Работа с базой данных Android

Неделю назад столкнулся с тем что нужно было создать базу данных довольно серьезную в телефоне, и я начал копаться и пытаться освоить готовые примеры которые висят в интернете, я скажу вам что их там… больше чем навалом но я хочу внести и свою лепту так как большинство примеров мне не понравились они были запутанные и с большим количеством непонятного кода.   
  
Мой пример не менее запутанный но во всяком случае я вам покажу как создавать базу как в неё записывать данные, редактировать и удалять из неё. Это будет большой и серьезный пример но зато после того как вы увидите как я это делаю вы сможете при любой необходимости использовать этот пример и не будет возникать никаких проблем и вопросов.   
  
Статья рассчитана на тех кто имеет хоть малейшее понятия что такое базы данных и хотя бы раз работал с любой из существующих баз, если же таковых знаний и умений нет то рекомендую [хабрахабр](http://habrahabr.ru/search/?q=%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0+%D1%81+%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8+%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85&target_type=posts)для чтения этих статей их там полно. Конкретной статьи не посоветую так как сам с базами больше 4 лет уже работаю, учился не по статьям, а по методу тыка (: Так вот и научился.  
  
Собственно ниже я начинаю рассказывать как сделать красивейшую базу с блекджеком и кхмм… печеньками.  
У нас будет 3 активности которые будут содержать три разных типа контента. Первая будет у нас главной в ней мы будем вводить данные, во второй мы будем смотреть в списке что мы вводили и третья активность эта та в которой мы будем смотреть более подробные данные о выбранном поле в списке.   
  
База наша будет элементарной всего три поля имя, фамилия и возраст. Этого достаточно что бы понять всю суть статьи. И так давайте начнем, создадим три класса активности которые будет выполнять наш функционал, и создадим три xml файла для разметки наших активностей.  
  
Начнем мы с того что создадим три файла для создания базы данных. Ихний код с комментариями ниже:  
  
Для начала класс создания базы данных.   
  
**DatabaseOpenHelper.java**

import database.DatabaseContract.Names;

import android.content.Context;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

import android.provider.BaseColumns;

import android.util.Log;

/\*\* Класс создающий, удаляющий и редактирующий базу \*/

public class DatabaseOpenHelper extends SQLiteOpenHelper {

        private static final String DATABASE\_NAME = "db.db";

        private static final int DATABASE\_VERSION = 1;

        private static final String DEBUG\_TAG = DatabaseOpenHelper.class.getSimpleName();

        private static final boolean LOGV = false;

        public DatabaseOpenHelper(Context context) {

                super(context, DATABASE\_NAME, null, DATABASE\_VERSION);

        }

        /\*\* Удаление всех таблиц из базы

         \*

         \* @param db

         \*            - object of SQLiteDatabase \*/

        public void dropTables(SQLiteDatabase db) {

                if (LOGV) {

                        Log.d(DEBUG\_TAG, "onDropTables called");

                }

                db.execSQL("DROP TABLE IF EXISTS " + Names.TABLE\_NAME);

        }

        @Override

        public void onCreate(SQLiteDatabase db) {

                if (LOGV) {

                        Log.v(DEBUG\_TAG, "onCreate()");

                }

                db.execSQL("CREATE TABLE " + Names.TABLE\_NAME + " (" + BaseColumns.\_ID

                                + " INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL , " + Names.NamesColumns.NAME

                                + " TEXT NOT NULL, " + Names.NamesColumns.AGE + " INTEGER NOT NULL, "

                                + Names.NamesColumns.FNAME + " TEXT NOT NULL );");

        }

        @Override

        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

                Log.d(DEBUG\_TAG, "onUpgrade called");

        }

}

В этом классе мы создаем базу которую дальше мы будем использовать. Собственно больше нечего сказать.  
  
