SQLite

Introduction

- Android uses the SQLite database engine, a self-contained, transactional database engine.

www.sqlite.org

SQLite Implementation

- The SQLite database uses a simple plain file. All data structures making up the relational database is stored within that file.

The `SQLiteOpenHelper` Class

- Working with SQLite DB we should first define a new class that extends the `SQLiteOpenHelper` abstract class.
- This abstract helper class assists with managing the database creation and managing its versions.

The `SQLiteOpenHelper` Class

- We should define a new subclass that implements

  `onCreate(SQLiteDatabase)`

  `onUpgrade(SQLiteDatabase, int, int)`

  and optionally

  `onOpen(SQLiteDatabase).`

- This class takes care of opening the database if it exists, creating it if it does not, and upgrading it as necessary.

The `onCreate` Method

- This method is called when the database is created for the first time.
- Within this method we will place our code for creating the required tables.

The `onUpgrade` Method

- This method is called when the database needs to be upgraded.
- We should use this method for dropping tables we no longer need, add new ones and complete any required task following the new schema version.

The `onOpen` Method

- This method is called when the database is opened for our work.

The `getWriteableDatabase()` Method

- Calling the `getWriteableDatabase()` method on the object instantiated from the class we defined as one that extends the `SQLiteOpenHelper` class we will get a reference for a `SQLiteDatabase` object.
- We will be able to use the `SQLiteDatabase` object we get both for reading and for writing.

```
...
DatabaseHelper helper = new DatabaseHelper(this);
SQLiteDatabase database = helper.getWritableDatabase();
...
```

The `getReadableDatabase()` Method

- Calling the `getReadableDatabase()` method on the object instantiated from the class we defined as one that extends the `SQLiteOpenHelper` class we will get a reference for a `SQLiteDatabase` object.
- We will be able to use the `SQLiteDatabase` object we get for reading only.

```
...
DatabaseHelper helper = new DatabaseHelper(this);
SQLiteDatabase database = helper.getReadableDatabase();
...
```

The `SQLiteDatabase` Class

- An object instantiated from this class represents a specific SQLite database on our handset.
- The various methods this class defines include the methods `delete()`, `insert()`, `execSQL()` and others.
- Once we finish working with our `SQLiteDatabase` object we should call the method `close()`.

The `execSQL()` Method

- Calling this method we can pass over any SQL statement we want to execute which doesn't return a result.

```
...
public void onCreate(SQLiteDatabase database)
{
   database.execSQL("CREATE TABLE " + TABLE_NAME
         + " (_id INTEGER PRIMARY KEY AUTOINCREMENT, "
         + COUNTRY_NAME+" TEXT, " + COUNTRY_POPULATION
         + " REAL);");
}
...
```

The `insert()` Method

❖ Calling this method we can add new values to specific table.

```
...
ContentValues values = new ContentValues();
values.put(COUNTRY_NAME, "Israel");
values.put(COUNTRY_POPULATION, 7509300);
database.insert(TABLE_NAME, COUNTRY_NAME, values);
values.put(COUNTRY_NAME, "India");
values.put(COUNTRY_POPULATION, 1178816000);
database.insert(TABLE_NAME, COUNTRY_NAME, values);
...
```

The `delete()` Method

- Calling this method we can delete a specific row in a specific table in our database.

```
...
String[] vec =   { String.valueOf(rowId) };
database.delete(DatabaseHelper.TABLE_NAME, "_ID=?", vec);
...
```

The `rawQuery()` Method

- Calling this method we get a `Cursor` object through which it is possible to iterate all values.

```
...
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    database = (new DatabaseHelper(this)).
        getReadableWritableDatabase();
    countriesCursor = database.rawQuery(
        "SELECT _ID, name, population FROM countries ORDER BY name",
        null);
    ...
}
...
```

The `BaseColumns._ID` Column

- When using the `CursorAdapter` or one of its subclasses (such as the `SimpleCursorAdapter` the result set of our query *must* contain an integer column that its name is `BaseColumns._ID`. That column must have a unique value for each and every row.

