

Bab 7

Praktikum

Database



Flutter

Dosen : Bambang Sugiarto, ST, MT

**Program Studi S1 Teknik Informatika
Fakultas Teknik**

Universitas Sangga Buana YPKP Bandung

*Dirangkum dari berbagai sumber referensi
(hanya untuk penggunaan internal/tidak untuk dipublikasikan)*

Tujuan Praktikum

- Mahasiswa mengetahui pemrograman database pada Flutter.
- Mahasiswa mampu membuat pemrograman database pada Flutter.
- Mahasiswa mampu membuat implementasi pemrograman database dengan membuat aplikasi mobile sederhana pada Flutter.



Pendahuluan

- Sesuai dengan materi yang disampaikan pada kuliah teori tentang database pada flutter dengan menggunakan paket *sqflite*, maka pada praktikum kita kali ini akan mencoba untuk mengaplikasikan database SQLite pada pemrograman Flutter.
- Praktikum kali ini adalah mencoba untuk memasang paket *sqflite* pada komputer kita, sehingga kita dapat menggunakan library *sqflite* pada pemrograman database SQLite kita.



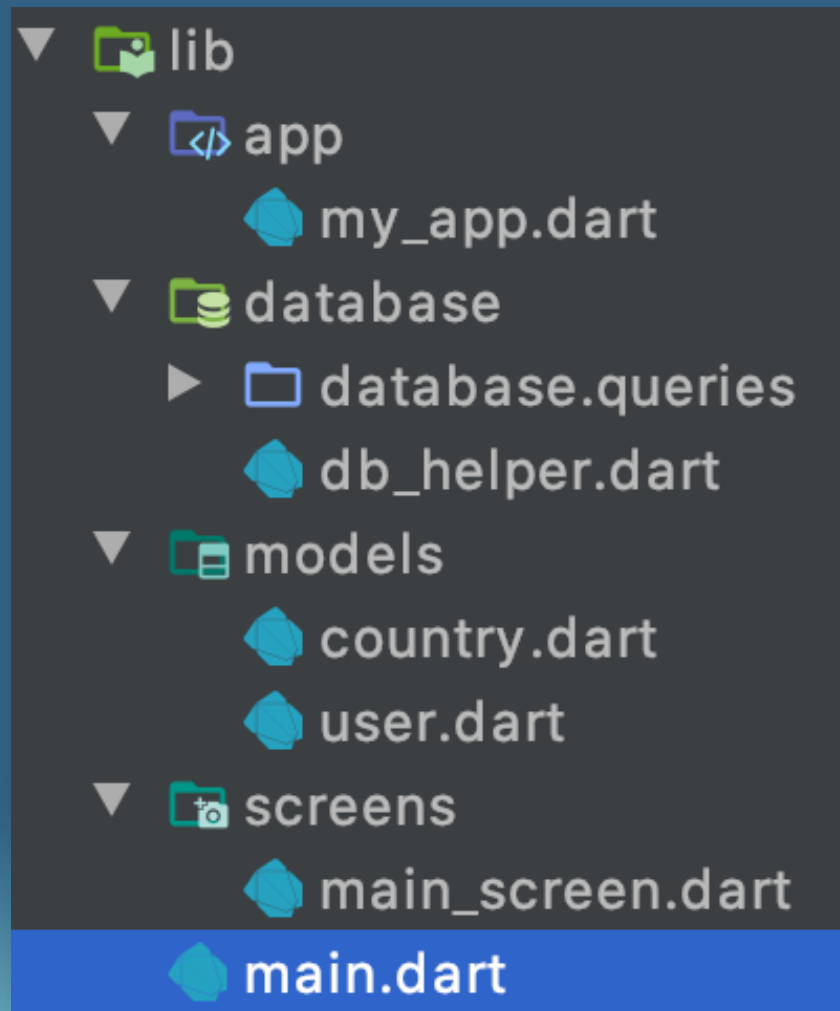
Praktikum Membuat Database SQLite dengan sqflite

