

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

MainActivity.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.a
   ndroid.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="#F2F2F2"
8.     android:padding="10dp"
9.     tools:context=".MainActivity">
10.
11.         <TextView
12.             android:id="@+id/appName"
13.             android:layout_width="wrap_content"
14.             android:layout_height="wrap_content"
15.             android:layout_marginTop="16dp"
16.             android:fontFamily="@font/faster_one"
17.             android:text="@string/appName"
18.             android:textSize="24sp"
19.             app:layout_constraintEnd_toEndOf="parent"
20.             app:layout_constraintHorizontal_bias="0.5"
21.             app:layout_constraintStart_toStartOf="parent"
22.             app:layout_constraintTop_toTopOf="parent" />
23.
24.         <EditText
25.             android:id="@+id/et_empID"
26.             android:layout_width="match_parent"
27.             android:layout_height="wrap_content"
28.             android:layout_marginTop="24dp"
29.             android:ems="10"
30.             android:fontFamily="@font/roboto"
31.             android:hint="@string/idHint"
32.             android:inputType="textPersonName"
33.             android:textSize="16sp"
34.             app:layout_constraintEnd_toEndOf="parent"
35.             app:layout_constraintHorizontal_bias="0.5"
36.             app:layout_constraintStart_toStartOf="parent"
37.             app:layout_constraintTop_toBottomOf="@+id/appName" />
38.
39.         <EditText
40.             android:id="@+id/et_empName"
41.             android:layout_width="match_parent"
42.             android:layout_height="wrap_content"
43.             android:layout_marginTop="8dp"
44.             android:ems="10"
45.             android:fontFamily="@font/roboto"
46.             android:hint="@string/nameHint"
47.             android:inputType="textPersonName"
48.             android:textSize="16sp"
49.             app:layout_constraintEnd_toEndOf="parent"
50.             app:layout_constraintHorizontal_bias="0.5"
51.             app:layout_constraintStart_toStartOf="parent"
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
52.         app:layout_constraintTop_toBottomOf="@+id/et_empID" />
53.
54.     <EditText
55.         android:id="@+id/et_empDpt"
56.         android:layout_width="match_parent"
57.         android:layout_height="wrap_content"
58.         android:layout_marginTop="8dp"
59.         android:ems="10"
60.         android:fontFamily="@font/roboto"
61.         android:hint="@string/dptHint"
62.         android:inputType="textPersonName"
63.         android:textSize="16sp"
64.         app:layout_constraintEnd_toEndOf="parent"
65.         app:layout_constraintHorizontal_bias="0.5"
66.         app:layout_constraintStart_toStartOf="parent"
67.         app:layout_constraintTop_toBottomOf="@+id/et_empName" />
68.
69.     <EditText
70.         android:id="@+id/et_empYoj"
71.         android:layout_width="match_parent"
72.         android:layout_height="wrap_content"
73.         android:layout_marginTop="8dp"
74.         android:ems="10"
75.         android:fontFamily="@font/roboto"
76.         android:hint="@string/yojHint"
77.         android:inputType="textPersonName"
78.         android:textSize="16sp"
79.         app:layout_constraintEnd_toEndOf="parent"
80.         app:layout_constraintHorizontal_bias="0.5"
81.         app:layout_constraintStart_toStartOf="parent"
82.         app:layout_constraintTop_toBottomOf="@+id/et_empDpt" />
83.
84.     <EditText
85.         android:id="@+id/et_empContact"
86.         android:layout_width="match_parent"
87.         android:layout_height="wrap_content"
88.         android:layout_marginTop="8dp"
89.         android:ems="10"
90.         android:fontFamily="@font/roboto"
91.         android:hint="@string/contactHint"
92.         android:inputType="textPersonName"
93.         android:textSize="16sp"
94.         app:layout_constraintEnd_toEndOf="parent"
95.         app:layout_constraintHorizontal_bias="0.5"
96.         app:layout_constraintStart_toStartOf="parent"
97.         app:layout_constraintTop_toBottomOf="@+id/et_empYoj" />
98.
99.     <Button
100.         android:id="@+id/btn_register"
101.         android:layout_width="0dp"
102.         android:layout_height="wrap_content"
103.         android:layout_marginStart="8dp"
104.         android:layout_marginTop="8dp"
105.         android:layout_marginEnd="8dp"
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```

106.         android:fontFamily="@font/dosis"
107.         android:text="@string/btnTxt"
108.         android:textSize="16sp"
109.         android:textStyle="bold"
110.         app:layout_constraintEnd_toStartOf="@+id/btn_viewAll"
111.         app:layout_constraintHorizontal_bias="0.5"
112.         app:layout_constraintStart_toStartOf="parent"
113.         app:layout_constraintTop_toBottomOf="@+id/et_empContact" />
114.
115.     <Button
116.         android:id="@+id/btn_viewAll"
117.         android:layout_width="0dp"
118.         android:layout_height="wrap_content"
119.         android:layout_marginEnd="8dp"
120.         android:text="@string/btnViewAll"
121.         app:layout_constraintBaseline_toBaselineOf="@+id/btn_register"
122.
123.         app:layout_constraintEnd_toEndOf="parent"
124.         app:layout_constraintHorizontal_bias="0.5"
125.         app:layout_constraintStart_toEndOf="@+id/btn_register" />
126.
127.     <ListView
128.         android:id="@+id/lv_empList"
129.         android:layout_width="0dp"
130.         android:layout_height="0dp"
131.         android:layout_marginTop="8dp"
132.         app:layout_constraintBottom_toBottomOf="parent"
133.         app:layout_constraintEnd_toEndOf="parent"
134.         app:layout_constraintHorizontal_bias="0.5"
135.         app:layout_constraintStart_toStartOf="parent"
136.         app:layout_constraintTop_toBottomOf="@+id/btn_register" />
137. </androidx.constraintlayout.widget.ConstraintLayout>

```

activity\_all\_employees\_list.xml

