

## Experiment 11

Develop an android application to store data in the SQLite database.

Activity\_main.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#FFFFFF"
8.     android:padding="10dp"
9.     tools:context=".MainActivity">
10.
11.     <TextView
12.         android:id="@+id/appTitle"
13.         android:layout_width="0dp"
14.         android:layout_height="wrap_content"
15.         android:fontFamily="@font/roboto"
16.         android:text="@string/appTitle_txt"
17.         android:textAlignment="center"
18.         android:textSize="24sp"
19.         app:layout_constraintLeft_toLeftOf="parent"
20.         app:layout_constraintRight_toRightOf="parent"
21.         app:layout_constraintTop_toTopOf="parent" />
22.
23.     <TextView
24.         android:id="@+id/lb_activity_name"
25.         style="@style/heading"
26.         android:layout_width="80dp"
27.         android:layout_height="wrap_content"
28.         android:text="@string/lb_act_name_hint"
29.         app:layout_constraintBaseline_toBaselineOf="@+id/et_activity_name"
30.         app:layout_constraintEnd_toStartOf="@+id/et_activity_name"
31.         app:layout_constraintHorizontal_bias="0.5"
32.         app:layout_constraintHorizontal_chainStyle="packed"
33.         app:layout_constraintStart_toStartOf="parent" />
34.
35.     <TextView
36.         android:id="@+id/lb_activity_txt"
37.         style="@style/heading"
38.         android:layout_width="80dp"
39.         android:layout_height="wrap_content"
40.         android:text="@string/lb_actv_txt_hint"
41.         app:layout_constraintBaseline_toBaselineOf="@+id/et_activity_desp"
42.         app:layout_constraintEnd_toStartOf="@+id/et_activity_desp"
43.         app:layout_constraintHorizontal_bias="0.5"
44.         app:layout_constraintHorizontal_chainStyle="packed"
45.         app:layout_constraintStart_toStartOf="parent" />
46.
47.     <TextView
48.         android:id="@+id/lb_activity_deadline"
49.         style="@style/heading"
50.         android:layout_width="80dp"
51.         android:layout_height="wrap_content"
52.         android:text="@string/lb_act_time_text"
53.         app:layout_constraintBaseline_toBaselineOf="@+id/et_set_deadline"
54.         app:layout_constraintEnd_toStartOf="@+id/et_set_deadline"
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
55.         app:layout_constraintHorizontal_bias="0.5"
56.         app:layout_constraintHorizontal_chainStyle="packed"
57.         app:layout_constraintStart_toStartOf="parent" />
58.
59.     <EditText
60.         android:id="@+id/et_activity_name"
61.         android:layout_width="0dp"
62.         android:layout_height="wrap_content"
63.         android:layout_marginTop="16dp"
64.         android:ems="10"
65.         android:fontFamily="@font/roboto"
66.         android:hint="@string/ed_act_name_hint"
67.         android:inputType="textPersonName"
68.         app:layout_constraintEnd_toEndOf="parent"
69.         app:layout_constraintHorizontal_bias="0.5"
70.         app:layout_constraintStart_toEndOf="@+id/lb_activity_name"
71.         app:layout_constraintTop_toBottomOf="@+id/appTitle" />
72.
73.     <EditText
74.         android:id="@+id/et_activity_desp"
