TALLER 18: APP CON BASES DE DATOS LIGERAS (SQLite)

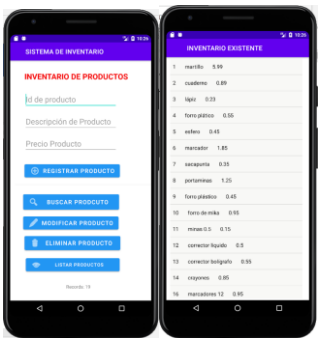
Requerimientos

- ✓ Crear un Proyecto "Empty Activity" (Lollipod 5.0)
- ✓ Crear +2 Activity(Layout)
- ✓ Insertar los siguientes componentes ***MainActivity1***:
  - 1 EditTex
  - 5 Button (operciones)
  - 2 TextView (titulo, contador)
- ✓ ***MainActivity2 :***
  ListView (Legacy)
- ✓ ***Implementar un aplicación que permite administrar una base de datos ligera con todos sus operaciones básicas.***
- ✓ ***Usar el Software BD Software SQLite para comprobar los datos.***
- ✓ Agregar un AVD (Virtual Device) para correr la app móvil (optimo)
- ✓ Compilar y Revisar Errores.

Descargar en instalar DB browser for SQlite
https://sqlitebrowser.org/dl/

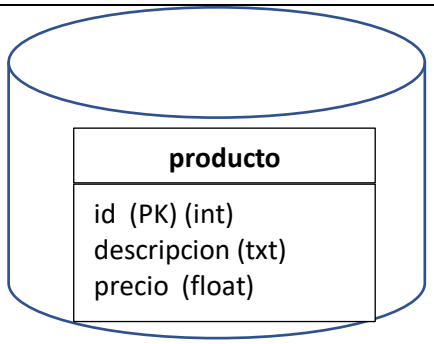
https://www.youtube.com/watch?v=TxkdWX3UaNk

Pantallas

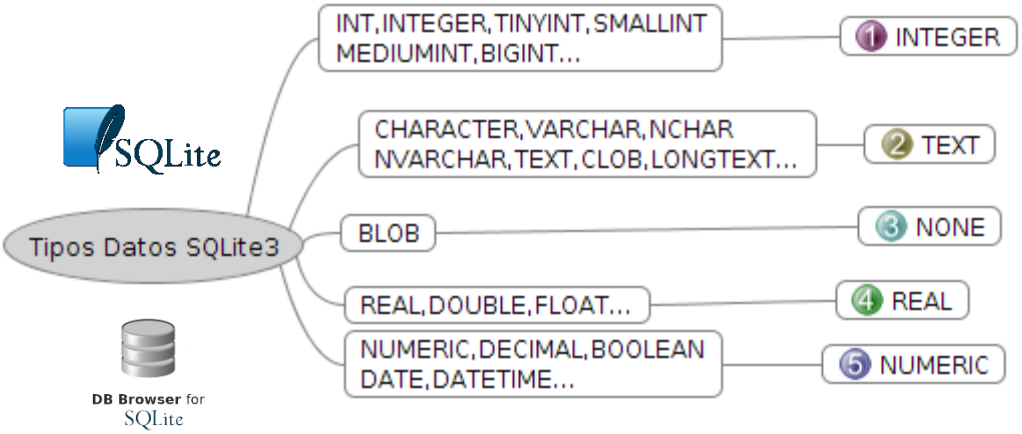


Diseño BD

MiBase




```
create table producto
(
id int primary key
descripcion text
precio real
)
```

