TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK

MODUL X

DATA STORAGE (BAGIAN I)

Disusun Oleh :

Zivana Afra Yulianto / 2211104039

SE-06-02

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

TUGAS PENDAHULUAN

SOAL

1. **Jelaskan secara singkat fungsi SQLite dalam pengembangan aplikasi mobile!**

   SQLite adalah sistem manajemen basis data relasional (RDBMS) berbasis file yang ringan dan sering digunakan dalam pengembangan aplikasi mobile. Fungsi utama SQLite dalam pengembangan aplikasi mobile meliputi:

   1. Penyimpanan Data Lokal: SQLite memungkinkan aplikasi menyimpan data secara lokal di perangkat pengguna tanpa memerlukan koneksi internet. Contoh: menyimpan data pengguna, preferensi aplikasi, atau cache.
   2. Pengelolaan Data Relasional: SQLite mendukung operasi basis data relasional seperti tabel, relasi antar data, dan kueri SQL (SELECT, INSERT, UPDATE, DELETE).
   3. Efisiensi dan Kinerja: SQLite ringan, cepat, dan tidak memerlukan server terpisah, sehingga cocok untuk aplikasi dengan sumber daya terbatas seperti aplikasi mobile.
   4. Mudah Digunakan: SQLite tersedia secara bawaan di banyak platform (termasuk Android dan iOS), sehingga pengembang dapat langsung menggunakannya tanpa instalasi tambahan.
   5. Portabilitas Data: Data yang disimpan dalam SQLite berupa file database tunggal yang dapat dengan mudah dipindahkan antar perangkat.

2. **Apa saja yang dimaksud dengan operasi CRUD? Berikan penjelasan singkat untuk masing-masing operasi!**

   Operasi CRUD adalah singkatan dari empat operasi dasar yang digunakan untuk mengelola data dalam basis data. Berikut penjelasan singkat untuk masing-masing operasi:

   1. Create (Membuat)
      Operasi ini digunakan untuk menambahkan data baru ke dalam basis data.
      Contoh: Menambahkan entri baru ke tabel seperti informasi pengguna atau produk.
      SQL: `INSERT INTO users (name, email) VALUES ('Zivana', 'zivana@example.com');`
   2. Read (Membaca)
      Operasi ini digunakan untuk mengambil data dari basis data, baik sebagian maupun seluruhnya.
      Contoh: Menampilkan daftar semua produk atau mencari data pengguna berdasarkan ID.
      SQL: `SELECT * FROM users WHERE id = 1;`
   3. Update (Memperbarui)
      Operasi ini digunakan untuk mengubah data yang sudah ada dalam basis data.
      Contoh: Memperbarui alamat email pengguna yang sudah terdaftar.
      SQL: `UPDATE users SET email = 'newemail@example.com' WHERE id = 1;`

4. Delete (Menghapus)
   Operasi ini digunakan untuk menghapus data dari basis data.
   Contoh: Menghapus akun pengguna yang tidak aktif.
   SQL: `DELETE FROM users WHERE id = 1;`

Operasi CRUD adalah inti dari pengelolaan data dalam sistem basis data, baik dalam aplikasi kecil maupun besar.

3. **Tuliskan kode SQL untuk membuat tabel bernama *users* dengan kolom berikut :**
   - **id (integer, primary key, auto increment)**
   - **name (text)**
   - **email (text)**
   - **createdAt (timestamp, default value adalah waktu sekarang)**

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    email TEXT NOT NULL,
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

4. **Sebutkan langkah-langkah utama untuk menggunakan plugin sqflite di dalam Flutter!**

Tambahkan dependensi `sqflite` dan `path` di file `pubspec.yaml`:

```
dependencies:
  flutter:
    sdk: flutter
  sqflite: ^2.2.0
  path: ^1.8.3
```

Di file Dart, import plugin yang diperlukan:

```
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart';
```

Gunakan fungsi openDatabase untuk membuat atau membuka database:

```
Future<Database> initDatabase() async {
  final dbPath = await getDatabasesPath();
  return openDatabase(
    join(dbPath, 'my_database.db'),
    onCreate: (db, version) {
      return db.execute(
        '''
        CREATE TABLE users (
          id INTEGER PRIMARY KEY AUTOINCREMENT,
          name TEXT,
          email TEXT,
          createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP
        )
        ''',
      );
    },
    version: 1,
  );
}
```

Lakukan operasi CRUD menggunakan metode sqflite:

Create :

```
Future<void> insertUser(Database db, Map<String, dynamic> user) async {
  await db.insert(
    'users',
    user,
    conflictAlgorithm: ConflictAlgorithm.replace,
  );
}
```

Read :

```
Future<List<Map<String, dynamic>>> fetchUsers(Database db) async {
  return await db.query('users');
}
```

Update :

```
Future<void> updateUser(Database db, int id, Map<String, dynamic> user) async {
  await db.update(
    'users',
    user,
    where: 'id = ?',
    whereArgs: [id],
  );
}
```

Delete :

```
Future<void> deleteUser(Database db, int id) async {
  await db.delete(
    'users',
    where: 'id = ?',
    whereArgs: [id],
  );
}
```

Panggil metode seperti `initDatabase` di `main()` atau bagian awal aplikasi untuk memastikan database siap digunakan. Contoh:

```
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  final db = await initDatabase();

  // Contoh: Tambah data pengguna
  await insertUser(db, {'name': 'Zivana', 'email': 'zivana@example.com'});

  // Contoh: Baca data pengguna
  final users = await fetchUsers(db);
  print(users);
}
```

5. **Lengkapi kode berikut untuk membaca semua data dari tabel *users* menggunakan sqflite.**

```
static Future<List<Map<String, dynamic>>> getUsers() async {
  final db = await SQLHelper.db();
  return db.query(__________);
}
```

```
static Future<List<Map<String, dynamic>>> getUsers() async {
  final db = await SQLHelper.db();
  return db.query('users');
}
```

Penjelasan:

- `db.query('users')`: Memanggil semua data dari tabel `users` tanpa filter (setara dengan `SELECT * FROM users` dalam SQL).
- Fungsi ini mengembalikan sebuah list berisi map dari data di tabel `users`, di mana setiap row data direpresentasikan sebagai map dengan pasangan key-value.

Dengan kode ini, fungsi `getUsers` akan membaca seluruh data dari tabel `users`.