75.         android:layout_width="0dp"
76.         android:layout_height="wrap_content"
77.         android:layout_marginTop="32dp"
78.         android:ems="10"
79.         android:fontFamily="@font/roboto"
80.         android:gravity="start|top"
81.         android:hint="@string/ed_act_desp_hint"
82.         android:inputType="textMultiline"
83.         app:layout_constraintEnd_toEndOf="parent"
84.         app:layout_constraintHorizontal_bias="0.5"
85.         app:layout_constraintStart_toEndOf="@+id/lb_activity_txt"
86.         app:layout_constraintTop_toBottomOf="@+id/et_activity_name" />
87.
88.     <EditText
89.         android:id="@+id/et_set_deadline"
90.         android:layout_width="0dp"
91.         android:layout_height="wrap_content"
92.         android:layout_marginTop="32dp"
93.         android:ems="10"
94.         android:fontFamily="@font/roboto"
95.         android:hint="@string/ed_act_date_hint"
96.         android:inputType="date"
97.         app:layout_constraintEnd_toEndOf="parent"
98.         app:layout_constraintHorizontal_bias="0.5"
99.         app:layout_constraintStart_toEndOf="@+id/lb_activity_deadline"
100.        app:layout_constraintTop_toBottomOf="@+id/et_activity_desp" />
101.
102.     <Button
103.         android:id="@+id/btn_add_activity"
104.         android:layout_width="wrap_content"
105.         android:layout_height="wrap_content"
106.         android:fontFamily="@font/roboto"
107.         android:text="@string/btn_add_act"
108.         app:layout_constraintBaseline_toBaselineOf="@+id/btn_view_activities"
109.         app:layout_constraintEnd_toStartOf="@+id/btn_view_activities"
110.         app:layout_constraintHorizontal_bias="0.5"
111.         app:layout_constraintStart_toStartOf="parent" />
112.
113.     <Button
114.         android:id="@+id/btn_view_activities"
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
115.         android:layout_width="wrap_content"
116.         android:layout_height="wrap_content"
117.         android:layout_marginTop="16dp"
118.         android:text="@string/btn_view_act"
119.         app:layout_constraintEnd_toEndOf="parent"
120.         app:layout_constraintHorizontal_bias="0.5"
121.         app:layout_constraintStart_toEndOf="@+id/btn_add_activity"
122.         app:layout_constraintTop_toBottomOf="@+id/et_set_deadline" />
123.
124.         <ListView
125.             android:id="@+id/lv_allActivities"
126.             android:layout_width="0dp"
127.             android:layout_height="0dp"
128.             android:layout_marginTop="16dp"
129.             android:cacheColorHint="#FFFFFF"
130.             android:divider="@android:color/background_light"
131.             android:dividerHeight="5dp"
132.             app:layout_constraintBottom_toBottomOf="parent"
133.             app:layout_constraintEnd_toEndOf="parent"
134.             app:layout_constraintHorizontal_bias="0.533"
135.             app:layout_constraintStart_toStartOf="parent"
136.             app:layout_constraintTop_toBottomOf="@+id/btn_view_activities"
137.             app:layout_constraintVertical_bias="0.647" />
138.
139.     </androidx.constraintlayout.widget.ConstraintLayout>
```

