**+**

ITMD 455/555 *Intelligent Device Applications* Lab 6

#### SQLite Database Queries- 50 points

**Introduction**. This lab will have you run through standard operations of creating, reading, updating, and

deleting from a database. This concept is known as CRUD which stands for Create, Read, Update and

Delete.

**Objective:** For this lab you will create a localized SQLite database called **Student\_db**, with the table name **student\_table**. Included will be a helper class that will allow you to work your query operations. A GUI will be included to work the CRUD methods. All activities on the dbase will be logged and thus viewed in LogCat as well.

Snapshot of sample GUI display and logcat output follow

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**STEP 1 Creating a New Android Project**

Create a new project called **Marks**.

Create an Empty Views Activity. Use the default layout name.

**STEP 2 Craft a GUI layout for activity\_main similarly as shown above. Include the**

**following for your setup.**

|  |  |  |
| --- | --- | --- |
| Element Type | Element id | Element text |
| TextView | **android:id="@+id/textView1"** | **android:text="Name"** |
| TextView | **android:id="@+id/textView2"** | **android:text="Marks"** |
| TextView | **android:id="@+id/textView3"** | **android:text="ID"** |
| EditText | **android:id="@+id/editText\_name"** |  |
| EditText | **android:id="@+id/editText\_marks"** |  |
| EditText | **android:id="@+id/editText\_id"** |  |
| Button | **android:id="@+id/button\_add"** | **android:text="Add Data"** |
| Button | **android:id="@+id/button\_viewAll"** | **android:text="View All"** |
| Button | **android:id="@+id/button\_update"** | **android:text="Update"** |
| Button | **android:id="@+id/button\_delete"** | **android:text="Delete"** |

**STEP 3 Create a class called DatabaseHelper in your package**

This class is your supportive API class to provide merely the necessary query responses for your database.

The functionality of the class is thus:

* + - Create a new class **DatabaseHelper** that extends **SQLiteOpenHelper**.
    - Have the **DatabaseHelper** constructor call the **super** class constructor.
    - Override **onCreate()** method to create table(s).
    - Override **onUpgrade()** to drop old tables (as an option) and create new ones and/or do whatever actions are needed.
    - Work any of your CRUD implementations that call SQLite APIs.

Enter imports/code as follows:

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

**public class** DatabaseHelper **extends** SQLiteOpenHelper {

**public static final** String ***DATABASE\_NAME*** = **"Student.db"**;  
 **public static final** String ***TABLE\_NAME*** = **"student\_table"**;  
 **public static final** String ***COL\_1*** = **"ID"**;  
 **public static final** String ***COL\_2*** = **"NAME"**;  
 **public static final** String ***COL\_3*** = **"MARKS"**;  
  
 **public** DatabaseHelper(Context context) {  
 **super**(context, ***DATABASE\_NAME***, **null**, 1);  
 }  
  
  @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 db.execSQL(**"create table "** + ***TABLE\_NAME*** + **" "** +  
 **"(ID INTEGER PRIMARY KEY AUTOINCREMENT,"** +  
 **"NAME TEXT,"** +  
 **"MARKS INTEGER)"**);

}  
  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  
 db.execSQL(**"DROP TABLE IF EXISTS "** + ***TABLE\_NAME***);  
 onCreate(db);  
 }  
  
 **public boolean** insertData(String name, String marks) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 ContentValues contentValues = **new** ContentValues();  
 contentValues.put(***COL\_2***, name);  
 contentValues.put(***COL\_3***, marks);  
 **long** result = db.insert(***TABLE\_NAME***, **null**, contentValues);  
 **if** (result == -1)  
 **return false**;  
 **else  
 return true**;  
 }  
  
 **public** Cursor getAllData() {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 Cursor res = db.rawQuery(**"select \* from "** + ***TABLE\_NAME***, **null**);  
 **return** res;  
 }  
  
 **public boolean** updateData(String id, String name, String marks) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 ContentValues contentValues = **new** ContentValues();  
 contentValues.put(***COL\_1***, id);  
 contentValues.put(***COL\_2***, name);  
 contentValues.put(***COL\_3***, marks);  
 db.update(***TABLE\_NAME***, contentValues, **"ID = ?"**, **new** String[]{id});  
 **return true**;  
 }  
  
 **public** Integer deleteData(String id) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 **return** db.delete(***TABLE\_NAME***, **"ID = ?"**, **new** String[]{id});  
 }  
 }

**STEP 4 Code MainActivty to include methods to work with your DatabaseHelper class**

Add in imports for your Activity as follows

**import** android.database.Cursor;  
 **import** android.util.Log;  
 **import** android.view.View;  
 **import** android.widget.Button;  
 **import** android.widget.EditText;

Include objects for your Activity as follows

DatabaseHelper **myDb**;

*//elements*

EditText **editName**, **editMarks**, **editId**;  
 Button **btnAddData**;  
 Button **btnViewAll**;

Button **btnUpdate**;  
 Button **btnDelete**;

Inside your **onCreate** method, assign appropriate ids from your layout to each of the above

elements.

Also in your **onCreate** method, for your **myDb** object, include the following object instantiation

**myDb** = **new** DatabaseHelper(**this**);

Finish your **onCreate** method with the following button listeners, that when triggered, call

functions from your DatabaseHelper class.

**btnDelete**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Integer deletedRows = **myDb**.deleteData(**editId**.getText().toString());  
 }  
 }  
 );  
  
 **btnUpdate**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **boolean** isUpdate = **myDb**.updateData(**editId**.getText().toString(),  
 **editName**.getText().toString(),   
 **editMarks**.getText().toString());  
 }  
 }  
 );

**btnAddData**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **boolean** isInserted = **myDb**.insertData(**editName**.getText().toString(),  
 **editMarks**.getText().toString());  
 }  
 }  
 );

**btnViewAll**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Cursor res = **myDb**.getAllData();

StringBuffer buffer = **new** StringBuffer();

buffer.append("Record Data...\n");

**while** (res.moveToNext()) { *//cycle thru result set*  
 buffer.append(**"Id :"** + res.getString(0) + **"\n"**);  
 buffer.append(**"Name :"** + res.getString(1) + **"\n"**);  
 buffer.append(**"Marks :"** + res.getString(2) + **"\n\n"**);  
 }  
 *// Show all data* Log.*i*(**"Data"**, buffer.toString());

}  
 }  
 );

} *//end onCreate()*

**STEP 5 Run app at this point**

Run your app and test it. Add 4 records of choice into the db. **No need to enter anything in the id Text field, as that is used only for update and delete actions**.

Press your View All button and check your logcat if data for your inserted records are

now displayed.

**STEP 6 Modify your code to display messages for each operation below to your logcat**

Include the following messages/output broadcasting to logcat for the given operations

1. A message that says “record added”
2. A message that says “record deleted”
3. A message that says “no record deleted”
4. A message that says “record updated”
5. A message that says “no record updated”
6. All Data records

Run your app again to get your messages to the logcat output as follows

-For any updates to data, make sure to include a record id (which was/is automatically generated). Update one of your records by the id. Try updating an id that doesn’t exist.

-For a delete, just include the id. Delete the last record added in. Try deleting an id that doesn’t exist.

-Finally make sure to show all your records after the adds/update/delete actions.

**STEP 7 Submit results**

Include a snapshot (single snapshot is fine) of all your messages for items a-f including   
 modifcations, in **Step 6** above.

Include into Blackboard your java files, xml layout code and snapshot(s) into

appropriate pdf file(s) for credit in addition to a zip file of your entire project files.