DOP: / /2023 **DOS: / /2023**

Experiment No:6

Title: Developing user interactive Database applications (Using SQLite or other) in Android.

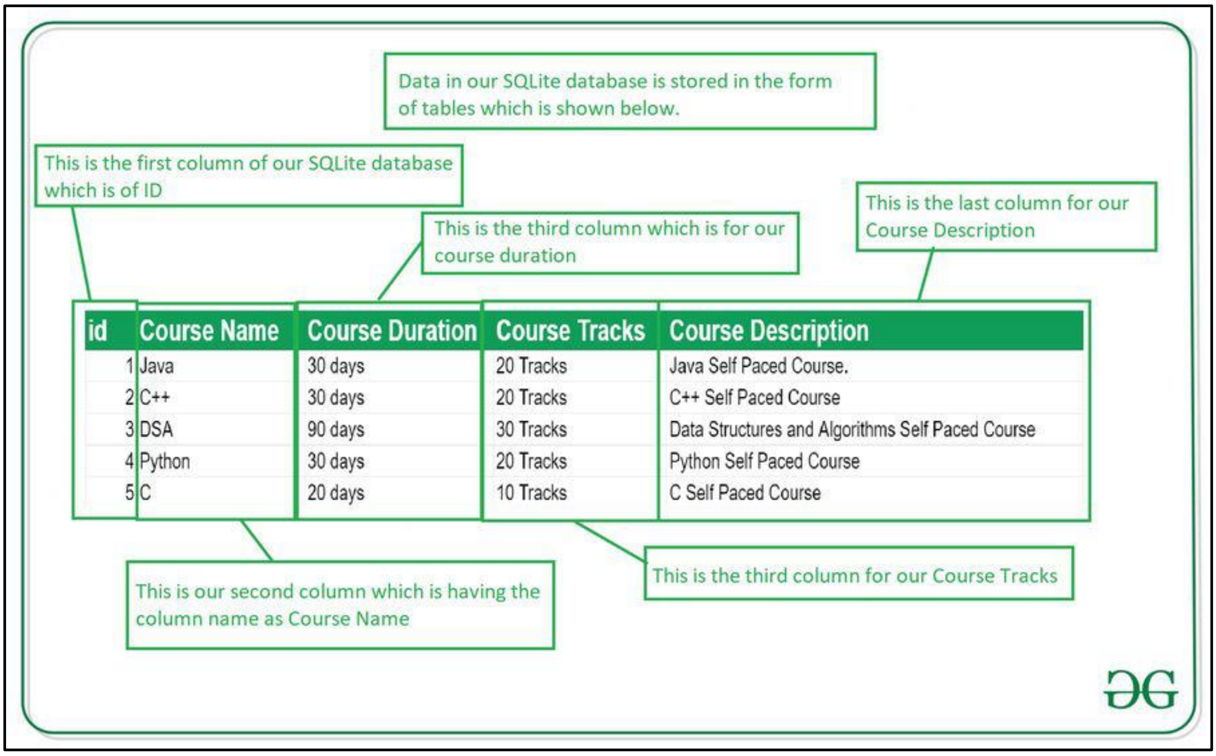
Theory:

What is SQLite Database?

SQLite Database is an open-source database provided in Android which is used to store data inside the user's device in the form of a Text file. We can perform so many operations on this data such as adding new data, updating, reading, and deleting this data. SQLite is an offline database that is locally stored in the user's device and we do not have to create any connection to connect to this database**.**

How Data is Being Stored in the SQLite Database?

Data is stored in the SQLite database in the form of tables. When we stored this data in our SQLite database it is arranged in the form of tables that are similar to that of an excel sheet. Below is the representation of our SQLite database which we are storing in our SQLite database.



Data in our SQLite database is stored in the form of tables which is shown below.