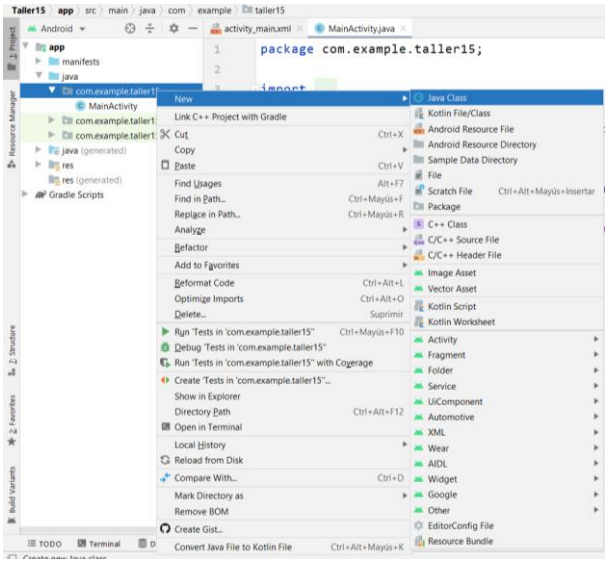


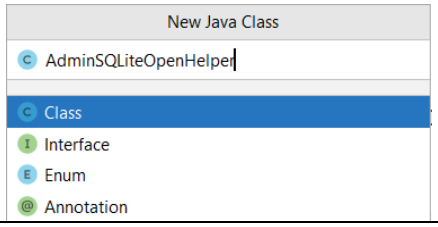
Procedimiento

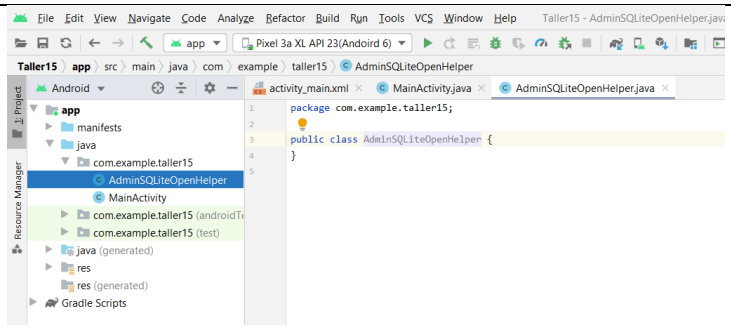
1. Implementar una clase de JAVA para habilitar la librería de conexión a bases de datos SQLite: **SQLiteOpenHelper**

//Crear clase para conexión

Click derecho (árbol de proyecto) en la carpeta que mantiene la parte lógica del app (java►com…..) New-JavaClass, y asignamos un nombre "AdminSQLiteOpenHelper"







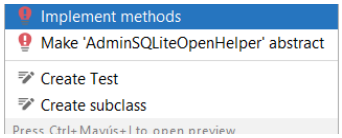
Importamos la librería "import android.database.sqlite.SQLiteOpenHelper"

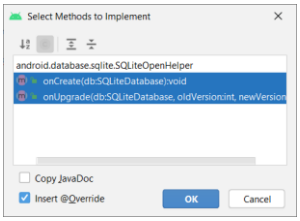
```
package com.example.taller15;

import android.database.sqlite.SQLiteOpenHelper;

public class AdminSQLiteOpenHelper {
}
```

Relacionamos mi clase con la librería importada ***extends***

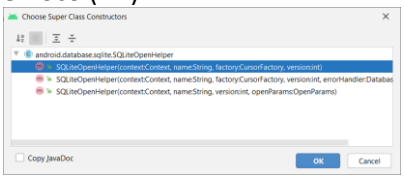
```
public class AdminSQLiteOpenHelper extends SQLiteOpenHelper{
}
```

Generamos automáticamente las clases OnCreate y OnUpgrate
Click en el foco ( ) implementar métodos OnCreate y OnUpgrate …OK





Crear el Constructor Click en el foco ( ) Create constructor matching super (4 parámetros) y ok



Quedará asi:

```
package com.example.taller15;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import androidx.annotation.Nullable;
```

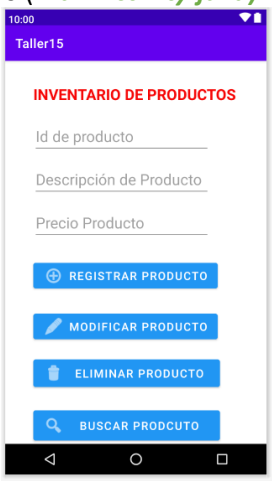
```
public class AdminSQLiteOpenHelper extends SQLiteOpenHelper{
    public AdminSQLiteOpenHelper(@Nullable Context context, @Nullable String name, @Nullable
SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {

    }

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

    }
}
```

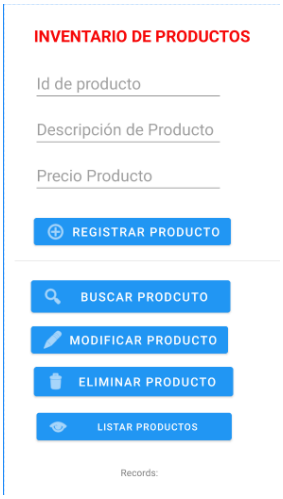
Le cambianos el nombre de la variable de `sqLiteDatabase` a `db`