### Acitivity\_style\_item.xml

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.      xmlns:app="http://schemas.android.com/apk/res-auto"
4.      xmlns:tools="http://schemas.android.com/tools"
5.      android:layout_width="match_parent"
6.      android:layout_height="wrap_content"
7.      android:background="#E8E1F6">
8.
9.      <TextView
10.         android:id="@+id/tv_act_title"
11.         android:layout_width="0dp"
12.         android:layout_height="wrap_content"
13.         android:fontFamily="@font/roboto"
14.         android:hint="@string/lb_act_name_hint"
15.         android:paddingStart="5dp"
16.         android:textColor="#000000"
17.         android:textSize="18sp"
18.         android:textStyle="bold"
19.         app:layout_constraintBottom_toTopOf="@+id/tv_act_time"
20.         app:layout_constraintEnd_toEndOf="@+id/tv_act_time"
21.         app:layout_constraintStart_toStartOf="parent"
22.         app:layout_constraintTop_toTopOf="parent"
23.         app:layout_constraintVertical_chainStyle="packed" />
24.
25.      <TextView
26.         android:id="@+id/tv_act_info"
27.         android:layout_width="0dp"
28.         android:layout_height="wrap_content"
29.         android:fontFamily="@font/roboto"
30.         android:hint="@string/lb_actv_txt_hint"
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
31.         android:paddingStart="5dp"
32.         android:paddingEnd="2dp"
33.         android:textColor="#000000"
34.         android:textSize="15sp"
35.         app:layout_constraintEnd_toStartOf="@+id/btn_delete"
36.         app:layout_constraintStart_toStartOf="parent"
37.         app:layout_constraintTop_toBottomOf="@+id/tv_act_time" />
38.
39.     <TextView
40.         android:id="@+id/tv_act_time"
41.         android:layout_width="0dp"
42.         android:layout_height="wrap_content"
43.         android:fontFamily="@font/roboto"
44.         android:hint="@string/lb_act_time_text"
45.         android:paddingStart="5dp"
46.         android:textColor="#000000"
47.         android:textSize="16sp"
48.         android:textStyle="italic"
49.         app:layout_constraintBottom_toTopOf="@+id/tv_act_info"
50.         app:layout_constraintEnd_toEndOf="@+id/tv_act_info"
51.         app:layout_constraintStart_toStartOf="parent"
52.         app:layout_constraintTop_toBottomOf="@+id/tv_act_title" />
53.
54.     <Button
55.         android:id="@+id/btn_delete"
56.         android:layout_width="wrap_content"
57.         android:layout_height="wrap_content"
58.         android:layout_marginStart="8dp"
59.         android:layout_marginTop="8dp"
60.         android:layout_marginEnd="16dp"
61.         android:layout_marginBottom="8dp"
62.         android:background="#3949AB"
63.         android:fontFamily="@font/roboto"
64.         android:text="@string/btn_delete_txt"
65.         android:textAlignment="center"
66.         android:textColor="#FFFFFF"
67.         android:textStyle="bold|italic"
68.         app:layout_constraintBottom_toBottomOf="parent"
69.         app:layout_constraintEnd_toEndOf="parent"
70.         app:layout_constraintStart_toEndOf="@+id/tv_act_title"
71.         app:layout_constraintTop_toTopOf="parent" />
72.
73. </androidx.constraintlayout.widget.ConstraintLayout>
```

