СУБД тримати дані у той чи іншій формі, що значно прискорює доступ к цим даним. Як працюють індекси я тут пояснювати детально не буду, можу усно, просто скажу, що індекс — це механізм СУБД, що змушує її зберігати дані указаної колонки у вигляді оптимізованої структури даних, як btree, hash table ітд.

Код створення індексів.

```
CREATE INDEX ON doctor USING btree (first_name);
CREATE INDEX ON doctor_appointment USING btree (time DESC);
CREATE INDEX ON equipment USING btree (model);
CREATE INDEX ON doctor_specialization USING btree (name);
CREATE INDEX ON building USING btree (name);
```

```
hospitalsystem.public> SELECT
    doc.first_name,
    ds.name,
    da.time,
    da.reason
FROM doctor doc
JOIN doctor_specialization ds ON specialization_id = ds.id
JOIN doctor_appointment da ON doc.id = doctor_id
WHERE ds.name = 'Хирург'
[2021-12-25 10:56:40] 2 rows retrieved starting from 1 in 157 ms (execution: 9 ms, fetching: 148 ms)
hospitalsystem.public> SELECT
    doc.first_name,
    ds.name,
    da.time,
    da.reason
FROM doctor doc
JOIN doctor_specialization ds ON specialization_id = ds.id
JOIN doctor_appointment da ON doc.id = doctor_id
WHERE ds.name = 'Хирург'
[2021-12-25 10:56:58] 2 rows retrieved starting from 1 in 38 ms (execution: 6 ms, fetching: 32 ms)
```

```

[2021-12-25 11:01:37] 4 rows retrieved starting from 1 in 90 ms (execution: 21 ms, fetching: 69 ms)
hospitalssystem.public> SELECT
    doctor.first_name || ' ' || doctor.last_name AS full_name,
    doctor_timetable,
    d.name,
    address
FROM get_all_available_doctors_at_specific_time('2022-04-05T08:15:43.258Z'::TIMESTAMP)
JOIN doctor ON doctor.id = doctor_id
JOIN department d ON department_id = d.id
JOIN building b ON building_id = b.id
WHERE b.name = 'National Hospital of Chernigiv'
[2021-12-25 11:02:12] 4 rows retrieved starting from 1 in 42 ms (execution: 7 ms, fetching: 35 ms)

```

```

[2021-12-21 21:24:02] Connected
hospitalssystem.public> SELECT
    *
FROM equipment
WHERE model = 'hello world'
[2021-12-21 21:24:02] 0 rows retrieved in 83 ms (execution: 7 ms, fetching: 76 ms)
hospitalssystem.public> SELECT
    *
FROM equipment
WHERE model = 'hello world'
[2021-12-21 21:24:57] 0 rows retrieved in 61 ms (execution: 7 ms, fetching: 54 ms)

```

ВИСНОВКИ

Метою курсової роботи було проектування бази даних лікувального закладу. Для виконання курсової роботи були проведені необхідні дослідження, що стосуються структури системи. Спочатку ми провели повний аналіз нашої предметної області і тоді вже почали її описувати.

Після цього була побудована концептуальна модель для якої ми використовували мову ER-опису предметної області, яка базується на концепції, що інформаційна модель предметної області може бути описана із застосування таких понять, як сутність, атрибут, зв'язок. Після побудови ER діаграми, ми провели нормалізацію таблиць нашої бази даних. І ми створили даталогічну модель бази даних. І тільки після цих завдань ми приступили к практичній частині і почали створювати саму базу даних, її таблиць, потім створили діаграму, щоб перевірити цілісність нашої бази даних і коректність. Далі ми заповняли таблиці бази даних, створювали тригери, представлення, функції та застосовували наші знання щодо вибірки інформації.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

SQL тренажер - <https://stepik.org/lesson/297508/step/1?auth=login&unit=279268>

Quick start with DataGrip - <https://www.jetbrains.com/help/datagrip/quick-start-with-datagrip.html>