**DatabaseContract.java**

import android.provider.BaseColumns;

public class DatabaseContract {

        /\*\* Describes History Table and model. \*/

        public static class Names {

                /\*\* Default "ORDER BY" clause. \*/

                //сортируем по фамилии в убывающем порядке

                public static final String DEFAULT\_SORT = NamesColumns.FNAME + " DESC";

                //имя таблицы

                public static final String TABLE\_NAME = "People";

                //поле имя

                private String name;

                //наш айдишник

                private long id;

                //фамилия

                private String fname;

                //и сколько лет

                private int age;

                //

                // Ниже идут сетеры и гетеры для захвата данных из базы

                //

                public String getName() {

                        return name;

                }

                public long getId() {

                        return id;

                }

                public String getFname() {

                        return fname;

                }

                public double getAge() {

                        return age;

                }

                public void setName(String name) {

                        this.name = name;

                }

                public void setId(long id) {

                        this.id = id;

                }

                public void setFname(String fname) {

                        this.fname = fname;

                }

                public void setAge(int age) {

                        this.age = age;

                }

                /\*

                 \* (non-Javadoc)

                 \*

                 \* @see java.lang.Object#toString()

                 \*/

                @Override

                public String toString() {

                        StringBuilder builder = new StringBuilder();

                        builder.append(fname);

                        return builder.toString();

                }

                //Класс с именами наших полей в базе

                public class NamesColumns implements BaseColumns {

                        /\*\* Strings \*/

                        public static final String NAME = "name";

                        /\*\* String \*/

                        public static final String FNAME = "fname";

                        /\*\* String \*/

                        public static final String AGE = "age";

                }

        }

}

Этот класс в себе содержит все самое лучше что есть в программе, он содержит поля для записи в базу и сетеры гетеры которые будут захватывать данные когда вы запросите их.  
  
Дальше идет класс который обеспечивает запись, удаление и изменение данных.  
  
**ManController.java**

package database;

import database.DatabaseContract.Names;

import database.DatabaseContract.Names.NamesColumns;

import android.content.Context;

import android.database.Cursor;

import android.database.SQLException;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteException;

import android.provider.BaseColumns;

import android.util.Log;

import java.util.ArrayList;

public class ManController {

        private static final boolean LOGV = false;

        private static int maxRowsInNames = -1;

        private static final String TAG = ManController.class.getSimpleName();

        private ManController() {

        }

        public static int getMaxRowsInNames() {

                return maxRowsInNames;

        }

        /\*\*  Функция возвращает все данные из базы при запросе к ней

         \*

         \* @param context

         \* @return \*/

        public static ArrayList<Names> readNames(Context context) {

                ArrayList<Names> list = null;

                try {

                        DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(context);

                        SQLiteDatabase sqliteDB = dbhelper.getReadableDatabase();

                        String[] columnsToTake = { BaseColumns.\_ID, NamesColumns.FNAME };

                        Cursor cursor = sqliteDB.query(Names.TABLE\_NAME, columnsToTake, null, null, null, null,

                                        Names.DEFAULT\_SORT);

                        if (cursor.moveToFirst()) {

                                list = new ArrayList<Names>();

                        }

                        while (cursor.moveToNext()) {

                                Names oneRow = new Names();

                                oneRow.setId(cursor.getLong(cursor.getColumnIndexOrThrow(BaseColumns.\_ID)));

                                oneRow.setFname(cursor.getString(cursor.getColumnIndexOrThrow(NamesColumns.FNAME)));

                                list.add(oneRow);

                        }

                        cursor.close();

                        dbhelper.close();

                } catch (Exception e) {

                        Log.e(TAG, "Failed to select Names.", e);

                }

                return list;

        }

        public static void setMaxRowsInNames(int maxRowsInNames) {

                ManController.maxRowsInNames = maxRowsInNames;

        }

        /\*\*Изменение строки в списке\*/

        public static void update(Context context, String comment, long l) {

                try {

                        DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(context);

                        SQLiteDatabase sqliteDB = dbhelper.getWritableDatabase();

                        String quer = null;

                        int countRows = -1;

                        Cursor cursor = sqliteDB.query(Names.TABLE\_NAME, new String[] { "count(\*)" }, null, null, null,

                                        null, Names.DEFAULT\_SORT);

                        if (cursor.moveToFirst()) {

                                countRows = cursor.getInt(0);

                                if (LOGV) {

                                        Log.v(TAG, "Count in Names table" + String.valueOf(countRows));

                                }

                        }

                        cursor.close();

                        quer = String.format("UPDATE " + Names.TABLE\_NAME + " SET " + Names.NamesColumns.FNAME

                                        + " = '" + comment + "' WHERE " + BaseColumns.\_ID + " = " + l);

                        Log.d("", "" + quer);

