

Lab 21: Android SQLite Database

Introduction

For tasks like storing, altering, or retrieving persistent data from the database on Android devices, SQLite is an open-source relational database. It comes pre-installed on Android. Therefore, no database setup or management tasks are required. The ability to use the SQLite database is provided by the SQLiteOpenHelper class.

Let's get Started:

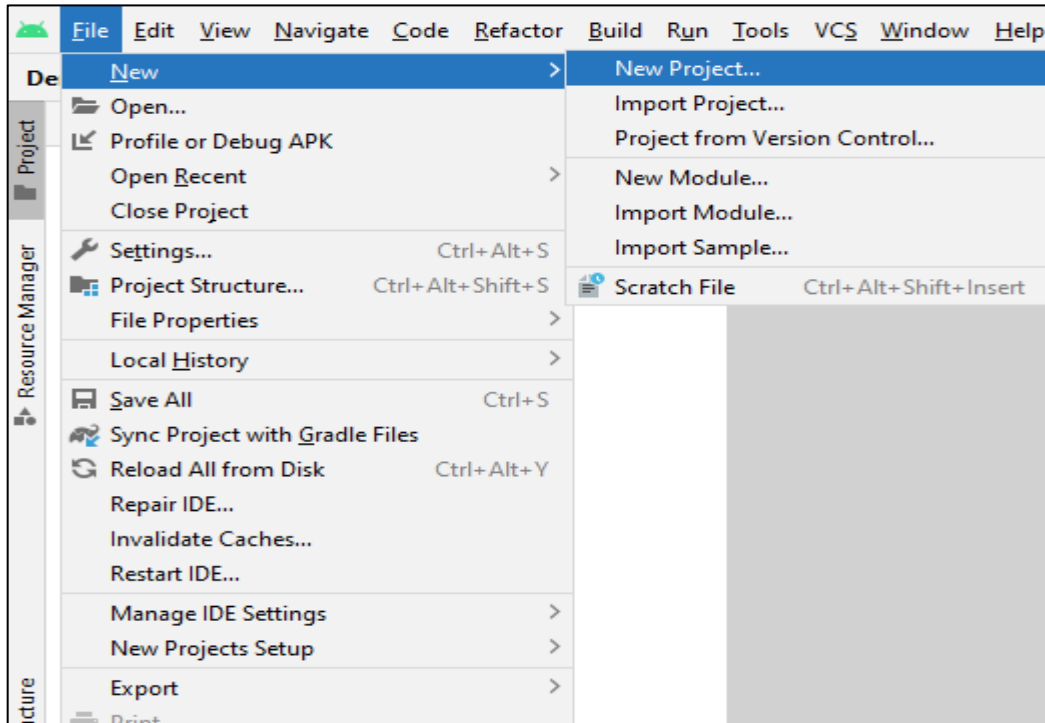
In this experiment we will develop an Android App to demonstrate the use of Android SQLite Database.

- Launching File Explorer
- access the data directory
- Look up the name of your application package in the data directory.
- Go to databases in your application package to access your database (contactsManager)
- A copy of your database can be saved.
- any tool or browser extension for SQLite. DB Browser for SQLite, for instance
- Open your database in the programme (DB Browser for SQLite) by launching it.
- You can then choose and view the data in your database depending on the tool you're using.
- To see the stored data, for instance, pick your table (contacts) from the Browse Data menu in the DB Browser for SQLite.

