

实施方案 数据操作之增删改查

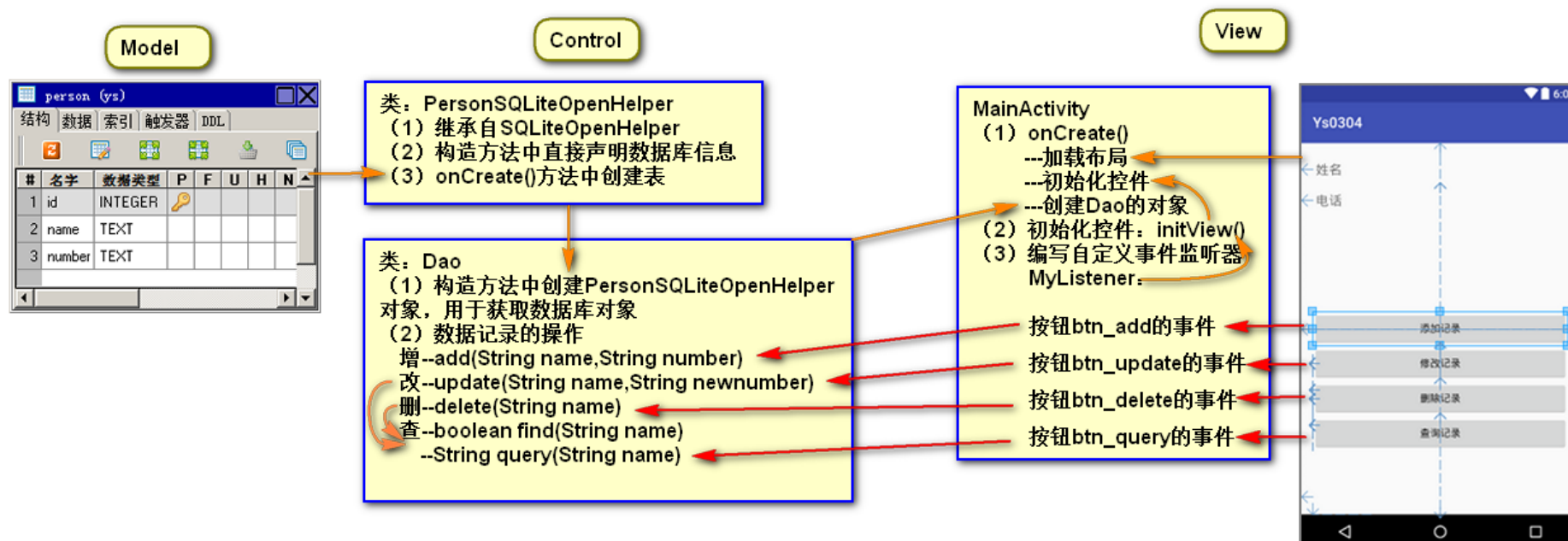
=====任务描述=====

通讯录：用于保存姓名及电话号码，可以添加、删除、修改及查询记录。

- (1) 输入姓名、电话，点击添加记录按钮，可以向数据表中插入记录，并在底部显示状态信息“添加记录成功”。
- (2) 输入姓名，点击添查询记录按钮，可以从数据表中查找该人的电话，查找到的电话显示在电话编辑框内，若没找到该人记录则在底部显示状态信息“XXX不存在”。
- (3) 查询到记录后，单击删除记录按钮可以删除当前显示的记录并在底部显示信息“删除记录成功”。
- (4) 查询到记录后，在电话编辑框中输入新的电话号码后点击修改记录可以更新当前记录，并在底部显示信息“修改记录成功”。
- (5) 【拓展】添加记录时判断数据表中是否已存在该人，若不存在则添加记录成功，若存在则在底部显示信息“XXX已存在，添加记录失败”。
- (6) 【拓展】修改记录和删除记录按钮初始为不可用状态，当查询记录成功后方可点击。修改、删除之后按钮重置为不可用状态。



