МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Кафедра наноелектроніки та модифікації поверхні

БАКАЛАВРСЬКА РОБОТА

зі спеціальності 153 – «Мікро- та наносистемна техніка»

на тему:

«Керування базою даних медичного закладу засобами SQLite»

Зиморой Віталій Віталійович

Студент групи ФЕ-81
_____ В.В. Зиморой
«___»_____2022 р.

Науковий керівник
_____ доц. О.В. Ющенко
«___»_____2022 р.

Суми 2022

РЕФЕРАТ

Робота складається з вступу, дослідження різного матеріалу, де розібрана мова програмування SQL, також її можливості для формування різних запитів і таблиць, сама програма в якій велася розробка під назвою "Database Browser for SQLite", швидкий розбір ER-діаграми, створення таблиць і запитів в базі даних за допомогою SQL мови, правила техніки безпеки та охорони праці при роботі за комп’ютером, висновків та списку використаних джерел.

Звіт містить 50 сторінок, 26 рисунків та 31 літературних джерел.

В цій роботі я використовував базу даних офтальмологічної клініки “ЦЕНТР ЗОРУ”. Приватна офтальмологічна клініка надає платні послуги з діагностики, корекції та лікування зору. База даних клініки містить велику кількість інформації яку й потрібно обробити.

Мета роботи – показати можливості, за яких стає можливим швидко оброблювати дані в даному випадку медичного закладу, відмовитися від паперової документації і оперативно знаходити потрібні записи, документи та інформацію.

КЛЮЧОВІ СЛОВА: МОВА SQL, БАЗА ДАНИХ, МЕДИЧНИЙ ЗАКЛАД, ОБРОБКА ІНФОРМАЦІЇ, DATABASE BROWSER FOR SQLITE.

ЗМІСТ

ВСТУП

В сучасному світі майже вся діяльність підприємців переходить на електронні данні, для зберігання інформації. В цьому способі зберігання інформації є безліч переваг над паперовим способом документації, який вже давно у минулому. Серед них можна назвати швидкість в роботі з документами, так як пошук потрібного документа, його копіювання та друк займають секунди. Також надійність цього способу не викликає сумнівів, так як цифрові документи не псуються з часом і можуть зберігатись нескінченну кількість часу. Серед переваг можна відмітити дистанційний доступ, який можна отримувати на будь-якому пристрої маючи відповідний доступ та навички.

У цій роботі я привів приклад створення бази даних у програмі DB Browser for SQLite. Ця програма повністю доступна, тобто в інтернеті вона у вільному доступі і може бути легко використана в особистих цілях. База даних яку створили може зберігатися на персональному комп’ютері або викинута в загальний доступ. Програма DB Browser for SQLite дозволяє створювати таблиці в інтерактивному режимі також дозволяє пов'язувати таблиці. При формуванні запитів користувач має знати мову SQL. Було створено базу даних клініки зору, де міститься інформація про пацієнтів, лікарів, обслуговування, відвідування тощо.

У першому розділі ми розглянемо мову запитів SQL та програму DB Browser for SQLite. У другому розділі ми розглянемо створення і редагування таблиць, що входять до цієї бази даних. І в третьому розглянемо формування і виконання SQL запитів.

РОЗДІЛ 1 SQL МОВА

1.1 Для чого потрібен SQL

SQL (Structured Query Language) — це мова запитів структурованого типу, яка створена для того, щоб отримувати з бази даних необхідну інформацію якщо простими словами то це звичайна мова запитів для роботи з реляційними базами даних.

Спочатку була реляційна алгебра вже потім з’явилася мова SQL, а розроблення прототипу лягло на плечі IBM Research наприкінці 1970 років. Після цього вже був реалізований перший прототип системи управління реляційною базою даних IBM System R. Пізніше ця мова почала використовуватися в багатьох системах управління базами даних, і завдяки своїй популярності ця система крок за кроком стала неофіційним але стандартом для мов управління даними «де-факто» в системах маніпулювання реляційними базами даних.

Якщо описати принцип роботи SQL простими словами, користувач формує запит і відправляє його в базу даних. Відразу після того, як процес обробить цю інформацію, він «розуміє» точні потреби користувача та надсилає відповідь. Дані зберігаються у формі таблиці, яка структурована та впорядкована в рядки та стовпці, щоб полегшити маніпулювання. Такий спосіб зберігання інформації називається реляційною базою даних. Назва вказує на те, що об'єкти в таких базах даних пов'язані певним зв'язком.

Наприклад, маємо базу, в якій зібрано інформацію про всі медичні заклади в місті: назви, лікарі, ціни, графік роботи та інше. Під час аналізу по кількості хворих я вирішив з'ясувати, скільки хворих приймають таблетки з вітамінами та який лікар їх обслуговує. Щоб отримати подібний список, достатньо всього написати потрібний запит в SQL.

SQL мова з'явилася достатньо давно, але досі стійко тримається в популярності що робить її однією з затребуваних мов в подібній сфері діяльності.

Як вже відомо SQL використовується в реляційних базах даних. Серед таких можна виділити доволі популярні: SQLite, MySQL, PostgreSQL, SQL Server Microsoft та багато інших.

Як уже можна зрозуміти мовою SQL написати програму або сайт не вийде, але при цьому внутрішня робота сайту неможлива без запитів. Пошук інформації в пошуковій системі яких зараз в інтернеті вистачає таких як Google або Bing - це як раз саме модель використання SQL. Користувач задає параметри, які його цікавлять, та надсилає запит, потім відбувається обчислювальні процеси і в пошуковій системі з'являються результати, що відповідають саме цьому запиту.

З SQL можна працювати в різних сферах, наприклад: Аналітики де знання SQL допомагає працівникам не залежати від програмістів, а самостійно отримувати та обробляти дані.

Також тестувальники, за допомогою SQL вони можуть самостійно проектувати бази для швидкої та надійної роботи з даними, за допомогою цієї мови обробляти и покращувати сайти та програми з якими працівники працюють.

Різні програми потребують різних навичок в застосуванні SQL, але також існують і програми в яких знання мови SQL потрібні мінімальні що допомагає працювати в такому середовищі навіть звичайній не досвідченій людині.

Якщо брати на зразок сайти в інтернеті, в яких публікуються різного роду документи і які знаходяться у вільному доступі то юзер який їх знаходить має можливість роздруковувати ці папери на принтері. Друкування таких документів або любої іншої інформації вимагає від користувача установку спеціального контролеру друку на свій комп’ютер. У контролер входять апаратне та програмне забезпечення, що дозволяє йому не тільки обробляти дані для друку, але й завдяки якому користувач має можливість змінювати калібрування, керувати кольором та інше маючи інтерфейс користувача. Розглядаючи таку ситуацію можна сказати що тестувальник виконуючи свою роботу повинен зробити так, щоб друкувати документи стало можливо з різних браузерів, операційних систем, принтерів і з усім цим система ще повинна перевіряти якість друку файлу або інформації яка потрібна користувачу. Для здійснення такої роботи не важливі навички SQL але досвід цієї мови знадобиться, щоб здійснювати перевірку тих самих тестових даних у базі, видалення, оновлення та інші процеси.

Бекенд-тестування (програмна частина, що відповідає за функціонування внутрішньої частини сервісу) потрібно добре знати SQL-запити. В тестуванні такого роду для з’ясування повного і точного функціонування інтерфейсу можна взяти за приклад вихідну інформацію бази даних і інтерфейсу вводячи туди однакові значення. Кожен раз коли в системі відбувається зміна вхідних значень, адміністратор який в цей момент працює з базою даних повинен дати тестувальникам громіздкі запити використовуючи команду select. Крім цього щоб проводити перевірку тих самих тестових в SQL необхідно використовувати різні типи операторів.

Ще одна сфера де можна використовувати SQL це керівники та менеджери. В цій роботі SQL дозволить спеціалістам на відповідних посадах самостійно звертатися до бази і контролювати роботу компанії , також всі дані і контроль ними можна здійснювати в реальному часі отримуючи запитом наприклад стан справ якогось відділу.

1.2 Що таке база даних у SQL

SQL запити звертаються до даних як таблиці, тобто до реляційних баз даних. Спрощений варіант такої бази – це таблиці Excel, у яких інформація також упорядкована у стовпці та рядки.

Основні поняття реляційної моделі це відношення (таблиця яка складається зі стовпців та рядків), атрибут (стовпець у таблиці який містить один конкретний параметр: назву, тип, дату, тощо), домен (допустимі значення для атрибута. У стовпці «first_NAME» або «last_NAME» значення повинні бути набором літер), кортеж (табличний рядок із порядковим номером або запис), значення (елемент таблиці, який перебуває на перетині стовпців та рядків), ключ (важливий стовпець у таблиці за допомогою якого відбувається пов'язування таблиць в базі даних між собою).

В будь-якій базі даних ключі бувають різних типів і кожен ключ виконує свою функцію.

Первинний ключ — це ідентифікатор, який може бути індексом або артиклем.

Потенційний ключ — це унікальне значення, яке може бути ідентифікатором.

Зовнішній ключ – це ключ, що використовується для об'єднання двох таблиць, але є головне правило що кожне значення зовнішнього ключа відповідає первинному ключу в іншій таблиці.

На рисунку 1.1 можна побачити приклад спрощеного варіанту реляційної бази даних.

	A	B	C	D	E
1	p_ID	p_first_name	p_last_name	p_date_of_	p_phone_number
2	p1	Raymond	Brown	1977-02-01	+38(063)610-95-60
3	p2	Darlene	Williams	2018-03-23	+38(067)681-57-98
4	p3	Rodney	Vargas	2006-12-07	+38(068)091-34-23
5	p4	Patricia	Turner	1996-10-27	+38(066)532-14-40
6	p5	Robin	Jenkins	1958-10-09	+38(063)609-17-96
7	p6	Robert	Brock	1984-08-16	+38(067)889-57-85
8	p7	Roland	Williams	1955-04-11	+38(098)771-19-49
9	p8	Carrie	Hill	2011-07-08	+38(097)816-26-73
10	p9	Richard	Adams	1964-03-29	+38(050)823-52-99
11	p10	Kevin	Johnson	1989-12-14	+38(050)886-56-23
12	p11	James	Scott	1987-12-28	+38(099)955-90-44
13	p12	James	Stevens	2005-01-02	+38(095)601-27-45
14	p13	Jeanne	Green	1996-08-07	+38(066)681-98-00
15	p14	Michael	Christensen	1967-10-31	+38(063)962-68-60
16	p15	Edward	Rios	1970-06-07	+38(050)167-11-19
17	p16	Kay	Brown	1975-01-09	+38(050)210-05-66
18	p17	Erin	Jones	1964-05-30	+38(068)471-38-10
19	p18	Betty	Smith	1987-08-28	+38(097)823-68-35
20	p19	Mary	Shaw	2002-05-22	+38(050)021-14-19
21	p20	Victor	Hayes	1958-10-23	+38(066)208-86-40

Рисунок 1.1 - Спрощений варіант реляційної бази даних

1.3 SQL-оператори

Працювати з даними в SQL допомагають оператори.

Оператори - це слово або символи, які використовуються в SQL для виконання конкретної операції. Наприклад, для порівняння або вибору безлічі за конкретним

параметром або для опису умов у SQL виразі та обслуговування групи SQL виразів, тощо. Якщо нам потрібно взяти всіх пацієнтів і відсортувати тих, у яких є певна конкретна хвороба, потрібно використовувати оператор SELECT, цей оператор означає вибір даних відповідно до заданої умови.

SQL оператори поділяються на конкретні групи відповідно до завдань, які вони можуть вирішувати.

У мові запитів SQL існує чотири типи операторів:

Оператори опису даних: CREATE, DROP, ALTER та ін. Також можна побачити таке позначення цього оператора як DDL або Data Definition Language цей оператор працює з об'єктами або як вже можна зрозуміти з таблицями бази даних. Якщо певну базу потрібно оновити таблицею з новими даними або, навпаки, видалити одну з таблиць з даними які виявилися не дійсними, в цьому випадку потрібно використовувати набір операторів опису.

Наступний оператор - це оператор маніпуляції даними: INSERT, DELETE, SELECT, UPDATE та ін. Так як і з попереднім оператором його ще називають DML або Data Manipulation Language. Цей оператор працює також із таблицями але якщо конкретніше то із вмістом таблиць це рядки, значення, атрибути, тощо. За допомогою цього оператора можна вносити зміни у конкретне значення. Наприклад, замінити поле в колонці «last_NAME» у рядку з даними лікарів або пацієнтів цієї лікарні після того, як у них за різних обставин змінилося прізвище. Або видалити рядок із даними звільненого співробітника або виписаного пацієнта.

Ще один оператор із чотирьох який може надавати права доступу в базі даних: GRANT / REVOKE, LOCK / UNLOCK, SET LOCK MODE та ін. Друга назва цього оператора DCL або Data Control Language. Робота цього оператора полягає в доступі до бази даних тобто, хто з користувачів може надсилати запити до цієї бази, змінювати певні об'єкти та їхні значення. Приклад такого оператора можна побачити коли робітник

може відкликати доступ у співробітника, який перейшов до іншого відділу, або навпаки відкрити доступ до бази для нового робітника або розробника.

Останні оператори які є в мові SQL – це оператори керування транзакціями: BEGIN TRANSACTION, COMMIT TRANSACTION, ROLLBACK TRANSACTION, SAVE TRANSACTION та ін. Що це взагалі транзакції в середовищі SQL – це набір команд, які виконуються по черзі. Якщо система виконує всі задані команди і в результаті вини виконані то транзакція вважається успішно виконаною, а якщо навпаки десь сталася помилка то транзакція робить відкат назад, скасовуючи усі виконані команди. Транзакція SQL починається при виконанні процесу деякого модуля або навпаки з прямого виклику оператора SQL. Транзакція завершується коли виконується оператори COMMIT або ROLLBACK. Якщо SQL-транзакція завершується успішним виконанням оператора COMMIT, то всі зміни які зроблені цією транзакцією над даними стають постійно збереженими. Якщо транзакція завершується оператором ROLLBACK або якщо виконання оператора COMMIT виявляється провальним, всі зміни, зроблені транзакцією над даними відкочуються.

Дуже гарний приклад такої транзакції – це оплата онлайн із певного пристрою, коли банк робить запит на введення суми та інформації одержувача, перевірити та підтвердити операцію і в кінці ввести одноразовий код. На кожному з цих етапів оплату можна скасувати та транзакція відкотиться назад до самого початку.

1.4 Системи для роботи в SQL

Сама по собі база даних неспроможна виконувати різні операції, а система управління базами даних може. Ця система спроможна створювати нові таблиці, видаляти дані, налаштовувати ключі і обробляти запити.

Існують різні види таких систем, їх розроблюють різні компанії, на зразок Google або Microsoft. Через різні потреби в роботі з базами даних з'явилися різні види мови SQL їх ще називають SQL-діалекти. Кожна система управління базами даних має діалект, тобто свої особливості які в результаті не зможуть працювати в іншій системі.

Система управління базами даних може бути як власністю якоїсь компанії для користування якої потрібно домовлятися і купувати повні версії програми так і мати відкритий код тобто доступною для кожного користувача. Системи керування з відкритим кодом можна безкоштовно використовувати в особистих цілях як для користування так і для вдосконалення процесу роботи з системою.

В інтернеті можна знайти багато різних систем для управління але є системи які користуються великою популярністю і зручністю в користуванні.

PostgreSQL система яка може обробляти дані як абстрактні об'єкти. Кожен об'єкт, на відміну від простих табличних значень, може мати власні характеристики та унікальні методи взаємодії з іншими об'єктами.

MySQL - проста та функціональна система, яка працює з сайтами та веб-додатками. Найчастіше використовується в системах управління інформацією сайтів, у планувальниках, чатах та форумах. MySQL вважається одним з найбезпечніших та швидкісних програм.

SQLite - це полегшена вбудована версія системи управління базами даних. Ця система вважається швидкою та потужною. SQLite використовується для обробки запитів на різних сайтах, а також в мобільних додатках та іграх але не в багатокористувацькому режимі. Перевага такої системи — файлова структура, тобто база SQLite складається з одного файлу і її дуже легко переносити. Саме цю систему я використовую в своїй дипломній роботі.

Oracle Database – це система управління базами даних від компанії Oracle. Це також дуже популярна система управління базами даних, цією системою також неодноразово користувалися великі компанії. За своїми можливостями та функціональністю Oracle Database є однією з найпотужніших систем, і вартість її повнофункціональної версії дуже висока.

1.5 Запити в SQL

Основна функція SQL - це отримання даних із системи управління базами даних. Для того щоб написати різні запити до бази даних потрібно використовувати оператор SELECT . Цей оператор дозволяє виконувати складні перевірки та обробку даних. В SQL є загальна схема шляху від користувача до даних які йому потрібні в базі.