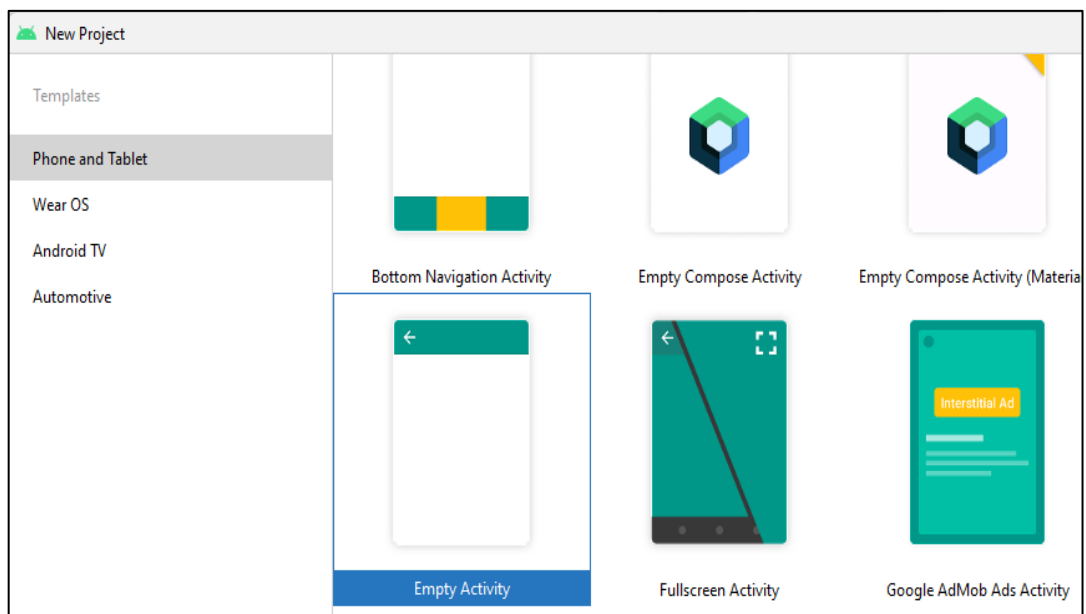
Download & Install

- [DB Browser for SQLite - Standard installer for 64-bit Windows](#)

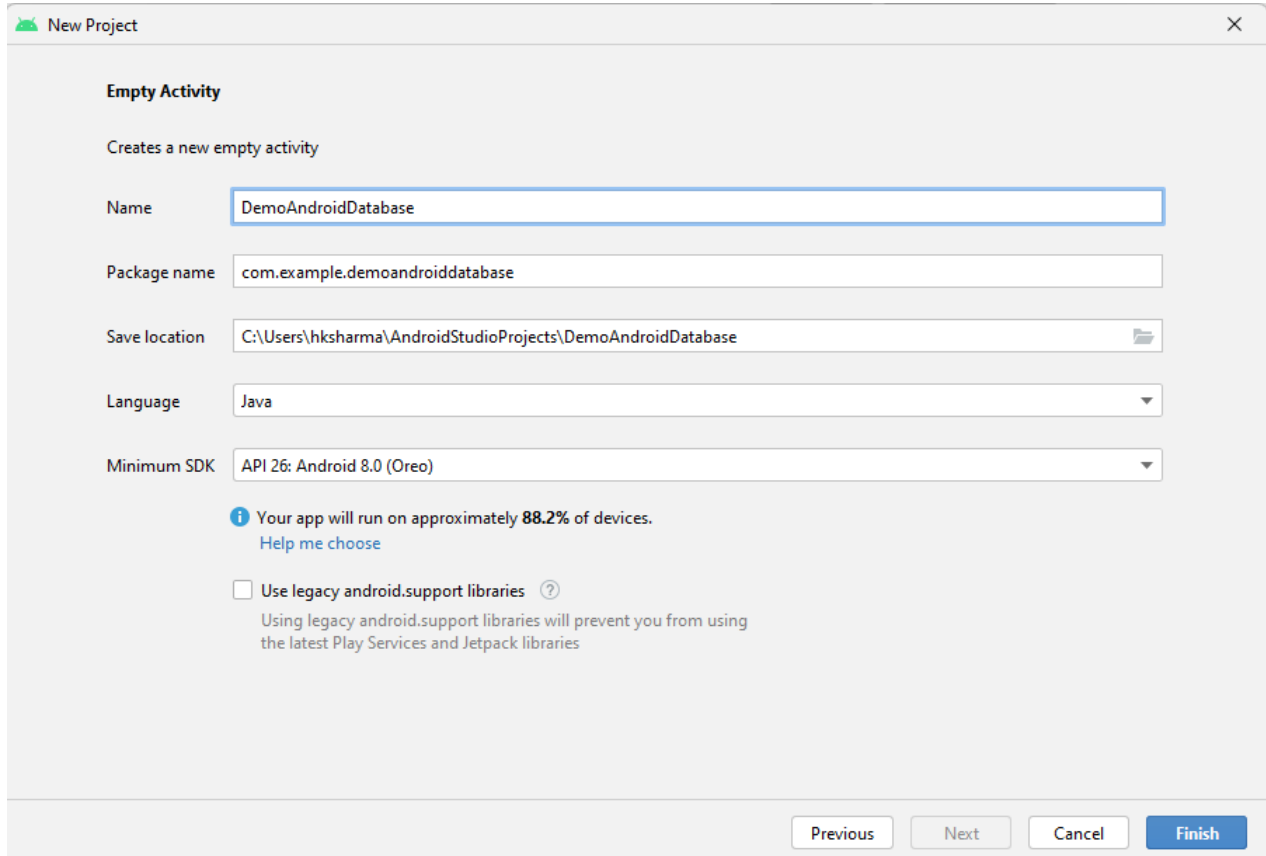
Step 1: Create a New Project in Android Studio as shown below