## MainActivity.java

```
1. package com.example.sqlitedb_basix;
2.
3. import android.os.Bundle;
4. import android.view.View;
5. import android.widget.Button;
6. import android.widget.EditText;
7. import android.widget.ListView;
8. import android.widget.Toast;
9.
10. import androidx.appcompat.app.AppCompatActivity;
11.
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
12. public class MainActivity extends AppCompatActivity {
13.
14.     EditText et_act_name, et_act_info, et_act_date;
15.     Button btn_add_act, btn_view_acts;
16.     ListView activityList;
17.     LayoutAdapter customAdapter;
18.     DatabaseHelper db_helper;
19.
20.     @Override
21.     protected void onCreate(Bundle savedInstanceState) {
22.         super.onCreate(savedInstanceState);
23.         setContentView(R.layout.activity_main);
24.
25.         //db initialization
26.         db_helper = new DatabaseHelper(MainActivity.this);
27.
28.         //view binding
29.         et_act_name = findViewById(R.id.et_activity_name);
30.         et_act_info = findViewById(R.id.et_activity_desp);
31.         et_act_date = findViewById(R.id.et_set_deadline);
32.         btn_add_act = findViewById(R.id.btn_add_activity);
33.         activityList = findViewById(R.id.lv_allActivities);
34.         btn_view_acts = findViewById(R.id.btn_view_activities);
35.
36.         //list view initialization
37.         ShowAllActivities();
38.
39.         btn_add_act.setOnClickListener(new View.OnClickListener() {
40.             @Override
41.             public void onClick(View view) {
42.                 if(et_act_name.length() != 0 && et_act_info.length() != 0 && et_act_date.length() != 0) {
43.                     try {
44.                         UserActivityModel u1 = new UserActivityModel(-
45. 1, et_act_name.getText().toString(), et_act_info.getText().toString(), et_act_date.getText().toString());
46.                         boolean insertStatus = db_helper.addActivity(u1);
47.                         Toast.makeText(getApplicationContext(), "Insert Status: " + insertStatus, Toast.LENGTH_LONG).show();
48.                         ShowAllActivities();
49.                         clearAllFields();
50.                     } catch (Exception e) {
51.                         Toast.makeText(getApplicationContext(), "Error creating activity!", Toast.LENGTH_LONG).show();
52.                     }
53.                 }
54.                 else {
55.                     Toast.makeText(getApplicationContext(), "All fields are required!", Toast.LENGTH_LONG).show();
56.                 }
57.             }
58.         });
59.
60.         btn_view_acts.setOnClickListener(new View.OnClickListener() {
61.             @Override
62.             public void onClick(View view) {
63.                 try{
64.                     ShowAllActivities();
65.
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
66.         }catch(Exception e){
67.             Toast.makeText(getApplicationContext(), "Error fetching activity!",
        Toast.LENGTH_LONG).show();
68.         }
69.     }
70. });
71.
72. }
73.
74.
75.     private void ShowAllActivities() {
76.         //custom adapter
77.         customAdapter = new LayoutAdapter(getApplicationContext(),R.layout.activity_st
yle_item,db_helper.getAllActivities());
78.         activityList.setAdapter(customAdapter);
79.
80.         //activities = new ArrayAdapter<UserActivityModel>(MainActivity.this,android.R.
layout.simple_list_item_1 ,db_helper.getAllActivities());
81.         //activityList.setAdapter(activities);
82.     }
83.     private void clearAllFields(){
84.         et_act_name.setText("");
85.         et_act_info.setText("");
86.         et_act_date.setText("");
87.     }
88. }
```