                        sqliteDB.execSQL(quer);

                        sqliteDB.close();

                        dbhelper.close();

                } catch (SQLiteException e) {

                        Log.e(TAG, "Failed open database. ", e);

                } catch (SQLException e) {

                        Log.e(TAG, "Failed to update Names. ", e);

                }

        }

        /\*\*Удаление строки из списка\*/

        public static void delete(Context context, long l) {

                        DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(context);

                        SQLiteDatabase sqliteDB = dbhelper.getWritableDatabase();

                        sqliteDB.delete(Names.TABLE\_NAME, BaseColumns.\_ID  + " = " + l, null);

                        sqliteDB.close();

                        dbhelper.close();

        }

        /\*\* Эта функция создает запрос которые дальше записывает данные в нашу базу данных

         \*

         \* @param context

         \* @param latitude

         \* @param longitude \*/

        public static void write(Context context, String name, String fname, int age) {

                try {

                        //создали нашу базу и открыли для записи

                        DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(context);

                        SQLiteDatabase sqliteDB = dbhelper.getWritableDatabase();

                        String quer = null;

                        int countRows = -1;

                        //Открыли курсор для записи

                        Cursor cursor = sqliteDB.query(Names.TABLE\_NAME, new String[] { "count(\*)" }, null, null, null,

                                        null, Names.DEFAULT\_SORT);

                        if (cursor.moveToFirst()) {

                                countRows = cursor.getInt(0);

                                if (LOGV) {

                                        Log.v(TAG, "Count in Names table" + String.valueOf(countRows));

                                }

                        }

                        cursor.close();

                        if ((maxRowsInNames == -1) || (maxRowsInNames >= countRows)) {

                                //дальще наш запрос в базу для записи полученны х дынных из функции

                                quer = String.format("INSERT INTO %s (%s, %s, %s) VALUES (%s, %s, %s);",

                                                // таблица

                                                Names.TABLE\_NAME,

                                                // колонки

                                                Names.NamesColumns.NAME, Names.NamesColumns.AGE,

                                                Names.NamesColumns.FNAME,

                                                // поля

                                                name, age, fname);

                        }

                        //закрыли всю базу

                        sqliteDB.execSQL(quer);

                        sqliteDB.close();

                        dbhelper.close();

                } catch (SQLiteException e) {

                        Log.e(TAG, "Failed open rimes database. ", e);

                } catch (SQLException e) {

                        Log.e(TAG, "Failed to insert Names. ", e);

                }

        }

}

Собстно ничего трудного как по мне еще не было, все функции однотипные, я кстати привел два примера для запросов первый пример это в методах update() и write(), а второй пример это метод delete(), если обратите внимание то там в последнем я использую готовую функцию для удаления, андроиде есть уже готовые функции которые можно использовать подставив только свои нужные параметры, скажу очень удобно.  
  
Теперь нам осталось создать три активности о которых я говорил в самом начале. Создаем первую и самую главную активность которая собсно будет осуществлять запись в базу:  
  
**MainActivity.java**

import database.ManController;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.EditText;

public class MainActivity  extends Activity {

    @Override

    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.write\_to\_db);

        final EditText name = (EditText) findViewById(R.id.name);

        final EditText fname = (EditText) findViewById(R.id.fname);

        final EditText age = (EditText) findViewById(R.id.age);

        Button btn = (Button) findViewById(R.id.button1);

        btn.setOnClickListener(new OnClickListener() {

                        @Override

                        public void onClick(View arg0) {

                                ManController.write(getBaseContext(), '"'+name.getText().toString()+'"', '"'+fname.getText().toString()+'"', Integer.parseInt(age.getText().toString()));

                        }

                });

    }

    @Override

        public boolean onCreateOptionsMenu(Menu menu) {

                menu.add(Menu.NONE, 1, 0, "Список записей");

                return super.onCreateOptionsMenu(menu);

        }

        @Override

        public boolean onOptionsItemSelected(MenuItem item) {

                switch (item.getItemId()) {

                        case 1: {

                                Intent intent = new Intent(this, ListActivity.class);

                                startActivity(intent);

                        }

                                break;

                }

                return true;

        }

}

Тут все просто у нас есть в разметке которую я приведу ниже три поля для записи данных и один баттон который по нажатию вызывает функцию записи и передает ей параметры наших полей. Скажу что эти кавычки

'"'+fname.getText().toString()+'"'

не спроста, без них у вас будет выбивать ошибку так как для того что бы записать текстовый формат в базу нужно что бы он был выделен как текст, числовые же данные не нужно так выделять, так как они и так записываются.  
  
