Mỗi ứng dụng đều sử dụng dữ liệu, dữ liệu có thể đơn giản hay đôi khi là cả 1 cấu trúc. Trong Android thì hệ cơ sở dữ liệu được sử dụng là SQLite Database, đây là hệ thống mã nguồn mở được sử dụng rộng rãi trong các ứng dụng.

Trong Android , cơ sở dữ liệu mà bạn tạo cho 1 ứng dụng thì chỉ ứng dụng đó có quyền truy cập và sử dụng, các ứng dụng khác thì không. Khi đã được tạo, cơ sở dữ liệu SQLite được chứa trong thư mục **/data/data/<package\_name>/databases**

Các bước thực hiện với CSDL:

|  |
| --- |
| 1. Tạo 1 CSDL ( thông thường chỉ cần làm 1 lần)  2. Mở CSDL đó  3. Thêm giá trị vào trong table  4. Truy vấn.  5. Đóng CSDL |

1. [Tham khảo] tạo 1 CSDL SQLite, thêm vào CSDL danh sách các user, sau đó lấy dữ liệu từ CSDL hiện lên listview:



* Nội dung activity\_main.xml

|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"**>   <**ListView  android:id="@+id/lv\_user"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"** /> </**LinearLayout**> |

* Nội dung file item\_user.xml

|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**TextView xmlns:android="http://schemas.android.com/apk/res/android"  android:id="@+id/tv\_user"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:padding="@dimen/margin\_base"**/> |

* Tạo 1 class **DatabaseHelper**. Override 2 phương thức onCreate() và onUpgrade() để quản lý việc tạo CSDL và version của CSDL đó.

|  |
| --- |
| **package** lab01\_1.uit.edu.vn.lab03\_2\_1.helper;  **import** android.content.Context; **import** android.database.sqlite.SQLiteDatabase; **import** android.database.sqlite.SQLiteOpenHelper;  **public class** DatabaseHelper **extends** SQLiteOpenHelper {   **private static final** String ***DATABASE\_CREATE*** = **"create table users (\_id integer primary key autoincrement, "** + **"name text not null);"**;    **public** DatabaseHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, **int** version) {  **super**(context, name, factory, version);  *//* ***TODO Auto-generated constructor stub*** }   @Override  **public void** onCreate(SQLiteDatabase db) {  *//* ***TODO Auto-generated method stub*** db.execSQL(***DATABASE\_CREATE***);  }   @Override  **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  *//* ***TODO Auto-generated method stub*** db.execSQL(**"DROP TABLE IF EXISTS users"**);  onCreate(db);  } } |

* Đầu tiên các bạn add 1 class **DbAdapter** để xử lý tất cả các thao tác liên quan đến CSDL. Tất cả các chức năng đó đều được cung cấp bởi lớp SQLiteDatabase, các bạn chỉ cần cụ thể hóa bằng các câu truy vấn là được. Trong ví dụ này sẽ có các hàm mở, đóng CSDL; thêm, xóa, get user…

|  |
| --- |
| **package** lab01\_1.uit.edu.vn.lab03\_2\_1.adapter;  **import** android.content.ContentValues; **import** android.content.Context; **import** android.database.Cursor; **import** android.database.sqlite.SQLiteDatabase;  **import** lab01\_1.uit.edu.vn.lab03\_2\_1.helper.DatabaseHelper;  **public class** DbAdapter {  **public static final** String ***KEY\_ID*** = **"\_id"**;  **public static final** String ***KEY\_NAME*** = **"name"**;   **private** DatabaseHelper **dbHelper**;  **private** SQLiteDatabase **sqLiteDatabase**;  **private static final** String ***DATABASE\_NAME*** = **"Database\_Demo"**;  **private static final** String ***DATABASE\_TABLE*** = **"users"**;  **private static final int *DATABASE\_VERSION*** = 2;  **private final** Context **context**;   **public** DbAdapter(Context ctx) {  **this**.**context** = ctx;  }   **public** DbAdapter open() {  **dbHelper** = **new** DatabaseHelper(**context**, ***DATABASE\_NAME***, **null**, ***DATABASE\_VERSION***);  **sqLiteDatabase** = **dbHelper**.getWritableDatabase();  **return this**;  }   **public void** close() {  **dbHelper**.close();  }   **public long** createUser(String name) {  ContentValues inititalValues = **new** ContentValues();  inititalValues.put(***KEY\_NAME***, name);  **return sqLiteDatabase**.insert(***DATABASE\_TABLE***, **null**, inititalValues);  }   **public boolean** deleteUser(**long** rowId) {  **return sqLiteDatabase**.delete(***DATABASE\_TABLE***, ***KEY\_ID*** + **"="** + rowId, **null**) > 0;  }   **public boolean** deleteAllUsers() {  **return sqLiteDatabase**.delete(***DATABASE\_TABLE***, **null**, **null**) > 0;  }   **public** Cursor getAllUsers() {  **return sqLiteDatabase**.query(***DATABASE\_TABLE***, **new** String[]{***KEY\_ID***, ***KEY\_NAME***}, **null**, **null**, **null**, **null**, **null**);  } } |