This is the first column of our SQLite database which is of ID

This is the third column which is for our course duration

This is the last column for our Course Description

id	Course Name	Course Duration	Course Tracks	Course Description
1	Java	30 days	20 Tracks	Java Self Paced Course.
2	C++	30 days	20 Tracks	C++ Self Paced Course
3	DSA	90 days	30 Tracks	Data Structures and Algorithms Self Paced Course
4	Python	30 days	20 Tracks	Python Self Paced Course
5	C	20 days	10 Tracks	C Self Paced Course

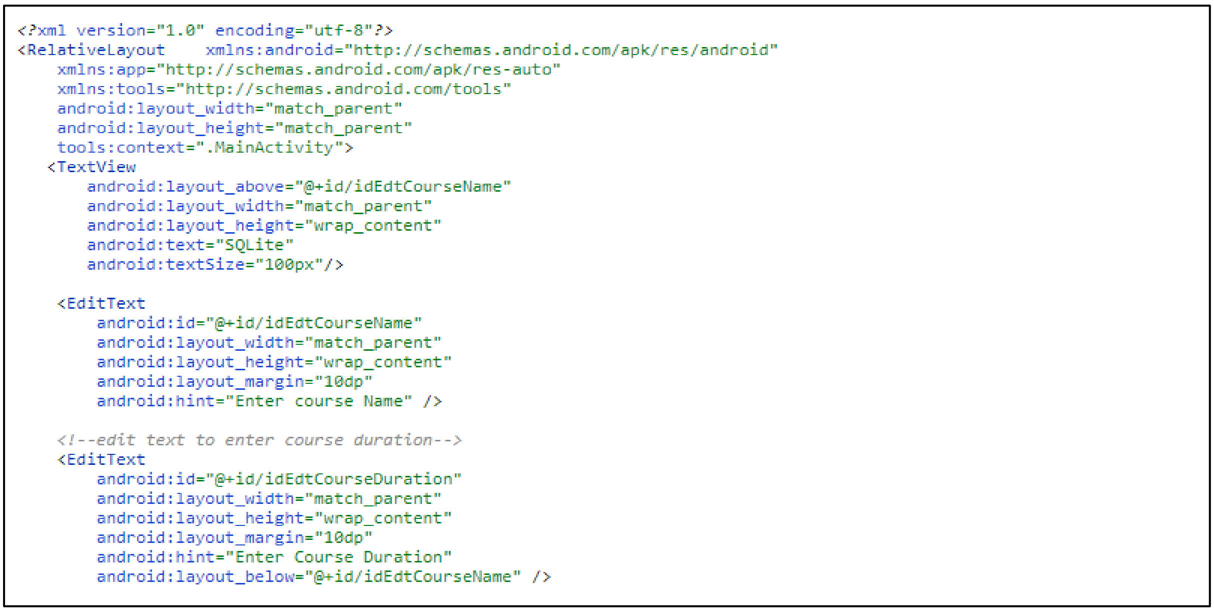
This is our second column which is having the column name as Course Name

This is the third column for our Course Tracks

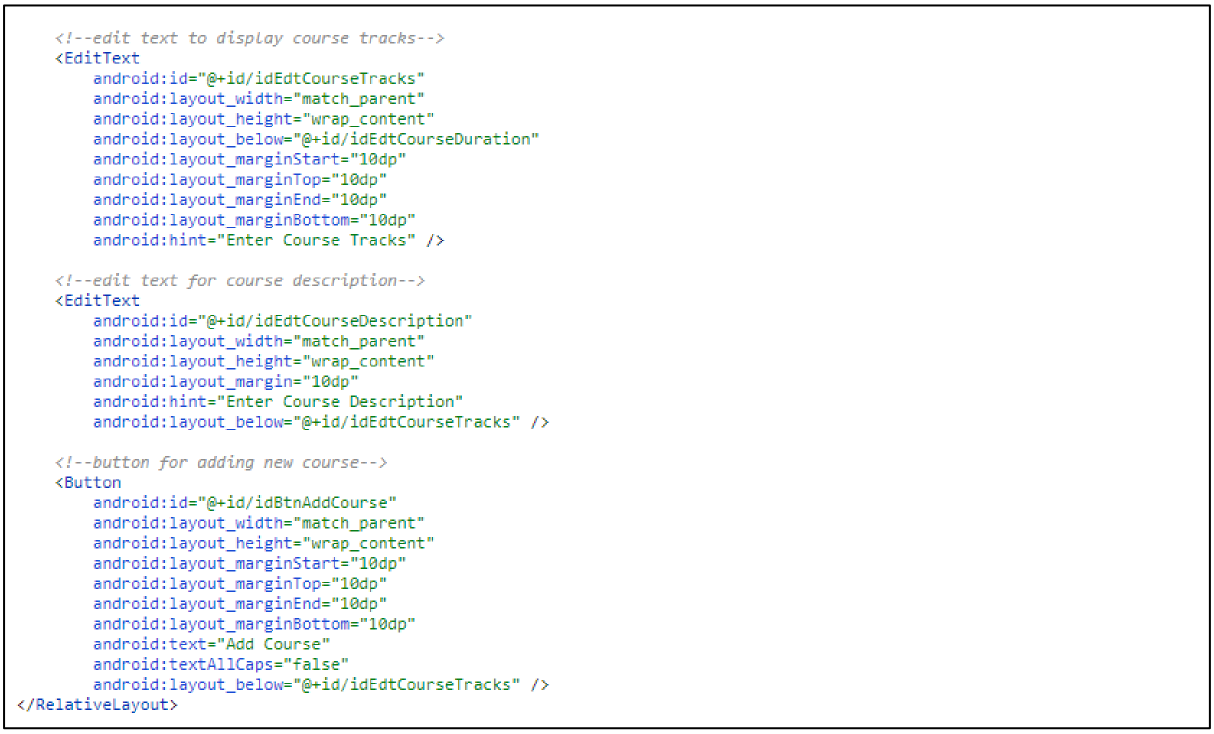
Important Methods in SQLite Database:

Method	Description
getColumnNames()	This method is used to get the Array of column names of our SQLite table.
getCount()	This method will return the number of rows in the cursor.
isClosed()	This method returns a Boolean value when our cursor is closed.
getColumnCount()	This method returns the total number of columns present in our table.
getColumnName(int columnIndex)	This method will return the name of the column when we passed the index of our column in it.
getColumnIndex(String columnName)	This method will return the index of our column from the name of the column.
getPosition()	This method will return the current position of our cursor in our table.

Input:

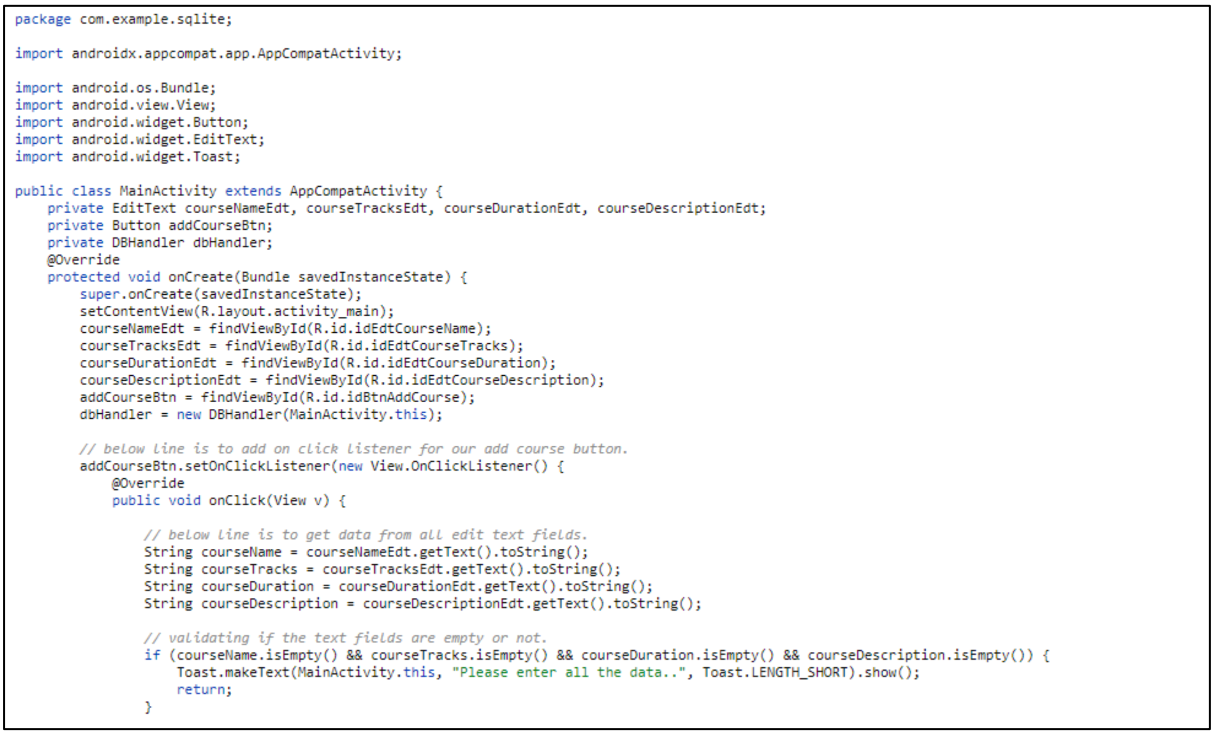


```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
   <TextView
       android:layout_above="@+id/idEdtCourseName"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:text="SQLite"
       android:textSize="100px"/>

    <EditText
        android:id="@+id/idEdtCourseName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter course Name" />

    <!--edit text to enter course duration-->
    <EditText
        android:id="@+id/idEdtCourseDuration"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter Course Duration"
        android:layout_below="@+id/idEdtCourseName" />
```