Код разметки для MainActivity:  
  
write\_to\_db.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    android:orientation="vertical" >

    <EditText

        android:id="@+id/name"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:ems="10" >

        <requestFocus />

    </EditText>

    <EditText

        android:id="@+id/fname"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:ems="10" >

        <requestFocus />

    </EditText>

    <EditText

        android:id="@+id/age"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:numeric="integer"

        android:ems="10" >

        <requestFocus />

    </EditText>

    <Button

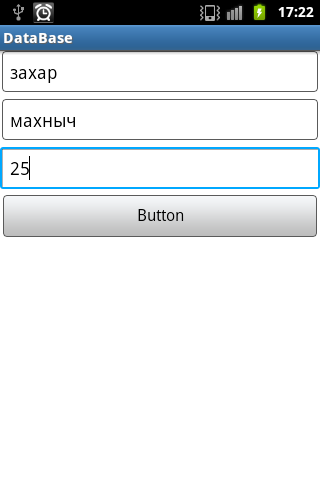
        android:id="@+id/button1"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:text="Button" />

</LinearLayout>

  
  
Дальше мы создадим нашу активность со списком. Её код ниже:  
  
**ListActivity.java**

package com.example.database;

import database.ManController;

import database.DatabaseContract.Names;

import database.DatabaseContract.Names.NamesColumns;

import database.DatabaseOpenHelper;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.Context;

import android.content.DialogInterface;

import android.content.Intent;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

import android.provider.BaseColumns;

import android.view.LayoutInflater;

import android.view.View;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemClickListener;

import android.widget.AdapterView.OnItemLongClickListener;

import android.widget.EditText;

import android.widget.ListView;

import android.widget.SimpleCursorAdapter;

public class ListActivity extends Activity {

        final Context context = this;

        int rowId = 0;

        @Override

        public void onCreate(Bundle savedInstanceState) {

                super.onCreate(savedInstanceState);

                setContentView(R.layout.listview);

                DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(getBaseContext());

                SQLiteDatabase sqliteDB = dbhelper.getReadableDatabase();

                final String[] from = { NamesColumns.FNAME, BaseColumns.\_ID };

                final Cursor c = sqliteDB.query(Names.TABLE\_NAME, null, null, null, null, null,

                                Names.DEFAULT\_SORT);

                final int i = c.getCount();

                final int[] to = new int[] { R.id.text1 };

                final SimpleCursorAdapter adapter = new SimpleCursorAdapter(getApplicationContext(), R.layout.list,

                                c, from, to);

                final ListView lv = (ListView) findViewById(R.id.listView1);

                lv.setAdapter(adapter);

                lv.setOnItemClickListener(new OnItemClickListener() {

                        @Override

                        public void onItemClick(AdapterView<?> a, View v, int position, long id) {

                                Intent intent = new Intent(ListActivity.this, DataActivity.class);

                                intent.putExtra("\_id", id);

                                startActivity(intent);

                                finish();

                        }

                });

                lv.setOnItemLongClickListener(new OnItemLongClickListener() {

                        @Override

                        public boolean onItemLongClick(AdapterView<?> arg0, View arg1, final int pos, long id) {

                                final CharSequence[] items = { "Удалить", "Переименовать" };

                                AlertDialog.Builder builder3 = new AlertDialog.Builder(ListActivity.this);

                                builder3.setTitle("Введите новое имя").setItems(items,

                                                new DialogInterface.OnClickListener() {

                                                        @Override

                                                        public void onClick(DialogInterface dialog, int item) {

                                                                switch (item) {

                                                                        case 0: {

                                                                                DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(getBaseContext());

                                                                                SQLiteDatabase sqliteDB = dbhelper.getReadableDatabase();

                                                                                ManController.delete(getBaseContext(), adapter.getItemId(pos));

                                                                                final Cursor c = sqliteDB.query(Names.TABLE\_NAME, null, null, null, null, null,

                                                                                                Names.DEFAULT\_SORT);

                                                                                adapter.changeCursor©;

