

Bài 08: SQLITE DATABASE

GVGD: ThS. Đặng Thế Hân

Biên soạn: ThS. Giang Hào Côn

Mục tiêu

Cung cấp cho sinh viên kiến thức cơ bản về bộ nhớ trong ứng dụng Android và cách sử dụng SQLITE DATABASE trong ngôn ngữ lập trình ứng dụng trên Android.

Nội dung

- 1) Giới thiệu SQLITE
- 2) Tạo SQLITE DATABASE
- 3) Thao tác thêm sửa xoá trong SQLITE
- 4) Truy Vấn trong SQLITE

1. Giới Thiệu

- Khi thực hiện một ứng dụng, việc chọn nền tảng Cơ sở dữ liệu (CSDL) là một việc khá quan trọng. Chọn được cơ sở dữ liệu phù hợp với ứng dụng, sẽ giúp ứng dụng hoạt động dễ dàng và trơn tru hơn.
- Hiện nay, có khá nhiều CSDL để bạn có thể sử dụng cho việc lưu trữ trên ứng dụng của mình như **Web Service**, **Firebase**,... và **SQLite** là một trong số đó.
- **SQLite** là một cơ sở dữ liệu quan hệ, mã nguồn mở và được tích hợp sẵn trên Android. **SQLite** thường được sử dụng trong các ứng dụng ở Local, như các ứng dụng **Danh bạ**, **Tin nhắn**, **Ghi chú**, **Quản lý thông tin cá nhân**, Các tùy chọn thiết lập (Setting) trong phần mềm,...v....v

1. Giới Thiệu

- SQLite hỗ trợ cú pháp truy vấn giống SQL.
- SQLite hỗ trợ các kiểu dữ liệu như: **TEXT** (giống như kiểu chuỗi), **INTEGER** (giống như kiểu Long), **REAL** (giống như kiểu Double). Tất cả các kiểu dữ liệu khác cần được chuyển đổi thành một trong ba kiểu dữ liệu trên trước khi lưu vào Database.
- SQLite được nhúng vào các thiết bị Android. Sử dụng Sqlite Database trong Android không yêu cầu cài đặt cũng như xin quyền truy xuất.
- Khi ứng dụng tạo một Database thì đường dẫn mặc định được tạo ra là: **data/data/tên_ứng_dụng/databases/filename**.

1. Giới Thiệu

- Để tạo và cập nhật một database trong ứng dụng ta cần tạo ra một class kế thừa lớp **SQLiteOpenHelper**.
- Trong phương thức khởi tạo của class này tại phương thức `super()` truyền vào **tên database** và **Version** của database.
- Trong class này cần phải override 2 phương thức để tạo và cập nhật database là **onCreate** và **onUpgrade**. Hai phương thức này sẽ nhận đối tượng **SQLiteDatabase** như là 1 tham số.
- Class **SQLiteOpenHelper** cung cấp **getReadableDatabase()** và **getWritableDatabase()** để truy cập đến đối tượng **SQLiteDatabase**.

2. Giới Thiệu Câu lệnh SQLite

1) NHÓM TRUY VẤN KHÔNG TRẢ VỀ DỮ LIỆU:

- Tạo một bảng:

Create Ten_bang (Tên_Cột Kiểu_Dữ_Liệu(Kích_thuoc)).

- Xóa một bảng:

Drop Table Ten_bang.

- Thêm một dòng vào bảng:

Insert Into Ten_bang(Cac_cot) **Values** (Gia_tri_tuong_ung).

- Sửa một dòng trong bảng:

Update Ten_bang **Set** Tên_Cột = Giá_Trị **Where** Dieu_kien.

- Xóa một dòng trong bảng:

Delete From Ten_bang **Where** Dieu_kien.

2. Giới Thiệu Câu lệnh SQLite

2. NHÓM TRUY VẤN TRẢ VỀ DỮ LIỆU:

- Lấy ra các dòng trong bảng:

Select Ten_cot **From** Ten_bang **Where** Dieu_kien.

- Lấy ra tất cả dòng trong bảng:

Select * **From** Ten_bang.

Các câu truy vấn trong SQLite được tạo thông qua 2 phương thức là **rawquery()** và **query()** hoặc thông qua class **SQLiteQueryBuilder**.

