Smart Device Programming

SQLite DB With Android

Android Studio

Kotlin Android SQLite

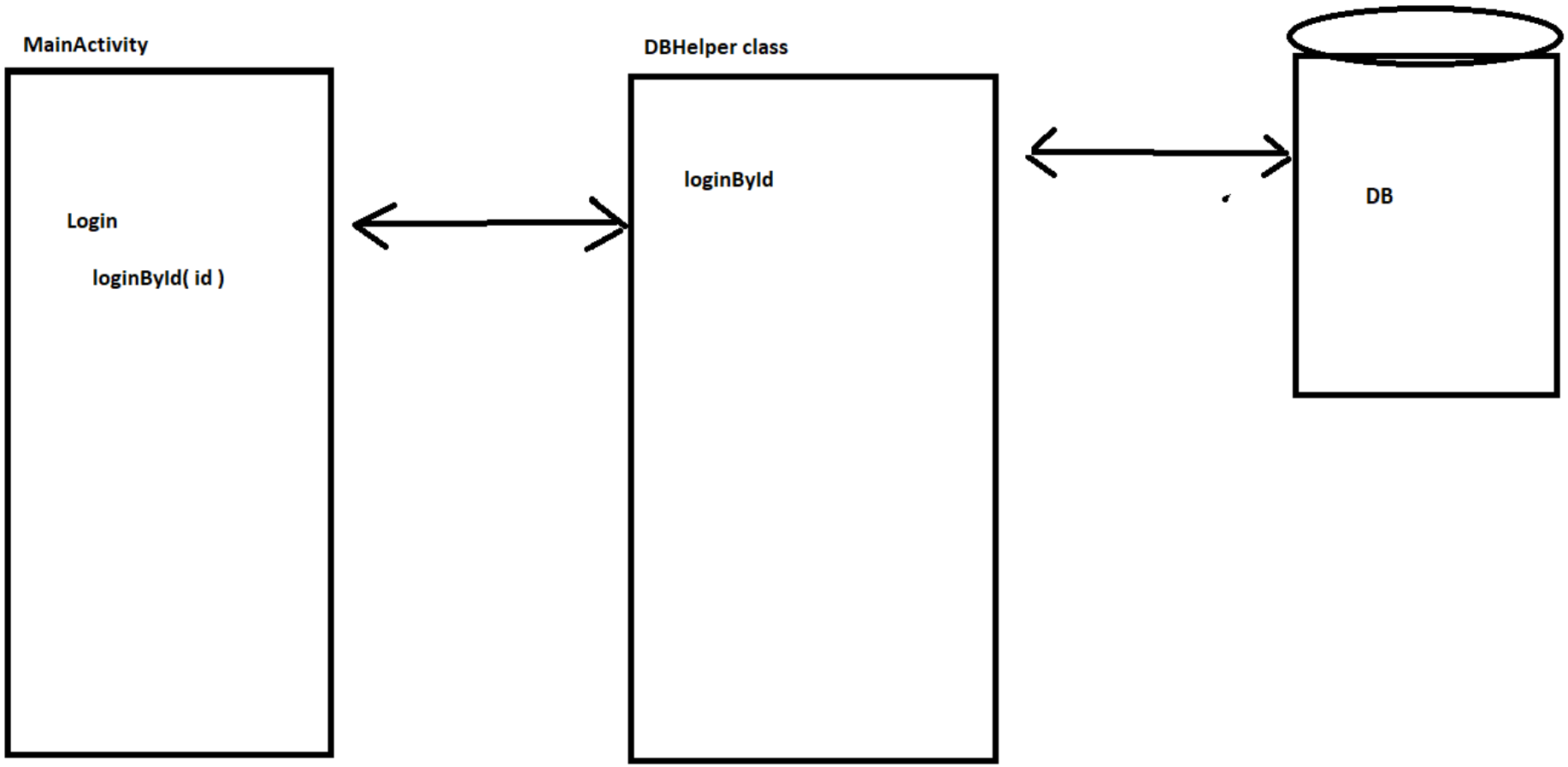
SQLite is

- an **open-source**
- **relational database**
- used to perform **database operations** on **Android** devices, such as storing, manipulating or retrieving persistent data from the database.
- By default **SQLite** database is **embedded** in android. So, there is **no need to perform any database setup** or administration task.
- The **SQLiteOpenHelper** class **provides the functionality** to use the **SQLite** database.

SQLiteOpenHelper class

- The **android.database.sqlite.SQLiteOpenHelper** class is used for database creation and version management.
- For performing any database operation,
- you have to provide the implementation of **onCreate()** and **onUpgrade()** methods of SQLiteOpenHelper class.