<https://thengoding.com/2020/09/14/flutter-cara-mudah-menggunakan-sqflite/>

- Pada praktikum kali ini, kita akan mempraktekan pemrograman database SQLite pada flutter dengan menggunakan paket/library sqflite dengan langkah-langkah berikut :
 - Memasang Library sqflite :
 - ✓ <https://pub.dev/packages/sqflite/install>
 - ✓ Ikuti petunjuk cara petunjuk di halaman web tersebut.
 - Membuat folder
 - ✓ Pada folder *lib* tambahkan beberapa folder.
 - ✓ Hal ini agar kita mudah memaintanance aja sih, Jika tidak membuatnya pun juga tidak masalah.
 - ✓ Struktur foldernya bisa dilihat pada gambar



✓ Struktur folder :



- Membuat Kelas Query dari masing-masing tabel yang dibuat
 - ✓ Pada bagian ini, kita akan membuat sebuah kelas query dari JSON yang nantinya akan kita buat sebagai model juga berikut JSON nya.
 - ✓ JSON User :

```
{  
  "id":123,  
  "name":"congfandi",  
  "address":"Pamekasan, Madura"  
}
```

- ✓ JSON Country :

```
{  
  "id":123,  
  "name":"indonesia"  
}
```



- ✓ Pastikan model dan query untuk membuat tablenya sama agar nanti dapat di-casting dengan mudah menjadi sebuah model.
- ✓ class UserQuery :

```
class UserQuery {  
    static const String TABLE_NAME = "users";  
    static const String CREATE_TABLE =  
        " CREATE TABLE IF NOT EXISTS $TABLE_NAME ( id INTEGER PRIMARY KEY AUTOINCREMENT, n  
ame TEXT , address TEXT ) ";  
    static const String SELECT = "select * from $TABLE_NAME";  
}
```

✓ class User :



```
class User {
    String address;
    int id;
    String name;

    User({this.address, this.id, this.name});

    factory User.fromJson(Map<String, dynamic> json) {
        return User(
            address: json['address'],
            id: json['id'],
            name: json['name'],
        );
    }

    Map<String, dynamic> toJson() {
        final Map<String, dynamic> data = new Map<String, dynamic>();
        data['address'] = this.address;
        data['id'] = this.id;
        data['name'] = this.name;
        return data;
    }
}
```


✓ Class CountryQuery :

```
class CountryQuery {
    static const String TABLE_NAME = "countries";
    static const String CREATE_TABLE =
        " CREATE TABLE IF NOT EXISTS $TABLE_NAME ( id INTEGER PRIMARY KEY AUTOINCREMENT, NAME TEXT ) ";
    static const String SELECT = "select * from $TABLE_NAME";
}
```

✓ Class Country

```
class Country {
    int id;
    String name;

    Country({this.id, this.name});

    factory Country.fromJson(Map<String, dynamic> json) {
        return Country(
            id: json['id'],
            name: json['NAME'],
        );
    }

    Map<String, dynamic> toJson() {
        final Map<String, dynamic> data = new Map<String, dynamic>();
        data['id'] = this.id;
        data['NAME'] = this.name;
        return data;
    }
}
```

➤ Membuat Kelas DBHelper

```
import 'package:flutter_atomic_design/database/queries/country_query.dart';
import 'package:flutter_atomic_design/database/queries/user_query.dart';
import 'package:sqflite/sqflite.dart' as sqflite;
import 'package:sqflite/sqflite_api.dart';
import 'package:path/path.dart' as path;

class DBHelper {
  //membuat method singleton
  static DBHelper _dbHelper = DBHelper._singleton();

  factory DBHelper() {
    return _dbHelper;
  }

  DBHelper._singleton();

  //baris terakhir singleton

  final tables = [
    UserQuery.CREATE_TABLE,
    CountryQuery.CREATE_TABLE
  ]; // membuat daftar table yang akan dibuat
```

```

Future<Database> openDB() async {
  final dbPath = await sqlite.getDatabasesPath();
  return sqlite.openDatabase(path.join(dbPath, 'thengoding.db'),
    onCreate: (db, version) {
      tables.forEach((table) async {
        await db.execute(table).then((value) {
          print("berashil ");
        }).catchError((err) {
          print("errornya ${err.toString()}");
        });
      });
      print('Table Created');
    }, version: 1);
}

insert(String table, Map<String, Object> data) {
  openDB().then((db) {
    db.insert(table, data, conflictAlgorithm: ConflictAlgorithm.replace);
  }).catchError((err) {
    print("error $err");
  });
}

Future<List> getData(String tableName) async {
  final db = await openDB();
  var result = await db.query(tableName);
  return result.toList();
}
}

