# 实验七：数据存取（二）

## 【实验目的】

1.学会使用 SQLite 数据库保存数据。

2.学会对 SQLite 数据库里的数据进行增删改查等操作。

## 【实验内容】

实现一个通讯录查看程序：

A） 要求使用 SQLite 数据库保存通讯录，使得每次运行程序均能显示当前联系人列表。

B） 主界面包含一个添加联系人按钮和一个联系人列表，每一项显示联系人学号，姓名，手机号码。

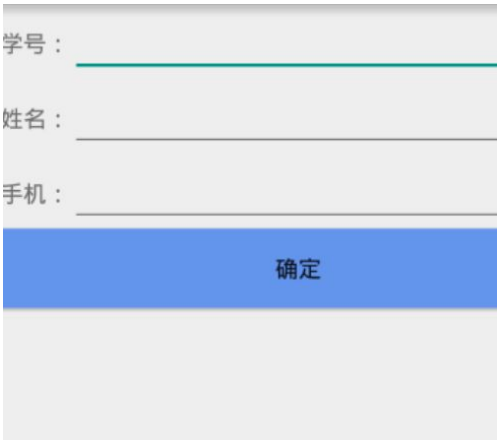
C） 点击添加联系人按钮能添加新的联系人。

D） 在次界面输入联系人信息之后点击确定按钮会返回主界面（如果输入的联系人信息有效的话，更新主界面的联系人列表）。

界面仅供参考

显示界面：


添加界面：



## 【实验参考】

使用 SQLiteOpenHelper 的子类能更方便实现要求：

1. 创建对象

```
public class MyDatabaseHelper extends SQLiteOpenHelper {
    private static final String DB_NAME = "Contacts.db";
    private static final String TABLE_NAME = "Contacts";
    private static final int DB_VERSION = 1;

    public MyDatabaseHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }
```

2. 创建数据库：直接执行创建数据库的 SQL 语句即可。

```
@Override
public void onCreate(SQLiteDatabase db) {
    String CREATE_TTABLE = "create table " + TABLE_NAME + "(_id integer primary key autoincrement, " + "_no text not null, _name text not null, _pnumber text);";
    db.execSQL(CREATE_TTABLE);
}
```

注意，SQL 语句最后的符号是分号。

3. 以下方法在本实验不需要用到，但 Android 要求重写才能实例化：

```
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXITS " + TABLE_NAME);
    onCreate(db);
}
```

4. 实现增加、更新和删除这 3 种操作有两种方法：

a) 可以用 execSQL 方法直接执行 SQL 语句：

b) 使用相应的 insert、 update 和 delete 方法：

i. Insert 方法需要使用 ContentValues：

```
public long insert(Contact entity) {
    SQLiteDatabase db = getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put("_no", entity.getNo());
    values.put("_name", entity.getName());
    values.put("_pnumber", entity.getPnumber());

    long id = db.insert(TABLE_NAME, null, values);
    db.close();
    return id;
}
```

ii. update 方法需要使用 ContentValues 和 Where 语句：

```
public int update(Contact entity) {
    SQLiteDatabase db = getWritableDatabase();
    String whereClause = "_no = ?";
    String[] whereArgs = {entity.getNo()};

    ContentValues values =  new ContentValues();
    values.put("_no", entity.getNo());
    values.put("_name", entity.getName());
    values.put("_pnumber", entity.getPnumber());

    int rows = db.update(TABLE_NAME, values, whereClause, whereArgs);
    db.close();
    return rows;
}
```

iii. delete 方法需要使用 Where 语句：

```
public int delete(Contact entity) {
    SQLiteDatabase db = getWritableDatabase();
    String whereClause = "_no = ?";
    String[] whereArgs = {entity.getNo()};

    int rows = db.delete(TABLE_NAME, whereClause, whereArgs);
    db.close();
    return rows;
}
```