Insertamos la sentencia para crear la tabla dentro de la base de datos en el método ***OnCreate***

```
@Override
public void onCreate(SQLiteDatabase db) {
   db.execSQL("create table producto(id int primary key, descripcion text, precio real)");

}
```

2. Diseñar el Interfaz de la app (componentes) , crear variables propias y asignar las componentes a las variables propias.(***MainActivity.java)***






```
public class MainActivity extends AppCompatActivity {

    EditText txtid, txtdescripcion, txtprecio;
    Button btnregistrar, btnmodificar, btneliminar, btnbuscar;

        FloatingActionButton btncerrar;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        getSupportActionBar().setTitle("      SISTEMA DE INVENTARIO");


        txtid=(EditText)findViewById(R.id.editTextTextPersonName);
        txtdescripcion=(EditText)findViewById(R.id.editTextTextPersonName2);
        txtprecio=(EditText)findViewById(R.id.editTextTextPersonName3);

        btnregistrar = (Button)findViewById(R.id.button);
        btnmodificar = (Button)findViewById(R.id.button2);
        btneliminar = (Button)findViewById(R.id.button3);
        btnbuscar = (Button)findViewById(R.id.button4);

         FloatingActionButton btncerrar;

    }
}
```

3. Crearemos los métodos para las operaciones : Registrar, Modificar, Eliminar, Buscar, Contar, Mostrar:

Se pueden crear métodos propios, o usar los métodos OnClickListener
Para trabajar con BD SQLite es importante mantener este orden.

- Crear/Conectarse BD
- Abrir BD
- Ejecutar SQL
- Cerrar BD

//INSERCIÓN

Método 1: ContentValues
Método 2: sentencias sql

**********METODO 1 : ContentValues **********

```
//metodo registrar producto
public void registrarproducto(View view)
{
     //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();



      //campturo lo que esta en los text
      String id = txtid.getText().toString();
      String descripcion = txtdescripcion.getText().toString();
      String precio = txtprecio.getText().toString();


     if(!id.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty())
     {


    //Ejecutar SQL
    ContentValues values = new ContentValues();
    values.put("id", id);
    values.put("descripcion", descripcion);
    values.put("precio", precio);

    BaseDeDatos.insert("producto", null, values);


    //Cerrar BD
     BaseDeDatos.close();
     Toast.makeText(this,"Dato Almacenado",Toast.LENGTH_SHORT).show();

   //limpiar campos
          txtid.setText("");
          txtdescripcion.setText("");
          txtprecio.setText("");
          txtid.requestFocus();



     }
     else {
         Toast.makeText(this,"Existen Campos vacios",Toast.LENGTH_SHORT).show();

             If(id.isEmpty){txt.setError("Falta ID")}
             If(descripcion.isEmpty){txt.setError("Falta Descripcion")}
             If(precio.isEmpty){txt.setError("Falta Precio")}

         }

      }
```

********** METODO 2 : Sentencias Sql **********

```
//metodo registrar producto
public void registrarproducto(View view)
{
     //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);
```

```
    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();



       //capturo lo que esta en los text
       String id = txtid.getText().toString();
       String descripcion = txtdescripcion.getText().toString();
       String precio = txtprecio.getText().toString();

     if(!id.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty())
     {



    //Ejecutar SQL

   //insert into producto (id, descripcion, precio) values(1,'martillo', 12.99)

    String sql="insert into producto (id, descripcion, precio) values("+ id +",'"+ descripcion +"',"+ precio +")";
    BaseDeDatos.execSQL(sql);

    //Cerrar BD
     BaseDeDatos.close();
     Toast.makeText(this,"Dato Almacenado",Toast.LENGTH_SHORT).show();

//limpiar campos
           txtid.setText("");
           txtdescripcion.setText("");
           txtprecio.setText("");
           txtid.requestFocus();



     }
     else {
         Toast.makeText(this,"Existen Campos vacios",Toast.LENGTH_SHORT).show();

     }

}
```

Nota: Se recomienda trabajar con Try Cath, para evitar que se cuelgue la app.