=====项目解析=====



=====关键技术=====

数据库对象SQLiteDateBase db

建表、记录的增删改: db.execSQL(sql语句, 条件参数);

记录的查询: Cursor db.rawQuery(sql语句, 条件参数);

1

增加记录:

记录编号 insert(表名, null, 数据集);
Long insert(String,null,ContentValues);

删除记录:

记录条数 delete(表名, 删除条件, 删除条件参数);
int delete(String, String, String[])

更新记录:

记录条数 update(表名, 数据集, 更新条件, 更新条件参数);
int update(String, ContentValues, String, String[]);

查询记录:

游标 query(表名, 字段名列表, 查询条件, 查询条件参数,
分组, 分组条件, 排序);
Cursor query(String, String[], String, String[],
String, String, String);

数据集: ContentValues
(1) 用法同Map集合
(2) key--表的字段名
value--值

2

=====项目实施=====

步骤1、显示模块设计

```
MainActivity.java x
11 public class MainActivity extends AppCompatActivity {
12     private EditText edtName, edtNumber; //姓名、电话编辑框
13     private Button btnAdd, btnUpdate, btnDelete, btnQuery; //添加记录、修改记录、删除记录、查询记录按钮
14     private TextView tvStatus; //状态文本显示框
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19         initView();
20     }
21     //初始化控件
22     private void initView() {
23         //-----获取界面组件-----
24         edtName=(EditText)findViewById(R.id.edt_name);
25         edtNumber=(EditText)findViewById(R.id.edt_number);
26         btnAdd=(Button)findViewById(R.id.btn_add);
27         btnUpdate=(Button)findViewById(R.id.btn_update);
28         btnDelete=(Button)findViewById(R.id.btn_delete);
29         btnQuery=(Button)findViewById(R.id.btn_query);
30         tvStatus=(TextView)findViewById(R.id.tv_status);
31         //-----为按钮添加事件监听-----
32         MyListener listener=new MyListener();
33         btnAdd.setOnClickListener(listener);
34         btnUpdate.setOnClickListener(listener);
35         btnDelete.setOnClickListener(listener);
36         btnQuery.setOnClickListener(listener);
37     }
38     //集中管理各按钮的事件监听
39     private class MyListener implements OnClickListener{
40         @Override
41         public void onClick(View v) {
42             switch (v.getId()){
43                 case R.id.btn_add: //添加记录
44                     break;
45                 case R.id.btn_update: //更新记录
46                     break;
47                 case R.id.btn_delete: //删除记录
48                     break;
49                 case R.id.btn_query: //查询记录
50                     break;
51                 default:
52             }
53         }
54     }
55 }
56 }
57 }
58 }
59 }
```

步骤2、控制模块设计

```
PersonSQLiteOpenHelper.java x
7 public class PersonSQLiteOpenHelper extends SQLiteOpenHelper {
8     public PersonSQLiteOpenHelper(Context context) {
9         super(context, "person.db", null, 1);
10        //直接在构造方法中定义数据库的名称、版本信息
11    }
12    @Override
13    public void onCreate(SQLiteDatabase db) {
14        db.execSQL("create table person(id integer primary key autoincrement, name,number)");
15    }
16    @Override
17    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {...}
18 }

Dao.java x
7 /*
8  * 数据库操作类，封装记录的增删改查操作
9  * Created by Administrator on 2017/4/1.
10  */
11 public class Dao {
12     private PersonSQLiteOpenHelper helper;
13
14     public Dao(Context context) {
15         helper=new PersonSQLiteOpenHelper(context);
16        //创建Dao的对象时，同步创建helper对象以备使用
17    }
18
19    /**
20     *将姓名、电话号码添加到person表中
21     */
22     public long add(String name,String number){
23         SQLiteDatabase db=helper.getWritableDatabase(); //连接数据库
24         ContentValues values=new ContentValues();
25         values.put("name",name);
26         values.put("number",number);
27         //key-value: 参数1(key)—表的字段名; 参数2(value)—调用方法时提供的实参，要添加到表中的值
28         Long id=db.insert("person",null,values);
29         db.close(); //关闭数据库
30         return id;
31     }
32 }
```