Step 2: Select Empty Activity as shown below



Step 3: Provide a Project Name as shown below



New Project

Empty Activity

Creates a new empty activity

Name:

Package name:

Save location:

Language:

Minimum SDK:

i Your app will run on approximately **88.2%** of devices.
[Help me choose](#)

☐ Use legacy android.support libraries **?**
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Previous Next Cancel **Finish**

Step 4: Update MainActivity.kt as per the code given below

```
package com.example.demokotlindb
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        addName.setOnClickListener{
            val db = DBHelper(this, null)
            val name = enterName.text.toString()
            val age = enterAge.text.toString()
            db.addName(name, age)
            Toast.makeText(this, name + " added to database",
```

```

Toast.LENGTH_LONG).show()
        enterName.text.clear()
        enterAge.text.clear()
    }
    printName.setOnClickListener{
        val db = DBHelper(this, null)
        val cursor = db.getName()
        cursor!!.moveToFirst()

        Name.append(cursor.getString(cursor.getColumnIndex(DBHelper.NAME_COL)) + "\n")
        Age.append(cursor.getString(cursor.getColumnIndex(DBHelper.AGE_COL)) + "\n")

        while(cursor.moveToNext()){
            Name.append(cursor.getString(cursor.getColumnIndex(DBHelper.NAME_COL)) + "\n")
            Age.append(cursor.getString(cursor.getColumnIndex(DBHelper.AGE_COL)) + "\n")
        }
        cursor.close()
    }
}
}

```

Step 5: Create DBHelper.kt as per the code given below

```

package com.example.demokotlindb
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) {

    override fun onCreate(db: SQLiteDatabase) {
        val query = ("CREATE TABLE " + TABLE_NAME + " ("
            + ID_COL + " INTEGER PRIMARY KEY, " +
            NAME_COL + " TEXT," +
            AGE_COL + " TEXT" + ")")
        db.execSQL(query)
    }

    override fun onUpgrade(db: SQLiteDatabase, p1: Int, p2: Int) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME)
        onCreate(db)
    }

    fun addName(name : String, age : String ){
        val values = ContentValues()
        values.put(NAME_COL, name)
        values.put(AGE_COL, age)
        val db = this.writableDatabase
    }
}

```

```

        db.insert(TABLE_NAME, null, values)
        db.close()
    }
    fun getName(): Cursor? {
        val db = this.readableDatabase
        return db.rawQuery("SELECT * FROM " + TABLE_NAME, null)
    }

    companion object{
        private val DATABASE_NAME = "snap"
        private val DATABASE_VERSION = 1
        val TABLE_NAME = "snap_table"
        val ID_COL = "id"
        val NAME_COL = "name"
        val AGE_COL = "age"
    }
}

```

Step 6: Update activity_main.xml for Relative Layout as per the code given below

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <!--Edit text to enter course name-->
    <EditText
        android:id="@+id/idEdtCourseName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter course Name" />

    <!--edit text to enter course duration-->
    <EditText
        android:id="@+id/idEdtCourseDuration"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter Course Duration" />

    <!--edit text to display course tracks-->
    <EditText
        android:id="@+id/idEdtCourseTracks"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter Course Tracks" />

    <!--edit text for course description-->
    <EditText

```

```

        android:id="@+id/idEdtCourseDescription"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter Course Description" />

