

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний аерокосмічний університет ім. М.Є. Жуковського  
“Харківський авіаційний інститут”

Кафедра комп’ютерних систем, мереж і кібербезпеки

Лабораторна робота № 5

з дисципліни “Обробка даних засобами Python”

Робота с базами данных SQL в Python

Виконав студент гр. 555iM Литвинов.О.А

\_\_\_\_\_  
(підпис, дата)

Перевірів к.т.н зав.кафедри каф. 301

\_\_\_\_\_  
(науковий ступінь, вчене звання, посада)

Дергачов.К.Ю

\_\_\_\_\_  
(підпис, дата)

(П.І.Б.)

2023

Мета: Дослідження методів роботи з базами даних у Python

## 18. Система документообігу

### Хід виконання

1. Вивчити методи роботи з базами даних SQLite у Python.

2. Відповідно до заданої предметної області створити базу даних що складається не менше ніж з (!) 4-х взаємопов'язаних таблиць.

(!) Кожна таблиця повинна містити щонайменше 4 полів та 15 записів.

3. Розробити та виконати 5 SQL-запитів до бази даних на вибірку даних різних типів (простий, перехресний, з використанням різних таблиць, визначення агрегованих характеристик тощо)

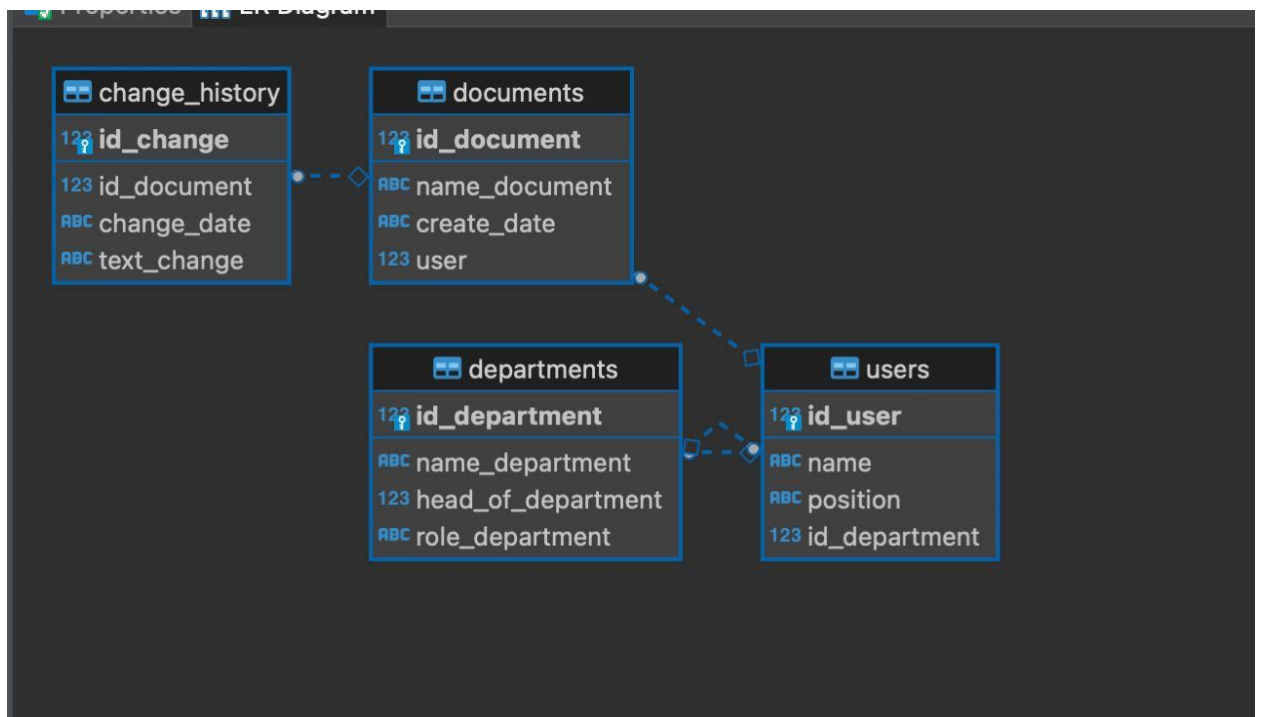


Рисунок 1 – Концептуальна модель бази даних

main.py

```
import database
import test_data_for_db
import test_request

# Запуск методів
database.create_db()
test_data_for_db.create_test_data()
test_request.test_requests()
```