Лекційний матеріал з предмету «Бази даних»

Intro to SQL - <https://www.kaggle.com/learn/intro-to-sql>

Advanced SQL - <https://www.kaggle.com/learn/advanced-sql>

SELECT QUERIES:

```
-----  
--- Counting the number of doctors for each  
--- specialization  
-----
```

```
SELECT  
    ds.name AS specializatoin_name,  
    count(*) AS number_of_specialists  
FROM doctor doc  
    JOIN doctor_specialization ds ON doc.specialization_id = ds.id  
GROUP BY ds.name  
ORDER BY number_of_specialists DESC;
```

```
-----  
--- Counting the number of doctors for each  
--- specialization for each department  
-----
```

```
SELECT  
    d.name AS department_name,  
    ds.name AS specialization_name,  
    count(*) AS number_of_doctors  
FROM doctor doc  
    JOIN doctor_specialization ds ON doc.specialization_id = ds.id  
    JOIN department d ON doc.department_id = d.id  
GROUP BY department_name, specialization_name  
ORDER BY number_of_doctors DESC;
```

```
-----  
--- Get all doctor appointments, ordered by  
--- number of appointments of the doctor using WITH statement  
-----
```

```
WITH ordered_doctors(doc_id, number_of_appointments) AS (  
    SELECT doc.id, count(*) FROM doctor doc  
        JOIN doctor_appointment d ON doc.id = d.doctor_id  
    GROUP BY doc.id  
)  
SELECT  
    doc.id,  
    doc.first_name || ' ' || doc.last_name AS full_name,  
    ds.name AS specialization_name,  
    da.time AS appointment_time,  
    da.cabinet_number,  
    number_of_appointments
```



```

FROM doctor doc
  JOIN doctor_appointment da ON doc.id = da.doctor_id
  JOIN ordered_doctors ON ordered_doctors.doc_id = doc.id
  JOIN doctor_specialization ds ON doc.specialization_id = ds.id
ORDER BY number_of_appointments DESC, full_name;

```

```

-----
--- Get all doctor appointments, ordered by
--- number of appointments of the doctor using WINDOWS FUNCTIONSa
-----

```

```

SELECT
  doc.id,
  doc.first_name || ' ' || doc.last_name AS full_name,
  da.time AS appointment_time,
  da.cabinet_number,
  count(*) OVER (PARTITION BY doc.id) AS number_of_appointments
FROM doctor doc
  JOIN doctor_specialization ds ON doc.specialization_id = ds.id
  JOIN doctor_appointment da ON doc.id = da.doctor_id
ORDER BY number_of_appointments DESC, full_name;

```

```

-----
--- Get all available doctors for 2022-01-25 10:27:49.203000
-----

```

```

SELECT
  doctor.first_name || ' ' || doctor.last_name AS full_name,
  doctor_timetable,
  ds.name
FROM get_all_available_doctors_at_specific_time('2022-01-25 10:27:49.203000')
  JOIN doctor ON doctor.id = doctor_id
  JOIN department d ON department_id = d.id
  JOIN building b ON building_id = b.id
  JOIN doctor_specialization ds ON specialization_id = ds.id
WHERE b.name = 'National Hostital of Chernigiv';

```

```

-----
--- Count appointments of every surgery
-----

```

```

SELECT
  doc.first_name,
  ds.name,
  da.time,
  da.reason
FROM doctor doc
  JOIN doctor_specialization ds ON specialization_id = ds.id

```

```

    JOIN doctor_appointment da ON doc.id = doctor_id
WHERE ds.name = 'Хирург'
ORDER BY time;

```

```

-----
--- Get average age of patients by
--- doctors specialization
-----

```

```

SELECT
    ds.name AS specialization_name,
    count(*) AS number_of_appointments,
    round(avg(patient.age), 2) AS average_age
FROM patient
    JOIN doctor_appointment da ON patient.id = patient_id
    JOIN doctor d ON doctor_id = d.id
    JOIN doctor_specialization ds ON specialization_id = ds.id
GROUP BY ds.name