DBHelper Class



Android SQLite Database in Kotlin

Step By Step Implementation

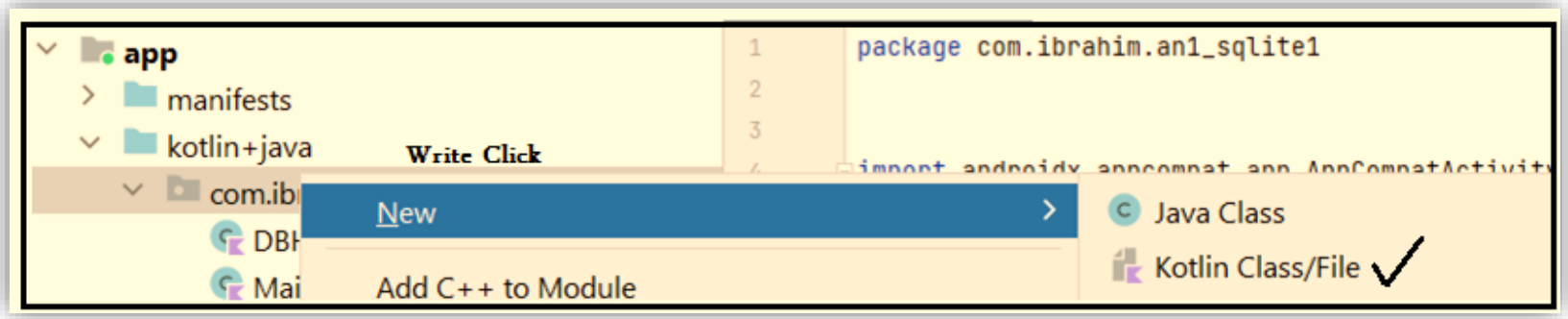
Step 1: Create a New Project (AN1_SQLite1).

Step 2: Giving permission to access the storage in the *AndroidManifest.xml* file

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Step 3: Working with the activity_main.xml file.

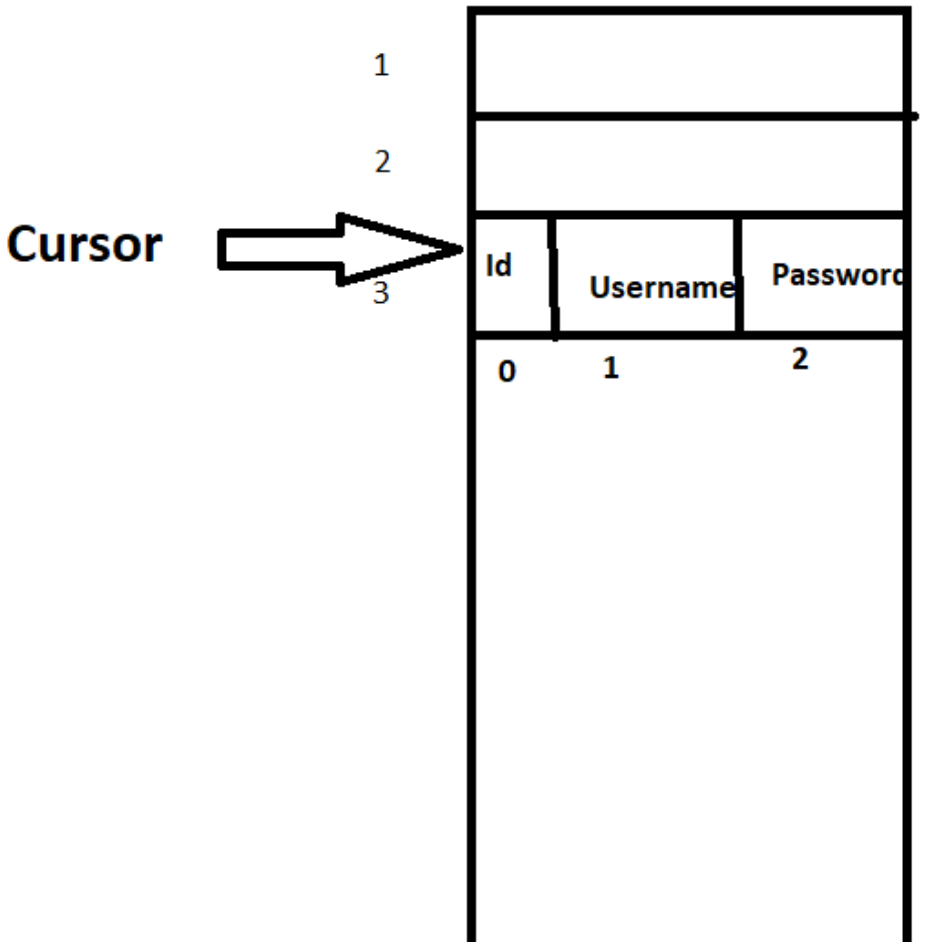
Step 4: Creating a new class for SQLite operations (**Kotlin class** and name it as **DBHelper**)