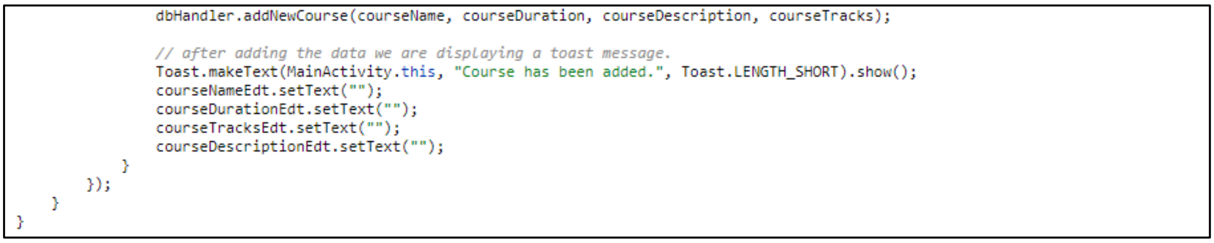


```
    <!--edit text to display course tracks-->
    <EditText
        android:id="@+id/idEdtCourseTracks"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/idEdtCourseDuration"
        android:layout_marginStart="10dp"
        android:layout_marginTop="10dp"
        android:layout_marginEnd="10dp"
        android:layout_marginBottom="10dp"
        android:hint="Enter Course Tracks" />

    <!--edit text for course description-->
    <EditText
        android:id="@+id/idEdtCourseDescription"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:hint="Enter Course Description"
        android:layout_below="@+id/idEdtCourseTracks" />

    <!--button for adding new course-->
    <Button
        android:id="@+id/idBtnAddCourse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="10dp"
        android:layout_marginEnd="10dp"
        android:layout_marginBottom="10dp"
        android:text="Add Course"
        android:textAllCaps="false"
        android:layout_below="@+id/idEdtCourseTracks" />
</RelativeLayout>
```

Java file:

