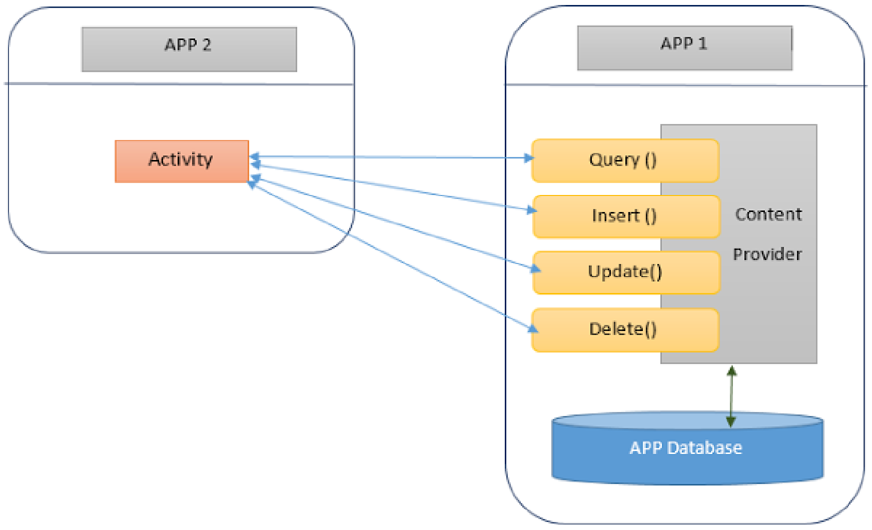
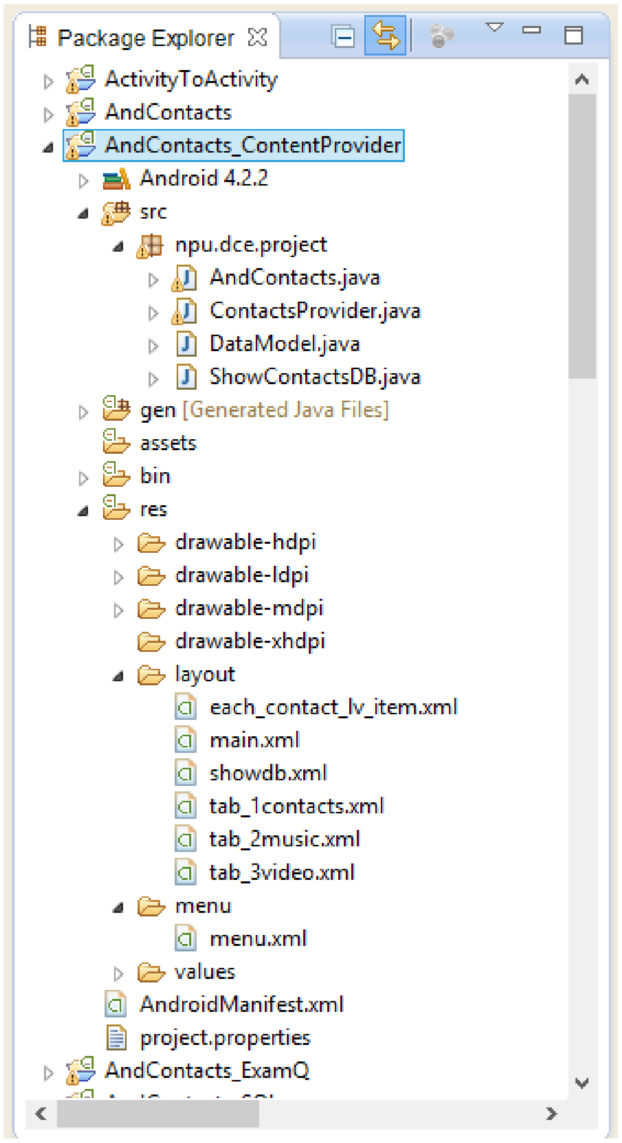
Manisha Vyas

Question :