Ví dụ rawQuery

```
Cursor cursor = getReadableDatabase().  
    rawQuery("select * from todo where _id = ?", new String[] { id });
```


2. Giới Thiệu Câu lệnh SQLite

2. NHÓM TRUY VẤN TRẢ VỀ DỮ LIỆU:

- Lấy ra các dòng trong bảng:

Select Ten_cot **From** Ten_bang **Where** Dieu_kien.

- Lấy ra tất cả dòng trong bảng:

Select * **From** Ten_bang.

Các câu truy vấn trong SQLite được tạo thông qua 2 phương thức là **rawquery()** và **query()** hoặc thông qua class **SQLiteQueryBuilder**.


Ví dụ query

```
Cursor cursor = database.query(Table_Name, new String[] {field1, field2, ...}, null, null, ...);
```

2. Giới Thiệu Câu lệnh SQLite

Sau khi tạo 3 phương thức trên, Class của bạn sẽ như sau:

```
class DataSqlite extends SQLiteOpenHelper {  
    DataSqlite(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {  
        super(context, name, factory, version);  
    }  
  
    @Override  
    public void onCreate(SQLiteDatabase sqLiteDatabase) {  
  
    }  
  
    @Override  
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
  
    }  
}
```



Tên CSDL cần tạo.

2. Giới Thiệu Câu lệnh SQLite

Sau khi tạo 3 phương thức trên, Class của bạn sẽ như sau:

```
class DataSqlite extends SQLiteOpenHelper {  
    DataSqlite(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {  
        super(context, name, factory, version);  
    }  
  
    @Override  
    public void onCreate(SQLiteDatabase sqLiteDatabase) {  
  
    }  
  
    @Override  
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
  
    }  
}
```

Được gọi bởi Framework.
Nếu CSDL chưa được tạo
thì nó tạo ra CSDL, bảng.

2. Giới Thiệu Câu lệnh SQLite

Sau khi tạo 3 phương thức trên, Class của bạn sẽ như sau:

```
class DataSqlite extends SQLiteOpenHelper {  
    DataSqlite(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {  
        super(context, name, factory, version);  
    }  
  
    @Override  
    public void onCreate(SQLiteDatabase sqLiteDatabase) {  
  
    }  
  
    @Override  
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
  
    }  
}
```

Được gọi khi Database tăng trong ứng dụng. Nó cho phép cập nhật, xoá Database.

2. Giới Thiệu Câu lệnh SQLite

Lúc này, ta tạo thêm 2 hàm nữa để phục vụ cho việc Truy vấn của chúng ta sau này.

1. Hàm đầu tiên phục vụ cho việc truy vấn KHÔNG trả về dữ liệu:

```
void TruyVanKhongTraVe(String sql)
{
    SQLiteDatabase db=getWritableDatabase();
    db.execSQL(sql);
}
```


2. Giới Thiệu Câu lệnh SQLite

- Lúc này, ta tạo thêm 2 hàm nữa để phục vụ cho việc Truy vấn của chúng ta sau này.
- 2. Hàm thứ 2 phục vụ cho việc truy vấn Trả về dữ liệu. Hàm này sẽ
 - return về một dữ liệu dạng Con trỏ (Cursor):

```
Cursor TruyVanTraVe(String sql)
{
    SQLiteDatabase db = getReadableDatabase();
    return db.rawQuery(sql, null);
}
```

Như vậy, [Class DataSqlite](#) của chúng ta đầy đủ sẽ như sau:

2. Giới Thiệu Câu lệnh SQLite

Ví dụ Demo: xây dựng một App nhỏ để Quản lý sinh viên, ta sẽ có 1 bảng lưu có 2 trường là ID và TenSV. (ID tự động tăng để biết thêm cách cho 1 ID tự động tăng trong SQLite).