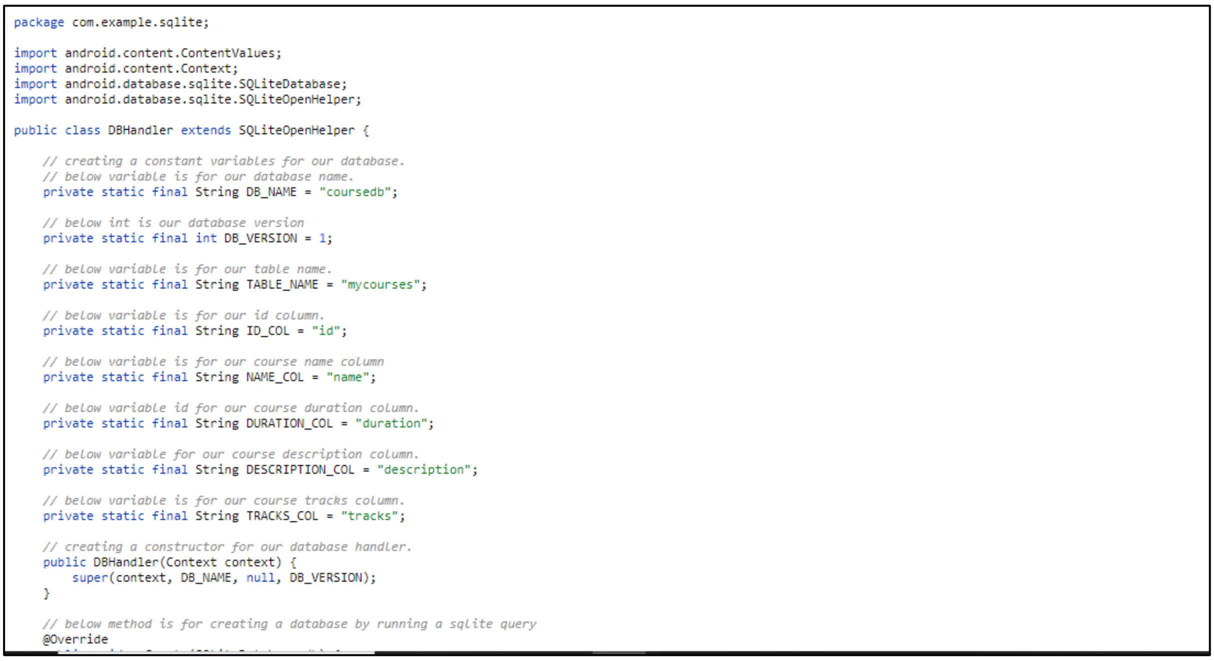


```
package com.example.sqlite;

import androidx.appcompat.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 {
    private EditText courseNameEdt, courseTracksEdt, courseDurationEdt, courseDescriptionEdt;
    private Button addCourseBtn;
    private DBHandler dbHandler;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        courseNameEdt = findViewById(R.id.idEdtCourseName);
        courseTracksEdt = findViewById(R.id.idEdtCourseTracks);
        courseDurationEdt = findViewById(R.id.idEdtCourseDuration);
        courseDescriptionEdt = findViewById(R.id.idEdtCourseDescription);
        addCourseBtn = findViewById(R.id.idBtnAddCourse);
        dbHandler = new DBHandler(MainActivity.this);

        // below line is to add on click listener for our add course button.
        addCourseBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                // below line is to get data from all edit text fields.
                String courseName = courseNameEdt.getText().toString();
                String courseTracks = courseTracksEdt.getText().toString();
                String courseDuration = courseDurationEdt.getText().toString();
                String courseDescription = courseDescriptionEdt.getText().toString();

                // validating if the text fields are empty or not.
                if (courseName.isEmpty() && courseTracks.isEmpty() && courseDuration.isEmpty() && courseDescription.isEmpty()) {
                    Toast.makeText(MainActivity.this, "Please enter all the data..", Toast.LENGTH_SHORT).show();
                    return;
                }
```

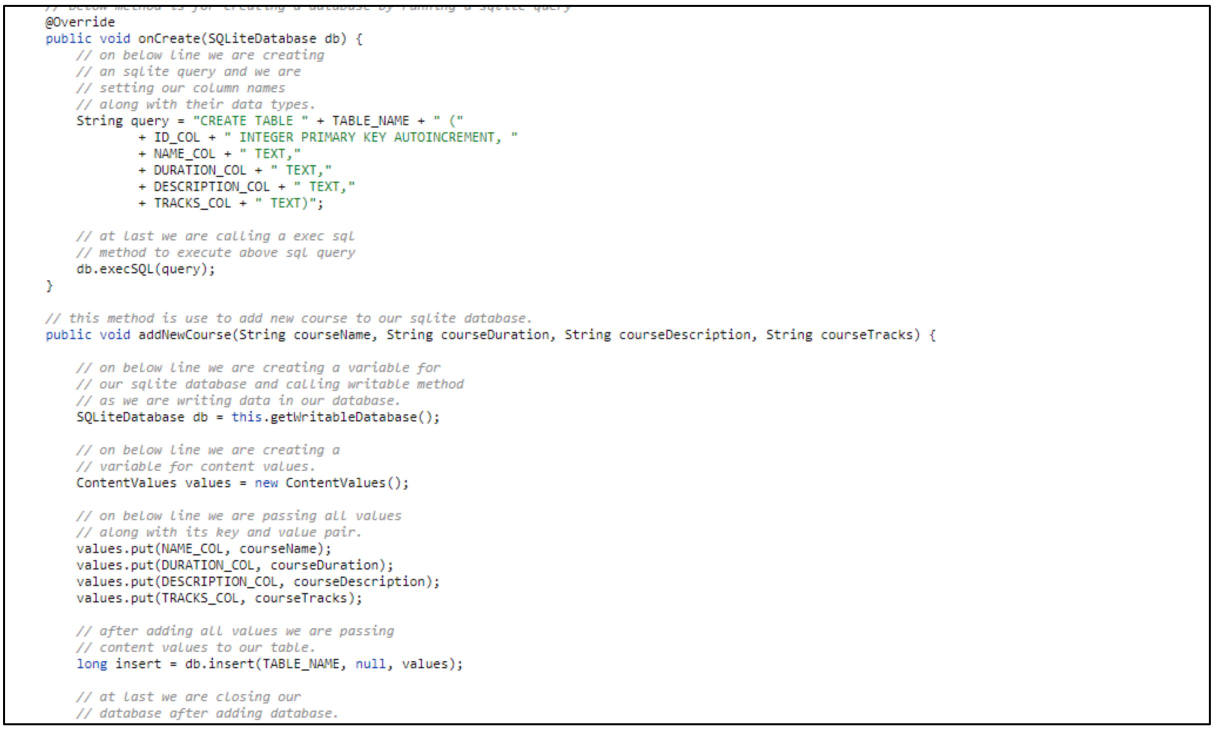


```
                dbHandler.addNewCourse(courseName, courseDuration, courseDescription, courseTracks);

                // after adding the data we are displaying a toast message.
                Toast.makeText(MainActivity.this, "Course has been added.", Toast.LENGTH_SHORT).show();
                courseNameEdt.setText("");
                courseDurationEdt.setText("");
                courseTracksEdt.setText("");
                courseDescriptionEdt.setText("");
            }
        });
    }
}
```

