Database Comparisons

1. SQLite

- **Type**: Relational Database.
- **Key Features**:
  - Serverless and lightweight.
  - SQL-based queries for structured data.
  - Embedded within the app.
- **Best For**:
  - Applications requiring structured data storage and complex queries.
- **Offline Support**: Fully offline.
- **Integration**:
  - Can be accessed in Flutter via the **Sqflite** plugin.
- **Performance**:
  - Good for read-heavy apps but may lag slightly with large-scale write operations.
- **Use Case**:
  - Apps with relationships (e.g., history of diseases with related details).

---

2. Sqflite

- **Type**: Flutter plugin for SQLite.
- **Key Features**:
  - Provides an interface for SQLite in Dart/Flutter.
  - Offers SQL queries and transactions.
  - Extends SQLite's functionality for Flutter apps.
- **Best For**:
  - Flutter developers who need a relational database with raw SQL capabilities.
- **Offline Support**: Fully offline.
- **Integration**:
  - Works directly with SQLite.
- **Performance**:
  - Depends on SQLite; suitable for small to medium-scale apps.
- **Use Case**:
  - Apps needing local relational data storage and full SQL query control.

---

3. Hive

- **Type**: Key-Value NoSQL Database.

- **Key Features**:
    - Lightweight and very fast.
    - Pure Dart implementation with no native dependencies.
    - Does not use SQL; instead, stores data in key-value pairs.
- **Best For**:
    - Apps requiring minimal setup and high-speed data access.
    - Offline-first apps with simple data storage needs.
- **Offline Support**: Fully offline.
- **Integration**:
    - Works seamlessly in Flutter with simple setup.
- **Performance**:
    - Extremely fast for read/write operations.
- **Use Case**:
    - Storing user preferences, configurations, or disease history in a simple format.

---

4. Moor (Drift)

- **Type**: Relational Database Abstraction for SQLite.
- **Key Features**:
    - Wraps SQLite with a higher-level API.
    - Provides strong typing and reactive streams.
    - Supports queries using Dart code instead of raw SQL.
- **Best For**:
    - Developers who want SQLite’s power with a modern Dart-based API.
- **Offline Support**: Fully offline.
- **Integration**:
    - Built directly on SQLite.
- **Performance**:
    - Comparable to SQLite with added developer convenience.
- **Use Case**:
    - Apps with complex data relationships, requiring reactive programming and clean code.

---

5. Firestore (Firebase)

- **Type**: Cloud-based NoSQL Database.
- **Key Features**:
    - Real-time synchronization.
    - Supports structured and unstructured data.
    - Designed for online-first apps with offline caching.
- **Best For**:

    - Apps requiring cloud data syncing across devices.
- **Offline Support**:
    - Limited; data syncs only when connected.
- **Integration**:
    - Tight integration with Firebase Authentication and Hosting.
- **Performance**:
    - Optimized for real-time operations; slower than local solutions for offline use.
- **Use Case**:
    - Collaborative apps or those needing cross-device access.
- ss.

Comparison Table

Feature	**SQLite**	**Sqflite**	**Hive**	**Moor (Drift)**	**Firestore**
Type	Relational	Relational (Plugin)	Key-Value (NoSQL)	Relational (Wrapper)	NoSQL (Cloud)
Offline Support	Fully Offline	Fully Offline	Fully Offline	Fully Offline	Limited (Caching)
Ease of Use	Moderate	Moderate	Easy	Easy	Easy (Cloud setup)
Scalability	Moderate	Moderate	High	High	Very High
Performance	Good	Good	Very Fast	Good	Real-time (Cloud)
Complex Queries	Yes	Yes	No	Yes	No
Setup Complexity	Low	Low	Very Low	Low	Moderate
Best For	Structured Data	Structured Data	Simple Data Storage	Structured Data	Cloud Sync

---

Recommendation

- **Offline-Only with Structured Data**: Use **Sqflite** or **Moor** (Drift).
- **Offline-Only with Simple Data**: Use **Hive** for high performance and simplicity.
- **Cloud Sync & Scalability**: Use **Firestore** if real-time sync and scalability are crucial.
- **Hybrid Approach**:
  - Use **Sqflite** or **Hive** for local storage.
  - Add **Firestore** for optional cloud backup or cross-device access.