database.py

```
import sqlite3

# Створення БД
def create_db():
```

```
conn = sqlite3.connect('database.db')

cursor = conn.cursor()
# На випадок якщо вже є така БД
cursor.execute('''DROP TABLE IF EXISTS documents;''')
cursor.execute('''DROP TABLE IF EXISTS users;''')
cursor.execute('''DROP TABLE IF EXISTS departments;''')
cursor.execute('''DROP TABLE IF EXISTS change_history;''')
# Таблиця документи
cursor.execute('''
CREATE TABLE IF NOT EXISTS documents (
    id_document INT PRIMARY KEY,
    name_document TEXT,
    create_date DATE,
    user INT,
    FOREIGN KEY (user) REFERENCES users (id_user)
);
''')
# Таблиця користувачі(робітники)
cursor.execute('''
CREATE TABLE IF NOT EXISTS users (
    id_user INT PRIMARY KEY,
    name TEXT,
    position TEXT,
    id_department INT,
    FOREIGN KEY (id_department) REFERENCES departments (id_department)
);
''')
# Таблиця відділи
cursor.execute('''
CREATE TABLE IF NOT EXISTS departments (
    id_department INT PRIMARY KEY,
    name_department TEXT,
    head_of_department INT,
    role_department TEXT,
    FOREIGN KEY (head_of_department) REFERENCES users (id_user)
);
''')
# Таблиця змін
cursor.execute('''
CREATE TABLE IF NOT EXISTS change_history (
    id_change INTEGER PRIMARY KEY,
    id_document INT,
    change_date DATE,
    text_change TEXT,
    FOREIGN KEY (id_document) REFERENCES documents (id_document)
);
''')

conn.commit()
conn.close()
```

## test\_data\_for\_db.py

```
import sqlite3
from datetime import datetime, date, timedelta
from random import randint

# Створення тестових даних для БД та додавання їх у БД
def create_test_data():
    conn = sqlite3.connect('database.db')
    cursor = conn.cursor()
    # Створення даних
    documents_data = [(i, f'document_{i}', datetime.now().date(),
                        randint(1, 15)) for i in range(1, 16)]
    users_data = [(i, f'user_{i}', f'position_{i + randint(1, 100)}',
                    randint(1, 15)) for i in range(1, 16)]
    departments_data = [(i, f'department №{i}', f'user_{16 - i}',
                         f'role_department_{16 - i}') for i in range(1, 16)]
    change_history_data = [(i, 16 - i, date(2023, 1, 1) + timedelta(days=i),
                            f'some change text №{i}') for i in range(1, 16)]
    # Массове додавання у БД
    cursor.executemany('INSERT INTO documents VALUES (?, ?, ?, ?)',
                       documents_data)
    cursor.executemany(f'INSERT INTO users VALUES (?, ?, ?, ?)', users_data)
    cursor.executemany(f'INSERT INTO departments VALUES (?, ?, ?, ?)',
                       departments_data)
    cursor.executemany(f'INSERT INTO change_history VALUES (?, ?, ?, ?)',
                       change_history_data)

    conn.commit()
```

