DATABASE CRUD OPERATIONS

In Gradle Scripts Open the buildgradle.kts file and add this dependency.

Dependencies {

…….

```
implementation("androidx.sqlite:sqlite:2.1.0")

}
```

MainActivity.kt

```
package com.example.database

import android.os.Bundle
import android.widget.*
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

    private lateinit var dbHelper: DatabaseHelper
    private lateinit var editTextName: EditText
    private lateinit var editTextAge: EditText
    private lateinit var editTextEmail: EditText
    private lateinit var editTextPhone: EditText
    private lateinit var tableLayoutUsers: TableLayout
    private var selectedUserId: Int? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        dbHelper = DatabaseHelper(this)
        editTextName = findViewById(R.id.editTextName)
        editTextAge = findViewById(R.id.editTextAge)
        editTextEmail = findViewById(R.id.editTextEmail)
        editTextPhone = findViewById(R.id.editTextPhone)
        tableLayoutUsers = findViewById(R.id.tableLayoutUsers)

        findViewById<Button>(R.id.buttonAdd).setOnClickListener { addUser()
}
        findViewById<Button>(R.id.buttonUpdate).setOnClickListener {
updateUser() }
        findViewById<Button>(R.id.buttonDelete).setOnClickListener {
deleteUser() }

        loadUsers()
    }
```

```
    private fun addUser() {
        val name = editTextName.text.toString()
        val age = editTextAge.text.toString().toIntOrNull() ?: 0
        val email = editTextEmail.text.toString()
        val phone = editTextPhone.text.toString()

        val user = User(0, name, age, email, phone)
        dbHelper.addUser(user)
        clearInputs()
        loadUsers()
    }

    private fun updateUser() {
        selectedUserId?.let {
            val name = editTextName.text.toString()
            val age = editTextAge.text.toString().toIntOrNull() ?: 0
            val email = editTextEmail.text.toString()
            val phone = editTextPhone.text.toString()

            val user = User(it, name, age, email, phone)
            dbHelper.updateUser(user)
            clearInputs()
            loadUsers()
        }
    }

    private fun deleteUser() {
        selectedUserId?.let {
            dbHelper.deleteUser(it)
            clearInputs()
            loadUsers()
        }
    }

    private fun loadUsers() {
        tableLayoutUsers.removeViews(1, tableLayoutUsers.childCount - 1) //
Remove all rows except the header
        val users = dbHelper.getAllUsers()

        for (user in users) {
            val row = TableRow(this)
            row.setPadding(8, 8, 8, 8) // Optional: Add some padding for
aesthetics

            val nameTextView = TextView(this).apply {
                text = user.name
                layoutParams = TableRow.LayoutParams(0,
TableRow.LayoutParams.WRAP_CONTENT, 1f)
                setOnClickListener { selectUser(user.id) }
            }
            row.addView(nameTextView)

            val ageTextView = TextView(this).apply {
                text = user.age.toString()
                layoutParams = TableRow.LayoutParams(0,
TableRow.LayoutParams.WRAP_CONTENT, 1f)
            }
            row.addView(ageTextView)

            val emailTextView = TextView(this).apply {
                text = user.email
```

```
                layoutParams = TableRow.LayoutParams(0,
TableRow.LayoutParams.WRAP_CONTENT, 1f)
                setOnClickListener { selectUser(user.id) }
            }
            row.addView(emailTextView)

            val phoneTextView = TextView(this).apply {
                text = user.phone
                layoutParams = TableRow.LayoutParams(0,
TableRow.LayoutParams.WRAP_CONTENT, 1f)
                setOnClickListener { selectUser(user.id) }
            }
            row.addView(phoneTextView)

            tableLayoutUsers.addView(row)
        }
    }

    private fun selectUser(userId: Int) {
        val user = dbHelper.getUserById(userId)
        editTextName.setText(user.name)
        editTextAge.setText(user.age.toString())
        editTextEmail.setText(user.email)
        editTextPhone.setText(user.phone)
        selectedUserId = user.id
    }

    private fun clearInputs() {
        editTextName.text.clear()
        editTextAge.text.clear()
        editTextEmail.text.clear()
        editTextPhone.text.clear()
        selectedUserId = null
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Name"
        android:padding="12dp"
        android:background="@drawable/edit_text_background_color"
        android:textColor="#000000"
        android:textColorHint="#B0B0B0" />

    <EditText
        android:id="@+id/editTextAge"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:hint="Enter Age"
        android:inputType="number"
        android:padding="12dp"
        android:background="@drawable/edit_text_background_color"
        android:textColor="#000000"
        android:textColorHint="#B0B0B0" />

    <EditText
        android:id="@+id/editTextEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Email"
        android:padding="12dp"
        android:background="@drawable/edit_text_background_color"
        android:textColor="#000000"
        android:textColorHint="#B0B0B0" />

    <EditText
        android:id="@+id/editTextPhone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Phone Number"
        android:inputType="phone"
        android:padding="12dp"
        android:background="@drawable/edit_text_background_color"
        android:textColor="#000000"
        android:textColorHint="#B0B0B0" />

    <Button
        android:id="@+id/buttonAdd"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Add User"
        android:layout_marginTop="8dp"
        android:padding="12dp"
        android:backgroundTint="#03DAC5"
        android:textColor="@android:color/white" />

    <Button
        android:id="@+id/buttonUpdate"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Update User"
        android:layout_marginTop="8dp"
        android:padding="12dp"
        android:backgroundTint="#03DAC5"
        android:textColor="@android:color/white" />

    <Button
        android:id="@+id/buttonDelete"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Delete User"
        android:layout_marginTop="8dp"
        android:padding="12dp"
        android:backgroundTint="#03DAC5"
        android:textColor="@android:color/white" />

    <!-- Table Layout for displaying users -->
    <TableLayout
```

```
        android:id="@+id/tableLayoutUsers"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="16dp"
        android:stretchColumns="0,1,2,3"
        android:divider="?android:attr/listDivider"
        android:dividerHeight="1dp">

        <TableRow
            android:background="#DDDDDD">
            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="Name"
                android:textStyle="bold"
                android:padding="8dp"
                android:gravity="center"
                android:textColor="#000000" />

            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="Age"
                android:textStyle="bold"
                android:padding="8dp"
                android:gravity="center"
                android:textColor="#000000" />

            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="Email"
                android:textStyle="bold"
                android:padding="8dp"
                android:gravity="center"
                android:textColor="#000000" />

            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="Phone"
                android:textStyle="bold"
                android:padding="8dp"
                android:gravity="center"
                android:textColor="#000000" />
        </TableRow>
    </TableLayout>
</LinearLayout>
```