Code Sample

```
public class DatabaseHelper extends SQLiteOpenHelper
{
    private static final String DATABASE_NAME = "countries_db";
    public static final String COUNTRY_NAME = "name";
    public static final String COUNTRY_POPULATION = "population";
    public static final String TABLE_NAME = "countries";

    public DatabaseHelper(Context context)
    {
        super(context, DATABASE_NAME, null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase database)
    {
        database.execSQL("CREATE TABLE "+ TABLE_NAME
            +" ("+BaseColumns._ID+" INTEGER PRIMARY KEY AUTOINCREMENT, "
            +COUNTRY_NAME+" TEXT, "+COUNTRY_POPULATION+" REAL);");
        ContentValues values = new ContentValues();
        values.put(COUNTRY_NAME, "Israel");
        values.put(COUNTRY_POPULATION, 7509300);
        database.insert(TABLE_NAME, COUNTRY_NAME, values);
```

Code Sample

```
        values.put(COUNTRY_NAME, "India");
        values.put(COUNTRY_POPULATION, 1178816000);
        database.insert(TABLE_NAME, COUNTRY_NAME, values);
        values.put(COUNTRY_NAME, "Japan");
        values.put(COUNTRY_POPULATION, 127430000);
        database.insert(TABLE_NAME, COUNTRY_NAME, values);
        values.put(COUNTRY_NAME, "Spain");
        values.put(COUNTRY_POPULATION, 46087170);
        database.insert(TABLE_NAME, COUNTRY_NAME, values);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
    {
        db.execSQL("DROP TABLE IF EXISTS "+TABLE_NAME);
        onCreate(db);
    }
}
```

Code Sample

```
public class CountriesDataActivity extends ListActivity
{
    private static final int ADD = 1;
    private static final int DELETE = 2;
    private static final int EXIT = 3;
    private SQLiteDatabase database = null;
    private Cursor countriesCursor = null;

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        database = (new DatabaseHelper(this)).getWritableDatabase();
        countriesCursor = database.rawQuery(
                "SELECT _ID, name, population FROM countries ORDER BY name",
                null);
        ListAdapter adapter = new SimpleCursorAdapter(
                this, R.layout.row, countriesCursor,
                new String[]
                {
                    DatabaseHelper.COUNTRY_NAME,
                    DatabaseHelper.COUNTRY_POPULATION
                },
                new int[] { R.id.country_name, R.id.country_population }
        );
```

Code Sample

```
    setListAdapter(adapter);
    registerForContextMenu(getListView());
}

@Override
public void onDestroy()
{
    super.onDestroy();

    countriesCursor.close();
    database.close();
}

@Override
public boolean onCreateOptionsMenu(Menu menu)
{
    menu.add(Menu.NONE, ADD, Menu.NONE, "Add Country");
    menu.add(Menu.NONE, EXIT, Menu.NONE, "Exit Application");
    return (super.onCreateOptionsMenu(menu));
}
```

Code Sample

```
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    switch (item.getItemId())
    {
        case ADD:
            add();
            return (true);

        case EXIT:
            finish();
            return (true);
    }
    return (super.onOptionsItemSelected(item));
}

@Override
public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenu.ContextMenuInfo menuInfo)
{
    menu.add(Menu.NONE, DELETE, Menu.NONE, "Delete Country");
}
```

Code Sample

```
@Override
public boolean onContextItemSelected(MenuItem item)
{
    switch (item.getItemId())
    {
        case DELETE:
            AdapterView.AdapterContextMenuInfo info;
            info = (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();
            delete(info.id);
            return (true);
    }
    return (super.onOptionsItemSelected(item));
}
```

Code Sample

```
private void add()
{
    LayoutInflater inflater = LayoutInflater.from(this);
    View addView = inflater.inflate(R.layout.add, null);
    final DialogWrapper wrapper = new DialogWrapper(addView);
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle(R.string.add_title);
    builder.setView(addView);
    builder.setPositiveButton(R.string.ok,
            new DialogInterface.OnClickListener()
            {
                public void onClick(DialogInterface dialog, int whichButton)
                {
                    addCountry(wrapper);
                }
            });
    builder.setNegativeButton(R.string.cancel,
            new DialogInterface.OnClickListener()
            {
                public void onClick(DialogInterface dialog, int whichButton)
                {
                    // do nothing
                }
            });
    builder.show();
}
```