```

1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.a
   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.     tools:context=".AllEmployeesList">
8.
9.     <TextView
10.         android:id="@+id/tv_orgName"
11.         android:layout_width="wrap_content"
12.         android:layout_height="wrap_content"
13.         android:layout_marginTop="24dp"
14.         android:fontFamily="@font/faster_one"
15.         android:text="@string/orgName"
16.         android:textSize="30sp"
17.         app:layout_constraintEnd_toEndOf="parent"

```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
18.         app:layout_constraintHorizontal_bias="0.5"
19.         app:layout_constraintStart_toStartOf="parent"
20.         app:layout_constraintTop_toTopOf="parent" />
21.
22.     <ListView
23.         android:id="@+id/lv_allEmps"
24.         android:layout_width="0dp"
25.         android:layout_height="wrap_content"
26.         android:layout_marginTop="8dp"
27.         app:layout_constraintEnd_toEndOf="parent"
28.         app:layout_constraintHorizontal_bias="0.5"
29.         app:layout_constraintStart_toStartOf="parent"
30.         app:layout_constraintTop_toBottomOf="@+id/tv_orgName" />
31. </androidx.constraintlayout.widget.ConstraintLayout>
```

Readable\_item\_style.xml

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.a
   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.
8.      <TextView
9.          android:id="@+id/tv_ID"
10.         android:layout_width="0dp"
11.         android:layout_height="wrap_content"
12.         android:fontFamily="@font/faster_one"
13.         android:paddingStart="5dp"
14.         android:text="Text View"
15.         android:textSize="18sp"
16.         app:layout_constraintEnd_toEndOf="parent"
17.         app:layout_constraintHorizontal_bias="0.5"
18.         app:layout_constraintStart_toStartOf="parent"
19.         app:layout_constraintTop_toTopOf="parent" />
20.
21.      <TextView
22.          android:id="@+id/tv_name"
23.          android:layout_width="0dp"
24.          android:layout_height="wrap_content"
25.          android:layout_marginTop="8dp"
26.          android:fontFamily="@font/dosis"
27.          android:paddingStart="5dp"
28.          android:text="TextView"
29.          android:textSize="16sp"
30.          android:textStyle="bold"
31.          app:layout_constraintEnd_toEndOf="parent"
32.          app:layout_constraintHorizontal_bias="0.501"
33.          app:layout_constraintStart_toStartOf="parent"
34.          app:layout_constraintTop_toBottomOf="@+id/tv_ID" />
35.
36.      <TextView
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
37.         android:id="@+id/tv_dpt"
38.         android:layout_width="0dp"
39.         android:layout_height="wrap_content"
40.         android:layout_marginTop="8dp"
41.         android:fontFamily="@font/dosis"
42.         android:paddingStart="5dp"
43.         android:text="TextView"
44.         android:textSize="16sp"
45.         android:textStyle="bold"
46.         app:layout_constraintEnd_toEndOf="parent"
47.         app:layout_constraintHorizontal_bias="0.501"
48.         app:layout_constraintStart_toStartOf="parent"
49.         app:layout_constraintTop_toBottomOf="@+id/tv_name" />
50. </androidx.constraintlayout.widget.ConstraintLayout>
```

style\_list\_item.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.a
   ndroid.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.
8.
9.     <TextView
10.         android:id="@+id/tv_empID"
11.         android:layout_width="match_parent"
12.         android:layout_height="23dp"
13.         android:layout_marginStart="8dp"
14.         android:layout_marginTop="16dp"
15.         android:layout_marginEnd="8dp"
16.         android:fontFamily="@font/faster_one"
17.         android:paddingStart="5dp"
18.         android:text="TextView"
19.         android:textSize="18sp"
20.         app:layout_constraintEnd_toEndOf="parent"
21.         app:layout_constraintHorizontal_bias="0.5"
22.         app:layout_constraintStart_toStartOf="parent"
23.         app:layout_constraintTop_toTopOf="parent" />
24.
25.
26.     <TextView
27.         android:id="@+id/tv_empName"
28.         android:layout_width="match_parent"
29.         android:layout_height="wrap_content"
30.         android:layout_marginStart="8dp"
31.         android:layout_marginTop="8dp"
32.         android:layout_marginEnd="8dp"
33.         android:fontFamily="@font/dosis"
34.         android:paddingStart="5dp"
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
35.         android:text="TextView"
36.         android:textSize="16sp"
37.         android:textStyle="bold"
38.         app:layout_constraintEnd_toEndOf="parent"
39.         app:layout_constraintHorizontal_bias="0.5"
40.         app:layout_constraintStart_toStartOf="parent"
41.         app:layout_constraintTop_toBottomOf="@+id/tv_empID" />
42.     <!-- et Name hidden, when visible replaces tv Name -->
43.     <EditText
44.         android:id="@+id/up_empName"
45.         android:layout_width="match_parent"
46.         android:layout_height="wrap_content"
47.         android:layout_marginStart="8dp"
48.         android:layout_marginTop="8dp"
49.         android:layout_marginEnd="8dp"
50.         android:fontFamily="@font/dosis"
51.         android:paddingStart="5dp"
52.         android:textSize="16sp"
53.         android:textStyle="bold"
54.         app:layout_constraintEnd_toEndOf="parent"
55.         app:layout_constraintHorizontal_bias="0.5"
56.         app:layout_constraintStart_toStartOf="parent"
57.         app:layout_constraintTop_toBottomOf="@+id/tv_empID"
58.         android:visibility="gone"/>
59.
60.     <TextView
61.         android:id="@+id/tv_empDpt"
62.         android:layout_width="match_parent"
63.         android:layout_height="wrap_content"
64.         android:layout_marginStart="8dp"
65.         android:layout_marginTop="8dp"