步骤3、添加记录业务实现

The image shows two Java files in an IDE. The left file, `MainActivity.java`, contains a `MyListener` class that implements `OnClickListener`. It has a `switch` statement with cases for `btn_add`, `btn_update`, `btn_delete`, and `btn_query`. The `btn_add` case calls `dao.add(personName, personNumber)` and then `reset("添加记录成功")`. The right file, `Dao.java`, contains a `Dao` class with a `helper` of type `PersonSQLiteOpenHelper`. It has an `add` method that connects to the database, inserts a new record into the `person` table, and returns the generated ID. Annotations include a yellow box labeled "添加记录方法定义" pointing to the `add` method in `Dao.java`, and a yellow box labeled "添加记录方法使用" pointing to the `dao.add` call in `MainActivity.java`. A red arrow points from the `dao.add` call to the `add` method definition. A blue arrow points from the `reset` call in `MainActivity.java` to the `reset` method definition in the same file.

```
41 //集中管理各按钮的事件监听
42 private class MyListener implements OnClickListener{
43     @Override
44     public void onClick(View v) {
45         switch (v.getId()) {
46             case R.id.btn_add: //添加记录
47                 personName=edtName.getText().toString();
48                 personNumber=edtNumber.getText().toString();
49                 dao.add(personName, personNumber);
50                 reset("添加记录成功");
51                 break;
52             case R.id.btn_update: //更新记录
53                 break;
54             case R.id.btn_delete: //删除记录
55                 break;
56             case R.id.btn_query: //查询记录
57                 break;
58             default:
59                 break;
60         }
61     }
62 }
63
64 //界面状态重置
65 private void reset(String msg){
66     tvStatus.setText(msg); //状态显示框显示提示信息（操作结果）
67     edtName.setText(""); //清空编辑框
68     edtNumber.setText("");
69     personName=null; //清空字符串
70     personNumber=null;
71 }
72
73
74
```

```
12 public class Dao {
13     private PersonSQLiteOpenHelper helper;
14
15     public Dao(Context context) { helper=new PersonSQLiteOpenHelper(context); }
16
17     //添加记录--将姓名、电话号码组成一条记录插入person表
18     public Long add(String name,String number) {
19         //-----连接数据库-----
20         SQLiteDatabase db=helper.getWritableDatabase();
21         //-----操作数据记录-----
22         //db.execSQL("insert table person values(null,?,?)",new String[]{name,number});
23         ContentValues values=new ContentValues();
24         //ContentValues用法同Map集合，key--表的字段名，value--具体的字段值
25         values.put("name",name);
26         values.put("number",number);
27         Long id=db.insert("person",null,values);
28         //参数1--表名，参数2--null，参数3--ContentValues类型的对象，封装“字段-值”的键值对
29         //-----关闭数据库-----
30         db.close();
31         return id;
32     }
33
34     //修改记录--通过姓名查找记录，修改指定人对应的电话号码
35     public void update(String name,String newnumber){...}
36
37     //删除记录--通过姓名查找记录，删除指定人的信息
38     public void delete(String name){...}
39
40     //查找记录--通过姓名查找记录，判断是否存在指定人
41     public boolean find(String name){...}
42
43     //查找记录--查找姓名对应的电话号码
44     public String query(String name){...}
45 }
```

