Hybrid Application Development Lab

Dr. Mahesh Kumar K.M

Asst. Prof., Dept. of Information Science and Engineering, NMIT-Bangalore

December 14, 2021


android

Android SQLite Database

SQLite Databases are regular Relational DBMS designed for lower or mid range devices. Some of the features of SQLite databases are listed below:

- SQLite database along with the **packages sums up to the size of 600Kb** which is **suitable for mobile devices**
- SQLite databases are **serverless unlike MySQL or Oracle** databases
- In SQLite datbases, **the data is encrypted and stored on to regular files** and **the read-write operations are directly performed as if one accessing the files which makes the database run very agile**.
- SQLite is **ACID complaint**, does not support advanced concepts such as Triggers
- SQLite databases are **written in C and is part of android libraries**, there is no need to install the package explicitly

SQLiteOpenHelper Class

- The APIs related to SQLite database are managed by an Android component called SQLite Manager.
- It is the component which provides the functions, methods, & classes for accessing, manipulating and versioning of SQLite databases.
- The SQLite Manager provides a class to do all the operations on the database called as SQLiteOpenHelper Class.
- To use the features of the SQLiteOpenHelper class, the model class must extend and make the class as inheritor of the features from Super class SQLiteOpenHelper as shown below

```
public class DBHelper extends SQLiteOpenHelper{}
```

DBHelper - Constructor

- DBHelper constructor takes context, Database name, cursor factory, Database version as arguments, and override the functions.
- The constructor should override the constructor of the super class taking same params as the constructor arguments as shown below:

```
public DBHelper(@Nullable Context context,@Nullable String name,@Nullable
SQLiteDatabase.CursorFactory factory,int version) {
super(context, dbName, null, dbVersion);
}
```

- The context specifies the current class context, The second argument is a Database name which should be defined earlier as String and the CursorFactory specifies whether we are using default cursor object provided by SQLiteOpenHelper class or using custom cursor.
- By default, this object is set to NULL to tell that we are using default cursor object. The final argument is used for the database version.

DBHelper - OnCreate()

- The second function to be override by extending the SQLiteOpenHelper class is called as onCreate(SQLiteDatabase db) that takes SQLiteDatabase object pointing to the current database.
- It is called only once when the DB is created for the first time.
- The override function is as shown below:

```
@Override
public void onCreate(SQLiteDatabase db) {
// Logic to create Database
}
```

DBHelper - onUpgrade()

- Another default override function is onUpgrade(SQLiteDatabase db, int prev_version, int new version).
- It is used to upgrade database to another version or any preliminary sanitisation function like dropping tables if already exists, etc.

```
@Override
public void onUpgrade(SQLiteDatabase db, int prev_version, int new version) {
// Logic to upgrade Database & perform preliminary setups
}
```

DBHelper - onDowngrade() and close()

- Similar to onUpgrade, the database can be downgraded using onDowngrade function which recieves the same arguments as onUpgrade and syntax is similar as well.
- It is shown below.

```
@Override
public void onDowngrade(SQLiteDatabase db, int old_version, int new version) {
// Logic to downgrade Database
}
```

- Another function that is provided by SQLiteOpenHelper is close(). which is used to close the database upon the completion of operations on database.

SQLiteDatabase Class

- The Android System also provides SQLiteDatabase class that facilitates the process of table creation, and basic CRUD operation on the database.
- The following are some of the member functions of SQLiteDatabase class that are often used:

1. execSQL()
2. getReadableDatabase() and getWritableDatabase()
3. insert()
4. delete()
5. update()
6. query()
7. rawQuery()

SQLiteDatabase Class contd...

1. execSQL function is used to execute raw SQL statements which is called by SQLiteDatabase object

```
SQLiteDatabase db ; //instance of SQliteDatabase object
db.execSQL(String SQL_statement) ; // SQL statement
```

2. getReadableDatabase() and getWritableDatabase() functions are used to open the Database file in Read and Write/ mode respectively. Both are demonstrated here

```
SQLiteDatabase db = = context.getWritableDatabase(); // DB is opened in Write mode
SQLiteDatabase db = = context.getReadableeDatabase(); // DB is opened in Read mode
```

SQLiteDatabase Class contd...

3. To insert data into database, data is actually inserted in the form of (Key,value) pair which is prepared by the ContentValue object. The default values can also be made null with the help of nullColumnHack argument. The API is as shown below,

```
long insert(String tablename, String nullColumnHack, ContentValues values);
```

4. To delete, we use delete API, which takes three arguments, they are tablename, where clause and arguments for the where arguments. It is as shown below,

```
int delete(String table, String whereClause, String[] whereArgs);
```

5. To update, we use update API, which takes three arguments, they are tablename, where clause and arguments for the where arguments. It is as shown below,

```
int update(String table, ContentValues values, String whereClause, String[]
    whereArgs);
```

SQLiteDatabase Class contd...

