

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»  
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ  
(повна назва інституту/факультету)

КАФЕДРА інформатики та програмної інженерії  
(повна назва кафедри)

**КУРСОВА РОБОТА**

з дисципліни «Бази даних»  
(назва дисципліни)

на тему: База даних лікувального закладу

Студента (ки) 2 курсу ПІ-13 групи  
спеціальності 121 «Інженерія програмного  
забезпечення»

Бондаренка Максима Вікторовича  
(прізвище та ініціали)

Керівник: Ліщук Олександр Васильович  
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала \_\_\_\_\_

Кількість балів: \_\_\_\_\_ Оцінка ECTS \_\_\_\_\_

Члени комісії

_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)

Київ – 2022 рік

**Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет Інформатики та обчислювальної техніки  
(повна назва)

Кафедра Інформатики та програмної інженерії  
(повна назва)

Дисципліна Бази даних

Курс 2 Група ІІІ-13 Семестр 3

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

Бондаренку Максиму Вікторовичу

(прізвище, ім'я, по батькові)

1. Тема роботи: База даних лікувального закладу

керівник роботи: Ліщук Олександр Васильович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 25.01.2023

3. Вихідні дані до роботи завдання на розробку бази даних лікувального закладу

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

1) Аналіз предметного середовища

2) Побудова ER-моделі

3) Побудова реляційної схеми з ER-моделі

4) Створення бази даних, у форматі обраної СУБД

5) Створення користувачів бази даних

6) Імпорт даних з використанням засобів СУБД в базу даних

7) Створення мовою SQL запитів

8) Оптимізація роботи запитів

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

6. Дата видачі завдання: 08.11.2022

## КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання курсового проекту	Строк виконання етапів проекту	Примітка
1	Аналіз предметного середовища	19.01.23	
2	Побудова ER-моделі	19.01.23	
3	Побудова реляційної схеми з ER-моделі	20.01.23	
4	Створення бази даних, у форматі обраної системи управління базою даних	20.01.23	
5	Створення користувачів бази даних	20.01.23	
6	Імпорт даних з використанням засобів СУБД в створену базу даних	21.01.23	
7	Створення мовою SQL запитів	23.01.23	
8	Оптимізація роботи запитів	24.01.23	
9	Оформлення пояснювальної записки	25.01.23	
10	Захист курсової роботи	26.01.23	

Студент

\_\_\_\_\_  
(підпис )

Бондаренко М.В.  
(прізвище та ініціали)

Керівник роботи

\_\_\_\_\_  
(підпис )

Ліщук О. В.  
(прізвище та ініціали)

## ЗМІСТ

ВСТУП .....	5
1 Опис предметного середовища .....	6
2 Постановка задачі .....	7
3 ER-діаграма .....	8
3.1 Бізнес-правила .....	8
3.2 Вибір сутностей.....	8
3.3 Набори атрибутів сутностей .....	9
4 Реляційна модель бази даних .....	12
4.1 Побудова необхідних відношень, визначення первинних та зовнішніх ключів .....	12
5 Реалізація бази даних .....	13
5.1 Створення бази даних.....	13
5.2 Імпортування даних .....	18
6 Створення користувачів бази даних .....	20
6.1 Адміністратор.....	20
6.2 Лікар .....	20
6.3 Пацієнт .....	20
7 SQL запити .....	21
7.1 Створення тригерів на таблиці .....	21
7.2 Створення процедур .....	25
7.3 Створення функцій .....	28
7.4 Створення представлень.....	30
7.5 Створення різних запитів .....	32
7.6 Створення індексів.....	43

Висновок .....	44
Перелік посилань.....	45

## ВСТУП

Бази даних стали невід'ємною частиною сучасних обчислювальних систем, надаючи життєво важливу послугу для зберігання та керування великими обсягами даних. Вони широко використовуються в багатьох галузях і додатках і відіграють вирішальну роль в організації, пошуку та аналізі структурованої інформації.

Бази даних надають організаціям інструменти, необхідні для ефективного зберігання та отримання цієї інформації, вони також пропонують потужні інструменти для обробки та аналізу даних. Крім того, вони забезпечують послідовність і точність даних. Варто відмітити, що також присутні заходи безпеки, такі як автентифікація користувачів і шифрування даних для захисту конфіденційної інформації.

У цьому курсі ми заглибимося в основи баз даних, включаючи їх дизайн, структуру та функціональність, а також дослідимо використання ефективного сховища даних при розробці бази даних лікувального закладу.

## 1 ОПИС ПРЕДМЕТНОГО СЕРЕДОВИЩА

Лікувальний заклад – це складне та динамічне середовище, яке потребує надійної та ефективної бази даних для керування та організації величезного обсягу інформації, яку вона генерує. Предметне середовище для бази даних лікувального закладу включає різні сутності, такі як Patient, Doctor, Appointment, Service, Room, Medication та ін. Кожна сутність має власний набір атрибутів і зв'язків з іншими сутностями, які потрібно фіксувати та керувати ними в базі даних.

Пацієнти є основною метою медичного закладу, і їхня інформація, така як особисті дані, історія хвороби, інформація про страхування та медичні записи, має точно зберігатися та бути легкодоступною. База даних також повинна відстежувати зустрічі пацієнтів з лікарями та медсестрами та лікування, яке вони отримують.

Лікарі є ще одними важливими суб'єктами медичного закладу. Їхня інформація, така як ім'я, спеціальність, досвід та контактна інформація, має зберігатися та бути легкодоступною. База даних також повинна відстежувати зустрічі лікарів з пацієнтами та послуги, які вони надають.

Ліки є життєво важливим аспектом догляду за пацієнтами, і інформація про них, як-от назва, дозування, доступність і ціна, має зберігатися та бути легкодоступною. База даних також повинна відстежувати ліки, призначені пацієнтам під час прийому, та їх використання.

Палати є критично важливим компонентом середовища медичного закладу, і інформацію про них необхідно відстежувати та керувати в базі даних. Сутність кімнати включає різні атрибути, такі як номер кімнати, тип, місткість, доступність. База даних повинна відстежувати заселення пацієнтів в кімнатах.

Загалом тематичне середовище бази даних для лікувального закладу передбачає керування та організацію великого обсягу інформації, пов'язаної з пацієнтами, лікарями, призначеннями, лікуванням, ліками та ін. База даних повинна мати можливість обробляти великий обсяг даних і забезпечувати легкий доступ до інформації для медичного персоналу.

## 2 ПОСТАНОВКА ЗАДАЧІ

Метою роботи є розробка бази даних лікувального закладу, а саме організація даних таким чином, щоб робота закладу виконувалась максимально ефективно.

Основними задачами бази даних буде зберігання інформації, пов'язаної з лікувальним закладом та надання змоги користувачам ефективно її використовувати. Таким чином лікарі повинні мати змогу дивитись та змінювати інформацію про прийоми, виписувати ліки до кожного прийому, передивлятися повну інформацію про пацієнтів. Пацієнти в свою чергу мають вільно дивитись всю інформацію про ліки, які їм виписують, прийоми, щоб знати дату та іншу інформацію, інформацію про лікарів, щоб розуміти їх досвід, власні медичні дані та інформацію про палати, куди вони можуть лягти.



### 3 ER-ДІАГРАМА

#### 3.1 Бізнес-правила

1. За один прийом сплата може відбуватись лише з однієї картки.
2. В одному прийомі може бути лише один доктор та один пацієнт.
3. Пацієнт може одночасно бути заселеним лише в одну палату та дата заселення має бути раніше за дату виселення.
4. До одного прийому може бути виписано декілька видів ліків.
5. Один пацієнт має лише одну медичну картку.
6. Під час одного прийому може надаватись лише одна послуга.