添加记录方法定义

添加记录方法使用

步骤4、查询记录业务实现

```
MainActivity.java x Dao.java x

11 public class MainActivity extends AppCompatActivity {
12     private EditText edtName, edtNumber; //姓名、电话编辑框
13     private Button btnAdd, btnUpdate, btnDelete, btnQuery; //添加记录、修改记录、删除记录、查询记录
14     private TextView tvStatus; //状态文本显示框
15     private String personName, personNumber; //保存姓名、电话
16     private Dao dao; //数据库操作服务
17     @Override
18     protected void onCreate(Bundle savedInstanceState) {...}
19     //初始化控件
20     private void initView() {...}
21     //集中管理各按钮的事件监听
22     private class MyListener implements OnClickListener {
23         @Override
24         public void onClick(View v) {
25             switch (v.getId()) {
26                 case R.id.btn_add: //添加记录
27                     ...
28                     break;
29                 case R.id.btn_update: //更新记录
30                     ...
31                     break;
32                 case R.id.btn_delete: //删除记录
33                     ...
34                     break;
35                 case R.id.btn_query: //查询记录
36                     personName = edtName.getText().toString();
37                     if (dao.find(personName)) {
38                         personNumber = dao.query(personName);
39                         edtNumber.setText(personNumber);
40                     }
41                     else {
42                         reset(personName + "不存在");
43                     }
44                     break;
45                 default:
46                     ...
47             }
48         }
49     }
50     //界面状态重置
51     private void reset(String msg) {...}
52 }

13 public class Dao {
14     private PersonSQLiteOpenHelper helper;
15     public Dao(Context context) { helper = new PersonSQLiteOpenHelper(context); }
16     //添加记录--将姓名、电话号码组成一条记录插入person表
17     public Long add(String name, String number) {...}
18     //修改记录--通过姓名查找记录，修改指定人对应的电话号码
19     public void update(String name, String newnumber) {...}
20     //删除记录--通过姓名查找记录，删除指定人的信息
21     public void delete(String name) {...}
22     //查找记录--通过姓名查找记录，判断是否存在指定人
23     public boolean find(String name) {
24         SQLiteDatabase db = helper.getReadableDatabase();
25         Cursor cursor = db.query("person", null, "name=?", new String[] {name}, null, null, null);
26         //参数1-表名 (from子句)，参数2-要查询的字段列表数组 (select子句)
27         //参数3、4-查询条件：参数3是带? 占位符的字符串式条件表达式，参数4是? 对应的值变量列表 (数组)
28         //参数5、6、7-分组、分组条件、排序
29         boolean result = cursor.moveToNext(); //将游标移动到第1条记录，判断当前是否有记录
30         cursor.close();
31         db.close();
32         return result;
33     }
34     //查找记录--查找姓名对应的电话号码
35     public String query(String name) {
36         String number = null;
37         SQLiteDatabase db = helper.getReadableDatabase();
38         Cursor cursor = db.query("person", null, "name=?", new String[] {name}, null, null, null);
39         //参数1-表名 (from子句)，参数2-要查询的字段列表数组 (select子句)
40         //参数3、4-查询条件：参数3是带? 占位符的字符串式条件表达式，参数4是? 对应的值变量列表 (数组)
41         //参数5、6、7-分组、分组条件、排序
42         if (cursor.moveToNext()) //将游标移动到第1条记录，判断当前是否有记录
43             number = cursor.getString(cursor.getColumnIndex("number"));
44         cursor.close();
45         db.close();
46         return number;
47     }
48 }
```

步骤5、更新删除记录业务实现

The image shows two Java files in an IDE: `MainActivity.java` and `Dao.java`.

