МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Київський національний університет будівництва та архітектури

ОРГАНІЗАЦІЯ БАЗ ДАНИХ І ЗНАНЬ

Методичні вказівки
до виконання лабораторних робіт
для здобувачів спеціальності 122 «Комп’ютерні науки»,
галузь знань 12 «Інформаційні технології».

Київ 2024

УДК 004(075.8)
М

Укладач Бардін Ярослав Олегович, асистент кафедри інформаційних технологій

Рецензент О.В. Горда, канд. техн. наук, доцент, Київський національний університет будівництва і архітектури

Відповідальна за випуск завідувач кафедри інформаційних технологій Т.А. Гончаренко, канд. техн. наук, доцент.

Затверджено на засіданні кафедри інформаційних технологій, протокол № 7 від 09 лютого 2024 року.

В авторській редакції.

Організація баз даних і знань: методичні вказівки до виконання лабораторних робіт / уклад.: К Бардін Я.О. – Київ: КНУБА, 2024. – 30 с.

Містять зміст, порядок оформлення і вказівки до виконання лабораторних робіт.

Призначено для здобувачів спеціальностей 122 «Комп'ютерні науки», галузь знань 12 «Інформаційні технології».

Зміст

Загальні положення 4

Лабораторна робота № 1 Концептуальне (інфологічне) моделювання БД 6

Лабораторна робота № 2 Даталогічне проектування 8

Лабораторна робота № 3 Налаштування СУБД та створення бази даних 10

Лабораторна робота № 4 Створення первинних ключів БД 12

Лабораторна робота № 5 Створення зовнішніх ключів БД 13

Лабораторна робота № 6 Створення індексів БД 15

Лабораторна робота № 7 Команди додавання та модифікації даних 17

Лабораторна робота № 8 Використання запитів SQL для відображення інформації 20

Лабораторна робота № 9 Використання агрегатних функцій в мові SQL 22

Лабораторна робота № 10 З’єднання таблиць та групові операції 23

Лабораторна робота № 11 Створення view (подання, погляду) БД 25

Лабораторна робота № 12 Створення тригерів БД 26

Лабораторна робота № 13 Використання підзапитів мовою SQL 27

Контрольні питання 28

Список використаної літератури 29

ЗАГАЛЬНІ ПОЛОЖЕННЯ

Метою лабораторних робіт є закріплення і практичне опрацювання лекційного матеріалу, ознайомлення зі спеціалізованими утилітами призначеними для адміністрування баз даних та серверів, отримання навичок вирішення конкретних задач при проектуванні та реалізації застосувань в середовищі СУБД SQLITE.

Лабораторні роботи включають різнопланові завдання, що повністю охоплюють теоретичний курс і направлені на підготовку здобувачів до виконання курсової роботи. Лабораторні роботи мають бути виконані на базі персональних комп'ютерів з використанням засобів СУБД SQLITE та візуального середовища розробки DBEAVER.

Робота виконується індивідуально кожним здобувачем. В окремих випадках допускається об’єднання студентів в групи (2 здобувачі) для роботи над складними темами.

Результати виконання кожної лабораторної роботи подаються у вигляді створених файлів бази даних та відлагоджених програмних скриптів. Відлагодження програмних скриптів здійснюється здобувачами в предметному середовищі, що є предметом дослідницької роботи здобувача.

Лабораторна робота № 1

Тема: Концептуальне (інфологічне) моделювання БД.

Мета: закріпити та поглибити знання здобувачів, які отримані ними на лекціях та під час самостійної роботи з питань розробки концептуальної моделі бази даних.

Завдання: розробити концептуальну модель бази даних для обраної предметної області.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи.

Рекомендації до виконання

1.1. Вибір предметної області для проектування БД

Спектр використання баз даних досить широкий і має міждисциплінарний характер – це державне управління, науково-дослідницька діяльність, банківська та страхова справа, логістика, управління складським господарством на підприємствах, геоінформаційні системи, бібліотечна справа та створення різноманітних електронних каталогів, управління контентом інтернет-сайтів тощо.

Розглянемо основні поняття і терміни, які використовуються в теорії баз даних.

База даних – це сукупність структурованих, логічно взаємопов’язаних даних, що зберігаються та використовуються спільно. Це потужний засіб для збирання та впорядкування даних, який призначений для задоволення інформаційних потреб користувачів.

Кожна база даних описує певну предметну область, тобто певну визначену частину реального світу.

На першому етапі виконання лабораторної роботи необхідно визначитись з предметною областю для проектування власної бази даних. Пропонується розробити базу із переліку, наведеного в табл. 1.1.

1.2. Інфологічне (концептуальне) проектування БД.

На першому етапі проектування бази даних необхідно визначитись з групами об’єктів, інформація про які буде в ній зберігатися. Ці групи однотипних об’єктів можуть бути будь-якої природи та мають назву «сутності».

Таблиця 1.1

Перелік тем для проектування БД

Варіант	Назва теми
1.	Бібліотека та читацький зал
2.	Відділ кадрів
3.	Облік проживання в гуртожитку
4.	Облік контрактного навчання в університеті
5.	Розклад занять в університеті
6.	Відвідування занять та рейтинг студентів
7.	Облік інтернет послуг в гуртожитку
8.	Деканат факультету
9.	Облік робочого часу кафедри
10.	Облік відвідувачів університету («турнікети»)
11.	Склад та оренда майна в гуртожитку
12.	Розрахунок стипендії
13.	Облік на складі будівельних матеріалів
14.	Кошторис витрат та облік ремонтних робіт в університеті

При складанні інфологічної моделі необхідно визначитись з переліком сутностей. Під час опису сутностей потрібно дослідити зв’язки між ними та зазначити їх тип. При аналізі сутностей предметної області також визначають обмеження, що накладаються на зв’язки між сутностями та на діапазони допустимих значень, що будуть зберігатись у БД. У звіті про виконання лабораторної роботи необхідно навести графічну діаграму типу сутність-зв'язок (ER-діаграму) із зазначенням всіх сутностей та зв’язків між ними (рис.1.1).

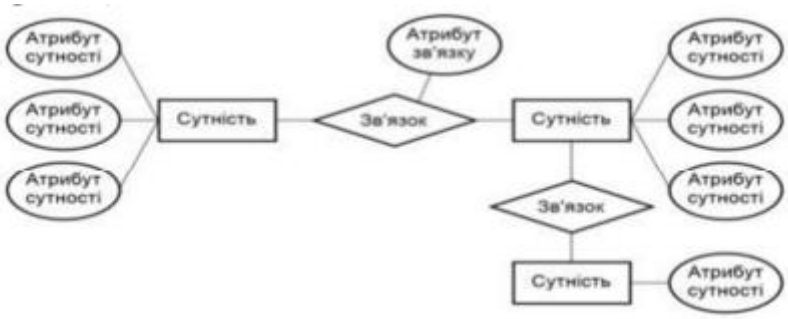


Рисунок 1.1. Приклад оформлення діаграми типу сутність-зв'язок

Для автоматизації процесу підготовки таких моделей можна використовувати Microsoft Visio, SAP Power Designer або он-лайн сервіси diagrams.net та lucidchart.com.

Лабораторна робота № 2

Тема: Даталогічне проектування БД. Нормалізація відношень.

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань розробки даталогічної моделі бази даних.

Завдання: розробити даталогічну модель бази даних для обраної предметної області.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

1.1. Даталогічне (логічне) проектування БД.

Зазвичай, сутності описують за допомогою таблиць, де властивості екземплярів даної групи об'єктів представлені у вигляді множини атрибутів. Отже, атрибути – це властивості, що описують сутність. Множина атрибутів повинна однозначно визначати кожен екземпляр сутності, тобто бути унікальною.

Для спрощення ідентифікації окремих екземплярів сутностей використовують поняття «первинний ключ». Це поле чи сукупність полів, що містять унікальні значення, за якими однозначно визначають той чи інший екземпляр сутності.

Попередньо розроблену інфологічну (концептуальну) модель БД необхідно перевести у даталогічну (схему бази даних). Для кожної сутності необхідно призначити окрему таблицю та вказати для неї атрибути, зв'язки та ключові поля. Для атрибутів треба вказати назви полів, типи даних, обмеження для даних, синоніми полів (якщо планується їх використання).