* Nội dung **MainActivity**

|  |
| --- |
| **package** vn.edu.uit.lab03\_2\_1;  **import** android.database.Cursor; **import** android.os.Bundle; **import** android.support.v7.app.ActionBarActivity; **import** android.widget.ArrayAdapter; **import** android.widget.ListView;  **import** java.util.ArrayList; **import** java.util.List;  **import** vn.edu.uit.lab03\_2\_1.adapter.DbAdapter;  **public class** MainActivity **extends** ActionBarActivity {   **private** DbAdapter **dbAdapter**;  **private** Cursor **cursor**;  **private** List<String> **users**;   @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_main***);   **dbAdapter** = **new** DbAdapter(**this**);  **dbAdapter**.open();  **dbAdapter**.deleteAllUsers();  **for** (**int** i = 0; i < 10; i++) {  **dbAdapter**.createUser(**"Nguyễn Văn An "** + i);  }   **users** = getData();  showData();  }   **private** List<String> getData() {  List<String> users = **new** ArrayList<>();   **cursor** = **dbAdapter**.getAllUsers();  **while** (**cursor**.moveToNext()) {  users.add(**cursor**.getString(**cursor**.getColumnIndex(DbAdapter.***KEY\_NAME***)));  }   **return** users;  }   **private void** showData() {  ListView lvUser = (ListView) findViewById(R.id.***lv\_user***);  ArrayAdapter<String> userAdapter = **new** ArrayAdapter<String>(MainActivity.**this**, R.layout.***item\_user***, **users**);  lvUser.setAdapter(userAdapter);  } } |

1. [Thực hành] Quản lý Contact
2. CSDL dữ liệu có cấu trúc như sau:

* **Database Name:** contactsManager
* **Table Name:** contacts

|  |  |  |
| --- | --- | --- |
| **Columns** | **DataType** | **Key** |
| **id** | INTEGER | Primary key |
| **name** | TEXT |  |
| **phone\_number** | TEXT |  |

1. Hướng dẫn thực hiện
2. Tạo một lớp Contact:

|  |
| --- |
| **public** **class** Contact {  private **int** id;  private String name;  private String phoneNumber;    //….  } |

1. Tạo 1 lớp DatabaseHandler kế thừa từ SQLiteOpenHelper

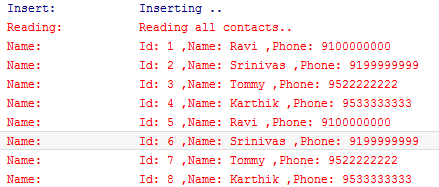
* **Override hàm onCreate**() để tạo table
* **Override hàm onUpgrade**() hủy table cũ, tạo table mới
* Thêm các phương thức để insert, select, update và delete contact

