Práctica Bases de Datos para dispositivos Móviles con SQLite

***Fecha de entrega**: 09/11/16 de 6-8pm*
***Forma de entrega:** Individual o en parejas en el emulador o teléfono. La APP debe tener en la parte superior derecha los nombres de los integrantes sino no será admisible*

9.2. Crear una base de datos con Android

Para crear una base de datos SQLite, o abrir una ya existente, creamos un objeto de la clase `SQLiteDatabase`. Para ello se usa el método `openOrCreateDatabase` de la clase `ContextWrapper`, que es una superclase de Activity. En la siguiente aplicación creamos una base de datos de música mediante la instrucción

```
db=this.openOrCreateDatabase("musica.db",MODE_PRIVATE,null);
```

Esto crea el fichero `musica.db`, que es una base de datos SQLite localizada en el directorio de datos de nuestro dispositivo. Este fichero puede examinarse mediante la perspectiva DDMS de Eclipse o bien abriendo una shell con *avd shell*. En este ejemplo, la base de datos se almacena en el directorio

```
data/data/es.ugr.basededatos/databases
```

Dicha base de datos es completamente compatible con sqlite3. Sus contenidos pueden consultarse desde la línea de comandos o, como haremos a continuación, desde nuestra aplicación de Android. Con este fin utilizaremos los métodos disponibles de la clase `SQLiteDatabase` para ejecutar directamente instrucciones en lenguaje SQL.

El método `execSQL` admite como argumento una cadena con una instrucción SQL que no devuelve ningún resultado. Así, para crear la tabla `operas` con cuatro columnas, empleamos `create table if not exists`.

```
db.execSQL("create table if not exists "+
           " operas (id integer primary key, titulo text,"+
           "compositor text, year integer);");
```

Nótese que para definir la primera columna con SQL, hemos usado el tipo `integer primary key`, es decir, un número entero como clave primaria. En lenguaje SQL, este número indica el número de orden de cada fila, que debe ser único, y se autoincrementará automáticamente, si no lo hacemos nosotros, al crear una nueva fila.

Para insertar una nueva fila procedemos normalmente, usando la cláusula SQL `insert into`.

```
db.execSQL("insert into operas (titulo,compositor,year) "+
          " values('Don Govanni','W.A. Mozart',1787);");
```

Realizar una búsqueda en lenguaje SQL es igual de sencillo. Para ello se usa el método `rawQuery`, que devuelve un objeto de tipo `Cursor`. Para buscar todos los elementos de la tabla escribiríamos

```
Cursor cursor= db.rawQuery("select * from operas",null);
```

El objeto `Cursor` contiene el resultado de la búsqueda, además de información sobre las filas y columnas de la tabla. Como su propio nombre indica, podemos imaginarlo como una flecha que señala una fila de la búsqueda. Para extraer la primera fila, primero hay que colocarlo señalando el primer lugar mediante `moveToFirst()`. Para extraer las columnas almacenadas en el Cursor, usamos los métodos `getInt(i)` o `getString(i)`, dependiendo de si la columna número *i* contiene un número entero o una cadena. Extraídos estos elementos, hay que moverlo al segundo lugar mediante `moveToNext()`, y procederíamos del mismo modo. El método `moveToNext()` devuelve `false` si el Cursor está situado en la última fila y `true` en caso contrario. El número de filas contenidas en el Cursor se obtiene con `getCount()` y el número de columnas, con `getColumnCount()`.

Para la aplicación `BaseDeDatosActivity` usamos el siguiente layout:

```
<?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"
    android:background="#ffffff"
    >

    <TextView
        android:id="@+id/textView"
```

```
        android:textColor="#000000"
        android:textSize="18sp"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Base de Datos SQLite" />

</LinearLayout>
```

A continuación se detalla la aplicación `BaseDeDatosActivity`. Nótese que hemos definido dos métodos: `ejecutaSQL()`, para incluir todas las manipulaciones de la tabla y búsquedas, y `muestraTabla()`, para mostrar los contenidos del Cursor. Esto permitirá que, posteriormente, podamos realizar modificaciones del programa de un modo más cómodo. Cada vez que ejecutemos este programa, se abrirá la base de datos y se insertará el mismo registro, la ópera Don Giovanni, en una fila de la tabla. En la figura 9.2. se muestra el resultado de ejecutar este programa siete veces.

Figura 9.2. *Una base de datos con cuatro columnas y siete filas.*

```
package es.ugr.amaro.basededatos;

import android.app.Activity;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
```

```
import android.widget.TextView;

public class BaseDeDatosActivity extends Activity {

    TextView tv;
    String texto="";
    SQLiteDatabase db=null;
    Cursor cursor=null;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        tv= (TextView) findViewById(R.id.textView);

        //---abre o crea base de datos---
        db=this.openOrCreateDatabase(
                        "musica.db",MODE_PRIVATE,null);

        //---crea una tabla si no existe
        db.execSQL("create table if not exists "+
           " operas (id integer primary key, titulo text,"+
           " compositor text, year integer);");

        ejecutaSQL();
        muestraTabla();
        db.close();
        tv.append(texto);
    }

    void ejecutaSQL(){

        //---inserta datos en la tabla---
        db.execSQL(
          "insert into operas (titulo,compositor,year) "
        +" values('Don Govanni','W.A. Mozart',1787);");

        //---selecciona todos los datos en un Cursor---
        cursor= db.rawQuery("select * from operas",null);

    }  // end ejecutaSQL

    void muestraTabla(){

        tv.append("\nTamaño: "+db.getPageSize());
        int numeroDeColumnas=cursor.getColumnCount();
        tv.append("\nColumnas: "+numeroDeColumnas);
        int numeroDeFilas=cursor.getCount();
```

```
        tv.append("\nFilas: "+numeroDeFilas);

         cursor.moveToFirst();
         for (int i=1;i<=numeroDeFilas;i++){
         //---loop sobre las filas---
            int id=cursor.getInt(0);
            String titulo=cursor.getString(1);
            String compositor=cursor.getString(2);
            int year=cursor.getInt(3);
            texto=texto+"\n "+id+", "+titulo+
            ", "+compositor+", "+year;
            cursor.moveToNext();

         }
    }  //  end muestraTabla
}
```