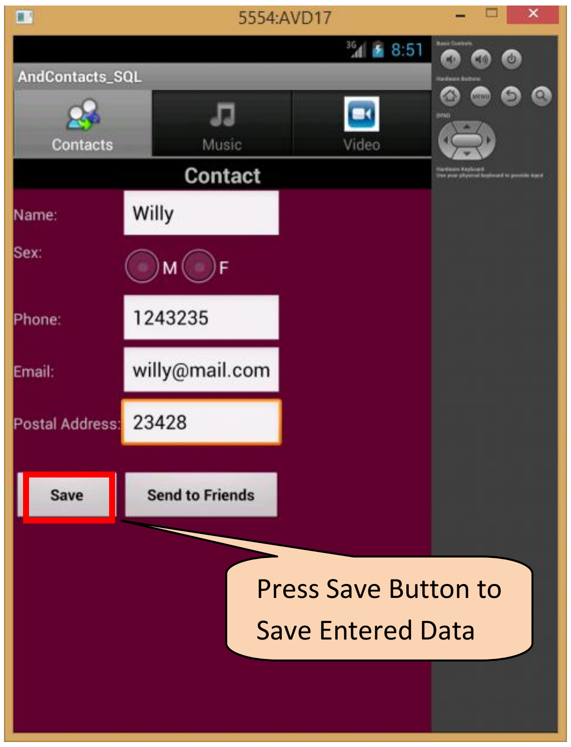
Continue the Tab Activity so that the data entered by the users will be saved in a data base. The database is then used as a content provider.

- An Activity will then act as a content resolver to interact with the content provider to retrieve the data from the databse.
- You need to demonstrate database operation that you can show at least more than 1 recorded data (a list of contacts), as well as it should have more than 1 field (column in database table) being retrieved (such that name, phone, email, and address). Please
- it can be used inside other activity too!



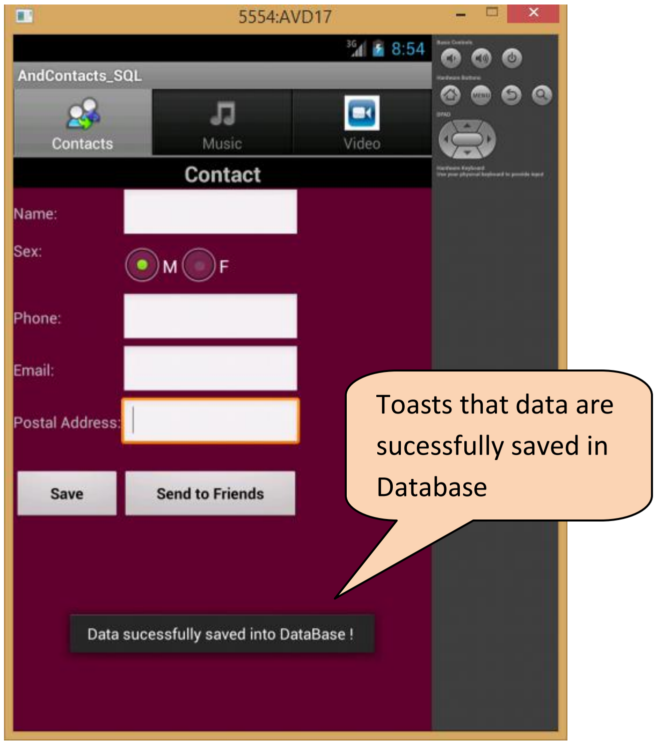

Package Explorer

```
▷ ActivityToActivity
▷ AndContacts
◢ AndContacts_ContentProvider
    ▷ Android 4.2.2
    ◢ src
        ◢ npu.dce.project
            ▷ AndContacts.java
            ▷ ContactsProvider.java
            ▷ DataModel.java
            ▷ ShowContactsDB.java
    ▷ gen [Generated Java Files]
      assets
    ▷ bin
    ◢ res
        ▷ drawable-hdpi
        ▷ drawable-ldpi
        ▷ drawable-mdpi
          drawable-xhdpi
        ◢ layout
              each_contact_lv_item.xml
              main.xml
              showdb.xml
              tab_1contacts.xml
              tab_2music.xml
              tab_3video.xml
        ◢ menu
              menu.xml
        ▷ values
      AndroidManifest.xml
      project.properties
▷ AndContacts_ExamQ
```



