# Module - 8

## 1. Difference Between Local Storage Options (shared_preferences, SQLite, Hive)

### shared_preferences:

- **What it is**: A simple key-value store, ideal for saving small pieces of data like user settings, flags, or preferences (e.g., theme mode, language selection).
- **Use case**: Store simple, primitive data types (strings, booleans, integers).
- **Pros**: Easy to use, lightweight, quick read and write operations.
- **Cons**: Not suitable for complex or large datasets, not relational.

### SQLite:

- **What it is**: A full relational database that supports SQL queries and structured data with tables, rows, and columns.
- **Use case**: Storing structured, complex, or large datasets that require relational data management.
- **Pros**: Supports complex queries, indexing, and transactions.
- **Cons**: More overhead in setup and maintenance, requires knowledge of SQL.

### Hive:

- **What it is**: A lightweight, NoSQL database for Flutter. It stores data in a simple and efficient key-value format, but it’s optimized for performance and scalability.
- **Use case**: Store non-relational, large datasets with fast read/write operations. Often used for complex data, but with a simpler API than SQLite.
- **Pros**: Very fast, works offline, does not require a database server, easy to integrate.
- **Cons**: No support for relational data, limited querying capabilities compared to SQLite.

## 2. CRUD Operations and How They Are Implemented in SQLite or Hive

### CRUD:

CRUD stands for **Create, Read, Update, Delete**, which are the basic operations for interacting with data in a database or storage system.

### SQLite CRUD Operations:

- **Create**: Insert new data into a table using SQL INSERT statements.
  - Example: INSERT INTO users (id, name, age) VALUES (1, 'John', 25);
- **Read**: Fetch data from a table using SQL SELECT statements.
  - Example: SELECT * FROM users WHERE id = 1;

- **Update**: Modify existing data in a table using SQL UPDATE statements.
    - Example: UPDATE users SET name = 'Jane' WHERE id = 1;
- **Delete**: Remove data from a table using SQL DELETE statements.
    - Example: DELETE FROM users WHERE id = 1;

### Hive CRUD Operations:

- **Create**: Add data to a box (Hive's storage unit) using the put() method.
    - Example: box.put('user1', User('John', 25));
- **Read**: Retrieve data from a box using the get() method.
    - Example: var user = box.get('user1');
- **Update**: To update data, overwrite it using the put() method.
    - Example: box.put('user1', User('Jane', 26));
- **Delete**: Remove data from a box using the delete() method.
    - Example: box.delete('user1');

## 3. Advantages and Use Cases for shared_preferences

**Advantages**:

- **Simple to Use**: It’s designed to be easy to set up and use, making it perfect for beginners.
- **Lightweight**: It’s great for small, simple data storage like user preferences, flags, or simple settings.
- **Fast Access**: Read and write operations are quick, making it efficient for small data tasks.
- **Persistent**: Data stored remains persistent across app restarts, so user preferences or small settings are saved.

**Use Cases**:

- **User Preferences**: Storing app settings like language, theme (dark mode), or notification preferences.
- **Flags and State**: Storing simple flags, like whether the user has seen a tutorial or if they are logged in.
- **Simple Caching**: Caching small amounts of data, like tokens or session details, to improve app performance.