                                                                                dbhelper.close();

                                                                                sqliteDB.close();

                                                                        }

                                                                                break;

     case 1: {

                                                                                // подключаем наш кастомный диалог лайаут

                                                                                LayoutInflater li = LayoutInflater.from(context);

                                                                                View promptsView = li.inflate(R.layout.promt, null);

                                                                                AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(

                                                                                                context);

                                                                                // делаем его диалогом

                                                                                alertDialogBuilder.setView(promptsView);

                                                                                final EditText userInput = (EditText) promptsView

                                                                                                .findViewById(R.id.editTextDialogUserInput);

                                                                                // вешаем на него событие

                                                                                alertDialogBuilder.setCancelable(false).setPositiveButton("OK",

                                                                                                new DialogInterface.OnClickListener() {

                                                                                                        @Override

                                                                                                        public void onClick(DialogInterface dialog, int id) {

                                                                                                                DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(getBaseContext());

                                                                                                                SQLiteDatabase sqliteDB = dbhelper.getReadableDatabase();

                                                                                     ManController.update(getBaseContext(), userInput

                                                                                     .getText().toString(), adapter.getItemId(pos));

                                                                             final Cursor c = sqliteDB.query(Names.TABLE\_NAME, null, null, null, null, null, Names.DEFAULT\_SORT);

                                                                                                                adapter.changeCursor©;

                                                                                                                dbhelper.close();

                                                                                                                sqliteDB.close();

                                                                                                        }

                                                                                                }).setNegativeButton("Cancel",

                                                                                                new DialogInterface.OnClickListener() {

                                                                                                        @Override

                                                                                                        public void onClick(DialogInterface dialog, int id) {

                                                                                                                dialog.cancel();

                                                                                                        }

                                                                                                });

                                                                                // создаем диалог

                                                                                AlertDialog alertDialog = alertDialogBuilder.create();

                                                                                // показываем его

                                                                                alertDialog.show();

                                                                        }

                                                                                break;

                                                                }

                                                        }

                                                });

                                builder3.show();

                                return true;

                        }

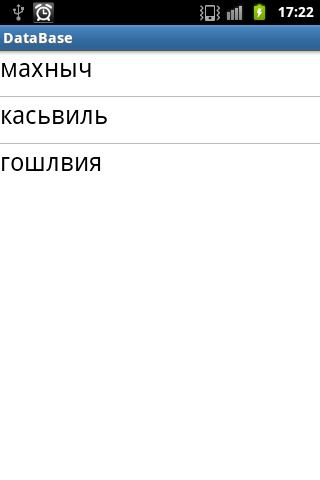
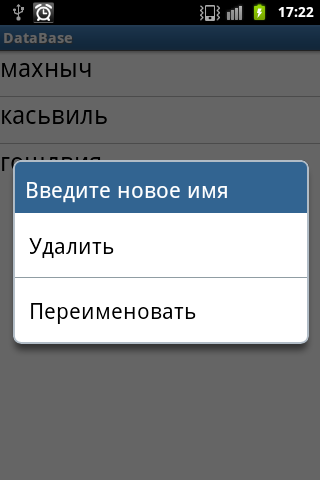
                });

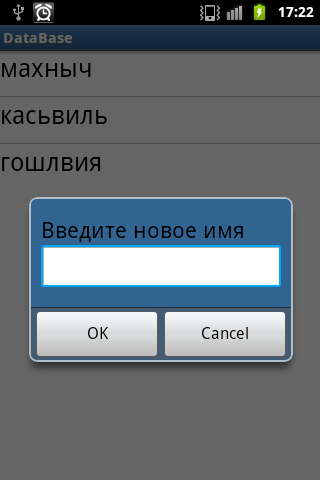
                dbhelper.close();

                sqliteDB.close();

        }

}

Тут я немного заморочился и решил сделать все красиво, в общем тут у нас все как обычно открываем базу для записи, и все дела, ниже я создаю листВью листенера который после обычного нажатия переходит на следующую активность для показа более детальных данных, а после долгого нажатия вызывается окно с выбором удалить или переименовать нажатое поле.  
  
 



Ниже я выложу три вьюшки, они нам понадобятся сохраните их у себя в res/layout по ходу дела разберетесь зачем они.  
  
Эта разметка для нашего ListActivity.java.  
  
**listview.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="fill\_parent"

    android:layout\_height="fill\_parent"

    android:orientation="vertical" >

    <ListView

        android:id="@+id/listView1"

        android:layout\_width="fill\_parent"

        android:layout\_height="wrap\_content" >

    </ListView>

</LinearLayout>

Эта вьюшка нам понадобится для вывода нашего списка в LisActivity.  
  
**list.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

  xmlns:android="http://schemas.android.com/apk/res/android"

  android:layout\_width="fill\_parent"

  android:orientation="horizontal"

  android:layout\_height="wrap\_content">

  <TextView

      android:id="@+id/text1"

      android:layout\_width="fill\_parent"

      android:layout\_height="46dp"

      android:textSize="25dip"

      android:textColor="#000" />

</LinearLayout>

Вьюшка что ниже нужна для редактирования полей в базе, она используется когда мы нажимаем после долгого нажатия на пункт меню «Переименовать».  
  
**promt.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:id="@+id/layout\_root"

    android:layout\_width="fill\_parent"

    android:layout\_height="fill\_parent"

    android:orientation="vertical"

    android:padding="10dp" >

    <TextView

        android:id="@+id/textView1"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="Введите новое имя"

        android:textAppearance="?android:attr/textAppearanceLarge" />

    <EditText

        android:id="@+id/editTextDialogUserInput"

        android:layout\_width="fill\_parent"

        android:layout\_height="wrap\_content" >

        <requestFocus />

    </EditText>

 </LinearLayout>

Теперь можем расслабиться и получать удовольствие, у нас осталось одна активность которая требует что бы её создали и все, конец (: База данных будет готова. Эта активность которая показывается после короткого нажатия на выбраное поле в списке с фамилиями.   
  
**DataActivity.java**

import database.DatabaseContract.Names;

import database.DatabaseContract.Names.NamesColumns;

import database.DatabaseOpenHelper;

import android.app.Activity;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

import android.provider.BaseColumns;

import android.util.Log;

import android.widget.EditText;

import android.widget.TextView;

public class DataActivity extends Activity {

    static final String TAG = DataActivity.class.getSimpleName();

    private Long mRowId;

    @Override

    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.test);

        //получаем из инта нужный нам айдишник и открываем нужное поле

        long id = getIntent().getLongExtra("\_id",-6);

        DatabaseOpenHelper dbhelper = new DatabaseOpenHelper(getBaseContext());

        SQLiteDatabase sqliteDB = dbhelper.getReadableDatabase();

        Cursor c = sqliteDB.query(Names.TABLE\_NAME, null, BaseColumns.\_ID + "=" + id, null, null, null,

                null);

        TextView lv = (TextView) findViewById(R.id.response);

        TextView tw = (TextView) findViewById(R.id.request);

        //выводим все в текствьюхи

        if (c.moveToFirst()) {

            tw.setText(c.getString(c.getColumnIndex(NamesColumns.NAME)));

            lv.setText(c.getString(c.getColumnIndex(NamesColumns.AGE)));

        }

        dbhelper.close();

        sqliteDB.close();

        Log.v(TAG, "ID=" + id);

    }

}

Делаем все то же, все так же, открыли базу для записи запилили прочитали вывели закрыли (:  
  
Код нашей разметки в активности будет вот такой.  
  
**test.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="fill\_parent"

    android:layout\_height="fill\_parent"

    android:orientation="vertical" >

    <TextView

        android:id="@+id/request"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="TextView"

        android:textSize="20sp" />

    <TextView

        android:id="@+id/response"

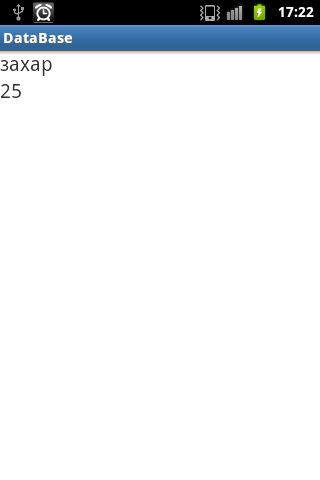
        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="TextView"

        android:textSize="20sp" />

</LinearLayout>

Два текстовых поля которые выводят нашу красату.  


Вот и все, база готова, осталось только прописать активности в манифесте и все будет