|  |
| --- |
| **import** java.util.ArrayList;  **import** java.util.List;  **import** android.content.ContentValues;  **import** android.content.Context;  **import** android.database.Cursor;  **import** android.database.sqlite.SQLiteDatabase;  **import** android.database.sqlite.SQLiteOpenHelper;  **public** **class** DatabaseHandler **extends** SQLiteOpenHelper {    // All Static variables  // Database Version  **private** **static** **final** **int** *DATABASE\_VERSION* = 1;    // Database Name  **private** **static** **final** String *DATABASE\_NAME* = "contactsManager";    // Contacts table name  **private** **static** **final** String *TABLE\_CONTACTS* = "contacts";    // Contacts Table Columns names  **private** **static** **final** String *KEY\_ID* = "id";  **private** **static** **final** String *KEY\_NAME* = "name";  **private** **static** **final** String *KEY\_PH\_NO* = "phone\_number";    **public** DatabaseHandler(Context context) {  **super**(context, *DATABASE\_NAME*, **null**, *DATABASE\_VERSION*);  }    // Creating Tables  @Override  **public** **void** onCreate(SQLiteDatabase db) {  String CREATE\_CONTACTS\_TABLE = "CREATE TABLE " + *TABLE\_CONTACTS* + "("  + *KEY\_ID* + " INTEGER PRIMARY KEY," + *KEY\_NAME* + " TEXT,"  + *KEY\_PH\_NO* + " TEXT" + ")";  db.execSQL(CREATE\_CONTACTS\_TABLE);  }    // Upgrading database  @Override  **public** **void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  // Drop older table if existed  db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_CONTACTS*);    // Create tables again  onCreate(db);  }    // Các hàm còn lại: để insert, select, update và delete contact sv tự viết  // Adding new contact  public void addContact(Contact contact) {}    // Getting single contact  public Contact getContact(int id) {}    // Getting All Contacts  public List<Contact> getAllContacts() {}  // Updating single contact  public int updateContact(Contact contact) {}    // Deleting single contact  public void deleteContact(Contact contact) {} |

1. MainActivity.java

|  |
| --- |
| **public** **class** MainActivity **extends** Activity {  @Override  **protected** **void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);    DatabaseHandler db = **new** DatabaseHandler(**this**);    /\*\*  \* CRUD Operations  \* \*/  // Inserting Contacts  Log.*d*("Insert: ", "Inserting ..");  db.addContact(**new** Contact("Ravi", "9100000000"));  db.addContact(**new** Contact("Srinivas", "9199999999"));  db.addContact(**new** Contact("Tommy", "9522222222"));  db.addContact(**new** Contact("Karthik", "9533333333"));    // Reading all contacts  Log.*e*("Reading: ", "Reading all contacts..");  List<Contact> contacts = db.getAllContacts();    **for** (Contact cn : contacts) {  String log = "Id: "+cn.getID()+" ,Name: " + cn.getName() + " ,Phone: " + cn.getPhoneNumber();  // Writing Contacts to log  Log.*e*("Name: ", log);  }  } |

Kết quả hiển thị hiện tại của ứng dụng (xem tại Log Cat)



**Sinh viên tùy chỉnh lại code để hiển thị list Contact ra listview. Khi nhấn giữ lâu vào 1 contact nào đó (long click) thì ứng dụng xóa contact đó khỏi Database.**

1. THAM KHẢO THÊM (Tự xem thêm)
2. Cách debug trong Android Studio
   * Hướng dẫn: <http://blog.strv.com/debugging-in-android-studio-as/>
   * Một số phím thường dùng khi debug: <http://www.praveenboyalapalli.com/2015/03/how-to-debug-project-android-studio/>
3. **GreenDao**
   * <http://greenrobot.org/greendao/>
   * <http://greenrobot.org/greendao/documentation/how-to-get-started/>
   * <https://github.com/greenrobot/greenDAO>
4. EventBus; cách dùng interface trong Java
   * Interface: <http://vietjack.com/java/interface_trong_java.jsp>; <http://www.nguyenvanquan7826.com/2015/04/05/java-su-dung-interface/>
   * EventBus: <https://github.com/greenrobot/EventBus>
5. NinePatch
   * <https://developer.android.com/studio/write/draw9patch.html>
6. Firebase Cloud Firestore
   * Documentation add Firebase to your project, follow **Option 1**: <https://firebase.google.com/docs/android/setup>
   * Get Started with Cloud Firestore: <https://firebase.google.com/docs/firestore/quickstart#java_2>
   * Queries data in Cloud Firestore: https://firebase.google.com/docs/firestore/query-data/queries#java