LẬP TRÌNH CHO CÁC THIẾT BỊ DI ĐỘNG

Bài 9: SQLITE

Giới thiệu

- SQLite là một cơ sở dữ liệu SQL mã nguồn mở, nó lưu trữ dữ liệu vào một tập tin văn bản trên một thiết bị. Nó mặc định đã được tích hợp trên thiết bị Android. Để truy cập dữ liệu này, ta không cần phải thiết lập bất kỳ loại kết nối nào cho nó như JDBC, ODBC, ...
- SQLite được Richard Hipp viết dưới dạng thư viện bằng ngôn ngữ lập trình C.

Ưu điểm

- 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

Cách sử dụng SQLite

- 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
- Lớp này có 2 hàm khởi tạo,
  - Hàm 4 tham số:

SQLiteOpenHelper(Context context, String name , SQLiteDatabase.CursorFactory factory, int version)

  - Hàm 5 tham số:

SQLiteOpenHelper(Context context, String name , SQLiteDatabase.CursorFactory factory, int version, DatabaseErrorHandler errorHandler)

SQLiteOpenHelper

- 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

SQLiteOpenHelper

- 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().
- Lớp này có 2 phương thức getReadableDatabase() (chỉ đọc) và getWriteableDatabase() (cho phép ghi đọc). Thông qua 2 phương thức này, ta có thể tạo ra một đối tượng SQLiteDatabase

SQLiteDatabase

- Để tạo ra một cơ sở dữ liệu, bạn chỉ cần gọi phương thức openOrCreateDatabase này với tên cơ sở dữ liệu của bạn và chế độ như một tham số. Nó trả về một thể hiện của cơ sở dữ liệu SQLite mà bạn phải nhận được trong cú pháp object.
- SQLiteDatabse mydatabase = openOrCreateDatabase("your database name",MODE_PRIVATE,null);

SQLiteDatabase

- openDatabase(String path, SQLiteDatabase.CursorFactory factory, int flags, DatabaseErrorHandler errorHandler): Phương thức này chỉ mở cơ sở dữ liệu hiện có với các chế độ cờ thích hợp. Các chế độ cờ phổ biến có thể là OPEN_READWRITE, OPEN_READONLY.
- openDatabase(String path, SQLiteDatabase.CursorFactory factory, int flags): Phương thức này tương tự như phương thức trên vì nó cũng mở ra các cơ sở dữ liệu hiện có, nhưng nó không định nghĩa bất kỳ xử lý nào để xử lý các lỗi cơ sở dữ liệu.

SQLiteDatabase

- openOrCreateDatabase(String path, SQLiteDatabase.CursorFactory factory): Phương thức này không chỉ mở ra mà còn tạo ra cơ sở dữ liệu nếu nó không tồn tại.
- openOrCreateDatabase(File file, SQLiteDatabase.CursorFactory factory): Phương thức này cũng tương tự như phương thức trên nhưng phải xác định File đối tượng như là một đường dẫn mà không phải là một chuỗi. Nó tương đương với file.getPath()

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

insert()

- Sử dụng phương thức này để insert một bản ghi vào CSDL
- Ví dụ. ta có một đối tượng database thuộc kiểu SQLiteDatabase
- 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.

insert()

```
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(TABLE_NAME,null,ct);
```

update()

- Phương thức update của SQLiteDatabase để cập nhật dữ liệu trong bảng theo một điều kiện bất kỳ nào đó. Phương thức này trả về số dòng bị ảnh hưởng. Ví dụ nếu có 3 dòng bị thay đổi thì nó trả về 3. nếu không có dòng nào bị ảnh hưởng thì nó trả về 0.
- public int update(String table, ContentValues values, String whereClause, String[] whereArgs)

update()

- Tham số 1: tên bảng
- Tham số 2: đối tượng muốn chỉnh sửa (với giá trị mới)
- Tham số 3: tập các điều kiện lọc (dùng dấu ? để tạo điều kiện lọc)
- Tham số 4: tập các giá trị của điều kiện lọc (lấy theo đúng thứ tự)

update()

```
public void updateStudent(Student student) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put(KEY_NAME, student.getName());
    values.put(KEY_ADDRESS, student.getAddress());
    values.put(KEY_PHONE_NUMBER,
student.getPhone_number());

    db.update(TABLE_NAME, values, KEY_ID + " = ?", new
String[] { String.valueOf(student.getId()) });
    db.close();
}
```

delete()