#### 3.2 Вибір сутностей

- Patient
- Doctor
- Room
- Room Housing
- Medical Card
- Medication Appointment
- Medication
- Appointment
- Service
- Payment

### 3.3 Набори атрибутів сутностей

Таблиця 3.1 – Сутності та їхні атрибути

Сутність	Атрибути
Patient	id PRIMARY KEY patient_name date_of_birth sex address phone_number
Doctor	id PRIMARY KEY doctor_name speciality years_of_experience phone_number
Room	id PRIMARY KEY room_number type capacity availability
Room Housing	id PRIMARY KEY patient_id FOREIGN KEY room_id FOREIGN KEY start_date end_date
Medical Card	id PRIMARY KEY patient_id FOREIGN KEY height weight blood_type allergies

Продовження таблиці 3.1

Сутність	Атрибути
	medical_history availability_of_insurance
Medication Appointment	id PRIMARY KEY appointment_id FOREIGN KEY medication_id FOREIGN KEY
Medication	id PRIMARY KEY medication_name dosage_in_mg availability price
Appointment	id PRIMARY KEY service_id FOREIGN KEY doctor_id FOREIGN KEY patient_id FOREIGN KEY payment_id FOREIGN KEY appointment_date
Service	id PRIMARY KEY service_name duration_minutes price
Payment	id PRIMARY KEY payment_date amount payment_method

Сутність Doctor має зв'язок 1:N із сутністю Appointment, оскільки один лікар може мати багато прийомів.

Сутність Payment має зв'язок 1:N із сутністю Appointment, оскільки одна транзакція може бути за декілька прийомів.

Сутність Service має зв'язок 1:N із сутністю Appointment, оскільки одна послуга може виконуватись на різних прийомах.

Сутність Medication має зв'язок 1:N із сутністю Medication Appointment, оскільки один вид ліків може виписуватись для різних прийомів.

Сутність Appointment має зв'язок 1:N із сутністю Medication Appointment, оскільки до одного прийому може бути виписано декілька видів ліків.

Сутність Patient має зв'язок 1:N із сутністю Appointment, оскільки один пацієнт може бути записаний на декілька прийомів.

Сутність Patient має зв'язок 1:N із сутністю Room Housing, оскільки один пацієнт може займати різні палати в різний час.

Сутність Medical Card має зв'язок 1:1 із сутністю Patient, оскільки одна медична карта належить лише одному пацієнту.

Сутність Room має зв'язок 1:N із сутністю Room Housing, оскільки одна палата може містити декілька пацієнтів.