Step 5: Working with MainActivity.kt file

https://www.geeksforgeeks.org/android-sqlite-database-in-kotlin/

Cursor




```
val name = enterName.text.toString()
val age = enterAge.text.toString()
cursor = db.login(name, age)
if (cursor.count > 0) {
    cursor.moveToFirst()

    Id.append(cursor.getInt(0).toString() + "\n")
    Name.append(cursor.getString(1) + "\n")
    Age.append(cursor.getString(2) + "\n")
    cursor.close()
}else
    Toast.makeText(this, " Name is Error", Toast.LENGTH_LONG).show()
```

DBHelper class

```kotlin
package com.idekhail.an1_sqlite1
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper
class DBHelper(context: Context, factory: SQLiteDatabase.CursorFactory?) :
    SQLiteOpenHelper(context, DATABASE_NAME, factory, DATABASE_VERSION) {
    companion object{
        // variable for database name
        private val DATABASE_NAME = "USERS"
        // variable for database version
        private val DATABASE_VERSION = 1
        // variable for table name
        val TABLE_NAME = "users_table"
        //variable for id column
        val ID_COL = "id"
        // variable for name column
        val NAME_COl = "name"
        // variable for age column
        val AGE_COL = "age"
    }
    //method for creating a database by a sqlite query
    override fun onCreate(db: SQLiteDatabase) {
        val query = ("CREATE TABLE " + TABLE_NAME + " ("
                + ID_COL + " INTEGER PRIMARY KEY, " +
                NAME_COl + " TEXT," +
                AGE_COL + " TEXT" + ")")

        // method for executing our query
        db.execSQL(query)
    }
```

DBHelper class

```
32    override fun onUpgrade(db: SQLiteDatabase, p1: Int, p2: Int) {
33        // this method is to check if table already exists
34        db.execSQL( sql: "DROP TABLE IF EXISTS " + TABLE_NAME)
35        onCreate(db)
36    }
37    // This method is for adding data in our database
38    fun addName(name : String, age : String ){
39        // creating content values variable
40        val values = ContentValues()
41        // we are inserting our values
42        values.put(NAME_COl, name)
43        values.put(AGE_COL, age)
44        // creating writable variable of our database
45        val db = this.writableDatabase
46        // all values are inserted into database
47        db.insert(TABLE_NAME,  nullColumnHack: null, values)
48        // closing our database
49        db.close()
50    }
51    // get all data from our database
52    fun getName(): Cursor? {
53        // creating a readable variable of our database
54        val db = this.readableDatabase
55        // returns a cursor to read data from the database
56        return db.rawQuery( sql: "SELECT * FROM " + TABLE_NAME,  selectionArgs: null)
57    }
58}
```

Main Activity

```
1  package com.idekhail.an1_sqlite1
2
3  import android.annotation.SuppressLint
4  import androidx.appcompat.app.AppCompatActivity
5  import android.os.Bundle
6  import android.widget.Button
7  import android.widget.EditText
8  import android.widget.TextView
9  import android.widget.Toast
10
11 class MainActivity : AppCompatActivity() {
12
13     @SuppressLint("Range")
14     override fun onCreate(savedInstanceState: Bundle?) {
15         super.onCreate(savedInstanceState)
16         setContentView(R.layout.activity_main)
17
18         val addName = findViewById<Button>(R.id.addName)
19         val printName = findViewById<Button>(R.id.printName)
20         val enterName = findViewById<EditText>(R.id.enterName)
21         val enterAge = findViewById<EditText>(R.id.enterAge)
22         val Name = findViewById<TextView>(R.id.Name)
23         val Age = findViewById<TextView>(R.id.Age)
24
```

```
25         addName.setOnClickListener{ it: View!
26             // creating an object of DBHelper class
27             val db = DBHelper( context: this,  factory: null)
28
29             // get entries
30             val name = enterName.text.toString()
31             val age = enterAge.text.toString()
32
33             // calling method to add name to our database
34             db.addName(name, age)
35
36             // Toast to message on the screen
37             Toast.makeText( context: this,  text: name +
38                     " added to database", Toast.LENGTH_LONG).show()
39
40             // clearing edit texts
41             enterName.text.clear()
42             enterAge.text.clear()
43         }
```