В якості предметної області оберемо склад будівельних матеріалів університету. Створення БД дозволить вирішувати багато поточних задач організації, зокрема: – автоматизувати облік; – підвищити ефективність. Для досягнення навчальної мети достатньо розробити спрощену структуру, яка налічує мінімально необхідну кількість сутностей (рис. 2.1).

Для обраної предметної області такими сутностями можуть бути: «Довідник матеріально відповідальних осіб», «Документи », «Картотека», «Матеріали по документу», «Операції по документу». Для прикладу, в якості системи управління базою даних оберемо Sqlite3 . Подальшу розробку та опис

атрибутів визначених сутностей будемо здійснювати з урахуванням специфіки створення баз даних для СУБД Sqlite.

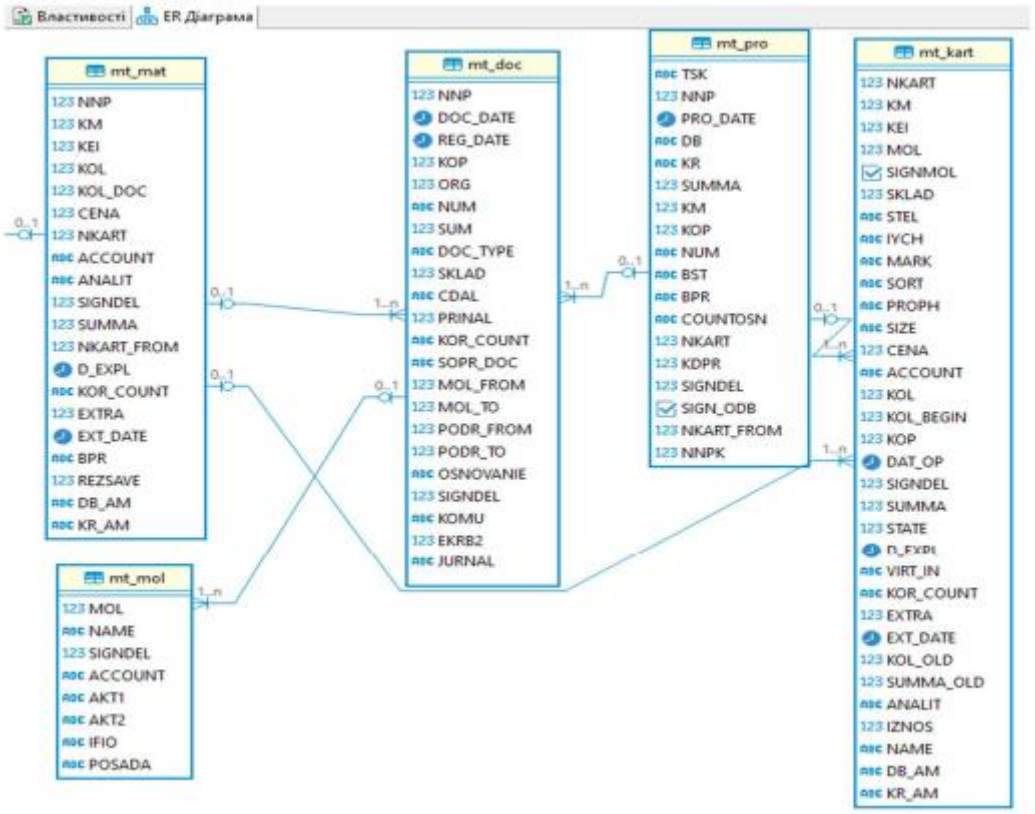


Рисунок 2.1. Приклад оформлення схеми БД

Таблиця 2.1. Приклад.

Атрибути сутності «Довідник матеріально-відповідальних осіб»

Назва атрибута (поля)	Тип даних	Опис атрибута
Mol	Integer	Код МВО
Name	Text	ПІБ МВО

Signdel	Integer	Ознака видалення
Account	Text	Обмеження
Akt1	Text	Прибуток
Akt2	Text	Видаток
Ifio	Text	Ініціали МВО
Posada	Text	Посада МВО

Лабораторна робота № 3

Тема: Налаштування СУБД та створення бази даних

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань налаштування СУБД та створення бази даних.

Завдання: обрати та налаштувати СУБД, створити структуру власної бази даних.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

1.1. Обрати одну з наведених СУБД для створення та управління базою даних за обраною темою:

–Sqlite 3;

– Microsoft SQL Server (SQL Server Management Studio);

– PostgreSQL (pgAdmin чи PostgreSQL Web Client);

– MySQL (MySQL Workbench).

Найбільш простим варіантом реалізації бази даних буде використання Sqlite та засобу для роботи з базами даних DBeaver.