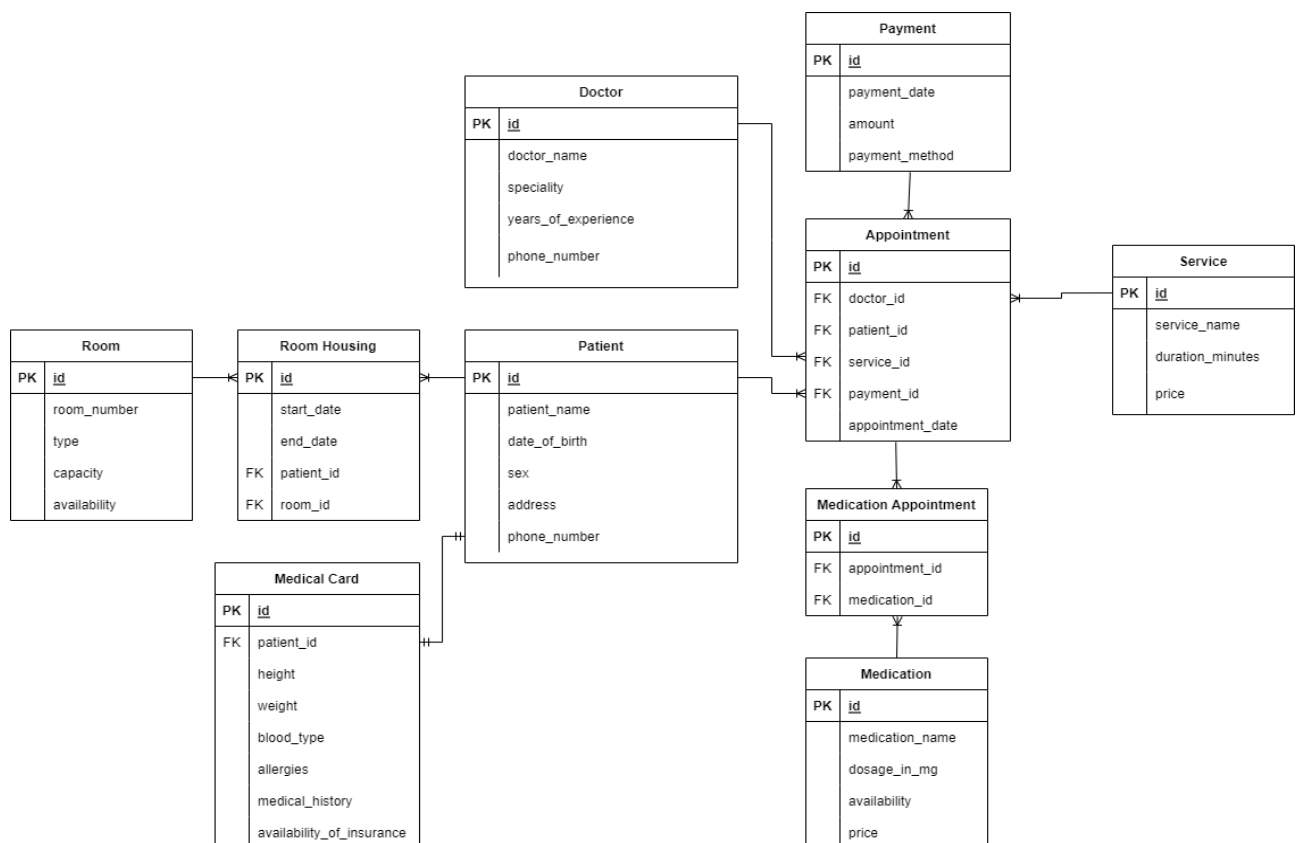


Рисунок 3.1 – ER-діаграма

## 4 РЕЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ

### 4.1 Побудова необхідних відношень, визначення первинних та зовнішніх ключів

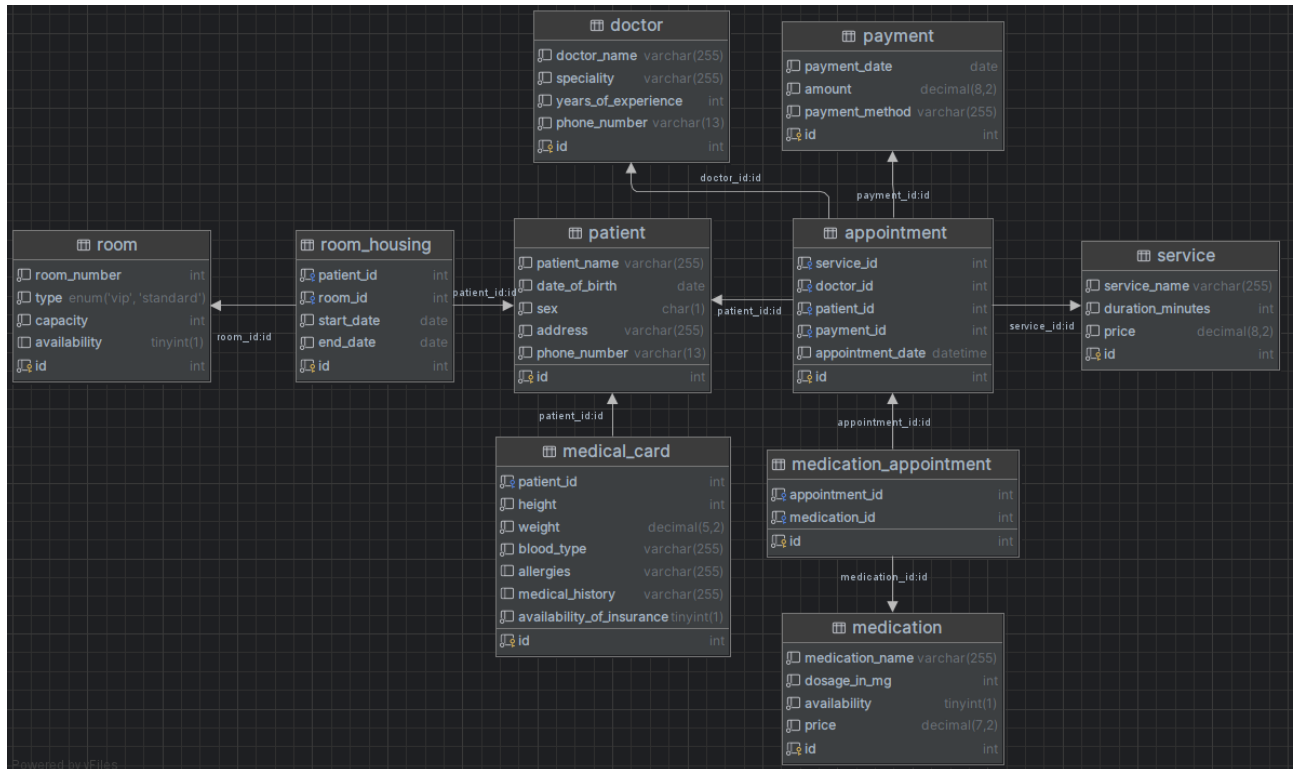


Рисунок 4.1– Реляційна схема бази даних

За схемою видно, що всі поля таблиць декомпозовані, також всі атрибути таблиць функціонально повно залежать від первинного ключа, кожен неключовий атрибут не є транзитивно залежним від первинного ключа, отже база даних знаходиться у третій нормальній формі.

1. Обов'язкові атрибути таблиць мають обмеження NOT NULL, для запобігання помилок при роботі з даними.
2. Забезпечуються каскадні дії при видаленні зовнішніх ключів однієї з таблиць (ON DELETE CASCADE).

## 5 РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

### 5.1 Створення бази даних

```
drop database hospital;  
create database if not exists hospital;  
use hospital;  
  
drop table if exists patient;  
drop table if exists doctor;  
drop table if exists room;  
drop table if exists room_housing;  
drop table if exists medical_card;  
drop table if exists medication_appointment;  
drop table if exists medication;  
drop table if exists appointment;  
drop table if exists service;  
drop table if exists payment;  
  
create table patient  
(  
    id int auto_increment,  
    patient_name varchar(255) not null,  
    date_of_birth date not null,  
    sex char not null,  
    address varchar(255) not null,  
    phone_number varchar(13) not null,  
    primary key (id)  
);
```

```
create table doctor
(
    id int auto_increment,
    doctor_name varchar(255) not null,
    speciality varchar(255) not null,
    years_of_experience int not null,
    phone_number varchar (13) not null,
    primary key (id)
);
```

```
create table room
(
    id int auto_increment,
    room_number int not null,
    type enum('VIP', 'Standard') not null,
    capacity int not null,
    availability bool default 1,
    primary key (id)
);
```

```
create table room_housing
(
    id int auto_increment,
    patient_id int not null,
    room_id int not null,
    start_date date not null,
    end_date date not null,
    primary key (id)
```

);

create table medical\_card

```
(
    id int auto_increment,
    patient_id int unique not null,
    height int not null,
    weight decimal(5,2) not null,
    blood_type varchar(255) not null,
    allergies varchar(255) null,
    medical_history varchar(255) null,
    availability_of_insurance bool not null,
    primary key (id)
);
```

create table medication\_appointment

```
(
    id int auto_increment,
    appointment_id int not null,
    medication_id int not null,
    primary key (id)
);
```

create table medication

```
(
    id int auto_increment,
    medication_name varchar(255) unique not null,
    dosage_in_mg int not null,
    availability bool not null,
```



```
    price decimal(7,2) not null,  
    primary key (id)  
);
```

```
create table appointment  
(  
    id int auto_increment,  
    service_id int not null,  
    doctor_id int not null,  
    patient_id int not null,  
    payment_id int not null,  
    appointment_date datetime not null,  
    primary key (id)  
);
```

```
create table service  
(  
    id int not null,  
    service_name varchar(255) unique not null,  
    duration_minutes int not null,  
    price decimal(8, 2) not null,  
    primary key (id)  
);
```

```
create table payment  
(  
    id int auto_increment,  
    payment_date date not null,  
    amount decimal(8, 2) not null,  
    payment_method varchar(255) not null,
```

```
primary key (id)
);

alter table room_housing
add constraint room_housing_patient_fk
foreign key (patient_id) references patient(id)
on delete cascade;

alter table room_housing
add constraint room_housing_room_fk
foreign key (room_id) references room(id)
on delete cascade;

alter table medical_card
add constraint medical_card_fk
foreign key (patient_id) references patient(id)
on delete cascade;

alter table appointment
add constraint appointment_doctor_fk
foreign key (doctor_id) references doctor(id)
on delete cascade;

alter table appointment
add constraint appointment_patient_fk
foreign key (patient_id) references patient(id)
on delete cascade;

alter table appointment
add constraint appointment_service_fk
```

```
foreign key (service_id) references service(id)  
on delete cascade;
```

```
alter table appointment  
add constraint appointment_payment_fk  
foreign key (payment_id) references payment(id)  
on delete cascade;
```

```
alter table medication_appointment  
add constraint medication_appointment_med_fk  
foreign key (medication_id) references medication(id)  
on delete cascade;
```

