1. SQLite

* QLite là một hệ quản trị cơ sở dữ liệu nhưng không theo mô hình client-server hư các hệ cơ sở lớn như MySQL,MSSQL,Oracle.v.v.v. được tích hợp sẵn tầng thư viện của cấu trúc hệ điều hành Android. nên ta không cần phải xây dựng bất kỳ connection nào .  SQLite sẽ hỗ trợ chúng ta xây dựng dữ liệu có cấu trúc, dễ dàng quản lý và thao tác với kiểu dữ liệu rõ dàng, có quan hệ v.v.v. Mỗi ứng dụng sẽ xây dựng được một database riêng và toàn quyền thao tác với nó.
* QLite sử dụng hầu như toàn bộ cú pháp câu truy vấn động
* SQLite có các ưu điểm sau:
* Tin cậy: các hoạt động transaction (chuyển giao) nội trong cơ sở dữ liệu được thực hiện trọn vẹn, không gây lỗi khi xảy ra sự cố phần cứng
* Tuân theo chuẩn SQL92 (chỉ có một vài đặc điểm không hỗ trợ)
* Không cần cài đặt cấu hình
* Kích thước chương trình gọn nhẹ, với cấu hình đầy đủ chỉ không đầy 300 kB
* Thực hiện các thao tác đơn giản nhanh hơn các hệ thống cơ sở dữ liệu khách/chủ khác
* Không cần phần mềm phụ trợ
* Phần mềm tự do với mã nguồn mở, được chú thích rõ ràng
* Đầu tiên, để thao tác với SQLite, ta phải dùng 2 đối tượng
* **SQLiteOpenHelper**: đối tượng dùng để tạo, nâng cấp, đóng mở kết nối CSDL
* **SQLiteDatabase**: đối tượng dùng để thực thi các câu lệnh SQL trên một CSDL
* SQLiteOpenHelper:
* Lớp này có 2 hàm khởi tạo, một hàm 4 tham số, một hàm 5 tham số, tuy nhiên, ta chủ yếu làm việc với hàm 4 tham số
* Tham số 1: **Context context**: Context là một lớp trừu tượng của hệ thống, chứa thông tin môi trường ứng dụng, cung cấp các phương thức để có thể tương tác với hệ điều hành, giúp chúng ta dễ dàng truy cập và tương tác tới các tài nguyên của hệ thống...
* Tham số 2: **String name**: Tên database
* Tham số 3: **CursorFactory factory**: thường để null
* Tham số 4: **Int version**: version của database
* Khi khởi tạo một đối tượng của lớp này, ta phải ghi đè 2 phương thức:
* **onCreate()**: phương thức này được gọi bởi framework, nếu có yêu cầu truy cập database mà lại chưa khởi tạo database, ở đây ta phải viết code khởi tạo database, cụ thể là khởi tạo bảng (chú ý: khi khởi tạo bảng, ta phải đặt tên khóa chính là **\_id**)
* **onUpgrade()**: phương thức này được dùng khi ứng dụng của bạn có nhiều phiên bản database đc thêm vào. Nó sẽ cập nhật database hiện có hoặc khởi tạo lại thông qua onCreate().
* SQLiteDatabase : Lớp này có các phương thức sau để làm việc với SQLite
* insert(), update(), delete()
* execSQL(): thực thi một câu lệnh SQL trực tiếp
* query(): dùng truy vấn
* ContentValues:
* Chúng ta không thể lưu trực tiếp dữ liệu xuống database, điều này cần phải sử dụng ContentValues. Các đối tượng ContentValues cho phép xác định khóa / giá trị. Các key đại diện nhận dạng cột bảng và value đại diện cho nội dung cho các bảng ghi trong cột này
* ContentValues có thể được sử dụng để chèn và cập nhật các mục cơ sở dữ liệu
* Cursor
* Đối tượng cursor hiểu đơn giản là một con trỏ, trỏ đến kết quả trả về của câu truy vấn. con trỏ này trỏ đến cái bảng trả về của câu truy vấn
* Ví dụ:

