Storing Data in a Database on Flutter to Solve Practice Problems (F_CMP2)

24/01/2024

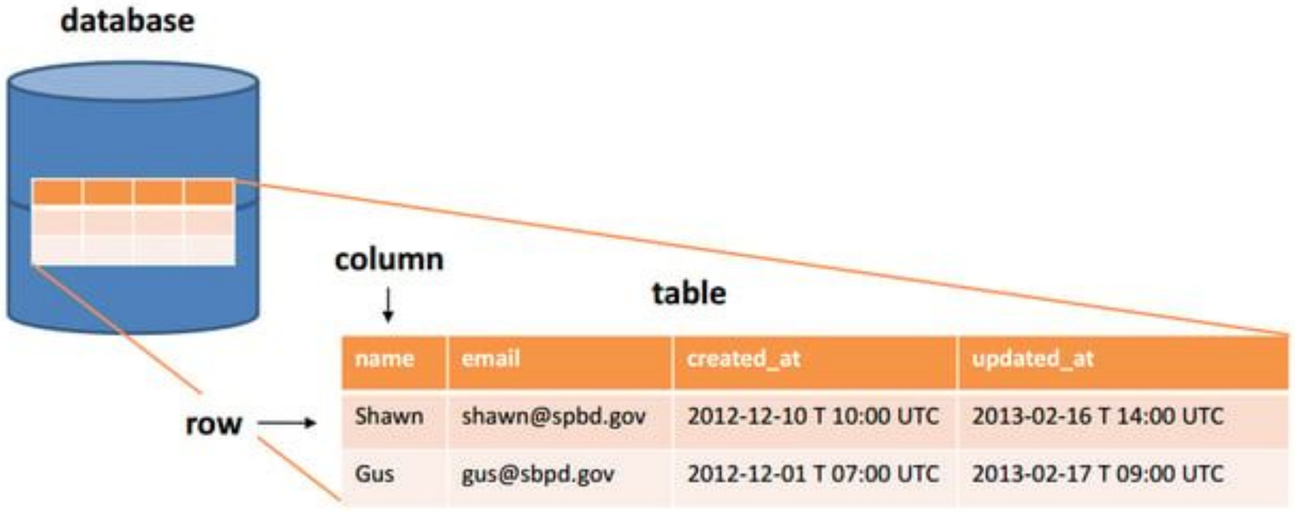
Contents

Overview about database

- Database definition : A database is a collection of data that is connected to each other, making it easy to see the relationship between information.
- Purpose of database : To ensure the data is stored correctly and easily searchable.
- Database components : Database components consist of data, tables, relationships, and database management system (DBMS).
- Types of databases : There are different types of databases, including relational databases, NoSQL databases, and object-oriented databases.

What is SQL?

- SQL is a programming language used to manage data in a relational database.
- A relational database is a database that uses tables to store data.



name	email	created_at	updated_at
Shawn	shawn@spbd.gov	2012-12-10 T 10:00 UTC	2013-02-16 T 14:00 UTC
Gus	gus@sbpd.gov	2012-12-01 T 07:00 UTC	2013-02-17 T 09:00 UTC

What is SQLite?

- SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database.
- SQLite is an in-memory database, which means that data is stored in the device's main memory.
- SQLite makes it ideal for applications that require quick access to data, such as mobile applications.

SQLite

Self-contained
Minimal support from external libraries

Serverless
Reads and writes directly from the database files on disk

Zero-configuration
No installation, no setup

Transactional
All changes in a transaction occur completely or not at all

Overview basic SQLite statements

- **CREATE TABLE :** Creates a new table.
- **INSERT INTO :** Adds new data to a table.
- **SELECT :** Retrieves data from a table.
- **UPDATE :** Changes the data in the table.
- **DELETE :** Deletes data from a table.

SQLite also provides some additional features, such as :

- **TRIGGER :** Events that occur when certain operations are performed on a table.
- **VIEW :** The virtual table represented by the query.
- **INDEX :** A data structure that improves query performance.

About Sqflite

- A Flutter plugin to access and manage SQLite databases in Flutter apps, both on Android and iOS devices.
- Sqflite allows you to perform various database operations, such as creating tables, adding data, updating data, and deleting data.
- Include it’s dependency in pubspec.yaml
- Sqflite consists of two main components, which are :
    - Database: A database is a collection of data stored in a relational format.
    - Tables: Tables are data structures used to store data in the database.

Advantages and disadvantages of Sqflite

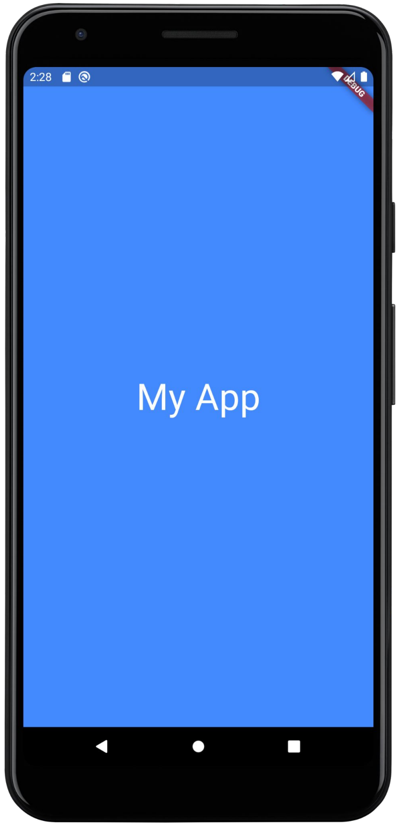
Advantages	Disadvantages
● Lightweight and portable ● Simple and easy to use ● Efficient and scalable ● Secure	● Can be complex ● Only supports SQLite