Користувач - клієнт - запит - система управління - база даних - таблиця з базами даних.

Як вже можна зрозуміти всі дані для роботи з SQL зберігаються у таблицях. На шляху від користувача до таблиці з даними ще є клієнт, система управління і база даних.

Клієнт – спосіб введення запиту. У випадку з пошуковою системою, наприклад, клієнтом буде пошуковий рядок в браузері, куди потрібно вводити вже сформульований запит для системи.

Система управління базами даних – це програми за допомогою яких можна керувати даними. Ця система допомагає таблицям зрозуміти чого хоче користувач, а користувачеві допомагає зрозуміти що відповідають таблиці.

База даних в даному випадку це система зберігання таблиць, де вони пов'язані між собою. База даних сама по собі не вміє маніпулювати інформацією - це просто середовище де вони можуть зберігатися.

В SQL є різні види запитів найчастіше ці запити поділяються на:

1. Запити, призначені для роботи зі структурою даних.
2. Запити, що використовуються безпосередньо в роботі з даними.
3. Запити, ціль яких надання або скасування прав доступу до бази даних.

Своєю чергою, кожен із видів SQL-запитів поділяється на різні типи:

1. Команди, які працюють із структурою бази даних. До них відносяться: CREATE, ALTER (модифікувати), DROP (видалити).
2. Команди, що мають справи з даними. До найбільш популярних запитів відносяться: SELECT, INSERT, UPDATE, DELETE, MERGE.

3. Команди, які працюють із правами доступу до системи. У їх перелік входять: GRANT (дозвіл для проведення певних операцій), REVOKE (видалення виданого дозволу), DENY (встановлення заборони, яка має пріоритет над усіма дозволами).

При складанні SQL-запиту до роботи з базами даних у системі управління вводяться різні параметри відбору:

1. Назва таблиці, з якої необхідно вилучити дані.
2. Поля, значення яких потрібно повернути до початкових після внесення змін до бази даних.
3. Зв'язок між таблицями.
4. Умови вибірки;
5. Допоміжні критерії відбору.

Для складання вибірки по пацієнтам, що відвідують лікарню найчастіше, для роботи з базами даних можна побудувати такий запит, що має вигляд:
select col1, col2, col3 де перерахуються колонки, які потрібно відобразити, from table (ім'я таблиці) where (наступний фільтр) clause (критерій відбору).

Завдяки простоті в використанні систем SQL, модифікувати запити для вирішення конкретних завдань можна швидко і без проблем.

1.6 DB Browser for SQL для робот з базами даних

Браузер баз даних для SQLite це візуальний інструмент який повністю доступний для кожного користувача. DB Browser це програма з відкритим вихідним кодом для створення, видалення, розробки та редагування файлів баз даних яка сумісна з SQLite. Програма була раніше відома як "SQLite Database Browser" та "Database Browser for SQLite".

Цей інструмент створений для користувачів та розробників, яким потрібно створювати і виконувати різні процеси з базами даних, шукати, редагувати дані в них.

Він використовує звичний табличний інтерфейс, тому для користування цією програмою користувачу не потрібно вивчати надскладні SQL-команди.

На рисунку 1.2 можна побачити інтерфейс і завантажену базу даних з таблицями і даними до них.

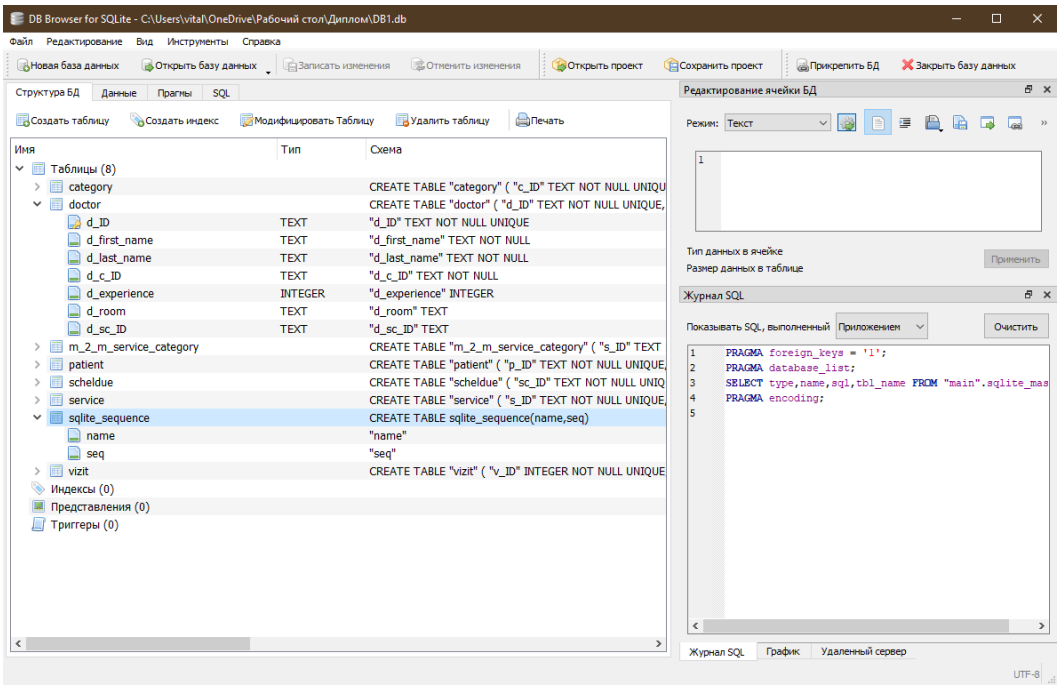


Рисунок 1.2 – База даних в «Database Browser for SQLite»

На рисунку 1.3 можна побачити засоби управління SQLite.

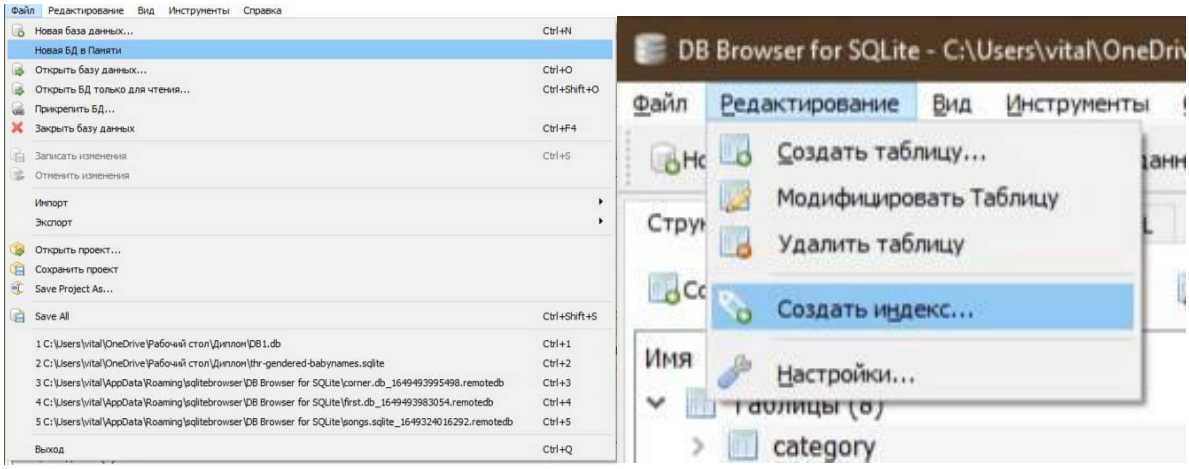


Рисунок 1.3 - Засоби управління SQLite

Засоби управління допомагають користувачеві виконувати різні процеси в базі даних:

- Створити файли бази даних.
- Формувати, обирати, змінювати та видаляти таблиці.
- Створювати та видаляти індекси.
- Робити будь які маніпуляції із записами.
- Імпортувати та експортувати текст.
- Імпортувати та експортувати таблиці.
- Імпортувати та експортувати бази даних.
- Писати SQL-запити та перевіряти результати в одній вкладці.
- Вивчати журнал SQL команд які доступні для цієї програми.

В більш нових версіях цього додатку з’являється режим WAL. Режим WAL – це техніка для забезпечення атомарності та стійкості базі даних тобто, процес або виконується повністю або зовсім не виконується. За допомогою цієї техніки можна використовувати одну і ту ж базу кількома додатками, як для читання, так і для записів.

SQLite також має і свої фішки на кшталт файлу в форматі sqlite3, і з його допомогою реалізовуються функції вже основної бібліотеки. Завдяки архітектурі так званого движка можливо використовувати DB Browser в різних вбудованих системах і в масштабних розмірах на окремих машинах з великими даними.

Формат файлу після збереження бази даних доволі цікавий у використанні так як один і той же формат може використовуватися в різних операційних системах або програмах, роблячи цей тип файлу крос-платформним.

При зберіганні бази даних в цьому застосунку є така функція як збереження в пам’яті і при цьому запис на носій інформації або на диск комп’ютера не відбувається. Зазвичай цієї функцією користуються для консольної утиліти sqlite3 при умові якщо файл не має ім’я.

SQLite являється основною програмою з якою я працював пишучи дипломну роботу.

РОЗДІЛ 2. ER-ДІАГРАМА, СТВОРЕННЯ ТАБЛИЦЬ І ЗАПОВНЕННЯ ЇХ ДАНИМИ В SQLITE

2.1 ER-діаграма

Щоб працювати з базою даних її спочатку треба створити і для цього складного процесу потрібно розробити ER-діаграму.

ER-діаграма(рис. 2.1) – це різновид схеми блочної структури, де можна побачити як різні об’єкти можуть бути пов’язані між собою всередині всієї цієї системи. ER-діаграми найчастіше використовуються для проектування, роботи і правок в реляційних базах даних.

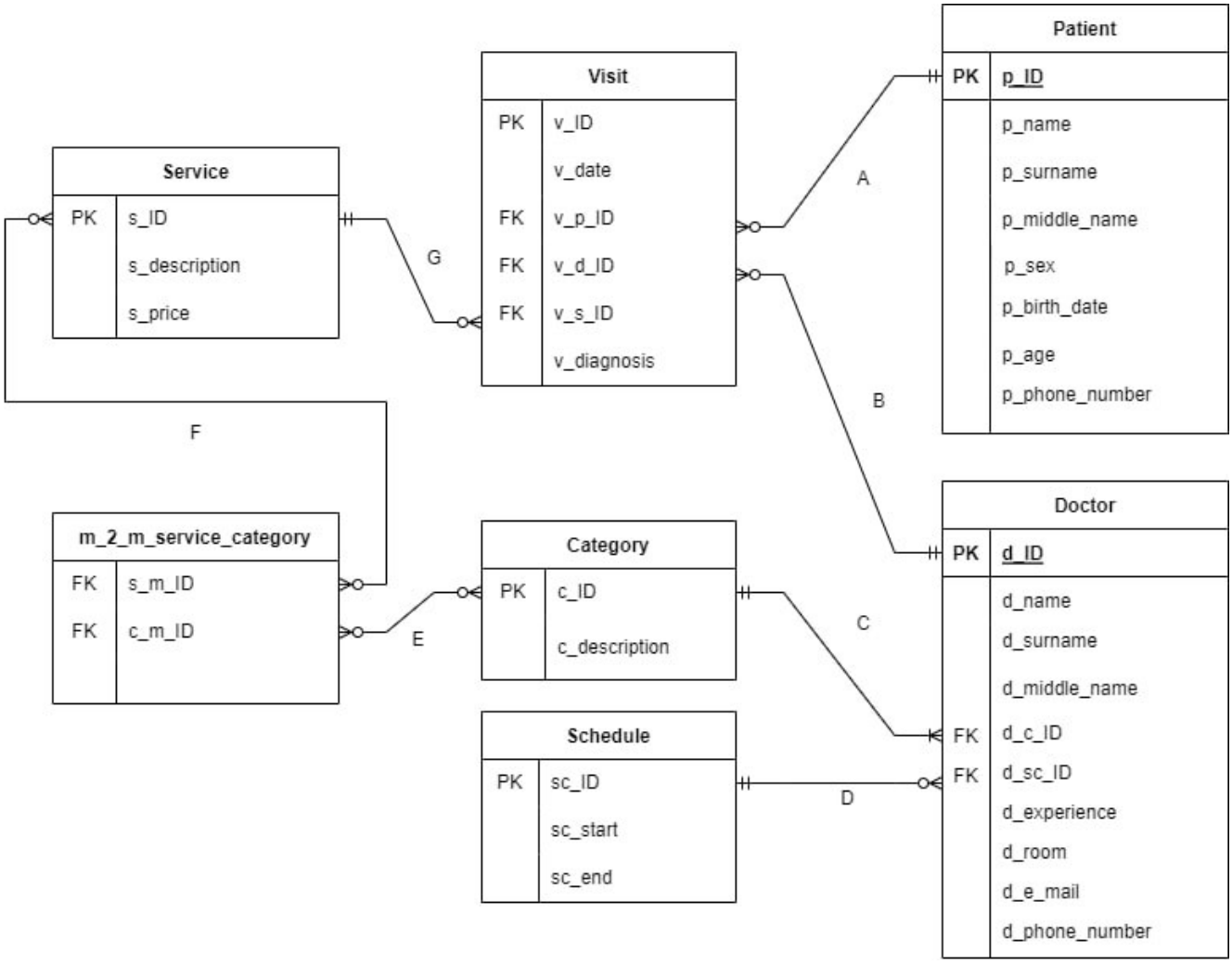


Рисунок 2.1 - ER-діаграма

В цій роботі я використовував базу даних офтальмологічної клініки “ЦЕНТР ЗОРУ”. Приватна офтальмологічна клініка надає платні послуги з діагностики, корекції та лікування зору. База даних клініки містить велику кількість інформації яку потрібно обробити.

Аналізуючи ER-діаграму можна побачити таблиці які при цьому ще й зв’язані між собою.

В блок-схемі знаходиться 7 різних таблиць з різними даними:

1. Таблиця “Patient” містить ім’я, прізвище та по-батькові пацієнта, його дату народження, кількість повних років, стать і його номер телефону.
2. Таблиця “Doctor” містить ім’я, прізвище та по-батькові лікаря, його досвід роботи в цій галузі, кабінет в якому він знаходиться, номер телефона і його електронна адреса.
3. Таблиця “Visit” містить дату візиту, діагноз який поставив лікар а також за допомогою зовнішніх ключів сюди притягуються пацієнти, лікарі і сервіси.
4. Таблиця “Category” показує категорію відповідного лікаря за допомогою зовнішнього ключа в таблиці “Doctor”.
5. Таблиця “Schedule” показує розклад роботи відповідного лікаря за допомогою зовнішнього ключа в таблиці “Doctor”.
6. Таблиця “Service” показує послуги які можуть надавати лікарі та ціну послуги.
7. Таблиця “M_2_m_service_category” показує зв’язок між послугами, що надаються клінікою, і категорією лікаря. Ця таблиця містить зовнішні ключі що поєднує таблиці “Service” і “Category” так як з’єднати “Service” і “Category” навпростець не можна в даному випадку.

2.2 Створення таблиць в SQlite

Для створення таблиць в програмі «Database Browser for SQLite» потрібно зайти в неї и створити там базу даних в форматі .db, після того як ця база відкрилася потрібно зайти і натиснути на поле “Створити таблицю”.

Щоб створити коректну базу даних потрібно користуватися вже створеною раніше ER-діаграмою, починаючи з таблиці “Patient” потрібно відтворити поля та властивості цих полів які вже були показані на діаграмі.

Зробивши всі вище сказані дії в результаті отримуємо таблицю “Patient” (рис. 2.2) яка містить ім’я, прізвище та по-батькові пацієнта, його дату народження, кількість повних років, стать і його номер телефону.

```
1   CREATE TABLE "Patient" (
2       "p_ID"  INTEGER NOT NULL UNIQUE,
3       "p_name"    TEXT NOT NULL,
4       "p_surname" TEXT NOT NULL,
5       "p_middle_name" TEXT,
6       "p_sex" TEXT NOT NULL,
7       "p_age" NUMERIC NOT NULL,
8       "p_birth_date"  NUMERIC NOT NULL,
9       "p_phone_number"    NUMERIC,
10      PRIMARY KEY("p_ID")
11  );
```

Рисунок 2.2 - Створення таблиці "Patient"

Після написання потрібних полів необхідно вказати який тип даних потрібен для вибраного поля.

