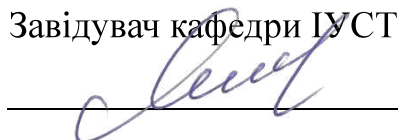


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття ступеня вищої освіти "магістр"

на тему:

**Системний аналіз засобів адміністрування та проєктування
утиліти для СУБД SQLite**

Виконав: студент групи 6145м



Ткаченко І.М.

(підпис)

Керівник роботи:

Д.Т.Н. ДОЦЕНТ

(посада, науковий ступень вчене звання)



Казимиренко Ю.О.

(підпис)

м. Миколаїв – 2025р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала
Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 124 Системний аналіз
Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми

д.т.н., проф. Ушкац М.В.

"___" "___" 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття ступеня вищої освіти «магістр»

Студенту Ткаченко Івану Миколайовичу

1. Тема роботи: **Системний аналіз засобів адміністрування та проєктування утиліти для СУБД SQLite**

Керівник роботи д.т.н., доц., проф. каф-ри Казимиренко Юлія Олексіївна

Затверджені наказом ректора №1355-уч від «14» листопада 2025 року

2. Термін подання роботи до 15 грудня 2025 р.

3. Вихідні дані по роботі: документація на аналоги систем, з організації баз даних, наукові праці з тематики досліджень.

4. Перелік питань, що належать до розробки (найменування розділів):

Розділ 1. Аналіз проблемних питань з проєктування утиліт для засобів адміністрування. Розділ 2. Дослідження системи управління базами даних SQLite. Розділ 3. Проєктування утиліти на основі результатів системного аналізу засобів адміністрування. Розділ 4. Техніко-економічний аналіз розробки. Розділ 5. Охорона праці. Розділ 6 Охорона навколишнього середовища.

5. Перелік презентаційних матеріалів виконаний в програмі Power Point: 1. Титульний слайд. 2. Мета і завдання роботи. 3. Аналіз проблемних питань з тематики дослідження. 4. Функціональні вимоги до утиліти. 5. Нефункціональні вимоги до утиліти. 6. UML-діаграма використання. 7. Концептуальна модель даних. 8. Проєкт інтерфейсу. 9. Приклад реалізації. 10. Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Казимиренко Ю.О., д.т.н., проф.	Казимиренко Ю.О.	Ткаченко І.М.
2	Казимиренко Ю.О., д.т.н., проф.	Казимиренко Ю.О.	Ткаченко І.М.
3	Казимиренко Ю.О., д.т.н., проф.	Казимиренко Ю.О.	Ткаченко І.М.
4	Казимиренко Ю.О., д.т.н., проф.	Казимиренко Ю.О.	Ткаченко І.М.
5	Казимиренко Ю.О., д.т.н., проф.	Казимиренко Ю.О.	Ткаченко І.М.
6	Казимиренко Ю.О., д.т.н., проф.	Казимиренко Ю.О.	Ткаченко І.М.

7. Дата видачі завдання 05 вересня 2024 р.**КАЛЕНДАРНИЙ ПЛАН**

№	Назва етапів роботи	Строк виконання етапів роботи	Примітка
1	Аналіз проблемних питань з проектування утиліт для засобів адміністрування	13.11.2025	Виконано
2	Дослідження системами управління базами даних SQLite	22.11.2025	Виконано
3	Проектування утиліти на основі результатів системного аналізу засобів адміністрування	29.11.2025	Виконано
4	Техніко-економічний аналіз розробки	05.12.2025	Виконано
5	Охорона праці	05.12.2025	Виконано
6	Охорона навколишнього середовища	05.12.2025	Виконано
7	Формулювання висновків, переліку використаних літературних джерел	08.12.2025	Виконано
8	Оформлення пояснювальної записки та слайд-презентацій	11.12.2025	Виконано
9	Подання магістерської роботи на перевірку на плагіат	15.12.2025	Виконано
10	Подання магістерської роботи рецензенту	15.12.2025	Виконано
11	Подання магістерської роботи на державний захист	22.12.2025	Виконано

Студент


 (підпис)
Ткаченко І.М.

(ПІБ)

Керівник роботи


 (підпис)
Казимиренко Ю.О.

(ПІБ)

АНОТАЦІЯ

Ткаченко І.М. Системний аналіз засобів адміністрування та проєктування утиліти для СУБД SQLite. Магістерська робота за спеціальністю 124: Системний аналіз, освітня програма «Системний аналіз» Національний університет кораблебудування імені адмірала Макарова. Миколаїв, 2025.

Робота присвячена вирішенню проблемних питань розвитку методологічного підґрунтя на основі системного підходу до розв’язання практичних задач з проєктування утиліти для СУБД SQLite.

Мета роботи – розроблення проєкту утиліти для СУБД SQLite на основі результатів системного аналізу засобів адміністрування.

Об’єкт дослідження – процеси адміністрування та керування даними в автономних системах баз даних.

Предметом дослідження є утиліта для СУБД SQLite.

Науково-практичне значення отриманих результатів полягатиме у спрощених моделях для проєктування утиліти, в основу розробки і дослідження яких покладено результати системно-цільового і системно-функціонального аналізу управління реляційними базами даних.

Методи дослідження: системно-цільовий аналіз, структурно-функціональний аналіз, методи і технології проєктування баз даних.

Робота написана на 64 сторінках друкованого тексту, містить 10 рисунків, 17 таблиці, додатки і список використаних джерел з 25 найменувань.

Ключові слова: системно-цільовий аналіз, системно-функціональний аналіз, реляційна база даних, ІТ-проєкт, система

ABSTRACT

Tkachenko I.M. System analysis of the tools for administration and design of the utility for the SQLite DBMS. Master's thesis in the specialty 124: System Analysis, educational program "System Analysis", Admiral Makarov National University of Shipbuilding, Mykolaiv, 2025.

The work is devoted to addressing the problematic issues of developing the methodological basis through a systematic approach to solving practical problems in designing a utility for the SQLite DBMS.

The purpose of the work is to develop a draft utility for the SQLite DBMS based on the results of system analysis of administration tools.

The object of research is the processes of data administration and management in autonomous database systems.

The subject of the study is the SQLite DBMS utility.

The scientific and practical significance of the results obtained will lie in simplified models for the design of the utility, the development and research of which is based on the results of system-target and system-functional analysis of relational database management.

Research methods: system-target analysis, structural-functional analysis, methods and technologies of database design.

The work is written on 64 pages of typewritten text, contains 10 figures, 17 tables, appendices and a list of used sources from 25 titles.

Keywords: system-target analysis, system-functional analysis, relational database, IT project, system.

ЗМІСТ

ВСТУП.....	9
РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМНИХ ПИТАНЬ З ПРОЄКТУВАННЯ УТИЛІТ ДЛЯ ЗАСОБІВ АДМНІСТРУВАННЯ.....	10
1.1. Засоби адміністрування: системний підхід до розробки і реалізації	10
1.2. Задачі проєктування утиліт для роботи з базами даних	144
1.3. Аналіз проблемних питань з тематики дослідження операцій	17
1.4. Висновки за розділом 1 і постановка завдань досліджень	19
РОЗДІЛ 2 ДОСЛІДЖЕННЯ СИСТЕМИ УПРАВЛІННЯ БАЗАМИ ДАНИХ SQLite.....	21
2.1. Вибір і обґрунтування методів і засобів дослідження	21
2.2. Обґрунтований вибір мови програмування.....	23
2.3. Аналіз процесів взаємодії користувача з базою даних	24
2.4. Висновки за розділом 2	28
РОЗДІЛ 3 ПРОЄКТУВАННЯ УТИЛІТИ НА ОСНОВІ РЕЗУЛЬТАТІВ СИСТЕМНОГО АНАЛІЗУ ЗАСОБІВ АДМІНІСТРУВАННЯ.....	30
3.1. Розробка архітектури програмної утиліти	30
3.2 Використання бібліотек SQLite (sqlite3, contextlib, os, SQLAlchemy тощо)	33
3.3. Розробка інтерфейсу	35
3.4. Апробація і реалізація утиліти.....	40
3.5 Висновки за розділом 3	44
РОЗДІЛ 4 ТЕХНІКО-ЕКОНОМІЧНИЙ АНАЛІЗ РОЗРОБКИ.....	45
4.1 Загальні положення	45
4.2 Розрахунок витрат на створення і впровадження утиліти	45

4.3 Витрати на впровадження з урахуванням вартості навчання кадрів і витрати на монтаж і настроювання.	48
4.4 Економічна ефективність розробки і впровадження	499
4.5 Висновки за розділом 4	50
РОЗДІЛ 5 ОХОРОНА ПРАЦІ.....	51
5.1 Вступ	51
5.2 Основні небезпечні та шкідливі виробничі фактори	52
5.3 Оцінка ризиків для здоров'я працівників.....	52
5.4 Забезпечення сприятливих умов праці при роботі з персональним комп'ютером	53
5.5 Висновки за розділом 5.	54
РОЗДІЛ 6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА.....	55
6.1 Вступ	55
6.2 Правова охорона навколишнього середовища у промисловості	56
6.3 Висновки за розділом 6.	57
ЗАГАЛЬНІ ВИСНОВКИ	58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	59
ДОДАТОК ФРАГМЕНТИ ЛІСТИНГУ	62

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БАД – багатофакторний аналіз даних

БД – база даних

БШ – багатовимірне шкалювання

ІС – інформаційна система

ІТ – інформаційні технології

РБД – реляційна база даних

СУБД – система управління базами даних

СФА – системно-функціональний аналіз

СЦА – системно-цільовий аналіз

ВСТУП

Актуальність дослідження зумовлена складністю процесів адміністрування баз даних у сучасних інформаційних системах та необхідністю розроблення утиліти для спрощення виконання базових операцій.

Мета роботи – розроблення проєкту утиліти для СУБД SQLite на основі результатів системного аналізу засобів адміністрування.

Для виконання поставленої мети в роботі необхідно виконати наступні **завдання**:

- 1) проаналізувати проблемні питання з проєктування утиліт для засобів адміністрування;
- 2) сформулювати вимоги до утиліти для СУБД SQLite;
- 3) розробити проєкт на основі результатів системного аналізу засобів адміністрування;
- 4) оцінити ефективність впровадження утиліти для СУБД SQLite;
- 5) розглянути заходи щодо охорони праці та навколишнього середовища.

Об’єкт дослідження – процеси адміністрування та керування даними в автономних системах баз даних.

Предметом дослідження є утиліта для СУБД SQLite.

Науково-практичне значення отриманих результатів полягатиме у спрощених моделях для проєктування утиліти, в основу розробки і дослідження яких покладено результати системно-цільового і системно-функціонального аналізу управління реляційними базами даних.

Методи дослідження: системно-цільовий аналіз, структурно-функціональний аналіз, методи і технології проєктування баз даних.

Робота пройшла апробацію на науково-методичних семінарах кафедри Інформаційних управляючих систем та технологій Навчально-наукового інституту комп’ютерних наук та управління проектами.

РОЗДІЛ 1

АНАЛІЗ ПРОБЛЕМНИХ ПИТАНЬ З ПРОЄКТУВАННЯ УТИЛІТ ДЛЯ ЗАСОБІВ АДМІНІСТРУВАННЯ

1.1. Засоби адміністрування: системний підхід до розробки і реалізації

Проблему підвищення ефективності засобів адміністрування баз даних у сучасних інформаційних системах (ІС) розглянуто як підрозділ системного аналізу, науково-практичне підґрунтя якого викладено у роботах [1-4]. Застосований для цього системний підхід ґрунтується на принципах і закономірностях взаємодії «частини і цілого» [1]. Протиріччям методологій цілісного і системного підходів присвячено роботу авторів [2], у якій встановлено форми відношень між цілісного і системного підходів і вводяться і аналізуються поняття «цілісності» і «інтегративності» (табл .1.1)

Таблиця 1.1

Сутність понять щодо гіпотези дослідження

Поняття	Сутність
Цілісність	У системі з'являються властивості, не притаманні її елементам. Проте система розглядається як результат взаємодії елементів
Інтегративність	Сукупність системаутворюючих чинників, у яких важлива роль відводиться неоднорідності і суперечливості елементів

Джерело: складено авторами за аналізом роботи [2]

На рис. 1.2 у схематичному вигляді виокремлено сформульовані протиріччя системного і цілісного підходів з ймовірними шляхами їх розв'язання, що були запропоновані авторами роботи [2].