Bước 01: Thiết kế giao diện. [activity_main.xml](#)

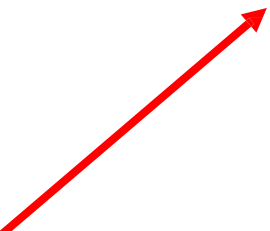
Bước 02: Tạo lớp tên [ObjectSinhvien](#). Để lưu các giá trị của mỗi sinh viên

```
class ObjectSinhvien {  
    Integer maSV;  
    String tenSV;  
  
    ObjectSinhvien(Integer maSV, String tenSV) {  
        this.maSV = maSV;  
        this.tenSV = tenSV;  
    }  
}
```

2. Giới Thiệu Câu lệnh SQLite

Bước 03: Mở file MainActivity.Java. Khai báo các thành phần

```
Button btnThem;  
EditText edtTenSV;  
ListView lvDanhSach;  
  
DataSqlite data;  
ArrayList<ObjectSinhvien> arrayData;
```



DataSqlite chính là tên của Class Java mà ta vừa **extends** từ **SQLiteOpenHelper** ở trên, ngoài ra ta khai báo thêm một mảng **ArrayList** để chứa dữ liệu khi ta lấy từ Database lên, các bạn tiến hành ánh xạ các Button, Listview và EditText như bình thường.

2. Giới Thiệu Câu lệnh SQLite

Bước 04: Để tạo một database, trong hàm onCreate(), các ta code như sau:

[Code demo](#)

```
data = new DataSQLite(MainActivity.this, "Demo.sqlite", null, 1);  
data.TruyVanKhongTraVe("Create Table If not exists QuanLySinhVien(  
    ID Integer Primary Key Autoincrement,  
    TenSV Varchar)");
```

Trong dòng đầu tiên, **Demo.sqlite** là tên Database. Ở câu thứ hai, chúng ta gọi hàm **TruyVanKhongTraVe**, “**If not exists**” có nghĩa là nếu bảng với tên **QuanLySinhVien** đã tồn tại rồi thì chúng ta không tạo nữa, “**Autoincrement**” là câu lệnh để cột ID tự động tăng trong Database. Sau khi câu lệnh 2 chạy, một bảng với tên QuanLySinhVien sẽ được tạo.

2. Giới Thiệu Câu lệnh SQLite

Bước 05: Thêm dữ liệu vào bảng: gọi sự kiện Click cho nút btnThem.

```
btnThem.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        if (!edtTenSV.getText().toString().trim().equals("")) {  
            data.TruyVanKhongTraVe("Insert into QuanLySinhVien values (null, '" + edtTenSV.getText().toString().trim() + "')");  
            Toast.makeText(MainActivity.this, "Thêm thành công sinh viên " + edtTenSV.getText().toString(),  
                Toast.LENGTH_SHORT).show();  
            edtTenSV.setText("");  
        } else {  
            Toast.makeText(MainActivity.this, "Hãy nhập tên sinh viên!!", Toast.LENGTH_SHORT).show();  
        }  
    }  
});
```

2. Giới Thiệu Câu lệnh SQLite

Bước 06: Lấy dữ liệu từ bảng ra ListView. Ta viết phương thức LoadData như sau:

```
private void LoadData() {  
    arrayData = new ArrayList<>();  
    Cursor cursor = data.TruyVanTraVe("Select * From QuanLySinhVien");  
    while (cursor.moveToNext()) {  
        arrayData.add(new ObjectSinhvien(cursor.getInt(0), cursor.getString(1)));  
    }  
    CustomAdapter adapter = new CustomAdapter(MainActivity.this, R.layout.item, arrayData);  
    adapter.setNotifyOnChange(true);  
    lvDanhSach.setAdapter(adapter);  
}
```

Code LoadData

SQLite Trong Android

Thao tác Sửa, Xóa và Tìm Kiếm trong SQLite

3. Sửa, Xóa và Tìm kiếm

Mở lại activity_main.xml ta sẽ thêm 3 Button để thực hiện chức năng Sửa, Xóa và Tìm kiếm, cụ thể như sau: [Code](#)

Trong **MainActivity.Java** chúng ta khai báo và ánh xạ 3 Button này:

```
Button btnThem, btnSua, btnXoa, btnTimKiem;  
EditText edtTenSV;  
ListView lvDanhSach;  
  
DataSqlite data;  
  
ArrayList<ObjectSinhvien> arrayData;  
  
int position;
```

Ở đây có khai báo thêm một biến **position**, biến này có công dụng là để lưu lại vị trí của dòng muốn sửa hay xóa, các btnSua, btnXoa, btnTimKiem.