Для створення нової бази даних послідовно виконайте дії на рис.3.1-3.3

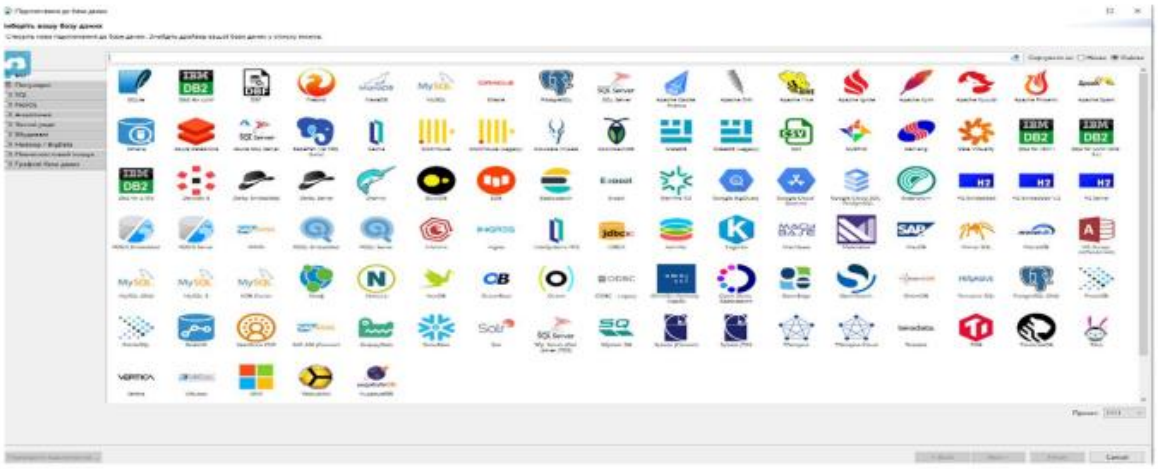


Рисунок 3.1

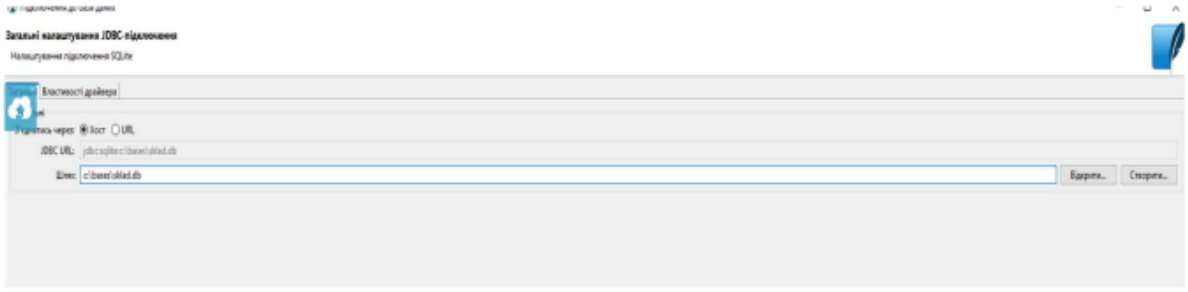


Рисунок 3.2

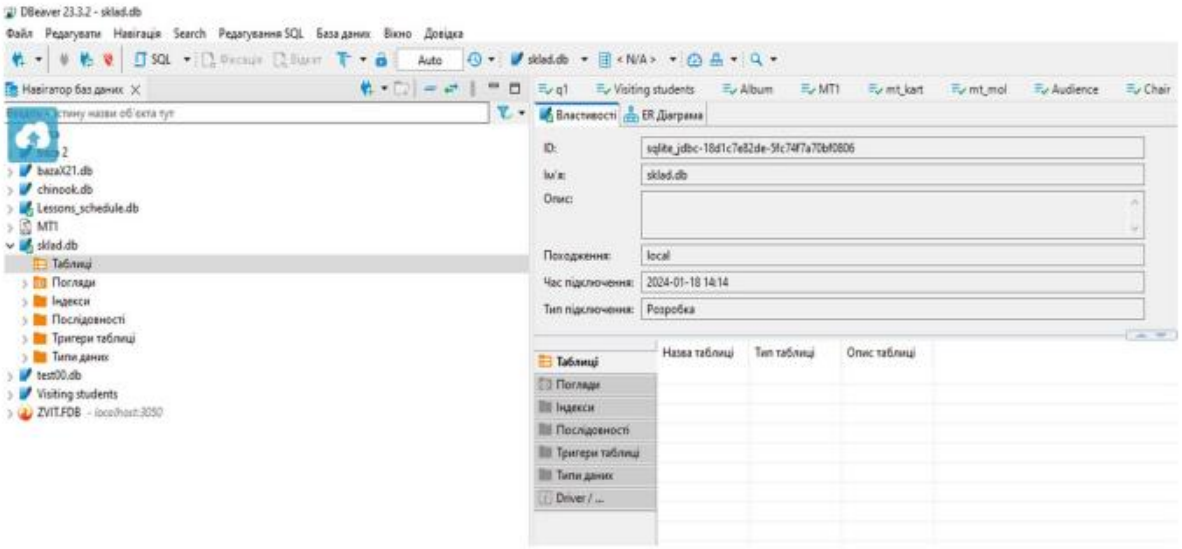


Рисунок.3.3

Використовуючи графічний інтерфейс Dbeaver або SQL консоль створіть структуру власної бази даних (рис.3.4 - 3.5).

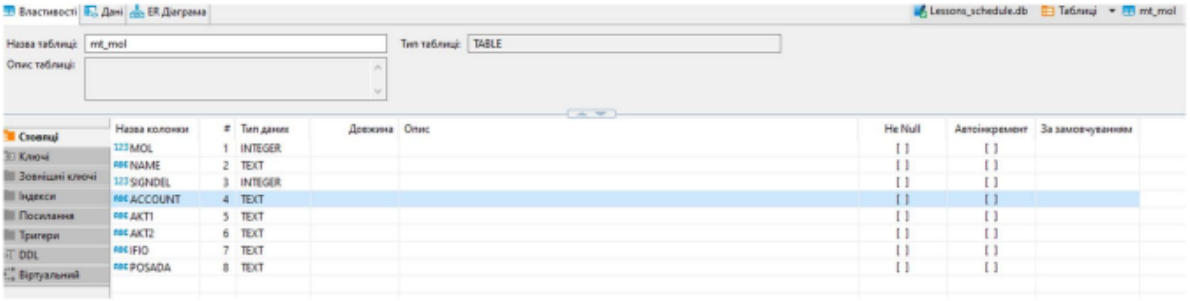
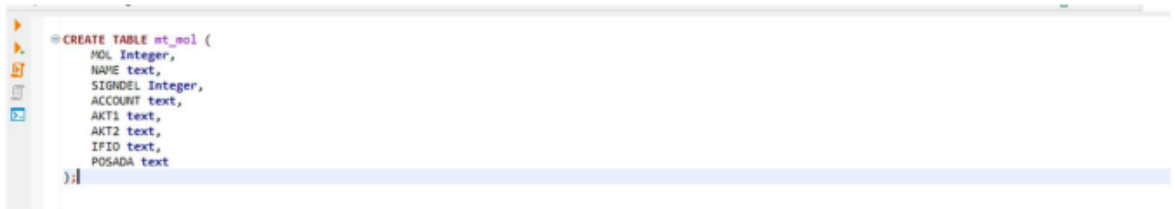


Рисунок 3.4 Приклад створення в Dbeaver



```
CREATE TABLE mt_mol (
    MOL Integer,
    NAME text,
    SIGNDEL Integer,
    ACCOUNT text,
    AKT1 text,
    AKT2 text,
    IFIO text,
    POSADA text
);
```

Рисунок 3.5 Приклад створення в SQL консолі

Лабораторна робота № 4

Тема: Створення первинних ключів БД.

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань створення бази даних. Завдання: Створити первинні ключі для таблиць власної бази даних.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

Первинний ключ – це поле або набір полів зі значеннями, унікальними в межах певної таблиці. За допомогою значень ключа можна посилатися на окремі записи, адже значення ключа кожного запису відрізняється. Кожна таблиця може мати лише один первинний ключ.

Для кожної таблиці з власної бази даних необхідно визначити первинний ключ, його властивості та за допомогою графічного інтерфейсу Dbeaver або SQL консолі створити їх (рис.4.1 - 4.2).

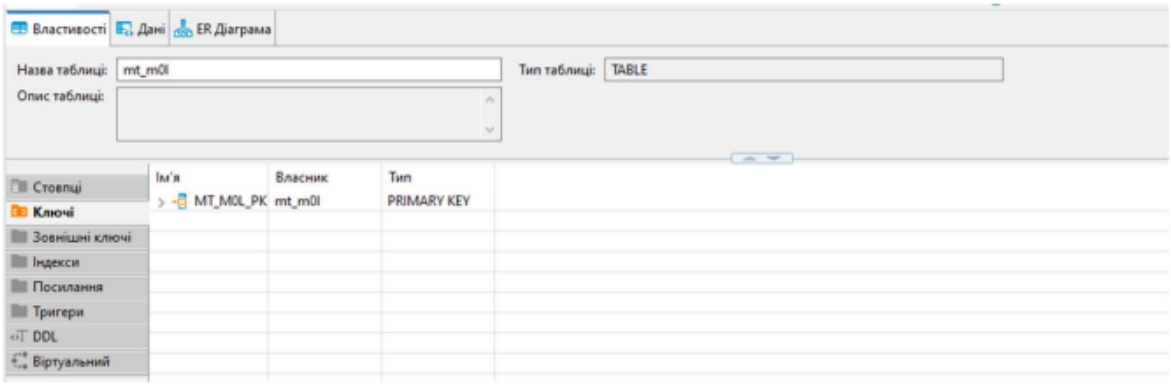


Рисунок 4.1 Приклад створенняв Dbeaver

```
CREATE TABLE mt_m0l (
    MOL  Integer PRIMARY KEY AUTOINCREMENT,
    NAME text,
    SIGNDEL Integer,
    ACCOUNT text,
    AKT1 text,
    AKT2 text,
    IFIO text,
    POSADA text
);
```

Рисунок 4.2 Приклад створення в SQL консолі

Лабораторна робота № 5

Тема: Створення зовнішніх ключів БД.
Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань створення бази даних.
Завдання: Створити зовнішні ключі для таблиць власної бази даних.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання

Рекомендації до виконання

Зовнішній ключ (FOREIGN KEY) потрібен для того, щоб зв’язати дві різні таблиці між собою. Зовнішній ключ може посилатися на будь-який стовпець у батьківській таблиці. Проте загальноприйнятою практикою є посилання зовнішнього ключа на первинний ключ (primary key) батьківської таблиці.
Для кожної таблиці з власної бази даних необхідно визначити чи потрібен зовнішній ключ, його властивості та за допомогою графічного інтерфейсу Dbeaver або SQL консолі створити їх (рис.5.1 - 5.4).