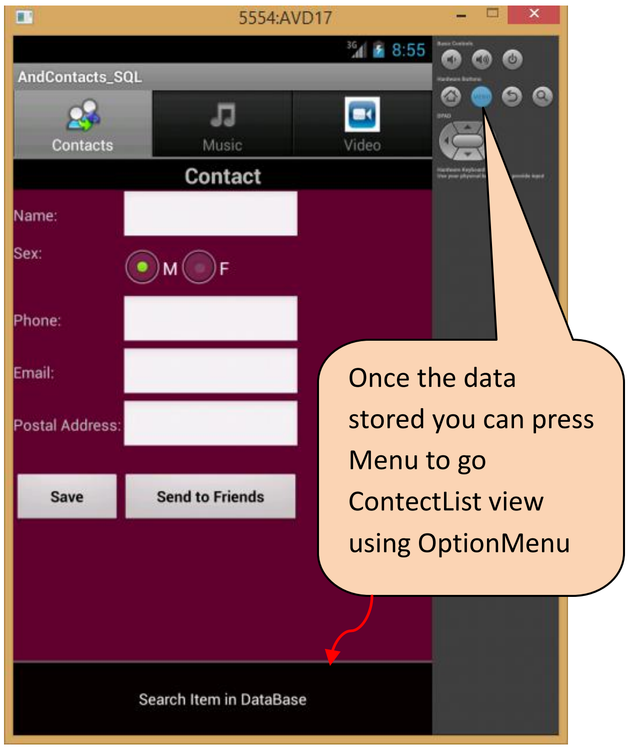
Name: Willy
Sex: M F
Phone: 1243235
Email: willy@mail.com
Postal Address: 23428

Press Save Button to Save Entered Data

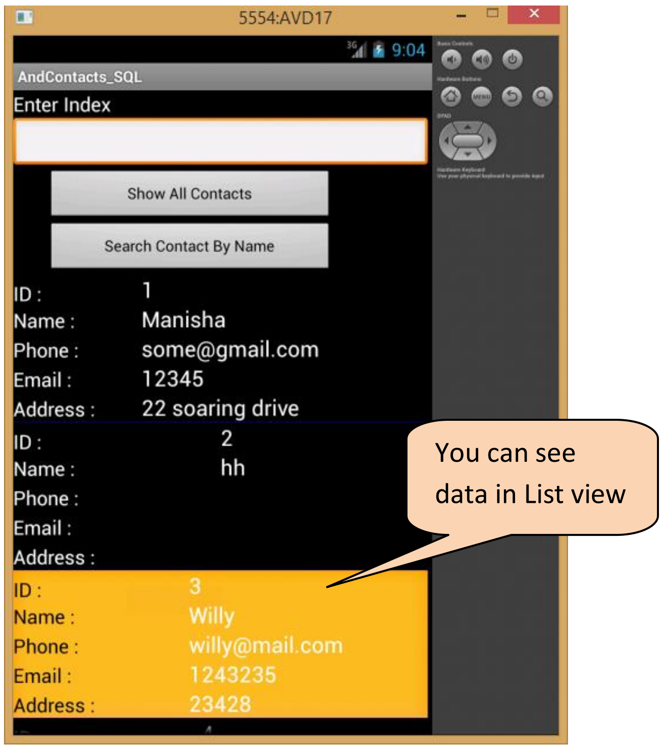


Toasts that data are sucessfully saved in Database

Data sucessfully saved into DataBase !

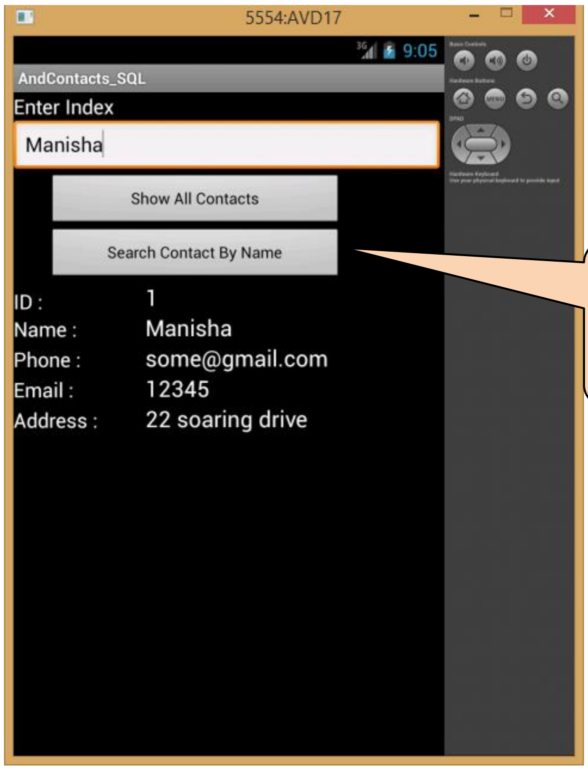


Once the data stored you can press Menu to go ContectList view using OptionMenu



ID : 1
Name : Manisha
Phone : some@gmail.com
Email : 12345
Address : 22 soaring drive
ID : 2
Name : hh
Phone :
Email :
Address :
ID : 3
Name : Willy
Phone : willy@mail.com
Email : 1243235
Address : 23428

You can see data in List view



```
ID :          1
Name :        Manisha
Phone :       some@gmail.com
Email :       12345
Address :     22 soaring drive
```

You can also search Content Provider using Name as search field.