Кожне значення, що зберігається в базі даних SQLite, має один із не багатьох класів зберігання. TEXT – це коли стовпці зберігають всі дані з використанням класів зберігання NULL, TEXT або BLOB. NUMERIC – це стовпець який може містити значення, використовуючи всі п’ять класів зберігання, зазвичай його обирають коли вводять в цей стовпець числові дані. INTEGER має такі ж властивості як стовпець з NUMERIC, за винятком виразу CAST який використовується для перетворення значення з типу даних в інший тип даних. REAL також як стовпець NUMERIC, за винятком того, що він виводить цілі значення у вигляд з плаваючою комою.

Як можна зауважити в таблиці є «p_ID» в якому тип INTEGER і який має властивість NOT NULL тобто він не може бути пустим і UNIQUE який показує що всі

значення в стовпці або групі стовпців відрізняються один від одного або є унікальними. Далі йде поле «p_name» в якому треба записати ім’я тобто ставимо тип TEXT і NOT NULL бо пацієнт не може бути без ім’я, «p_surname» з такою ж аналогією TEXT і NOT NULL, «p_middle_name» і «p_sex» мають такі ж типи, «p_age» буде з типом NUMERIC так як вік пацієнта це числові дані і NOT NULL, «p_birth_date» NUMERIC і NOT NULL з такою ж аналогією як і в «p_age», «p_phone_number» тип NUMERIC і NOT NULL тут не потрібен так як можуть бути різні ситуації коли номеру телефону не може бути у пацієнта. PRIMARY KEY(«p_ID» в даному випадку) – це первинний ключ і цей параметр встановлюється для однозначної ідентифікації того чи іншого запису таблиці. Він може бути тільки один і не може мати значення NULL. В подальшому первинний ключ зв’язується з зовнішнім ключем тим самим створюючи зв’язок між таблицями для обробки інформації.

Наступна таблиця “Visit” містить дату візиту, діагноз який поставив лікар а також за допомогою зовнішніх ключів сюди притягуються пацієнти, лікарі і сервіси, це можна побачити на рисунку 2.3

```
CREATE TABLE "Visit" (
    "v_ID"  INTEGER NOT NULL UNIQUE,
    "v_date"    NUMERIC NOT NULL,
    "v_p_ID"    INTEGER,
    "v_d_ID"    INTEGER,
    "v_s_ID"    INTEGER,
    "v_diagnosis"   TEXT,
    PRIMARY KEY("v_ID"),
    FOREIGN KEY("v_p_ID") REFERENCES "Patient"("p_ID"),
    FOREIGN KEY("v_d_ID") REFERENCES "Doctor"("d_ID"),
    FOREIGN KEY("v_s_ID") REFERENCES "Service"("s_ID")
);
```

Рисунок 2.3 - Створення таблиці "Visit"

В таблиці “Visit” є 3 зовнішніх ключа які зв’язуються з первинними ключами інших таблиць

FOREIGN KEY("v_p_ID") REFERENCES "Patient"("p_ID") в цьому рядку кода здійснюється прив’язка таблиці “Visit” до таблиці "Patient",

FOREIGN KEY("v_d_ID") REFERENCES "Doctor"("d_ID") в цьому рядку кода здійснюється прив’язка таблиці “Visit” до таблиці " Doctor ",

FOREIGN KEY("v_s_ID") REFERENCES "Service"("s_ID") в цьому рядку кода здійснюється прив’язка таблиці “Visit” до таблиці "Service".

Наступна таблиця “Doctor”(рис. 2.4).

Зазвичай на зовнішні ключі ставлять тип INTEGER як можна побачити в "d_c_ID" який потім відображає порядковий номер відповідного поля але в "d_sc_ID" тип TEXT так як таблиця до якої зовнішній ключ прив’язаний має первинний ключ такого ж типу TEXT.

Таблиця "Category" має такий вигляд (рис. 2.5).

В цій таблиці біля первинного ключа з’явився атрибут AUTOINCREMENT, мета цього атрибута запобігти повторному використанню SQLite значення, яке не було використано, або значення з попередньо видаленого рядка. Зазвичай цей атрибут не використовують так як він може створювати непотрібне навантаження на процесор, пам’ять або дисковий простір. В моєму випадку навантаження незначне і якогось особливого функціоналу цей атрибут не надає.

```
1   CREATE TABLE "Doctor" (
2       "d_ID"  INTEGER NOT NULL UNIQUE,
3       "d_name"    TEXT NOT NULL,
4       "d_surname" TEXT NOT NULL,
5       "d_middle_name" TEXT NOT NULL,
6       "d_c_ID"    INTEGER,
7       "d_sc_ID"   TEXT,
8       "d_experience"  INTEGER NOT NULL,
9       "d_room"    NUMERIC NOT NULL,
10      "d_e_address"   TEXT,
11      "d_phone_number"    NUMERIC NOT NULL,
12      PRIMARY KEY("d_ID"),
13      FOREIGN KEY("d_sc_ID") REFERENCES "Schedule"("sc_ID"),
14      FOREIGN KEY("d_c_ID") REFERENCES "Category"("c_ID")
15  );
```

Рисунок 2.4 - Створення таблиці "Doctor"

```
CREATE TABLE "Category" (
    "c_ID"  INTEGER NOT NULL UNIQUE,
    "c_description" TEXT NOT NULL,
    PRIMARY KEY("c_ID" AUTOINCREMENT)
);
```

Рисунок 2.5 - Створення таблиці "Category"

Таблиця “Schedule” і “Service” має властивості полів такі ж як і були наведені в таблицях вище і мають такий вигляд рисунок 2.6 “Schedule”, рисунок 2.7 “Service”.

```
CREATE TABLE "Schedule" (
    "sc_ID" TEXT NOT NULL UNIQUE,
    "sc_start"  NUMERIC NOT NULL,
    "sc_end"    NUMERIC NOT NULL,
    PRIMARY KEY("sc_ID")
);
```

Рисунок 2.6 - Створення таблиці "Schedule"

```
CREATE TABLE "Service" (
    "s_ID"  INTEGER NOT NULL UNIQUE,
    "s_description" TEXT NOT NULL,
    "s_price"   NUMERIC,
    PRIMARY KEY("s_ID")
);
```

Рисунок 2.7 - Створення таблиці "Service"

Остання таблиця яка наведена в ER-діаграмі це “M_2_m_service_category” вона була створена для здійснення зв’язку «багато до багатьох»: пов’язує список послуг з категорією лікаря, який може надавати відповідні послуги. У кожного лікаря є відповідна категорія, а кожна категорія лікаря відповідає різним послугам. Зв’язувати лікарів з “Service” буде проблематично тому прийнято рішення створити зв’язок «багато

до багатьох» в якому таблиця “Service” за допомогою зовнішнього ключа зв’язується з відповідними категоріями таблиці “Category”. Створення такої таблиці наведено на рисунку 2.8.

```
CREATE TABLE "m_2_m_service_category" (
    "s_m_ID"    INTEGER,
    "c_m_ID"    INTEGER,
    FOREIGN KEY("s_m_ID") REFERENCES "Service"("s_ID"),
    FOREIGN KEY("c_m_ID") REFERENCES "Category"("c_ID")
);
```

Рисунок 2.8 - Створення таблиці " M_2_m_service_category "

Як можна побачити ця таблиця складається з зовнішніх ключів.

FOREIGN KEY("s_m_ID") REFERENCES "Service"("s_ID"),

FOREIGN KEY("c_m_ID") REFERENCES "Category"("c_ID")

Якщо подивитися на діаграму, в ній наведено як таблиця "Category" зв’язується з “M_2_m_service_category” і в свою чергу зв’язується з "Service" утворюючи зв’язок «багато до багатьох».

Також після того як в одній із таблиць застосовується атрибут AUTOINCREMENT в базі даних автоматично створюється таблиця для обліку автоінкременту що зображено на рисунку 2.9.

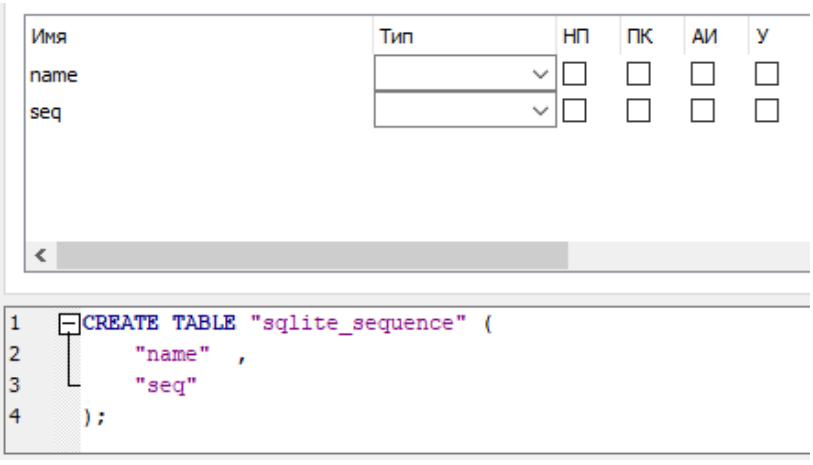


Рисунок 2.9 - Автоматично утворена таблиця " sqlite_sequence "

Ця таблиця не несе якогось функціоналу і в базі вона ніяк не використовується в моєму випадку.

2.3 Заповнення таблиць даними в SQlite

При роботі з різними платформами є різні способи заповнення даними таблиць але стандартизований спосіб це SQL підхід. SQL – це мова програмування яка відноситься до декларативного програмування тобто в якій відображується очікуваний результат а не спосіб його отримання. Ця мова широко використовується для керування, створення та зміни даних в реляційній базі даних і є дуже популярна у використанні в цій сфері діяльності.

Перш ніж ми перейдемо до SQL способу заповнення даних, потрібно розглянути не менш популярний спосіб – це імпорт таблиць і даних через відому програму EXCEL.

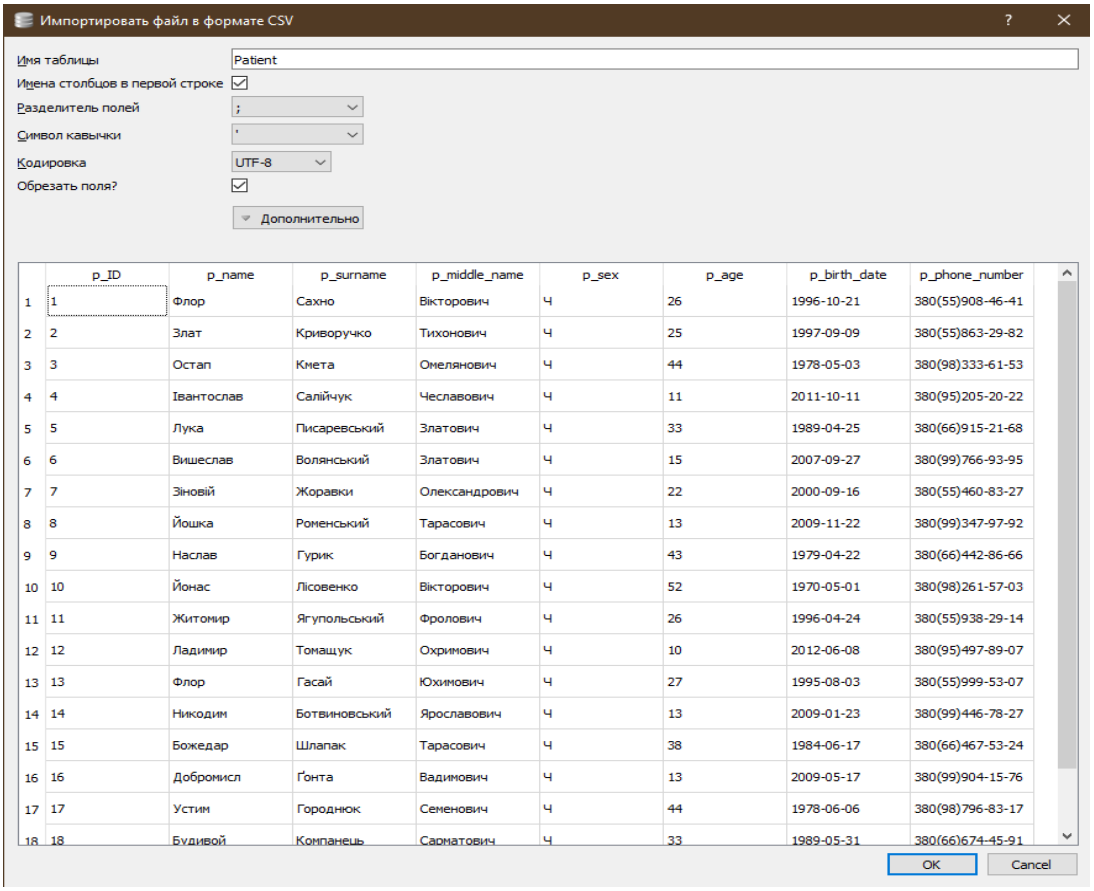
Для того щоб імпортувати таблицю і дані в "Database Browser for SQLite" потрібно створити лист в EXCEL відтворити таблицю і заповнити даними, файл EXCEL обов’язково повинен бути в потрібному форматі(.CSV), за приклад візьмемо таблицю “Patient” яку можна побачити на рисунку 2.10.

	A	B	C	D	E	F	G	H	I
1	p_ID	p_name	p_surname	p_middle_name	p_sex	p_age	p_birth_date	p_phone_number	
2	1	Флор	Сахно	Вікторович	Ч	26	21.10.1996	380(55)908-46-41	
3	2	Злат	Криворучко	Тихонович	Ч	25	09.09.1997	380(55)863-29-82	
4	3	Остап	Кмета	Омелянович	Ч	44	03.05.1978	380(98)333-61-53	
5	4	Івантослав	Салійчук	Чеславович	Ч	11	11.10.2011	380(95)205-20-22	
6	5	Лука	Писаревський	Златович	Ч	33	25.04.1989	380(66)915-21-68	
7	6	Вишеслав	Волянський	Златович	Ч	15	27.09.2007	380(99)766-93-95	
8	7	Зіновій	Жоравки	Олександрович	Ч	22	16.09.2000	380(55)460-83-27	
9	8	Йошка	Роменський	Тарасович	Ч	13	22.11.2009	380(99)347-97-92	
10	9	Наслав	Гурик	Богданович	Ч	43	22.04.1979	380(66)442-86-66	
11	10	Йонас	Лісовенко	Вікторович	Ч	52	01.05.1970	380(98)261-57-03	
12	11	Житомир	Ягупольський	Фролович	Ч	26	24.04.1996	380(55)938-29-14	
13	12	Ладимир	Томащук	Охримович	Ч	10	08.06.2012	380(95)497-89-07	
14	13	Флор	Гасай	Юхимович	Ч	27	03.08.1995	380(55)999-53-07	
15	14	Никодим	Ботвиновський	Ярославович	Ч	13	23.01.2009	380(99)446-78-27	
16	15	Божедар	Шлапак	Тарасович	Ч	38	17.06.1984	380(66)467-53-24	
17	16	Добромисл	Ґонта	Вадимович	Ч	13	17.05.2009	380(99)904-15-76	
18	17	Устим	Городнюк	Семенович	Ч	44	06.06.1978	380(98)796-83-17	
19	18	Будивой	Компанець	Сарматович	Ч	33	31.05.1989	380(66)674-45-91	
20	19	Цвітан	Андрієнко	Макарович	Ч	44	02.10.1978	380(98)100-42-32	
21	20	Іларіон	Корпанюк	Антонович	Ч	48	24.02.1974	380(98)991-79-52	
22	21	Никодим	Ріпецький	Олександрович	Ч	30	29.02.1992	380(55)892-48-68	
23	22	Полель	Малий	Леонідович	Ч	44	01.05.1978	380(98)306-75-37	
24	23	Цвітан	Костельник	Чеславович	Ч	30	11.06.1992	380(55)349-93-04	
25	24	Юрій	Галушка	Чеславович	Ч	10	08.10.2012	380(95)561-89-66	
26	25	Фауст	Кияк	Тарасович	Ч	9	29.04.2013	380(95)834-44-03	
27	26	Євстафій	Красіцький	Олегович	Ч	7	08.03.2015	380(95)009-38-41	
28	27	Єремій	Роменський	Юліанович	Ч	13	26.03.2009	380(99)227-32-54	
29	28	Мстивой	Курилюк	Милославович	Ч	17	24.03.2005	380(99)029-51-42	

Рисунок 2.10 – Таблиця “Patient” в EXCEL

Далі через команду імпорт потрібно перенести таблицю в SQLite і зберегти дані. Вигляд імпортованої таблиці можна побачити на рисунку 2.11.

