Lab 7
Implementation of SQLite Database

In android, we have different storage options such as shared preferences, internal storage, external storage, SQLite storage, etc. to store and retrieve the application data based on our requirements.

SQLite is an open-source lightweight relational database management system (RDBMS) to perform database operations, such as storing, updating, retrieving data from the database. To know more about SQLite, check this SQLite Tutorial with Examples.

Generally, in our android applications Shared Preferences, Internal Storage and External Storage options are useful to store and maintain a small amount of data. In case, if we want to deal with large amounts of data, then **SQLite database** is the preferable option to store and maintain the data in a structured format.

By default, Android comes with built-in SQLite Database support so we don't need to do any configurations.

Just like we save the files on the device's **internal storage**, Android stores our database in a private disk space that's associated with our application and the data is secure, because by default this area is not accessible to other applications.

The package **android.database.sqlite** contains all the required APIs to use an SQLite database in our android applications.

Create Database and Tables using SQLite Helper

In android, by using **SQLiteOpenHelper** class we can easily create the required database and tables for our application. To use **SQLiteOpenHelper**, we need to create a subclass that overrides the **onCreate()** and **onUpgrade()** call-back methods.

Method	Description
onCreate()	This method is called only once throughout the application after the database is created and the table creation statements can be written in this method.
onUpgrade()	This method is called whenever there is an updation in the database like modifying the table structure, adding constraints to the database, etc.

Following is the code snippet of creating the database and tables using the **SQLiteOpenHelper** class in our android application.

```
public class DbHandler extends SQLiteOpenHelper {
   private static final int DB_VERSION = 1;
   private static final String DB_NAME = "usersdb";
   private static final String TABLE_Users = "userdetails";
   private static final String KEY_ID = "id";
   private static final String KEY_NAME = "name";
   private static final String KEY_LOC = "location";
   private static final String KEY_DESG = "designation";
   public DbHandler(Context context){
      super(context,DB_NAME, null, DB_VERSION);
   }
   @Override
   public void onCreate(SQLiteDatabase db){
      String CREATE_TABLE = "CREATE TABLE " + TABLE_Users + "("
            + KEY_ID + " INTEGER PRIMARY KEY AUTOINCREMENT," + KEY_NAME + "
TEXT,"
            + KEY_LOC + " TEXT,"
            + KEY_DESG + " TEXT"+ ")";
      db.execSQL(CREATE_TABLE);
   }
   @Override
   public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion){
      // Drop older table if exist
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_Users);
      // Create tables again
      onCreate(db);
   }
}
```

Insert Data into SQLite Database

In android, we can insert data into the SQLite database by passing **ContentValues** to **insert()** method.

```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
//Create a new map of values, where column names are the keys
ContentValues cValues = new ContentValues();
cValues.put(KEY_NAME, name);
cValues.put(KEY_LOC, location);
cValues.put(KEY_DESG, designation);
// Insert the new row, returning the primary key value of the new row
long newRowId = db.insert(TABLE_Users,null, cValues);
```

Read the Data from SQLite Database

In android, we can read the data from the SQLite database using the **query()** method in android applications.

```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
Cursor cursor = db.query(TABLE_Users, new String[]{KEY_NAME, KEY_LOC, KEY_DESG},
KEY_ID+ "=?",new String[]{String.valueOf(userid)},null, null, null, null);
```

Update Data in SQLite Database

In android, we can update the data in the SQLite database using an **update()** method in android applications.

```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
ContentValues cVals = new ContentValues();
cVals.put(KEY_LOC, location);
cVals.put(KEY_DESG, designation);
int count = db.update(TABLE_Users, cVals, KEY_ID+" = ?",new String[]{String.valueOf(id)});
```

Delete Data from SQLite Database

In android, we can delete data from the SQLite database using the **delete()** method in android applications.

```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
db.delete(TABLE_Users, KEY_ID+" = ?",new String[]{String.valueOf(userid)});
```

Android SQLite Database Example

create a class file **DbHandler.java** in **\java\com.example.sqliteexample** path to implement SQLite database related activities for that right-click on your application folder → Go to **New** → select **Java Class** and give name as **DbHandler.java**.

Once we create a new class file **DbHandler.java**, open it and write the code like as shown below

DbHandler.java

```
package com.example.sqliteexample;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
```

```
import android.database.sqlite.SQLiteOpenHelper;
import java.util.ArrayList;
import java.util.HashMap;

public class DbHandler extends SQLiteOpenHelper {
   private static final int DB_VERSION = 1;
   private static final String DB_NAME = "usersdb";
   private static final String TABLE_Users = "userdetails";
   private static final String KEY_ID = "id";
   private static final String KEY_NAME = "name";
   private static final String KEY_LOC = "location";
   private static final String KEY_DESG = "designation";


public DbHandler(Context context){
      super(context,DB_NAME, null, DB_VERSION);
   }
   @Override
   public void onCreate(SQLiteDatabase db){
      String CREATE_TABLE = "CREATE TABLE " + TABLE_Users + "("
            + KEY_ID + " INTEGER PRIMARY KEY AUTOINCREMENT," + KEY_NAME + "
TEXT,"
            + KEY_LOC + " TEXT,"
            + KEY_DESG + " TEXT"+ ")";
      db.execSQL(CREATE_TABLE);
   }
   @Override
   public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion){
      // Drop older table if exist
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_Users);
      // Create tables again
      onCreate(db);
   }

   // Adding new User Details
   void insertUserDetails(String name, String location, String designation){
      //Get the Data Repository in write mode
      SQLiteDatabase db = this.getWritableDatabase();
      //Create a new map of values, where column names are the keys
      ContentValues cValues = new ContentValues();
      cValues.put(KEY_NAME, name);
      cValues.put(KEY_LOC, location);
      cValues.put(KEY_DESG, designation);
      // Insert the new row, returning the primary key value of the new row
      long newRowId = db.insert(TABLE_Users,null, cValues);
      db.close();
```

```
    }
    // Get User Details
    public ArrayList<HashMap<String, String>> GetUsers(){
        SQLiteDatabase db = this.getWritableDatabase();
        ArrayList<HashMap<String, String>> userList = new ArrayList<>();
        String query = "SELECT name, location, designation FROM "+ TABLE_Users;
        Cursor cursor = db.rawQuery(query,null);
        while (cursor.moveToNext()){
            HashMap<String,String> user = new HashMap<>();
            user.put("name",cursor.getString(cursor.getColumnIndex(KEY_NAME)));
            user.put("designation",cursor.getString(cursor.getColumnIndex(KEY_DESG)));
            user.put("location",cursor.getString(cursor.getColumnIndex(KEY_LOC)));
            userList.add(user);
        }
        return  userList;
    }
    // Get User Details based on userid
    public ArrayList<HashMap<String, String>> GetUserByUserId(int userid){
        SQLiteDatabase db = this.getWritableDatabase();
        ArrayList<HashMap<String, String>> userList = new ArrayList<>();
        String query = "SELECT name, location, designation FROM "+ TABLE_Users;
        Cursor cursor = db.query(TABLE_Users, new String[]{KEY_NAME, KEY_LOC, KEY_DESG},
KEY_ID+ "=?",new String[]{String.valueOf(userid)},null, null, null, null);
        if (cursor.moveToNext()){
            HashMap<String,String> user = new HashMap<>();
            user.put("name",cursor.getString(cursor.getColumnIndex(KEY_NAME)));
            user.put("designation",cursor.getString(cursor.getColumnIndex(KEY_DESG)));
            user.put("location",cursor.getString(cursor.getColumnIndex(KEY_LOC)));
            userList.add(user);
        }
        return  userList;
    }
    // Delete User Details
    public void DeleteUser(int userid){
        SQLiteDatabase db = this.getWritableDatabase();
        db.delete(TABLE_Users, KEY_ID+" = ?",new String[]{String.valueOf(userid)});
        db.close();
    }
    // Update User Details
    public int UpdateUserDetails(String location, String designation, int id){
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues cVals = new ContentValues();
        cVals.put(KEY_LOC, location);
        cVals.put(KEY_DESG, designation);
        int count = db.update(TABLE_Users, cVals, KEY_ID+" = ?",new String[]{String.valueOf(id)});
        return  count;
```

```
    }
}
```