Рис. 1.1. Протиріччя методологій системного та цілісного підходів [2]

Наведена на рис. 1.1 схема далі застосовується у роботі для формування архітектури моделі структуризації знань. В основу системного аналізу засобів адміністрування ІС покладено зміст «принципів організації існування та діяльності», у яких виділено дві частини: система управління

і об'єкт управління. Ієрархічні принципи моделювання системи управління показані на рис. 1.2, з чого виходить, що дослідження поведінки системи та встановлення законів управління процесами ґрунтується на моделюванні причинно-наслідкових зв'язків, що далі розглянуто у Розділі 3 магістерської роботи.



Рис. 1.2. Порівняння методів діяльності системи [2]

У вигляді графа на рис. 1.3 наведено співвідношення протиріч і аксіом, які їх повністю розкриватимуть [2].



Рис. 1.3. Співвідношення протиріч і аксіом [2]

Наведені співвідношення розкриватимуть створений баланс між «організованим цілим та системою» [2] та визначатимуть у подальшому гіпотезу магістерського дослідження, для підтвердження якої будуть застосовані методи системно-цільового та структурно-функціонального

аналізу, теоретичні основи і практична реалізація розглянуті в роботах [5-11].

1.2. Задачі проєктування утиліт для роботи з базами даних

В роботі [12] аналізуються сучасні тенденції у технологіях баз і сховищ даних, висновки про які зроблено на підставі напрацювань і узагальнень інформації фахівцями Національної академії статистики, обліку та аудиту. Автори аналізують поточне навантаження систем управління базами даних (СУБД) і на підставі чого використовують методи data mining, доповнені матеріалізованими управліннями щодо впливу інших об'єктів баз даних (БД) на ефективність оброблення інформації. Серед головних факторів визначено:

- роль засобів самоаналізування задля допомоги адміністраторам БД з метою налаштування фізичної моделі баз даних та її адаптації у поточному середовищі використання;
- необхідність впровадження засобів моніторингу, що розширює можливості адміністраторів щодо розпізнавання власних недоліків;
- скорочення «управлінських важелів» щодо налаштування додатків з мінімальною участю експертів;
- необхідність переосмислення архітектури баз даних.

Системи управління базами даних СУБД MySQL використовуються у банківському, аграрному секторі, у торговельних операціях, фармації, про йдеться у роботах [7-13].

При проєктуванні СУБД враховуються принципи безпеки і цілісності даних, спільне використання даних, масштабованість з обробкою великого обсягу, контролю над надмірністю даних тощо. Переваги і недоліки SQL БД наведено у табл.1.2.

Переваги і недоліки SQL БД

Переваги	Недоліки
• мають визначний стандарт з можливістю керувати великим обсягом даних • ефективність роботи за рахунок застосування структурованих і організованих даних • підтримка лінійки програмних інструментів	• робота виключно зі структурованими даними • обмеження по гнучкості структури даних та масштабісттю • високі операційні витрати на збереження інформації, яка може бути також надмірною

Джерело: складено автором за аналізом статті [14]

Найпопулярніші типи баз даних SQL наведено на рис. 1.4 [14].

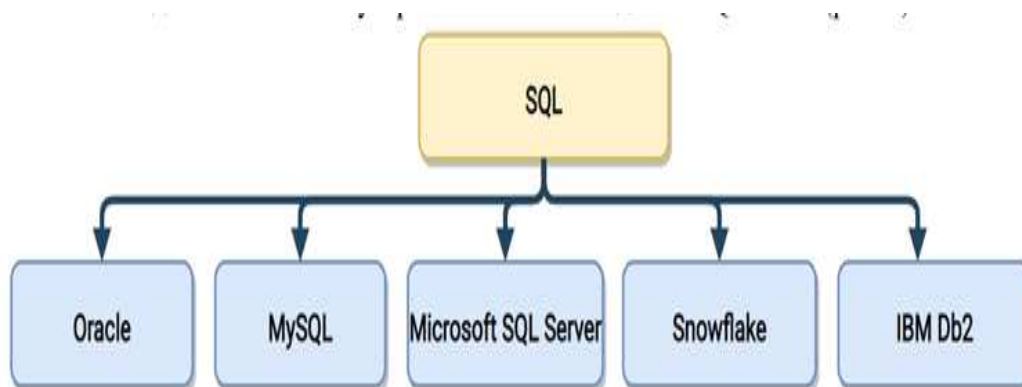


Рис. 1.4. Найбільш поширені типи баз даних SQL [14]

Найпопулярнішим видом баз даних є реляційні БД, які зберігають дані у табличній формі [15], для їх адміністрування перевага надається СУБД SQLite, що легко впроваджуються у інше програмне середовище. В роботі [13] авторами розглянуто досвід застосування системи моніторингу для сільськогосподарського підприємства на основі принципів геолокації, для якої використано базу даних, яка працює під СУБД SQLite. Це розширює можливості використовувати супутникові знімки, витяги з кадастрових

документів, результати обробки згенеровані у вигляді звітів і зберігаються в уніфікованому структурному форматі. Система дозволить аналізувати економічні і фінансові ризики, ґрунтуючись на «історичних» даних щодо обробленні полів.

Утилітарні завдання СУБД полягатимуть у створенні інструментів для автоматичного резервного копіювання даних; розробці форми запитів; створення системи управління правами адміністратора, користувача, редактора; розробці генераторів звітів; створення інструментів для резервного копіювання даних у автоматичному режимі. Вимоги до утиліти наведено у табл. 1.3.

Таблиця 1.3

Вимоги до утиліти

Вимоги	Реалізація
Створення інтерфейсу для користувачів	Створення форм для введення/перегляду даних, звітів для аналітики
Розробка засобів адміністрування	Функціонал для резервного копіювання, відновлення та управління доступом (паролі, дозволи).
Розробка застосунків	Створення програм, що взаємодіють з базами даних (запити, бізнес-логіка)
Забезпечення якості і безпеки	Надійність і стійкість: механізми резервного копіювання та відновлення, відмовостійкість.
	Конфіденційність: шифрування даних та паролів, контроль доступу, щоб дозволити доступ лише авторизованим користувачам

Джерело: складено автором за аналізом джерел [15, 16]

1.3. Аналіз проблемних питань з тематики дослідження операцій

Виконано ґрунтовний аналіз проблемних питань з тематики дослідження операцій. Узагальнені результати наведено у таблицях 1.4 і 1.5 і складаються з двох частин: перший етап – це виявлення проблемних питань з адміністрування СУБД (табл. 1.4). Це обмеження підтримка паралельного доступу, ризик пошкодження файлів БД, відсутність вбудованої системи автентифікації та авторизації.

Таблиця 1.4

Проблемні питання з адміністрування СУБД

Ціль системи	Проблемне питання	Причини виникнення проблеми	Наслідки для процесу адміністрування	Напрями розв'язання в утиліті
Забезпечення стабільної роботи БД	Обмежена підтримка паралельного доступу	Файлова архітектура SQLite; єдиний файл БД; один процес запису	Помилки блокування, зниження надійності, обмеження масштабування	Моніторинг блокувань, керування транзакціями, механізми черг запису
Підвищення надійності адміністрування	Ризик пошкодження файлу БД	Некоректне завершення роботи, апаратні збої, конфлікти доступу	Втрата даних, необхідність відновлення	Інтеграція перевірок цілісності, автоматизоване резервне копіювання
Забезпечення безпеки даних	Відсутність вбудованої системи автентифікації та авторизації	Орієнтація SQLite на локальне використання	Неможливість керування правами доступу на рівні СУБД	Реалізація логіки контролю доступу на рівні утиліти

Джерело: складено автором за оглядом літератури [11-16]

Це може призвести до помилок блокування, зниження надійності, обмеження масштабування через файлову архітектуру SQLite. Ризик пошкоджених файлів з бази даних зі втратою даних виникатиме через апаратні збої та конфлікти доступу. Безпека даних може бути забезпеченою шляхом подолання відсутності вбудованої системи автентифікації та авторизації. В табл. 1.5 наведено проблемні питання функціонування утиліти для СУБД SQLite.

Таблиця 1.5

Проблемні питання функціонування утиліти для СУБД SQLite

Ціль системи	Проблемне питання	Причини виникнення проблеми	Наслідки для процесу адміністрування	Напрями розв'язання в утиліті
Забезпечення конфіденційності	Відсутність стандартного шифрування	Мінімалістична архітектура SQLite	Ризик несанкціонованого доступу до файлу БД	Підтримка зовнішніх механізмів шифрування та захисту
Підвищення ефективності проектування БД	Обмежені засоби візуалізації структури	Відсутність штатних інструментів моделювання	Ускладнення аналізу схеми БД	Візуалізація таблиць, зв'язків, індексів
Оптимізація продуктивності	Обмежені можливості аналізу запитів	Спрощений механізм оптимізації SQLite	Низька ефективність складних запитів	Аналіз SQL-запитів, рекомендації з індексації
Ціль системи	Проблемне питання	Причини виникнення проблеми	Наслідки для процесу адміністрування	Напрями розв'язання в утиліті
Підтримка життєвого циклу даних	Відсутність централізованого резервного копіювання	Безсерверна архітектура	Ускладнення відновлення після збоїв	Реалізація «гарячого» та планового резервування
Автоматизація адміністрування	Висока залежність	Орієнтація SQLite на програмне використання	Підвищений ризик людських помилок	Автоматизація типових адміністративних сценаріїв

Джерело: складено автором за оглядом літератури [11-16]

Наведений у табл. 1.5 комплекс проблемних питань функціонування утиліти пов'язує цілі СУБД з напрямками розв'язання проблемних питань. Так, наприклад, забезпечення конфіденційності пов'язано з відсутністю стандартного шифрування через мінімалістичну архітектуру SQLite і може призвести до ризику несанкціонованого доступу до файлу Б. Це пропонується вирішувати через підтримку утиліти зовнішніх механізмів шифрування та захисту. Обмежені можливості аналізу запитів, без яких неможливий процес оптимізації системи і знижується ефективність складних запитів можна виправити шляхом застосування SQL-запитів та розробку рекомендацій з індексації.

1.4. Висновки за розділом 1 і постановка завдань досліджень

1. В основу системного аналізу засобів адміністрування інформаційної системи покладено зміст «принципів організації існування та діяльності», у яких виділено дві частини: система управління і об'єкт управління.

2. На підставі аналізу досвіду засобів адміністрування базами даних у банківському, аграрному секторі, у торговельних операціях, фармації сформульовано задачі проєктування утиліт для SQLite.

3. Проаналізовано проблемні питання з тематики дослідження операцій, визначено причини виникнення проблеми і наслідки процесу адміністрування.

Мета роботи – розроблення проєкту утиліти для СУБД SQLite на основі результатів системного аналізу засобів адміністрування.

Для виконання поставленої мети в роботі необхідно виконати наступні **завдання**:

1) проаналізувати проблемні питання з проєктування утиліт для засобів адміністрування;

- 2) сформулювати вимоги до утиліти для СУБД SQLite;
- 3) розробити проєкт на основі результатів системного аналізу засобів адміністрування;
- 4) оцінити ефективність впровадження утиліти для СУБД SQLite;
- 5) розглянути заходи щодо охорони праці та навколишнього середовища.

