18. Develop a program to perform (Search and Update) database operations using SQLite Database.

To perform database operations on android devices, the SQLite is used in Android. It is an open-source relational database that is used for storing, manipulating, or retrieving persistent data from the database. By default, SQLite is embedded in android, thus there is no need to perform any database setup or administration task.

The SQLiteOpenHelper class has two constructors. These are:

Constructor	**Uses**
SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version)	To create an object that can create, open and manage the database.
SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version, DatabaseErrorHandler errorHandler)	To create an object that can create, open and manage the database, and to also specify the error handler.

Methods of SQLiteDatabase class:

Method	**Uses**
void execSQL(String sql)	To execute the sql query.
long insert(String table, String nullColumnHack, ContentValues values)	To insert a record on the database. The table name is specified by the table. Completely null values are not allowed by the nullColumnHack. The null values are stored by android, if the second argument is null and the values are empty. The values to be stored are specified by the third argument.
int update(String table, ContentValues values, String whereClause, String[] whereArgs)	To update a row.
Cursor query(String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy)	To return a cursor over the result set.

Building and executing queries

```
private static final String DB_NAME = "MyCollegeDB";

private static final String TABLE_S_AND_G = " StudentsAndGrades";

 public static final String TABLE_ROW_ID = "_id";

 public static final String TABLE_ROW_NAME = "name";

public static final String TABLE_ROW_SCORE = "score";

 String name = "Onkar";

 int score = 95;

// Add all the details to the table

String query = "INSERT INTO " + TABLE_S_AND_G + " (" + TABLE_ROW_NAME + ", " + TABLE_ROW_SCORE +
") " + "VALUES (" + "'" + name + "'" + ", " + score + ");";
```

The above lines add a new entry to a database and incorporates Java variables into the SQL statement

```
// This is the actual database

private SQLiteDatabase db;

// Create an instance of our internal CustomSQLiteOpenHelper class

CustomSQLiteOpenHelper helper = new CustomSQLiteOpenHelper(context);

// Get a writable database

db = helper.getWritableDatabase();

// Run the query

db.execSQL(query);
```

When adding data to the database, use execSQL, and when getting data from the database, use the rawQuery method, as follows:

```
Cursor c = db.rawQuery(query, null);
```

Notice that the rawQuery method returns an object of type Cursor.

Next method, called delete, will delete a record from the database if it has a matching value in the name column to that of the passed-in name parameter. It achieves this using the SQL DELETE keyword. Add the delete method to the DataManager class:

```
// Delete a record

 public void delete(String name){
```

```
// Delete the details from the table if already exists

String query = "DELETE FROM " + TABLE_N_AND_A + " WHERE " + TABLE_ROW_NAME + " = '" + name + "';";
Log.i("delete() = ", query);

db.execSQL(query); }
```

The selectAll method, which also does as the name suggests. It achieves this with a SELECT query using the * parameter, which is equivalent to specifying all the columns individually. The method returns a Cursor.

Add the selectAll method to the DataManager class as follows:

```
// Get all the records

public Cursor selectAll()

 { Cursor c = db.rawQuery("SELECT *" +" from " + TABLE_N_AND_A, null);

return c; }
```

Add a searchName method, which has a String parameter for the name the user wants to search for. It also returns a Cursor, which will contain all the entries that were found. The SQL statement uses SELECT, FROM, and WHERE to achieve this

```
// Find a specific record

 public Cursor searchName(String name)

 { String query = "SELECT " + TABLE_ROW_ID + ", " + TABLE_ROW_NAME + ", " + TABLE_ROW_AGE + " from " + TABLE_N_AND_A + " WHERE " + TABLE_ROW_NAME + " = '" + name + "';";

Log.i("searchName() = ", query);

 Cursor c = db.rawQuery(query, null);

 return c; }
```

Q.1. List and explain different storage options in Android.

Q.2. Implement basic contacts applications that allows insertion, deletion and modification of contacts example on tutorialspoint.