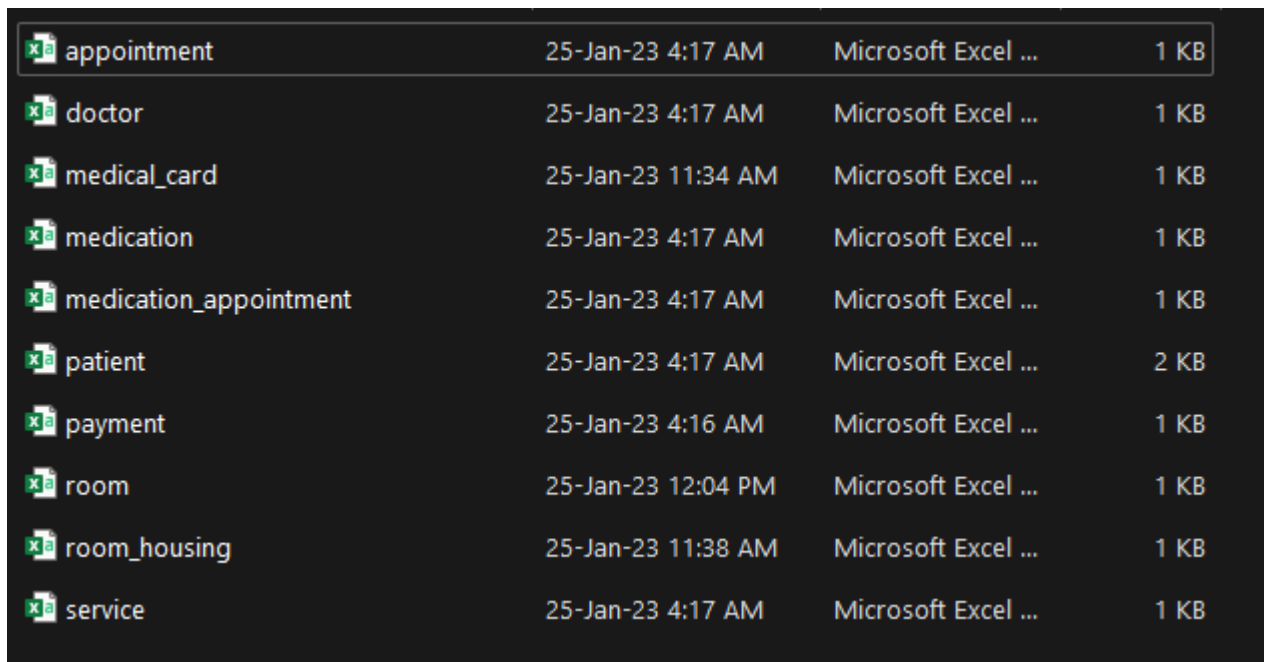
```
alter table medication_appointment  
add constraint medication_appointment_app_fk  
foreign key (appointment_id) references appointment(id)  
on delete cascade;
```

## 5.2 Імпортування даних

Для заповнення даними було використано можливість імпорту з CSV файлів.

Для кожної таблиці було створено окремий файл з відповідними даними.

Перелік CSV файлів:













 appointment	25-Jan-23 4:17 AM	Microsoft Excel ...	1 KB
 doctor	25-Jan-23 4:17 AM	Microsoft Excel ...	1 KB
 medical_card	25-Jan-23 11:34 AM	Microsoft Excel ...	1 KB
 medication	25-Jan-23 4:17 AM	Microsoft Excel ...	1 KB
 medication_appointment	25-Jan-23 4:17 AM	Microsoft Excel ...	1 KB
 patient	25-Jan-23 4:17 AM	Microsoft Excel ...	2 KB
 payment	25-Jan-23 4:16 AM	Microsoft Excel ...	1 KB
 room	25-Jan-23 12:04 PM	Microsoft Excel ...	1 KB
 room_housing	25-Jan-23 11:38 AM	Microsoft Excel ...	1 KB
 service	25-Jan-23 4:17 AM	Microsoft Excel ...	1 KB

Рисунок 5.1 – CSV файли з даними

Імпорт даних здійснювався за допомогою скриптів наступного вигляду:

```
load data infile 'D:/KPI/DataBase/semester work/csv_files/patient.csv' into table patient
fields terminated by ','
enclosed by '"'
lines terminated by '\n'
ignore 1 rows;
```

## 6 СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ

### 6.1 Адміністратор

```
drop role if exists administrator;  
create role administrator;  
grant all on hospital.* to administrator;  
drop user if exists administrator@localhost;  
create user administrator@localhost identified by 'administrator';  
grant administrator to administrator@localhost;
```

### 6.2 Лікар

```
drop role if exists doctor;  
create role doctor;  
grant select on hospital.* to doctor;  
grant insert, update, delete on table hospital.medical_card to doctor;  
grant insert, update, delete on table hospital.appointment to doctor;  
grant insert, update, delete on table hospital.medication_appointment to doctor;  
drop user if exists doctor@localhost;  
create user doctor@localhost identified by 'doctor';  
grant doctor to doctor@localhost;
```

### 6.3 Пацієнт

```
drop role if exists patient;  
create role patient;  
grant select on hospital.appointment to patient;  
grant select on hospital.medical_card to patient;  
grant select on hospital.doctor to patient;  
grant select on hospital.room to patient;  
grant select on hospital.service to patient;  
grant select on hospital.medication to patient;  
drop user if exists patient@localhost;  
create user patient@localhost identified by 'patient';  
grant patient to patient@localhost;
```

## 7 SQL ЗАПИТИ

### 7.1 Створення тригерів на таблиці

Тригер забороняє додавати пацієнтів в палату, де немає місць.

```
drop trigger if exists full_rooms;
```

```
delimiter $$
```

```
create trigger full_rooms before insert on room_housing for each row
```

```
begin
```

```
    if NEW.room_id in (select id
```

```
        from room
```

```
        where availability = 0) then
```

```
        signal sqlstate '45000'
```

```
        set message_text = 'The room has the maximum number of patients. Choose
other room.';
```

```
    end if;
```

```
end $$
```

```
delimiter ;
```

Приклад роботи:

Палата, що має id = 1 не має вільних місць (availability = 0):

	id	room_number	type	capacity	availability
1	1	11	Standard	3	0
2	2	12	VIP	1	0
3	3	13	Standard	4	1
4	4	14	Standard	4	1
5	5	15	Standard	6	1
6	6	21	Standard	4	1
7	7	22	VIP	2	1
8	8	23	Standard	3	1
9	9	24	Standard	6	1
10	10	25	Standard	4	1

Рисунок 7.1.1 – Інформація про палати

При додаванні пацієнта в дану палату отримаємо помилку, що унеможлиблює дану операцію:

```

27 ! insert into room_housing(id, patient_id, room_id, start_date, end_date) value (26,20,1, '2023-01-25', '2023-01-30');

[45000][1644] The room has the maximum number of patients. Choose other room.

```

Рисунок 7.1.2 – Виконання скрипту, що активує тригер

Тригер оновлює статус палати на зайняту, коли в неї додається пацієнт, а кількість вільних місць дорівнює одному.

```
drop trigger if exists update_room_availability;
```

```
delimiter $$
```

```
create trigger update_room_availability before insert on room_housing for each row
```

```
begin
```

```
    if NEW.room_id in (select distinct room_id
```

```
        from room_housing
```

```
        join room r on room_housing.room_id = r.id
```

```
        where (select check_free_beds(room_number)) = 1) then
```

```
        update room
```

```
            set availability = 0
```

```
        where NEW.room_id = room.id;
```

```
    end if;
```

```
end $$
```

```
delimiter ;
```

Приклад роботи:

Зі списку палат, оберемо палату з room\_id = 3:

	room_id
1	3
2	7
3	8

Рисунок 7.1.3 – Id палат, що мають одне вільне місце

У списку палат бачимо статус палат на даний момент (availability = 1):

	id	room_number	type	capacity	availability
1	1	11	Standard	3	0
2	2	12	VIP	1	0
3	3	13	Standard	4	1

Рисунок 7.1.4 – Інформація про палати

Додаємо пацієнта в дану палату:

```
✓ insert into room_housing(id, patient_id, room_id, start_date, end_date) value (25,19,3,'2023-01-25','2023-01-30');
```

Рисунок 7.1.5 – Скрипт додавання пацієнта в палату

Дивимось оновлений статус зайнятості палати:

	id	room_number	type	capacity	availability
1	1	11	Standard	3	0
2	2	12	VIP	1	0
3	3	13	Standard	4	0

Рисунок 7.1.6 – Інформація про палати

Тригер який унеможливилює видалення старих хвороб з історії хвороб

```
drop trigger if exists medical_history_update;
```

```
delimiter $$
```

```
create trigger medical_history_update before update on medical_card for each row
```

```
begin
```

```
set @history = concat(old.medical_history, '%');
```

```
if not new.medical_history like @history then
```

```
signal sqlstate '45000'
```

```
set message_text = 'Old diseases must remain';
```



```

end if;

end $$

delimiter ;