### DatabaseHelper.java(Class)

```
1. package com.example.sqlitedb_basix;
2.
3. import android.content.ContentValues;
4. import android.content.Context;
5. import android.database.Cursor;
6. import android.database.sqlite.SQLiteDatabase;
7. import android.database.sqlite.SQLiteOpenHelper;
8.
9. import androidx.annotation.Nullable;
10.
11. import java.util.ArrayList;
12. import java.util.List;
13.
14. public class DatabaseHelper extends SQLiteOpenHelper {
15.     public static final String ACTIVITY_TABLE = "ACTIVITY_TABLE";
16.     public static final String COL_ACTIVITY_NAME = "Activity_Name";
17.     public static final String COL_ACTIVITY_INFO = "Activity_Info";
18.     public static final String COL_ACTIVITY_DEADLINE = "Activity_Deadline";
19.     public static final String COL_ID = "ID";
20.
21.     public DatabaseHelper(@Nullable Context context) {
22.         super(context, "ActivityTracker.db", null, 1);
23.     }
24.
25.     //this method is automatically called when you try to access the
26.     // DB for the very first time.
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
27.    // this includes codes that generates the DB and table
28.    @Override
29.    public void onCreate(SQLiteDatabase db) {
30.        String createTableStm = "CREATE TABLE " + ACTIVITY_TABLE+ "(" + COL_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + COL_ACTIVITY_NAME + " TEXT NOT NULL, " + COL_ACTIVITY_INFO + " TEXT NOT NULL, " + COL_ACTIVITY_DEADLINE + " TEXT NOT NULL)";
31.        db.execSQL(createTableStm);
32.    }
33.
34.    //this method is invoked when your DB's version is updated.
35.    //Suppose your app is used by hundred of users and you wanted to change the DB's schema without crashing the existing app.
36.    // like adding new columns or adding multiple tables
37.    //this method comes to rescue and upgrades the schema of older app DB to the latest one.
38.    @Override
39.    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {
40.
41.    }
42.
43.    public boolean addActivity(UserActivityModel user_act){
44.        SQLiteDatabase db = this.getWritableDatabase(); //for insert actions
45.        ContentValues cv = new ContentValues(); //stores data in value pairs, similar to 'put Extra' in Intent
46.        cv.put(COL_ACTIVITY_NAME,user_act.getActivity_name());
47.        cv.put(COL_ACTIVITY_INFO,user_act.getActivity_info());
48.        cv.put(COL_ACTIVITY_DEADLINE,user_act.getActivity_deadline());
49.
50.        final long insertStatus = db.insert(ACTIVITY_TABLE, null, cv);//returns insert status as long (0> true, 0< false)
51.
52.        return insertStatus != -1; //implicit conversion long to bool
53.    }
54.    public void deleteActivity(UserActivityModel user_act){
55.        SQLiteDatabase db = this.getWritableDatabase();
56.        String qry = "DELETE FROM "+ACTIVITY_TABLE+" WHERE "+ COL_ID +" = " + user_act.getActivity_id();
57.        Cursor cursor = db.rawQuery(qry, null);
58.
59.        cursor.moveToFirst();
60.        db.close();
61.    }
62.    public List<UserActivityModel> getAllActivities(){
63.        List<UserActivityModel> resultantList = new ArrayList<>();
64.
65.        //retrieving data from the db
66.        String qry = "SELECT * FROM "+ACTIVITY_TABLE;
67.        SQLiteDatabase db = this.getReadableDatabase(); //for reading data from the db
68.
69.        //we can execute the qry 2 ways:
70.        // 1. execSql
71.        // 2. rawQuery returns cursor(set of results)
72.        Cursor resultSet = db.rawQuery(qry,null);
73.
74.        //checking if the result set is empty
75.        if(resultSet.moveToFirst()){
76.            //resultSet.moveToFirst(): returns a bool value if the dataset contains atleast 1 value
77.
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
78.          //loop through cursor list and store them in actual data list
79.          do{                                     // column index referenced with table
    schema
80.              String act_name = resultSet.getString(1);
81.              String act_info = resultSet.getString(2);
82.              String act_deadline = resultSet.getString(3);
83.              int act_ID = resultSet.getInt(0);
84.              UserActivityModel activityObj = new UserActivityModel(act_ID,act_name,a
ct_info,act_deadline);
85.              resultantList.add(activityObj);
86.          }while(resultSet.moveToNext());
87.      }else{
88.          //failure, nothing to return as a result
89.      }
90.      resultSet.close();
91.      db.close();
92.      return resultantList;
93.  }
94. }
```