```

```

-----
--- Get departments per buildings
-----

```

```

SELECT
    building.name AS building_name,
    building.address,
    d.name AS department_name
FROM building
    JOIN department d ON building.id = building_id
ORDER BY building.name, address, department_name;

```

```

-----
--- Counting difference between procedures
--- abd appointments
-----

```

```

WITH count_departments_procedures (department_id, number_of_procedures) AS (
    SELECT
        department.id,
        count(DISTINCT mp.id) AS number_of_procedures
    FROM department
        LEFT JOIN equipment e ON department.id = department_id
        LEFT JOIN medical_procedure mp ON e.id = equipment_id
    GROUP BY department.id
), count_department_appointmets (department_id, number_of_appointmets) AS (
    SELECT

```

```

        department.id,
        count(DISTINCT da.id) AS number_of_procedures
    FROM department
        JOIN doctor d ON department.id = department_id
        JOIN doctor_appointment da ON d.id = doctor_id
    GROUP BY department.id
)
SELECT
    h.name,
    department.name,
    number_of_procedures - number_of_appointments AS
difference_between_procedures_and_appointments
FROM department
    JOIN count_departments_procedures ON count_departments_procedures.department_id =
department.id
    JOIN count_department_appointments ON count_department_appointments.department_id =
department.id
    JOIN building h ON building_id = h.id
ORDER BY difference_between_procedures_and_appointments;

```

```

-----
--- Get appointment details for specific doctor
--- named Mariele Catie
-----

```

```

SELECT
    doctor.first_name AS doctor_name,
    p.first_name AS patient_name,
    da.time AS appointment_time,
    da.reason,
    da.duration_minutes
FROM doctor
    JOIN doctor_appointment da on doctor.id = da.doctor_id
    JOIN patient p on da.patient_id = p.id
WHERE doctor.first_name = 'Mariele' AND doctor.last_name = 'Catie'
ORDER BY doctor_name, patient_name;

```

```

-----
--- Get doctor ranks by number of appointments
--- by building
-----

```

```

WITH count_department_appointments (doctor_id, number_of_appointments) AS (
    SELECT
        doc.id,
        count(DISTINCT da.id) AS number_of_procedures
    FROM doctor doc
        JOIN doctor_appointment da ON doc.id = doctor_id

```

```

        GROUP BY doc.id
    )
SELECT
    h.name AS building_name,
    doc.first_name || ' ' || doc.last_name AS full_name,
    rank() OVER (PARTITION BY h.name ORDER BY cda.number_of_appointments DESC),
    cda.number_of_appointments
FROM doctor doc
    JOIN department d ON department_id = d.id
    JOIN building h ON building_id = h.id
    JOIN count_department_appointments cda ON cda.doctor_id = doc.id
ORDER BY building_name, number_of_appointments DESC;

```

```

-----
--- Get doctor with id = 1 timetable
--- for tuesday
-----

```

```

SELECT
    *
FROM get_doctor_timetable((SELECT id FROM doctor WHERE first_name = 'Ada' AND last_name
= 'Rossner'),
    (SELECT extract(ISODOW FROM CURRENT_DATE))::INT);

```

```

-----
--- Get all available doctors for time
--- 2022-04-05T08:15:43.258Z
-----

```

```

SELECT
    doctor.first_name || ' ' || doctor.last_name AS full_name,
    doctor_timetable,
    d.name,
    address
FROM get_all_available_doctors_at_specific_time('2022-04-05T08:15:43.258Z'::TIMESTAMP)
    JOIN doctor ON doctor.id = doctor_id
    JOIN department d ON department_id = d.id
    JOIN building b ON building_id = b.id
WHERE b.name = 'National Hospital of Chernigiv';

