多架构数据库技术

Android 使用 SQLite 数据库

ougd . ai . wangjie

创建数据库

Android 不自动提供数据库。在 Android 应用程序中使用 SQLite，必须自己创建数据库，然后创建表、索引，填充数据。Android 提供了 SQLiteOpenHelper 帮助创建一个数据库，只要继承 SQLiteOpenHelper 类，就可以轻松的创建数据库。SQLiteOpenHelper 类根据开发应用程序的需要，封装了创建和更新数据库使用的逻辑。SQLiteOpenHelper 的子类，至少需要实现三个方法：

构造函数，调用父类 SQLiteOpenHelper 的构造函数。这个方法需要四个参数：上下文环境（例如，一个 Activity），数据库名字，一个可选的游标工厂（通常是 Null），一个代表正在使用的数据库模型版本的整数。

onCreate（）方法，它需要一个 SQLiteDatabase 对象作为参数，根据需要对这个对象填充表和初始化数据。

onUpgrage() 方法，它需要三个参数，一个 SQLiteDatabase 对象，一个旧的版本号和一个新的版本号，这样就可以清楚如何把一个数据库从旧的模型转变到新的模型。

下面示例代码展示了如何继承 SQLiteOpenHelper 创建数据库：

数据库存储在 data/< 项目文件夹 >/databases/ 下。

```
1  public class DatabaseHelper extends SQLiteOpenHelper {
2   DatabaseHelper(Context context, String name, CursorFactory cursorFactory, int version)
3   {
4      super(context, name, cursorFactory, version);
5       }
6
7       @Override
8       public void onCreate(SQLiteDatabase db) {
9           // TODO 创建数据库后，对数据库的操作
10      }
11
12      @Override
13  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
14          // TODO 更改数据库版本的操作
15      }
16
17  @Override
18  public void onOpen(SQLiteDatabase db) {
```

```
        super.onOpen(db);
        // TODO 每次成功打开数据库后首先被执行
    }
}
```

接下来讨论具体如何创建表、插入数据、删除表等等。调用 getReadableDatabase() 或 getWriteableDatabase() 方法，可以得到 SQLiteDatabase 实例，具体调用那个方法，取决于是否需要改变数据库的内容：

```
db=(new DatabaseHelper(getContext())).getWritableDatabase();
 return (db == null) ? false : true;
```

上面这段代码会返回一个 SQLiteDatabase 类的实例，使用这个对象，就可以查询或者修改数据库。

当完成了对数据库的操作（例如的 Activity 已经关闭），需要调用 SQLiteDatabase 的 Close() 方法来释放掉数据库连接。

创建表和索引

为了创建表和索引，需要调用 SQLiteDatabase 的 execSQL() 方法来执行 DDL 语句。如果没有异常，这个方法没有返回值。

例如，可以执行如下代码：

```
db.execSQL("CREATE TABLE mytable (_id INTEGER PRIMARY KEY
        AUTOINCREMENT, title TEXT, value REAL);");
```

这条语句会创建一个名为 mytable 的表，表有一个列名为 _id，并且是主键，这列的值是会自动增长的整数（例如，当插入一行时，SQLite 会给这列自动赋值），另外还有两列：title（字符）和 value（浮点数）。SQLite 会自动为主键列创建索引。

通常情况下，第一次创建数据库时创建了表和索引。如果不需要改变表的 schema，不需要删除表和索引 . 删除表和索引，需要使用 execSQL() 方法调用 DROP INDEX 和 DROP TABLE 语句。

给表添加数据

上面的代码，已经创建了数据库和表，现在需要给表添加数据。有两种方法可以给表添加数据。

像上面创建表一样，可以使用 execSQL() 方法执行 INSERT, UPDATE, DELETE 等语句来更新表的数据。execSQL() 方法适用于所有不返回结果的

SQL 语句。例如：

```
db.execSQL("INSERT INTO widgets (name, inventory)"+
"VALUES ('Sprocket', 5)");
```

另一种方法是使用 SQLiteDatabase 对象的 insert(), update(), delete() 方法。这些方法把 SQL 语句的一部分作为参数。示例如下：

```
ContentValues cv=new ContentValues();
cv.put(Constants.TITLE, "example title");
cv.put(Constants.VALUE, SensorManager.GRAVITY_DEATH_STAR_I);
db.insert("mytable", getNullColumnHack(), cv);
```

update（）方法有四个参数，分别是表名，表示列名和值的 ContentValues 对象，可选的 WHERE 条件和可选的填充 WHERE 语句的字符串，这些字符串会替换 WHERE 条件中的“？”标记。update() 根据条件，更新指定列的值，所以用 execSQL() 方法可以达到同样的目的。

WHERE 条件和其参数和用过的其他 SQL APIs 类似。例如：

```
String[] parms=new String[] {"this is a string"};
db.update("widgets", replacements, "name=?", parms);
```

delete() 方法的使用和 update() 类似，使用表名，可选的 WHERE 条件和相应的填充 WHERE 条件的字符串。

查询数据库

类似 INSERT, UPDATE, DELETE，有两种方法使用 SELECT 从 SQLite 数据库检索数据。

1 ．使用 rawQuery() 直接调用 SELECT 语句；

使用 query() 方法构建一个查询。

Raw Queries

正如 API 名字，rawQuery() 是最简单的解决方法。通过这个方法就可以调用 SQL SELECT 语句。例如：

```
Cursor c=db.rawQuery(
      "SELECT name FROM sqlite_master WHERE type='table' AND name='mytable'", null);
```

在上面例子中，我们查询 SQLite 系统表（sqlite_master）检查 table 表是否存在。返回值是一个 cursor 对象，这个对象的方法可以迭代查询结果。如果查询是动态的，使用这个方法就会非常复杂。例如，当需要查询的列在程序编译的时候不能确定，这时候使用 query() 方法会方便很多。