Об'єкт дослідження – процеси адміністрування та керування даними в автономних системах баз даних.

Предметом дослідження є утиліта для СУБД SQLite.

РОЗДІЛ 2

ДОСЛІДЖЕННЯ СИСТЕМИ УПРАВЛІННЯ БАЗАМИ ДАНИХ SQLITE

2.1. Вибір і обґрунтування методів і засобів дослідження

В роботі застосовано системно-цільовий аналіз, структурно-функціональний аналіз, методи і технології проєктування баз даних [8-11, 15]. Поєднання цих методів формує протиріччя магістерського дослідження: структурно-функціональний аналіз системи розглядатиме її як стабільну, у той час коли системно-цільовий визначатиме багатоаспектність і взаємозв'язок процесів, які відбуваються. В основу головного принципу проєктування утиліти для СУБД SQLite покладено формування балансу між структурно-функціональним і системно-цільовим аналізом. Розглянемо детально обидва методи.

Структурно-функціональний аналіз (СФА), як правило, застосовується для дослідження соціальних явищ і процесів, де система розглядається як структурно розчленована цілісність: кожний елемент структури має певне функціональне призначення. Метод ґрунтується на двох підходах: перший – структурний: науковий пошук починається з аналізу структур, метою дослідження є виявлення функцій, що виконуються структурами. Другий підхід – функціональний: до системи висувається низка функціональних вимог, на підставі яких виявляються різні структури, що виконують виявлені функції. Дослідник оперує наступними поняттями: функція (наслідок діяльності, що сприяє адаптації системи); дисфункція (несприятливий наслідок); явна функція (об'єктивний та спрямований на пристосування або адаптацію системи наслідок); прихована функція (латентна функція, неусвідомлений наслідок, спрямований на відтворення системи).

Структурно-функціональний аналіз для адміністрування СУБД розглядає систему, для якій притаманні:

- структури: архітектура, компоненти, дані, користувачі, права;
- функції збереження, пошуку, безпеки, резервування.

Саме це допомагатиме визначитись з оптимальним вибором, впровадженням, конфігурацією та підтримкою СУБД. Адміністратор отримує ефективне управління, цілісність даних, безпеку та продуктивність. Окрему увагу приділяють компонентам даних: схемам, таблицям, індексам, представленням, тригерам та збереженим процедурам, оскільки саме вони формують логічну та фізичну організацію бази даних. Завершальним елементом структурного аналізу є оцінка використання системних ресурсів, зокрема взаємодії СУБД із сервером, оперативною пам'яттю, дисковою підсистемою та мережею.

Системно-цільовий аналіз (СЦА) розглядає дослідний об'єкт як сукупність взаємозв'язаних елементів за умовами їх функціональності. Реалізація поставлених задач досягається шляхом побудови моделі системи для аналізу і прогнозування її поведінки. Це розвиває можливості аналізувати великі обсяги даних, враховуючи зовнішнє середовище і внутрішню організацію.

Головними аспектами побудови дослідження є розробка довгострокових планів розвитку, програмно-цільове планування, аналіз проблем і ризиків, оцінка результативності, підвищення ефективності. Також розглядається система користувачів і ролей із визначенням рівнів доступу для забезпечення безпеки.

2.2. Обґрунтований вибір мови програмування

Проектування утиліти для SQLite передбачає комплексний підхід, що охоплює вибір мови програмування, застосування відповідних бібліотек,

визначення необхідного функціоналу та розробку інтерфейсу користувача з урахуванням специфіки цієї системи управління базами даних. Для реалізації утиліти можуть використовуватися мови програмування C, C++[17], Python [18] або C# [19], які забезпечують прямий доступ до механізмів SQLite через стандартні бібліотеки, зокрема `sqlite3` для мов C та Python. Визначення функціональних можливостей орієнтоване на підтримку базових операцій створення, читання, оновлення та видалення даних, засобів адміністрування, виконання SQL запитів і реалізації резервного копіювання. Вибір інтерфейсу може здійснюватися між командним рядком, графічним інтерфейсом або веб орієнтованим рішенням залежно від цільової аудиторії та умов експлуатації. При цьому враховується безсерверна та файлова природа SQLite, що зумовлює акцент на простоті розгортання і зручності використання в невеликих програмних застосунках.

Важливим етапом є вибір інструментарію, який визначає технологічну основу утиліти. Python у поєднанні з модулем `sqlite3` забезпечує швидку розробку та простоту підтримки, тоді як використання C або C++ із накативною бібліотекою SQLite дає змогу створювати високопродуктивні рішення. Для середовища .NET доцільним є застосування C# з технологіями ADO.NET або Entity Framework Core. Тип інтерфейсу також відіграє істотну роль, оскільки командний рядок зручний для автоматизації та скриптів, а графічний інтерфейс полегшує роботу з базою даних для кінцевих користувачів, що підтверджується практикою використання таких засобів, як DB Browser for SQLite.

Архітектурні рішення утиліти ґрунтуються на уявленні про SQLite як вбудовану систему, яка не потребує окремого серверного компонента. Утиліта фактично виконує роль клієнта, що працює безпосередньо з файлом бази даних і надсилає до нього SQL запити. Доцільним є застосування модульного підходу, за якого логіка доступу до даних відокремлюється від компонентів інтерфейсу, що підвищує гнучкість системи та спрощує її супровід і розширення.

Функціональні вимоги до утиліти включають реалізацію повного набору CRUD операцій, підтримку виконання довільних SQL запитів і управління транзакціями, а також засоби резервного копіювання та відновлення бази даних. Додатково передбачається можливість перегляду структури бази даних, зокрема схем, таблиць та індексів, що є важливим для задач адміністрування та аналізу.

Особливості SQLite істотно впливають на процес проєктування утиліти. Безсерверна модель усуває потребу в налаштуванні окремого серверного середовища, оскільки база даних зберігається у вигляді одного файлу. Підтримка атомарних транзакцій забезпечує цілісність даних навіть у разі збоїв. Водночас необхідно враховувати обмеження SQLite порівняно з повноцінними серверними СУБД, зокрема спрощені механізми управління правами доступу, що визначає сферу доцільного використання розроблюваної утиліти.

2.3. Аналіз процесів взаємодії користувача з базою даних

Застосовані методи змушені доповнювати один одного; на етапі проєктування бази даних структурний аналіз дає змогу оптимізувати модель даних і логіку зберігання інформації. Функціональний аналіз слугує основою для налаштування параметрів СУБД, зокрема механізмів кешування та блокування. У процесі експлуатації результати аналізу застосовуються для моніторингу стану системи, діагностики проблем і підтримки належного рівня безпеки шляхом контролю ролей, прав доступу та журналів аудиту.

2.3.1. Вхідні і вихідні дані

Важливим етапом проєктування утиліти є визначення вхідних і вихідних даних, для чого в таблицях 2.1 і 2.2 відповідно сформульовано функціональні (табл. 2.1) і нефункціональні (табл. 2.2) вимоги.

Функціональні вимоги

[складено автором]

Формулювання вимоги	Опис / зміст вимоги	Критерій виконання
Керування об'єктами БД	Забезпечення створення, редагування та видалення таблиць, індексів, подань і тригерів	Успішне виконання операцій без порушення цілісності БД
Виконання SQL-запитів	Надання інтерфейсу для введення, виконання та аналізу SQL-запитів	Коректний результат виконання або повідомлення про помилку
Візуалізація структури БД	Відображення структури таблиць і зв'язків між ними	Наявність графічного або структурованого подання схеми
Резервне копіювання та відновлення	Автоматизоване створення та відновлення копій БД	Успішне відновлення БД з резервної копії
Перевірка цілісності	Аналіз логічної та структурної цілісності даних і схеми	Виявлення та повідомлення про помилки
Аналіз продуктивності	Аналіз планів виконання запитів та індексації	Формування рекомендацій щодо оптимізації

Вхідні дані ґрунтуються на ресурсах, інформації, нормативно-правових документах. Вихідні дані орієнтуються на управлінські рішення, послуги, продукти.

Нефункціональні вимоги

[складено автором]

Формулювання вимоги	Опис / зміст вимоги	Критерій виконання
Надійність	Стійкість до збоїв та помилок блокування	Відсутність втрати даних

Формулювання вимоги	Опис / зміст вимоги	Критерій виконання
Продуктивність	Швидке виконання типових операцій	Час відповіді у допустимих межах
Кросплатформ-ність	Підтримка основних ОС	Коректна робота в різних середовищах
Масштабованість	Модульна та розширювана архітектура	Можливість додавання нових модулів
Зручність використання	Інтуїтивно зрозумілий інтерфейс	Мінімізація помилок користувача

2.3.2. Побудова моделі даних і логічної структури бази даних SQLite на основі функціональної і структурної моделі

Системно-цільовий аналіз СУБД зосереджується на дослідженні основних операцій роботи з даними, а саме створення, читання, оновлення та видалення інформації. Значну роль відіграє управління транзакціями, що включає забезпечення ACID властивостей, механізми блокування та підтримку паралельного доступу. Аналізуються засоби безпеки, зокрема автентифікація користувачів, авторизація доступу та аудит дій. Окремим напрямом є резервне копіювання та відновлення даних, що передбачає розроблення планів аварійного відновлення. Також розглядаються механізми оптимізації продуктивності, такі як планування виконання запитів, індексування та кешування, а завершальним етапом виступає моніторинг і аудит, які забезпечують відстеження активності та журналювання подій у системі.

Візуалізація функціональних вимог системи з точки зору користувача на рис. 2.1 наведено у вигляді UML-діаграми використання.