//Android.javax

```
package npu.dce.project;

import android.os.Bundle;
import android.app.~~TabActivity~~;
import android.content.ContentResolver;
import android.content.ContentValues;
import android.content.Intent;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TabHost;
import android.widget.Toast;

public class AndContacts extends ~~TabActivity~~ {

	private static final int SHOW_CONTACTS = 0;
	private Button saveb,cancelb;
	private EditText txtname,txtemail,txtphone,txtpostaladd;
	private String strName,strEmail,strPhone,strPostalAdd;

	//private ContactsProvider myDBAdapter;
	TabHost mTabHost = null;

	public AndContacts() {
		// TODO Auto-generated constructor stub
	}
	@Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        mTabHost = getTabHost();

        mTabHost.addTab(mTabHost.newTabSpec("tab_test1").setIndicator("Contacts",
getResources().getDrawable(R.drawable.contact)).setContent(R.id.contactsLayout));
        mTabHost.addTab(mTabHost.newTabSpec("tab_test2").setIndicator("Music",
getResources().getDrawable(R.drawable.music)).setContent(R.id.musicLayout));
        mTabHost.addTab(mTabHost.newTabSpec("tab_test3").setIndicator("Video",
getResources().getDrawable(R.drawable.video)).setContent(R.id.videoLayout));

        mTabHost.setCurrentTab(0);
       saveb = (Button) findViewById(R.id.buttonsave);
       cancelb = (Button) findViewById(R.id.buttoncancel);

       txtname = (EditText) findViewById(R.id.txtname);
       txtemail = (EditText) findViewById(R.id.txtemail);
       txtphone = (EditText) findViewById(R.id.txtphone);
       txtpostaladd = (EditText) findViewById(R.id.txtpostaladdress);

       //myDBAdapter = new ContactsProvider(this);
       //myDBAdapter.open();

       //myDBAdapter.deleteAllEntries();


       saveb.setOnClickListener(new View.OnClickListener() {
			@Override
			public void onClick(View v) {

				strName = txtname.getText().toString();
			strEmail = txtemail.getText().toString();
			strPhone = txtphone.getText().toString();
			strPostalAdd = txtpostaladd.getText().toString();

				DataModel dataModel = new
DataModel(strName,strEmail,strPhone,strPostalAdd);
		     // myDBAdapter.insertEntry(newContact);
		      //updateArray();
		      addNewContact(dataModel);

		      txtname.setText("");
		      txtemail.setText("");
		      txtphone.setText("");
		      txtpostaladd.setText("");
```

```
                   Toast.makeText(AndContacts.this,"Data sucessfully saved into DataBase
! ",Toast.LENGTH_LONG).show();
                    }
             });


        cancelb.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {

                           setResult(RESULT_CANCELED, null);
                           finish();
                    }
             });
        }

         private void addNewContact(DataModel dataModel) {
            ContentResolver cr = getContentResolver();
            if (cr.query(ContactsProvider.CONTENT_URI, null,null, null,
null).getCount()==0)
            {
              ContentValues values = new ContentValues();

              values.put(ContactsProvider.KEY_NAME,dataModel.getName());
              values.put(ContactsProvider.KEY_PHONE, dataModel.getPhone());
              values.put(ContactsProvider.KEY_EMAIL, dataModel.getEmail());
              values.put(ContactsProvider.KEY_POSTALADDR, dataModel.getPostaladdr());

              cr.insert(ContactsProvider.CONTENT_URI, values);

            }
         }



       public boolean onCreateOptionsMenu(Menu menu)
    {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.menu, menu);
        return true;
    }
       public boolean onOptionsItemSelected(MenuItem item) {
           super.onOptionsItemSelected(item);

           switch (item.getItemId())
           {
           case R.id.filter_name:
               Intent i = new Intent(this, ShowContactsDB.class);
               startActivityForResult(i, SHOW_CONTACTS);
               return true;

           default:
               return super.onOptionsItemSelected(item);
           }
         }
```

```
}
```