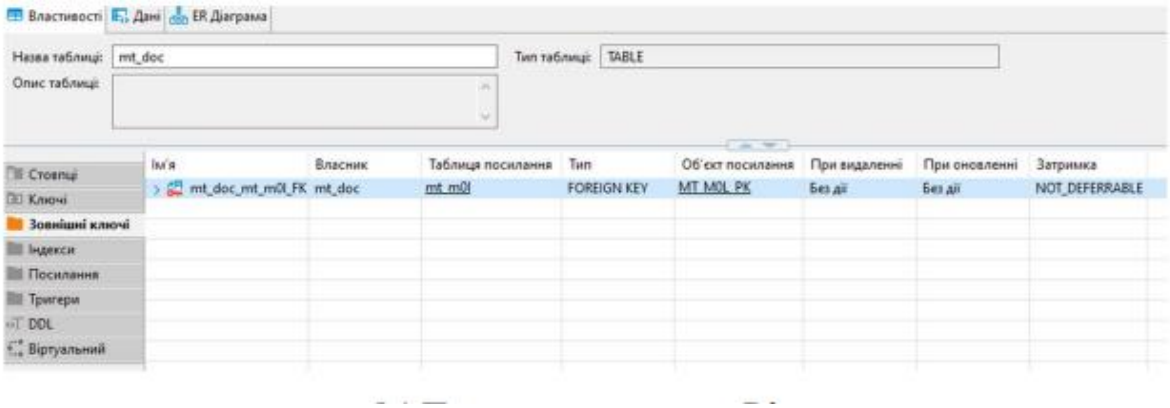


Рисунок 5.1 Приклад створення в Dbeaver

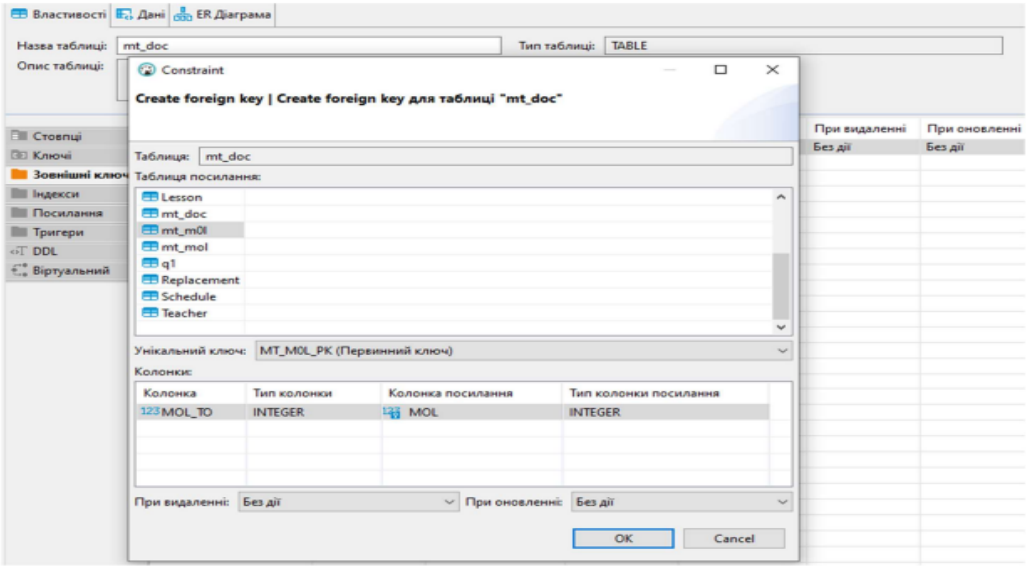


Рисунок 5.2 Приклад створення в Dbeaver

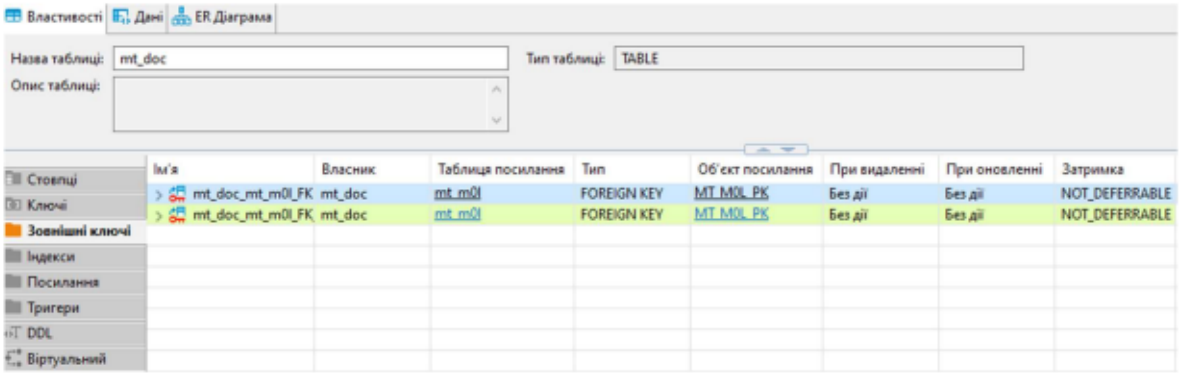
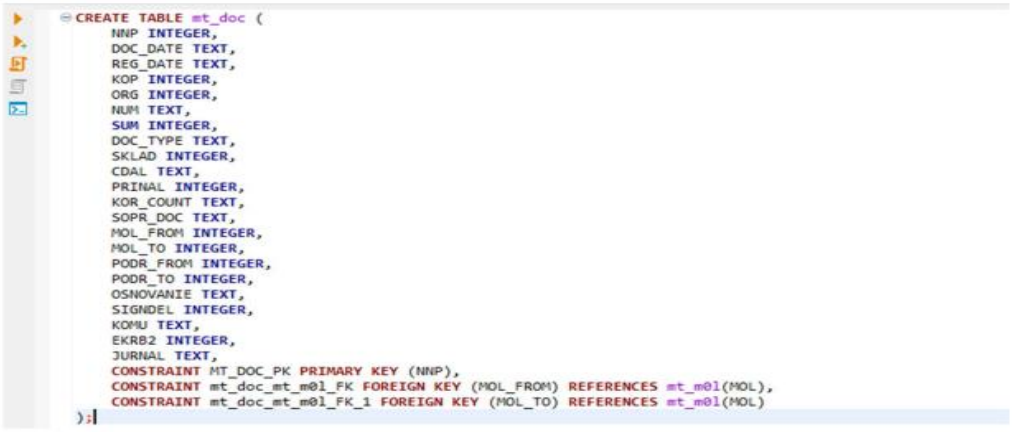


Рисунок 5.3 Приклад створення в Dbeaver



```
CREATE TABLE mt_doc (
    NNP INTEGER,
    DOC_DATE TEXT,
    REG_DATE TEXT,
    KOP INTEGER,
    ORG INTEGER,
    NUM TEXT,
    SUM INTEGER,
    DOC_TYPE TEXT,
    SKLAD INTEGER,
    CDAL TEXT,
    PRINAL INTEGER,
    KOR_COUNT TEXT,
    SOPR_DOC TEXT,
    MOL_FROM INTEGER,
    MOL_TO INTEGER,
    PODR_FROM INTEGER,
    PODR_TO INTEGER,
    OSNOVANIE TEXT,
    SIGNDEL INTEGER,
    KOMU TEXT,
    EKRB2 INTEGER,
    JURNAL TEXT,
    CONSTRAINT MT_DOC_PK PRIMARY KEY (NNP),
    CONSTRAINT mt_doc_mt_m0l_FK FOREIGN KEY (MOL_FROM) REFERENCES mt_m0l(MOL),
    CONSTRAINT mt_doc_mt_m0l_FK_1 FOREIGN KEY (MOL_TO) REFERENCES mt_m0l(MOL)
);
```

Рисунок 5.4 Приклад створення в SQL консолі

Лабораторна робота № 6

Тема: Створення індексів БД.

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань створення бази даних.

Завдання: Створити зовнішні ключі для таблиць власної бази даних.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

Індекс (англ. index) — об'єкт бази даних, що створений з метою підвищення ефективності виконання запитів. Таблиці в базі даних можуть мати велику кількість рядків, які зберігаються у довільному порядку, і їх пошук за заданим значенням шляхом послідовного перегляду таблиці рядок за рядком може займати багато часу. Індекс формується зі значень одного чи кількох стовпчиків таблиці і вказівників на відповідні рядки таблиці і, таким чином, дозволяє знаходити потрібний рядок за заданим значенням. Прискорення роботи з використанням індексів досягається в першу чергу за рахунок того, що індекс має структуру, що оптимізована для пошуку.

Для кожної таблиці з власної бази даних необхідно визначити чи потрібні індекси, їх властивості та за допомогою графічного інтерфейсу Dbeaver або SQL консолі створити їх (рис.6.1 - 6.4).