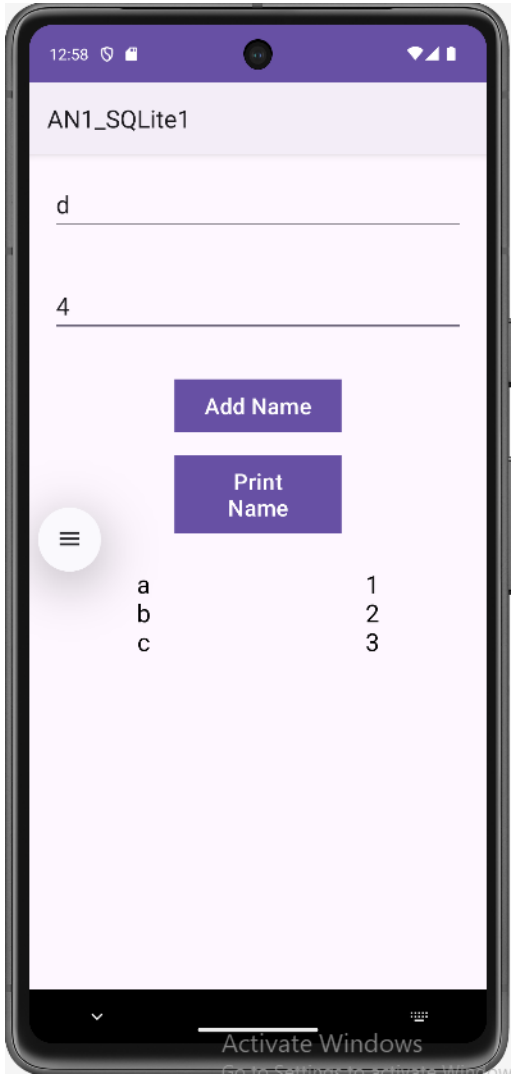
Check if name is not Empty
Show Toast massage

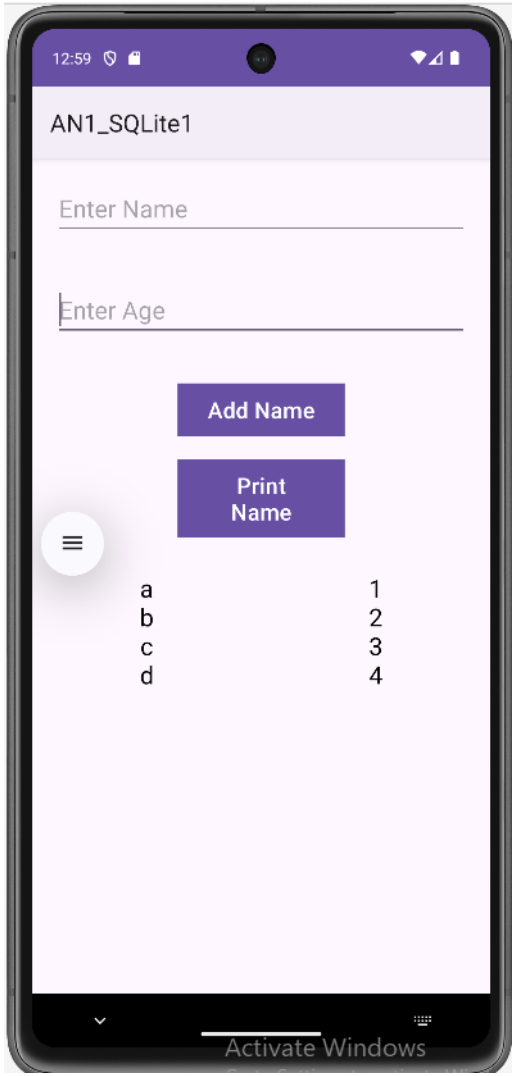
Main Activity

Check if cursor is null
Show Toast massage

```
45          printName.setOnClickListener{ it: View!
46              Name.setText("Name\n---------\n")
47              Age.setText("Age\n---------\n")
48              // creating an object of DBHelper class
49              val db = DBHelper( context: this,  factory: null)
50
51              // Calling method to get all names from our database
52              val cursor = db.getName()
53
54              // moving the cursor to first position
55              cursor!!.moveToFirst()
56              Name.append(cursor.getString(cursor.getColumnIndex(DBHelper.NAME_COl)) + "\n")
57              Age.append(cursor.getString(cursor.getColumnIndex(DBHelper.AGE_COL)) + "\n")
58
59              // moving our cursor to next position
60              while(cursor.moveToNext()){
61                  Name.append(cursor.getString(cursor.getColumnIndex(DBHelper.NAME_COl)) + "\n")
62                  Age.append(cursor.getString(cursor.getColumnIndex(DBHelper.AGE_COL)) + "\n")
63              }
64              // close our cursor
65              cursor.close()
66          }
67      }
68  }
```

Output:


12:58
AN1_SQLite1
d
4
Add Name
Print Name
a 1
b 2
c 3
Activate Windows


12:59
AN1_SQLite1
Enter Name
Enter Age
Add Name
Print Name
a 1
b 2
c 3
d 4
Activate Windows