**********Método 1**********

```
//metodo registrar producto
public void registrarproducto(View view)
{
     //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();



       //capturo lo que esta en los text
       String id = txtid.getText().toString();
       String descripcion = txtdescripcion.getText().toString();
       String precio = txtprecio.getText().toString();

     if(!id.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty())
     {


    ContentValues values = new ContentValues();
    values.put("id", id);
    values.put("descripcion", descripcion);
    values.put("precio", precio);


    try {

  //Ejecutar SQL
    BaseDeDatos.insert("producto", null, values);

    //Cerrar BD
     BaseDeDatos.close();
     Toast.makeText(this,"Dato Almacenado",Toast.LENGTH_SHORT).show();

   //aquí limpiar los campos
           txtid.setText("");
           txtdescripcion.setText("");
           txtprecio.setText("");
           txtid.requestFocus();


    }
    catch (Exception e)
    {
        Toast.makeText(this,"error "+e,Toast.LENGTH_SHORT).show();
        BaseDeDatos.close();
```

```
    }

    }
    else {
        Toast.makeText(this,"Existen Campos vacios",Toast.LENGTH_SHORT).show();

        Snackbar.make(view, "Existen Campos Vacios", Snackbar.LENGTH_LONG).show();

         // Snackbar.make(view, "Existen Campos Vacios",Snackbar.LENGTH_LONG).setAction("BOTON", new
         View.OnClickListener() {
             @Override
             public void onClick(View view) {
                 Toast.makeText(MainActivity.this, "Existen campos vacios", Toast.LENGTH_SHORT).show();
             }
         }).show();


    }

}
```

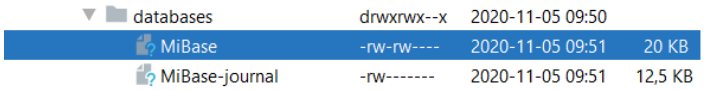
*********método 2**********

```
public void registrarproducto(View view)
{
     //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();



       //capturo lo que esta en los text
       String id = txtid.getText().toString();
       String descripcion = txtdescripcion.getText().toString();
       String precio = txtprecio.getText().toString();

     if(!id.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty())
     {


    //Ejecutar SQL
   //insert into producto (id, descripcion, precio) values(1,'martillo', 12.99)

     try {

    String sql="insert into producto (id, descripcion, precio) values("+ id +",'"+ descripcion +"',"+ precio +")";
    BaseDeDatos.execSQL(sql);

    //Cerrar BD
     BaseDeDatos.close();
     Toast.makeText(this,"Dato Almacenado",Toast.LENGTH_SHORT).show();

    //aquí limpiar los campos
          txtid.setText("");
          txtdescripcion.setText("");
          txtprecio.setText("");
          txtid.requestFocus();


     }
     catch (Exception e)
     {
         Toast.makeText(this,"error "+e,Toast.LENGTH_SHORT).show();
         BaseDeDatos.close();
     }


     }
     else {
         Toast.makeText(this,"Existen Campos vacios",Toast.LENGTH_SHORT).show();

     }

}
```

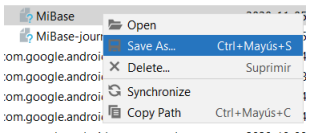
Verificar la Base de Datos usando ***BD Browser for SQLite***

Ir al panel de Explorador de Archivos del Dispositivo ubicado al lado derecho del IDE

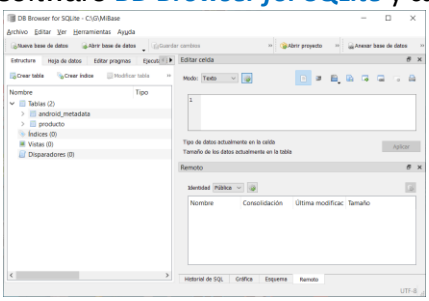
Paht: data▶data▶com.example.taller15▶databases

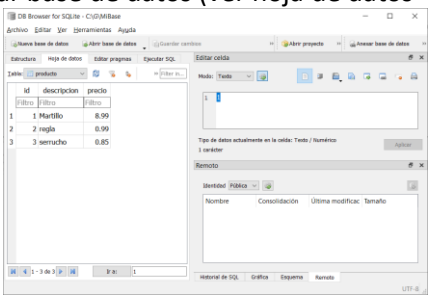


Click derecho SAVE AS



Abrir el software ***BD Browser for SQLite*** y cargar base de datos (ver hoja de datos - bd)