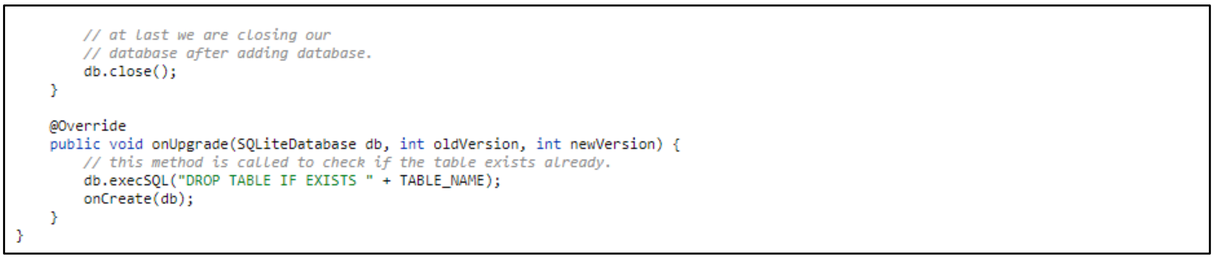
DBHandler.java:



```
package com.example.sqlite;

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

public class DBHandler extends SQLiteOpenHelper {

    // creating a constant variables for our database.
    // below variable is for our database name.
    private static final String DB_NAME = "coursedb";

    // below int is our database version
    private static final int DB_VERSION = 1;

    // below variable is for our table name.
    private static final String TABLE_NAME = "mycourses";

    // below variable is for our id column.
    private static final String ID_COL = "id";

    // below variable is for our course name column
    private static final String NAME_COL = "name";

    // below variable id for our course duration column.
    private static final String DURATION_COL = "duration";

    // below variable for our course description column.
    private static final String DESCRIPTION_COL = "description";

    // below variable is for our course tracks column.
    private static final String TRACKS_COL = "tracks";

    // creating a constructor for our database handler.
    public DBHandler(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    // below method is for creating a database by running a sqlite query
    @Override
```