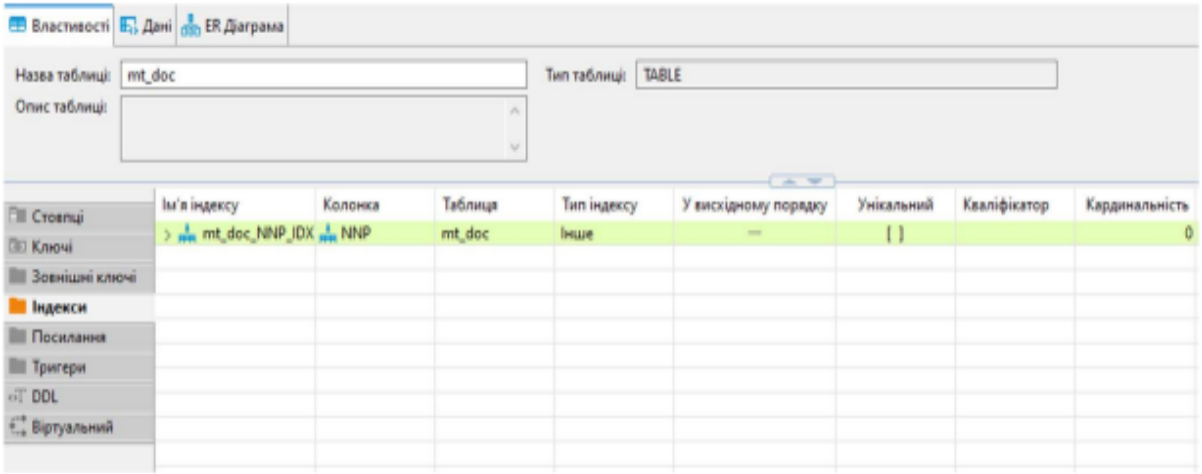


Рисунок 6.1 Приклад створення в Dbeaver

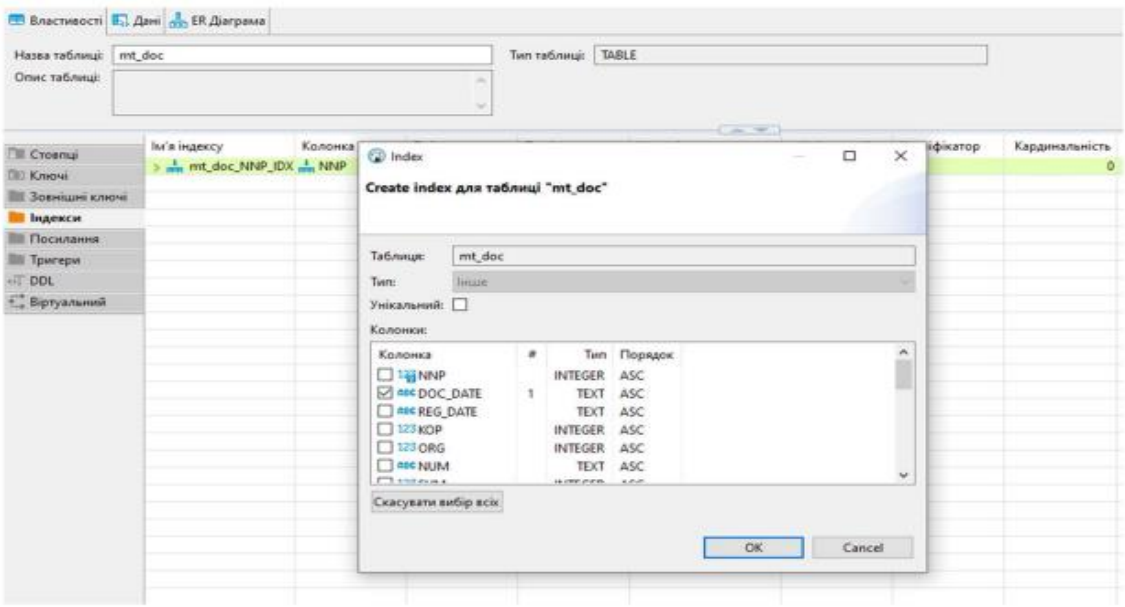


Рисунок 6.2 Приклад створення в Dbeaver

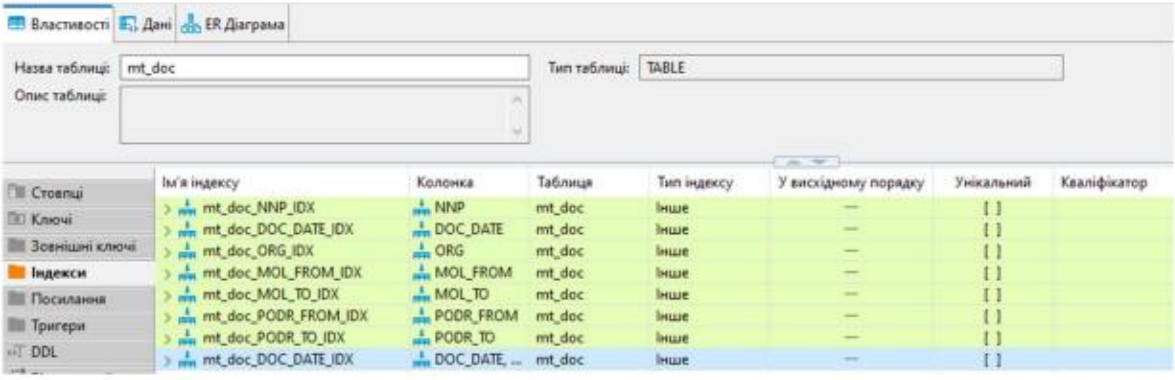


Рисунок 6.3 Приклад створення в Dbeaver

```
CREATE INDEX mt_doc_NNP_IDX ON mt_doc (NNP);
CREATE INDEX mt_doc_DOC_DATE_IDX ON mt_doc (DOC_DATE);
CREATE INDEX mt_doc_ORG_IDX ON mt_doc (ORG);
CREATE INDEX mt_doc_MOL_FROM_IDX ON mt_doc (MOL_FROM);
CREATE INDEX mt_doc_MOL_TO_IDX ON mt_doc (MOL_TO);
CREATE INDEX mt_doc_PODR_FROM_IDX ON mt_doc (PODR_FROM);
CREATE INDEX mt_doc_PODR_TO_IDX ON mt_doc (PODR_TO);
CREATE INDEX mt_doc_DOC_DATE_IDX ON mt_doc (DOC_DATE,NUM,DOC_TYPE);
```

Рисунок 6.4 Приклад створення в Dbeaver

Лабораторна робота № 7

Тема: Команди модифікації даних

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань використання команд модифікації даних.

Завдання: використовуючи оператор модифікації даних, створити запити за заданими умовами.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

1.1. Створити запити на використання оператору INSERT INTO, додавши нові дані до двох різних таблиць, які є довідниками (рис. 7.1-7.2).

Наприклад, додати себе, як МВО з можливістю видачі матеріалів зі складу.

```
INSERT INTO mt_m01
(MOL, NAME, SIGNDEL, ACCOUNT, AKT1, AKT2, IFIO, POSADA)
VALUES(1, 'Ткач Іван Федорович', 0,'True', 'True', 'True', 'І.Ф.', 'Завідувач складом');
```

Рисунок 7.1 Приклад запиту

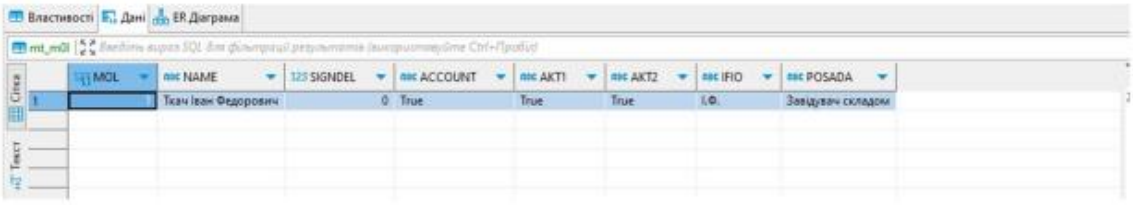


Рисунок 7..2 Результат виконання запиту

1.2. Створити 2 запита із використанням оператору UPDATE, змінивши дані, які були додані в завданні 1.1. Наприклад, змінити посаду на іншу.

1.3. Написати запит із використанням оператора INSERT INTO table SELECT, за допомогою якого створити копію однієї з таблиць (рис. 7.3 – 7.7).