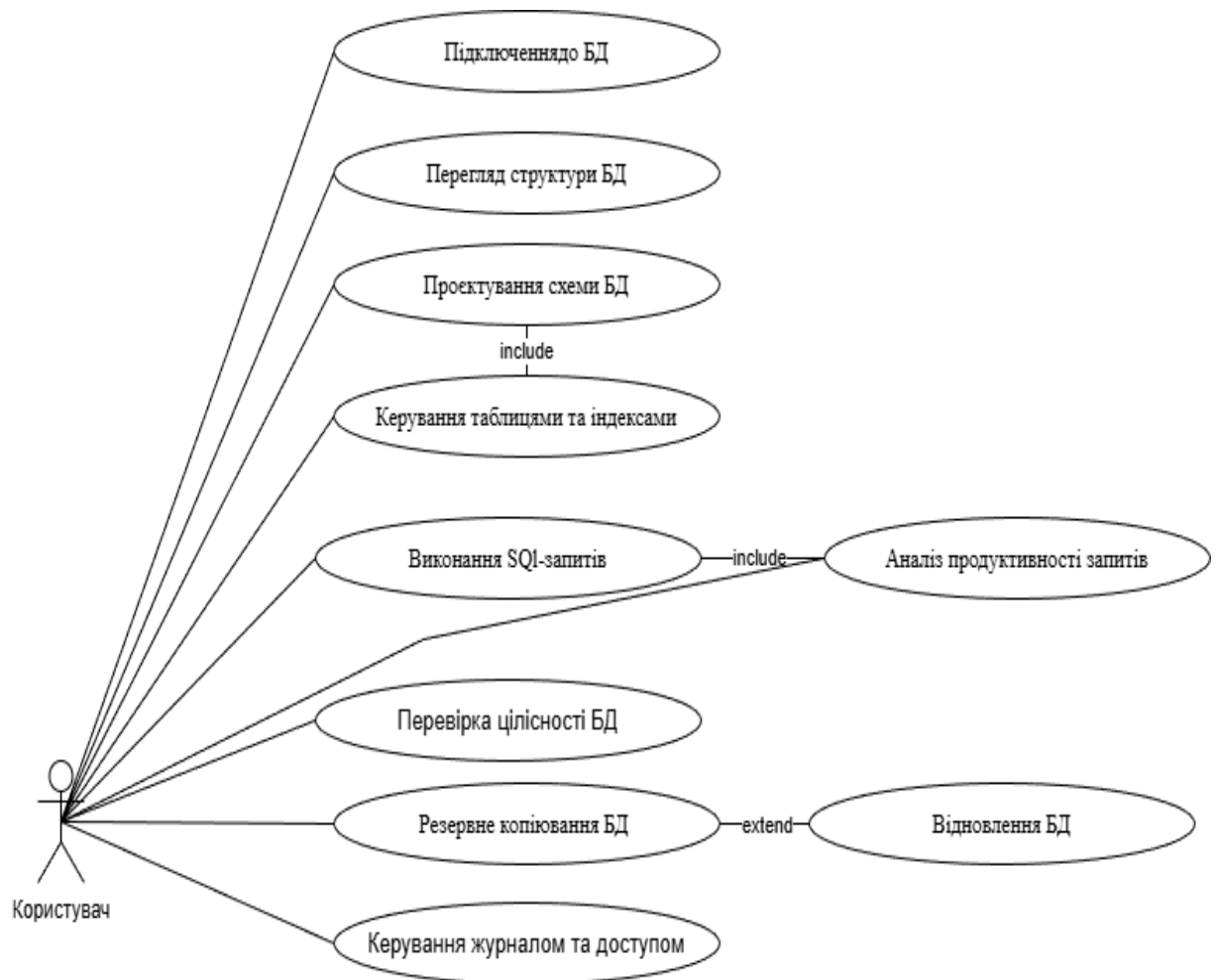


Рис.2.1. UML-діаграма використання
[складено автором]

Діаграма фокусується на поведінці системи і являє собою підґрунтя для розробки технічного завдання, тест-кейсів і проєктної документації.

Концептуальну модель даних наведено на рис. 2.2.

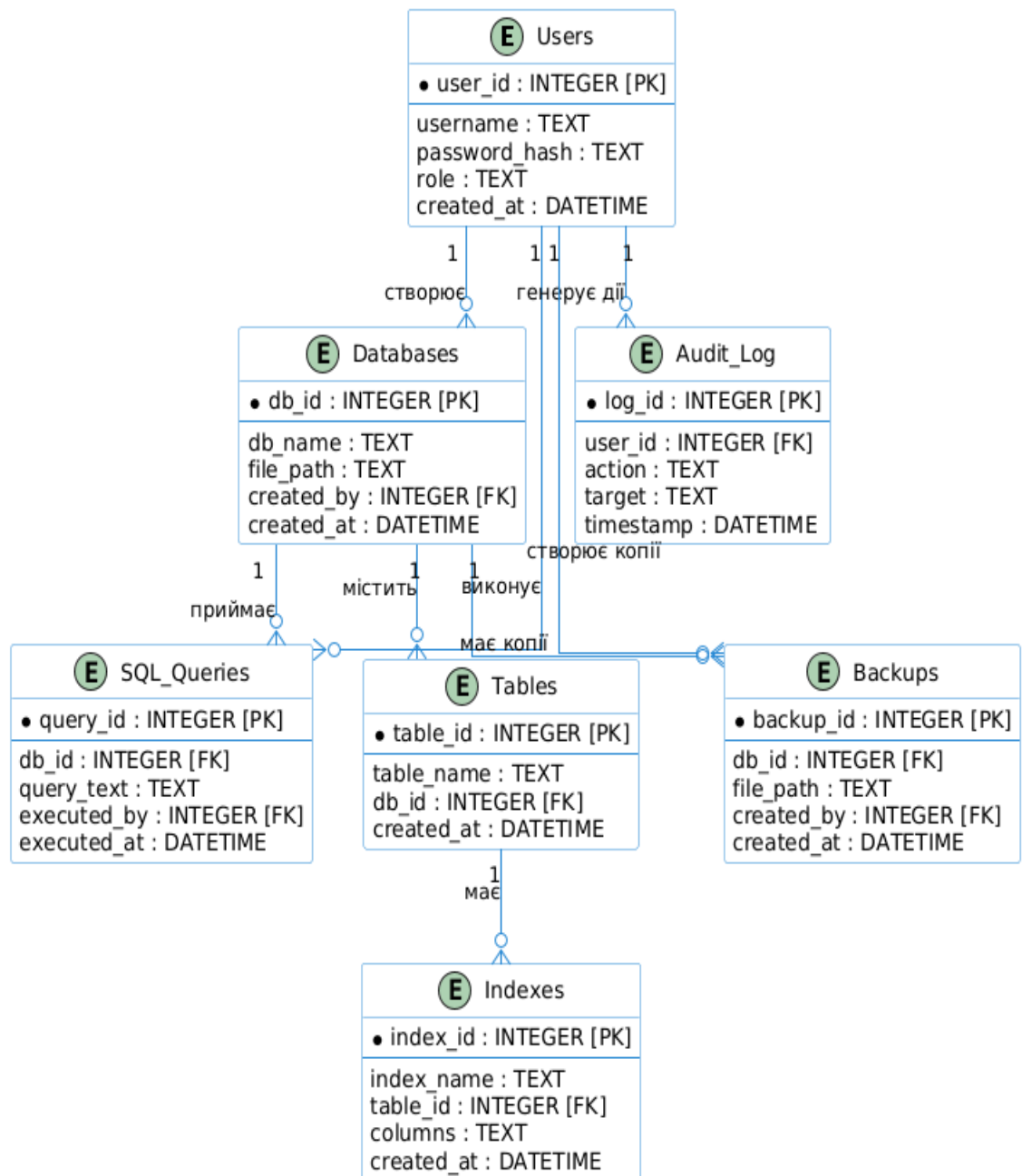


Рис. 2.2. Концептуальна модель даних
[складено автором]

2.4. Висновки до розділу 2

1. В основу головного принципу проєктування утиліти для СУБД SQLite покладено формування балансу між структурно-функціональним і системно-цільовим аналізом. Розглянуто пріоритети, переваги і недоліки цих методів.

2. Для реалізації утиліти можуть використовуватися мови програмування C, C++, Python або C#, які забезпечують прямий доступ до механізмів SQLite через стандартні бібліотеки.

3. Важливим етапом проєктування утиліти є визначення вхідних і вихідних даних, для чого сформульовано функціональні і нефункціональні вимоги.

4. Побудовано UML-діаграму використання; запропоновано концептуальну модель даних.

РОЗДІЛ 3

ПРОЄКТУВАННЯ УТИЛІТИ НА ОСНОВІ РЕЗУЛЬТАТІВ СИСТЕМНОГО АНАЛІЗУ ЗАСОБІВ АДМІНІСТРУВАННЯ

3.1. Розробка архітектури програмної утиліти

Архітектура програмної утиліти адміністрування та проєктування СУБД SQLite побудована за багаторівневим (layered) архітектурним підходом, що забезпечує чіткий розподіл відповідальності між компонентами, підвищує масштабованість, супроводжуваність і надійність програмного забезпечення.

Рівень представлення реалізований у вигляді графічного інтерфейсу користувача (GUI), який забезпечує взаємодію користувача з утилітою.

Основними функціями цього рівня є:

- введення команд та параметрів адміністрування;
- візуалізація структури бази даних;
- відображення результатів SQL-запитів і діагностичних повідомлень.

Рівень прикладної логіки (Application Layer) є ключовим функціональним ядром утиліти та складається з таких модулів:

Модуль керування користувачами – забезпечує автентифікацію, авторизацію та розмежування доступу.

Модуль адміністрування БД – реалізує операції створення, модифікації та видалення об'єктів бази даних.

Модуль проєктування схеми БД – відповідає за логічне моделювання, аналіз та візуалізацію структури БД.

Модуль виконання SQL-запитів – забезпечує виконання запитів та обробку результатів.

Модуль аналізу продуктивності – здійснює аналіз планів виконання запитів і формує рекомендації з оптимізації.

Модуль резервного копіювання та відновлення – реалізує механізми збереження та відновлення баз даних.

Модуль журналювання та безпеки – забезпечує аудит дій користувачів та захист адміністративних операцій.

Модульна структура цього рівня забезпечує можливість розширення функціоналу без порушення цілісності системи.

Рівень доступу до даних (Data Access Layer) реалізує взаємодію з СУБД SQLite через SQLite API. Він інкапсулює всі операції доступу до файлу бази даних, забезпечуючи:

- уніфікований доступ до даних;
- незалежність прикладної логіки від конкретної реалізації СУБД;
- централізовану обробку помилок доступу та блокувань.

До зовнішніх ресурсів належать:

- файл бази даних SQLite, що зберігає всі дані;
- файлова система резервних копій, яка використовується для збереження архівів та журналів.

Запропонована архітектура програмної утиліти (рис. 3.1) відповідає принципам модульності, слабкої зв'язаності та високої когерентності, що дозволяє ефективно реалізувати засоби адміністрування та проектування автономних баз даних SQLite, а також забезпечує можливість подальшого розвитку та інтеграції утиліти в прикладні інформаційні системи.

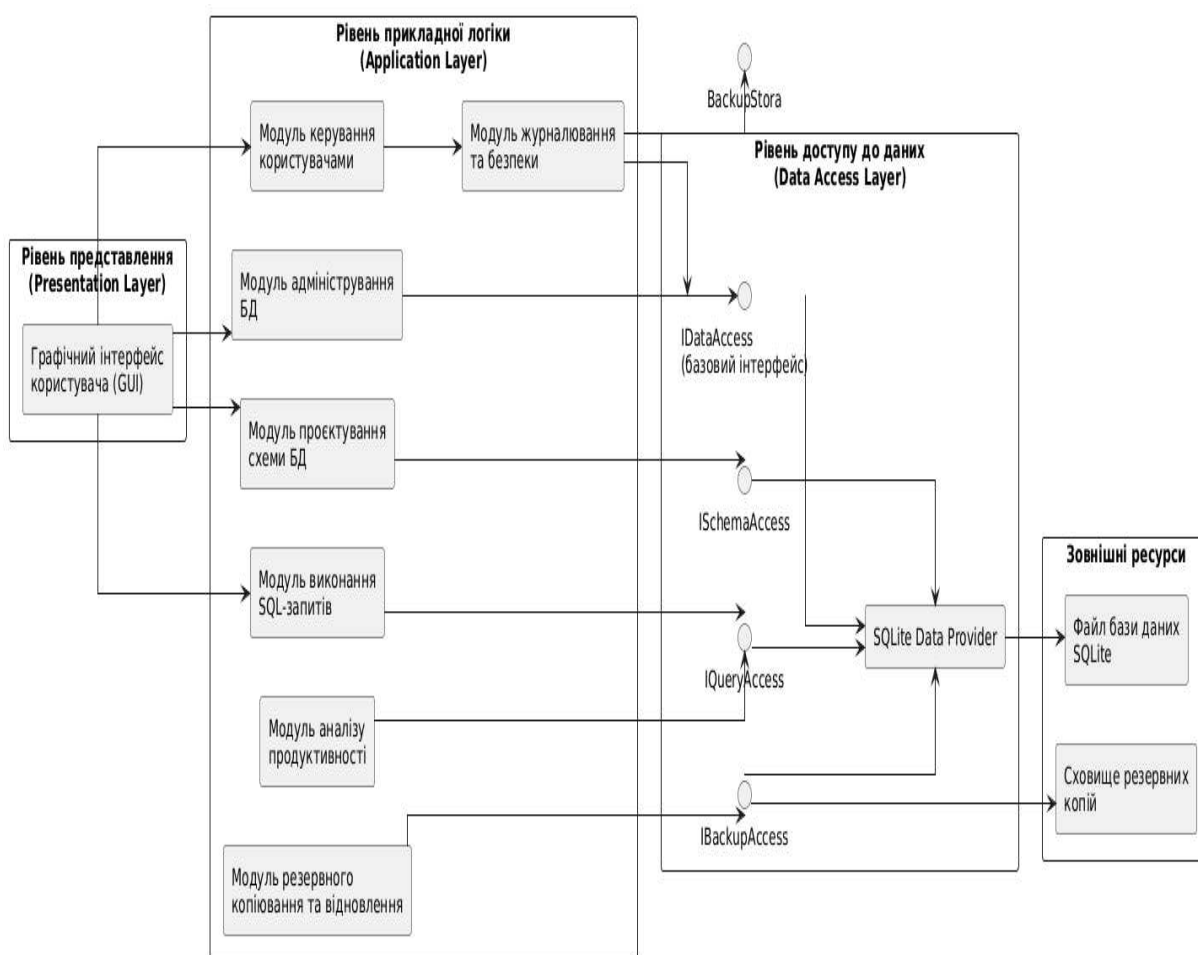


Рис. 3.1. Розробка програмної утиліти
[складено автором]

3.2 Використання бібліотек SQLite (sqlite3, contextlib, os, SQLAlchemy тощо)

Під час розроблення програмної утиліти адміністрування та проєктування баз даних SQLite було використано низку стандартних та сторонніх бібліотек мови програмування Python, які забезпечують ефективну взаємодію з базою даних, керування ресурсами файлової системи та абстракцію рівня доступу до даних.