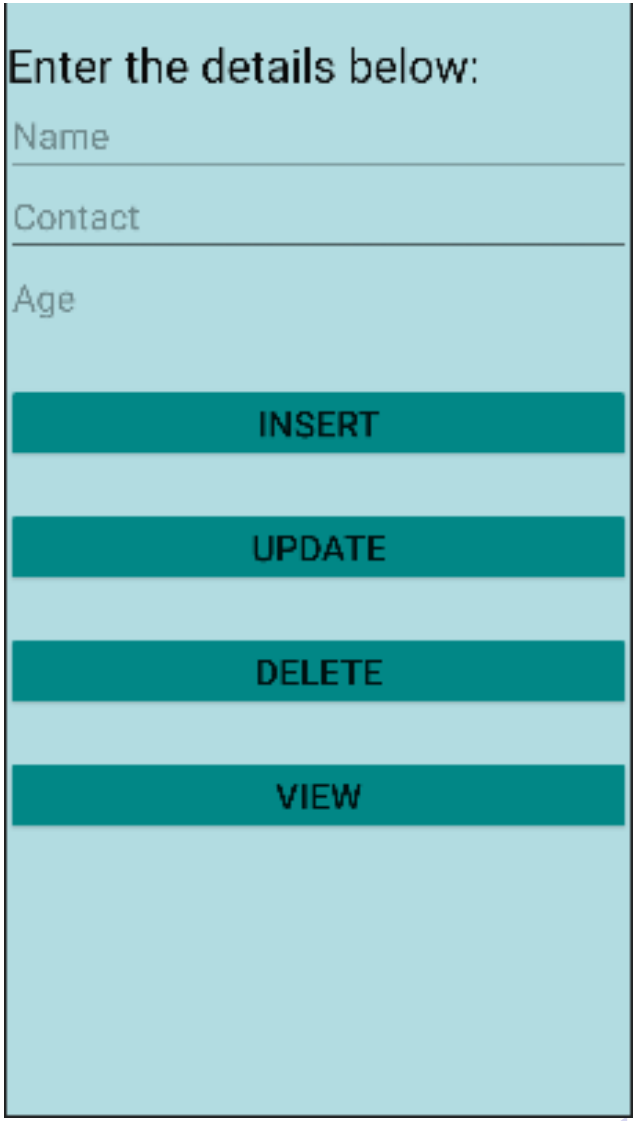
6. to execute a query we use query function which returns a cursor object. A cursor object points to the first row output of the SQL statement that got executed and its iteratable. The query functions uses arguments such as tablename, column name, selection and its arguments, groupby, having and orderby arguments respectively depending upon the user choices. Not all the arguments are usable at the same time and they can be set to null if its not being used.

```
Cursor cur = query(String table, String[] columns, String selection, String[]
    selectionArgs, String groupBy, String having, String orderBy, String limit);
```

7. rawQuery function is also used to execute the SQL query that returns a cursor object to the ResultSet.

```
Cursor cur = rawQuery(String sql, String[] selectionArgs);
```

SQLiteDatabase - UI



SQLiteDatabase - activity_main.xml I

```
<?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"
    android:background="#B2DCE1"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/tv1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:fontFamily="sans-serif"
        android:text="Enter the details below:"
        android:textColor="#090909"
        android:textSize="30sp" />

    <EditText
        android:id="@+id/name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/tv1"
        android:hint="Name"
        android:inputType="textPersonName"
        android:textSize="24dp"></EditText>


    <EditText
        android:id="@+id/contact"
```

SQLiteDatabase - activity_main.xml II

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/name"
    android:hint="Contact"
    android:inputType="number"
    android:textSize="24dp"></EditText>

<EditText
    android:id="@+id/age"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/contact"
    android:hint="Age"
    android:inputType="number"
    android:textSize="24dp"></EditText>


<Button
    android:id="@+id/btnInsert"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/age"
    android:layout_marginTop="30dp"
    android:backgroundTint="@color/teal_700"
    android:text="Insert"
    android:textSize="24dp"></Button>

<Button
    android:id="@+id/btnUpdate"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/btnInsert"
```

SQLiteDatabase - activity_main.xml III

```
        android:layout_marginTop="30dp"
        android:backgroundTint="@color/teal_700"
        android:text="Update"
        android:textSize="24dp"></Button>

    <Button
        android:id="@+id/btnDelete"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/btnUpdate"
        android:layout_marginTop="30dp"
        android:backgroundTint="@color/teal_700"
        android:text="Delete"
        android:textSize="24dp"></Button>

