

# U.T.6 PROGRAMACIÓN PARA DISPOSITIVOS MÓVILES. BASES DE DATOS



## **BASES DE DATOS. SQLite para Android.**

Las bases de datos son una herramienta de gran potencia en la creación de aplicaciones informáticas. Android incorpora la **librería SQLite** que nos permitirá utilizar bases de datos mediante el lenguaje SQL, de una forma sencilla y utilizando muy pocos recursos del sistema.

Para manipular una base de datos en Android usaremos la clase **SQLiteOpenHelper** que nos facilita tanto la creación de la base de datos, como el trabajar con futuras versiones de esta base de datos.

**SQLiteOpenHelper(context, NOMBREBD, null, VERSION);**

El constructor de la clase **SQLiteOpenHelper** contiene los argumentos: **contexto**, **nombre de la base de datos**, **objeto CursorFactory** (null por defecto) y **versión**.

# MÉTODOS DE LA CLASE SQLiteOpenHelper

Además de heredar de la clase `SQLiteOpenHelper` hay que implementar los métodos `onCreate()`, y `onUpgrade()` y opcionalmente `onOpen()`.

Además, esta clase tiene dos métodos `getReadableDatabase()` y `getWritableDatabase()` que abren la base de datos en modo solo lectura o lectura y escritura.

```
public void crearBD extends SQLiteOpenHelper {  
    public crearBD (Context context) {  
        super(context, NOMBREBD, null, VERSION);  
    }  
    public void onCreate(SQLiteDatabase db)  
    {  
        db.execSQL("create table if not exists tablaprueba  
(id integer primary key autoincrement not null,  
nombre text);");  
    }  
    public void onUpdate(SQLiteDatabase db){ }  
}
```

# MÉTODOS DE LA CLASE SQLiteDatabase

La clase `SQLiteDatabase` también proporciona métodos para crear, abrir y manipular la base de datos.

`openOrCreateDatabase` Permite crear una base de datos con su nombre de base de datos y el modo como parámetro.

`openDatabase` Permite abrir una base de datos ya existente.

`execSQL` ejecuta una sentencia SQL.

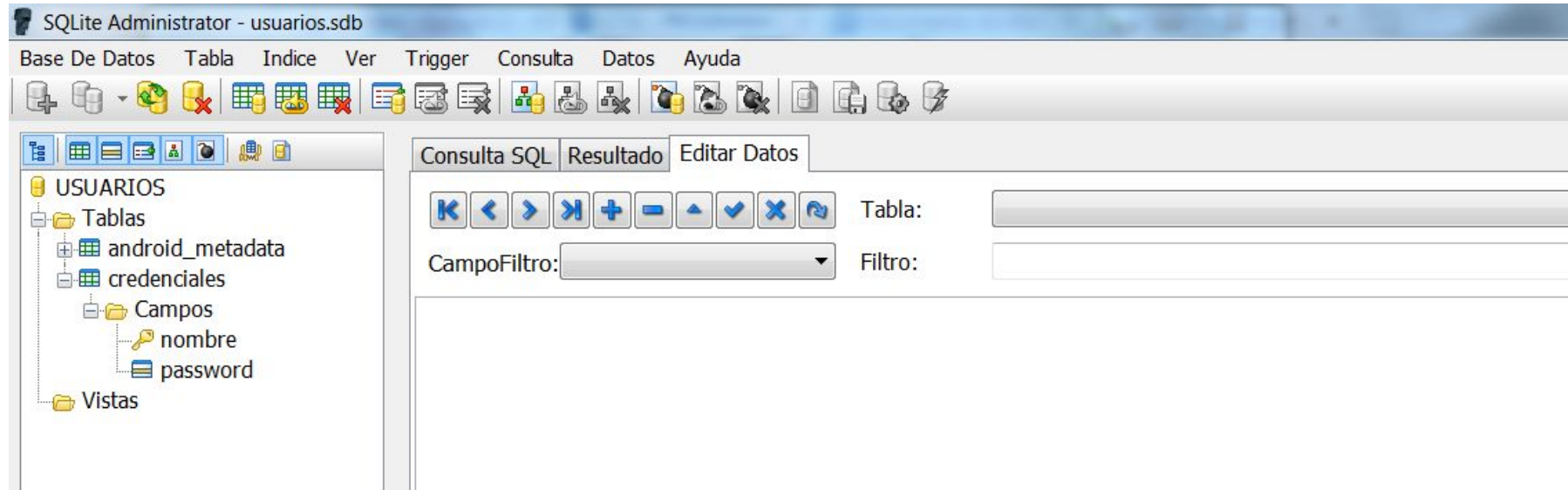
```
SQLiteDatabase mydatabase = openOrCreateDatabase("miBase",MODE_PRIVATE,null);
```

```
mydatabase.execSQL("CREATE TABLE IF NOT EXISTS Usuarios(Username VARCHAR>Password VARCHAR);");
```

```
mydatabase.execSQL("INSERT INTO Usuarios VALUES('admin','admin');");
```

# Entorno del administrador de SQLite

**SQLite** es una base de datos relacional, utiliza **SQL** para crear, seleccionar y manipular los datos que podrían ser de una aplicación. **SQLite Administrator** es una aplicación que permite operar de forma sencilla con la base de datos. Puedes bajar la aplicación en <http://sqliteadmin.orbmu2k.de/>



# Explorar ficheros en Android. File Explorer

El explorador de archivos del dispositivo en android: **File Explorer** nos permite localizar los archivos de la base de datos SQLite en la ruta:

**`/data/data/paquete.java.de.la.aplicacion/databases/nombre_base_datos`**

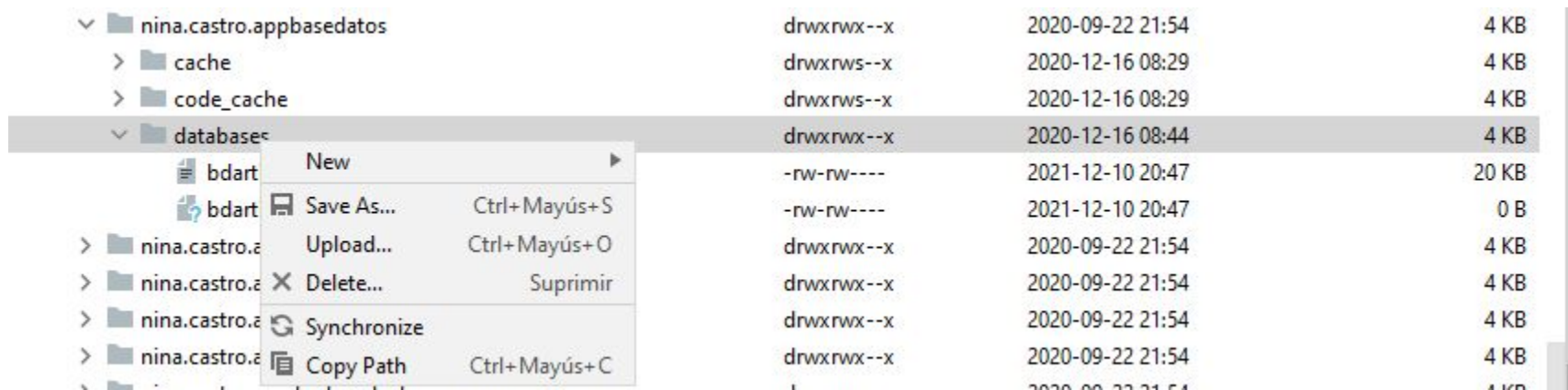
siempre que el dispositivo (AVD) esté iniciado.

Device File Explorer				
Emulator Nexus_5_API_26 Android 8.0.0, API 26				
Name	Permissions	Date	Size	
> nina.castro.appanimframe	drwxrwx--x	2020-09-22 21:54	4 KB	
▼ nina.castro.appbasedatos	drwxrwx--x	2020-09-22 21:54	4 KB	
> cache	drwxrws--x	2020-12-16 08:29	4 KB	
> code_cache	drwxrws--x	2020-12-16 08:29	4 KB	
▼ databases	drwxrwx--x	2020-12-16 08:44	4 KB	
bdarticulos.sdb	-rw-rw----	2021-12-10 20:47	20 KB	
bdarticulos.sdb-journal	-rw-rw----	2021-12-10 20:47	0 B	

# Explorar la base de datos creada en el emulador

Abre [Device File Explorer](#), selecciona el archivo de la base de datos y pulsa botón derecho del ratón y **"save as"** para guardar en disco. Entonces podemos consultarla con cualquier [administrador de bases de datos SQLite](#).

Consulta o modifica la base de datos desde el administrador SQLite y vuelve a cargar el archivo de bases de datos en el emulador, elige **"Upload"**



# Manipular datos en una tabla

Crea una instancia de la clase que hereda de SQLiteOpenHelper para crear la Base de Datos. Abre la base de datos para lectura y escritura

- `CrearBD crearBD= new CrearBD(this);`
- `SQLiteDatabase bd = crearBD.getWritableDatabase();`

Inserta una fila en la tabla tablaPrueba

- `bd.execSQL("INSERT INTO tablaprueba VALUES(1,'Pepito');");`

Actualizar datos

- `bd.execSQL("UPDATE tablaprueba SET nombre='Pepe' WHERE nombre='Pepito' ");`

Borrado de una fila.

- `bd.execSQL("DELETE FROM tablaprueba WHERE nombre='Ana'");`



# Consultar datos en una tabla

Primero debemos abrir la base de datos para lectura.

```
SQLiteDatabase bd = crearBD.getReadableDatabase();
```

Lo podemos realizar de dos formas:

- Con el método **rawQuery()** y la consulta:

```
Cursor cursor = bd.rawQuery("Select nombre from tablaprueba", null)
```

- Con el método **query(table,column,selection, args, groupby, having, order by, limit), :**

```
Cursor cursor = bd.query("tablaprueba", new String[]{"nombre"}, null, null, null,  
null, null);
```

# Cursor

Todos los datos obtenidos en la consulta se guardan en el objeto **Cursor**.

Para recorrer un cursor debemos tener en cuenta :

- El **índice** de la columna en la consulta, es decir si queremos consultar el nombre solamente, el índice sería el número 0 pero si queremos obtener el id y el nombre el índice del id sería el 0 y el del nombre sería el 1.
- El **tipo de dato** de cada campo. numérico, de texto,...

```
id=cursor.getInt(0);
```

```
nombre=cursor.getString(1);
```

Es conveniente antes de acceder al primer registro conocer el número de filas **cursor.getCount()** ya que si la tabla está vacía e intentamos ver el contenido nos dará error.

# Lectura de un cursor

```
public void consultarDatos(){  
    List <String> items = new ArrayList <String>();  
    bd = crearBD.getReadableDatabase();  
    Cursor cursor = bd.rawQuery("select * from tablaprueba", null);  
    int i=0;  
    String cad="";  
    //volcamos el contenido del cursor en el arrayList.  
    while (cursor.moveToNext() && i<cursor.getCount())  
    { //concatena los datos id y nombre en un String cad.  
        cad="" + cursor.getInt(0) + " " + cursor.getString(1) + "\n";  
        items.add(cad);  
        i++;  
    }  
}
```