Другий спосіб це заповнення даних за допомогою SQL а якщо точніше то SQL коду який являється стандартизованим і найпопулярнішим тому що цей метод використовується в багатьох різних платформах чого не можна сказати про EXCEL. Код за допомогою якого створено інформацію для таблиць скорочений так як в базі даних дуже велика кількість даних.



	p_ID	p_name	p_surname	p_middle_name	p_sex	p_age	p_birth_date	p_phone_number
1	1	Флор	Сахно	Вікторович	Ч	26	1996-10-21	380(55)908-46-41
2	2	Злат	Криворучко	Тихонович	Ч	25	1997-09-09	380(55)863-29-82
3	3	Остап	Кмета	Омелянович	Ч	44	1978-05-03	380(98)333-61-53
4	4	Івантослав	Салійчук	Чеславович	Ч	11	2011-10-11	380(95)205-20-22
5	5	Лука	Писаревський	Златович	Ч	33	1989-04-25	380(66)915-21-68
6	6	Вишеслав	Волянський	Златович	Ч	15	2007-09-27	380(99)766-93-95
7	7	Зіновій	Жоравки	Олександрович	Ч	22	2000-09-16	380(55)460-83-27
8	8	Йошка	Роменський	Тарасович	Ч	13	2009-11-22	380(99)347-97-92
9	9	Наслав	Гурик	Богданович	Ч	43	1979-04-22	380(66)442-86-66
10	10	Йонас	Лісовенко	Вікторович	Ч	52	1970-05-01	380(98)261-57-03
11	11	Житомир	Ягупольський	Фролович	Ч	26	1996-04-24	380(55)938-29-14
12	12	Ладимир	Томащук	Охримович	Ч	10	2012-06-08	380(95)497-89-07
13	13	Флор	Гасай	Юхимович	Ч	27	1995-08-03	380(55)999-53-07
14	14	Никодим	Ботвиновський	Ярославович	Ч	13	2009-01-23	380(99)446-78-27
15	15	Божедар	Шлапак	Тарасович	Ч	38	1984-06-17	380(66)467-53-24
16	16	Добромисл	Ґонта	Вадимович	Ч	13	2009-05-17	380(99)904-15-76
17	17	Устим	Городнюк	Семенович	Ч	44	1978-06-06	380(98)796-83-17
18	18	Будивой	Компанець	Сарматович	Ч	33	1989-05-31	380(66)674-45-91

Рисунок 2.11 – Імпорт таблиці “Patient” з EXCEL

Перша таблиця “Patient” заповнена за допомогою SQL має вигляд:

INSERT INTO "Patient" ("p_ID", "p_name", "p_surname", "p_middle_name", "p_sex",

"p_age", "p_birth_date", "p_phone_number") VALUES ('1', 'Флор', 'Сахно', 'Вікторович', 'Ч', '26', '1996-10-21', '380(55)908-46-41');

('2', 'Злат', 'Криворучко', 'Тихонович', 'Ч', '25', '1997-09-09', '380(55)863-29-82');

('3', 'Остап', 'Кмета', 'Омелянович', 'Ч', '44', '1978-05-03', '380(98)333-61-53');

('4', 'Івантослав', 'Салійчук', 'Чеславович', 'Ч', '11', '2011-10-11', '380(95)205-20-22');

('5', 'Лука', 'Писаревський', 'Златович', 'Ч', '33', '1989-04-25', '380(66)915-21-68');

('6', 'Вишеслав', 'Волянський', 'Златович', 'Ч', '15', '2007-09-27', '380(99)766-93-95');

('7', 'Зіновій', 'Жоравки', 'Олександрович', 'Ч', '22', '2000-09-16', '380(55)460-83-27');

('8', 'Йошка', 'Роменський', 'Тарасович', 'Ч', '13', '2009-11-22', '380(99)347-97-92');

('9', 'Наслав', 'Гурик', 'Богданович', 'Ч', '43', '1979-04-22', '380(66)442-86-66');

('10', 'Йонас', 'Лісовенко', 'Вікторович', 'Ч', '52', '1970-05-01', '380(98)261-57-03');

Таким способом я заповнив таблицю на 100 пацієнтів. За допомогою команди **INSERT** вставляю записи до бази даних. Можна вставити або один запис або відразу кілька. В цій таблиці я вставив 8 записів які можна побачити на початку кодового рядку ("p_ID", "p_name", "p_surname", "p_middle_name", "p_sex", "p_age", "p_birth_date", "p_phone_number"), команда **VALUES** записує в записи команди **INSERT** інформацію яку вказали в рядку SQL кода.

Друга таблиця “Doctor” має вигляд:

INSERT INTO "Doctor" ("d_ID", "d_name", "d_surname", "d_middle_name", "d_c_ID", "d_sc_ID", "d_experience", "d_room", "d_e_address", "d_phone_number") VALUES ('1', 'Нігослав', 'Кухар', 'Юхимович', '1', 'Повний день', '12', '100', 'meiquacrallaubru-9859@mail.com', '380(99)413-32-74');

('2', 'Яснолик', 'Зубрицький', 'Любомирович', '2', 'Перша половина дня', '5', '101', 'borecrapeimmi-2051@mail.com', '380(99)534-88-43');

('3', 'Йосеф', 'Кравчук', 'Володимирович', '2', 'Друга половина дня', '4', '102', 'gehocuwoca-7624@mail.com', '380(99)184-66-64');

('4', 'Цвітан', 'Кривенко', 'Жданович', '3', 'Повний день', '6', '103', 'zettucroizaque-8693@mail.com', '380(99)254-29-34');

('5', 'Макар', 'Нагірний', 'Олександрович', '3', 'Друга половина дня', '6', '104', 'nujatreubritto-4231@mail.com

', '380(99)577-75-29');

('6', 'Стожар', 'Сеник', 'Максимович', '1', 'Перша половина дня', '8', '105', 'nuprennoimuyo-4717@mail.com', '380(99)549-49-28');

('7', 'Ян', 'Соколенко', 'Найденович', '3', 'Повний день', '11', '106', 'wodemmolimo-4848@mail.com', '380(99)127-79-17');

('8', 'Шарль', 'Данильчук', 'Світанович', '4', 'Перша половина дня', '9', '107', 'hamottaxulu-6347@mail.com', '380(99)724-65-89');

('9', 'Йошка', 'Дмитрієнко', 'Герасимович', '2', 'Повний день', '7', '108', 'brissapreubebe-9366@mail.com', '380(99)678-19-72');

('10', 'Щедрогост', 'Штокало', 'Полянович', '2', 'Повний день', '5', '109', 'yexuhoippeila-7271@mail.com', '380(99)210-14-82');

Таким способом я заповнив таблицю на 20 лікарів. В цій і в наступних таблиці використовуються такі ж команди які використовувалися в таблиці “Patient”.

Третя таблиця “Visit” має вигляд:

INSERT INTO "Visit" ("v_ID", "v_date", "v_p_ID", "v_d_ID", "v_s_ID", "v_diagnosis") VALUES ('1', '2021-11-07', '1', '14', '2', 'Вроджена катаракта');

('2', '2021-11-07', '5', '6', '6', 'Астигматизм');

('3', '2021-11-12', '34', '18', '5', 'Патології сітківки ока');

('4', '2021-11-12', '20', '8', '3', 'Далекозорість');

('5', '2021-11-12', '25', '10', '7', 'Далекозорість');

('6', '2021-11-12', '73', '11', '7', 'Катаракта');

Четверта таблиця “Category” має таку інформацію:

INSERT INTO "Category" ("c_ID", "c_description") VALUES ('1', 'Хірург-офтальмолог');

('2', 'Дитячий офтальмолог');

('3', 'Лікар-офтальмолог вищої кваліфікаційної категорії');

('4', 'Лікар-офтальмолог початкової кваліфікаційної категорії');

П’ята таблиця “Schedule”:

INSERT INTO "Schedule" ("sc_ID", "sc_start", "sc_end") VALUES ('Повний день', '8:00', '19:00');

('Перша половина дня', '8:00', '13:30');

('Друга половина дня', '13:30', '19:00');

Шоста таблиця “Service”:

INSERT INTO "Service" ("s_ID", "s_description", "s_price") VALUES ('1', 'Діагностика зору', '1500 грн');

('2', 'Хірургічне лікування катаракти', '25000 грн');

('3', 'Хірургічне лікування глаукоми', '20000 грн');

('4', 'Лазерне лікування глаукоми', '14000 грн');

('5', 'Лазерна корекція', '29000 грн');

('6', 'Естетична хірургія', '10000 грн');

('7', 'Діагностика зору для дітей ', '1000 грн');

('8', 'Підбір окулярів', '300 грн');

Остання таблиця “M_2_m_service_category”:

INSERT INTO "m_2_m_service_category" ("s_m_ID", "c_m_ID") VALUES ('1', '1');

INSERT INTO "m_2_m_service_category" ("s_m_ID", "c_m_ID") VALUES ('1', '3');

INSERT INTO "m_2_m_service_category" ("s_m_ID", "c_m_ID") VALUES ('1', '4');

INSERT INTO "m_2_m_service_category" ("s_m_ID", "c_m_ID") VALUES ('2', '1');

INSERT INTO "m_2_m_service_category" ("s_m_ID", "c_m_ID") VALUES ('2', '3');

За допомогою простого SQL коду можна швидко отримати заповнені таблиці і вже готову базу даних. Готова таблиця в програмі "Database Browser for SQLite" виглядає як на рисунку 2.12 на прикладі таблиці “Doctor”.

Таблиця: Doctor — Filter in any column

	d_ID	d_name	d_surname	d_middle_name	d_c_ID	d_sc_ID	d_experience	d_room	d_e_address	d_phone_number
	Фи...	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр
1	1	Нігослав	Кухар	Юхимович	1	Повний день	12	100	meiquacrallaubru-9859@mail.com	380(99)413-32-74
2	2	Яснолик	Зубрицький	Любомирович	2	Перша половина дня	5	101	borecrapeimmi-2051@mail.com	380(99)534-88-43
3	3	Йосеф	Кравчук	Володимирович	2	Друга половина дня	4	102	gehocuwoca-7624@mail.com	380(99)184-66-64
4	4	Цвітан	Кривенко	Жданович	3	Повний день	6	103	zettucroizaque-8693@mail.com	380(99)254-29-34
5	5	Макар	Нагірний	Олександрович	3	Друга половина дня	6	104	nujatreubritto-4231@mail.com...	380(99)577-75-29
6	6	Стожар	Сеник	Максимович	1	Перша половина дня	8	105	nuprennoimuyo-4717@mail.com	380(99)549-49-28
7	7	Ян	Соколенко	Найденович	3	Повний день	11	106	wodemmolimo-4848@mail.com	380(99)127-79-17
8	8	Шарль	Данильчук	Світанович	4	Перша половина дня	9	107	hamottaxulu-6347@mail.com	380(99)724-65-89
9	9	Йошка	Дмитрієнко	Герасимович	2	Повний день	7	108	brissapreubebe-9366@mail.com	380(99)678-19-72
10	10	Щедрогост	Штокало	Полянович	2	Повний день	5	109	yexuhoippeila-7271@mail.com	380(99)210-14-82
11	11	Євлампій	Герцик	Захарович	3	Друга половина дня	5	110	crehufrazattu-9193@mail.com	380(95)483-25-67
12	12	Мечислав	Артимович	Радимович	1	Друга половина дня	7	111	renecaffowe-1299@mail.com	380(95)914-97-52
13	13	Євгенія	Трофименко	Адамівна	4	Повний день	9	112	beirofexegau-6104@mail.com	380(95)450-34-69
14	14	Цвітана	Ігнатченко	Августинівна	1	Перша половина дня	6	113	quivemuxajou-1696@mail.com	380(95)683-55-63
15	15	Вікторина	Шахрай	Чеславівна	2	Перша половина дня	6	114	fragexouseimmo-4328@mail.com	380(95)461-03-90
16	16	Божена	Холоша	Любомирівна	4	Перша половина дня	2	115	jimetreimoitrei-7379@mail.com	380(95)041-70-40
17	17	Христина	Сосновий	Федорівна	3	Повний день	1	116	lazequoiroipi-4151@mail.com	380(95)909-37-88
18	18	Твориславa	Журмани	Северинівна	3	Друга половина дня	1	117	xeyoufasaunna-6653@mail.com	380(95)729-28-21
19	19	Арсенія	Різниченко	Глібівна	2	Друга половина дня	8	118	grifferufiprau-1156@mail.com	380(95)934-22-54
20	20	Ісидора	Мінченко	Давидівна	4	Перша половина дня	9	119	roubrateisemo-2287@mail.com	380(95)061-10-16

Рисунок 2.12 – Вигляд готової таблиці “Doctor”

РОЗДІЛ 3. СТВОРЕННЯ І ВИКОНАННЯ SQL ЗАПИТІВ В СТВОРЕНІЙ БАЗІ ДАНИХ

3.1 Логічно змістовий запит за створеною базою даних. Інформація щодо візиту та висновків для пацієнта.

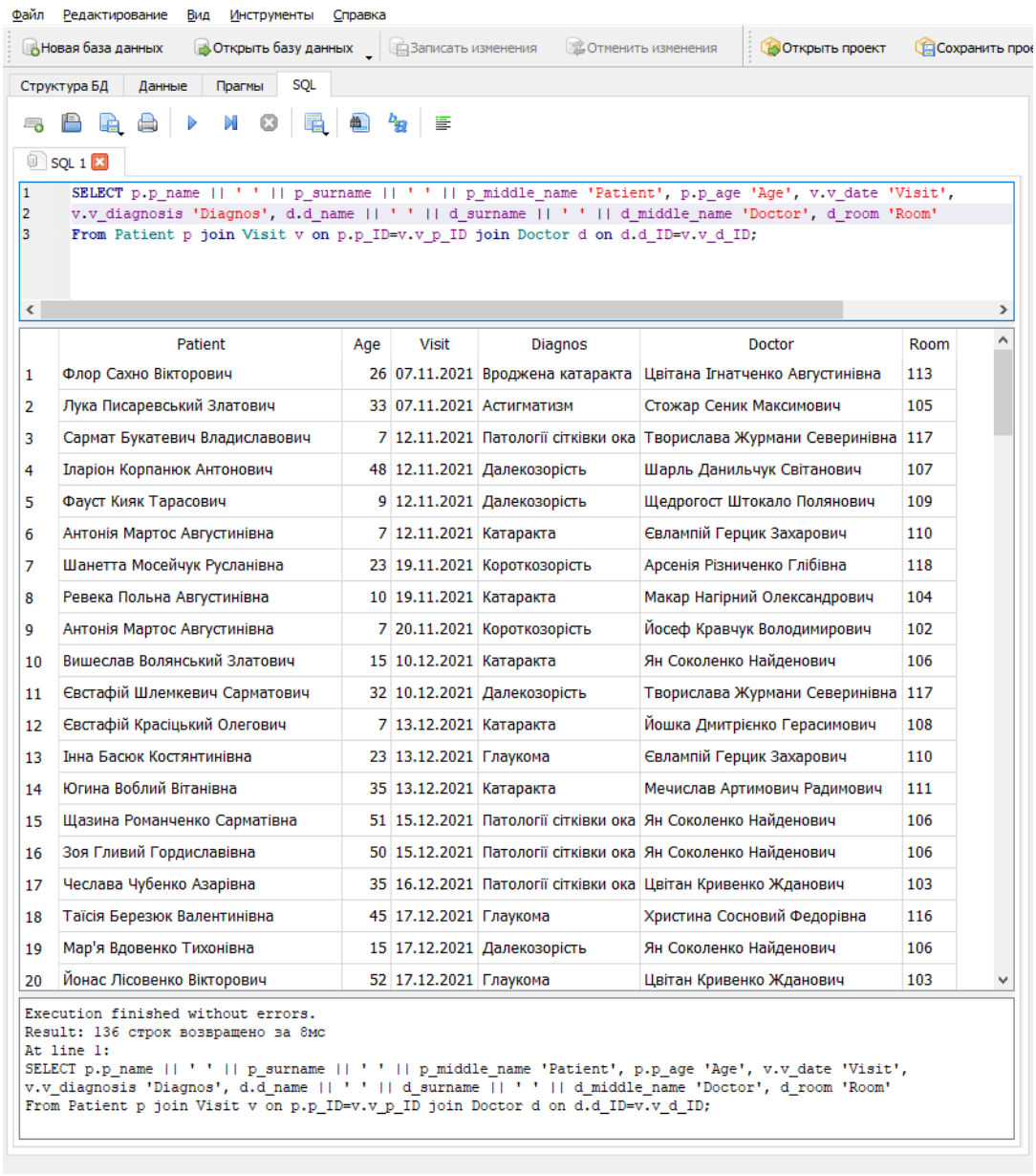
Цей запит (рис. 3.1) виводить список пацієнтів, їх вік, дату візиту до клініки, діагноз який був поставлений лікарем після візиту, лікаря який обслуговував відповідного пацієнта і кабінет в якому був прийом.

```
SELECT p.p_name || ' ' || p_surname || ' ' || p_middle_name 'Patient', p.p_age 'Age',
v.v_date 'Visit', v.v_diagnosis 'Diagnos',

d.d_name || ' ' || d_surname || ' ' || d_middle_name 'Doctor', d_room 'Room'

From Patient p join Visit v on p.p_ID=v.v_p_ID join Doctor d on d.d_ID=v.v_d_ID;
```

В цьому запиті за допомогою команди SELECT ми отримуємо і виводимо вказані дані, наприклад (p.p_age 'Age') p.p_age в даному випадку таблиця бази даних яка виводиться в запиті під назвою 'Age'. За допомогою команди FROM ми вказуємо з якої таблиці потрібно брати інформацію, а команда JOIN зв’язує таблиці за допомогою зовнішніх і первинних ключів які можна було побачити ще в ER-діаграмі. Зв’язка відбувається після атрибуту ON який допомагає команді JOIN з’єднувати ключі, (p.p_ID=v.v_p_ID) тут можна побачити первинний ключ таблиці “Patient” p.p_ID який зв’язується з зовнішнім ключем таблиці “Visit” v.v_p_ID. В результаті ми отримуємо логічно-зв’язані між собою дані з потрібною нам інформацією.