Cursor cursor = database.query(TABLE\_NAME, null, null, null, null, null, null);

* Phương thức **cursor.getCount()** trả về số dòng của bảng kết quả
* Đầu tiên ta di chuyển con trỏ này lên đầu bảng **cursor.moveToFirst();**  
  Sau khi đọc xong một dòng chạy lệnh **cursor.moveToNext();** để sang dòng tiếp theo  
  Khi cursor ở cuối bảng, không trỏ vào dòng nào, phương thức **cursor.isAfterLast()**  
  Sẽ trả về giá trị true  
  Để lấy ra thông tin, dung phương thức **cursor.getString(), cursor.getInt()**

1. Content Provider

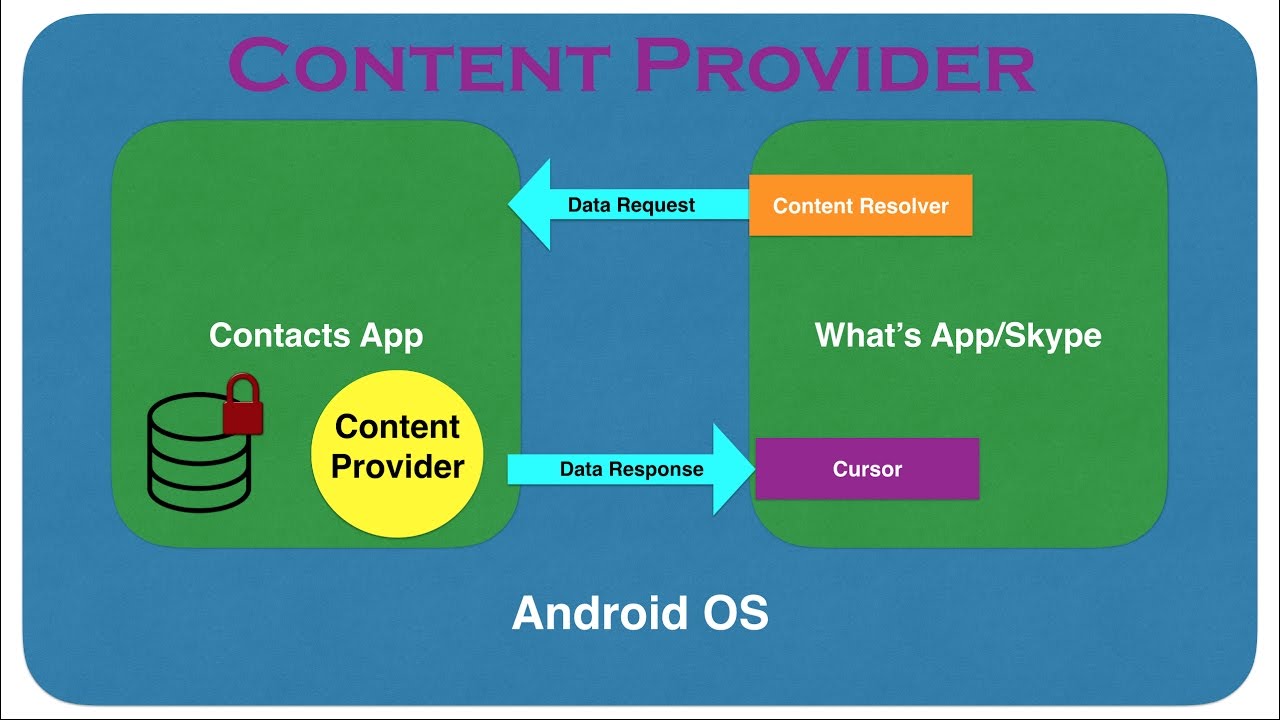
* Một thành phần Content Provider cung cấp dữ liệu từ một ứng dụng tới các ứng dụng khác theo yêu cầu. Những yêu cầu này được xử lý bởi các phương thức của lớp ContentResolver(đối tượng máy khách). Một Content Provider có thể sử dụng các cách khác nhau để lưu giữ dữ liệu và dữ liệu có thể được lưu giữ trong một cơ sở dữ liệu, một file, hoặc có thể thông qua một mạng..
* Các phương pháp [ContentResolver](https://developer.android.com/reference/android/content/ContentResolver.html?hl=vi) hoạt động giống cơ sở dữ liệu  cung cấp các chức năng "CRUD" (create, read, update , và delete) cơ bản của thiết bị lưu trữ liên tục.
* Content Provider điều phối việc truy cập tới bỗ lưu trữ dữ liệu thông qua các API và các component như hình dưới, nó bao gồm
* Chia sẻ dữ liệu từ ứng dụng của bán tới các ứng dụng khác
* Gửi dữ liệu sang widget
* Trả về một kết quả gợi ý khi search cho ứng dụng của bạn thông qua Seach Framework sử dụng [SearchRecentSuggestionsProvider](https://developer.android.com/reference/android/content/SearchRecentSuggestionsProvider.html)
* Đồng bộ dữ liệu của ứng dụng với server bằng cách sử dụng [AbstractThreadedSyncAdapter](https://developer.android.com/reference/android/content/AbstractThreadedSyncAdapter.html)
* Tải dữ liệu lên UI sử dụng [CursorLoader](https://developer.android.com/reference/android/content/CursorLoader.html)
* Để sử dụng Content Provider ta làm theo các bước sau:

1. Xác định kiểu dữ liệu
2. Xác định Uniform Resource Identifier (URI)
3. Khai báo Content Provider trong manifest
4. Implement lớp ContentProvider và các phương thức được yêu cầu.

* Các phương thức cần được Override trong lớp Content Provider:
* onCreate(): Phương thức này được gọi khi Provider được bắt đầu, nếu quá trình khởi tạo thành công trả về true, ngược lại là false
* query(): Phương thức nhận yêu cầu từ Client. Kết quả được trả về như một đối tượng Cursor.
* insert(): Phương thức chèn một dòng dữ liệu mới vào Content Provider.
* delete(): Phương thức xóa một dòng dữ liệu đã tồn tại.
* update(): Phương thức cập nhật một dòng dữ liệu nào đó đã tồn tại.
* getType(): Phương thức trả về kiểu MIME của dữ liệu tại các URI.
* Content URI
* Content URI là một URI định danh dữ liệu trong một provider. Content URI bao gồm kí hiệu tên của toàn bộ provider và một tên chỉ tới một bảng. Khi bạn gọi một phương thức truy cập tới bảng trong provider thì Content URI của bảng đó là sẽ là một tham số.  
  Để truy vấn data qua provider, ta sẽ sử dụng URI có định dạng như sau:

content://authority/path/id

* content: luôn là **content://**
* authority: một xâu để xác định tên của Content Provider, ví dụ như contact, browser,...
* path: có thể không có hoặc được chia làm nhiều phần, và được phân cách bằng gạch chéo "/", dùng để định ra các thành phần con của dữ liệu. Ví dụ như để lấy danh sách các liên hệ trong danh bạ thì URI sẽ là content://contacts/people.
* id: chỉ định rõ một bản ghi trong tập hợp dữ liệu, mỗi bản ghi sẽ được đánh dấu id là một số duy nhất.
* Một số ví dụ về URI:
* Content://media/internal/images – trả về List tất cả các ảnh lưu trong thiết bị
* Content://contact/people – Trả về list tất cả các tên trong danh bạ
* Content://contact/people/45 – Trả về một kết quả danh bạ có ID là 45



Bảng ContactsContract.Contacts Hàng thể hiện những người khác nhau dựa trên tổng hợp của các hàng liên lạc thô.

Bảng ContactsContract.RawContacts Hàng chứa một bản tổng hợp dữ liệu về một người, liên quan tới tài khoản và loại người dùng.

Bảng ContactsContract.Data Hàng chứa các thông tin chi tiết về liên lạc thô, chẳng hạn như địa chỉ e-mail hoặc số điện thoại.