//Modificación

- Crear/Conectarse BD
- Abrir BD
- Ejecutar SQL
- Cerrar BD

Método 1: ContentValues
Método 2: sentencias sql

**********METODO 1 : ContentValues **********

```
public void modificarproducto(View view)
{
    //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();
```

```
    //capturo lo que esta en los text
    String id = txtid.getText().toString();
    String descripcion = txtdescripcion.getText().toString();
    String precio = txtprecio.getText().toString();


    if(!id.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty())
    {
        ContentValues values = new ContentValues();
        values.put("id", id);
        values.put("descripcion", descripcion);
        values.put("precio", precio);

        //ejecutamos la TranSQL
         int  band = BaseDeDatos.update("producto",values, "id = " + id,null);

        //cerramos BD
        BaseDeDatos.close();

        if(band==1)
        {
            Toast.makeText(this,"Datos Modificados",Toast.LENGTH_SHORT).show();

            //limpiar campos
            txtid.setText("");
            txtdescripcion.setText("");
            txtprecio.setText("");
            txtid.requestFocus();
        }
        else
        {
            Toast.makeText(this,"Producto no existe, imposible modificar",Toast.LENGTH_SHORT).show();

            //limpiar campos
            txtid.setText("");
            txtdescripcion.setText("");
            txtprecio.setText("");
            txtid.requestFocus();
        }


    }
    else
    {
        Toast.makeText(this,"Existen Campos vacios",Toast.LENGTH_SHORT).show();
    }

}
```

*********método 2**********

```
public void modificarproducto(View view)
{
    //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();

    //capturo lo que esta en los text
    String id = txtid.getText().toString();
    String descripcion = txtdescripcion.getText().toString();
    String precio = txtprecio.getText().toString();


    if(!id.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty())
    {

      try {
      //ejecutar update producto set descripción='algo', precio=25.55 where id = 1

     String sql = "update producto set descripcion = '" + descripcion + "', precio = " + precio + " where id = " + id + " ";
                BaseDeDatos.execSQL(sql);
                Toast.makeText(this,"Datos Modificados",Toast.LENGTH_SHORT).show();

               //cierra la BD
                BaseDeDatos.close();

                //limpiar campos
                txtid.setText("");
                txtdescripcion.setText("");
                txtprecio.setText("");
                txtid.requestFocus();
        }
        catch (Exception e)
```

```
        {

                Toast.makeText(this,"Problemas: Producto no existe, imposible modificar",Toast.LENGTH_SHORT).show();

                //limpiar campos
                txtid.setText("");
                txtdescripcion.setText("");
                txtprecio.setText("");
                txtid.requestFocus();

                //cierra la BD
                BaseDeDatos.close();
            }


    }
    else
    {
        Toast.makeText(this,"Existen Campos vacios",Toast.LENGTH_SHORT).show();

           //Limpiar campos
                txtid.setText("");
                txtdescripcion.setText("");
                txtprecio.setText("");
                txtid.requestFocus();
    }

}
```

//Eliminación

- Crear/Conectarse BD
- Abrir BD
- Ejecutar SQL
- Cerrar BD

**********Método 1: ContentValues

```
public void eliminarproducto(View view) {

        //Crear/Conectarse BD
        AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper(this, "MiBase", null, 1);

        //abrir BD
        SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();

        //capturo el codigo del producto
        String id = txtid.getText().toString();


       if(!id.isEmpty())
        {
           //ejecutamos la transql
            int cantidad = BaseDeDatos.delete("producto","id ="+ id, null);

            //cerramos BD
            BaseDeDatos.close();


            if(cantidad==1)
            {
                Toast.makeText(this,"Producto Eliminado",Toast.LENGTH_SHORT).show();
                //limpiar campos
                txtid.setText("");
                txtdescripcion.setText("");
                txtprecio.setText("");
                txtid.requestFocus();

            }
            else
            {
                Toast.makeText(this,"Producto no existe",Toast.LENGTH_SHORT).show();
                //limpiar campos
                txtid.setText("");
                txtdescripcion.setText("");
                txtprecio.setText("");
                txtid.requestFocus();
            }


        }
        else
        {
            Toast.makeText(this,"No se ha especificado el ID del producto",Toast.LENGTH_SHORT).show();
        }


    }
```

**********Método 2: sentencias sql

```
public void eliminarproducto(View view) {

        //Crear/Conectarse BD
        AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper(this, "MiBase", null, 1);

        //abrir BD
        SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();

        //capturo el codigo del producto
        String id = txtid.getText().toString();

if (!id.isEmpty()) {


  try {

      //ejecutamos la transql : delete from producto where id = 1;
      String sql = "delete from producto where id = " + id;
      BaseDeDatos.execSQL(sql);

      //cerramos BD
      BaseDeDatos.close();

      Toast.makeText(this,"Producto Eliminado",Toast.LENGTH_SHORT).show();
      //limpiar campos
      txtid.setText("");
      txtdescripcion.setText("");
      txtprecio.setText("");
      txtid.requestFocus();

  }
  catch (Exception e)
  {
      Toast.makeText(this,"Producto No existe",Toast.LENGTH_SHORT).show();

      //cerramos BD
      BaseDeDatos.close();
  }

}
else
{
    Toast.makeText(this,"No se ha especificado el ID del producto",Toast.LENGTH_SHORT).show();
}


}
```