В коді перед даними таблиці, не у всіх випадках, але в деяких я вказую відповідну літеру яка потім прив’язується до таблиць бази даних, це можна побачити після SELECT (p.p_name) і після FROM (Patient p). Літера P означає що таблиця запиту 'Patient' з даними прив’язується для виводу інформації з таблицею 'Patient' бази даних.

Також в запиті присутні позначення “ || ' ' || ” . Ці позначки виконують функцію об’єднання рядків і вказують що потрібно бути вписано між таблицями які виводяться в запиті. У випадку p.p_name || ' ' || p_surname || ' ' || p_middle_name 'Patient' ми вказуємо що виводиться таблиця 'Patient' з даними ім’я, прізвища та по-батькові маючи між ними пробіл || ' ' || і завдяки цій позначці рядки p_name, p_surname, p_middle_name які об’єдналися виводяться в одній таблиці в даному випадку таблиці 'Patient'.



	Patient	Age	Visit	Diagnos	Doctor	Room
1	Флор Сахно Вікторович	26	07.11.2021	Вроджена катаракта	Цвітана Ігнатченко Августинівна	113
2	Лука Писаревський Златович	33	07.11.2021	Астигматизм	Стожар Сеник Максимович	105
3	Сармат Букатевич Владиславович	7	12.11.2021	Патології сітківки ока	Творислава Журмани Северинівна	117
4	Іларіон Корпанюк Антонович	48	12.11.2021	Далекозорість	Шарль Данильчук Світанович	107
5	Фауст Кияк Тарасович	9	12.11.2021	Далекозорість	Щедрогост Штокало Полянович	109
6	Антонія Мартос Августинівна	7	12.11.2021	Катаракта	Євлампій Герцик Захарович	110
7	Шанетта Мосейчук Русланівна	23	19.11.2021	Короткозорість	Арсенія Різниченко Глібівна	118
8	Ревека Польна Августинівна	10	19.11.2021	Катаракта	Макар Нагірний Олександрович	104
9	Антонія Мартос Августинівна	7	20.11.2021	Короткозорість	Йосеф Кравчук Володимирович	102
10	Вишеслав Волянський Златович	15	10.12.2021	Катаракта	Ян Соколенко Найденович	106
11	Євстафій Шлемкевич Сарматович	32	10.12.2021	Далекозорість	Творислава Журмани Северинівна	117
12	Євстафій Красіцький Олегович	7	13.12.2021	Катаракта	Йошка Дмитрієнко Герасимович	108
13	Інна Басюк Костянтинівна	23	13.12.2021	Глаукома	Євлампій Герцик Захарович	110
14	Югина Воблий Вітанівна	35	13.12.2021	Катаракта	Мечислав Артимович Радимович	111
15	Щазина Романченко Сарматівна	51	15.12.2021	Патології сітківки ока	Ян Соколенко Найденович	106
16	Зоя Гливий Гордиславівна	50	15.12.2021	Патології сітківки ока	Ян Соколенко Найденович	106
17	Чеслава Чубенко Азарівна	35	16.12.2021	Патології сітківки ока	Цвітан Кривенко Жданович	103
18	Таїсія Березюк Валентинівна	45	17.12.2021	Глаукома	Христина Сосновий Федорівна	116
19	Мар'я Вдовенко Тихонівна	15	17.12.2021	Далекозорість	Ян Соколенко Найденович	106
20	Йонас Лісовенко Вікторович	52	17.12.2021	Глаукома	Цвітан Кривенко Жданович	103

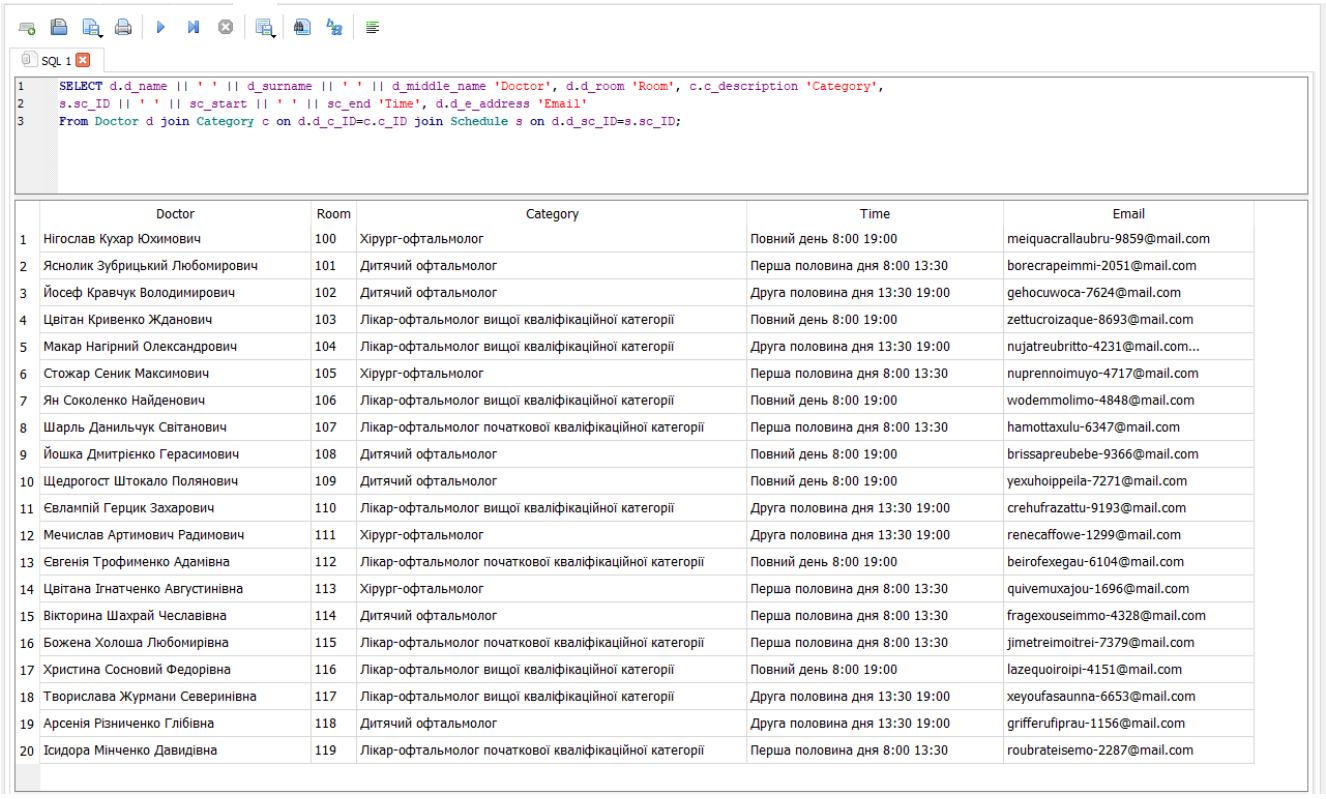
Рисунок 3.1 - Приклад виконання запиту “Логічно змістовий запит”

3.2 Запити на з’єднання таблиць. Повна інформація про лікарів.

Перший запит (рис. 3.2) виводить список лікарів, кабінет де він працює, їхню спеціалізацію, графік роботи і електронну адресу.

```
SELECT d.d_name || ' ' || d_surname || ' ' || d_middle_name 'Doctor',
d.d_room 'Room', c.c_description 'Category',
s.sc_ID || ' ' || sc_start || ' ' || sc_end 'Time', d.d_e_address 'Email'
From Doctor d join Category c on d.d_c_ID=c.c_ID
join Schedule s on d.d_sc_ID=s.sc_ID;
```

В цьому запиті застосовуються аналогічні команди як і в першому запиті.



SQL 1

```
SELECT d.d_name || ' ' || d_surname || ' ' || d_middle_name 'Doctor', d.d_room 'Room', c.c_description 'Category',
s.sc_ID || ' ' || sc_start || ' ' || sc_end 'Time', d.d_e_address 'Email'
From Doctor d join Category c on d.d_c_ID=c.c_ID join Schedule s on d.d_sc_ID=s.sc_ID;
```

	Doctor	Room	Category	Time	Email
1	Нігослав Кухар Юхимович	100	Хірург-офтальмолог	Повний день 8:00 19:00	meiquacrallaubru-9859@mail.com
2	Яснолик Зубрицький Любомирович	101	Дитячий офтальмолог	Перша половина дня 8:00 13:30	borecrapeimmi-2051@mail.com
3	Йосеф Кравчук Володимирович	102	Дитячий офтальмолог	Друга половина дня 13:30 19:00	gehocuwoca-7624@mail.com
4	Цвітан Кривенко Жданович	103	Лікар-офтальмолог вищої кваліфікаційної категорії	Повний день 8:00 19:00	zettucroizaque-8693@mail.com
5	Макар Нагірний Олександрович	104	Лікар-офтальмолог вищої кваліфікаційної категорії	Друга половина дня 13:30 19:00	nujatreubritto-4231@mail.com...
6	Стожар Сеник Максимович	105	Хірург-офтальмолог	Перша половина дня 8:00 13:30	nuprennoimuyo-4717@mail.com
7	Ян Соколенко Найденович	106	Лікар-офтальмолог вищої кваліфікаційної категорії	Повний день 8:00 19:00	wodemmolimo-4848@mail.com
8	Шарль Данильчук Світанович	107	Лікар-офтальмолог початкової кваліфікаційної категорії	Перша половина дня 8:00 13:30	hamottaxulu-6347@mail.com
9	Йошка Дмитрієнко Герасимович	108	Дитячий офтальмолог	Повний день 8:00 19:00	brissapreubebe-9366@mail.com
10	Щедрогост Штокало Полянович	109	Дитячий офтальмолог	Повний день 8:00 19:00	yexuhoippeila-7271@mail.com
11	Євлампій Герцик Захарович	110	Лікар-офтальмолог вищої кваліфікаційної категорії	Друга половина дня 13:30 19:00	crehufrazattu-9193@mail.com
12	Мечислав Артимович Радимович	111	Хірург-офтальмолог	Друга половина дня 13:30 19:00	renecaffowe-1299@mail.com
13	Євгенія Трофименко Адамівна	112	Лікар-офтальмолог початкової кваліфікаційної категорії	Повний день 8:00 19:00	beirofexegau-6104@mail.com
14	Цвітана Ігнатченко Августинівна	113	Хірург-офтальмолог	Перша половина дня 8:00 13:30	quivemuxajou-1696@mail.com
15	Вікторина Шахрай Чеславівна	114	Дитячий офтальмолог	Перша половина дня 8:00 13:30	fragexouseimmo-4328@mail.com
16	Божена Холоша Любомирівна	115	Лікар-офтальмолог початкової кваліфікаційної категорії	Перша половина дня 8:00 13:30	jimetreimoitrei-7379@mail.com
17	Христина Сосновий Федорівна	116	Лікар-офтальмолог вищої кваліфікаційної категорії	Повний день 8:00 19:00	lazequoiroipi-4151@mail.com
18	Творислава Журмани Северинівна	117	Лікар-офтальмолог вищої кваліфікаційної категорії	Друга половина дня 13:30 19:00	xeyoufasaunna-6653@mail.com
19	Арсенія Різниченко Глібівна	118	Дитячий офтальмолог	Друга половина дня 13:30 19:00	grifferufiprau-1156@mail.com
20	Ісидора Мінченко Давидівна	119	Лікар-офтальмолог початкової кваліфікаційної категорії	Перша половина дня 8:00 13:30	roubrateisemo-2287@mail.com

Рисунок 3.2 - Приклад виконання запиту “Запит на з’єднання таблиць”

Наступний запит (рис. 3.3) містить лікарів, номери послуг, розшифровку та робочий графік лікаря.

SELECT d_name ||' '|| d_surname 'Doctor',s_ID 'Service id', s_description 'Service', sc_start ||'-'|| sc_end 'Time'

from Doctor d join visit v on d.d_ID = v.v_d_ID

join Service s on v.v_s_ID=s.s_ID

join Schedule w on d.d_sc_ID=w.sc_ID;

В цьому запиті використовуються прості але вдало підібрані команди.

Команда SELECT яка дозволяє виводити потрібну вказану інформацію бази даних.

Команда FROM за допомогою якої вказується з яких таблиць потрібно виводити дані.

Команда JOIN яка допомагає попередній команді зв’язувати таблиці за допомогою відповідних зв’язків, атрибут ON який в свою чергу йде після JOIN виконує функцію умови зв’язку.

Також в запиті так само як в двох минулих присутні деякі позначення такі як “ ||' '|| ” і “||'-'|| ”. Ці позначки виконують функцію об’єднання рядків і означають що потрібно бути написано між таблицями які виводяться в даному запиті. У випадку d_name ||' '|| d_surname 'Doctor' ми вказуємо що виводиться таблиця 'Doctor' з даними ім’я та прізвища маючи між ними пробіл ||' '|| і завдяки цій позначці рядки які об’єдналися виводяться в одній таблиці в даному випадку таблиці 'Doctor'. А у випадку sc_start ||'-'|| sc_end 'Time' ми вказуємо що виводиться таблиця 'Time' з даними початку і кінця зміни маючи між ними тире ||'-'||. Якщо не вказувати в записі пробіл або інші потрібні позначки тоді при відображенні таблиці всі дані які в неї вписані будуть злиті в одне слово.

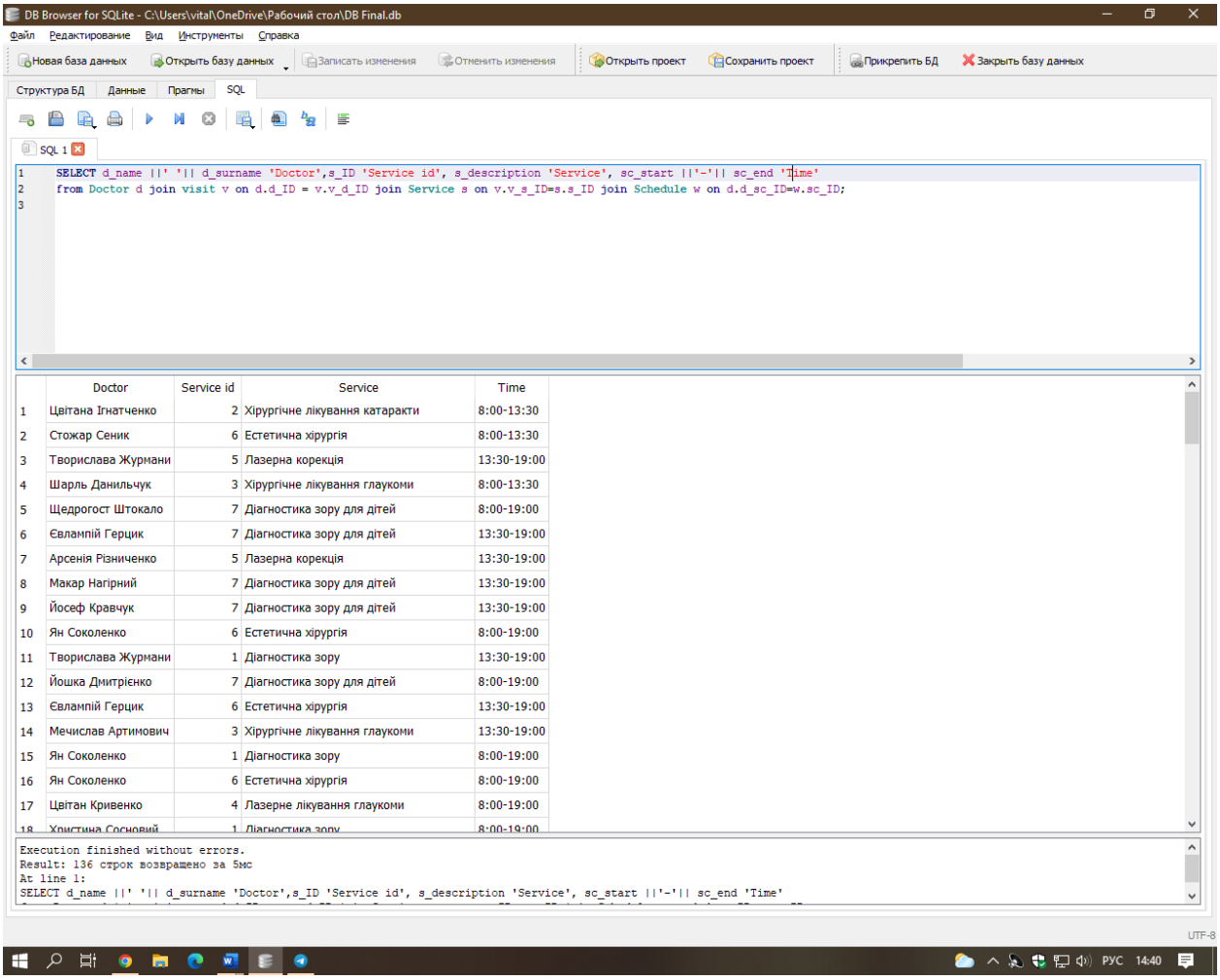


Рисунок 3.3 - Приклад виконання запиту “Запит на з’єднання таблиць”

3.3 Запити з умовами. Інформація щодо лікарів у відпустці. Лікарі на певній зміні.