3. Sửa, Xoá và Tìm kiếm

Thao tác Sửa trong Sqlite

Đầu tiên, để Sửa được một dòng trong SQLite, các bạn phải bắt sự kiện **OnItemClickListener()** của **lvDanhSach**, sự kiện này có tác dụng là khi Click vào 1 dòng trong **ListView**, nó sẽ hiển thị giá trị của dòng đó lên **EditText** Tên Sinh Viên để chúng ta sửa. Sự kiện **OnItemClickListener()** của chúng ta sẽ như sau:

```
lvDanhSach.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
    @Override  
    public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {  
        edtTenSV.setText(arrayData.get(i).tenSV);  
        position = i;  
    }  
});
```


3. Sửa, Xoá và Tìm kiếm

Thao tác Sửa trong Sqlite

Sự kiện Click của btnSua:

```
btnSua.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        if (!edtTenSV.getText().toString().trim().equals("")){  
            data.TruyVanKhongTraVe("Update QuanLySinhVien set TenSV='" + edtTenSV.getText().toString().trim() +  
                                   "' where ID=" + arrayData.get(position).maSV + "");  
            Toast.makeText(MainActivity.this, "Sửa thành công sinh viên" + edtTenSV.getText(),  
                           Toast.LENGTH_SHORT).show();  
            LoadData();  
        }  
        else {  
            Toast.makeText(MainActivity.this, "Hãy chọn sinh viên cần sửa!!", Toast.LENGTH_SHORT).show();  
        }  
    }  
});
```

Code Demo

3. Sửa, Xoá và Tìm kiếm

Thao tác Xoá trong Sqlite

Sự kiện Click của btnXoa:

```
btnXoa.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        if (!edtTenSV.getText().toString().trim().equals("")) {  
            data.TruyVanKhongTraVe("Delete from QuanLySinhVien where ID=" +  
                arrayData.get(position).maSV + "");  
            Toast.makeText(MainActivity.this, "Xóa thành công sinh viên " + edtTenSV.getText(),  
                Toast.LENGTH_SHORT).show();  
            edtTenSV.setText("");  
            LoadData();  
        } else {  
            Toast.makeText(MainActivity.this, "Hãy chọn sinh viên muốn xóa!!",  
                Toast.LENGTH_SHORT).show();  
        }  
    }  
});
```


3. Sửa, Xoá và Tìm kiếm

Thao tác Tìm kiếm trong Sqlite

Để Tìm kiếm, chúng ta sử dụng hàm TruyVanTraVe() của Class DataSqlite, Bản chất của việc Tìm kiếm trong SQLite đó là chúng ta sẽ dùng một câu lệnh Select nhưng đi kèm thêm với điều kiện, ở đây tìm kiếm theo **Tên sinh viên** nên điều kiện sẽ là Tên sinh viên, các bạn cũng có thể tìm kiếm theo mã, hoặc cùng lúc tìm kiếm theo Mã sinh viên và Tên sinh viên, chỉ cần chúng ta tùy biến lại câu Select mà thôi.

Trong ví dụ tìm kiếm tất cả Sinh viên có tên giống tên nhập vào EditText, sau đó đổ vào ArrayList và hiển thị lên ListView.

3. Sửa, Xoá và Tìm kiếm

Thao tác Tìm kiếm trong Sqlite

```
btnTimKiem.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        if (!edtTenSV.getText().toString().trim().equals("")) {  
            arrayData = new ArrayList<>();  
            Cursor cursor = data.TruyVanTraVe("Select * from QuanLySinhVien where TenSV like '%" +  
                                                + edtTenSV.getText().toString().trim() + "%'");  
            while (cursor.moveToNext()) {  
                arrayData.add(new ObjectSinhvien(cursor.getInt(0), cursor.getString(1)));  
            }  
            CustomAdapter adapter = new CustomAdapter(MainActivity.this, R.layout.item, arrayData);  
            lvDanhSach.setAdapter(adapter);  
        } else {  
            LoadData();  
        }  
    }  
});
```

SQLite Trong Android

Sử dụng Đối Tượng
SQLiteDatabase để thực thi các
câu lệnh SQL trên CSDL

Các Phương thức làm việc với SQLite

- ■ Insert()
- ■ Update()
- ■ Delete()
- ■ execSQL(): thực thi một câu lệnh SQL trực tiếp
- ■ Query(): truy vấn dữ liệu

Phương thức Query()

■ Cú pháp:

Điều kiện lọc
và giá trị của
tham số tham
gia điều kiện
lọc

```
query(String table,  
      String[] columns,  
      {String selection,  
       String[] selectionArgs,  
       String groupBy,  
       String having,  
       String orderBy})
```

Tên bảng cần truy vấn

Tên field, nếu là null
thì lấy tất cả fields

Phương thức query sử dụng để truy vấn dữ liệu trong bảng. **Phương thức này trả về đối tượng cursor.** Đối tượng cursor là một con trỏ, trỏ đến kết quả trả về của câu truy vấn. Kết quả trả về của câu truy vấn là cái bảng mà ta truy vấn.