Бібліотека sqlite3 є стандартним модулем Python, що надає низькорівневий інтерфейс для роботи з СУБД SQLite.

Вона використовується для безпосереднього виконання SQL-запитів, керування транзакціями та обробки результатів.

Основними можливостями використання є:

- встановлення з'єднання з файлом бази даних SQLite;
- виконання DDL- та DML-операцій (CREATE, INSERT, UPDATE, DELETE);
- керування транзакціями (commit, rollback);
- обробка винятків, пов'язаних із блокуванням та пошкодженням бази даних.

Застосування бібліотеки sqlite3 доцільне для реалізації критично важливих адміністративних операцій, де необхідний повний контроль над SQL-виконанням та транзакціями.

Бібліотека contextlib використовується для реалізації контекстних менеджерів, що забезпечують коректне керування ресурсами під час роботи з базою даних.

Основними перевагами використання є:

- автоматичне закриття з'єднань з базою даних;
- гарантія виконання операцій commit або rollback;
- підвищення надійності та читабельності коду.

Застосування `contextlib` дозволяє мінімізувати ризик витоків ресурсів і забезпечити коректне завершення транзакцій навіть у разі виникнення помилок.

Бібліотека `os` використовується для взаємодії з файловою системою операційної системи.

Основними напрямками використання є:

- перевірка наявності файлу бази даних;
- створення та керування каталогами для резервних копій;
- отримання інформації про шлях до файлу БД;
- керування правами доступу до файлів.

Використання `os` є необхідним з огляду на файлову природу SQLite, де база даних зберігається у вигляді одного файлу.

Бібліотека `SQLAlchemy` є високорівневою ORM-бібліотекою, що забезпечує абстракцію доступу до бази даних та спрощує роботу з об'єктною моделлю даних.

У межах утиліти `SQLAlchemy` використовується для:

- опису структури бази даних у вигляді Python-класів;
- реалізації концептуальної та логічної моделей даних;
- уніфікації доступу до SQLite незалежно від конкретної реалізації СУБД;
- зменшення кількості ручного SQL-коду.

Поєднання `SQLAlchemy` з `sqlite3` дозволяє реалізувати гібридний підхід: ORM використовується для стандартних операцій, тоді як `sqlite3` – для спеціалізованих адміністративних задач.

За потреби можуть використовуватися додаткові стандартні бібліотеки Python, зокрема `logging` для журналювання дій користувачів та системних подій; `shutil` для копіювання та відновлення файлів бази даних; `datetime` для фіксації часу виконання адміністративних операцій.

Використання бібліотек `sqlite3`, `contextlib`, `os` та `SQLAlchemy` забезпечує комплексний підхід до розроблення утиліти адміністрування

SQLite, поєднуючи низькорівневий контроль доступу до даних із високорівневою абстракцією та надійним керуванням ресурсами. Такий підхід підвищує стабільність, безпеку та масштабованість програмного рішення.

3.3. Розробка інтерфейсу

Інтерфейс користувача програмної утиліти адміністрування СУБД SQLite розроблено з урахуванням принципів системності, зручності використання та функціональної достатності. Основним завданням інтерфейсу є забезпечення ефективної взаємодії користувача з автономною базою даних шляхом візуалізації структури даних, надання інструментів керування та засобів виконання SQL-запитів.

Інтерфейс утиліти має модульну структуру та складається з верхньої панелі навігації, бічного меню, центральної робочої області та панелі системного статусу. Загальну архітектуру інтерфейсу наведено в таблиці 3.1.

Таблиця 3.1

Загальна структура інтерфейсу утиліти

[складено автором]

№	Компонент інтерфейсу	Тип елемента	Призначення
1	Верхня панель навігації	Панель керування	Глобальний пошук, доступ до профілю користувача та керування сесією
2	Бічне меню	Навігаційний список	Перехід між функціональними модулями утиліти
3	Центральна робоча область	Контейнер вмісту	Відображення активного функціонального модуля

№	Компонент інтерфейсу	Тип елемента	Призначення
4	Панель статусу	Інформаційний блок	Відображення системного стану та результатів виконання дій

Бічне меню забезпечує швидкий доступ до основних функцій адміністрування бази даних. Структуру пунктів навігації подано в табл. 3.2.

Таблиця 3.2

Елементи бічного меню навігації

[складено автором]

№	Назва пункту	Функціональне призначення
1	Dashboard	Огляд поточного стану системи
2	Databases	Перегляд і вибір баз даних SQLite
3	Tables	Керування таблицями бази даних
4	Indexes	Керування індексами
5	SQL Editor	Виконання SQL-запитів
6	Backups	Створення резервних копій
7	Audit Log	Перегляд журналу дій
8	Users Mgmt	Керування користувачами

Модуль перегляду таблиць призначений для аналізу структури бази даних та виконання базових адміністративних операцій. Основні елементи цього інтерфейсу наведено в таблиці 3.3.

Таблиця 3.3

Інтерфейс перегляду таблиць бази даних

[складено автором]

Елемент	Тип	Опис
Name	Текстове поле	Назва таблиці

Елемент	Тип	Опис
Columns	Числове поле	Кількість стовпців
Rows	Числове поле	Кількість записів
Created At	Дата	Дата створення
Actions	Кнопки	Редагування та видалення таблиці

Для роботи з довільними SQL-запитами в утиліті реалізовано SQL-редактор із підтримкою вкладок, швидкого виконання та відображення результатів. Основні компоненти SQL-редактора наведено в табл. 3.4.

Таблиця 3.4

Елементи SQL-редактора

[складено автором]

№	Компонент	Тип елемента	Призначення
1	Поле введення SQL	Текстова область	Введення SQL-запитів
2	Execute	Кнопка	Запуск виконання запиту
3	Вкладки	Панель вкладок	Паралельна робота з запитам
4	Результати	Таблиця	Відображення результатів SELECT
5	Повідомлення	Інформаційний блок	Виведення статусу виконання

Для аналізу внутрішньої структури таблиць реалізовано інтерфейс інспекції, що дозволяє переглядати стовпці, обмеження цілісності та індекси. Елементи інспекції наведено в табл. 3.5.

Таблиця 3.5

Елементи SQL-редактора

[складено автором]

Компонент	Опис
Назва таблиці	Поточна таблиця
Columns	Перелік стовпців і типів
Constraints	PK, FK, NOT NULL
Indexes	Пов'язані індекси
Backup Action	Створення резервної копії

Для забезпечення контролю над станом системи в інтерфейсі реалізовано блок системних повідомлень і журналу дій (табл.3.6)

Таблиця 3.6

Елементи SQL-редактора

[складено автором]

Тип повідомлення	Опис
System Status	Стан з'єднання
API Latency	Час виконання запитів
SQLite Version	Версія СУБД
Action Log	Журнал дій користувача

Запропонований інтерфейс програмної утиліти адміністрування SQLite (рисунки 3.2-3.3) забезпечує комплексну підтримку основних операцій керування автономними базами даних, поєднуючи зручність використання з функціональною повнотою. Інтерфейсна модель є масштабованою та може бути адаптована для різних сценаріїв використання в автономних та вбудованих інформаційних системах.

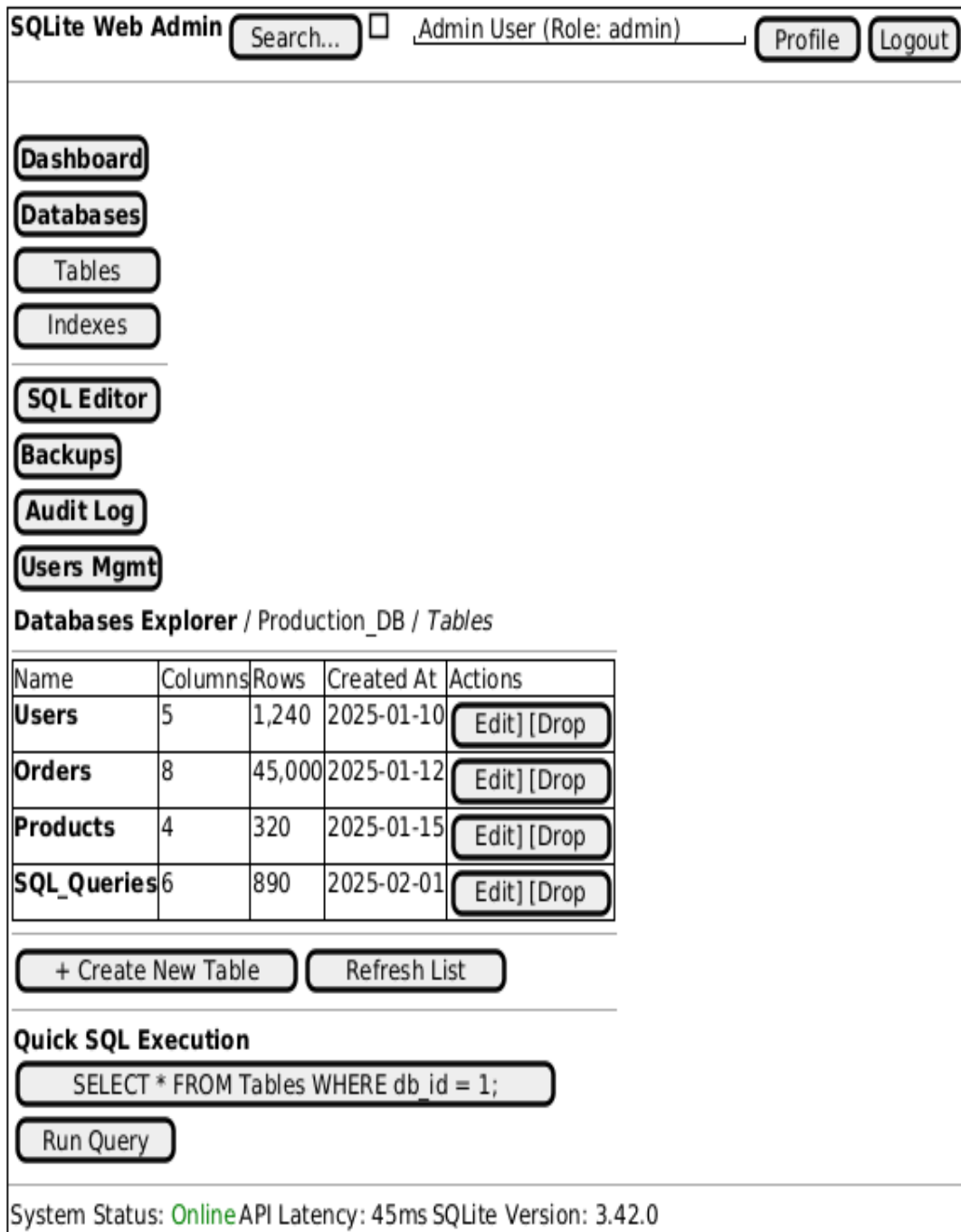


Рис. 3.2. Проект інтерфейсу 1
[складено автором]