Перший запит (рис. 3.4) має умову за допомогою якої можна дізнатися який лікар не приймав жодного пацієнта або простими словами знаходиться у відпустці тим самим дізнатися його дані, в даному випадку ім’я, прізвище та по-батькові.

SELECT d.d_ID 'Doctor id', d.d_name || ' ' || d_surname || ' ' || d_middle_name 'Doctor' from Doctor d left join visit v on d.d_ID = v.v_d_IDwhere v.v_d_ID is NULL;

В запиті використовується команда WHERE вона визначає умову, за якою будуть вибиратися рядки з бази даних, в цій ситуації обираються рядки лікарів у яких не має жодного візиту пацієнта тобто рядок зі значенням NULL.

Також тут присутня команда left join по сутності вона як і звичайний join але з однією відмінністю. Якщо join просто зв’язує таблиці то left join зв’язує таблиці за певними полями зв'язку. Іншими словами, всі рядки в першій таблиці включаються в набір другої, незалежно від того, є відповідні рядки в другій таблиці чи ні.

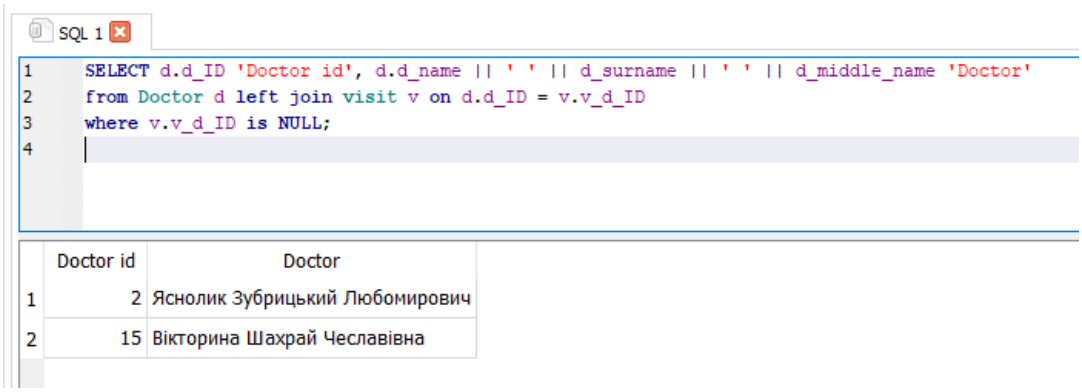


Рисунок 3.4 - Приклад виконання запиту “Запит з умовами”

Другий запит (рис. 3.5) має умову за допомогою якої можна дізнатися який лікар працює повний робочий день. В цьому запит можна дізнатися ім’я, прізвище, по-батькові лікаря, його робочу зміну в даному випадку лікаря який працює повний робочий день, електронну адресу лікаря.

Select d_name || ‘ ‘ || d_surname || ‘ ‘ || d_middle_name ‘Doctor name’, d_sc_ID ‘Half of day’, d_e_address ‘Email’

from Doctor where d_sc_ID=’Повний день’

В запиті (рис. 3.5) використовується команда WHERE вона визначає умову, за якою будуть вибиратися рядки з бази даних. В цьому запиті я вказав таблицю з

розкладом лікаря (d_sc_ID) який прирівнюється до одного з значень цієї ж таблиці, в даному випадку це значення ‘Повний день’.

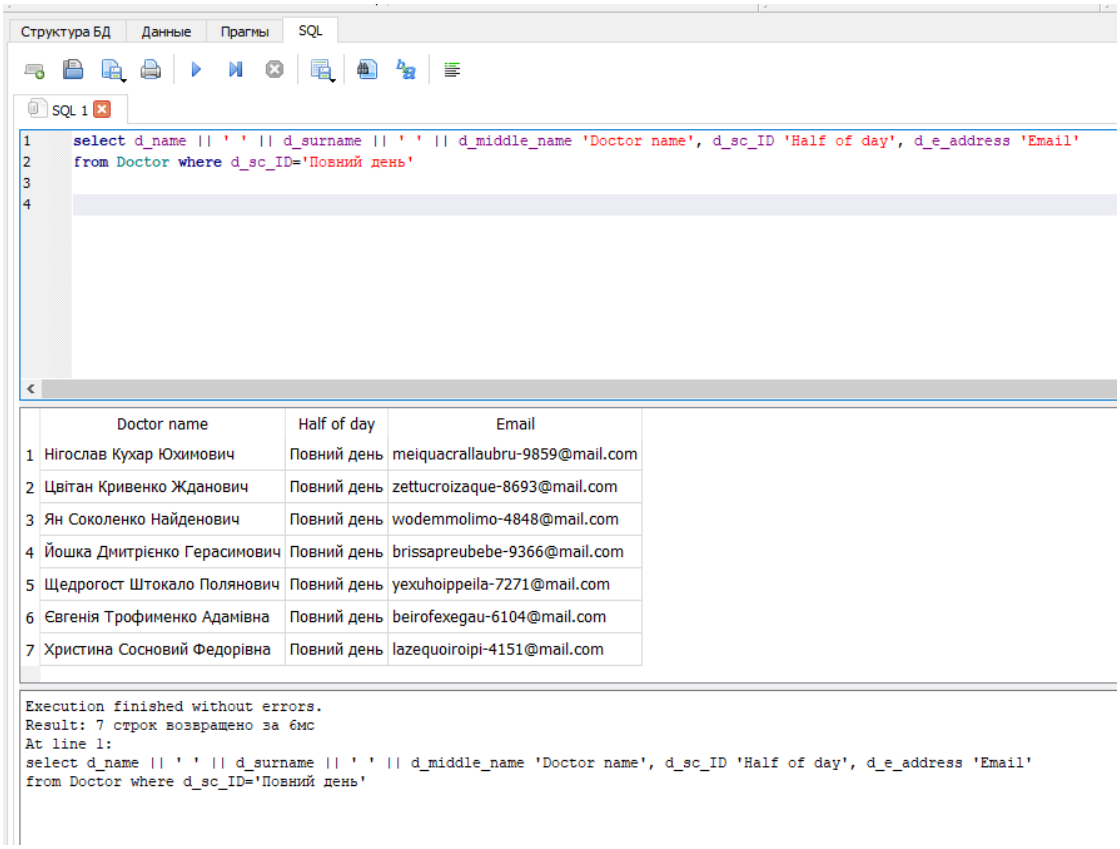


Рисунок 3.5 - Приклад виконання запиту “Запит з умовами”

3.4 Запит з умовами. Інформація щодо витрат пацієнтів дітей за різні послуги.

Цей запит (рис. 3.6) зв’язує відповідні таблиці виводячи інформацію пацієнта, його вік, лікаря та його кваліфікацію який працював з цим пацієнтом, а також ціну та саму послугу яку надає клініка.

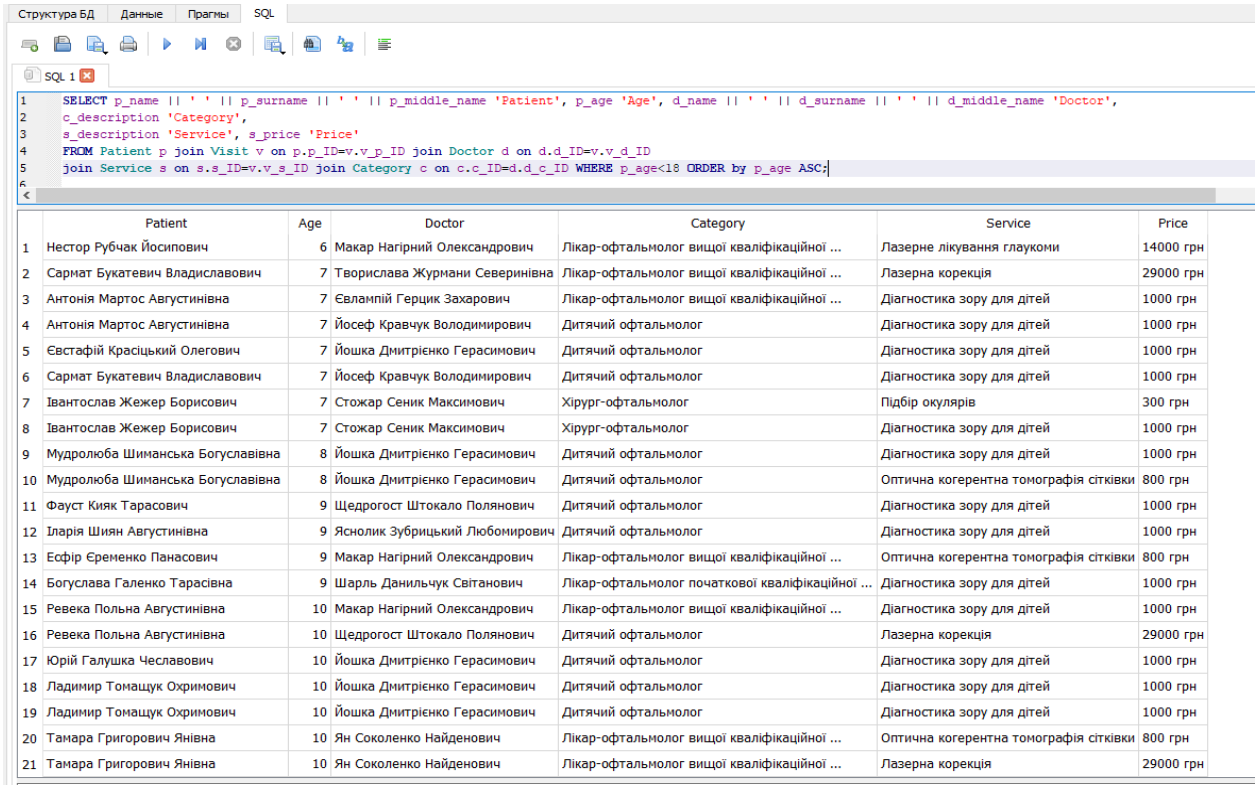
SELECT p_name || ' ' || p_surname || ' ' || p_middle_name 'Patient',

p_age 'Age',

d_name || ' ' || d_surname || ' ' || d_middle_name 'Doctor',

```
c_description 'Category',
s_description 'Service', s_price 'Price'
FROM Patient p join Visit v on p.p_ID=v.v_p_ID
join Doctor d on d.d_ID=v.v_d_ID
join Service s on s.s_ID=v.v_s_ID
join Category c on c.c_ID=d.d_c_ID WHERE p_age<18 ORDER by p_age ASC;
```

В запиті знову використовується команда WHERE яка вказує що вік пацієнта повинен бути менше 18(WHERE p_age<18) тобто запит повинен вивести список дітей, також використовується команда ORDER by яка дозволяє сортувати інформацію в порядку зростання.



	Patient	Age	Doctor	Category	Service	Price
1	Нестор Рубчак Йосипович	6	Макар Нагірний Олександрович	Лікар-офтальмолог вищої кваліфікаційної ...	Лазерне лікування глаукоми	14000 грн
2	Сармат Букатевич Владиславович	7	Творислава Журмани Северинівна	Лікар-офтальмолог вищої кваліфікаційної ...	Лазерна корекція	29000 грн
3	Антонія Мартос Августинівна	7	Євлампій Герцик Захарович	Лікар-офтальмолог вищої кваліфікаційної ...	Діагностика зору для дітей	1000 грн
4	Антонія Мартос Августинівна	7	Йосеф Кравчук Володимирович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
5	Євстафій Красіцький Олегович	7	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
6	Сармат Букатевич Владиславович	7	Йосеф Кравчук Володимирович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
7	Івантослав Жежер Борисович	7	Стожар Сеник Максимович	Хірург-офтальмолог	Підбір окулярів	300 грн
8	Івантослав Жежер Борисович	7	Стожар Сеник Максимович	Хірург-офтальмолог	Діагностика зору для дітей	1000 грн
9	Мудролюба Шиманська Богуславівна	8	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
10	Мудролюба Шиманська Богуславівна	8	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Оптична когерентна томографія сітківки	800 грн
11	Фауст Кияк Тарасович	9	Щедрогост Штокало Полянович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
12	Іларія Шиян Августинівна	9	Яснолик Зубрицький Любомирович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
13	Есфір Єременко Панасович	9	Макар Нагірний Олександрович	Лікар-офтальмолог вищої кваліфікаційної ...	Оптична когерентна томографія сітківки	800 грн
14	Богуслава Галенко Тарасівна	9	Шарль Данильчук Світанович	Лікар-офтальмолог початкової кваліфікаційної ...	Діагностика зору для дітей	1000 грн
15	Ревека Польна Августинівна	10	Макар Нагірний Олександрович	Лікар-офтальмолог вищої кваліфікаційної ...	Діагностика зору для дітей	1000 грн
16	Ревека Польна Августинівна	10	Щедрогост Штокало Полянович	Дитячий офтальмолог	Лазерна корекція	29000 грн
17	Юрій Галушка Чеславович	10	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
18	Ладимир Томащук Охримович	10	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
19	Ладимир Томащук Охримович	10	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Діагностика зору для дітей	1000 грн
20	Тамара Григорович Янівна	10	Ян Соколенко Найденович	Лікар-офтальмолог вищої кваліфікаційної ...	Оптична когерентна томографія сітківки	800 грн
21	Тамара Григорович Янівна	10	Ян Соколенко Найденович	Лікар-офтальмолог вищої кваліфікаційної ...	Лазерна корекція	29000 грн

Рисунок 3.6 - Приклад виконання запиту “Запит з умовами”

3.5 Запит з умовами. Інформація пацієнта за статтю, віком і досвідом лікаря.

Цей запит (рис. 3.7) сортує інформацію пацієнта за різними показниками, в результаті виводячи ПІБ пацієнта, його стать, вік, дату візиту, а також лікаря та його досвід.

SELECT p.p_name || ' ' || p_surname || ' ' || p_middle_name 'Patient', p.p_sex 'Gender', p.p_age 'Age', v.v_date 'Visit', d.d_name || ' ' || d_surname 'Doctor', d.d_experience 'Experience'

From Patient p join Visit v on p.p_ID=v.v_p_ID join Doctor d on d.d_ID=v.v_d_ID WHERE p.p_sex='Ч' AND d.d_experience>5 AND p.p_age<10 ORDER by p_age ASC;

В запиті використовується команда WHERE і AND яка дозволяє виконувати різні умови одночасно. За допомогою цього відбувається сортування таблиці пацієнтів за статтю де вказується що будуть виводитися одні чоловіки(p.p_sex='Ч'), за віком який повинен бути менше 10(p.p_age<10) і за досвідом лікаря що приймає пацієнта який повинен бути більше 5 років(d.d_experience>5).

