Introduction to the app database

A database is the organized collection of data that can be stored and accessed electronically from any device. The stored data can be accessed, modified, controlled, and organized, also it remains available to use unless it is intentionally erased.

The modern apps demand real-time data delivery every time it is requested by the end-users. Also, it's critical to maintain data integrity and sovereignty by updating data frequently with every change.

In general, databases are of two types, Relational and Non-Relational databases.

Relational databases:

Relational databases are the sets of data that have a relationship with each other. The data sets have data stored in the form of tables with rows and columns. Each table holds information about an object.

The column in the table holds the data values with a particular type, one value for each row of the database. It may contain the text, number, or even pointers to the file in the OS. The table's row, also called a tuple, indicates a single, implicitly structured data item in the table.

SQL and SQLite are examples of relational databases.

Non-Relational databases

It is different from the relational database, in the Non-relational database, the data is stored in the document with no relationship among them. NoSQL databases have the potential to handle a huge amount of unstructured data efficiently.

The well-known examples of NoSQL databases are Google Firebase, MongoDB, Objectbox, Hive, and Shared Preference.

Popular Flutter database options for relational and Non-relational databases

Relational database options:

sqflite:

SQflite is an implementation of SQLite for Flutter. It gives you complete control over your app database, queries, and relationships in your hands.

Pros:

- An efficient, highly reliable, embedded SQLite database for your app that offers complete control over the database.
- Supports transactions and batches.
- Automatic version management.
- Easily read third-party apps that can open SQLite databases.
- Helpers for insert/query/update and delete queries.
- Executes DB operations in the background thread on iOS and Android.

Cons:

- Has no web support.
- Writing queries can take enough time.
- Returned data isn't strongly typed.
- Data migration can be difficult.

When to use it?

Use SQflite if you need a relational database with full control over the database queries. It is the best option only if you are comfortable writing your own queries and code.

Hive:

Hive is a super-fast, lightweight, key-value database written in pure Dart. It is completely native to Dart. It lets you store data as a HiveObject, which allows relation between objects.

Pros:

- Provides SQL-like syntax to provide rapid development.
- Best for big data processing.
- Supports user-defined functions.
- Often used for data analysis where real-time requirements are not high.

Cons:

- Not a good fit for a complex data model.
- Has limitations of the MapReduce data processing flow.
- The efficiency of Hive is relatively low.

When to use it?

If you are going to use a simple database and don't want real-time data synchronization. If you want something that works anywhere, Hive integrates easily with Dart. It is best suited for Flutter projects that require data storage on the devices and then access it anywhere.

Now let's back to SqLite database to use it as example database and we will discuss how to use it with flutter apps.

How to use SQLite in Flutter:

There is a Flutter package called Sqflite which is used to connect your flutter app to your SQLite database.

So What is Sqflite?... SQFlite is a plugin in flutter which is used to store data locally it has a lot of operations like Create , Read , Update and Delete which is Also Called CRUD Operations .

What We Will Do:

- Add SQFlite :After Creating a Project inside your project go to the **pubspec. yaml** and add the sqfllite dependency under the dependencies:

```
dependencies:
  ...
  sqflite:
```

- Create New Dart File for Database modules.
- Create new class for database module, this class contains:
    - Database Initializing Method.
    - CRUD methods.

All of above is mentioned in individual file with complete code.

So the code contains of the following methods:

- SQLite package provides us **openDatabase** a method to open the database at a specified path. the **version** property helps us to assign a version to the database. **onCreate** the property takes a **Database** and **version** . Inside the **onCreate** property, we can build our database table using execute method.

- To insert the data or model inside the database we use the **insert** method. It takes a sql query which is contains the fields names and the values we need to insert.
- To fetch the data from the database we use the **query** method. this method takes the name of the table that is needed to be fetched.
- To update the table query we use **update** the method. It takes the table name and id of the query that we want to update. Similarly, we can use the **delete** method to delete the query.