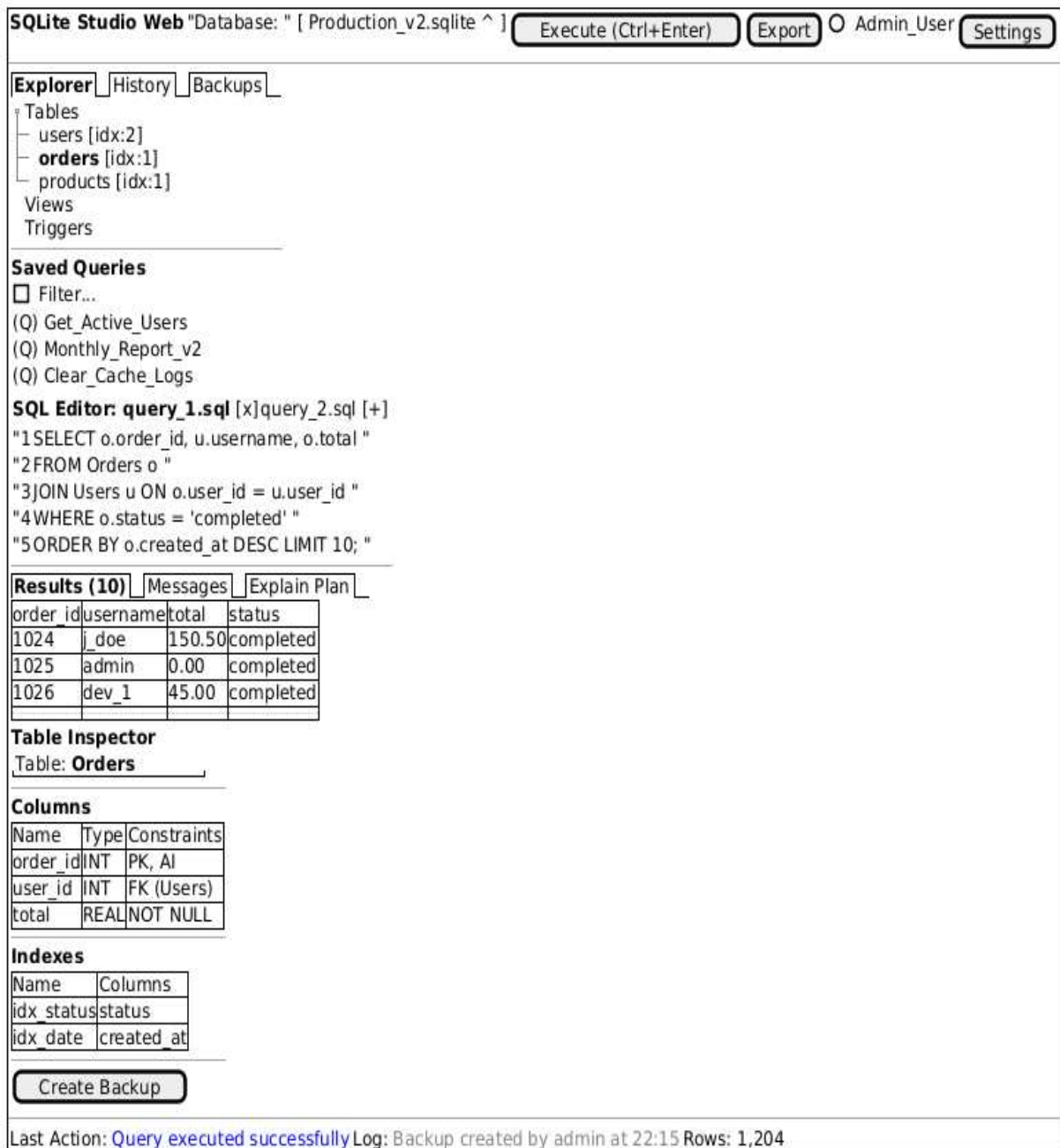


Рис. 3.3. Прокст інтерфейсу 2

[складено автором]

3.4. Апробація і реалізація утиліти

Апробація програмної утиліти адміністрування СУБД SQLite здійснювалася в процесі її практичної реалізації та експериментального використання в середовищі з автономними базами даних. Основною метою апробації було підтвердження коректності реалізованих функцій,

відповідності архітектурних рішень поставленим вимогам та перевірка зручності інтерфейсу користувача під час виконання типових адміністративних операцій.

Реалізація утиліти передбачає роботу з декількома файлами баз даних SQLite, що відображаються у вигляді ієрархічної навігаційної структури. Кожна база даних представлена у вигляді окремого вузла, що містить логічно згруповані об'єкти, зокрема таблиці, індекси та резервні копії. Такий підхід забезпечує наочне представлення структури бази даних і дозволяє користувачу швидко перемикатися між об'єктами без необхідності ручного введення SQL-команд.

У процесі апробації особливу увагу було приділено реалізації інтерфейсу редагування структури таблиць. Продемонстровано можливість перегляду списку стовпців, їх типів даних, наявності первинного ключа та обмежень цілісності. Графічний інтерфейс дозволяє вносити зміни до структури таблиці, зокрема додавати нові атрибути або видаляти існуючі, при цьому відповідні SQL-операції формуються автоматично. Це підтверджує ефективність реалізованого механізму трансляції дій користувача в коректні SQL-запити.

Важливою складовою реалізації утиліти є підтримка керування індексами. Інтерфейс відображає інформацію про асоційовані з таблицею індекси, включаючи їх назви, склад полів та властивість унікальності. Апробація показала, що користувач може здійснювати операції зі створення та видалення індексів без прямого доступу до внутрішніх структур СУБД, що значно спрощує оптимізацію запитів у автономних базах даних.

Для виконання довільних SQL-команд у складі утиліти реалізовано SQL-редактор, який дозволяє безпосередньо виконувати запити до бази даних і переглядати результати виконання. Результати апробації підтвердили стабільність роботи SQL-редактора та коректність відображення статусу виконання запитів, включаючи час їх обробки. Виведення інформації про

виконані команди у консольному журналі забезпечує прозорість адміністративних дій і підвищує довіру до системи.

Окрему увагу під час апробації було приділено механізму інформування користувача про стан системи. Інтерфейс відображає службові повідомлення щодо успішного виконання операцій, що дозволяє оперативно контролювати процес адміністрування. Такий підхід є особливо важливим у контексті автономних систем баз даних, де помилки можуть призвести до втрати цілісності даних.

Результати проведеної апробації (рисунок 3.4) підтверджують, що розроблена програмна утиліта забезпечує повний цикл базових операцій адміністрування СУБД SQLite, включаючи керування структурою таблиць, індексами та виконання SQL-запитів. Реалізовані інтерфейсні та архітектурні рішення демонструють практичну придатність утиліти для використання як у навчальних цілях, так і в реальних прикладних проєктах, де застосовуються автономні бази даних.

SQLite Cloud Admin ☐ Notifications **Admin (Dev role)** [v]

[+] **Project_DB.sqlite**
 [+] Tables
 [#] Users
 [#] Orders
 [#] **Products**
 [+] Indexes
 (l) idx_prod_name
 [+] Backups (3)

[+] New_Test_DB.sqlite

Settings
Audit Logs

Table Structure | Data Preview | SQL Editor | Statistics |

Table Name:

Column Name	Type	PK	Not Null	Actions
product_id	INTEGER	☒	☒	Del
name	TEXT	☐	☒	Del
price	REAL	☐	☐	Del
category_id	INTEGER	☐	☒	Del

Associated Indexes

Index Name	Columns	Unique	Actions
idx_prod_name	name	☒	Drop

Console Output
 ALTER TABLE Products ADD COLUMN stock INTEGER;
 Query executed successfully (0.05s)

Рис. 3.4. Приклад реалізації
 [складено автором]

Фрагменти лістингу наведені у Додатку.

3.5 Висновки до розділу 3

1. Розроблено архітектуру програмної утиліти адміністрування та проєктування СУБД SQLite, яка забезпечує чіткий розподіл відповідальності між компонентами, підвищує масштабованість, супроводжуваність і надійність програмного забезпечення.

2. Як інструментальні засоби для розробки програмної утиліти застосовано низку стандартних та сторонніх бібліотек (sqlite3, os, contextlib, SQLAlchemy) мови програмування Python, які забезпечують ефективну взаємодію з базою даних, керування ресурсами файлової системи та абстракцію рівня доступу до даних.

3. Побудовано користувацький інтерфейс та апробовано розроблений програмний продукт.

РОЗДІЛ 4

ТЕХНІКО-ЕКОНОМІЧНИЙ АНАЛІЗ РОЗРОБКИ

4.1 Загальні положення

Розвиток новітніх технологій та зростаюче використання обчислювальних ресурсів зумовили широке впровадження комп'ютерної техніки для вирішення завдань, пов'язаних із обробкою, контролем та аналізом інформації у виробничих процесах.

4.2 Розрахунок витрат на створення і впровадження утиліти

Витрати на розробку включають витрати на оплату праці розробника, амортизацію персонального комп'ютера, на якому виконується розробка, витрати на його експлуатацію, засоби розробки, а також витрати на матеріали та комплектуючі.

Виходячи з того, що основна заробітна плата розробників сайту складає 15000 грн./міс., вартість сучасного ПК складає 42000 грн. і вартість кіловат-години електроенергії складає 4,32 грн. з ПДВ, розрахуємо вартість розробки системи:

Розрахунок заробітної плати зробимо по формулі 4.1:

$$З_{зп} = ЗП_{роз} * T_{роз}, \quad (4.1)$$

де $ЗП_{роз}$ – зарплата розробника, за місяць;

$T_{роз}$ – тривалість розробки (дослідження, створення, налагодження і впровадження).