5.实现查询操作可以使用 rawQuery 或者 query 函数，它们的区别类似于上面，前者直接执行 SQL 语句，后者是通过参数组合产生 SQL 语句：

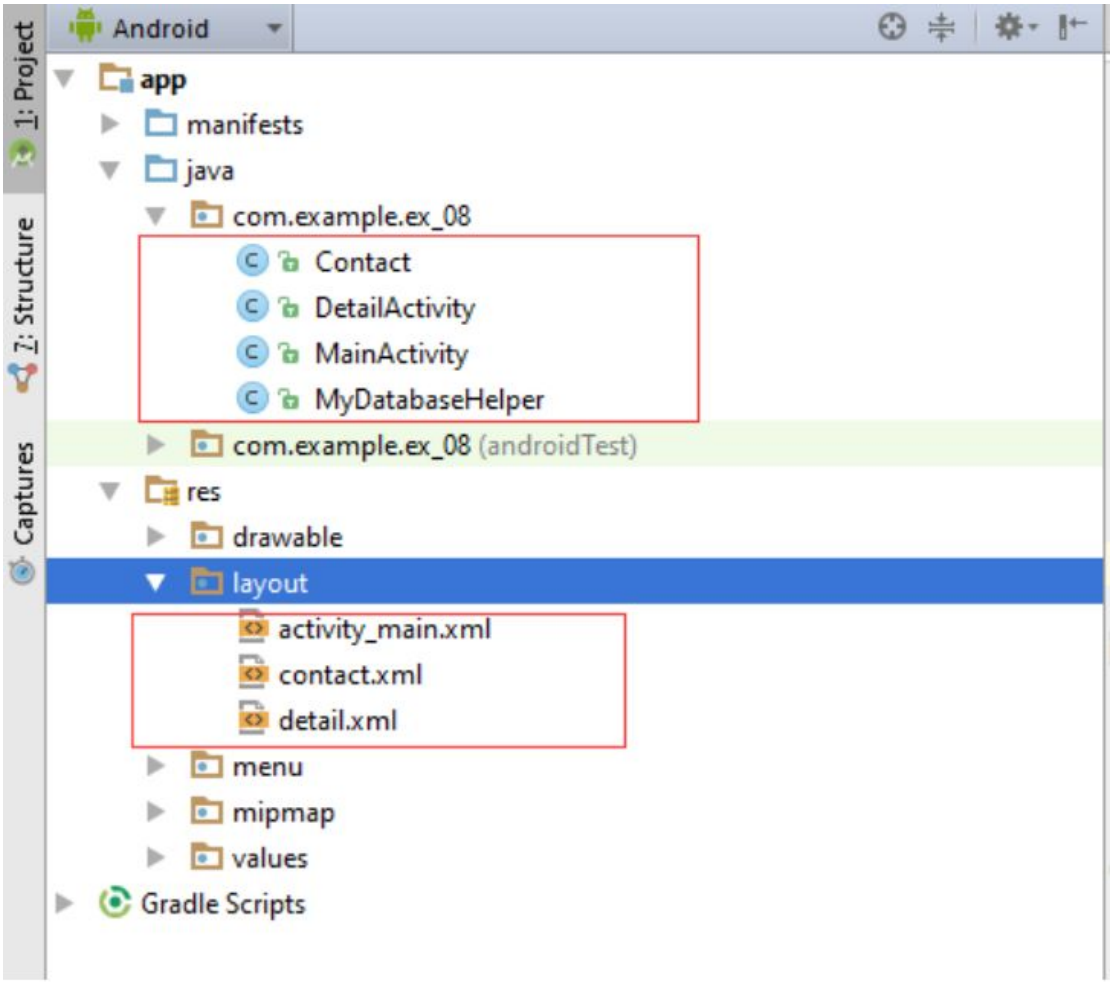
```
public Cursor query() {
    SQLiteDatabase db = getReadableDatabase();
    return db.query(TABLE_NAME, null, null, null, null, null, null);
}
```

rawQuery 或者 query 函数返回的都是 Cursor，关于 Cursor 类的详细介绍请看下面的链接：

http://www.cnblogs.com/TerryBlog/archive/2010/07/05/1771459.html

参考文件目录以及布局：

提示：Contact 是自定义的联系人类，包括学号、姓名、手机等信息及其访问方法。contact.xml 为 listView 中的 item 布局。




记得注册第二个活动：

```
    </activity>
    <activity android:name=".DetailActivity"></activity>
</application>
```

利用之前学到的对 ListView 的应用：

```
class MySimpleAdapter extends SimpleAdapter {
    private ArrayList<Map<String, String>> mData;
    public MySimpleAdapter(Context context, List<? extends Map<String, ?>> data, int resource, String[] from, int[] to) {
        super(context, data, resource, from, to);
        this.mData = (ArrayList<Map<String, String>>)data;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        final int mPosition = position;
        return super.getView(position, convertView, parent);
    }
}
```

【提示】也可以直接使用 SimpleAdapter。可参考：

http://blog.csdn.net/xing1716263268/article/details/7912665

```
public void setData(List<Map<String, String>> mDataList) {
    Map<String, String> mData;
    Cursor c = myDatabaseHelper.query();
    while (c.moveToNext()) {
        mData = new HashMap<String, String>();
        mData.put("no", c.getString(c.getColumnIndex("_no")));
        mData.put("name", c.getString(c.getColumnIndex("_name")));
        mData.put("pNumber", c.getString(c.getColumnIndex("_pnumber")));
        mDataList.add(mData);
    }
}
```

## 【扩展】

1.长按某个联系人，弹出对话框，询问是否删除该联系人。

2.单击某个联系人，弹出修改联系人的对话框。

关于自定义对话框，参考链接：

http://blog.csdn.net/it1039871366/article/details/9796965