```

```

-----
--- Get doctors timetables for current day
-----

```

```

SELECT
    doc.first_name || ' ' || doc.last_name AS full_name,
    CASE
        WHEN extract(ISODOW FROM current_date) IN (6, 7) THEN 'no timetable, its weekend'
        ELSE get_doctor_timetable(doc.id, extract(ISODOW FROM current_date)::INT)
    END
FROM doctor doc;

```

```

-----
--- Get all medical procedures for specific equipment
---
-----

```

```

SELECT
    equipment.model AS Model,
    d.name AS department_name,
    p.first_name AS patinet_first_name,
    p.last_name AS patient_last_name,
    to_char(time, 'YYYY-MM-DD HH24:MI') AS starting_time,
    duration_minutes
FROM equipment
    JOIN medical_procedure mp on equipment.id = mp.equipment_id
    JOIN patient p on mp.patient_id = p.id
    JOIN department d on equipment.department_id = d.id
WHERE equipment.model = 'Ингалятор' AND d.name = 'Интенсивной терапии'
ORDER BY equipment.model;

```

```

-----
--- Get all available cabinets for
--- 2022-01-25 10:27:49.203000 for duration of 10 minutes
--- for building named by Makarov
-----

```

```

SELECT
    cabinet_number AS free_cabinet
FROM get_free_cabinets((SELECT id FROM building WHERE building.name = 'Building named by
Makarov'),
    '2022-01-25 10:27:49.203000',
    10)
ORDER BY free_cabinet;

```

```

-----
--- Get details of all doctors in the
--- National Hostital of Chernigiv
-----

```

```

SELECT
    building.name,
    department.name,
    doctor.first_name,
    doctor.last_name,
    doctor.experience
FROM building
    JOIN department on building.id = department.building_id
    JOIN doctor on department.id = doctor.department_id
WHERE building.name = 'National Hostital of Chernigiv'
ORDER BY doctor.first_name, doctor.last_name;

```

--- Get specific patient appointments

```

SELECT
    doctor.first_name AS doctor_name,
    p.first_name AS patient_name,
    da.time AS appointment_time
FROM doctor
    JOIN doctor_appointment da on doctor.id = da.doctor_id
    JOIN patient p on da.patient_id = p.id
WHERE p.first_name = 'Christal' AND p.last_name = 'Graig'
ORDER BY doctor_name, patient_name;

```

--- Get summaric experience for each building
--- for each department

```

SELECT
    b.name AS building_name,
    d.name AS department_name,
    sum(doctor.experience) AS summaric_experience
FROM doctor
    JOIN department d ON department_id = d.id
    JOIN building b ON building_id = b.id
GROUP BY d.name, b.name
ORDER BY summaric_experience DESC;

```

--- Get all upcoming medical procedures

```

SELECT
    patient.first_name || ' ' || patient.last_name AS full_patient_name,
    time AS procedure_time,

```

```

    duration_minutes,
    model AS equipment_model,
    d.name AS department_name
FROM patient
JOIN medical_procedure mp ON patient.id = mp.patient_id
JOIN equipment equip ON mp.equipment_id = equip.id
JOIN department d ON equip.department_id = d.id
WHERE current_time < mp.time::TIME
ORDER BY time DESC;

```

TRIGGERS:

```

CREATE OR REPLACE FUNCTION check_appointment_time_func()
    RETURNS TRIGGER
    LANGUAGE plpgsql
AS
$check_app_time$
    DECLARE error_message VARCHAR(255);
    DECLARE day_of_week INT;
    DECLARE timetable VARCHAR(255);

    BEGIN
        SELECT extract(ISODOW FROM New.time) INTO day_of_week;
        SELECT get_doctor_timetable(New.doctor_id, day_of_week) INTO timetable;

        SELECT concat(
            'Patient: ',
            New.patient_id::VARCHAR, ' cant have an appointment with doctor: ',
            New.doctor_id::VARCHAR, ' at ',
            New.time::VARCHAR, ' because doctor is not working at that time.'
        ) INTO error_message;

        IF New.is_closed = True -- that means we are trying to close the appointment
        THEN
            RETURN New;
        END IF;

        IF day_of_week = 6 OR day_of_week = 7 -- checking weekends
        THEN
            SELECT raise_error(error_message);
        END IF;

        -- 16:00-19:15
        IF (extract(HOUR FROM New.time) > substr(timetable, 1, 2)::INT -- 16
            OR (extract(HOUR FROM New.time) = substr(timetable, 1, 2)::INT AND
                extract(MINUTE FROM New.time) >= substr(timetable, 4, 2)::INT) -- 00