Now open activity_main.xml file from \res\layout folder path and write the code like as shown below.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/fstTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="150dp"
        android:text="Name" />
    <EditText
        android:id="@+id/txtName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:ems="10"/>
    <TextView
        android:id="@+id/secTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Location"
        android:layout_marginLeft="100dp" />
    <EditText
        android:id="@+id/txtLocation"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:ems="10" />
    <TextView
        android:id="@+id/thirdTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Designation"
        android:layout_marginLeft="100dp" />
    <EditText
        android:id="@+id/txtDesignation"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:ems="10" />
    <Button
        android:id="@+id/btnSave"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:text="Save" />
</LinearLayout>
```

Now we will create another layout resource file **details.xml** in **\res\layout** path to show the details in custom listview from SQLite Database for that right click on your layout folder → Go to **New** → select **Layout Resource File** and give name as **details.xml**.

details.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <ListView
        android:id="@+id/user_list"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:dividerHeight="1dp" />
    <Button
        android:id="@+id/btnBack"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="20dp"
        android:text="Back" />
</LinearLayout>
```

Create an another layout file (list_row.xml) in /res/layout folder to show the data in listview, for that right click on layout folder → add new Layout resource file → Give name as list_row.xml and write the code like as shown below.

list_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
```

```
    android:padding="5dip" >
    <TextView
        android:id="@+id/name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:textSize="17dp" />
    <TextView
        android:id="@+id/designation"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/name"
        android:layout_marginTop="7dp"
        android:textColor="#343434"
        android:textSize="14dp" />
    <TextView
        android:id="@+id/location"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/designation"
        android:layout_alignBottom="@+id/designation"
        android:layout_alignParentRight="true"
        android:textColor="#343434"
        android:textSize="14dp" />
</RelativeLayout>
```

Now open your main activity file MainActivity.java from \java\com.example.sqliteexample path and write the code like as shown below

MainActivity.java

```
package com.example.sqliteexample;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    EditText name, loc, desig;
    Button saveBtn;
    Intent intent;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        name = (EditText)findViewById(R.id.txtName);
        loc = (EditText)findViewById(R.id.txtLocation);
        desig = (EditText)findViewById(R.id.txtDesignation);
        saveBtn = (Button)findViewById(R.id.btnSave);
        saveBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String username = name.getText().toString()+"\n";
                String location = loc.getText().toString();
                String designation = desig.getText().toString();
                DbHandler dbHandler = new DbHandler(MainActivity.this);
                dbHandler.insertUserDetails(username,location,designation);
                intent = new Intent(MainActivity.this,DetailsActivity.class);
                startActivity(intent);
                Toast.makeText(getApplicationContext(), "Details Inserted
Successfully",Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Now we will create another activity file DetailsActivity.java in \java\com.example.sqliteexample path to show the details from the SQLite database for that right-click on your application folder → Go to New → select Java Class and give name as DetailsActivity.java.

Once we create a new activity file **DetailsActivity.java**, open it and write the code like as shown below

DetailsActivity.java

```
package com.example.sqliteexample;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import java.util.ArrayList;
import java.util.HashMap;

public class DetailsActivity extends AppCompatActivity {
    Intent intent;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.details);
        DbHandler db = new DbHandler(this);
        ArrayList<HashMap<String, String>> userList = db.GetUsers();
        ListView lv = (ListView) findViewById(R.id.user_list);
        ListAdapter adapter = new SimpleAdapter(DetailsActivity.this, userList, R.layout.list_row,new
String[]{"name","designation","location"}, new int[]{R.id.name, R.id.designation, R.id.location});
        lv.setAdapter(adapter);
        Button back = (Button)findViewById(R.id.btnBack);
        back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                intent = new Intent(DetailsActivity.this,MainActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

Exercise

1. Complete CRUD operation in above application.

2. Create an application which will handle Student Details.

   - Create a database with name “MyDb”
   - Create a student table with id, rollno, name and marks.
   - Create a screen to allow user to input student details, and store the details to table. Display error message if data is empty.
   - Display all students entered.
   - Allow the user to edit or remove the student