DatabaseHelper.kt

```
package com.example.database

import android.content.ContentValues
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper
import android.database.Cursor

class DatabaseHelper(context: Context) : SQLiteOpenHelper(context,
DATABASE_NAME, null, DATABASE_VERSION) {

    companion object {
        private const val DATABASE_NAME = "users.db"
        private const val DATABASE_VERSION = 1
        const val TABLE_USERS = "users"
        const val COLUMN_ID = "id"
        const val COLUMN_NAME = "name"
        const val COLUMN_AGE = "age"
        const val COLUMN_EMAIL = "email"
        const val COLUMN_PHONE = "phone"
    }

    override fun onCreate(db: SQLiteDatabase) {
        val createTableSQL = ("CREATE TABLE $TABLE_USERS ("
                + "$COLUMN_ID INTEGER PRIMARY KEY AUTOINCREMENT, "
                + "$COLUMN_NAME TEXT, "
                + "$COLUMN_AGE INTEGER, "
                + "$COLUMN_EMAIL TEXT, "
                + "$COLUMN_PHONE TEXT" + ");")
        db.execSQL(createTableSQL)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion:
Int) {
        db.execSQL("DROP TABLE IF EXISTS $TABLE_USERS")
        onCreate(db)
    }

    // Add a new user
    fun addUser(user: User): Long {
        val db = writableDatabase
        val values = ContentValues()
        values.put(COLUMN_NAME, user.name)
        values.put(COLUMN_AGE, user.age)
        values.put(COLUMN_EMAIL, user.email)
        values.put(COLUMN_PHONE, user.phone)
        return db.insert(TABLE_USERS, null, values)
    }

    // Update an existing user
    fun updateUser(user: User): Int {
        val db = writableDatabase
        val values = ContentValues()
        values.put(COLUMN_NAME, user.name)
        values.put(COLUMN_AGE, user.age)
        values.put(COLUMN_EMAIL, user.email)
        values.put(COLUMN_PHONE, user.phone)
        return db.update(TABLE_USERS, values, "$COLUMN_ID = ?",
arrayOf(user.id.toString()))
    }
```

```
    // Delete a user
    fun deleteUser(id: Int): Int {
        val db = writableDatabase
        return db.delete(TABLE_USERS, "$COLUMN_ID = ?",
arrayOf(id.toString()))
    }

    // Get user by ID
    fun getUserById(id: Int): User {
        val db = readableDatabase
        val cursor = db.query(TABLE_USERS, null, "$COLUMN_ID = ?",
arrayOf(id.toString()), null, null, null)

        return if (cursor != null && cursor.moveToFirst()) {
            val name = cursor.getString(cursor.getColumnIndex(COLUMN_NAME))
            val age = cursor.getInt(cursor.getColumnIndex(COLUMN_AGE))
            val email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL))
            val phone =
cursor.getString(cursor.getColumnIndex(COLUMN_PHONE))
            User(id, name, age, email, phone)
        } else {
            throw Exception("User not found")
        }.also {
            cursor.close()
            db.close()
        }
    }

    // Get all users
    fun getAllUsers(): List<User> {
        val users = mutableListOf<User>()
        val db = readableDatabase
        val cursor = db.query(TABLE_USERS, null, null, null, null, null,
null)

        if (cursor != null && cursor.moveToFirst()) {
            do {
                val id = cursor.getInt(cursor.getColumnIndex(COLUMN_ID))
                val name =
cursor.getString(cursor.getColumnIndex(COLUMN_NAME))
                val age = cursor.getInt(cursor.getColumnIndex(COLUMN_AGE))
                val email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL))
                val phone =
cursor.getString(cursor.getColumnIndex(COLUMN_PHONE))
                users.add(User(id, name, age, email, phone))
            } while (cursor.moveToNext())
        }
        cursor.close()
        db.close()
        return users
    }
}
```

user.kt

```
package com.example.database

data class User(
    val id: Int,
    val name: String,
    val age: Int,
    val email: String,  // New field
    val phone: String   // New field
)
```

In drawable create a new file edit_background_text_color.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#FFFFFF" />
    <corners android:radius="8dp" />
    <stroke
        android:width="1dp"
        android:color="#CCCCCC" />
</shape>
```

In values Colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="colorPrimary">#6200EE</color>
    <color name="colorPrimaryDark">#3700B3</color>
    <color name="colorAccent">#03DAC5</color>
</resources>
```