    <Button
        android:id="@+id/btnView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/btnDelete"
        android:layout_marginTop="30dp"
        android:backgroundTint="@color/teal_700"
        android:text="View"
        android:textSize="24dp"></Button>


</RelativeLayout>
```

SQLiteDatabase - DBHelper.java I

```
package com.example.sqllitedatabase;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class DBHelper extends SQLiteOpenHelper {


    public DBHelper(@Nullable Context context) {
        super(context, "Userdata.db", null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create Table Userdetails(name TEXT primary key, contact TEXT, age TEXT)");

    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("drop Table if exists Userdetails");
    }

    public boolean insertData(String name,String contact,String age){
        SQLiteDatabase db = this.getWritableDatabase();
```

SQLiteDatabase - DBHelper.java II

```
    ContentValues contentValues = new ContentValues();
    contentValues.put("name",name);
    contentValues.put("contact",contact);
    contentValues.put("age",age);
    long result=db.insert("Userdetails", null, contentValues);
    if(result==-1)
        return  false;
    else
        return true;
}

public boolean updateData(String name,String contact,String age){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("contact",contact);
    contentValues.put("age",age);
    Cursor cursor = db.rawQuery("Select * from Userdetails where name = ?", new String[] {name});
    if(cursor.getCount() > 0) {
        long result = db.update("Userdetails", contentValues, "name=?", new String[] {name});
        if (result == -1)
            return false;
        else
            return true;
    } else {
        return false;
    }
}

public boolean deleteData(String name){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
```

SQLiteDatabase - DBHelper.java III

```
        Cursor cursor = db.rawQuery("Select * from Userdetails where name = ?", new String[] {name});
        if(cursor.getCount() > 0) {
            long result = db.delete("Userdetails", "name=?", new String[] {name});
            if (result == -1)
                return false;
            else
                return true;
        } else {
            return false;
        }
    }

    public Cursor viewData () {
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor cursor = db.rawQuery("Select * from Userdetails", null);
        return cursor;
    }


}
```

SQLiteDatabase - MainActivity.java I

```
package com.example.sqllitedatabase;

import android.database.Cursor;
import android.support.v7.app.AlertDialog;
import android.support.v7.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 {

    EditText name, contact, age;
    Button insert, update, delete, view;
    DBHelper DB;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        name=findViewById(R.id.name);
        contact=findViewById(R.id.contact);
        age=findViewById(R.id.age);

        insert=findViewById(R.id.btnInsert);
        update=findViewById(R.id.btnUpdate);
        delete=findViewById(R.id.btnDelete);
        view=findViewById(R.id.btnView);

        DB = new DBHelper(this);
```

SQLiteDatabase - MainActivity.java II

```
insert.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String nameTXT = name.getText().toString();
        String contactTXT = contact.getText().toString();
        String ageTXT = age.getText().toString();
        boolean qryStatus = DB.insertData(nameTXT,contactTXT,ageTXT);
        if (qryStatus==true)
            Toast.makeText(MainActivity.this, "New Record Created", Toast.LENGTH_SHORT).show();
        else
            Toast.makeText(MainActivity.this, "New Record Creation Failed", Toast.LENGTH_SHORT).show();
    }
});

update.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String nameTXT = name.getText().toString();
        String contactTXT = contact.getText().toString();
        String ageTXT = age.getText().toString();
        boolean qryStatus = DB.updateData(nameTXT,contactTXT,ageTXT);
        if (qryStatus==true)
            Toast.makeText(MainActivity.this, "Record Updated", Toast.LENGTH_SHORT).show();
        else
            Toast.makeText(MainActivity.this, "Record Updation Failed", Toast.LENGTH_SHORT).show();
    }
});

delete.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
```

SQLiteDatabase - MainActivity.java III

```
            String nameTXT = name.getText().toString();

            boolean qryStatus = DB.deleteData(nameTXT);
            if (qryStatus==true)
                Toast.makeText(MainActivity.this, "Record Deleted", Toast.LENGTH_SHORT).show();
            else
                Toast.makeText(MainActivity.this, "Record Deletion Failed", Toast.LENGTH_SHORT).show();
        }
    });

    view.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Cursor res = DB.viewData();
            if(res.getCount()==0)
                Toast.makeText(MainActivity.this, "No Record Exist", Toast.LENGTH_SHORT).show();
            else {
              StringBuffer buffer = new StringBuffer();
              while(res.moveToNext()){
                  buffer.append("Name :"+res.getString(0)+"\n");
                  buffer.append("Contact :"+res.getString(1)+"\n");
                  buffer.append("Age :"+res.getString(2)+"\n");
              }
                AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
                builder.setCancelable(true);
                builder.setTitle("User Data");
                builder.setMessage(buffer.toString());
                builder.show();
              }
            }
    });
} }
```