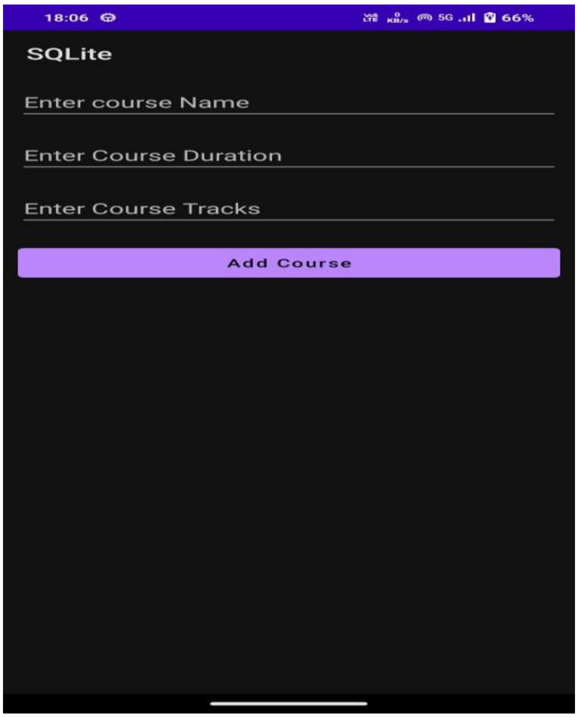


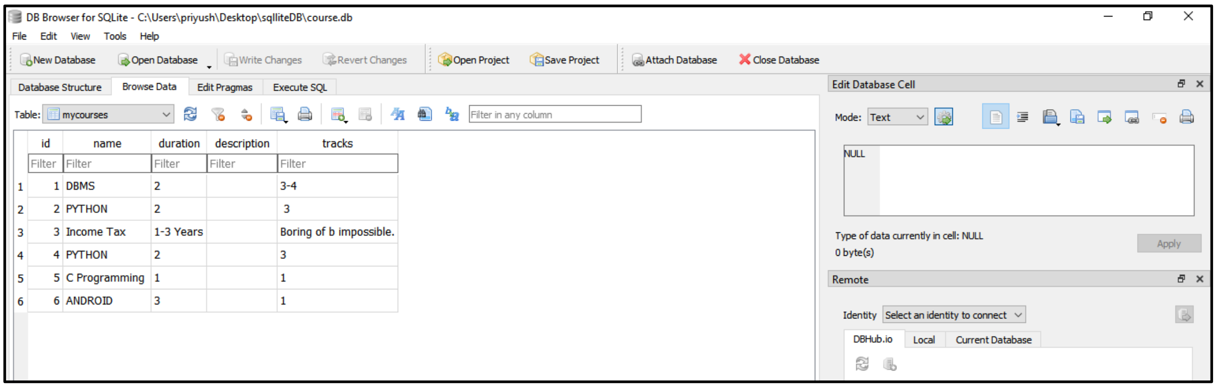
```
    @Override
    public void onCreate(SQLiteDatabase db) {
        // on below line we are creating
        // an sqlite query and we are
        // setting our column names
        // along with their data types.
        String query = "CREATE TABLE " + TABLE_NAME + " ("
                + ID_COL + " INTEGER PRIMARY KEY AUTOINCREMENT, "
                + NAME_COL + " TEXT,"
                + DURATION_COL + " TEXT,"
                + DESCRIPTION_COL + " TEXT,"
                + TRACKS_COL + " TEXT)";

        // at last we are calling a exec sql
        // method to execute above sql query
        db.execSQL(query);
    }

    // this method is use to add new course to our sqlite database.
    public void addNewCourse(String courseName, String courseDuration, String courseDescription, String courseTracks) {

        // on below line we are creating a variable for
        // our sqlite database and calling writable method
        // as we are writing data in our database.
        SQLiteDatabase db = this.getWritableDatabase();

        // on below line we are creating a
        // variable for content values.
        ContentValues values = new ContentValues();

        // on below line we are passing all values
        // along with its key and value pair.
        values.put(NAME_COL, courseName);
        values.put(DURATION_COL, courseDuration);
        values.put(DESCRIPTION_COL, courseDescription);
        values.put(TRACKS_COL, courseTracks);

        // after adding all values we are passing
        // content values to our table.
        long insert = db.insert(TABLE_NAME, null, values);

        // at last we are closing our
        // database after adding database.
```



```
        // at last we are closing our
        // database after adding database.
        db.close();
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // this method is called to check if the table exists already.
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }
}
```

Output:





DB Browser for SQLite - C:\Users\priyush\Desktop\sqlliteDB\course.db

Table: mycourses

	id	name	duration	description	tracks
	Filter	Filter	Filter	Filter	Filter
1	1	DBMS	2		3-4
2	2	PYTHON	2		3
3	3	Income Tax	1-3 Years		Boring of b impossible.
4	4	PYTHON	2		3
5	5	C Programming	1		1
6	6	ANDROID	3		1

Mode: Text

NULL

Type of data currently in cell: NULL
0 byte(s)

Identity Select an identity to connect

Conclusion: - Hence successfully performed Developing user interactive Database applications (Using SQLite or other) in Android.