- Ta sử dụng phương thức delete của SQLiteDatabase để xóa dữ liệu của một hoặc một số record trong bảng theo một điều kiện bất kỳ nào đó. Phương thức này trả về số dòng bị ảnh hưởng. Muốn xóa toàn bộ dữ liệu trong bảng thì ta truyền null vào 2 đối số cuối

delete()

- public int delete(String table, String whereClause, String[] whereArgs)
  - Tham số 1: tên bảng
  - Tham số 2: tập điều kiện lọc
  - Tham số 3: tập các giá trị của điều kiện lọc

delete()

```
public void deleteStudent(int studentId) {
    SQLiteDatabase db = this.getWritableDatabase();
    db.delete(TABLE_NAME, KEY_ID + " = ?", new String[] {
String.valueOf(studentId) });
    db.close();
}
```

execSQL()

- Phương thức này dùng để thực thi một câu lệnh SQL trực tiếp. Phương thức execSQL không chỉ thêm dữ liệu, mà còn được sử dụng để cập nhật hoặc sửa đổi dữ liệu đã tồn tại trong cơ sở dữ liệu bằng cách sử dụng tham số ràng buộc
- execSQL(String sql)

execSQL()

```
public void doCreateSinhvienTable()
{
    String sql="CREATE TABLE tblsinhvien ("+
            "masv TEXT PRIMARY KEY ,"+
            "tensv TEXT,"+
            "malop TEXT NOT NULL CONSTRAINT malop "+
            " REFERENCES tbllop(malop) ON DELETE CASCADE)";
    database.execSQL(sql);
}
```

query()

- Phương thức này sử dụng để truy vấn dữ liệu trong bảng. Là thao tác phức tạp nhất trong truy suất SQLite. Ta dùng Cursor để lưu trữ giá trị trả về của hàm.
- public Cursor query(String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy)

query()

Tham số	Giải thích
table	Tên bảng cần truy vấn
columns	Danh sách các cột cần truy vấn, nếu tham số null là lấy tất cả các cột
selection	Lọc điều kiện, giống như mệnh đề WHERE của SQL. Nếu truyền giá trị là null sẽ lấy tất cả dữ liệu. các truyền tên cột = ? ví dụ: Id=?.
selectionArgs	Giá trị cần lọc ở tham số selection.
groupBy	Nhóm các dòng giống nhau, giống mệnh đề GROUP BY của SQL . Truyền giá trị null sẽ không nhóm.
having	Cho phép lọc nhóm kết quả nào sẽ xuất hiện trong kết quả cuối cùng của bảng ghi
orderBy	Sắp xếp các dòng dữ liệu, giống mệnh đề ORDER BY trong SQL. Nếu truyền giá trị null sẽ sắp xếp mặc định, có thể là không sắp xếp.

query()

```
Cursor cursor = null;
cursor = sqlDB.query(TABLE_NAME, null, "student_id = "
+ studentID, null, null, null, null);
cursor.moveToFirst();
Student sv = new
Student(cursor.getInt(0),cursor.getString(1),
cursor.getString(2), cursor.getInt(3), 22);
```

Đối tượng 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

```
Cursor cursor = database.query(TABLE_NAME, null, null, null,
                                null, null, null);
```

- cursor.getCount(): Phương thức trả về số dòng của bảng kết quả.
- cursor.moveToFirst(): Phương thức này di chuyển con trỏ này lên đầu bảng .
- cursor.moveToNext(): Phương thức này để di chuyển sang dòng tiếp theo.
- cursor.isAfterLast(): Kiểm tra xem cursor ở cuối bảng?, không trỏ vào dòng nào, phương thức này sẽ trả về giá trị true.
- cursor.getString(), cursor.getInt(): Hai phương này để lấy ra thông tin cột theo tên cột hoặc index (chỉ mục).

Đối tượng 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

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