```

Приклад роботи:

Змінимо запис в історії хвороб у медичній карті під id = 3 на 'Asthma':

	id :	patient_id :	height :	weight :	blood_type :	allergies :	medical_history :	availability_of_insurance :
1	1	1	168	68.37	A+	Penicillin	Diabetes-Heart Disease	1
2	2	2	179	71.32	B-	<null>	<null>	1
3	3	3	170	127.17	AB+	Insulin	Asthma-Obesity	1

Рисунок 7.1.4 – Інформація з медичних карт пацієнтів

Отримаємо помилку, що унеможлиблює зміну історії хвороб, бо хвороба, що була раніше, зникла:

```

22  !  update medical_card
23      set medical_history = 'Asthma'
24      where patient_id = 3;

```

[45000][1644] Old diseases must remain

Рисунок 7.1.4 – Виконання скрипту, що активує тригер

## 7.2 Створення процедур

Процедура виводить усіх пацієнтів, що живуть в палаті, id якої передається параметром.

```
drop procedure if exists check_room_patients;
```

```
delimiter $$
```

```
create procedure check_room_patients(IN room_num int)
```

```
begin
```

```
    select patient_name, start_date, end_date
```

```
    from patient
```

```
    join room_housing on patient.id = room_housing.patient_id
```

```
    join room r on r.id = room_housing.room_id
```

```
    where room_number = room_num and curdate() between start_date and end_date-1;
```

```
end $$
```

```
delimiter ;
```

```
call check_room_patients(13);
```

	patient_name	start_date	end_date
1	Max Valeri	2023-01-15	2023-01-28
2	Jarrett McGrah	2023-01-05	2023-02-12
3	Curtice Weight	2023-01-09	2023-01-31

Рисунок 7.2.1 - Усі мешканці палати

Процедура показує всі прийоми, назначені на день, що передається параметром.

```
drop procedure if exists check_appointments;
```

```
delimiter $$
```

```
create procedure check_appointments(IN app_date date)
```

```
begin
```

```

select service_name, appointment_date, doctor_name, patient_name
from appointment
join service s on appointment.service_id = s.id
join doctor d on appointment.doctor_id = d.id
join patient p on appointment.patient_id = p.id
where (select date(appointment_date))= app_date;
end $$
delimiter ;
call check_appointments('2023-01-20');

```

	service_name	appointment_date	doctor_name	patient_name
1	Physical therapy	2023-01-20 11:15:00	Tom Cruise	Max Valeri
2	X-ray	2023-01-20 16:00:00	Hogan Goundsy	Daren Espy
3	X-ray	2023-01-20 10:00:00	Hogan Goundsy	Franz Eisig

Рисунок 7.2.2 – Усі прийоми на 20.01.2023

Процедура виписує пацієнта з палати, оновлює статус палати на вільний та змінює дату кінця терміну перебування в палаті пацієнта.

```

drop procedure if exists discharge_patient;
delimiter $$
create procedure discharge_patient(IN p_id int)
begin
    update room
        set availability = 1
        where id = (select room_id from room_housing where p_id = patient_id and
curdate() between start_date and end_date-1);

    update room_housing
        set end_date = curdate()

```

```

where p_id = patient_id and curdate() between start_date and end_date-1;
end $$
delimiter ;
call discharge_patient(19);

```

	patient_name	start_date	end_date
1	Max Valeri	2023-01-15	2023-01-28
2	Jarrett McGrah	2023-01-05	2023-02-12
3	Curtice Weight	2023-01-09	2023-01-31
4	Cordey Dulany	2023-01-25	2023-01-30

Рисунок 7.2.3 – Усі пацієнти, що записані в палаті 13 до виконання процедури

	patient_name	start_date	end_date
1	Max Valeri	2023-01-15	2023-01-28
2	Jarrett McGrah	2023-01-05	2023-02-12
3	Curtice Weight	2023-01-09	2023-01-31

Рисунок 7.2.4 – Усі пацієнти, що записані в палаті 13 після виконання процедури

	id	patient_id	room_id	start_date	end_date
1	25	19	3	2023-01-25	2023-01-25

Рисунок 7.2.5 – Змінена кінцева дата перебування пацієнта, що був виписаний

### 7.3 Створення функцій

Функція повертає кількість вільних місць у палаті, id якої передається параметром.

```
drop function if exists check_free_beds;
delimiter $$
create function check_free_beds(room_num int) returns int
deterministic
begin
declare beds_amount int;
select capacity - patients_amount into beds_amount
from (select room_number, capacity, count(*) as patients_amount
      from room_housing
      join room r on room_housing.room_id = r.id
      where curdate() between start_date and end_date-1
      group by room_id, capacity) as room_temp
where room_num = room_number;
return beds_amount;
end $$
delimiter ;
select check_free_beds(13);
```

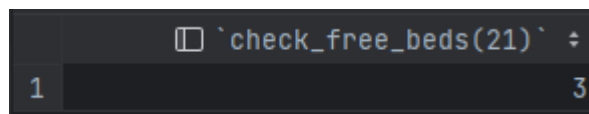


Рисунок 7.3.1 – Кількість вільних місць у палаті 3

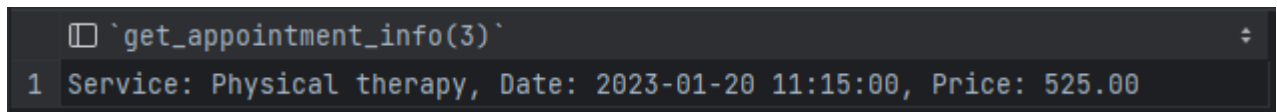
Функція повертає основну інформацію про останній прийом клієнта, id якого передано параметром.

```
drop function if exists get_appointment_info;
delimiter $$
create function get_appointment_info(client_id int) returns varchar(255)
deterministic
```

```

begin
    declare appointment_info varchar(255);
    select concat('Service: ',service_name,', Date: ', appointment_date,', Price: ',
coalesce(service.price, 0)) into appointment_info
    from appointment
    join service on appointment.service_id = service.id
    where appointment.patient_id = client_id
    order by appointment_date desc
    limit 1;
    return coalesce(appointment_info, 'This patient has no appointments');
end $$
delimiter ;
select get_appointment_info(3);

```



```

❏ `get_appointment_info(3)`
1 Service: Physical therapy, Date: 2023-01-20 11:15:00, Price: 525.00