//ShowContactDB.java

```
package npu.dce.project;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.ContentResolver;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;

public class ShowContactsDB extends Activity
{
	private Button showall,showbyid;
	private EditText txt_searchbox;
	private ContentResolver cr;
	private ListView lv;
	private Cursor cursor = null;
	private SimpleCursorAdapter dataAdapter;

	@Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.showdb);

	showbyid = (Button) findViewById(R.id.btn_showbyid);
	showall = (Button) findViewById(R.id.btn_showall);

	lv = (ListView)findViewById(R.id.lv_contacts);

	 cr= getContentResolver();

		displayListView("ALL");

	 showbyid.setOnClickListener(new View.OnClickListener() {
			@Override
			public void onClick(View v) {
				txt_searchbox = (EditText) findViewById(R.id.txt_searchbox);
				displayListView(txt_searchbox.getText().toString());
			}
		});

	showall.setOnClickListener(new View.OnClickListener() {
			@Override
			public void onClick(View v) {
```

```
                          displayListView("ALL");
                    }
              });
    }

      @SuppressLint("NewApi")
      private void displayListView(String whichView)
      {
             if (whichView.equals("ALL"))//which view -> all Contacts
             {
                    cursor = cr.query(ContactsProvider.CONTENT_URI, null, null, null, 
null);
             }
             else //which view -> by Contact Name
             {
                    cursor = cr.query(ContactsProvider.CONTENT_URI, null, 
ContactsProvider.KEY_NAME + " = '" + whichView + "'", null, null);
             }

               // The desired columns to be bound
               String[] columns = new String[] {
                 ContactsProvider.KEY_ID,
                 ContactsProvider.KEY_NAME,
                 ContactsProvider.KEY_PHONE,
                 ContactsProvider.KEY_EMAIL,
                 ContactsProvider.KEY_POSTALADDR
               };

               // the XML defined views which the data will be bound to
               int[] to = new int[] {
                 R.id.lv_id,
                 R.id.lv_name,
                 R.id.lv_phone,
                 R.id.lv_email,
                 R.id.lv_add
               };

        dataAdapter = new SimpleCursorAdapter(this, 
R.layout.each_contact_lv_item,cursor, columns, to,0);
        lv.setAdapter(dataAdapter);

       }
}
```