Наприклад, скопіювати таблицю "Документи" в таблицю "Документи_копія".

1.4. Створити запит, за допомогою якого очистити вміст таблиці, створеної в завданні 1.3, використавши для цього оператор DELETE. Наприклад, очистити вміст таблиці "Mt_doc_copy".

```
UPDATE Mt_m01 SET POSADA = "заступник директора"
WHERE name = "Ткач Іван Федорович" ;
```

Рисисунок 7.3.

	MOL	NAME	SIGNDEL	ACCOUNT	AKT1	AKT2	IFIO	POSADA
1	1	Ткач Іван Федорович	0	True	True	True	І.Ф.	заступник директора

Рисунок 7.4 Приклад та результат виконання запиту

```
CREATE TABLE mt_doc_copy (
    NNP INTEGER,
    DOC_DATE TEXT,
    REG_DATE TEXT,
    KOP INTEGER,
    ORG INTEGER,
    NUM TEXT,
    SUM INTEGER,
    DOC_TYPE TEXT,
    SKLAD INTEGER,
    CDAL TEXT,
    PRINAL INTEGER,
    KOR_COUNT TEXT,
    SOPR_DOC TEXT,
    MOL_FROM INTEGER,
    MOL_TO INTEGER,
    PODR_FROM INTEGER,
    PODR_TO INTEGER,
    OSNOVANIE TEXT,
    SIGNDEL INTEGER,
    KOMU TEXT,
    EKRB2 INTEGER,
    JURNAL TEXT
    );
insert into mt_doc_copy SELECT * FROM mt_doc
```

Рисунок 7.5 Приклад запиту. Варіант 1.

```
CREATE TABLE mt_doc_copy As Select * FROM mt_doc;
insert into mt_doc_copy SELECT * FROM mt_doc
```

Рисунок 7.6.Приклад запиту. Варіант 2.

```
delete from mt_doc_copy
```

Рисунок 7.7 Приклад команди

1.5. Створити 2 запити із використанням команди DELETE з використання параметру WHERE для видалення записів із таблиці (рис.7.8 – 7.9).
DELETE FROM …WHERE …
1.6. Створити 2 запити, в яких за допомогою інструкції DELETE видалити фіксовану кількість записів, що відповідають заданій умові. Умову вказати у підзапиті.
1.7. Створити 2 запити на внесення змін до таблиці за допомогою команди UPDATE із використанням підзапиту та параметру.
1.8. Створити запит на видалення таблиці за допомогою команди DROP TABLE.

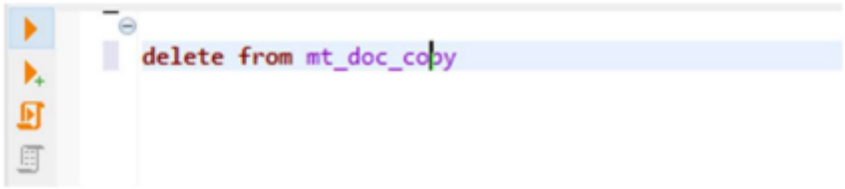


Рисунок 7.8 Приклад команди

1.9. Створити 2 запити з використанням команди ALTER TABLE ADD, за допомогою яких додати до таблиць нову колонку. Наприклад, додати до таблиці "МВО" стовпець із назвою «Графік»

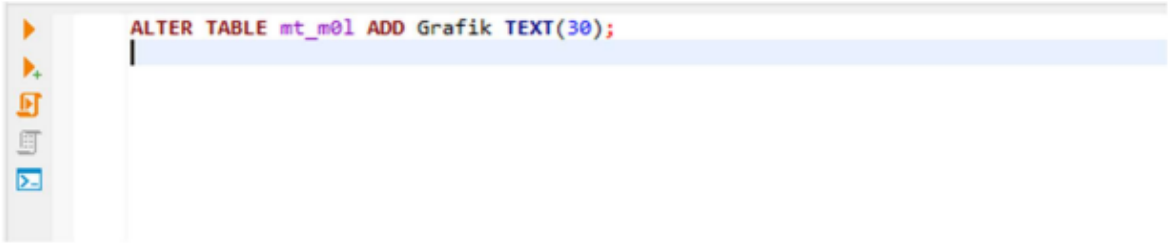


Рис.7.9 Приклади команди

1.10. Створити 2 запити із використанням команди ALTER TABLE DROP, за допомогою яких видалити деякі колонки із таблиці.

Лабораторна робота № 8

Тема: Використання мови SQL для відображення інформації

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань використання команди SELECT мови SQL.

Завдання: використовуючи команду SELECT, створити запити за заданими умовами.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

Найбільш популярною інструкцією мови SQL є інструкція SELECT. Загальна структура цієї інструкції в SQLITE має вигляд (рис.8.1):

```
SELECT DISTINCT column_list
FROM table_list
  JOIN table ON join_condition
WHERE row_filter
ORDER BY column
LIMIT count OFFSET offset
GROUP BY column
HAVING group_filter;
```

Рисунок 8.1. Загальна структура інструкції в SQLITE

1.1. Використовуючи інструкцію SELECT, створити 3 запита на виведення всіх записів з обраних вами таблиць (одна таблиця – один запит) Наприклад, вивести всі записи, що зберігаються в таблицях «Документи» (рис.8.2).

Рисунок 8.2.Запит та фрагмент результату виконання запиту.

1.2. За допомогою інструкції SELECT створити 3 запита на виведення окремих атрибутів обраної таблиці.
1.3. Створити 3 запита на виведення даних без повторів значень.
Наприклад, можна вивести список дат, коли оформлялись документи (рис.8.3).

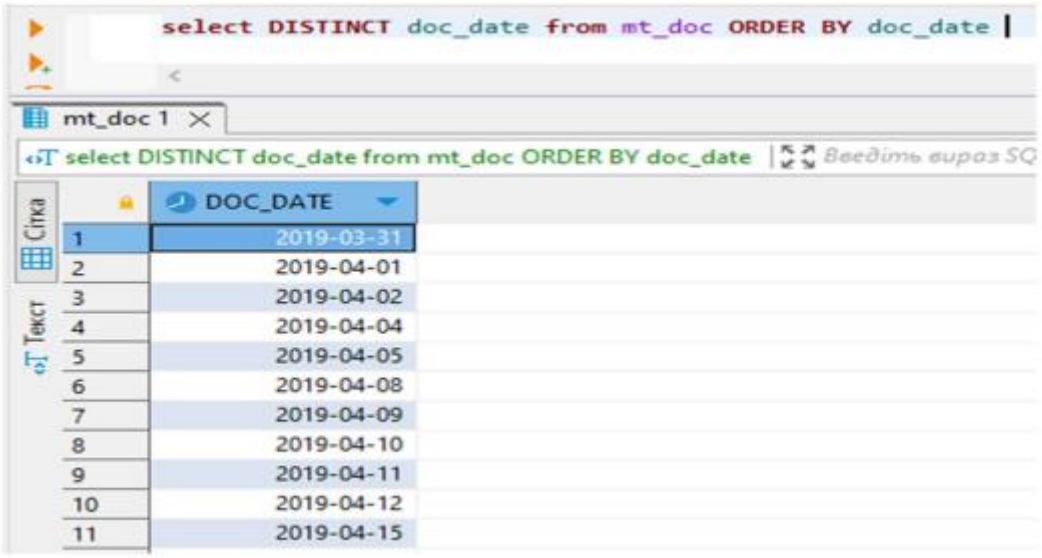


Рисунок 8.3 Запит та фрагмент результату виконання запиту.

1.4. Створити 3 запита на виведення даних з подвійною умовою.
1.5. Створити 3 запита на виведення даних з подвійною умовою, використанням логічних операторів (AND, OR, NOT) та реляційних операторів (=, <>, >,=,<=, !=, !<).
1.6. Створити 3 запита на використання спеціального оператору IN.
1.7. Створити 3 запита на використання спеціального оператору BETWEEN.
8.8. Створити 3 запита на використання оператора IS NULL для пошуку порожніх значень у комірках таблиць.

Лабораторна робота № 9

Тема: Агрегатні функції

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань використання агрегатних функцій в запитах.

Завдання: використовуючи агрегатні функції мови SQL, створити запити за заданими умовами.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

Агрегатні функції розраховують єдине значення для колекції вхідних значень.