```

```

        AND (extract(HOUR FROM New.time) < substr(timetable, 7, 2)::INT -- 19
              OR (extract(HOUR FROM New.time) = substr(timetable, 7, 2)::INT AND
extract(MINUTE FROM New.time) <= substr(timetable, 10, 2)::INT))) -- 15
    THEN
        RETURN New;
    END IF;

    SELECT raise_error(error_message);

END
$check_app_time$;

```

```

CREATE OR REPLACE FUNCTION check_medical_procedure_available_func()
    RETURNS TRIGGER
    LANGUAGE plpgsql
    AS
$check_medical_proc_func$
    DECLARE error_message VARCHAR(355);

    BEGIN
        SELECT concat(
            'Patient: ',
            New.patient_id::VARCHAR, ' cant have an procedure with equipment: ',
            New.equipment_id::VARCHAR, ' at ',
            New.time::VARCHAR, ' because it is busy at that moment.'
        ) INTO error_message;

        IF exists(
            SELECT 1 FROM medical_procedure
            WHERE (equipment_id = New.equipment_id OR patient_id = New.patient_id) AND
New.time >= time
            AND New.time <= (time + (duration_minutes || ' minutes')::INTERVAL)
        )
        THEN
            SELECT raise_error(error_message);
        END IF;

        RETURN New;
    END
$check_medical_proc_func$;

```

```

CREATE TRIGGER check_medical_procedure_available BEFORE UPDATE OR INSERT ON
medical_procedure
    FOR EACH ROW EXECUTE PROCEDURE check_medical_procedure_available_func();

```

```

CREATE OR REPLACE FUNCTION check_uniques_of_appointment_func()
  RETURNS TRIGGER
  LANGUAGE plpgsql
  AS
$check_uniques_appointment$
  DECLARE error_message VARCHAR(255);

  BEGIN
    SELECT concat(
      'Patient: ',
      New.patient_id::VARCHAR, ' cant have an appointment with doctor: ',
      New.doctor_id::VARCHAR, ' at ',
      New.time::VARCHAR, ' because cabinet is busy at that moment or doctor or patient is
having another appointment at that time'
    ) INTO error_message;

    IF New.is_closed = True -- that means we are trying to close the appointment
    THEN
      RETURN new;
    END IF;

    IF exists(
      SELECT 1 FROM doctor_appointment
      WHERE (doctor_id = New.doctor_id OR patient_id = New.patient_id OR
(New.cabinet_number = cabinet_number AND New.building_id = building_id))
      AND New.time >= time
      AND New.time <= (time + (duration_minutes || ' minutes')::INTERVAL)
      AND is_closed = False
    )
    THEN
      SELECT raise_error(error_message);
    END IF;

    RETURN NEW;
  END;
$check_uniques_appointment$;

CREATE TRIGGER check_uniques_of_appointment BEFORE INSERT OR UPDATE ON
doctor_appointment
  FOR EACH ROW EXECUTE PROCEDURE check_uniques_of_appointment_func();

INSERT INTO doctor_appointment (id, patient_id, doctor_id, building_id, cabinet_number, time,
reason, is_closed, duration_minutes) VALUES
(100, 1, 7, 3, 149, '2021-11-30 15:18:51.593000', 'some reason', false, 23);