MainActivity.java (lines 45-88):

```
45 private class MyListener implements View.OnClickListener {
46     private String personName; //保存姓名
47     private String personNumber; //保存电话号码
48     @Override
49     public void onClick(View v) {
50         switch (v.getId()) {
51             case R.id.btn_add: //添加记录
52                 ...
57             case R.id.btn_update: //更新记录
58                 personName=edtName.getText().toString();
59                 personNumber=edtNumber.getText().toString();
60                 if(dao.find(personName)){
61                     dao.update(personName, personNumber);
62                     reset("更新记录成功");
63                 }
64                 else
65                     reset(personName+"不存在");
66                 break;
67             case R.id.btn_delete: //删除记录
68                 personName=edtName.getText().toString();
69                 if(dao.find(personName)){
70                     dao.delete(personName);
71                     reset("删除记录成功");
72                 }
73                 else
74                     reset(personName+"不存在");
75                 break;
76             case R.id.btn_query: //查询记录
77                 ...
85             default:
86
87         }
88     }
89 }
```

Dao.java (lines 12-72):

```
12 public class Dao {
13     private PersonSQLiteOpenHelper helper;
14     public Dao(Context context) {...}
18     //将姓名、电话号码组成一条记录添加到person表中
19     public long add(String name, String number) {...}
32     //查询指定人（姓名）的电话号码
33     public String query(String name) {...}
47     //通过姓名查询指定记录是否存在
48     public boolean find(String name) {
49         SQLiteDatabase db=helper.getWritableDatabase();
50         Cursor cursor=db.query("person", null, "name=?", new String[] {name}, null, null, null);
51         boolean result=cursor.moveToNext();
52         cursor.close();
53         db.close();
54         return result;
55     }
56     //删除指定姓名的记录
57     public int delete(String name) {
58         SQLiteDatabase db=helper.getWritableDatabase();
59         int n=db.delete("person", "name=?", new String[] {name});
60         db.close();
61         return n;
62     }
63     //更新，将指定人（姓名）的电话号码修改为新输入的电话号码
64     public int update(String name, String newNumber) {
65         SQLiteDatabase db=helper.getWritableDatabase();
66         ContentValues values=new ContentValues();
67         values.put("number", newNumber);
68         int n=db.update("person", values, "name=?", new String[] {name});
69         db.close();
70         return n;
71     }
72 }
```

Annotations:

- A yellow box with the text "先确认是否存在指定记录" (First confirm if the specified record exists) has arrows pointing to the `if(dao.find(personName))` checks in both the `update` and `delete` methods of `MainActivity.java`.
- Red circles highlight the `dao.update(personName, personNumber)` and `dao.delete(personName)` calls in `MainActivity.java`.

Comments at the bottom of MainActivity.java:

- Line 88: `//执行操作后的状态重置`

=====代码清单=====

PersonSQLiteOpenHelper.java

```
public class PersonSQLiteOpenHelper extends SQLiteOpenHelper {
    public PersonSQLiteOpenHelper(Context context) {
        super(context, "person.db", null, 1);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table person(id integer primary key autoincrement,name,number)");
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    }
}
```

Dao.java

```
public class Dao {
    private PersonSQLiteOpenHelper helper;
    public Dao(Context context) {
        helper=new PersonSQLiteOpenHelper(context);
    }
    public long add(String name,String number){
        SQLiteDatabase db=helper.getWritableDatabase();
        ContentValues values=new ContentValues();
        values.put("name",name);
        values.put("number",number);
        Long id=db.insert("person",null,values);
        db.close();
        return id;
    }
    public int update(String name,String newnumber){
        SQLiteDatabase db=helper.getWritableDatabase();
        ContentValues values=new ContentValues();
        values.put("number",newnumber);
        int number=db.update("person",values,"name=?",new String[]{name});
        db.close();
        return number;
    }
    public int delete(String name){
        SQLiteDatabase db=helper.getWritableDatabase();
        int number=db.delete("person","name=?",new String[]{name});
        db.close();
        return number;
    }
}
```

```

public boolean find(String name) {
    SQLiteDatabase db=helper.getWritableDatabase();
    Cursor cursor=db.query("person",null,"name=?",new String[]{name},null,null,null);
    boolean result=cursor.moveToNext();
    cursor.close();
    db.close();
    return result;
}

public String query(String name) {
    SQLiteDatabase db=helper.getWritableDatabase();
    Cursor cursor=db.query("person",null,"name=?",new String[]{name},null,null,null);
    cursor.moveToNext();
    int index=cursor.getColumnIndex("number");
    String number=cursor.getString(index);
    return number;
}
}