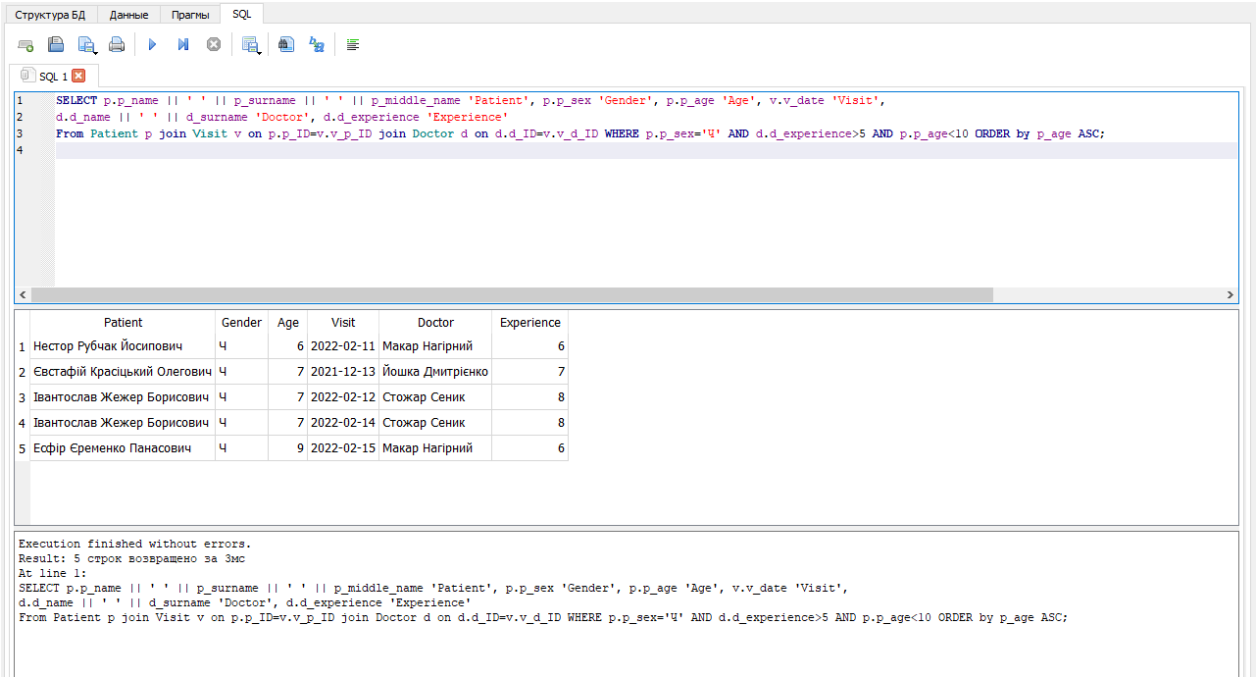


Рисунок 3.7 - Приклад виконання запиту “Запит з умовами”

3.6 Два різні способи виводу відповідних даних. Загальні витрати пацієнтів за весь час що відвідували клініку хоча б 2 рази.

Перший спосіб запит в запиті(рис. 3.8).

За допомогою відповідних команд видяться дані пацієнта його рік народження, кількість візитів і сумарна сума витрат за різні послуги клініки.

```
Select t1.Patient, t1.Date, count(t1.ID) as 'Visit', SUM(t1.Price) as 'Money'
from
(Select p.p_ID as 'ID',
 p.p_name || ' ' || p.p_surname || ' ' || p.p_middle_name as 'Patient',
s.s_description, s.s_price as 'Price',
strftime('%Y',p_birth_date)'Date'
from
Visit v join patient p on
v.v_p_ID=p.p_ID join service s on s.s_ID=v.v_s_ID order by p.p_ID)
t1 GROUP by t1.ID HAVING count(ID)>=2
ORDER by SUM(t1.Price) DESC;
```

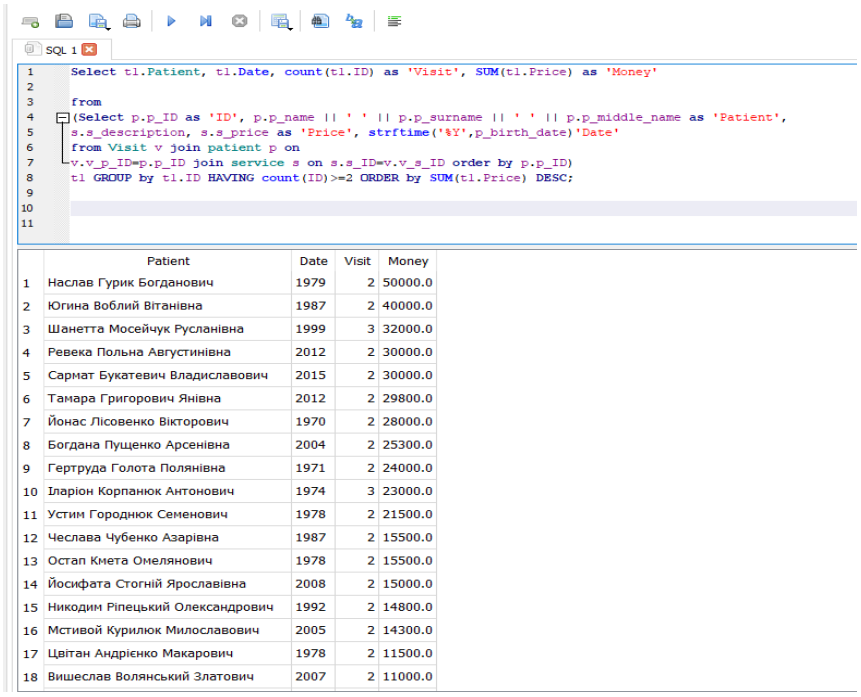
В запиті (рис. 3.8) можна побачити підзапит – він використовується для повернення даних, які будуть використовуватися в основному запиті як умова подальшого обмеження даних, що підлягають вилученню. Також наявні різні команди і функції:

1. COUNT підраховує кількість записів у таблиці.
2. SUM підсумовує значення вказаного поля по всіх вибраних рядках.

3. Strftime це функція яка повертає дату, відформатовану відповідно до рядка, зазначеного в аргументі першим.

4. GROUP BY дозволяє групувати результати під час вибірки з бази даних.

5. HAVING дозволяє фільтрувати результат угрупування, зробленого за допомогою команди GROUP BY.



	Patient	Date	Visit	Money
1	Наслав Гурик Богданович	1979	2	50000.0
2	Югина Воблий Вітанівна	1987	2	40000.0
3	Шанетта Мосейчук Русланівна	1999	3	32000.0
4	Ревека Польна Августинівна	2012	2	30000.0
5	Сармат Букатевич Владиславович	2015	2	30000.0
6	Тамара Григорович Янівна	2012	2	29800.0
7	Йонас Лісовенко Вікторович	1970	2	28000.0
8	Богдана Пущенко Арсенівна	2004	2	25300.0
9	Гертруда Голота Полянівна	1971	2	24000.0
10	Іларіон Корпанюк Антонович	1974	3	23000.0
11	Устим Городнюк Семенович	1978	2	21500.0
12	Чеслава Чубенко Азарівна	1987	2	15500.0
13	Остап Кмета Омелянович	1978	2	15500.0
14	Йосифата Стогній Ярославівна	2008	2	15000.0
15	Никодим Ріпецький Олександрович	1992	2	14800.0
16	Мстивой Курилюк Милославович	2005	2	14300.0
17	Цвітан Андрієнко Макарович	1978	2	11500.0
18	Вишеслав Волянський Златович	2007	2	11000.0

Рисунок 3.8 - Приклад виконання запиту “Запит в запиті”

Другий спосіб виводу даних(рис. 3.9).

В цьому способі показано приклад такого ж запиту але без використання можливостей підзапиту що являє собою простіший процес написання але урізані можливості ніж в першому способі.

SELECT p.p_name || ' ' || p.p_surname || ' ' || p.p_middle_name 'Patient',

strftime('%Y',p_birth_date)'Date',count(v.v_p_ID) as 'Number of visit',

```
SUM(s_price) as sum
FROM Patient p join Visit v on p.p_ID=v_p_ID
join Service s on v_s_ID=s_ID
GROUP BY p.p_ID HAVING count(v_p_ID)>=2
order by SUM(s_price) DESC;
```

Звернувши увагу на останній рядок коду можна побачити властивість сортування DESC яка дозволяє сортувати вказані дані в порядку спадання.

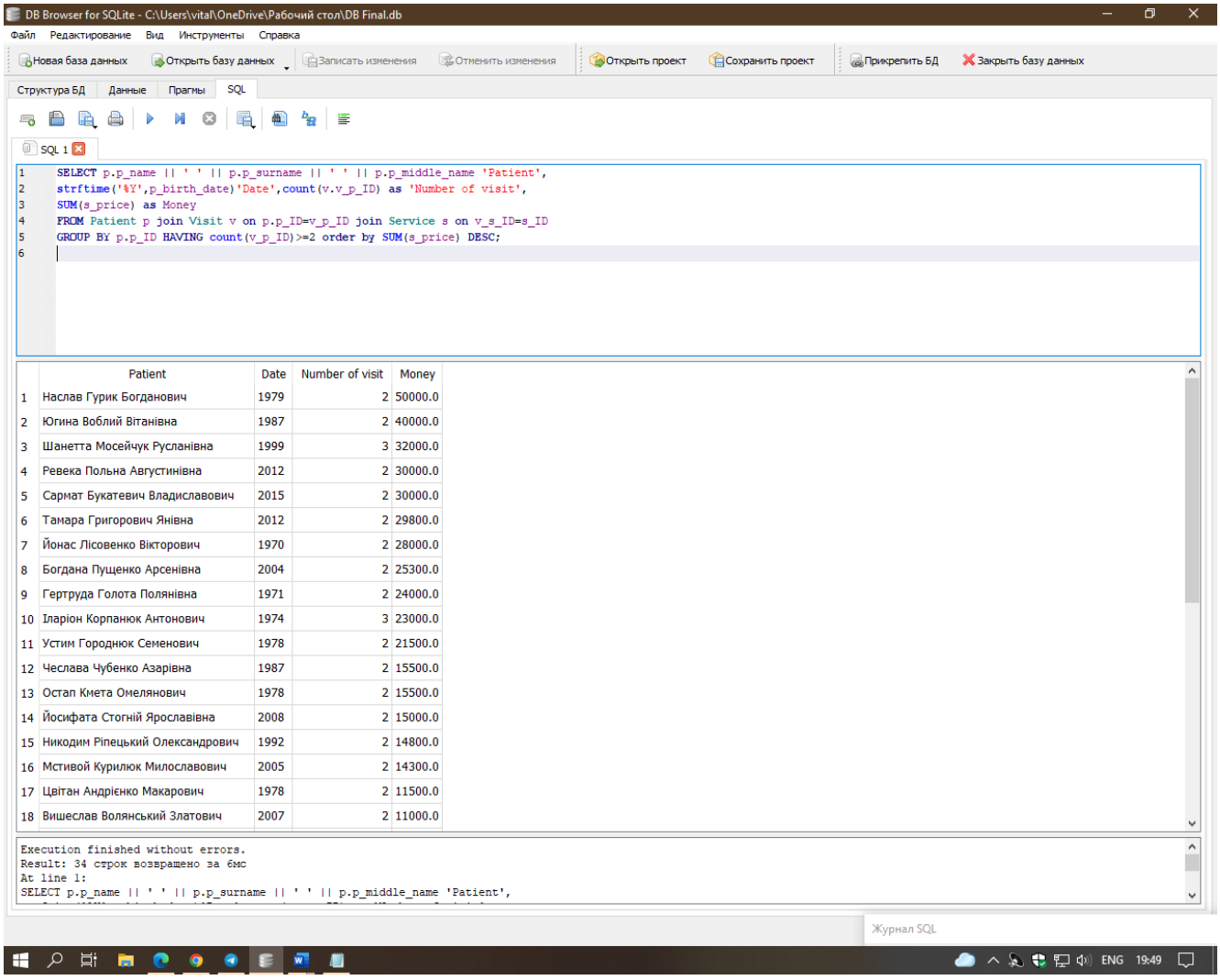


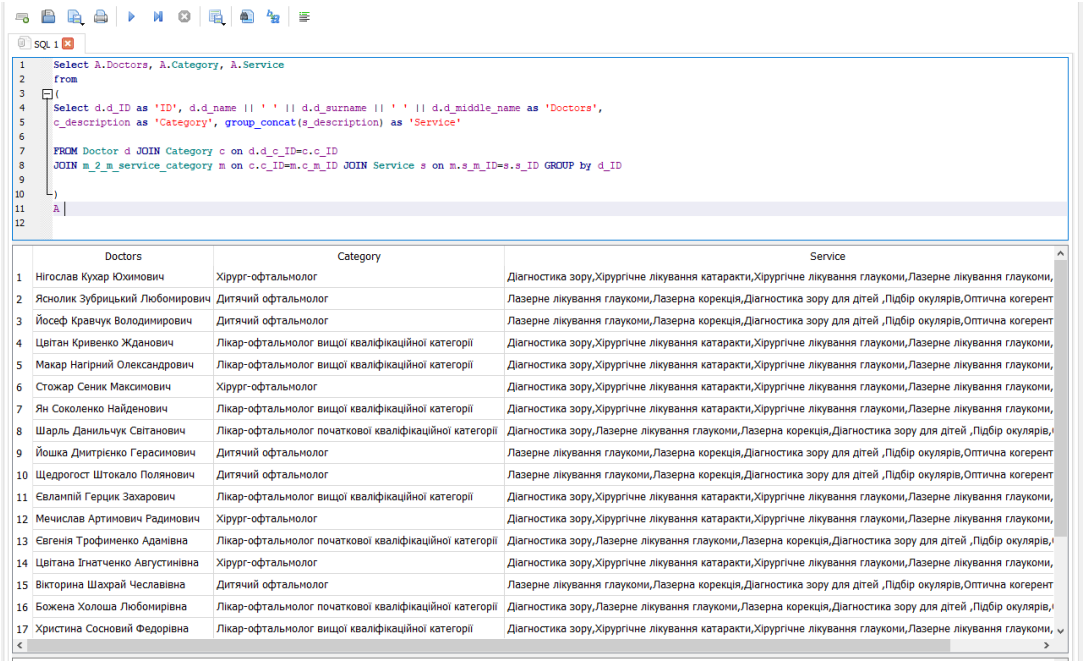
Рисунок 3.9 - Приклад виконання запиту “Запит з умовами”

3.7 Запит в запиті або підзапит. Повні можливості лікарів.

В цьому запиті (рис. 3.10) виводиться повна інформація можливостей лікарів а саме: прізвище, ім’я, по-батькові, спеціалізація лікаря і всі послуги над якими він може працювати.

Select A.Doctors, A.Category, A.Service From (Select d.d_ID as 'ID', d.d_name || ' ' || d.d_surname || ' ' || d.d_middle_name as 'Doctors', c_description as 'Category', group_concat(s_description) as 'Service' FROM Doctor d JOIN Category c on d.d_c_ID=c.c_ID JOIN m_2_m_service_category m on c.c_ID=m.c_m_ID JOIN Service s on m.s_m_ID=s.s_ID GROUP by d_ID) A;

Використовуючи підзапит для розширення можливостей наповнення основного запиту я також використав функцію group_concat вона є агрегатною тобто діє по відношенню до значень стовпця з метою отримання єдиного результуючого значення, якщо простіше то об’єднує всі ненульові значення в стовпці.



	Doctors	Category	Service
1	Нігослав Кухар Юхимович	Хірург-офтальмолог	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
2	Яснолик Зубрицький Любомирович	Дитячий офтальмолог	Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,Оптична когерент
3	Йосеф Кравчук Володимирович	Дитячий офтальмолог	Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,Оптична когерент
4	Цвітан Кривенко Жданович	Лікар-офтальмолог вищої кваліфікаційної категорії	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
5	Макар Нагірний Олександрович	Лікар-офтальмолог вищої кваліфікаційної категорії	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
6	Стожар Сеник Максимович	Хірург-офтальмолог	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
7	Ян Соколенко Найденович	Лікар-офтальмолог вищої кваліфікаційної категорії	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
8	Шарль Данильчук Світанович	Лікар-офтальмолог початкової кваліфікаційної категорії	Діагностика зору,Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,
9	Йошка Дмитрієнко Герасимович	Дитячий офтальмолог	Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,Оптична когерент
10	Щедрогост Штокало Полянович	Дитячий офтальмолог	Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,Оптична когерент
11	Євлампій Герцик Захарович	Лікар-офтальмолог вищої кваліфікаційної категорії	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
12	Мечислав Артимович Радимович	Хірург-офтальмолог	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
13	Євгенія Трофименко Адамівна	Лікар-офтальмолог початкової кваліфікаційної категорії	Діагностика зору,Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,
14	Цвітана Ігнатченко Августинівна	Хірург-офтальмолог	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,
15	Вікторина Шахрай Чеславівна	Дитячий офтальмолог	Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,Оптична когерент
16	Божена Холоша Любомирівна	Лікар-офтальмолог початкової кваліфікаційної категорії	Діагностика зору,Лазерне лікування глаукоми,Лазерна корекція,Діагностика зору для дітей ,Підбір окулярів,
17	Христина Сосновий Федорівна	Лікар-офтальмолог вищої кваліфікаційної категорії	Діагностика зору,Хірургічне лікування катаракти,Хірургічне лікування глаукоми,Лазерне лікування глаукоми,

Рисунок 3.10 - Приклад виконання запиту “Запит в запиті”

3.8 Запит з об’єднанням. Інформація пацієнтів, які були в певні періоди.