66.         android:layout_marginEnd="8dp"
67.         android:fontFamily="@font/dosis"
68.         android:paddingStart="5dp"
69.         android:text="TextView"
70.         android:textSize="16sp"
71.         android:textStyle="bold"
72.         app:layout_constraintEnd_toEndOf="parent"
73.         app:layout_constraintHorizontal_bias="0.5"
74.         app:layout_constraintStart_toStartOf="parent"
75.         app:layout_constraintTop_toBottomOf="@+id/tv_empName" />
76.     <!-- et Dpt hidden, when visible replaces tv Dpt -->
77.     <EditText
78.         android:id="@+id/up_empDpt"
79.         android:layout_width="match_parent"
80.         android:layout_height="wrap_content"
81.         android:layout_marginStart="8dp"
82.         android:layout_marginTop="8dp"
83.         android:layout_marginEnd="8dp"
84.         android:fontFamily="@font/dosis"
85.         android:paddingStart="5dp"
86.         android:textSize="16sp"
87.         android:textStyle="bold"
88.         app:layout_constraintEnd_toEndOf="parent"
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
89.         app:layout_constraintHorizontal_bias="0.5"
90.         app:layout_constraintStart_toStartOf="parent"
91.         app:layout_constraintTop_toBottomOf="@+id/up_empName"
92.         android:visibility="gone"/>
93.
94.
95.     <TextView
96.         android:id="@+id/tv_empYoj"
97.         android:layout_width="match_parent"
98.         android:layout_height="wrap_content"
99.         android:layout_marginStart="8dp"
100.        android:layout_marginTop="8dp"
101.        android:layout_marginEnd="8dp"
102.        android:fontFamily="@font/dosis"
103.        android:paddingStart="5dp"
104.        android:text="TextView"
105.        android:textSize="16sp"
106.        android:textStyle="bold"
107.        app:layout_constraintEnd_toEndOf="parent"
108.        app:layout_constraintHorizontal_bias="0.5"
109.        app:layout_constraintStart_toStartOf="parent"
110.        app:layout_constraintTop_toBottomOf="@+id/tv_empDpt" />
111.    <!-- et yoj hidden, when visible replaces tv yoj -->
112.    <EditText
113.        android:id="@+id/up_empYoj"
114.        android:layout_width="match_parent"
115.        android:layout_height="wrap_content"
116.        android:layout_marginStart="8dp"
117.        android:layout_marginTop="8dp"
118.        android:layout_marginEnd="8dp"
119.        android:fontFamily="@font/dosis"
120.        android:paddingStart="5dp"
121.        android:textSize="16sp"
122.        android:textStyle="bold"
123.        app:layout_constraintEnd_toEndOf="parent"
124.        app:layout_constraintHorizontal_bias="0.5"
125.        app:layout_constraintStart_toStartOf="parent"
126.        app:layout_constraintTop_toBottomOf="@+id/up_empDpt"
127.        android:visibility="gone"/>
128.
129.    <TextView
130.        android:id="@+id/tv_empContact"
131.        android:layout_width="match_parent"
132.        android:layout_height="wrap_content"
133.        android:layout_marginStart="8dp"
134.        android:layout_marginTop="8dp"
135.        android:layout_marginEnd="8dp"
136.        android:fontFamily="@font/dosis"
137.        android:paddingStart="5dp"
138.        android:text="TextView"
139.        android:textSize="16sp"
140.        android:textStyle="bold"
141.        app:layout_constraintEnd_toEndOf="parent"
142.        app:layout_constraintHorizontal_bias="0.5"
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
143.         app:layout_constraintStart_toStartOf="parent"
144.         app:layout_constraintTop_toBottomOf="@+id/tv_empY0J" />
145.     <!-- et yoj hidden, when visible replaces tv yoj -->
146.     <EditText
147.         android:id="@+id/up_empContact"
148.         android:layout_width="match_parent"
149.         android:layout_height="wrap_content"
150.         android:layout_marginStart="8dp"
151.         android:layout_marginTop="8dp"
152.         android:layout_marginEnd="8dp"
153.         android:fontFamily="@font/dosis"
154.         android:paddingStart="5dp"
155.         android:textSize="16sp"
156.         android:textStyle="bold"
157.         app:layout_constraintEnd_toEndOf="parent"
158.         app:layout_constraintHorizontal_bias="0.5"
159.         app:layout_constraintStart_toStartOf="parent"
160.         app:layout_constraintTop_toBottomOf="@+id/up_empY0J"
161.         android:visibility="gone"/>
162.
163.
164.     <Button
165.         android:id="@+id/btn_dlt"
166.         android:layout_width="0dp"
167.         android:layout_height="wrap_content"
168.         android:layout_marginStart="8dp"
169.         android:layout_marginTop="8dp"
170.         android:layout_marginEnd="8dp"
171.         android:fontFamily="@font/dosis"
172.         android:text="@string/dltBtn_txt"
173.         android:textSize="16sp"
174.         android:textStyle="bold"
175.         app:layout_constraintEnd_toStartOf="@+id/btn_edit"
176.         app:layout_constraintHorizontal_bias="0.5"
177.         app:layout_constraintHorizontal_chainStyle="packed"
178.         app:layout_constraintStart_toStartOf="parent"
179.         app:layout_constraintTop_toBottomOf="@+id/tv_empContact" />
180.
181.     <!-- Cancel btn hidden when visible replaces Delete btn -->
182.     <Button
183.         android:id="@+id/btn_cancel"
184.         android:layout_width="0dp"
185.         android:layout_height="wrap_content"
186.         android:layout_marginStart="8dp"
187.         android:layout_marginTop="8dp"
188.         android:layout_marginEnd="8dp"
189.         android:fontFamily="@font/dosis"
190.         android:text="@string/cancelBtn"
191.         android:textSize="16sp"
192.         android:textStyle="bold"
193.         app:layout_constraintEnd_toStartOf="@+id/btn_update"
194.         app:layout_constraintHorizontal_bias="0.5"
195.         app:layout_constraintHorizontal_chainStyle="packed"
196.         app:layout_constraintStart_toStartOf="parent"
```



Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
197.         app:layout_constraintTop_toBottomOf="@+id/up_empContact"
198.         android:visibility="gone"/>
199.
200.         <Button
201.             android:id="@+id/btn_edit"
202.             android:layout_width="0dp"
203.             android:layout_height="wrap_content"
204.             android:layout_marginStart="8dp"
205.             android:layout_marginEnd="8dp"
206.             android:fontFamily="@font/dosis"
207.             android:text="@string/btnEdit"
208.             android:textSize="16sp"
209.             android:textStyle="bold"
210.             app:layout_constraintBaseline_toBaselineOf="@+id/btn_dlt"
211.             app:layout_constraintEnd_toEndOf="parent"
212.             app:layout_constraintHorizontal_bias="0.5"
213.             app:layout_constraintStart_toEndOf="@+id/btn_dlt" />
214.
215.         <!-- Update btn hidden when visible replaces Edit btn -->
216.         <Button
217.             android:id="@+id/btn_update"
218.             android:layout_width="0dp"
219.             android:layout_height="wrap_content"
220.             android:layout_marginStart="8dp"
221.             android:layout_marginEnd="8dp"
222.             android:fontFamily="@font/dosis"
223.             android:text="@string/updateBtn"
224.             android:textSize="16sp"
225.             android:textStyle="bold"
226.             app:layout_constraintBaseline_toBaselineOf="@+id/btn_cancel"
227.             app:layout_constraintEnd_toEndOf="parent"
228.             app:layout_constraintHorizontal_bias="0.5"
229.             app:layout_constraintStart_toEndOf="@+id/btn_cancel"
230.             android:visibility="gone"/>
231.
232.     </androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java

```
1. package com.example.employeeedb_app;
2.
3. import androidx.appcompat.app.AppCompatActivity;
4.
5. import android.content.Intent;
6. import android.os.Bundle;
7. import android.view.View;
8. import android.view.WindowManager;
9. import android.widget.Button;
10. import android.widget.EditText;
11. import android.widget.ListView;
12. import android.widget.Toast;
13.
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
14. public class MainActivity extends AppCompatActivity {
15.
16.     EditText emp_ID, emp_name, emp_dpt, emp_yoj, emp_contact;
17.     Button btn_register, btn_viewAll;
18.     ListView emp_list;
19.     LayoutAdapter2 customAdapter;
20.     DBHelper db_help;
21.
22.     @Override
23.     protected void onCreate(Bundle savedInstanceState) {
24.         super.onCreate(savedInstanceState);
25.         setContentView(R.layout.activity_main);
26.
27.         //initializing db with context to our main activity
28.         db_help = new DBHelper(MainActivity.this);
29.
30.         emp_ID = findViewById(R.id.et_empID);
31.         emp_name = findViewById(R.id.et_empName);
32.         emp_dpt = findViewById(R.id.et_empDpt);
33.         emp_yoj = findViewById(R.id.et_empYOJ);
34.         emp_contact = findViewById(R.id.et_empContact);
35.         emp_list = findViewById(R.id.lv_empList);
36.         btn_register = findViewById(R.id.btn_register);
37.         btn_viewAll = findViewById(R.id.btn_viewAll);
38.
39.
40.         ShowAllEmployees();
41.
42.         //view all emps
43.         btn_viewAll.setOnClickListener(new View.OnClickListener() {
44.             @Override
45.             public void onClick(View v) {
46.                 Intent allEmps = new Intent(getApplicationContext(),AllEmployeesLi
st.class);
47.                 startActivity(allEmps);
48.             }
49.         });
50.
51.         //registering emp on clicking btn
52.         btn_register.setOnClickListener(new View.OnClickListener() {
53.             @Override
54.             public void onClick(View v) {
55.                 if(emp_ID.length() != 0 && emp_name.length() != 0 && emp_dpt.lengt
h() != 0 && emp_yoj.length() != 0 && emp_contact.length() != 0){
56.
57.                     try {
58.                         //retrieve all info
59.                         EmployeeModel emp = new EmployeeModel(-
1,emp_ID.getText().toString().trim(),emp_name.getText().toString().trim(),emp_dpt.
getText().toString().trim(),emp_yoj.getText().toString().trim(),emp_contact.getTex
t().toString().trim());
60.                         boolean registerStatus = db_help.addEmployeeDetails(emp);
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
61.             Toast.makeText(MainActivity.this,"Employee Registered Succ
essfully? : "+registerStatus,Toast.LENGTH_LONG).show();
62.             ShowAllEmployees();
63.             clearAllFields();
64.         } catch (Exception e) {
65.             Toast.makeText(MainActivity.this, "Error Registering Emplo
yee! error: "+e.getMessage(), Toast.LENGTH_LONG).show();
66.         }
67.
68.         }else
69.             Toast.makeText(MainActivity.this,"All Fields are required!",To
ast.LENGTH_LONG).show();
70.     }
71. });
72. }
73.
74. private void clearAllFields() {
75.     emp_ID.setText("");
76.     emp_name.setText("");
77.     emp_dpt.setText("");
78.     emp_yoj.setText("");
79.     emp_contact.setText("");
80. }
81.
82. private void ShowAllEmployees() {
83.     //custom adapter
84.     customAdapter = new LayoutAdapter2(getApplicationContext(),R.layout.readab
le_item_style,db_help.getAllEmployees());
85.     emp_list.setAdapter(customAdapter);
86. }
87.
88.
89. }
```