//ContactProvider.java

```
package npu.dce.project;

//npu.dce.project.ContactsProvider

import android.content.ContentProvider;
import android.content.ContentUris;
import android.content.ContentValues;
import android.content.Context;
import android.content.UriMatcher;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;
import android.database.sqlite.SQLiteQueryBuilder;
import android.net.Uri;
import android.text.TextUtils;
import android.util.Log;

public class ContactsProvider extends ContentProvider{

      private static final String DATABASE_NAME = "conDatabase.db";
    private static final String DATABASE_TABLE = "contactsTable";
    private static final int DATABASE_VERSION = 2;

    static final String PROVIDER_NAME = "npu.dce.project.ContactsProvider";
      static final String URL = "content://" + PROVIDER_NAME + "/cte";
      static final Uri CONTENT_URI = Uri.parse(URL);

    //EACH COLUMN IN DATABASE TABLE
    public static final String KEY_ID = "_id"; //primary key, CursorAdapter will use
this
    public static final String KEY_NAME = "NAME";
    public static final String KEY_PHONE = "PHONE";
    public static final String KEY_EMAIL = "EMAIL";
    public static final String KEY_POSTALADDR = "POSTALADDR";

    private SQLiteDatabase db;

    private static final UriMatcher uriMatcher;

    private static final int CONTACTS = 1;
    private static final int CONTACT_ID = 2;

    static {
        uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
        uriMatcher.addURI(PROVIDER_NAME, "cte", CONTACTS);
        uriMatcher.addURI(PROVIDER_NAME, "cte/*", CONTACT_ID);
    }

    @Override
    public int delete(Uri uri, String where, String[] whereArgs) {
```

```
    int count;

    switch (uriMatcher.match(uri)) {
      case CONTACTS:
        count = db.delete(DATABASE_TABLE, where, whereArgs);
        break;

      case CONTACT_ID:
        String segment = uri.getPathSegments().get(1);
        count = db.delete(DATABASE_TABLE, KEY_ID + "="
                                     + segment
                                     + (!TextUtils.isEmpty(where) ? " AND ("
                                     + where + ')' : ""), whereArgs);
        break;

      default: throw new IllegalArgumentException("Unsupported URI: " + uri);
    }

    getContext().getContentResolver().notifyChange(uri, null);
    return count;
  }

  public String getType(Uri uri) {
    switch (uriMatcher.match(uri)) {
      case CONTACTS: return "vnd.android.cursor.dir/cte";
      case CONTACT_ID: return "vnd.android.cursor.dir/cte";

      default: throw new IllegalArgumentException("Unsupported URI: " + uri);
    }
  }

  @Override
  public Uri insert(Uri _uri, ContentValues _initialValues) {
    // Insert the new row, will return the row number if
    // successful.
    long rowID = db.insert(DATABASE_TABLE, "quake", _initialValues);

    // Return a URI to the newly inserted row on success.
    if (rowID > 0) {
      Uri uri = ContentUris.withAppendedId(CONTENT_URI, rowID);
      getContext().getContentResolver().notifyChange(uri, null);
      return uri;
    }
    throw new SQLException("Failed to insert row into " + _uri);
  }

  @Override
   public boolean onCreate() {
          Context context = getContext();
          myDatabaseOpenHelper dbHelper = new myDatabaseOpenHelper(context);
          db = dbHelper.getWritableDatabase();
          if (db != null) {
                 return true;
          }
          return false;
    }

  @Override
  public Cursor query(Uri uri,
                      String[] projection,
                      String selection,
                      String[] selectionArgs,
                      String sort) {

    SQLiteQueryBuilder qb = new SQLiteQueryBuilder();

    qb.setTables(DATABASE_TABLE);

    // If this is a row query, limit the result set to the passed in row.
    switch (uriMatcher.match(uri)) {
      case CONTACT_ID: qb.appendWhere(KEY_ID + "=" + uri.getPathSegments().get(1));
                       break;
      default      : break;
    }

    // If no sort order is specified sort by id
    String orderBy;
    if (TextUtils.isEmpty(sort)) {
      orderBy = KEY_ID;
    } else {
      orderBy = sort;
    }

    // Apply the query to the underlying database.
    Cursor c = qb.query(db,
                        projection,
                        selection, selectionArgs,
                        null, null,
                        orderBy);

    // Register the contexts ContentResolver to be notified if
    // the cursor result set changes.
    c.setNotificationUri(getContext().getContentResolver(), uri);

    // Return a cursor to the query result.
    return c;
  }




  @Override
  public int update(Uri uri, ContentValues values, String where, String[] whereArgs) {
    int count;
    switch (uriMatcher.match(uri)) {
      case CONTACTS: count = db.update(DATABASE_TABLE, values,
                                               where, whereArgs);
                     break;
```

```
        case CONTACT_ID: String segment = uri.getPathSegments().get(1);
                       count = db.update(DATABASE_TABLE, values, KEY_ID
                               + "=" + segment
                               + (!TextUtils.isEmpty(where) ? " AND ("
                               + where + ')' : ""), whereArgs);
                       break;

        default: throw new IllegalArgumentException("Unknown URI " + uri);
      }

      getContext().getContentResolver().notifyChange(uri, null);
      return count;
    }

      ///////////////////////////////////////////////////////////////////////////////
      ////////////////////////////////////HELPER CLASS////////////////////////////////
      ///////////////////////////////////////////////////////////////////////////////
      private static class myDatabaseOpenHelper extends SQLiteOpenHelper
      {
            public myDatabaseOpenHelper(Context context, String name,
            CursorFactory factory, int version) {
            super(context, name, factory, version);
            }

            private static final String CREATE_TABLE =
            "create table " + DATABASE_TABLE + " (" +
            KEY_ID + " integer primary key autoincrement, " +
            KEY_NAME + " text not null, " +
            KEY_PHONE + " text, " +
            KEY_EMAIL + " text, " +
            KEY_POSTALADDR + " text);";

            myDatabaseOpenHelper(Context context)
            {
                  super(context, DATABASE_NAME, null, DATABASE_VERSION);
            }

            @Override
            public void onCreate(SQLiteDatabase db)
            {
                  db.execSQL(CREATE_TABLE);
            }

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