

# DATA STORAGE IN ANDROID Using SQLite Database

## 1. Introduction to SQLite in Android

SQLite is a **lightweight, open-source, file-based relational database** built directly into Android. It allows applications to **store structured data locally** without requiring any server or internet connection.

### Key Features

- **Zero-configuration** → No installation or server required
- **Stores data in a single .db file**
- **Fast & Reliable** for local storage
- **Supports SQL** (Structured Query Language)
- Used for apps that need to store:
  - Notes
  - Contacts
  - Tasks
  - Offline data
  - Records, logs, etc.

## 2. Why Use SQLite Instead of SharedPreferences or Internal Storage?

Storage Type	Usage
SharedPreferences	Small key-value data (login name, settings)
Internal Storage	Files, images, documents
SQLite Database	Structured data, multiple records, search, sorting

SQLite is best when:

- You have **multiple rows and columns**
- You need **insert/update/delete**
- You need **search or filtering**
- You want **persistent offline storage**

## 3. How SQLite Works in Android

SQLite stores data in **tables**, similar to MySQL or Oracle.

### ✓ Table Example

id	note
1	Buy grocery
2	Pay bills
3	Call friend

Each row represents a record.

You interact with the database using:

- **SQL queries**
- **SQLiteDatabase** class

- **SQLiteOpenHelper** class

#### 4. Important SQL Commands (Crash Course)

##### Create Table

```
CREATE TABLE notes(id INTEGER PRIMARY KEY AUTOINCREMENT, note TEXT);
```

##### Insert Data

```
INSERT INTO notes(note) VALUES ('Buy Grocery');
```

##### Read Data

```
SELECT * FROM notes;
```

##### Update Data

```
UPDATE notes SET note='Buy bread' WHERE id=1;
```

##### Delete Data

```
DELETE FROM notes WHERE id=1;
```

#### 5. Role of SQLiteOpenHelper in Android

Android provides the **SQLiteOpenHelper** class to simplify working with SQLite.

##### SQLiteOpenHelper Helps With:

- ✓ Creating the database
- ✓ Upgrading the database version
- ✓ Opening a readable/writable database
- ✓ Managing SQL statements

#### 6. SQLiteOpenHelper Class — Details

To use SQLite, you must create a class extending:

```
class DBHelper(context: Context)
    : SQLiteOpenHelper(context, "notes.db", null, 1)
```

**You MUST override two methods:**

##### onCreate()

Runs **only once** → When the app installs and DB is created.

Used to create tables:

```
override fun onCreate(db: SQLiteDatabase) {
    db.execSQL("CREATE TABLE notes(id INTEGER PRIMARY KEY AUTOINCREMENT, note TEXT)")
}
```

##### onUpgrade()

Runs when database **version increases**, e.g., from 1 → 2.

Used when you want to:

- Add new columns
- Modify table structure

**Example:**

```
override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
    db.execSQL("DROP TABLE IF EXISTS notes")
    onCreate(db)
}
```

## 7. SQLiteDatabase — Important Functions

Function	Purpose
writableDatabase	Insert, update, delete
readableDatabase	Fetch / read data
execSQL()	Run SQL without return values
rawQuery()	Run SELECT statements

## 8. CRUD Operations in Android SQLite

CRUD stands for:

✓ **C – Create (Insert)**

✓ **R – Read (Select)**

✓ **U – Update**

✓ **D – Delete**

These operations are performed using:

- ContentValues
- SQLiteDatabase functions
- SELECT queries with Cursor

## 9. Important Android Classes Used

### ◆ ContentValues

Used to store data in key-value pairs for insert & update.

### ◆ Cursor

Used to read SELECT results row-by-row.

### ◆ SQLiteDatabase

Provides methods:

- insert()
- update()
- delete()
- rawQuery()
- execSQL()

## 10. Real-Life Apps Using SQLite

✓ WhatsApp (stores chat, messages locally)

✓ Notes apps

✓ Contact apps

✓ Banking apps (cached transactions)

✓ Task managers

✓ Expense trackers

SQLite is perfect for **offline-first apps**.

## 11. Advantages of SQLite

- Fast & efficient
- No server setup

- Built into Android
- Supports complex queries (JOIN, LIKE, ORDER BY)
- Data persists even if app restarts
- Good for medium-sized data

## 12. Limitations of SQLite

- Not suitable for large-scale syncing
- No concurrent multi-user access
- Manual SQL writing (unless using Room)
- Data loss possible if DB file corrupts

## 13. When Should You Use SQLite?

Use SQLite when your app handles:

- Multiple related data items
- Records need to be updated or deleted
- Search, filter, or sorting
- Offline data support

### Examples:

- Grocery list app
- Student database
- Notes app
- CRUD teaching apps
- Inventory systems

## 14. Note

Topic	Summary
SQLite	Local relational DB
SQLiteOpenHelper	Manages DB creation & upgrades
Tables	Rows & columns
CRUD	Insert/Read/Update/Delete
Cursor	Reads data
SQL	SQLite uses SQL commands
Uses	Notes, contacts, tasks