AllEmployeesList.java

```
1. package com.example.employeedb_app;
2.
3. import androidx.appcompat.app.AppCompatActivity;
4.
5. import android.content.Intent;
6. import android.os.Bundle;
7. import android.view.View;
8. import android.widget.Button;
9. import android.widget.ListView;
10.
11. public class AllEmployeesList extends AppCompatActivity {
12.
13.     ListView emps_list;
14.     LayoutAdapter customAdapter;
15.     DbHelper db_help;
16.     Button btnBack;
17. }
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
18.     @Override
19.     protected void onCreate(Bundle savedInstanceState) {
20.         super.onCreate(savedInstanceState);
21.         setContentView(R.layout.activity_all_employees_list);
22.
23.         emps_list = findViewById(R.id.lv_allEmps);
24.         btnBack = findViewById(R.id.btnBack);
25.
26.         //initializing db with context to our main activity
27.         db_help = new DbHelper(AllEmployeesList.this);
28.         customAdapter = new LayoutAdapter(AllEmployeesList.this,R.layout.style_list_item,db_help.getAllEmployees());
29.         emps_list.setAdapter(customAdapter);
30.
31.         btnBack.setOnClickListener(new View.OnClickListener() {
32.             @Override
33.             public void onClick(View v) {
34.                 Intent home = new Intent(getApplicationContext(),MainActivity.class);
35.                 startActivity(home);
36.             }
37.         });
38.
39.     }
40. }
```

DbHepler.java(Class file)

```
1. package com.example.employeeedb_app;
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 DbHelper extends SQLiteOpenHelper {
15.     public static final String EMP_TABLE = "EMP_TABLE";
16.     public static final String COL_EMP_SR= "Emp_SR";
17.     public static final String COL_EMP_ID = "Emp_ID";
18.     public static final String COL_EMP_NAME = "Emp_Name";
19.     public static final String COL_EMP_DPT= "Emp_Dpt";
20.     public static final String COL_EMP_YOJ = "Emp_YOJ";
21.     public static final String COL_EMP_CONTACT = "Emp_Contact";
22.
23.
24.     public DbHelper(@Nullable Context context) {
25.         super(context, "S7Employees.db", null, 1);
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```

26.     }
27.
28.     @Override
29.     public void onCreate(SQLiteDatabase db) {
30.         String createTableStm = "CREATE TABLE " + EMP_TABLE+ "(" + COL_EMP_SR + "
INTEGER PRIMARY KEY AUTOINCREMENT, " + COL_EMP_ID + " TEXT NOT NULL, " + COL_EMP_N
AME + " TEXT NOT NULL, " + COL_EMP_DPT + " TEXT NOT NULL, " + COL_EMP_YOJ + " TEXT
NOT NULL, " + COL_EMP_CONTACT + " TEXT NOT NULL)";
31.         db.execSQL(createTableStm);
32.     }
33.
34.     @Override
35.     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
36.         //not require as of now
37.         //this method is invoked when your DB's version is updated.
38.     }
39.
40.
41.     //INSERTING EMP DATA
42.     public boolean addEmployeeDetails(EmployeeModel emp){
43.
44.         //setting up vars
45.         SQLiteDatabase db = this.getWritableDatabase(); //accessing the DB in writ
e mode
46.         ContentValues cv = new ContentValues(); //for storing information in key,v
al pair
47.
48.         //initializing table columns value
49.         cv.put(COL_EMP_ID,emp.getEmp_ID());
50.         cv.put(COL_EMP_NAME,emp.getEmp_name());
51.         cv.put(COL_EMP_DPT,emp.getEmp_dpt());
52.         cv.put(COL_EMP_YOJ, emp.getEmp_yoj());
53.         cv.put(COL_EMP_CONTACT,emp.getEmp_contact());
54.
55.         //writing cv into the respective table
56.         final long insertStatus = db.insert(EMP_TABLE,null,cv); //return +ve for
true and -ve for false
57.         db.close();
58.         return insertStatus != -1; //returns bool value after comparision.
59.
60.     }
61.
62.
63.     //DELETING EMP FROM DB
64.     public void deleteEmployeeData(EmployeeModel emp){
65.         SQLiteDatabase db = this.getWritableDatabase();
66.         String qry = "DELETE FROM " + EMP_TABLE + " WHERE " + COL_EMP_SR + " = " +
emp.getEmp_SR();
67.         Cursor cursor = db.rawQuery(qry, null);
68.
69.         cursor.moveToFirst();
70.         db.close();
71.     }
72.

```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
73.    //UPDATING EMP
74.    public void updateEmployeeData(EmployeeModel emp) {
75.        SQLiteDatabase db = this.getWritableDatabase();
76.        ContentValues cv = new ContentValues(); //for storing information in key,v
    al pair
77.
78.        //initializing table columns value
79.        cv.put(COL_EMP_NAME,emp.getEmp_name());
80.        cv.put(COL_EMP_DPT,emp.getEmp_dpt());
81.        cv.put(COL_EMP_YOJ, emp.getEmp_yoj());
82.        cv.put(COL_EMP_CONTACT,emp.getEmp_contact());
83.
84.        db.update(EMP_TABLE,cv,COL_EMP_ID+" = '"+emp.getEmp_ID()+"",null);
85.        db.close();
86.    }
87.    //GETTING ALL EMP LIST
88.    public List<EmployeeModel> getAllEmployees(){
89.        //inserting all emp records
90.        List<EmployeeModel> emp_list = new ArrayList<>();
91.
92.        //retreiving data from the db
93.        String qry = "SELECT * FROM "+EMP_TABLE;
94.        SQLiteDatabase db = this.getReadableDatabase(); //for reading data form th
    e db
95.
96.        //we can execute the qry 2 ways:
97.        // 1. exeSql
98.        // 2. rawQuery returns cursor(set of results)
99.        Cursor resultSet = db.rawQuery(qry,null);
100.
101.        //checking if the result set is empty
102.        if(resultSet.moveToFirst()){
103.            //resultSet.moveToFirst(): returns a bool value if the dataset
    contains atleast 1 value
104.
105.            //loop through cursor list and store them in actual data list
106.            do{
107.                // column index referenced with table schema
108.                int emp_sr = resultSet.getInt(0);
109.                String emp_id = resultSet.getString(1);
110.                String emp_name = resultSet.getString(2);
111.                String emp_dpt = resultSet.getString(3);
112.                String emp_yoj = resultSet.getString(4);
113.                String emp_contact = resultSet.getString(5);
114.
115.                EmployeeModel emp = new EmployeeModel(emp_sr, emp_id, emp_n
    ame,emp_dpt,emp_yoj,emp_contact);
116.                emp_list.add(emp);
117.            }while(resultSet.moveToNext());
118.
119.        }else{
120.            //failure, nothing to return as a result
121.        }
122.        resultSet.close();
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
123.         db.close();
124.         return emp_list;
125.     }
126.
127.
128. }
```

EmployeeModel.java (Class file)