```

VIEWS:

```

CREATE OR REPLACE VIEW get_full_information_about_appointments AS
SELECT
    p.first_name || ' ' || p.last_name AS patient_name,
    d.first_name || ' ' || d.last_name AS doctor_name,
    b.name AS building_name,
    cabinet_number,
    reason,
    time AS appointment_time,
    duration_minutes
FROM doctor_appointment da
    JOIN doctor d ON doctor_id = d.id
    JOIN patient p ON patient_id = p.id
    JOIN building b ON building_id = b.id
WHERE is_closed = false;

```

```

SELECT * FROM get_full_information_about_appointments;

```

```

CREATE OR REPLACE VIEW get_full_information_about_medical_procedures AS
SELECT
    p.first_name || ' ' || p.last_name AS patient_full_name,
    e.model AS model,
    time,
    duration_minutes
FROM medical_procedure
    JOIN patient p ON patient_id = p.id
    JOIN equipment e ON equipment_id = e.id
WHERE time AT TIME ZONE 'UTC' > current_timestamp;

```

```

SELECT * FROM get_full_information_about_medical_procedures;

```

FUNCTIONS

```

CREATE OR REPLACE FUNCTION close_appointment(appointment_id INT)
RETURNS VOID
LANGUAGE plpgsql
AS
$close_appointment$
BEGIN
    UPDATE doctor_appointment
    SET is_closed = True
    WHERE id = appointment_id;
END;
$close_appointment$;

```

```

CREATE OR REPLACE FUNCTION create_appointment(doctor_id INT, patient_id INT,
appointment_time TIMESTAMP, appointment_reason TEXT,
appointment_duration_minutes INT)
    RETURNS VOID
    LANGUAGE plpgsql
    AS
$create_appoint$
    DECLARE doctor_building_id INT;
    DECLARE free_cabinets INT ARRAY;
    DECLARE free_cabinets_array_length INT;
    DECLARE random_cabinet_number INT;

    BEGIN
        SELECT building_id
            INTO doctor_building_id
        FROM doctor
            JOIN department d on doctor.department_id = d.id
            JOIN building h on d.building_id = h.id
        WHERE doctor.id = doctor_id;

        free_cabinets := ARRAY(
            SELECT
                da.cabinet_number
            FROM doctor_appointment da
            WHERE da.building_id = doctor_building_id AND
da.is_closed = False AND (
                appointment_time < time OR
                appointment_time > (time +
(appointment_duration_minutes || ' minutes')::INTERVAL)
            )
        );

        SELECT array_length(free_cabinets, 1)
            INTO free_cabinets_array_length;

        SELECT free_cabinets[get_random_integer(1, free_cabinets_array_length)]
            INTO random_cabinet_number;

        INSERT INTO doctor_appointment(patient_id, doctor_id, building_id,
cabinet_number, time, reason, is_closed, duration_minutes)
            VALUES
                ($2, $1, doctor_building_id, random_cabinet_number, appointment_time,
appointment_reason, False, appointment_duration_minutes);
    END
$create_appoint$

CREATE OR REPLACE FUNCTION get_doctor_timetable(doc_id INT, day_of_week INT)
    RETURNS VARCHAR(255)
    LANGUAGE plpgsql
    AS
$get_docktor_tt$
    DECLARE doctor_timetable VARCHAR;
    BEGIN