Regular Queries

query() 方法用 SELECT 语句段构建查询。SELECT 语句内容作为 query() 方法的参数，比如：要查询的表名，要获取的字段名，WHERE 条件，包含可选的位置参数，去替代 WHERE 条件中位置参数的值，GROUP BY 条件，HAVING 条件。

除了表名，其他参数可以是 null。所以，以前的代码段可以可写成：

```
String[] columns={"ID", "inventory"};
 String[] parms={"snicklefritz"};
 Cursor result=db.query("widgets", columns, "name=?",parms, null, null, null);
```

使用游标

不管如何执行查询，都会返回一个 Cursor，这是 Android 的 SQLite 数据库游标，使用游标，可以：

通过使用 getCount() 方法得到结果集中有多少记录；

通过 moveToFirst(), moveToNext(), 和 isAfterLast() 方法遍历所有记录；

通过 getColumnNames() 得到字段名；

通过 getColumnIndex() 转换成字段号；

通过 getString()，getInt() 等方法得到给定字段当前记录的值；

通过 requery() 方法重新执行查询得到游标；

通过 close() 方法释放游标资源；

例如，下面代码遍历 mytable 表

```
Cursor result=db.rawQuery("SELECT ID, name, inventory FROM mytable");
    result.moveToFirst();
    while (!result.isAfterLast()) {
        int id=result.getInt(0);
        String name=result.getString(1);
        int inventory=result.getInt(2);
        // do something useful with these
        result.moveToNext();
      }
      result.close();
```

在 Android 中使用 SQLite 数据库管理工具

在其他数据库上作开发，一般都使用工具来检查和处理数据库的内容，而不是仅仅使用数据库的 API。使用 Android 模拟器，有两种可供选择的方法来管理数据库。

首先，模拟器绑定了 sqlite3 控制台程序，可以使用 adb shell 命令来调用他。只要进入了模拟器的 shell，在数据库的路径执行 sqlite3 命令就可以了。数据库文件一般存放在：

/data/data/your.app.package/databases/your-db-name

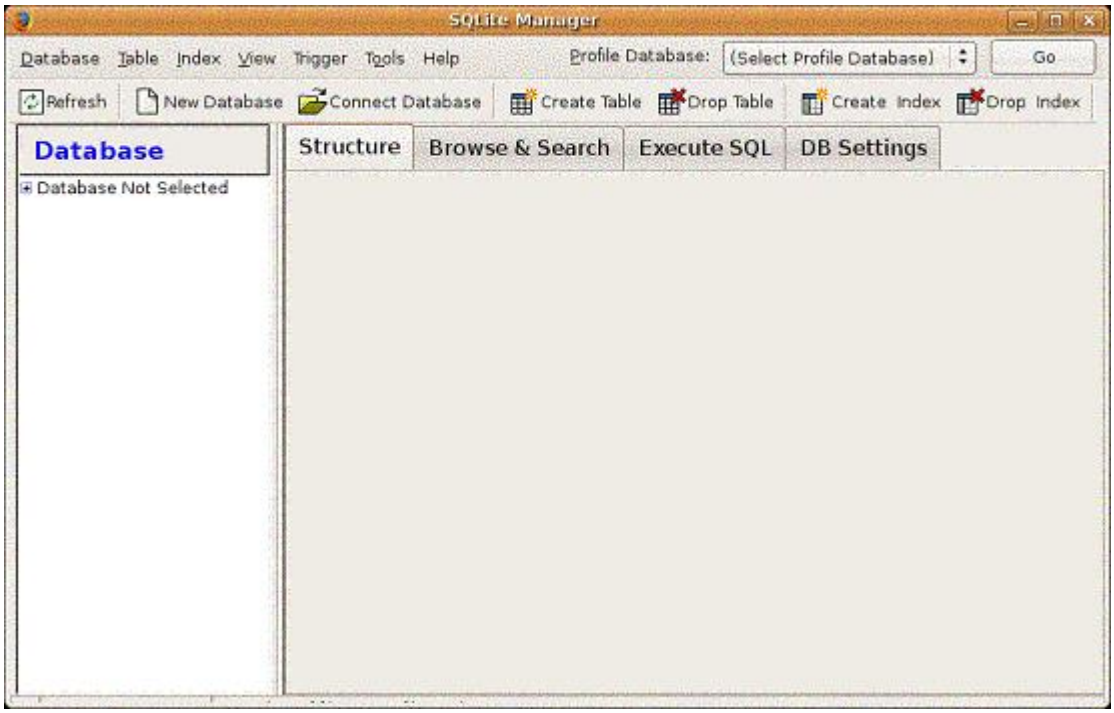
如果喜欢使用更友好的工具，可以把数据库拷贝到的开发机上，使用 SQLite-aware 客户端来操作它。这样的话，在一个数据库的拷贝上操作，

如果想要的修改能反映到设备上，需要把数据库备份回去。

把数据库从设备上考出来，可以使用 adb pull 命令（或者在 IDE 上做相应操作）。存储一个修改过的数据库到设备上，使用 adb push 命令。

一个最方便的 SQLite 客户端是 FireFox SQLite Manager 扩展，它可以跨所有平台使用。

图 2. SQLite Manager



结束语

如果想要开发 Android 应用程序，一定需要在 Android 上存储数据，使用 SQLite 数据库是一种非常好的选择。本文介绍了如何在 Android 应用程序中使用 SQLite 数据库 ，主要介绍了在 Android 应用程序中使用 SQLite 创建数据库和表、添加数据、更新和检索数据，还介绍了比较常用的 SQLite 管理工具，通过阅读本文，可以在 Android 中轻松操作 SQLite 数据库。