```
1. package com.example.employeeedb_app;
2.
3. public class EmployeeModel {
4.
5.     private String emp_name;
6.     private String emp_dpt;
7.     private String emp_contact;
8.     private String emp_yoj;
9.     private String emp_ID;
10.    private int emp_SR;
11.
12.
13.    public EmployeeModel(int emp_sr, String emp_ID, String emp_name, String emp_dp
14.        t, String emp_yoj, String emp_contact) {
15.        this.emp_name = emp_name;
16.        this.emp_dpt = emp_dpt;
17.        this.emp_contact = emp_contact;
18.        this.emp_yoj = emp_yoj;
19.        this.emp_ID = emp_ID;
20.        this.emp_SR = emp_sr;
21.    }
22.
23.    //Auto Inc row no. for the respective emp obj.
24.    public int getEmp_SR() {
25.        return emp_SR;
26.    }
27.    //ID
28.    public String getEmp_ID() {
29.        return emp_ID;
30.    }
31.    public void setEmp_ID(String emp_ID) {
32.        this.emp_ID = emp_ID;
33.    }
34.    //Name
35.    public String getEmp_name() {
36.        return emp_name;
37.    }
38.    public void setEmp_name(String emp_name) {
39.        this.emp_name = emp_name;
40.    }
41.    //Department
42.    public String getEmp_dpt() {
43.        return emp_dpt;
44.    }
45.    public void setEmp_dpt(String emp_dpt) {
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
45.         this.emp_dpt = emp_dpt;
46.     }
47.     //Contact
48.     public String getEmp_contact() {
49.         return emp_contact;
50.     }
51.     public void setEmp_contact(String emp_contact) {
52.         this.emp_contact = emp_contact;
53.     }
54.     //YOJ
55.     public String getEmp_yoj() {
56.         return emp_yoj;
57.     }
58.     public void setEmp_yoj(String emp_yoj) {
59.         this.emp_yoj = emp_yoj;
60.     }
61.
62.
63.     //toString() to print data of the class object
64.     @Override
65.     public String toString() {
66.         return "EmployeeModel{" +
67.             "emp_name='" + emp_name + '\'' +
68.             ", emp_dpt='" + emp_dpt + '\'' +
69.             ", emp_contact='" + emp_contact + '\'' +
70.             ", emp_yoj='" + emp_yoj + '\'' +
71.             ", emp_ID=" + emp_ID +
72.             '}';
73.     }
74. }
```

LayoutAdapter.java(Class file for updating emp record)

```
1. package com.example.employeeedb_app;
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.EditText;
10. import android.widget.TextView;
11. import android.widget.Toast;
12.
13. import androidx.annotation.NonNull;
14. import androidx.annotation.Nullable;
15. import androidx.appcompat.app.AppCompatActivity;
16.
17. import java.util.List;
18. import java.util.Objects;
19.
```



Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
20. public class LayoutAdapter extends ArrayAdapter<EmployeeModel> {
21.
22.     private Context appContext;
23.     private int itemLayout; //the xml layout we created to style each list item
24.     private List<EmployeeModel> allEmps;
25.     private DbHelper db_help;
26.
27.     public LayoutAdapter(@NonNull Context context, int layout, List<EmployeeModel>
emps) {
28.         super(context, layout, emps);
29.         this.appContext = context;
30.         this.itemLayout = layout;
31.         this.allEmps = emps;
32.
33.     }
34.     /*
35.      * getView() is responsible for rendering out each row
36.      * Here you define what information shows and where it sits within the ListVie
w.
37.      * */
38.     @NonNull
39.     @Override
40.     public View getView(final int position, @Nullable View rowView, @NonNull ViewGroup
roup generatedLayout) {
41.
42.         //initializing the inflater obj as per the context passed as an argument
43.         LayoutInflater inflater = LayoutInflater.from(appContext);
44.         //db initialization
45.         db_help = new DbHelper(appContext);
46.
47.         //layout inflater gonna inflate each rowView as per the layout resource(in
this case itemLayout) we pass a parameter.
48.         rowView = inflater.inflate(itemLayout, generatedLayout, false);
49.         //generatedLayout: ViewGroup, is basically a view replica of 'itemLayout' f
ile we have created to style each item in our list.
50.         //generatedLayout decides how our rowView gonna look like.
51.         //dynamically generated for each item/row.
52.
53.
54.         //Default layout
55.         final TextView empID = rowView.findViewById(R.id.tv_empID);
56.         final TextView empName = rowView.findViewById(R.id.tv_empName);
57.         final TextView empDpt = rowView.findViewById(R.id.tv_empDpt);
58.         final TextView empYoj = rowView.findViewById(R.id.tv_empYoj);
59.         final TextView empContact = rowView.findViewById(R.id.tv_empContact);
60.
61.         final Button btn_dlt = rowView.findViewById(R.id.btn_dlt);
62.         final Button btn_edit = rowView.findViewById(R.id.btn_edit);
63.
64.         //Update operation layout
65.         final Button btn_update = rowView.findViewById(R.id.btn_update);
66.         final Button btn_cancel = rowView.findViewById(R.id.btn_cancel);
67.
68.         final EditText up_empName = rowView.findViewById(R.id.up_empName);
```

**Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.**

```
69.         final EditText up_empDpt = rowView.findViewById(R.id.up_empDpt);
70.         final EditText up_empYOJ = rowView.findViewById(R.id.up_empYOJ);
71.         final EditText up_empContact = rowView.findViewById(R.id.up_empContact);
72.
73.         empID.setText("S7.dev/emp:"+Objects.requireNonNull(getItem(position)).getE
mp_ID());
74.         empName.setText(Objects.requireNonNull(getItem(position)).getEmp_name());
75.         empDpt.setText(Objects.requireNonNull(getItem(position)).getEmp_dpt());
76.         empYOJ.setText(Objects.requireNonNull(getItem(position)).getEmp_yoj());
77.         empContact.setText(Objects.requireNonNull(getItem(position)).getEmp Contac
t());
78.
79.         //deleting specific activity from the DB
80.         btn_dlt.setOnClickListener(new View.OnClickListener() {
81.             @Override
82.             public void onClick(View v) {
83.                 EmployeeModel targetEmp = (EmployeeModel) allEmps.get(position); /
/got the activity obj
84.                 db_help.deleteEmployeeData(targetEmp);
85.                 allEmps.remove(position);
86.                 notifyDataSetChanged();
87.                 Toast.makeText(appContext, "Employee Data Deleted!", Toast.LENGTH_SH
ORT).show();
88.             }
89.         });
90.
91.         //Just preparing the layout for the update operation
92.         btn_edit.setOnClickListener(new View.OnClickListener() {
93.             @Override
94.             public void onClick(View v) {
95.
96.                 //Replacing all btns
97.                 btn_edit.setVisibility(View.INVISIBLE);
98.                 btn_update.setVisibility(View.VISIBLE);
99.                 btn_dlt.setVisibility(View.INVISIBLE);
100.                 btn_cancel.setVisibility(View.VISIBLE);
101.
102.                 //replacing all tvs with ets
103.                 //Name
104.                 empName.setVisibility(View.INVISIBLE);
105.                 up_empName.setVisibility(View.VISIBLE);
106.                 up_empName.setText(empName.getText().toString());
107.                 //Dpt
108.                 empDpt.setVisibility(View.INVISIBLE);
109.                 up_empDpt.setVisibility(View.VISIBLE);
110.                 up_empDpt.setText(empDpt.getText().toString());
111.                 //YOJ
112.                 empYOJ.setVisibility(View.INVISIBLE);
113.                 up_empYOJ.setVisibility(View.VISIBLE);
114.                 up_empYOJ.setText(empYOJ.getText().toString());
115.                 //Contact
116.                 empContact.setVisibility(View.INVISIBLE);
117.                 up_empContact.setVisibility(View.VISIBLE);
```

**Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.**

```

118.         up_empContact.setText(empContact.getText().toString());
119.
120.
121.     }
122. });
123.
124.     //performing update operation
125.     btn_update.setOnClickListener(new View.OnClickListener() {
126.         @Override
127.         public void onClick(View v) {
128.
129.             EmployeeModel targetEmp = (EmployeeModel) allEmps.get(position); //got the activity obj
130.
131.
132.             //getting updated values
133.             if(up_empName.length() != 0 && up_empDpt.length() != 0 && up_empYoj.length() != 0 && up_empContact.length() != 0){
134.                 targetEmp.setEmp_name(up_empName.getText().toString());
135.                 targetEmp.setEmp_dpt(up_empDpt.getText().toString());
136.                 targetEmp.setEmp_yoj(up_empYoj.getText().toString());
137.                 targetEmp.setEmp_contact(up_empContact.getText().toString());
138.             }
139.             try {
140.                 db_help.updateEmployeeData(targetEmp);
141.                 notifyDataSetChanged();
142.                 Toast.makeText(appContext, "Successfully Updated!", Toast.LENGTH_LONG).show();
143.             } catch (Exception e) {
144.                 Toast.makeText(appContext, "Updation Error! error: " + e.getMessage(), Toast.LENGTH_LONG).show();
145.             }
146.         }
147.     } else
148.         Toast.makeText(appContext, "Fields can't be empty!", Toast.LENGTH_LONG).show();
149.
150.
151.     //showing defaults btns: dlt & upt
152.     btn_edit.setVisibility(View.VISIBLE);
153.     btn_update.setVisibility(View.GONE);
154.     btn_dlt.setVisibility(View.VISIBLE);
155.     btn_cancel.setVisibility(View.GONE);
156.     //replacing all tvs with ets
157.     //Name
158.     empName.setVisibility(View.VISIBLE);
159.     up_empName.setVisibility(View.GONE);
160.     //Dpt
161.     empDpt.setVisibility(View.VISIBLE);
162.     up_empDpt.setVisibility(View.GONE);
163.     //Yoj
164.     empYoj.setVisibility(View.VISIBLE);
165.     up_empYoj.setVisibility(View.GONE);

```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```

166.                //Contact
167.                empContact.setVisibility(View.VISIBLE);
168.                up_empContact.setVisibility(View.GONE);
169.
170.            }
171.        });
172.
173.        //perform cancel update operation
174.        btn_cancel.setOnClickListener(new View.OnClickListener() {
175.            @Override
176.            public void onClick(View v) {
177.
178.
179.                //showing defaults btns: dlt & upt
180.                btn_edit.setVisibility(View.VISIBLE);
181.                btn_update.setVisibility(View.GONE);
182.                btn_dlt.setVisibility(View.VISIBLE);
183.                btn_cancel.setVisibility(View.GONE);
184.                //replacing all tvs with ets
185.                //Name
186.                empName.setVisibility(View.VISIBLE);
187.                up_empName.setVisibility(View.GONE);
188.                //Dpt
189.                empDpt.setVisibility(View.VISIBLE);
190.                up_empDpt.setVisibility(View.GONE);
191.                //Y0J
192.                empY0J.setVisibility(View.VISIBLE);
193.                up_empY0J.setVisibility(View.GONE);
194.                //Contact
195.                empContact.setVisibility(View.VISIBLE);
196.                up_empContact.setVisibility(View.GONE);
197.
198.            }
199.        });
200.
201.        return rowView;
202.    }
203.
204. }

```

LayoutAdpater2.java(simple list for emp records)