<!--button for adding new course-->
<Button
    android:id="@+id/idBtnAddCourse"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:text="Add Course"
    android:textAllCaps="false" />

</LinearLayout>

```

Step 7: Update AndroidManifest.xml for as per the code given below

```

<?xml version="1.0" encoding="utf-8" ?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
    />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.DemoKotlinDB"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

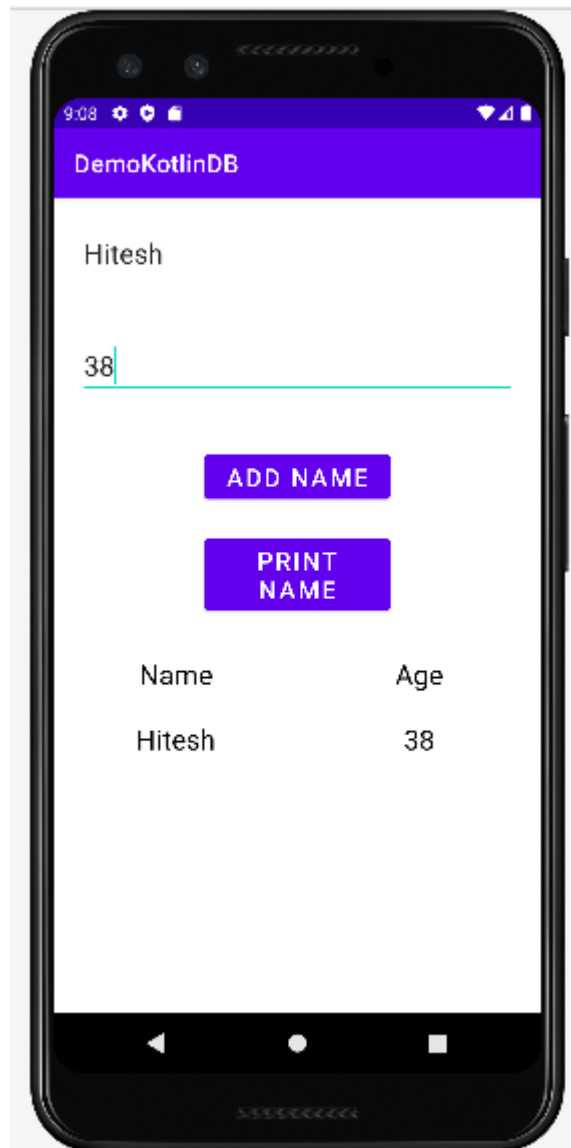
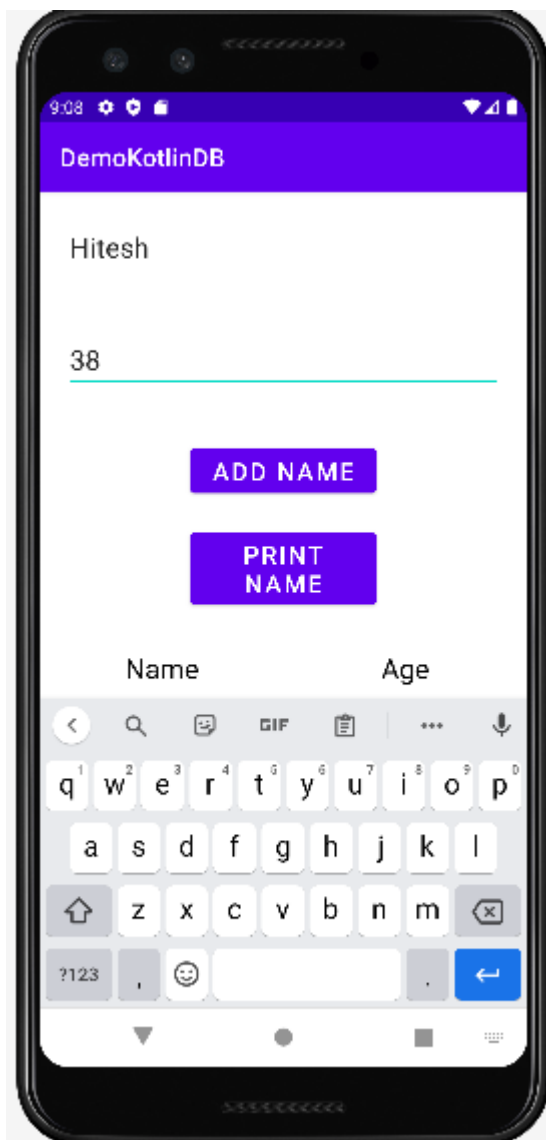
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
    </application>