Phương thức Query()

- Các phương thức của cursor

<code>getCount()</code>	Trả về số dòng của bảng kết quả
<code>moveToFirst()</code>	Di chuyển lên dòng đầu của bảng
<code>moveToNext()</code>	Di chuyển sang dòng tiếp theo
<code>isAfterLast()</code>	Kiểm tra xem đã đến cuối bảng ?
<code>getString(), getInt()</code>	Lấy thông tin theo tên cột hoặc chỉ mục

Phương thức Query()

- Ví dụ sử dụng phương thức query

```
Cursor cursor = null;
```

```
cursor = sqlDB.query("SinhVien", null, "mssv = ? ", new String[] { String.valueOf(msv)}, null, null, null);
```

Biến
CSDL

tên
Table

Lấy tất
cả fields

Điều kiện lọc

Phương thức insert()

- Sử dụng phương thức này để insert một bản ghi vào CSDL. Ví dụ. ta có một đối tượng database thuộc kiểu SQLiteDatabases. Dùng đối tượng **ContentValues** để đưa dữ liệu vào bảng. Đối tượng này có các phương thức **put (tên cột , dữ liệu)** . Sau đó gọi phương thức insert để đưa đối tượng (dòng này) vào bảng.

```
ContentValues ct = new ContentValues();  
ct.put("full_name", "Thích Thi Rớt");  
ct.put("student_id", "20131271");  
ct.put("gender", 1);  
ct.put("year", 21);  
database.insert("Sinhvien", null, ct);
```


Phương thức update()

- cập nhật dữ liệu trong bảng theo một điều kiện bất kỳ.

```
public void updateStudent(SinhVien sv) {  
    SQLiteDatabase db = this.getWritableDatabase();  
  
    ContentValues ct = new ContentValues();  
    ct.put("name", sv.getName());  
    ct.put("address", sv.getAddress());  
    ct.put("phone_number", sv.getPhone_number());  
  
    db.update("Sinhvien", values, "masv = ?", new String[] { String.valueOf(sv.getId()) });  
    db.close();  
}
```

Phương thức delete()

- để xóa một hoặc một số record trong bảng theo một điều kiện.

```
public void deleteStudent(int msv) {  
    SQLiteDatabase db = this.getWritableDatabase();  
  
    db.delete("Sinhvien", "masv = ?", new String[] { String.valueOf(msv) });  
    db.close();  
}
```

Phương thức execSQL()

- Phương thức này dùng để thực thi một câu lệnh SQL trực tiếp.

```
//Thêm mới  
void insertProduct(Product product)  
{  
    SQLiteDatabase db = getWritableDatabase();  
  
    db.execSQL("INSERT INTO product (name, price ) VALUES (?,?)",  
               new String[]{product.name, product.price});  
    db.Close();  
}
```

Phương thức execSQL()

- Phương thức này dùng để thực thi một câu lệnh SQL trực tiếp.

```
//Cập nhật  
void updateProduct(Product product)  
{  
    SQLiteDatabase db = getWritableDatabase();  
  
    db.execSQL("UPDATE product SET name=?, price = ? where id = ?",  
               new String[]{product.name, product.price, product.productID});  
    db.Close()  
}
```

Phương thức execSQL()

- Phương thức này dùng để thực thi một câu lệnh SQL trực tiếp.

```
//Xoá sản phẩm
void deleteProductByID(int ProductID)
{
    SQLiteDatabase db = getWritableDatabase();

    db.execSQL("DELETE FROM product where id = ?",
               new String[] {String.valueOf(ProductID)});
    db.Close();
}
```

Câu hỏi thảo luận

- 1) Trình bày từng bước tạo Sqlite Database
- 2) Trình bày các thao tác thêm, sửa và xoá trong Sqlite Database
- 3) Trình bày thao tác truy vấn trong Sqlite Database