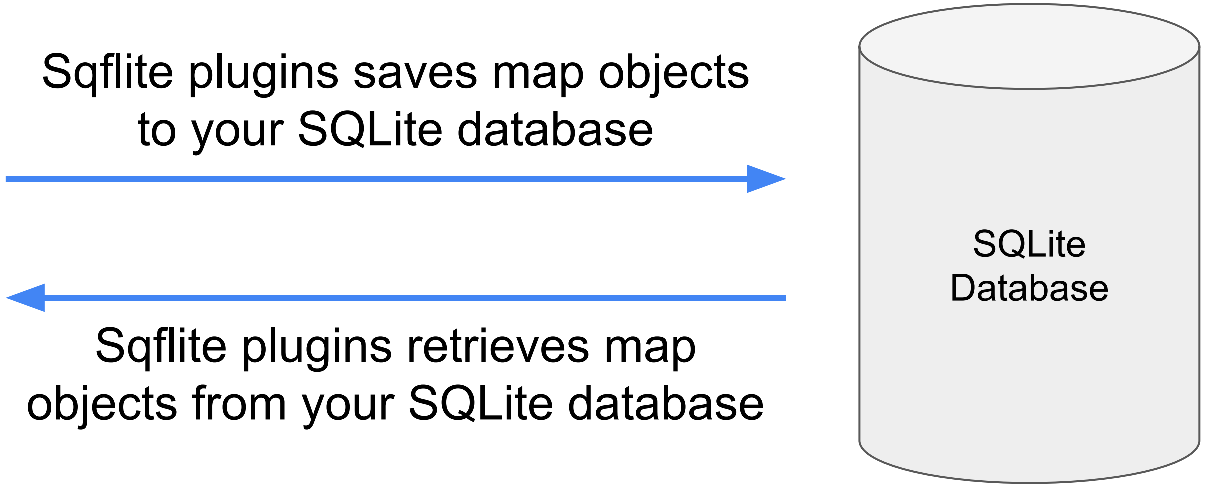
Data type requirements of the Map object

- The Map object is a simple key/value pair.
- To declare Map object have to enclose the key-value pairs within a pair of curly brackets **"{ }"**.
- Example :

```
void main() {
  var detail = {'Usrname':'enzy','Password':'pass@456'};
  detail['key'] = 'value';
  String? name = detail['Usrname'];
  print(name); //output: enzy
  print(detail);
  //output: {Usrname: enzy, Password: pass@456, key: value}
}
```

How Sqflite works




Sqflite plugins saves map objects to your SQLite database

Sqflite plugins retrieves map objects from your SQLite database

- In short, before saving data to database, you need to convert data into MAP object.
- In short, when you retrieve data from database, you get a MAP object. You need to convert it to simple object before using it.

Installation of Sqflite in Flutter

Adding Package : Add the sqflite package in your pubspec.yaml file under dependencies :

```
dependencies:
  sqflite: any
```

Installing Package : Open the terminal in your IDE or command prompt and navigate to your project directory, where pubspec.yaml is located. Run the command flutter pub get to retrieve the packages.

```
$ flutter pub get
```

Flutter SQLite CRUD operations : Setting up

1. **Importing Package :** Import the sqflite package inside your Flutter Dart file.

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

Flutter SQLite CRUD operations : Setting up

2. **Opening Database :** Use the openDatabase method to open a connection to the database.

```
void createTable() async {
  final db = await openDatabase('notedb');
}
```

- After calling openDatabase() in Sqflite, it will immediately create a folder on device :
    - **Android :** in the directory data/data/<your_package_name>/notedb
    - **iOS :** in the directory /var/mobile/Containers/Data/Application/<ID>/Documents/notedb

Flutter SQLite CRUD operations : Setting up

3. **Creating Table :** To create a table in your SQLite database, use the execute method after opening the database.

```
void createTable() async{
  final db = await openDatabase(
    join(await getDatabasesPath(), 'notedb'),
    onCreate: (db, version) {
      return db.execute('CREATE TABLE notes(id INTEGER
PRIMARY KEY, note TEXT)');
    },
    version: 1,
  );
}
```

Flutter SQLite CRUD operations : Writing queries

Using Raw SQL

Using Helper Functions and just pass parameter

Flutter SQLite CRUD operations : Writing queries

Difference between using raw SQL and using helper function :

```
db.rawQuery("SELECT * FROM notedb")          //Writing raw SQL
db.query("notes");                          //Helper function

db.rawInsert("YOUR SQL STATEMENT");          //Writing raw SQL
db.insert(param1, param2, param3);           //Helper function

db.rawDelete("YOUR SQL STATEMENT");          //Writing raw SQL
db.delete(param1, param2, param3);           //Helper function

db.rawUpdate("YOUR SQL STATEMENT");          //Writing raw SQL
db.update(param1, param2, param3);           //Helper function
```

Flutter SQLite CRUD operations : Writing queries

Each parameter in helper function :

```
db.query(“notes”);

db.insert(“notes”, Map data, sqflite conflictAlgorithm);

db.delete(“notes”, where: “id=?”, whereArgs: [1]);

db.update(“notes”, Map data require, where: “id=?”,
whereArgs: [1]);
```