```

- Testing DBHelper dengan insert
 - ✓ Untuk melakukan testing database dapat dipanggil di main class seperti kode dibawah ini :

```
final DBHelper _helper = new DBHelper();  
@Override  
void initState() {  
    super.initState();  
    _helper.insert(CountryQuery.TABLE_NAME, {"NAME":"Singapura"});  
}
```

- ✓ Apabila ada terlihat tulisan berhasil seperti gambar dibawah artinya table sudah berhasil dibuat dan data sudah berhasil di insert ke lokal db kita



```

28 //baris terakhir singleton
29
30 final tables = [
31   UserQuery.CREATE_TABLE,
32   CountryQuery.CREATE_TABLE
33 ]; // membuat daftar table yang akan dibuat
34
35 Future<Database> openDB() async {
36   final dbPath = await sqlite.getDatabasesPath();
37   return sqlite.openDatabase(path.join(dbPath, 'thergooding.db'),
38     onCreate: (db, version) {
39     tables.forEach((table) async {
40       await db.execute(table).then((value) {
41         print("berashil ");
42       }).catchError((err) {
43         print("errornya $err.toString());
44       });
45     });
46     print("Table Created");
47   }, version: 1);
48 }
49
50 insert(String table, Map<String, Object> data) {
51   openDB().then((db) {
52     db.insert(table, data, conflictAlgorithm: ConflictAlgorithm.replace);
53   }).catchError((err) {
54     print("error $err");
55   });
56 }
57
58 Future<List> getData(String tableName) async {

```

Run: main.dart x

```


Console
+ Launching lib/main.dart on iPhone 11 Pro Max in debug mode...
+ Running Xcode build...
+ Xcode build done. 34.2s
+ Waiting for iPhone 11 Pro Max to report its views...
+ Debug service listening on ws://127.0.0.1:52113/w7d8fjsdKaI/ws
+ Syncing files to device iPhone 11 Pro Max...
+ flutter: Table Created
+ flutter: berashil
+ flutter: berashil

```

- Print data yang sudah tersimpan
 - ✓ Untuk melihat hasilnya, silahkan ganti kode untuk insert menjadi seperti dibawah ini :

```
final DBHelper _helper = new DBHelper();

@Override
void initState() {
    super.initState();
    // _helper.openDB();
    // _helper.insert(CountryQuery.TABLE_NAME, {"NAME":"Singapura"});
    _helper.getData(CountryQuery.TABLE_NAME).then((value) {
        value.forEach((element) {
            Country country = Country.fromJson(element);
            print(country.toJson());
        });
    });
}
```

- 
- ✓ Dan jika hasilnya seperti dibawah ini, maka tandanya data kita sudah berhasil di insert ke lokal DB kita.

The screenshot shows an IDE with a Flutter project. The left sidebar displays the Project Files tree, including folders like .dart_tool, .idea, android, build, ios, lib, and test. The main editor shows the code for main_screen.dart, which imports a package and defines a MainScreen class and its state. The code uses a DbHelper class to interact with a database. The bottom console shows the output of a hot restart, indicating that the application was successfully restarted on an iPhone 11 Pro Max and displays three Flutter widgets.

```
14 import 'package:flutter_atomic_design/models/country.dart';
15
16 class MainScreen extends StatefulWidget {
17   @override
18   _MainScreenState createState() => _MainScreenState();
19 }
20
21 class _MainScreenState extends State<MainScreen> {
22   final DbHelper _helper = new DbHelper();
23
24   @override
25   void initState() {
26     super.initState();
27     // _helper.openDB();
28     // _helper.insert(CountryQuery.TABLE_NAME, {"NAME":"Singapura"});
29     _helper.getData(CountryQuery.TABLE_NAME).then((value) {
30       value.forEach((element) {
31         Country country = Country.fromJson(element);
32         print(country.toJson());
33       });
34     });
35   }
36
37   @override
38   Widget build(BuildContext context) {
39     return Scaffold(
40       appBar: AppBar(
41         title: Text("SQLITE Demo"),
42       ), // AppBar
43     ); // Scaffold
44   }
```

Run: main.dart x

Console

Performing hot restart...
Syncing files to device iPhone 11 Pro Max...
Restarted application in 1,527ms.
flutter: {id: 1, NAME: Indonesia}
flutter: {id: 2, NAME: Singapura}
flutter: {id: 3, NAME: Singapura}