## test\_request.py

```
import sqlite3

# Клас кольорів, для зміни кольору виводу у термінал
class Colors:
    DEFAULT = '\033[0m'
    CHANGED = '\033[95m'

# Тестові запити
def test_requests():
    conn = sqlite3.connect('database.db')

    cursor = conn.cursor()
    # Простий запит с однієї таблиці
    print(f'{Colors.CHANGED}Simple request:')
    simple_response = cursor.execute('''
        SELECT name_document, create_date FROM documents;
    ''').fetchall()
    print_response(simple_response)
    # Перекресний запит
    print(f'{Colors.CHANGED}Cross join request:')
```

```

cross_join = cursor.execute("""
    SELECT documents.name_document, users.name
    FROM documents INNER JOIN users
    ON documents.user = users.id_user;
""")
print_response(cross_join)
# Запит зі всіма таблицями
print(f'{Colors.CHANGED}Using different tables request:')
using_different_tables = cursor.execute('''
    SELECT documents.name_document,
    users.name, departments.name_department,
    change_history.change_date
    FROM documents
    JOIN users ON documents.user = users.id_user
    JOIN departments ON users.id_department = departments.id_department
    JOIN change_history ON change_history.id_document =
documents.id_document;
''')
print_response(using_different_tables)
# Запит с прорахунками
print(f'{Colors.CHANGED}aggregated characteristics request:')
aggregated_characteristics = cursor.execute('''
    SELECT documents.name_document, change_history.change_date
    FROM documents
    LEFT JOIN change_history ON documents.id_document =
change_history.id_document
    WHERE change_history.change_date > '2023-01-05';
''')
print_response(aggregated_characteristics)
conn.close()

# Вивід відповіді
def print_response(obj):
    for line in obj:
        print(f'{Colors.DEFAULT}', line)

```

```
Simple request:
('document_1', '2023-12-01')
('document_2', '2023-12-01')
('document_3', '2023-12-01')
('document_4', '2023-12-01')
('document_5', '2023-12-01')
('document_6', '2023-12-01')
('document_7', '2023-12-01')
('document_8', '2023-12-01')
('document_9', '2023-12-01')
('document_10', '2023-12-01')
('document_11', '2023-12-01')
('document_12', '2023-12-01')
('document_13', '2023-12-01')
('document_14', '2023-12-01')
('document_15', '2023-12-01')
Cross join request:
('document_1', 'user_1')
('document_2', 'user_14')
('document_3', 'user_10')
('document_4', 'user_11')
('document_5', 'user_15')
('document_6', 'user_15')
('document_7', 'user_8')
('document_8', 'user_9')
('document_9', 'user_1')
('document_10', 'user_15')
('document_11', 'user_1')
('document_12', 'user_3')
('document_13', 'user_3')
('document_14', 'user_10')
('document_15', 'user_2')
```

Рисунок 2 – Результати запитів до ДБ

```

Using different tables request:
('document_15', 'user_2', 'department №10', '2023-01-02')
('document_14', 'user_10', 'department №6', '2023-01-03')
('document_13', 'user_3', 'department №9', '2023-01-04')
('document_12', 'user_3', 'department №9', '2023-01-05')
('document_11', 'user_1', 'department №9', '2023-01-06')
('document_10', 'user_15', 'department №4', '2023-01-07')
('document_9', 'user_1', 'department №9', '2023-01-08')
('document_8', 'user_9', 'department №12', '2023-01-09')
('document_7', 'user_8', 'department №8', '2023-01-10')
('document_6', 'user_15', 'department №4', '2023-01-11')
('document_5', 'user_15', 'department №4', '2023-01-12')
('document_4', 'user_11', 'department №15', '2023-01-13')
('document_3', 'user_10', 'department №6', '2023-01-14')
('document_2', 'user_14', 'department №4', '2023-01-15')
('document_1', 'user_1', 'department №9', '2023-01-16')
aggregated characteristics request:
('document_11', '2023-01-06')
('document_10', '2023-01-07')
('document_9', '2023-01-08')
('document_8', '2023-01-09')
('document_7', '2023-01-10')
('document_6', '2023-01-11')
('document_5', '2023-01-12')
('document_4', '2023-01-13')
('document_3', '2023-01-14')
('document_2', '2023-01-15')
('document_1', '2023-01-16')

```

Рисунок 3 – Результати запитів до ДБ

Висновок:

Для виконання роботи було використано стандартну бібліотеку для python sqlite3. Було використано такі типи даних як INTEGER(цілі чисельні значення), та TEXT(текст). Для більшості дій достатньо команди cursor.execute, за допомогою якої робиться запит до сервера, також була використана cursor.executemany для масового запиту, такого як вставка масиву даних. Сама робота з бд для мене не є новою так як, я обширно використовую sqlite у мобільній розробці.

Є основні категорії запитів:

SELECT – для виборки, а точніше для перегляду інформації з БД;

CREATE TABLE – створення таблиці;

INSERT INTO – завантаження інформації до таблиці;

DELETE – видалити інформацію з таблиці;

DROP TABLE – видалити таблицю;