В цьому (рис. 3.11) запиті використовуючи команду об’єднання я вивів список пацієнтів які відвідували клініку в певний проміжок часу, також інформацію по даті візиту, номеру оператора пацієнта, їх стать і послугу на яку вони прийшли.

```
Select substr(p_name,1,1)|| '. ' || substr(p_middle_name,1,1)|| '. ' || p.p_surname 'Patient name', v_date 'Visit date',

rtrim(p_phone_number,'0123456789-')'Operator number',

p_sex 'Gender', s_description 'Service'

from Patient p join visit v on v.v_p_ID = p.p_ID

join Service s on v.v_s_ID=s_ID where date(v_date) < '2021-11-20'

UNION

Select substr(p_name,1,1)|| '. ' || substr(p_middle_name,1,1)|| '. ' || p.p_surname 'Patient name', v_date 'Visit date',

rtrim(p_phone_number,'0123456789-')'Operator number',

p_sex 'Gender', s_description 'Service'

from Patient p join visit v on v.v_p_ID = p.p_ID

join Service s on v.v_s_ID=s_ID

where date(v_date) > '2022-02-25'

ORDER by v_date;
```

Використовуючи команду об’єднання UNION грубо кажучи потрібно об’єднати два однакових але різних за деякими властивостями запита, в першій частині я вказав показати пацієнтів які відвідували клініку до 2021-11-20 а в другій частині показати відвідування від 2022-02-25.

В запиті використовуються різні команди:

1. Select, звичайна команда для отримання таблиць з бази даних, використовуючи її та позначення || '. ' || ми отримуємо відповідні таблиці з даними.
2. Substr, функція яка дозволяє починати рядок з вказаної позиції, із попередньо визначеною довжиною (substr(p_name,1,1)|| '. ' ||), в даному випадку використовуючи таблицю p_name я вказую з якої букви потрібно починати виводити інформацію (p_name,1,1) тобто з першої а також вказую довжину рядка (p_name,1,1) тобто довжина в один символ. Користуючись позначенням || '. ' || об'єдную рядки (substr(p_name,1,1)|| '. ' || substr(p_middle_name,1,1)|| '. ' || p.p_surname 'Patient name') формуючи таблицю 'Patient name' яка в результаті буде виглядати як ім'я, прізвище та по-батькові пацієнта але у відповідній абрівіатурі.
3. Rtrim, функція яка дозволяє видаляти відповідні символи в кінці рядка(rtrim(p_phone_number,'0123456789-')). В даному випадку обираючи таблицю p_phone_number я завдаю відповідні значення які потрібно прибрати '0123456789-' і в результаті з повного номеру телефону 380(99)335-66-87 я отримую лише 380(99) і ця таблиця в запиті має назву 'Operator number'.
4. From звичайна функція яка допомагає брати інформацію з бази даних.
5. DATE, ця функція отримує дату з відповідної таблиці (date(v_date)).
6. Where функція визначає умову за допомогою якої будуть обиратися рядки з бази даних(where date(v_date) > '2022-02-25'). В запиті команда WHERE звертається до функції DATE задаючи певну умову що виводяться дані в яких дата більша ніж вказана.
7. ORDER by звичайна функція за допомогою якої відбувається сортування по заданій таблиці в порядку зростання.
8. UNION головна команда цього запиту яка об'єднує два запита з різними вимогами в один.

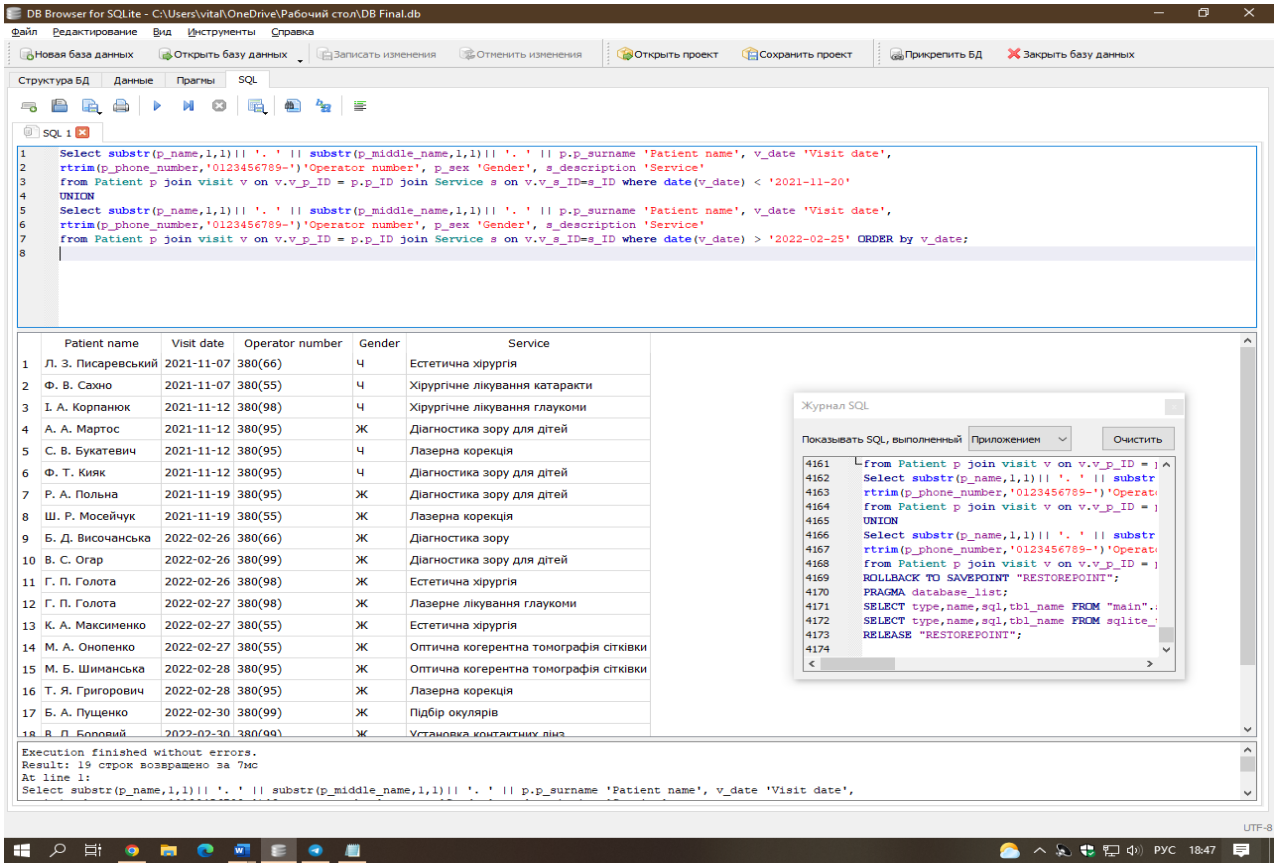


Рисунок 3.11 - Приклад виконання запиту “Запит з об’єднанням”

РОЗДІЛ 4. ПРАВИЛА ТЕХНІКИ БЕЗПЕКИ ПРИ РОБОТІ ЗА КОМП'ЮТЕРОМ

Працюючи за комп'ютером, рекомендовано дотримуватися правил:

- Відстань стільця до столу так щоб між краєм столу та Вами був вільний простір не менше 5 см, налаштування висоти стільця для комфорту під час роботи.

- Коли працюєш з комп'ютером потрібно не забувати держати спину в рівному положенні.

- Плечі повинні бути розслаблені.

- Сидячи за комп'ютером ноги не повинні звисати, вони обов'язково мають опиратися на підлогу або якщо стілець високий то на спеціальну платформу, підставку; використовуючи підлокітники потрібно впевнитися що вони під прямим кутом.

- При використанні клавіатури потрібно перед роботою впевнитися що відстань між вами та клавіатури приблизно від 10 до 25 сантиметрів, також у конструкції клавіатури повинні передбачатися опорні пристрої які можуть змінювати кут нахилу від 5 і до 15 градусів.

- При роботі з монітором потрібно слідкувати за тим щоб відстань від очей до екрану була не менше ніж 60 сантиметрів. У випадку коли центр екрану зміщений то можуть виникнути проблеми у вигляді "синдрому сухого ока", для уникнення таких наслідків рекомендується при тривалих роботах за комп'ютером робити не великі перерви які допоможуть відпочити та зволожити ваші очі також після закінчення роботи з комп'ютером треба не забувати робити спеціальну гімнастику для очей.

- Бувають випадки коли замість монітору використовують повноцінний телевізор, але це категорично не рекомендовано так як випромінювання телевізора в багато разів більше ніж від монітору, телевізор завжди був призначений для перегляду на відстані і при роботі на короткій дистанції з часом виникнуть великі проблеми з зором.

- Працюючи за ПК час роботи без перерви не повинен перевищувати двох годин.

- Працюючи з текстовими документами для більш продуктивної роботи колір шрифту повинен бути темного кольору, а колір фону повинен бути світлого кольору при цьому щоб не напружувати очі шрифт не повинен бути занадто мілким.

- За робочим місцем де знаходиться комп'ютер щодня потрібно доглядати і виконувати вологе прибирання;

- Працюючи за комп'ютером треба завжди пам'ятати про провітрювання приміщення, рекомендовано це робити щогодини;

Рекомендації щодо безпеки під час роботи на комп'ютері:

Працюючи за ПК необхідно слідкувати за станом монітора: він повинен бути в прилежному вигляді, без пилу та плям на екрані. При роботі з використанням окулярів (комп'ютерних чи звичайних) також потрібно зберігати їх в чистоті. Уникати контакту з проводами живлення і заземлення, задніх стінок комп'ютера чи кабелів. Уразі виникнення аварійної ситуації необхідно негайно вимкнути ПК від електропостачання.

Вимоги до освітленості приміщень поряд з персональним комп'ютером :

- Працюючи за комп'ютером освітленість місця повинна відповідати розмірам того об'єкта який треба роздивитися;

- Кімната де знаходиться робоче місце повинна мати природне та штучне освітлення при чому не тільки за комп'ютером а і за його межами;

- На робочій поверхні різкі тіні так само як і відблиски можуть заважати та напружувати зір тому їх бути не повинно;

- Розміщення вікон в кімнаті повинно бути переважно на північ або північний схід. Рекомендовано обладнати вікна занавісками або жалюзі для регулювання освітлення;

- В любий час дня при роботі з комп’ютером слід дотримувати оптимальну спрямованість світлового потоку та склад світла. Освітлення робочої поверхні столу повинна бути не менше 400 лк а освітленість екрану не повинна бути більше за 200 лк.

У випадку коли на робочому місці використовується в основному тільки штучне освітлення то рекомендовано використовувати різного роду лампи, наприклад люмінесцентні, але використовувати штучне освітлення без розсіювачів та екрануючих решіток не рекомендовано.

ВИСНОВКИ

В сучасному світі дуже актуальною є тема автоматизації або комп’ютеризації всіх видів діяльності. Це спрощує роботу з різними видами інформацією, так як для її зберігання або ведення не потрібно зберігати велику кількість паперових документів, з якими складно працювати, які можуть загубитися тощо. Достатньо лише ввімкнути комп’ютер та запустити необхідний додаток.

Як можна дізнатися мови за допомогою яких маніпулюють даними були створені ще за довго до появи перших реляційних баз даних. Вони були розроблені для різних систем управління базами і для виконання операцій з даними які представлені у вигляді файлу. Це потребувало від користувачів різних знань в організації, зберіганні та розміщені даних та знань як покроково їх отримати.

Розглянута мова SQL має напрямок для здійснення операцій з даними які мають вигляд логічно взаємозв’язаних сукупностей таблиць. Особливості цієї мови у тому, що вона орієнтується зазвичай на кінцевий результат обробки інформації, а не на процес обробки. SQL як би сам обирає, де саме знаходяться дані, які послідовності операцій, індексів ефективніші і як саме слід використовувати їх для отримання цих даних.

При написанні даної роботи я розглянув структуру мови SQL, працював з базою даних офтальмологічної клініки “ЦЕНТР ЗОРУ”, обробляв дані цієї клініки створюючи різні запити та на прикладі показав як це працює. Показано, область застосування та основні функції мови SQL, застосунку "Database Browser for SQLite". В сучасний час мова SQL є стандартом для використання в реляційних базах даних. Перспективи розвитку баз даних дуже великі і ці перспективи розвиваються в різних сферах діяльності як в медицині так і в повсякденному житті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Дейт К.Дж. Введение в системы баз данных. 6-е изд. - М.: Вильямс. (2000).

[2] В.В. Фаронов Основы программирования в SQL. - М.: Издатель Молгачева С.В., (2002).

[3] SQL. Сборник рецептов. Энтони Молинаро (2009).

[4] SQL: Руководство по изучению языка. Крис Фиайли (2013).

[5] Алан Бьюли "Изучаем SQL" (2007).

[6] Алекс Кригель и др. "SQL. Библия пользователя", 2-е издание (2010).

[7] Дейт К. Дж. Введение в системы баз данных, 8-е издание. . - М.: Издательский дом "Вильямс", (2006).

[8] Кузнецов М. В., Симдянов И. В. MySQL5.- СПб.: БХВ-Петербург, (2006).

[9] Моисеенко С. И. SQL. Задачи и решения. - СПб.: Питер, (2006).

[10] The Definitive Guide to SQLite by Grant Allen, Mike Owens

[11] Форта Б. SQL за 10 минут / Бен Форта. – 4-е изд. – М.: ООО “И.Д. Вильямс”, (2014).

[12] SQLite Database System Design and Implementation (2015) by Sibsankar Haldar

[13] Введення в базу даних Sqlite - Yoip. Yoip. URL: http://yoip.com.ua/vvedennya-v-bazu-danih-sqlite/

[14] Интерактивные онлайн-курсы HTML Academy. HTML Academy. URL: https://htmlacademy.ru/tutorial/php/sql

[15] Курсы SQL - обучение основ SQL для начинающих с нуля на itProger. itProger - Сообщество программистов. URL: https://itproger.com/course/sql

[16] Основные команды SQL, которые должен знать каждый программист. Tproger. URL: https://tproger.ru/translations/sql-recap/

[17] Примеры SQL запросов в sqlite - SpecialistOff.NET. SpecialistOff.NET. URL: https://specialistoff.net/question/574

[18] Синтаксис языка запросов SQLite. IT блоги по программированию и разработке для программистов. URL: https://unetway.com/tutorial/sqlite-syntax

[19] Создание таблицы в базе данных sqlite. URL: https://zametkinapolyah.ru/zametki-o-mysql/chast-10-1-sozdanie-tablic-v-bazax-dannyx-sqlite.html

[20] Учасники проектів Вікімедіа. Реляційна база даних – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Реляційна_база_даних

[21] Учасники проектів Вікімедіа. SQLite – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/SQLite

[22] Что такое SQL. Назначение и основа | Info-Comp.ru - IT-блог для начинающих. Заметки IT специалиста. URL: https://info-comp.ru/programmirovanie/749-what-is-sql.html

[23] Що таке sqlite? - визначення з техопедії - Бази даних - 2022. Icy Science. URL: https://uk.theastrologypage.com/sqlite

[24] ER-модель. URL: https://ua.wikipedia.org/wiki/ER-модель

[25] SQLITE команди URL: http://old.code.mu/sql/union.html

[26] SQL запроси URL: https://zametkinapolyah.ru/zametki-o-mysql/tema-12-sql-zaprosy-select-v-bazax-dannyx-sqlite.html

[27] SQLite | Введение. METANIT.COM - Сайт о программировании. URL: https://metanit.com/sql/sqlite/1.1.php

[28] SQLite типы запросов. URL: https://oracleplsql.ru/sql-query-types-sqlite.html

[29] SQLite Home Page. URL: https://www.sqlite.org/index.html

[30] SQLite Tutorial - An Easy Way to Master SQLite Fast. SQLite Tutorial. URL: https://www.sqlitetutorial.net/

[31] SQLite. URL: https://lecturesdb.readthedocs.io/databases/sqlite.html