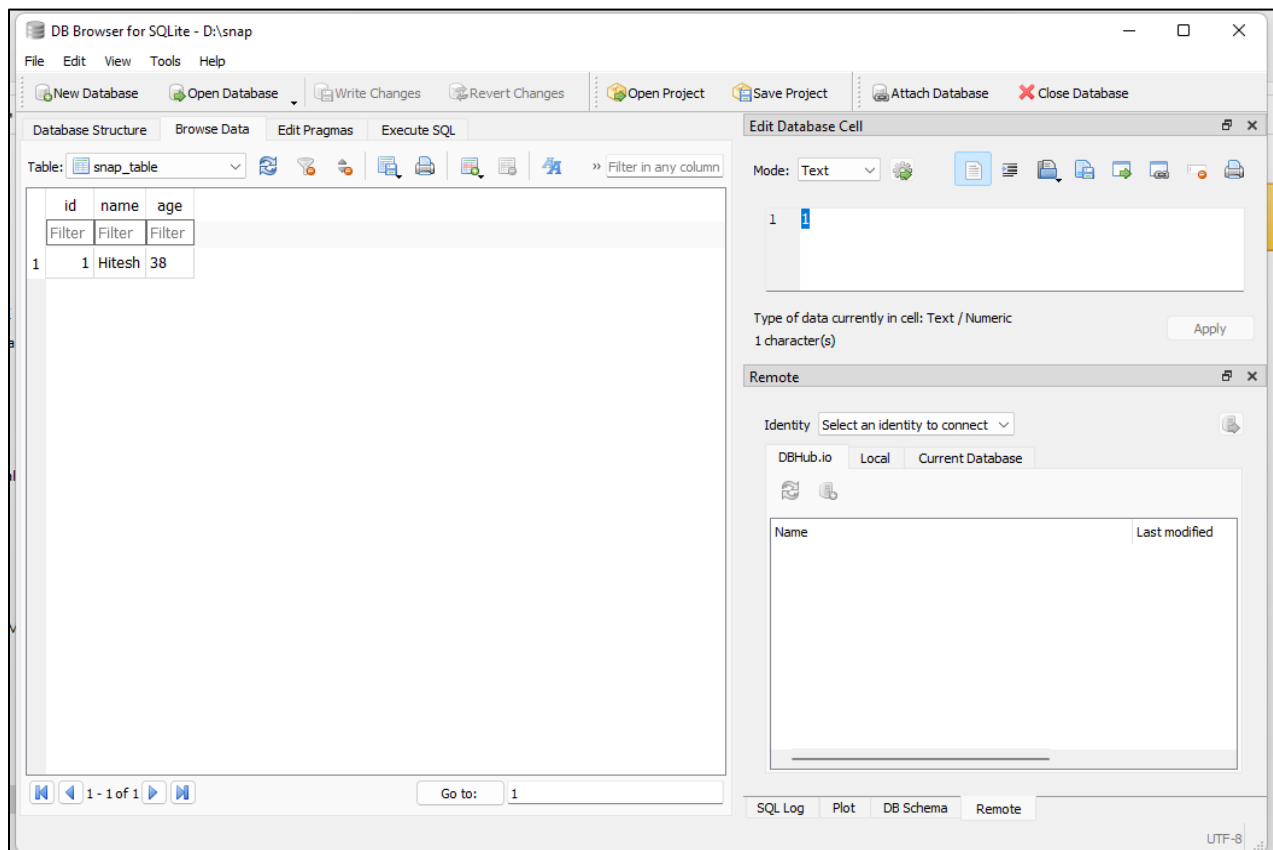
</manifest>

```

Step 8: Check Output on Android Emulator and it should look like as given below



Step 9: Check Database as given below



Voila!! We have successfully completed this lab.