```

Рисунок 7.3.2 – Основна інформація про останній прийом клієнта з id = 3

#### 7.4 Створення представлень

Представлення показує загальну інформацію про пацієнтів.

```
drop view if exists patient_data;
create or replace view patient_data as
select patient.id, patient_name, sex, height, weight, blood_type,
       allergies, medical_history
from patient
join medical_card mc on patient.id = mc.patient_id;
select * from patient_data;
```

	id	patient_name	sex	height	weight	blood_type	allergies	medical_history
1	1	Rianon Yanuk	F	168	68.37	A+	Penicillin	Diabetes-Heart Disease
2	2	Marena Allum	M	179	71.32	B-	<null>	<null>
3	3	Max Valeri	M	170	127.17	AB+	Insulin	Asthma-Obesity
4	4	Chloris Hanning	M	151	49.25	O+	<null>	<null>
5	5	Cordelia Addis	M	155	40.35	A+	Anesthesia	Asthma
6	6	Bax Crix	M	190	80.45	B+	<null>	Cancer
7	7	Vernice Crosdill	M	176	88.95	AB+	Latex	Diabetes
8	8	Margaretha Wittke	M	197	78.04	O+	<null>	Mental illnesses
9	9	Daren Espy	M	191	78.79	A-	Antibiotic	<null>
10	10	Catherine Undy	M	174	49.09	B-	<null>	Heart Disease
11	11	Rurik Idel	F	177	89.80	AB-	<null>	<null>
12	12	Gerrard D'Oyley	F	172	93.42	O-	Aspirin	Heart Disease-Obesity
13	13	Merwin Barraclough	M	176	67.79	A-	<null>	Asthma
14	14	Chryste Stibbs	M	190	75.30	B+	<null>	<null>
15	15	Jarrett McGrah	M	196	102.79	O+	<null>	<null>
16	16	Rudolf Thurling	M	162	50.95	A-	Antipsychotic	Diabetes
17	17	Curtice Weight	F	171	80.79	A+	<null>	Osteoarthritis
18	18	Mellisa Pude	F	197	133.63	B+	Statin	Obesity
19	19	Cordey Dulany	F	179	50.88	A-	<null>	Cancer
20	20	Franz Eisig	F	173	67.78	AB+	Diuretic	<null>

Рисунок 7.4.1 – Представлення з загальною інформацією про пацієнтів

Представлення показує загальну інформацію про прийоми.

```
drop view if exists appointment_data;
create or replace view appointment_data as
select appointment.id, appointment_date, doctor_name, speciality,
       patient_name, service_name, duration_minutes, price
from appointment
join patient p on appointment.patient_id = p.id
join doctor d on appointment.doctor_id = d.id
join service s on appointment.service_id = s.id
```

```
order by appointment_date;  
  
select * from appointment_data;
```

	id	appointment_date	doctor_name	speciality	patient_name	service_name	duration_minutes	price
1	12	2023-01-15 08:30:00	Hogan Goundsy	Radiologist	Rudolf Thurling	X-ray	20	350.50
2	10	2023-01-15 08:40:00	Mabel Cracoe	Dentist	Gerrard D'Oyley	Teeth cleaning	60	850.50
3	8	2023-01-16 12:00:00	Olivero Bridgnell	Surgeon	Margaretha Wittke	Surgery	180	15499.00
4	2	2023-01-16 13:30:00	Olenolin Critoph	Massage therapist	Marena Allum	Medical massage	60	660.00
5	11	2023-01-16 13:35:00	Kameko Redwin	Pathologist	Merwin Barraclough	Laboratory testing	15	325.20
6	6	2023-01-16 15:45:00	Allianora Mulankey	Therapist	Daren Espy	General check-up	55	2200.00
7	4	2023-01-17 09:25:00	Christophorus Jest	Vaccine provider	Chloris Hanning	Vaccination	10	150.80
8	13	2023-01-18 08:00:00	Mabel Cracoe	Dentist	Gerrard D'Oyley	Teeth cleaning	60	850.50
9	5	2023-01-18 08:00:00	Kameko Redwin	Pathologist	Bax Crix	Laboratory testing	15	325.20
10	1	2023-01-18 12:00:00	Mabel Cracoe	Dentist	Max Valeri	Teeth cleaning	60	850.50
11	9	2023-01-19 10:50:00	Tom Cruise	Physical Medicine and Rehabilitation	Catherine Undy	Physical therapy	45	525.00
12	14	2023-01-20 10:00:00	Hogan Goundsy	Radiologist	Franz Eisig	X-ray	20	350.50
13	3	2023-01-20 11:15:00	Tom Cruise	Physical Medicine and Rehabilitation	Max Valeri	Physical therapy	45	525.00
14	7	2023-01-20 16:00:00	Hogan Goundsy	Radiologist	Daren Espy	X-ray	20	350.50

Рисунок 7.4.2 – Представлення з загальною інформацією про прийоми



### 7.5 Створення різних запитів

Запит показує пацієнтів та інформацію про палати, де вони лежать.

```
select end_date, patient_name, room_number, type
from room_housing
join patient p on room_housing.patient_id = p.id
join room r on room_housing.room_id = r.id
where curdate() between start_date and end_date
order by room_number;
```

	end_date	patient_name	room_number	type
1	2023-02-01	Cordelia Addis	11	Standard
2	2023-01-30	Margaretha Wittke	11	Standard
3	2023-01-30	Chryste Stibbs	11	Standard
4	2023-02-03	Rianon Yanuk	12	VIP
5	2023-01-28	Max Valeri	13	Standard
6	2023-02-12	Jarrett McGrah	13	Standard
7	2023-01-31	Curtice Weight	13	Standard
8	2023-01-25	Cordey Dulany	13	Standard
9	2023-01-29	Bax Crix	14	Standard
10	2023-01-28	Daren Espy	14	Standard
11	2023-02-07	Marena Allum	15	Standard
12	2023-01-29	Chloris Hanning	15	Standard
13	2023-02-02	Vernice Crosdill	15	Standard
14	2023-02-06	Catherine Undy	21	Standard
15	2023-01-29	Rurik Idel	22	VIP
16	2023-01-30	Gerrard D'Oyley	23	Standard
17	2023-01-29	Merwin Barracclough	23	Standard
18	2023-01-30	Rudolf Thurling	25	Standard

Рисунок 7.5.1 – Пацієнти та палати, в яких вони лежать

Запит показує загальну суму оплати та її метод, здійсненої кожним пацієнтом.

```
select patient.id, patient_name, payment_method,
```

```

sum(amount) as amount_of_payments
from patient
left join appointment a on patient.id = a.patient_id
left join payment p on a.payment_id = p.id
group by patient.id, payment_method
order by amount_of_payments desc;

```

	id	patient_name	payment_method	amount_of_payments
1	8	Margaretha Wittke	credit card	15499.00
2	9	Daren Espy	debit card	5101.00
3	12	Gerrard D'Oyley	debit card	3402.00
4	3	Max Valeri	cash	2751.00
5	4	Chloris Hanning	credit card	2200.00
6	2	Marena Allum	cash	660.00
7	10	Catherine Undy	debit card	525.00
8	16	Rudolf Thurling	cash	350.50
9	20	Franz Eisig	cash	350.50
10	6	Bax Crix	cash	325.20
11	13	Merwin Barracclough	cash	325.20
12	1	Rianon Yanuk	<null>	<null>
13	5	Cordelia Addis	<null>	<null>
14	7	Vernice Crosdill	<null>	<null>
15	11	Rurik Idel	<null>	<null>
16	14	Chryste Stibbs	<null>	<null>
17	15	Jarrett McGrah	<null>	<null>
18	17	Curtice Weight	<null>	<null>
19	18	Mellisa Pude	<null>	<null>
20	19	Cordey Dulany	<null>	<null>

Рисунок 7.5.2 – Сума оплата кожного пацієнта

Запит показує всіх клієнтів та інформацію про медикаменти, що виписані їм.

```

select patient.id, patient_name, medication_name, dosage_in_mg, price
from patient
left join appointment a on patient.id = a.patient_id
left join medication_appointment ma on a.id = ma.appointment_id
left join medication m on ma.medication_id = m.id;