Для розробки утиліти необхідно 6 місяців (за експертною оцінкою часу на розробку аналогічних утиліт). З цього випливає, що загальна сума витрат на заробітну плату складе:

$$З_{зп} = ЗП_{роз} * T_{роз} = 15000 * 6 = 90000 \text{ (грн);}$$

Додаткова заробітна плата складає 20% від основної заробітної плати:

$$З_{\text{дод}} = З_{\text{зп}} * 0,2 = 18000 \text{ (грн.)}.$$

Відрахування на соціальні заходи складають 38% від основної і додаткової заробітної плати, розраховуються по формулі 4.2:

$$В_{\text{сз}} = (З_{\text{зп}} + З_{\text{дод}}) * 38\% \quad (4.2)$$

$$В_{\text{сз}} = (90000 + 18000) * 0,38 = 41040 \text{ (грн)}$$

Витрати на амортизацію персонального комп'ютера (ПК), на якому виконується розробка, розраховується по формулі 4.3:

$$З_{\text{амор}} = Ц_{\text{варт}} * A * T_{\text{роз}} \quad (4.3)$$

де $Ц_{\text{варт}} = 420000$ грн. – балансова вартість ПК;

$A = 40\%$ – амортизація за рік;

$T_{\text{роз}} = 0,5$ року – час, необхідний для розробки.

$$З_{\text{амор}} = 420000 * 0,4 * 0,5 = 84000 \text{ (грн);}$$

Витрати на експлуатацію ПК, на якому виконується розробка, полягають в оплаті споживаної нею електричної енергії і розраховуються по формулі 4.4:

$$З_{\text{елек}} = P_{\text{еом}} * T_{\text{розр}} * N_{\text{рд}} * N_{\text{рч}} * E_{\text{ел}} \quad (4.4)$$

де $P_{\text{еом}} = 0,2$ квт./год. – норма споживання електроенергії ПК;

$T_{\text{розр}} = 6$ міс. – тривалість розробки;

$N_{\text{рд}} = 21$ день – число робочих днів у місяці;

$N_{\text{рч}} = 8$ годин – число годин у робочому дні;

$E_{\text{ел}} = 4,32$ грн. – вартість 1 квт. електроенергії.

$$З_{\text{елек}} = 0,2 * 6 * 21 * 8 * 4,32 = 868,22 \text{ (грн)}$$

Загальновиробничі і адміністративні затрати складають 50% від основної та додаткової заробітної плати, розраховуються за формулою 4.5:

$$З_{\text{зв}} = (З_{\text{зп}} + З_{\text{дод}}) * 50\% \quad (4.5)$$

$$З_{\text{зв}} = (90000 + 18000) * 0,5 = 54000 \text{ грн}$$

Під час розробки будуть інші поточні затрати, які включають у себе канцелярські товари, флеш накопичувачі та інші витратні матеріали, які складають 10% від основної заробітної плати:

$$B_{\text{пот}} = 90000 * 0,1 = 9000 \text{ грн}$$

Для розробки необхідні наступні засоби розробки (таблиця 4.1).

Таблиця 4.1

Програмні засоби розробки

№	Найменування	Вартість, грн.
1	MySQL Server +J Connector	0
2	Eclipse	0
Разом		0

4.2.7. Загальний кошторис витрат на створення наведено в таблиці 4.2.

Таблиця 4.2

Загальний кошторис витрат на створення модуля

Найменування витрат	Сума, грн.
Заробітна плата основна	90000,00
Заробітна плата додаткова (20% від п.1)	18000,00
Витрати на соціальні заходи	41040,00
Загальновиробничі витрати	54000,00
Витрати на амортизацію ПК	8400,00
Витрати на експлуатацію ПК	868,22
Інші витрати	9000,00
Витрати на програмні засоби розробки	0,00
Разом на створення модуля	221308,22

4.3 Витрати на впровадження з урахуванням вартості навчання кадрів і витрати на монтаж і налаштування.

4.3.1. Впровадження включає в себе встановлення серверу з високошвидкісним доступом до глобальної мережі Інтернет і джерелом безперервного живлення. Витрати на обладнання наведені у таблиці 4.3.

Таблиця 4.3

Витрати на обладнання	
Обладнання	Сума, грн
Сервер HP ProLiant DL380p SFF Gen8 Refurbished 525	18318,00
Джерело безперебійного живлення APC Back-UPS 1100VA (BX1100CI-RS)	5630,00
Разом:	23948,00

Доступ до глобальної мережі Інтернет можна вважати вже існуючим і не потребує витрат.

4.3.2. Витрати на технічну підтримку і підготовку кадрів наведена в таблиці 4.5

Таблиця 4.4

Витрати на технічну підтримку і підготовку кадрів	
Найменування	Сума, грн.
Підготовка кадрів	3000,00
Технічна підтримка за рік користування (супровод)	1050,00
Разом:	4050,00

Додаткові капітальні вкладення з урахуванням витрат на проєктування, створення і функціонування – загальна сума витрат на

створення і впровадження складає: 221308,22 + 23948,00 + 4050,00 = 249 306,22 грн.

4.4 Економічна ефективність розробки і впровадження

Основним показником економічної ефективності функціонування системи є підвищення ефективності керування інформацією у вигляді зниження витрат на керування при одночасному збільшенні швидкості і якості одержання потрібного результату.

Обов'язковою умовою визначення економічної ефективності програми є оцінка усіх показників у часі, за цінами й іншими нормами, використовуваними для визначення показників, за змістом і колом елементів витрат.

Використання даної утиліти дозволяє вивільнити 20% робочого часу керівника проєкту а також 10% робочого часу програмістів. Таким чином ми можемо сказати, що для реалізації проєкту нам необхідно буде на 10% менше програмістів і на 20% менше керівника проєкту.

Визначаємо пряму економічну ефективність ґрунтуючись на тому, що бази даних обробляють програміст, спеціаліст по роботі з базами даних, адміністратор сайту. При розрахунку можна вважати, що при впровадженні модуля завантаженість вищезазначених спеціалістів знизиться в 2 рази.

Зарплата програміста складає

$$\Delta C_k = 10000 * 12 = 120000 (\text{грн.}).$$

Зарплата спеціаліст по роботі з базами даних та адміністратора сайту складає

$$\Delta C_\phi = 6000 * 12 * 2 = 144\,000 (\text{грн.})$$

Річний економічний ефект розраховується по формулі 4.6:

$$E_{\text{рік}} = \Delta C_\phi + \Delta C_k - C_{\text{супр}} - E_{\text{п}} * k \quad (4.6)$$

де E_n – нормативний коефіцієнт економічної ефективності капітальних вкладень у галузь – для обчислювальної техніки приймається = 0,6;

$C_{\text{супр}}$ – вартість супроводження;

k – додаткові капітальні вкладення з урахуванням витрат на проєктування, створення і функціонування.

$$E_{\text{рік}} = 120000 + 144000 - 1050,00 - 0,6 \\ * 249\,306,22 = 113\,366,27 \text{ (грн.)}$$

Строк окупності модуля розраховується по формулі 4.7:

$$I = \frac{k}{E_{\text{рік}}} = \frac{249\,306,22}{113\,366,27} \approx 2,20 \text{ (року)} \approx 2 \text{ роки } 2 \text{ місяці}$$

(4.7)

Тобто, модуль окупиться приблизно через 2 роки 2 місяці.

4.5 Висновки по розділу 4

1. У ході проведених розрахунків було здійснено аналіз ключових фінансових показників, пов'язаних з витратами на впровадження та функціонування утиліти. Порівняльна оцінка передбаченого бюджету із альтернативними варіантами реалізації засвідчила, що запропонований підхід є економічно обґрунтованим.

2. Обрана модель демонструє оптимальне співвідношення між витратами та функціональними перевагами, забезпечує гнучкість у масштабуванні та сприяє ефективному управлінню. Це дозволяє вважати запропоноване рішення доцільним у довгостроковій перспективі як з економічної, так і з організаційної точки зору.

РОЗДІЛ 5

ОХОРОНА ПРАЦІ

5.1 Вступ

Охорона праці є системою правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів, спрямованих на збереження життя, здоров'я та працездатності людини у процесі трудової діяльності. У реальних умовах повністю безпечні виробничі процеси практично не існують, тому основним завданням охорони праці є мінімізація ризиків травматизму та професійних захворювань при збереженні ефективності праці.

Правові засади охорони праці в Україні визначаються Законом України «Про охорону праці», який встановлює основні принципи державної політики у цій сфері, гарантує працівникам право на безпечні та нешкідливі умови праці, а також передбачає відповідальність роботодавця за порушення нормативних вимог [22].

Державний нагляд за дотриманням законодавства з охорони праці здійснюють Державна служба України з питань праці, органи пожежного, санітарного та радіаційного контролю, а також органи місцевого самоврядування в межах їхніх повноважень. Вищий нагляд за дотриманням законодавства покладено на Офіс Генерального прокурора України [22].

Важливу роль у забезпеченні безпечних умов праці відіграє стандартизація. Система стандартів безпеки праці встановлює єдині вимоги до організації робочих місць, технологічних процесів і засобів індивідуального захисту.

5.2 Основні небезпечні та шкідливі виробничі фактори

У процесі трудової діяльності на працівників можуть впливати небезпечні та шкідливі виробничі фактори, які класифікуються за характером впливу на фізичні, хімічні, біологічні, механічні та психофізіологічні.

До фізичних факторів належать шум, вібрація, несприятливі температурні умови та іонізуюче випромінювання. Їх тривалий вплив може призводити до зниження працездатності та розвитку професійних захворювань.

Хімічні фактори представлені шкідливими газами, парами, аерозолями та пилом, які можуть спричиняти інтоксикацію організму, ураження органів дихання та алергічні реакції.

Біологічні фактори пов'язані з дією мікроорганізмів та алергенів і становлять небезпеку виникнення інфекційних та алергічних захворювань.

Механічні фактори включають ризик травмування внаслідок дії рухомих частин обладнання та порушення правил експлуатації технічних засобів.

Психофізіологічні фактори зумовлені перевантаженням нервової системи, стресами та недостатнім відпочинком, що підвищує ризик нещасних випадків.

5.3 Оцінка ризиків для здоров'я працівників

Оцінка професійних ризиків є важливим елементом системи охорони праці та передбачає виявлення небезпечних факторів, аналіз можливих наслідків їх впливу та визначення ймовірності виникнення шкоди для здоров'я працівників.

Процес оцінки ризиків включає ідентифікацію джерел небезпеки, оцінку ймовірності їх прояву та визначення тяжкості можливих наслідків.

Ризик розглядається як поєднання ймовірності виникнення небезпечної події та рівня шкоди, яку вона може спричинити.

За результатами оцінки ризиків розробляються технічні, організаційні та санітарно-гігієнічні заходи, спрямовані на зменшення або усунення небезпек відповідно до вимог чинного законодавства [22].

5.4 Забезпечення сприятливих умов праці при роботі з персональним комп'ютером

Організація робочого місця при роботі з персональним комп'ютером повинна відповідати санітарно-гігієнічним та ергономічним вимогам з метою збереження здоров'я працівників і підвищення ефективності праці. Основні вимоги до умов праці з відеодисплейними терміналами регламентуються державними санітарними правилами та нормами [24].

Монітор повинен розміщуватися на відстані 40–45 см від очей користувача, на рівні або трохи нижче лінії зору. Частота оновлення зображення має бути не менше 72 Гц, а робоче місце слід розташовувати так, щоб уникати відблисків і надмірного зорового навантаження [24].

Крісло та робочий стіл повинні бути регульованими та забезпечувати правильне положення тіла. Клавіатура розміщується таким чином, щоб положення рук було природним і не спричиняло перенапруження м'язів і суглобів.

Вимоги безпеки під час експлуатації електронно-обчислювальних машин, а також обов'язковість перерв у роботі та проведення інструктажів визначаються відповідними правилами охорони праці [24].

Рациональна організація робочого місця та дотримання режиму праці й відпочинку сприяють профілактиці професійних захворювань і збереженню працездатності працівників.

5.5 Висновки по розділу 5.

1. Раціональна організація робочого місця при роботі з персональним комп'ютером є важливою умовою забезпечення безпечних, ергономічних і психофізіологічно комфортних умов праці в умовах цифровізації.

2. Запропоновані заходи забезпечують безпечні та комфортні умови роботи за персональним комп'ютером і сприяють підвищенню продуктивності та якості виконання функціональних обов'язків.

РОЗДІЛ 6

ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ

Охорона навколишнього природного середовища є системою державних, правових, економічних та організаційних заходів, спрямованих на раціональне використання природних ресурсів, збереження екосистем і забезпечення екологічної безпеки. Вона охоплює охорону атмосферного повітря, водних ресурсів, земель, флори і фауни, а також геологічного середовища.

Метою охорони довкілля є запобігання та мінімізація негативного впливу антропогенної діяльності, відновлення порушених природних комплексів і забезпечення сприятливих умов життєдіяльності людини. У загальному вигляді проблема охорони навколишнього середовища зводиться до раціонального природокористування та збереження екологічної рівноваги.