```

1. package com.example.employeedb_app;
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.TextView;
9.
10. import androidx.annotation.NonNull;
11. import androidx.annotation.Nullable;
12.

```

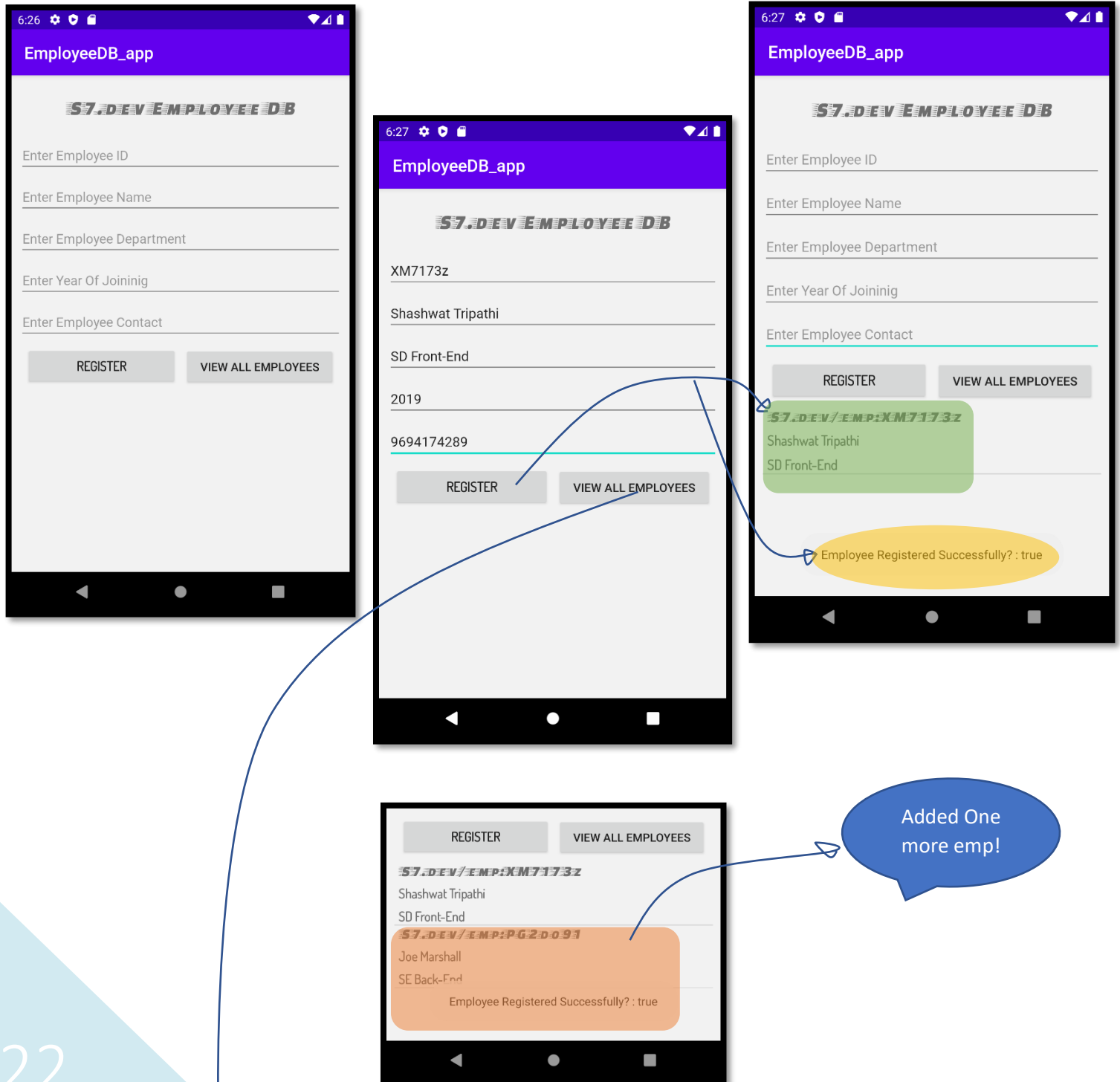
Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
13. import org.w3c.dom.Text;
14.
15. import java.util.List;
16. import java.util.Objects;
17.
18. public class LayoutAdapter2 extends ArrayAdapter<EmployeeModel> {
19.
20.     private Context appContext;
21.     private int itemLayout; //the xml layout we created to style each list item
22.     private List<EmployeeModel> allEmps;
23.     private DbHelper db_help;
24.
25.     public LayoutAdapter2(@NonNull Context context, int layout, List<EmployeeModel>
    > emps) {
26.         super(context, layout, emps);
27.         this.appContext = context;
28.         this.itemLayout = layout;
29.         this.allEmps = emps;
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 ListVie
    w.
35.      * */
36.     @NonNull
37.     @Override
38.     public View getView(final int position, @Nullable View rowView, @NonNull ViewG
    roup 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_help = new DbHelper(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' f
    ile we 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.         //getting views
52.         TextView ID = rowView.findViewById(R.id.tv_ID);
53.         TextView Name = rowView.findViewById(R.id.tv_name);
54.         TextView Dpt = rowView.findViewById(R.id.tv_dpt);
55.
56.         //setting views
57.         ID.setText("S7.dev/emp:" + Objects.requireNonNull(getItem(position)).getEmp
        _ID());
58.         Name.setText(Objects.requireNonNull(getItem(position)).getEmp_name());
59.         Dpt.setText(Objects.requireNonNull(getItem(position)).getEmp_dpt());
60.
```

Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

```
61.         return rowView;
62.     }
63. }
```

Snapshots:



Assignment 3: Develop an app for the demonstration of basic queries (create table, insert, select update, delete) in SQLite. Taking up a case of a company wishing to maintain records of their employees.