```

	id	patient_name	medication_name	dosage_in_mg	price
1	1	Rianon Yanuk	<null>	<null>	<null>
2	2	Marena Allum	Diclofenac	130	200.00
3	3	Max Valeri	Dentinox	160	100.00
4	3	Max Valeri	Hyaluronidase	260	1000.00
5	4	Chloris Hanning	Pfizer	240	800.00
6	5	Cordelia Addis	<null>	<null>	<null>
7	6	Bax Crix	<null>	<null>	<null>
8	7	Vernice Crosdill	<null>	<null>	<null>
9	8	Margaretha Wittke	Influgan	160	500.00
10	9	Daren Espy	Paracetamol	50	40.25
11	9	Daren Espy	<null>	<null>	<null>
12	10	Catherine Undy	Hyaluronidase	260	1000.00
13	11	Rurik Idel	<null>	<null>	<null>
14	12	Gerrard D'Oyley	Dentinox	160	100.00
15	12	Gerrard D'Oyley	Dentinox	160	100.00
16	13	Merwin Barracrough	<null>	<null>	<null>
17	14	Chryste Stibbs	<null>	<null>	<null>
18	15	Jarrett McGrah	<null>	<null>	<null>
19	16	Rudolf Thurling	<null>	<null>	<null>
20	17	Curtice Weight	<null>	<null>	<null>
21	18	Mellisa Pude	<null>	<null>	<null>
22	19	Cordey Dulany	<null>	<null>	<null>
23	20	Franz Eisig	<null>	<null>	<null>

Рисунок 7.5.3 – Пацієнти та їх медикаменти

Запит показує усіх пацієнтів, що мають медичне страхування.

```
select patient.id, patient_name
from patient
join medical_card on patient.id = medical_card.patient_id
where medical_card.availability_of_insurance = 1;
```

	id	patient_name
1	1	Rianon Yanuk
2	2	Marena Allum
3	3	Max Valeri
4	4	Chloris Hanning
5	6	Bax Crix
6	7	Vernice Crosdill
7	8	Margaretha Wittke
8	9	Daren Espy
9	11	Rurik Idel
10	12	Gerrard D'Oyley
11	13	Merwin Barracrough
12	17	Curtice Weight
13	18	Mellisa Pude
14	19	Cordey Dulany
15	20	Franz Eisig

Рисунок 7.5.4 – Усі пацієнти з медичним страхуванням

Запит показує усі прийоми, здійснені протягом останнього тижня.

```
select service_name, doctor_name, appointment_date
from appointment
join service on appointment.service_id = service.id
join doctor on appointment.doctor_id = doctor.id
where appointment_date between DATE_ADD(curdate(), INTERVAL -7 DAY) and
curdate()
order by appointment_date;
```

	service_name	doctor_name	appointment_date
1	Laboratory testing	Kameko Redwin	2023-01-18 08:00:00
2	Teeth cleaning	Mabel Cracoe	2023-01-18 08:00:00
3	Teeth cleaning	Mabel Cracoe	2023-01-18 12:00:00
4	Physical therapy	Tom Cruise	2023-01-19 10:50:00
5	X-ray	Hogan Goundsy	2023-01-20 10:00:00
6	Physical therapy	Tom Cruise	2023-01-20 11:15:00
7	X-ray	Hogan Goundsy	2023-01-20 16:00:00

Рисунок 7.5.5 – Усі прийоми за останній тиждень

Запит показує кількість прийомів, здійснених кожним лікарем, та загальну суму грошей, що були зароблені на цих прийомах.

```
select doctor.id, doctor_name, count(*) as appointments_amount,
       sum(price) as earned_money, service_name
from doctor
join appointment a on doctor.id = a.doctor_id
join service s on a.service_id = s.id
group by doctor.id, service_name
order by earned_money desc;
```

	id	doctor_name	appointments_amount	earned_money	service_name
1	8	Olivero Bridgnell	1	15499.00	Surgery
2	1	Mabel Cracoe	3	2551.50	Teeth cleaning
3	6	Allianora Mularkey	1	2200.00	General check-up
4	7	Hogan Goundsy	3	1051.50	X-ray
5	3	Tom Cruise	2	1050.00	Physical therapy
6	2	Olenolin Critoph	1	660.00	Medical massage
7	5	Kameko Redwin	2	650.40	Laboratory testing
8	4	Christophorus Jest	1	150.80	Vaccination

Рисунок 7.5.6 – Усі лікарі, їх кількість прийомів та зароблених грошей за них

Запит показує кількість пацієнтів в кожній палаті на даний момент.

```
select room_number, type, count(*) as patients_amount
from room_housing
join room r on room_housing.room_id = r.id
group by room_number, type;
```

	room_number	type	patients_amount
1	11	Standard	3
2	12	VIP	1
3	13	Standard	4
4	14	Standard	2
5	15	Standard	3
6	21	Standard	3
7	22	VIP	1
8	23	Standard	3
9	24	Standard	1
10	25	Standard	3

Рисунок 7.5.7 – Усі палати та кількість мешканців в даний момент

Запит показує медикаменти та кількість разів, що вони були виписані пацієнтам.

```
select medication_name, dosage_in_mg, count(*) as prescription
from medication
join medication_appointment ma on medication.id = ma.medication_id
group by medication_name, dosage_in_mg;
```

	medication_name	dosage_in_mg	prescription
1	Pfizer	240	1
2	Diclofenac	130	1
3	Paracetamol	50	1
4	Dentinox	160	3
5	Influgan	160	1
6	Hyaluronidase	260	2

Рисунок 7.5.8 – Ліки та кількість разів, що вони були виписані

Запит показує кількість медикаментів, які потрібно замовити (виписані пацієнтам, але нема в наявності).

```
select medication_name, count(*) as required_amount
from medication_appointment
join medication m on m.id = medication_appointment.medication_id
where availability = 0
group by medication_name;
```

	medication_name	required_amount
1	Diclofenac	1
2	Paracetamol	1

Рисунок 7.5.9 – Кількість невисначаючих медикаментів

Запит показує кількість унікальних пацієнтів у кожного лікаря.

```
select doctor.id, doctor_name, speciality,
       count(distinct patient_id) as unique_patients
from doctor
left join appointment a on doctor.id = a.doctor_id
group by doctor.id, speciality;
```

	id	doctor_name	speciality	unique_patients
1	1	Mabel Cracoe	Dentist	2
2	2	Olenolin Critoph	Massage therapist	1
3	3	Tom Cruise	Physical Medicine and Rehabilitation	2
4	4	Christophorus Jest	Vaccine provider	1
5	5	Kameko Redwin	Pathologist	2
6	6	Allianora Mularkey	Therapist	1
7	7	Hogan Goundsy	Radiologist	3
8	8	Olivero Bridgnell	Surgeon	1

Рисунок 7.5.10 – Лікарі та кількість їх пацієнтів

Запит показує пацієнтів, що не мають алергій та жодних записів у історії хвороб.

```
select patient_name
```

```
from patient
```

```
join medical_card mc on patient.id = mc.patient_id
```

```
where allergies is null and medical_history is null;
```

	patient_name
1	Marena Allum
2	Chloris Hanning
3	Rurik Idel
4	Chryste Stibbs
5	Jarrett McGrah

Рисунок 7.5.11 – Усі пацієнти без алергій та записів у історії хвороб

Запит показує усіх пацієнтів, що мають четверту позитивну групу крові та можуть бути універсальними реципієнтами.

```
select patient_name
```

```
from patient
```

```
join medical_card mc on patient.id = mc.patient_id
```

```
where blood_type = 'AB+';
```

	patient_name
1	Max Valeri
2	Vernice Crosdill
3	Franz Eisig

Рисунок 7.5.12 – Усі пацієнти з четвертою позитивною групою крові

Запит показує звіт по цінам до кожного прийому, включаючи вартість виписаних медикаментів та самої послуги

```
select appointment.id, coalesce(m.price, 0) as med_price, s.price as service_price,
       (select (sum(med_price + service_price))) as general_price
from appointment
left join service s on appointment.service_id = s.id
left join medication_appointment ma on appointment.id = ma.appointment_id
left join medication m on ma.medication_id = m.id
order by id;
```

	id	med_price	service_price	general_price
1	1	100.00	850.50	950.50
2	2	200.00	660.00	860.00
3	3	1000.00	525.00	1525.00
4	4	800.00	150.80	950.80
5	5	0.00	325.20	325.20
6	6	40.25	2200.00	2240.25
7	7	0.00	350.50	350.50
8	8	500.00	15499.00	15999.00
9	9	1000.00	525.00	1525.00
10	10	100.00	850.50	950.50
11	11	0.00	325.20	325.20
12	12	0.00	350.50	350.50
13	13	100.00	850.50	950.50
14	14	0.00	350.50	350.50