Актуальність екологічної проблематики зумовлена зростанням антропогенного навантаження, урбанізацією, розвитком промисловості та інфраструктури, що призводить до забруднення довкілля, деградації земель і виснаження природних ресурсів. Значні порушення природного середовища виникають унаслідок промислової та гірничо-технічної діяльності, що у ряді випадків може мати незворотні наслідки.

Охорона навколишнього природного середовища та забезпечення екологічної безпеки є необхідною умовою сталого соціально-економічного розвитку України. Відносини у цій сфері регулюються Законом України «Про охорону навколишнього природного середовища», а також спеціальними нормативно-правовими актами земельного, водного, лісового та природоохоронного законодавства [25].

Основними принципами охорони навколишнього природного середовища є пріоритет екологічної безпеки, запобіжний характер природоохоронних заходів, обов'язковість екологічної експертизи, науково обґрунтоване нормування впливу господарської діяльності, компенсація завданої шкоди та екологізація виробництва [25].

6.2 Правова охорона навколишнього середовища у промисловості

Промисловість є одним з основних джерел забруднення навколишнього природного середовища. Найбільший негативний вплив спричиняють металургійна, паливно-енергетична, хімічна та нафтопереробна галузі. У зв'язку з цим екологізація виробничих процесів є обов'язковою складовою сучасної економічної діяльності.

Зменшення шкідливого впливу промисловості досягається шляхом удосконалення технологічних процесів, впровадження очисних споруд, застосування маловідходних і безвідходних технологій, а також суворого дотримання нормативів гранично допустимих викидів і скидів забруднюючих речовин.

Загальні вимоги охорони довкілля поширюються на всі стадії господарської діяльності: передпроектну, експлуатаційну та післяексплуатаційну. Вони включають екологічну паспортизацію підприємств, нормування використання природних ресурсів, отримання дозволів на викиди та здійснення екологічного контролю.

Екологічний паспорт підприємства є нормативно-технічним документом, що містить узагальнену інформацію про використання природних ресурсів, обсяги забруднення та заходи з охорони навколишнього природного середовища. Він слугує основою для контролю екологічної безпеки виробничої діяльності.

Чинне законодавство України встановлює обов'язок суб'єктів господарювання раціонально використовувати природні ресурси, запобігати

їх забрудненню та відшкодовувати шкоду, завдану довкіллю. За порушення природоохоронних вимог передбачена дисциплінарна, адміністративна, цивільна та кримінальна відповідальність [25].

Особливу увагу приділено поводженню з відходами виробництва. Підприємства зобов'язані здійснювати заходи щодо зменшення обсягів відходів, їх переробки, знешкодження або безпечного захоронення відповідно до встановлених лімітів та нормативів.

6.3 Висновки по розділу 6.

1. Охорона навколишнього природного середовища є важливою складовою сталого розвитку та повинна враховуватися під час супроводу інформаційних систем, зокрема шляхом скорочення паперового документообігу та впровадження цифрових технологій.

2. Перехід до електронних форм комунікації та обробки інформації сприяє зменшенню антропогенного навантаження на довкілля та відповідає вимогам екологічної безпеки, визначеним чинним законодавством України.

ЗАГАЛЬНІ ВИСНОВКИ

Робота присвячена вирішенню проблемних питань розвитку методологічного підґрунтя на основі системного підходу до розв’язання практичних задач з проєктування утиліти для СУБД SQLite.

1. Проаналізовано проблемні питання з проєктування утиліт для засобів адміністрування
2. Сформульовано вимоги до утиліти для СУБД SQLite
3. Розроблено проєкт на основі результатів системного аналізу засобів адміністрування
4. Оцінено ефективність впровадження утиліти для СУБД SQLite
5. Розглянуто заходи щодо охорони праці та навколишнього середовища

Науково-практичне значення отриманих результатів полягатиме у спрощених моделях для проєктування утиліти, в основу розробки і дослідження яких покладено результати системно-цільового і системно-функціонального аналізу управління реляційними базами даних.

Робота пройшла апробацію на науково-методичних семінарах кафедри Інформаційних управляючих систем та технологій Навчально-наукового інституту комп’ютерних наук та управління проєктами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бутко М.П., Бутко І.М., Дітковська М.Ю. Системний підхід і моделювання в наукових дослідженнях : підручник. – К.: «Центр учбової літератури». 2014. – 360 с.
2. Доценко С.І., Брежнев Є.В., Будніченко Є.М. Природні інтелектуальні системи: протиріччя методологій цілісного і системного підходів та шляхи їх подолання. *Інтелектуальні системи і методи керування*. 2021. № 1 (97). С. 9-30.
3. Прокопенко Т.О. Теорія систем і системний аналіз : навч. посібник. – Черкаси: ЧДТУ, 2019. – 139 с.
4. Гриценко К.Г., Яценко В.В., Братушка С.М. Системний аналіз та економічна кібернетика : підручник. – Суми : СумДУ, 2024. – 322 с.
5. Криворучко О.В., Костюк Ю.В., Самойленко Ю.О. Сценарно-цільовий аналіз технологічного процесу виробництва вершкового масла. *Таврійський вісник*. 2021. С. № 6. С. 66-76.
6. Теслер Г.С., Косс В.А. Методика системного аналізу з позиції метрології системного підходу для потреб проектування систем управління. *Математичні машини і системи*. 2002. № 1. С. 139-150.
7. Доровський О.В. Структурно-цільовий аналіз сценаріїв розвитку фармацевтичної галузі України. *Математичні методи і моделі в економіці*. 2015. № 1. С. 340-348 с.
8. Варенко В.М., Братусь І.В., Дорошенко В.С., Смольников Ю.Б., Юрченко В.О. Системний аналіз інформаційних процесів: навч. посібник. – К.: Університет «Україна», 2013. – 203 с.
9. Тарасенко А.М. Структурно-функціональний аналіз відповідальності: пошук детермінант. *Наука і освіта*. 2016. № 9. С. 169-174.
10. Дьяченко О.П. Структурно-функціональний аналіз механізмів державного регулювання процесів протидії розвитку тіньової економіки в Україні. *Інвестиції: практика та досвід*. 2018. № 6. С. 71-75.

11. Шелковська І., Козарб В., Міхно П., Гальченко . Структурно-функціональний аналіз землекористувань в містах Полтавського регіону. Просторовий розвиток. 2024. № 9. С. 291-333.

12. Єршова О.Л., Ставицький О.В. Дослідження сучасних тенденцій у технологіях без даних і сховищ даних як технологічної та архітектурної основи для створення програмних і інтелектуальних систем засобами сучасних мов програмування. *Науковий вісник Національної академії наук та аудиту*. 2022. № 1-2. С. 77-85.

13. Шелестов А.Ю., Кассуль Н.М., Скакун С.В., Кравченко О.М., Волошин С.В., Загородній Є.В. Геоінформаційна система моніторингу для сільськогосподарського підприємства. *Наукові праці Дон НТУ. Серія «Інформатика, кібернетика та обчислювальна техніка»*. 2011. № 13 (185). С. 121-125.

14. Резніченко О.В., Ляшенко О.А., Архіпова В.В. Вибір між реляційними та нереляційними базами даних. *Технічні науки та технології*. 2024. № 3 (37). С. 162-166.

15. Шаряк В. Методи дослідження системних характеристик моделей бази даних. *Приладобудування та інформаційно-вимірювальні системи*. 2024. С. 116-121.

16. Кузьма К.Т. Програмування мобільних пристроїв : навч. посібник для дистанційного навчання. – Миколаїв : СПД Румянцева Г.В., 2021. – 128 с.

17. Водка О.О., Дашкевич А.О., Іванченко І.В., Розова Л.В., Сенько А.В. Основи програмування на C++ : навч. посібник. Нац. техн. ун-тет «Харківський політ. ін-т». – Харків, 2021. – 112 с.

18. Яковенко А.В. Основи програмування Python. Частина 1 : навч. посібник. – Київ: КПІ ім. Ігоря Сікорського, 2018. – 195 с.

19. Соловей Л.В., Мірошніченко Н.М., Бабак Т.Г. Програмування мовою C#Visual Studio. 2019. Лабораторний практикум: у 3-х ч. Ч.1. – Харків : НТУ «ХПІ», 2022. – 120 с.

20. Проданова Л.В., Томчук О.В. Економічна політика підтримки розвитку ІТ-сфери. *Економіка і організація управління*. 2023. Вип. № 1 (49). С. 45-60.

21. Голінько В.І., Іконніков М.Ю., Лебедєв Я.Я. Охорона праці в галузі інформаційних технологій : навч. посібник. – Д.: НТУ, 2015. – 246 с.

22. Закон України «Про охорону праці» від 14.10.1992 № 2694-XII (чинна редакція). Верховна Рада України. URL: <https://zakon.rada.gov.ua/go/2694-12>

23. ДСанПіН 3.3.2-007-98 «Гігієнічні вимоги до організації роботи з відеодисплейними терміналами» (постанова Головного держсанлікаря України від 10.12.1998 № 7).

24. НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин».

25. Закон України «Про охорону навколишнього природного середовища». URL: <https://zakon.rada.gov.ua/laws/show/1264-12#Text>

ДОДАТОК

ФРАГМЕНТИ ЛІСТИНГУ

Структура утиліти

```
sqlite_admin/  
├── db/  
│   └── example.db  
├── core/  
│   ├── database.py  
│   ├── models.py  
│   └── admin.py  
├── ui/  
│   └── cli.py  
└── main.py
```

Рівень доступу до даних

```
import os  
import sqlite3  
from contextlib import contextmanager  
  
DB_PATH = os.path.join("db", "example.db")  
  
@contextmanager  
def get_connection(db_path=DB_PATH):  
    conn = sqlite3.connect(db_path)  
    conn.row_factory = sqlite3.Row  
    try:  
        yield conn  
        conn.commit()  
    except Exception:  
        conn.rollback()  
        raise  
    finally:  
        conn.close()
```

SQLAlchemy-модель (проектування структури)

```
from sqlalchemy import Column, Integer, String, Float  
from sqlalchemy.orm import declarative_base  
  
Base = declarative_base()
```

```
class Product(Base):
    __tablename__ = "products"

    product_id = Column(Integer, primary_key=True)
    name = Column(String, nullable=False)
    price = Column(Float)
    stock = Column(Integer)
```

Прикладні адміністративні операції

```
from core.database import get_connection
```

```
class SQLiteAdmin:

    def list_tables(self):
        with get_connection() as conn:
            cursor = conn.execute(
                "SELECT name FROM sqlite_master WHERE type='table';"
            )
            return [row["name"] for row in cursor.fetchall()]

    def describe_table(self, table_name):
        with get_connection() as conn:
            cursor = conn.execute(f"PRAGMA table_info({table_name});")
            return cursor.fetchall()

    def execute_sql(self, sql: str):
        with get_connection() as conn:
            cursor = conn.execute(sql)
            return cursor.fetchall() if cursor.description else None
```

Консольний інтерфейс (апробація)

```
from core.admin import SQLiteAdmin
```

```
class CLI:
    def __init__(self):
        self.admin = SQLiteAdmin()

    def run(self):
        while True:
            print("\nSQLite Admin Utility")
            print("1. List tables")
            print("2. Describe table")
            print("3. Execute SQL")
            print("0. Exit")
```

```
choice = input("Select option: ")

if choice == "1":
    tables = self.admin.list_tables()
    print("Tables:", tables)

elif choice == "2":
    name = input("Table name: ")
    info = self.admin.describe_table(name)
    for col in info:
        print(dict(col))

elif choice == "3":
    sql = input("SQL> ")
    result = self.admin.execute_sql(sql)
    if result:
        for row in result:
            print(dict(row))
    else:
        print("Query executed successfully.")

elif choice == "0":
    break
```