UserActivityModel.java(Class)

```
1.  package com.example.sqlitedb_basix;
2.
3.
4.  public class UserActivityModel {
5.
6.      private String activity_name, activity_info,activity_deadline;
7.      private int activity_id;
8.      public UserActivityModel(int activity_id,String activity_name, String activity_info
, String activity_deadline) {
9.
10.         this.activity_id = activity_id;
11.         this.activity_name = activity_name;
12.         this.activity_info = activity_info;
13.         this.activity_deadline = activity_deadline;
14.     }
15.
16.     //toString to print data of the class object
17.     @Override
18.     public String toString() {
19.         return "UserActivityModel{" +
20.             "activity_name='" + activity_name + '\'' +
21.             ", activity_info='" + activity_info + '\'' +
22.             ", activity_deadline='" + activity_deadline + '\'' +
23.             ", activity_id=" + activity_id +
24.             '}';
25.     }
26.
27.     public int getActivity_id() {
28.         return activity_id;
29.     }
30.
31.     public void setActivity_id(int activity_id) {
32.         this.activity_id = activity_id;
33.     }
34. }
```



## Experiment 11

Develop an android application to store data in the SQLite database.

```
35.     public String getActivity_deadline() {
36.         return activity_deadline;
37.     }
38.
39.     public void setActivity_deadline(String activity_deadline) {
40.         this.activity_deadline = activity_deadline;
41.     }
42.
43.     public UserActivityModel() {
44.     }
45.
46.     public String getActivity_name() {
47.         return activity_name;
48.     }
49.
50.     public void setActivity_name(String activity_name) {
51.         this.activity_name = activity_name;
52.     }
53.
54.     public String getActivity_info() {
55.         return activity_info;
56.     }
57.
58.     public void setActivity_info(String activity_info) {
59.         this.activity_info = activity_info;
60.     }
61.
62.
63. }
```

### LayoutAdapter.java

```
1.  package com.example.sqlitedb_basix;
2.
3.  import android.content.Context;
4.  import android.view.LayoutInflater;
5.  import android.view.View;
6.  import android.view.ViewGroup;
7.  import android.widget.ArrayAdapter;
8.  import android.widget.Button;
9.  import android.widget.TextView;
10. import android.widget.Toast;
11.
12. import androidx.annotation.NonNull;
13. import androidx.annotation.Nullable;
14.
15.
16. import java.util.List;
17. import java.util.Objects;
18.
19. public class LayoutAdapter extends ArrayAdapter<UserActivityModel> {
20.     private Context appContext;
21.     private int itemLayout;    // the xml layout we created to style each list item
22.     private List<UserActivityModel> allActivities;
23.     private DatabaseHelper db_helper;
24.
25.     public LayoutAdapter(@NonNull Context context, int layout, @NonNull List<UserActivi
tyModel> objects) {
26.         super(context, layout, objects);
```

## Experiment 11

Develop an android application to store data in the SQLite database.

```
27.         this.appContext = context;
28.         this.itemLayout = layout;
29.         this.allActivities = objects;
30.     }
31.
32.     /*
33.      * getView() is responsible for rendering out each row
34.      * Here you define what information shows and where it sits within the ListView.
35.      */
36.     @NonNull
37.     @Override
38.     public View getView(final int position, @Nullable View rowView, @NonNull ViewGroup
generatedLayout) {
39.
40.         //initializing the inflater obj as per the context passed as an argument
41.         LayoutInflater inflater = LayoutInflater.from(appContext);
42.         //db initialization
43.         db_helper = new DatabaseHelper(appContext);
44.
45.         //layout inflater gonna inflate each rowView as per the layout resource(in this
case itemLayout) we pass a parameter.
46.         rowView = inflater.inflate(itemLayout,generatedLayout,false);
47.         //generatedLayout:ViewGroup, is basically a view replica of 'itemLayout' file w
e have created to style each item in our list.
48.         //generatedLayout decides how our rowView gonna look like.
49.         //dynamically generated for each item/row.
50.
51.
52.         TextView act_title = rowView.findViewById(R.id.tv_act_title);
53.         TextView act_date = rowView.findViewById(R.id.tv_act_time);
54.         TextView act_info = rowView.findViewById(R.id.tv_act_info);
55.         Button btn_dlt = rowView.findViewById(R.id.btn_delete);
56.
57.         act_title.setText(Objects.requireNonNull(getItem(position)).getActivity_name())
;
58.         act_date.setText(Objects.requireNonNull(getItem(position)).getActivity_deadline
());
59.         act_info.setText(Objects.requireNonNull(getItem(position)).getActivity_info());
60.
61.         //deleting specific activity from the DB
62.         btn_dlt.setOnClickListener(new View.OnClickListener() {
63.             @Override
64.             public void onClick(View v) {
65.                 UserActivityModel clickedActivity = (UserActivityModel) allActivities.g
et(position); //got the activity obj
66.                 db_helper.deleteActivity(clickedActivity);
67.                 allActivities.remove(position);
68.                 notifyDataSetChanged();
69.                 Toast.makeText(appContext, "Activity Deleted!", Toast.LENGTH_SHORT).show(
);
70.             }
71.         });
72.
73.         return rowView;
74.     }
75.
76.
77. }
```

## Experiment 11

Develop an android application to store data in the SQLite database.

Snapshots:

