Солтисюк Дмитро, студент ТУ-12-21-ІПЗ

Лабораторна робота №4.1

Тема

База даних для складного імпорту.

Мета

Розробити схему бази даних, що дозволяє здійснювати імпорт даних про результати сесії студентів.

Виконання

1. **Створення бази даних**

Спершу, створимо файл бази даних SQLite. Задля цього використаємо програмну утиліту `usql` - програма, що дозволяє під'єднуватись до більшості популярних баз даних.

```
usql sqlite://suppliers.db
```

Результатом даної команди є створення нової бази даних або доступ до існуючої. Якщо база даних з вказаною назвою не існує, `usql` створить її автоматично, відкривши шлях для подальшого створення таблиць та управління даними.

2. **Створення таблиць**

```
CREATE TABLE IF NOT EXISTS Students (
  student_id INTEGER PRIMARY KEY AUTOINCREMENT,
  full_name VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE IF NOT EXISTS Subjects (
  subject_id INTEGER PRIMARY KEY AUTOINCREMENT,
  name VARCHAR(255) NOT NULL,
  exam_date DATE NOT NULL,
  CHECK (exam_date > '2000-01-01')
);

CREATE TABLE IF NOT EXISTS Exams (
  exam_id INTEGER PRIMARY KEY AUTOINCREMENT,
  student_id INTEGER NOT NULL,
  subject_id INTEGER NOT NULL,
  score INTEGER CHECK (score >= 0 AND score <= 100),
  FOREIGN KEY (student_id) REFERENCES Students(student_id),
  FOREIGN KEY (subject_id) REFERENCES Subjects(subject_id)
);
```

Тут ми створили схему БД для збереження даних в таблиці, створивши 3 таблиці: `Students`, `Subjects`, `Exams`. В схемі використали обмеження цілісності БД, використавши: `check`, `unique`. та `foreign key`.

3. **Додавання даних**

```
INSERT INTO Students (full_name) VALUES
('Тимко Б.А.'),
('Ющенко М.О.');
-- ... (додаткові студенти) ...

INSERT INTO Subjects (name, exam_date) VALUES
('Біологія', '2018-06-18');
-- ... (додаткові предмети) ...

INSERT INTO Exams (student_id, subject_id, score) VALUES
(1, 1, 57),
(2, 1, 80);
-- ... (додаткові результати) ...
```

4. **Створення огляду (view)**

```
CREATE VIEW IF NOT EXISTS SessionResultsView AS
SELECT
  s.full_name,
  subj.name,
  subj.exam_date,
  ex.score
FROM
  Exams ex
JOIN Students s ON ex.student_id = s.student_id
JOIN Subjects subj ON ex.subject_id = subj.subject_id;
```

На даному етапі ми використали операцію прямого поєднання таблиць `JOIN`, а також створили огляд (`VIEW`) - `SessionResultsView` для демонстрації результатів екзаменів з трьох таблиць.

5. **Запит для демонстрації результатів**

```
SELECT * FROM SessionResultsView;
```

```
Connected with driver sqlite3 (SQLite3 3.45.1)
Type "help" for help.

sq:suppliers.db=> CREATE TABLE IF NOT EXISTS Students (
sq:suppliers.db(>   student_id INTEGER PRIMARY KEY AUTOINCREMENT,
sq:suppliers.db(>   full_name VARCHAR(255) UNIQUE NOT NULL
sq:suppliers.db(> );
CREATE TABLE
sq:suppliers.db=>
sq:suppliers.db=> CREATE TABLE IF NOT EXISTS Subjects (
sq:suppliers.db(>   subject_id INTEGER PRIMARY KEY AUTOINCREMENT,
sq:suppliers.db(>   name VARCHAR(255) NOT NULL,
sq:suppliers.db(>   exam_date DATE NOT NULL,
sq:suppliers.db(>   CHECK (exam_date > '2000-01-01')
sq:suppliers.db(> );
CREATE TABLE
sq:suppliers.db=>
sq:suppliers.db=> CREATE TABLE IF NOT EXISTS Exams (
sq:suppliers.db(>   exam_id INTEGER PRIMARY KEY AUTOINCREMENT,
sq:suppliers.db(>   student_id INTEGER NOT NULL,
sq:suppliers.db(>   subject_id INTEGER NOT NULL,
sq:suppliers.db(>   score INTEGER CHECK (score >= 0 AND score <= 100),
sq:suppliers.db(>   FOREIGN KEY (student_id) REFERENCES Students(student_id),
sq:suppliers.db(>   FOREIGN KEY (subject_id) REFERENCES Subjects(subject_id)
sq:suppliers.db(> );
CREATE TABLE
sq:suppliers.db=> INSERT INTO Students (full_name) VALUES
sq:suppliers.db-> ('Тимко Б.А.'),
sq:suppliers.db-> ('Ющенко М.О.');
INSERT 2
sq:suppliers.db=> -- ... (додаткові студенти) ...
sq:suppliers.db->
sq:suppliers.db-> INSERT INTO Subjects (name, exam_date) VALUES
sq:suppliers.db-> ('Біологія', '2018-06-18');
INSERT 1
sq:suppliers.db=> -- ... (додаткові предмети) ...
sq:suppliers.db->
sq:suppliers.db-> INSERT INTO Exams (student_id, subject_id, score) VALUES
sq:suppliers.db-> (1, 1, 57),
sq:suppliers.db-> (2, 1, 80);
INSERT 2
sq:suppliers.db=> -- ... (додаткові результати) ...
sq:suppliers.db-> CREATE VIEW IF NOT EXISTS SessionResultsView AS
sq:suppliers.db-> SELECT
sq:suppliers.db->   s.full_name,
sq:suppliers.db->   subj.name,
sq:suppliers.db->   subj.exam_date,
sq:suppliers.db->   ex.score
sq:suppliers.db-> FROM
sq:suppliers.db->   Exams ex
sq:suppliers.db-> JOIN Students s ON ex.student_id = s.student_id
sq:suppliers.db-> JOIN Subjects subj ON ex.subject_id = subj.subject_id;
CREATE VIEW 2
sq:suppliers.db=> SELECT * FROM SessionResultsView;
  full_name  |   name   |      exam_date       | score
-------------+----------+----------------------+-------
 Тимко Б.А.  | Біологія | 2018-06-18T00:00:00Z |    57
 Ющенко М.О. | Біологія | 2018-06-18T00:00:00Z |    80
(2 rows)

sq:suppliers.db=> exit
```

Висновок

На даній лабораторній роботі було успішно створено структуру бази даних для збереження інформації про результати сесії студентів. Таблиці `Students`, `Subjects`, та `Exams` було налаштовано з дотриманням обмежень цілості даних, включаючи використання обмежень `CHECK`, `UNIQUE`, та `FOREIGN KEY`. Огляд `SessionResultsView` був створений для зручності відображення результатів. Всі необхідні дані були вставлені у таблиці, що дозволило виконати завдання лабораторної роботи, продемонструвавши роботу з операторами SQL та принципами розробки баз даних.