Функція AVG – обчислює середнє арифметичне значення для заданого набору значень (для поля). При розрахунках ігноруються порожні (NULL) поля. Разом з ключовим полем ABS повертає абсолютне значення середнього значення поля .

Функція COUNT – обчислює загальну кількість записів, які отримані через запит. Обробляє значення, записані у будь-якому типі даних, зокрема текстовому. При розрахунках ігноруються порожні (NULL) поля.Для визначення загальної кількості рядків у таблиці достатньо використати простий вираз COUNT(*).

Функція MIN або MAX – повертає найменше або найбільше значення з набору даних, які отримані через запит. Цими агрегатними функціями можна обробляти не лише константи, а й поля, в яких використовуються обчислення чи інші функції (крім інших агрегатних функцій).

Функція SUM або TOTAL – повертає суму всіх чисел з набору значень, які отримані через запит. При розрахунках ігноруються порожні (NULL) поля. З її допомогою можна обробляти не тільки константи, а й поля, в яких використовуються обчислення або інші функції.

Функція group_concat() або group_concat() або string_agg()– спеціальні агрегатні функції для текстових стов

1.1. Створити 3 запита на використання спеціального оператору AVG з виведенням результату в поіменований стовпчик, використавши інструкцію AS.

1.2. Створити 3 запита на використання спеціального оператору COUNT з виведенням результату в поіменований стовпчик, використавши інструкцію AS (рис.9.1).
Наприклад, вивести в стовпчик з ім’ям "Кількість документів"

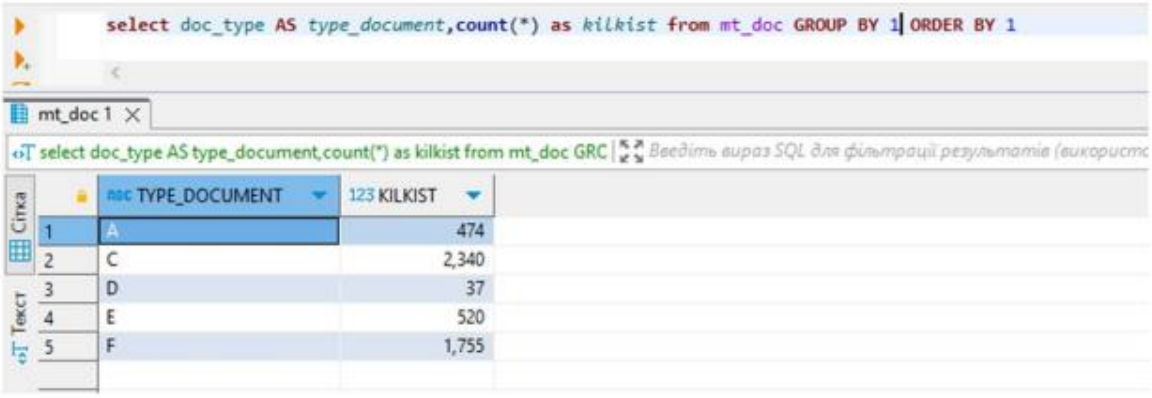


Рисунок 9.1.Запит та результати виконання запиту

1.3. Створити 3 запита на використання спеціального оператору COUNT
1.4. Створити 3 запита на використання спеціальних операторів MIN і MAX в поєднанні з умовами "WHERE" для інших атрибутів.
1.5. Створити 3 запита на використання спеціального оператору SUM в поєднанні з умовами "WHERE" для інших атрибутів.

Лабораторна робота № 10

Тема: З’єднання таблиць та групові операції

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань конкатенації та групових операцій над даними та з’єднання таблиць.

Завдання: використовуючи оператори JOIN і GROUP BY, створити запити за заданими умовами.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

JOIN – операція з'єднання двох таблиць, що утворює нову тимчасову таблицю (з'єднану таблицю).

CROSS JOIN – операція з'єднання за умовами декартового добутку. Після виконання даної операції ми отримаємо таблицю, що буде являти собою набір всіх можливих поєднань рядків двох таблиць.
INNER JOIN – операція з'єднання двох таблиць, при якій, кожен рядок з першої таблиці зіставляється з кожним рядком другої таблиці, після чого відбувається перевірка умови. Якщо умова істинна, рядки потрапляють у результуючу таблицю.
LEFT JOIN – операція з'єднання двох таблиць, яка повертає всі записи лівої таблиці доповнені відповідними кортежами з правої таблиці. У випадку, якщо в правій таблиці немає збігу, то запис буде доповнено значеннями NULL.
RIGHT JOIN – операція з'єднання двох таблиць, яка додає до записів правої таблиці відповідні кортежі лівої таблиці або значення NULL.
FULL JOIN – операція, яка поєднує записи двох таблиць за відповідним критерієм. Записи для яких не було знайдено збігу будуть доповнені значеннями NULL.
Речення ORDER BY використовують для впорядкування результатів запиту за зростанням чи зменшенням значень.
Речення GROUP BY дозволяє визначати підмножину значень в певному полі (або групі полів) в термінах іншого поля, і застосовувати функцію агрегатування до підмножини.
1.1. Створити 2 запита на використання оператору INNER JOIN, поєднавши
1.2. Створити 2 запита на використання оператору INNER JOIN, поєднавши дані з трьох таблиць.
1.3. Створити 2 запита на використання оператора ORDER BY для подвійного впорядкування даних у двох поєднаних таблицях.
1.4. Створити 2 запита на використання оператора INNER JOIN та умови WHERE з діапазоном даних про дату або час з подальшим упорядкуванням за обраним атрибутом.
1.5. Створити 2 запита на використання оператора GROUP BY спільно з агрегатними функціями.
1.6. Створити 3 запита на використання оператора GROUP BY спільно з математичними операціями.
1.7. Створити 3 запита на використання оператора GROUP BY з умовою WHERE, реченням HAVING, агрегатною функцією та впорядкуванням результату.

Лабораторна робота № 11

Тема: Створення VIEW (подання, погляду) БД.

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань оптимізації запитів бази даних.

Завдання: Створити view (подання, погляд) для власної бази даних.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

Подання (VIEW) – це об’єкт бази даних, який зберігає в собі запит SELECT і в разі звернення до даного об’єкту буде повернуто результуючий набір даних, який формує запит, зазначений у визначенні подання.

Іншими словами, це віртуальна (логічна) таблиця, вона не містить в собі даних, але до неї можна звертатися як до звичайної таблиці, і вона буде повертати вам дані. Якщо досить часто в SQL запитах використовуються однотипні вкладені запити, які повертають табличні дані, тобто є похідними таблицями, то для зручності і скорочення коду можна зберегти такі вкладені запити у вигляді подання та потім, де потрібно використовувати саме той набір даних, який вказано в запиті, можна вказати назву подання.

Для власної бази даних необхідно визначити чи потрібні подання, який набір даних буде використано та за допомогою графічного інтерфейсу Dbeaver або SQL консолі створити їх (рис.11.1 - 11.3).