```

MainActivity.java

```

public class MainActivity extends AppCompatActivity {
    private EditText edtName,edtNumber;
    private Button btnAdd,btnUpdate,btnDelete,btnFind;
    private TextView tvStatus;
    private Dao dao;
    private String personName,personNumber;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        initView();
        dao=new Dao(this);
    }
    /**
     * 初始化控件
     */
    private void initView() {
        edtName=(EditText)findViewById(R.id.edt_name);
        edtNumber=(EditText)findViewById(R.id.edt_number);
        btnAdd=(Button)findViewById(R.id.btn_add);
        btnUpdate=(Button)findViewById(R.id.btn_update);
        btnDelete=(Button)findViewById(R.id.btn_delete);
        btnFind=(Button)findViewById(R.id.btn_find);
        tvStatus=(TextView)findViewById(R.id.tv_status);
        btnAdd.setOnClickListener(new MyListener());
        btnUpdate.setOnClickListener(new MyListener());
    }
}

```

```

        btnDelete.setOnClickListener(new MyListener());
        btnFind.setOnClickListener(new MyListener());
    }
    /**
     * 各按钮的事件监听
     */
    private class MyListener implements View.OnClickListener{
        @Override
        public void onClick(View v) {
            int i;
            switch (v.getId()) {
                case R.id. btn_add:
                    personName=edtName.getText().toString();
                    personNumber=edtNumber.getText().toString();
                    dao.add(personName, personNumber);
                    reset("添加记录成功");
                    break;
                case R.id. btn_update:
                    personName=edtName.getText().toString();
                    personNumber=edtNumber.getText().toString();
                    if(dao.find(personName)) {
                        i = dao.update(personName, personNumber);
                        reset("更新记录成功");
                    }
                    else
                        reset(personName+"不存在");
                    break;
                case R.id. btn_delete:
                    personName=edtName.getText().toString();
                    if(dao.find(personName)) {
                        i = dao.delete(personName);
                        reset("删除记录成功");
                    }
                    else
                        reset(personName+"不存在");
                    break;
                case R.id. btn_find:
                    personName=edtName.getText().toString();
                    if(dao.find(personName)) {
                        personNumber = dao.query(personName);
                        edtNumber.setText(personNumber);
                    }
                    else
                        reset(personName+"不存在");
            }
        }
    }

```

```

        break;
    default:
    }
}

private void reset(String msg) {
    tvStatus.setText(msg);
    edtName.setText("");
    edtNumber.setText("");
    personName=null;
    personNumber=null;
}
}
}

```

activity_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.jvtc.ad.y0304.MainActivity">
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:ems="10"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:id="@+id/edt_name"
        android:hint="姓名" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:ems="10"
        android:layout_below="@+id/edt_name"
        android:layout_alignParentLeft="true"

```

```

        android:layout_alignParentStart="true"
        android:id="@+id/edt_number"
        android:hint="电话" />
<Button
    android:text="添加记录"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerVertical="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:id="@+id/btn_add" />
<Button
    android:text="修改记录"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/btn_add"
    android:layout_centerHorizontal="true"
    android:id="@+id/btn_update" />
<Button
    android:text="删除记录"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/btn_update"
    android:layout_alignLeft="@+id/btn_update"
    android:layout_alignStart="@+id/btn_update"
    android:id="@+id/btn_delete" />
<Button
    android:text="查询记录"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/btn_delete"
    android:layout_alignLeft="@+id/btn_delete"
    android:layout_alignStart="@+id/btn_delete"
    android:id="@+id/btn_find" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:id="@+id/tv_status" />
</RelativeLayout>

```