//Buscar

- Crear/Conectarse BD
- Abrir BD
- Ejecutar SQL
- Cerrar BD

```
public void buscarproducto(View view)
{
    //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper(this, "MiBase", null, 1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();

    //capturo el codigo del producto
    String id = txtid.getText().toString();

    if(!id.isEmpty())
    {
        //ejecutar transql
         Cursor fila = BaseDeDatos.rawQuery("select * from producto where id = "+ id , null);

         if(fila.moveToFirst())
         {
             //mostrar datos en los campos
```

```
            txtdescripcion.setText(fila.getString(1));
            txtprecio.setText(fila.getString(2));

            //cerramos BD
            BaseDeDatos.close();
        }
        else
        {
            Toast.makeText(this,"Producto no existe",Toast.LENGTH_SHORT).show();
            txtid.setText("");
            txtdescripcion.setText("");
            txtprecio.setText("");
            txtid.requestFocus();

            //cerramos BD
            BaseDeDatos.close();

        }

    }
    else
    {
        Toast.makeText(this,"Debe ingresar Id de producto para buscar",Toast.LENGTH_SHORT).show();

    }

}
```

//Contar Registros

- Crear/Conectarse BD
- Abrir BD
- Ejecutar SQL
- Cerrar BD
-

```
public int contarregitros ()
{


    //Crear/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper(this, "MiBase", null, 1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getWritableDatabase();

    //ejecuta sql
    String sql= "select id from producto";
    int num = BaseDeDatos.rawQuery(sql,null).getCount();
    //cerramos BD
    BaseDeDatos.close();


    return num;
}
```

//mostramos el número de registros en un TextView(label) en el evento Create

```
txttotal.setText("Records: "+ contarregitros());
```

//Mostrar Registros – List View(SIMPLE)

- Crear/Conectarse BD
- Abrir BD para lectura
- Ejecutar SQL
- Recorrer lista y llenar ArrayList
- Cerrar BD

//en la Activity2

//Agregamos al Layout el componete ListView (Legacy)(ocupando todo el screen)

//creamos una función que muestre y devuelva una Lista de Datos "ArrayList"

```
public ArrayList mostrarprodutos()
{
 //conectar/Conectarse BD
    AdminSQLiteOpenHelper cnx = new AdminSQLiteOpenHelper( this, "MiBase", null,1);

    //abrir BD
    SQLiteDatabase BaseDeDatos = cnx.getReadableDatabase();

    //ejecutar sql
    String sql = "Select * from producto";
    Cursor registros = BaseDeDatos.rawQuery(sql,null);


//Recorrer la lista
    ArrayList datos = new ArrayList();


    while(registros.moveToNext())
    {
        //vista lineal
        datos.add(registros.getString(0) + "   \n   "+ registros.getString(1) + "   \n   " + registros.getString(2) );

         //sino se muestra ponemos esto TOAST para destrabar y depsues lo comentamos
          //Toast.makeText(MainActivity3.this, ""+registros.getString(0),Toast.LENGTH_LONG ).show();

    }

     //cerramos BD
    BaseDeDatos.close();


      return  datos;
}
```

//mostramos los datos en el ListView

//creamos las variables globales necesarias

```
ListView listview;


//variables para mostrar
ArrayList lista ;
ArrayAdapter adaptador;
```

//en el evento necesario cargamos los datos al ListView; en este caso en el créate del Activity

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main2);

    getSupportActionBar().setTitle("         INVENTARIO EXISTENTE");
    listview = (ListView) findViewById(R.id.ListView);



    //mostrar datos lisview
    lista = mostrarprodutos();
    adaptador = new ArrayAdapter(this, R.layout.support_simple_spinner_dropdown_item,lista);
    listview.setAdapter(adaptador);


}
```

NOTA: se puede poner unos textvie por encima del listview para mostrar como etiquetas

Seleccionar un Item del Listview

```
//Evento seleccionar un item del lisview
listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
        Toast.makeText(MainActivity3.this , ""+mostrarprodutos().get(i), Toast.LENGTH_SHORT).show();


    }
});
```