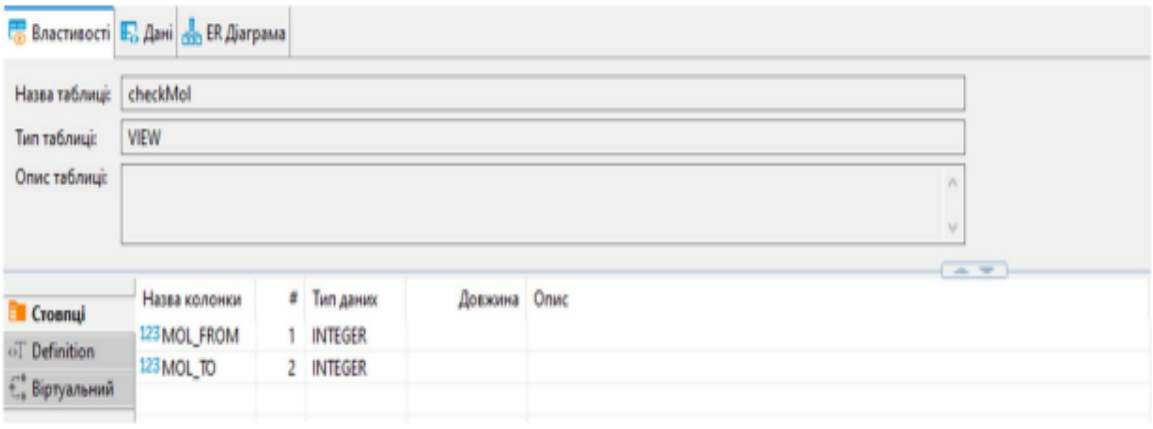


Рисунок 11.1 Приклад створення в Dbeaver

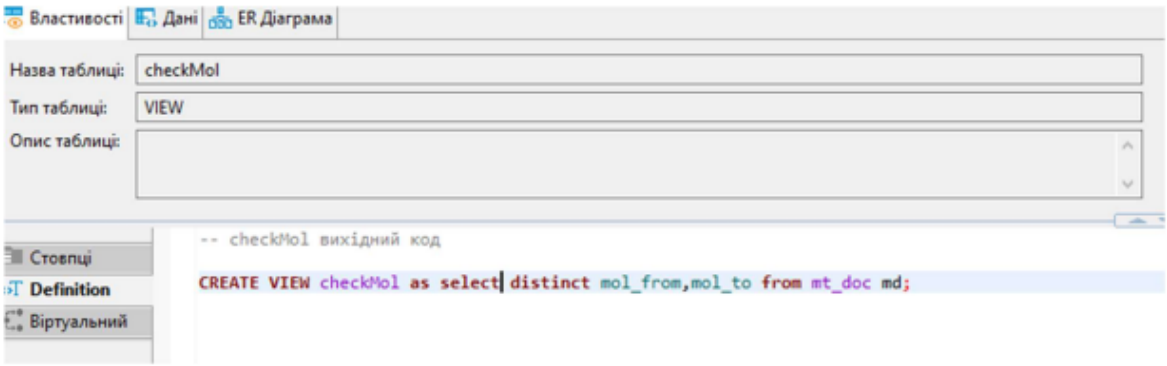


Рисунок 11.2 Приклад створення в Dbeaver

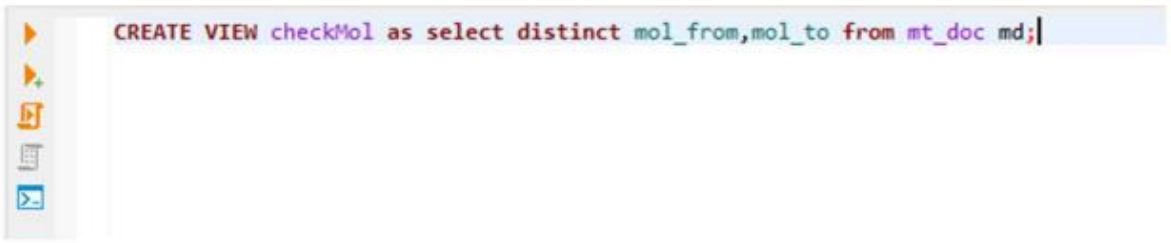


Рисунок 11.3 Приклад створення в SQL консолі

Лабораторна робота № 12

Тема: Створення тригерів БД.

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань цілісності даних та реалізації бізнес-логіки .

Завдання: Створити тригери аудиту для ключових таблиць власної бази даних.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

Тригер (TRIGGER) — це збережена процедура особливого типу, яку користувач не викликає явно, а використання якої обумовлено настанням визначеної події (дії) у реляційній базі даних: додаванням INSERT, вилученням рядка в заданій таблиці DELETE, або зміною даних у певному стовпці заданої таблиці UPDATE.

Тригери застосовуються для забезпечення цілісності даних і реалізації складної бізнес-логіки. Тригер запускається сервером автоматично при

спробі зміни даних у таблиці, з якою він пов'язаний. Всі здійснені ним модифікації даних розглядаються як виконані в транзакції, в якій виконано дію, що викликало спрацювання тригера. Відповідно, у разі виявлення помилки або порушення цілісності даних може відбутися відкат цієї транзакції.

Для власної бази даних необхідно визначити ключові таблиці БД та за допомогою SQL консолі створити тригери аудиту.

Лабораторна робота № 13

Тема: Використання підзапитів мовою SQL

Мета: закріпити та поглибити знання студентів, які отримані ними на лекціях та під час самостійної роботи з питань використання підзапитів мовою SQL.

Завдання: використовуючи оператор SELECT, створити підзапити за заданими умовами.

План заняття

1. Виконання завдання згідно з інструкцією.
2. Захист роботи та відповіді на контрольні питання.

Рекомендації до виконання

1.1 Створити 2 запита з використанням одного підзапиту, обмеження та агрегатної функції.

1.2. Створити 2 запита з використанням подвійного підзапиту.

1.3. Створити 2 запита з пошуку другої позиції в списку за вказаним атрибутом.

1.4. Створити 2 запита з пошуку екземплярів сутності, які відповідають певній умові, з груп сутностей.

1.5. Створити 2 запита з пошуку значення атрибуту однієї сутності за умови введення користувачем діапазону даних параметра іншої сутності. За умови, що сутності зв’язані між собою через ключове поле.

1.6. Створити 2 запита з пошуку найбільш популярного запису в списку за введеним користувачем параметром.

Контрольні питання

1. Які основні завдання вирішуються шляхом проектування баз даних?
2. Назвіть основні етапи проектування бази даних та коротко опишіть їх.

3. Для чого розробляють даталогічну модель бази даних?
4. Які види ключів ви знаєте? Дайте їм характеристику
5. Яким чином налаштовуються зв’язки між таблицями в обраній вами СУБД?
6. Якими командами можна знайти та вивести дані, що належать до певної множини значень?
7. Для чого використовують інструкцію WHERE?
8. Які типи даних використовуються у вашій БД?
9. Опишіть загальну структуру команди SELECT.
10.З якою метою використовують функцію AS?
11.Як знайти та вивести поточну дату, місяць, день, час?
12.Виконання яких логічних та арифметичних операцій над даними підтримує обрана вами СУБД?
13.Чим застосування умови WHERE відрізняється від застосування умови HAVING?
14.Порівняйте результат від застосування операторів RIGHT JOIN і LEFT JOIN.
15.Які особливості застосування оператора GROUP BY?
16.Який з операторів – WHERE чи HAVING інструкції SELECT буде виконаний першим?
17.Які можливості надає використання підзапитів?
18.Які можливості надає використання оператора IN?
19.Як видалити заповнену таблицю з бази даних?
20.Як за допомогою оператора INSERT внести дані лише в деякі задані

Список використаної літератури

1. Верес О. М., Пасічник В. В. Системи баз даних та знань. Книга 1: організація баз даних та знань. Київ: Магнолія, 2019. 440 с.
2. Верес О. М., Пасічник В. В., Берко А. Ю. Системи баз даних та знань. Книга 2: системи управління базами даних та знань. Київ: Магнолія, 2019. 584 с
3. Гайна Г. А. Основи проектування баз даних. Київ: Кондор, 2018. 208 с. 29
4. Демиденко М. А. Введення в сучасні бази даних: навч. посіб. НТУ «Дніпровська політехніка». Дніпро, 2020. 38 с.
5. Ярцев В. П. Організація баз даних та знань: навч. посіб. Київ. ДУТ, 2018. 214 с.
6. https://www.sqlite.org/docs.html
7. https://www.sqlitetutorial.net/
8. https://dbeaver.com/docs/dbeaver

Навчально-методичне видання

ОРГАНІЗАЦІЯ БАЗ ДАНИХ І ЗНАНЬ

Методичні вказівки
до виконання лабораторних робіт
для здобувачів спеціальності 122 «Комп'ютерні науки»
галузь знань 12 «Інформаційні технології».

Укладач Ярослав БАРДІН

Комп'ютерне верстання

Підписано до друку 22.02.2024 Формат 60 × 84 1/ 16
Ум. друк. арк. 1,16. Обл.-вид. арк. 1,25.
Електронний документ. Вид № 59/ІІІ-17.

Видавець і виготовлювач
Київський національний університет будівництва і архітектури

Повітрофлотський проспект, 31, Київ, Україна, 03037
Свідоцтво про внесення до Державного реєстру суб'єктів
видавничої справи ДК № 808 від 13.02.2002 р.