Рисунок 7.5.13 – Звіт про ціни до кожного прийому



Запит показує середню вартість одного прийому, мінімальну та максимальну ціну прийому та загальну вартість усіх прийомів, враховуючи медикаменти та послуги, що надаються.

```
select round(avg(coalesce(m.price, 0) + s.price),2) as avg_price,
       max(coalesce(m.price, 0) + s.price) as max_price,
       min(coalesce(m.price, 0) + s.price) as min_price,
       sum(coalesce(m.price, 0) + s.price) as sum_price
from appointment
left join service s on appointment.service_id = s.id
left join medication_appointment ma on appointment.id = ma.appointment_id
left join medication m on ma.medication_id = m.id;
```

	avg_price ÷	max_price ÷	min_price ÷	sum_price ÷
1	1975.25	15999.00	325.20	27653.45

Рисунок 7.5.14 – Звіт з середньою, максимальною, мінімальною вартістю прийому та суму вартостей усіх прийомів

Запит показує усіх пацієнтів, що мають критичний індекс маси тіла.

```
select patient_name, height, weight, weight/(height*height/10000) as BMI
from patient
join medical_card mc on patient.id = mc.patient_id
where not weight/(height*height/10000) between 16.5 and 30;
```

	patient_name ÷	height ÷	weight ÷	BMI ÷
1	Max Valeri	170	127.17	44.003460
2	Catherine Undy	174	49.09	16.214163
3	Gerrard D'Oyley	172	93.42	31.577880
4	Mellisa Pude	197	133.63	34.432735
5	Cordey Dulany	179	50.88	15.879654

Рисунок 7.5.15 – Пацієнти з критичним ІМТ

Запит показує кількість та загальну суму оплат кожного виду.

```
select payment_method, count(*) as payments_amount, sum(amount) as
payments_sum
from payment
group by payment_method;
```

	payment_method	payments_amount	payments_sum
1	cash	6	3386.90
2	credit card	2	17699.00
3	debit card	3	4776.50

Рисунок 7.5.16 – Звіт по методами оплати

Запит показує часові межі кожного прийому та їх тривалість.

```
select appointment.id, appointment_date, duration_minutes,
date_add(appointment_date, interval duration_minutes minute) as end_datetime
from appointment
join service s on appointment.service_id = s.id;
```

	id	appointment_date	duration_minutes	end_datetime
1	1	2023-01-18 12:00:00	60	2023-01-18 13:00:00
2	10	2023-01-15 08:40:00	60	2023-01-15 09:40:00
3	13	2023-01-18 08:00:00	60	2023-01-18 09:00:00
4	2	2023-01-16 13:30:00	60	2023-01-16 14:30:00
5	3	2023-01-20 11:15:00	45	2023-01-20 12:00:00
6	9	2023-01-19 10:50:00	45	2023-01-19 11:35:00
7	4	2023-01-17 09:25:00	10	2023-01-17 09:35:00
8	5	2023-01-18 08:00:00	15	2023-01-18 08:15:00
9	11	2023-01-16 13:35:00	15	2023-01-16 13:50:00
10	6	2023-01-16 15:45:00	55	2023-01-16 16:40:00
11	7	2023-01-20 16:00:00	20	2023-01-20 16:20:00
12	12	2023-01-15 08:30:00	20	2023-01-15 08:50:00
13	14	2023-01-20 10:00:00	20	2023-01-20 10:20:00
14	8	2023-01-16 12:00:00	180	2023-01-16 15:00:00

Рисунок 7.5.17 – Часові межі кожного прийому

Запит показує історію хвороб та наявність страхування у пацієнтів уразливого віку (60+).

```
select patient.id, patient_name, medical_history,
       date_format(from_days(datediff(now(),date_of_birth)), '%Y')+0 AS age,
       availability_of_insurance
from patient
join medical_card mc on patient.id = mc.patient_id
having age > 60;
```

	id	patient_name	medical_history	age	availability_of_insurance
1	2	Marena Allum	<null>	69	1
2	7	Vernice Crosdill	Diabetes	72	1
3	8	Margaretha Wittke	Mental illnesses	63	1
4	10	Catherine Undy	Heart Disease	78	0
5	16	Rudolf Thurling	Diabetes	62	0
6	17	Curtice Weight	Osteoarthritis	61	1
7	20	Franz Eisig	<null>	67	1

Рисунок 7.5.18 – Пацієнти уразливого віку

## 7.6 Створення індексів

Продемонструємо роботу індексів на прикладі нижче. Для початку виконаємо звичайний запит, з якого дістанемо дані про записи в палаті, зроблені 30.12.2022.

```
explain
```

```
select *
```

```
from room_housing
```

```
where start_date = '2022-12-30';
```

На рисунку нижче можна побачити, що загальний час виконання запиту становив 60 мс.

```
[HY000][1003] /* select#1 */ select 'hospital'.room_housing.id AS 'id', 'hospital'.room_housing.patient_id AS 'patient_id', 'hospital'.room_housing.room_id AS 'room_id', 'hospital'.room_housing.start_date AS 'start_date'
1 row retrieved starting from 1 in 60 ms (execution: 7 ms, fetching: 53 ms)
```

Рисунок 7.6.1 – Аналіз виконання запиту

Далі створимо індекс та повторимо той самий запит ще раз.

```
create index test
```

```
on service(price);
```

Отримаємо результат в 42 мс, що значно швидше, ніж попереднє виконання цього ж запиту.

```
[HY000][1003] /* select#1 */ select 'hospital'.room_housing.id AS 'id', 'hospital'.room_housing.patient_id AS 'patient_id', 'hospital'.room_housing.room_id AS 'room_id', 'hospital'.room_housing.start_date AS 'start_date'
1 row retrieved starting from 1 in 42 ms (execution: 8 ms, fetching: 34 ms)
```

Рисунок 7.6.2 – Аналіз виконання запиту

З отриманих результатів можна наглядно побачити, як індекси оптимізують роботу запитів та значно збільшують швидкість їх виконання.

## ВИСНОВОК

Щоб розробити комплексну базу даних лікувального закладу, спочатку був розроблений комплексний план, було визначено ключові суб'єкти та їх відносини, а також встановлено основні бізнес-правила та положення. Завдяки цим зусиллям було успішно створено та впроваджено повнофункціональну базу даних. Загалом було створено 10 окремих таблиць, які пов'язані разом за допомогою зовнішніх ключів. Крім того, було продемонстровано досвід у створенні та застосуванні різних елементів бази даних, таких як тригери, функції, процедури та складні запити, які оптимізовано за допомогою індексів.

## ПЕРЕЛІК ПОСИЛАНЬ

1. <https://dev.mysql.com/doc/>
2. <https://www.jetbrains.com/help/datagrip/meet-the-product.html>
3. <https://www.w3schools.com/mysql/default.asp>