```

```

SELECT
CASE
    WHEN day_of_week = 1 THEN monday_timetable
    WHEN day_of_week = 2 THEN thursday_timetable
    WHEN day_of_week = 3 THEN wednesday_timetable
    WHEN day_of_week = 4 THEN thursday_timetable
    WHEN day_of_week = 5 THEN friday_timetable
    ELSE 'No timetable'
END
FROM doctor_timetable
WHERE id IN (SELECT timetable_id FROM doctor WHERE id = doc_id) INTO
doctor_timetable;

RETURN doctor_timetable;
END;
$get_docktor_tt$;

CREATE OR REPLACE FUNCTION get_all_available_doctors_at_specific_time(for_time
TIMESTAMP)
RETURNS TABLE (doctor_id INT, doctor_name VARCHAR, doctor_timetable VARCHAR)
LANGUAGE plpgsql
AS
$$
DECLARE row RECORD;

BEGIN
    IF (extract(ISODOW FROM for_time)::INT IN (6, 7))
    THEN
        RAISE EXCEPTION 'Dont try to fool me, its a weekend';
    END IF;

    FOR row IN
        SELECT
            doc.id AS doc_id,
            doc.first_name AS doc_name,
            get_doctor_timetable(doc.id, extract(ISODOW FROM for_time)::INT) AS timetable
        FROM doctor doc
    LOOP
        IF (
            (
                extract(HOUR FROM for_time) > substr(row.timetable, 1, 2)::INT
                OR (extract(HOUR FROM for_time) = substr(row.timetable, 1, 2)::INT AND
extract(MINUTE FROM for_time) >= substr(row.timetable, 4, 2)::INT)
            )
            AND
            (
                extract(HOUR FROM for_time) < substr(row.timetable, 7, 2)::INT
                OR (extract(HOUR FROM for_time) = substr(row.timetable, 7, 2)::INT AND
extract(MINUTE FROM for_time) <= substr(row.timetable, 10, 2)::INT)
            )
        )
    END LOOP

```

```

) THEN
    doctor_id := row.doc_id;
    doctor_name := row.doc_name;
    doctor_timetable := row.timetable;

    RETURN NEXT;
ELSE
    RAISE INFO 'Doctor timetable: %', row.timetable;
END IF;
END LOOP;
END;
$$;

```

```

SELECT * FROM get_all_available_doctors_at_specific_time('2022-04-
05T08:15:43.258Z'::TIMESTAMP)

```

```

CREATE OR REPLACE FUNCTION get_free_cabinets(building_id INT, for_time TIMESTAMP,
duration_minutes INT)
    RETURNS TABLE (cabinet_number INT)
    LANGUAGE plpgsql
    AS
$$
BEGIN
    RETURN QUERY
    SELECT
        da.cabinet_number
    FROM doctor_appointment da
    WHERE da.building_id = $1 AND da.is_closed = False AND (
        $2 < time OR
        $2 > (time + ($3 || ' minutes')::INTERVAL)
    );
END;
$$

```

```

CREATE OR REPLACE FUNCTION raise_error(text VARCHAR(255))
    RETURNS VOID
    LANGUAGE plpgsql
    AS
$raise_err$
BEGIN
    RAISE EXCEPTION '%s', $1;
END;
$raise_err$;

```

```

CREATE OR REPLACE FUNCTION get_random_integer(low INT, high INT)
    RETURNS INT
    LANGUAGE plpgsql

```

```

AS
$$
BEGIN
    RETURN floor(random() * (high-low + 1) + low);
END;
$$

```

CREATE TABLES

```

CREATE TABLE IF NOT EXISTS Building (
    id SERIAL PRIMARY KEY,
    address VARCHAR(255) NOT NULL,
    name VARCHAR(255) NOT NULL
);

```

```

CREATE TABLE IF NOT EXISTS Department (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    building_id INT NOT NULL,

    CONSTRAINT fk_on_building
        FOREIGN KEY (building_id) REFERENCES Building (id) ON DELETE CASCADE ON UPDATE
        CASCADE
);

```

```

CREATE TABLE IF NOT EXISTS Equipment (
    id SERIAL PRIMARY KEY,
    model VARCHAR(255) NOT NULL,
    department_id INT NULL,

    CONSTRAINT fk_on_department
        FOREIGN KEY (department_id) REFERENCES department (id) ON DELETE CASCADE ON
        UPDATE CASCADE
);

```

```

CREATE TABLE IF NOT EXISTS Doctor_specialization (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL
);

```

```

CREATE TABLE IF NOT EXISTS Patient (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    age INT NULL,

```

```

sex VARCHAR(10) NULL,

CONSTRAINT check_sex
    CHECK (sex = 'male' OR sex = 'female'),
CONSTRAINT check_age
    CHECK (age > 0 AND age < 150)
);

CREATE TABLE IF NOT EXISTS Cabinet (
    building_id INT NOT NULL,
    number INT NOT NULL,

    PRIMARY KEY (building_id, number)
);

CREATE TABLE IF NOT EXISTS Doctor(
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    experience INT NULL,

    specialization_id INT NULL,
    department_id INT NULL,
    timetable_id INT NULL,

    FOREIGN KEY (specialization_id) REFERENCES Doctor_specialization (id),
    FOREIGN KEY (department_id) REFERENCES Department(id)
);

CREATE TABLE IF NOT EXISTS Doctor_timetable (
    id SERIAL PRIMARY KEY,
    monday_timetable VARCHAR(255),
    tuesday_timetable VARCHAR(255),
    wednesday_timetable VARCHAR(255),
    thursday_timetable VARCHAR(255),
    friday_timetable VARCHAR(255)
);

ALTER TABLE Doctor
ADD CONSTRAINT fk_on_timetable
FOREIGN KEY (timetable_id) REFERENCES Doctor_timetable(id);

CREATE TABLE IF NOT EXISTS Medical_procedure (
    id SERIAL PRIMARY KEY,
    equipment_id INT NOT NULL,
    time TIMESTAMP NOT NULL,
    duration_minutes INT NOT NULL,
    patient_id INT NOT NULL,

```

```

FOREIGN KEY (equipment_id) REFERENCES Equipment (id),
FOREIGN KEY (patient_id) REFERENCES Patient (id)
);

CREATE TABLE IF NOT EXISTS Doctor_appointment (
    id SERIAL PRIMARY KEY,
    patient_id INT NOT NULL,
    doctor_id INT NOT NULL,
    building_id INT NOT NULL,
    cabinet_number INT NOT NULL,
    time TIMESTAMP NOT NULL,
    reason TEXT NULL,
    is_closed BOOL DEFAULT FALSE,
    duration_minutes INT NOT NULL,

    FOREIGN KEY (patient_id) REFERENCES Patient(id),
    FOREIGN KEY (doctor_id) REFERENCES Doctor(id),
    FOREIGN KEY (building_id, cabinet_number) REFERENCES Cabinet(building_id, number)
);

```

FILL TABLES

```

COPY building
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/building.csv'
DELIMITER ','
CSV HEADER;

```

```

COPY department
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/departments.csv'
DELIMITER ','
CSV HEADER;

```

```

COPY doctor_specialization
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/doctor-
specializations.csv'
DELIMITER ','
CSV HEADER;

```

```

COPY doctor_timetable
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/timetables.csv'
DELIMITER ','
CSV HEADER;

```

```

COPY cabinet
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/cabinets.csv'
DELIMITER ','
CSV HEADER;

```

```

COPY patient
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/patients.csv'

```



```
DELIMITER ','  
CSV HEADER;
```

```
COPY doctor  
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/doctors.csv'  
DELIMITER ','  
CSV HEADER;
```

```
COPY equipment  
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/equipment.csv'  
DELIMITER ','  
CSV HEADER;
```

```
COPY medical_procedure(id,time,duration_minutes,equipment_id,patient_id)  
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/medical-  
procedures.csv'  
DELIMITER ','  
CSV HEADER;
```

```
COPY  
doctor_appointment(id,patient_id,doctor_id,time,duration_minutes,building_id,cabinet_number,r  
eason,is_closed)  
FROM '/home/kinfi4/Documents/db/hospital-database-system/source/data/doctor-  
appointments.csv'  
DELIMITER ','  
CSV HEADER;
```