Code Sample

```
private void delete(final long rowId)
{
    if (rowId > 0)
    {
        new AlertDialog.Builder(this).setTitle(R.string.delete_title)
                .setPositiveButton(R.string.ok,
                        new DialogInterface.OnClickListener()
                        {
                            public void onClick(DialogInterface dialog,
                                    int whichButton)
                            {
                                deleteCountry(rowId);
                            }
                        }).setNegativeButton(R.string.cancel,
                        new DialogInterface.OnClickListener()
                        {
                            public void onClick(DialogInterface dialog,
                                    int whichButton)
                            {
                                // do nothing
                            }
                        }).show();
    }
}
```

Code Sample

```
private void addCountry(DialogWrapper wrapper)
{
    ContentValues values = new ContentValues(2);
    values.put(DatabaseHelper.COUNTRY_NAME,
        wrapper.getCountryName());
    values.put(DatabaseHelper.COUNTRY_POPULATION,
        wrapper.getCountryPopulation());
    database.insert(DatabaseHelper.TABLE_NAME,
        DatabaseHelper.COUNTRY_NAME, values);
    countriesCursor.requery();
}

private void deleteCountry(long rowId)
{
    String[] vec =  { String.valueOf(rowId) };
    database.delete(DatabaseHelper.TABLE_NAME, "_ID=?", vec);
    countriesCursor.requery();
}
```

Code Sample

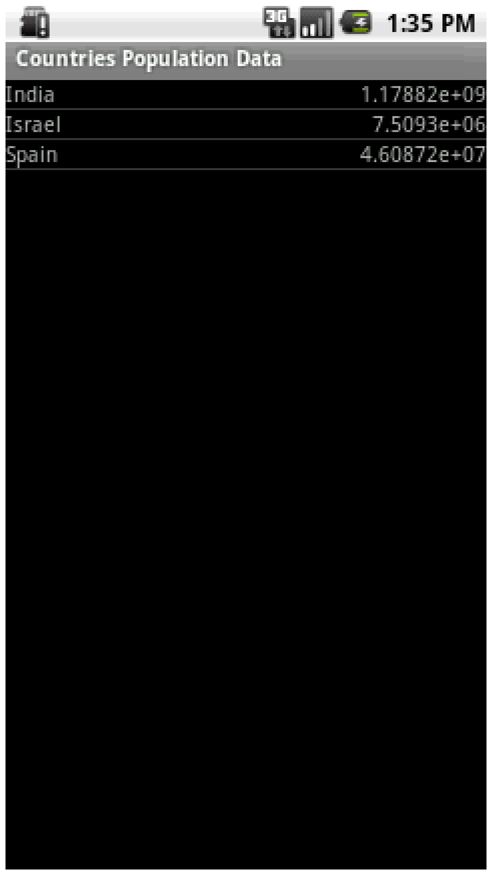
```
    class DialogWrapper
    {
        View base = null;

        DialogWrapper(View base)
        {
            this.base = base;
        }

        String getCountryName()
        {
            EditText etcn = (EditText)base.findViewById(R.id.country_name_val);
            return (etcn.getText().toString());
        }

        int getCountryPopulation()
        {
            EditText etcp =
                (EditText)base.findViewById(R.id.country_population_val);
            return new Integer(etcp.getText().toString());
        }
    }
}
```

Code Sample



Countries Population Data

```
India                1.17882e+09
Israel                7.5093e+06
Spain                4.60872e+07
```

The `query()` Method

- Calling this method we can pass over discrete parts of a query statement.

```
public Cursor query(boolean distinct,
                    String  table,
                    String[] columns,
                    String selection,
                    String[] selectionArgs,
                    String groupBy,
                    String having,
                    String orderBy,
                    String limit)
```

Code Sample

```
public class CountriesDataActivity extends ListActivity
{
    ...

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        database = (new DatabaseHelper(this)).getWritableDatabase();
        countriesCursor = database.query(
                    DatabaseHelper.TABLE_NAME,
                    new String[] {"_ID","name","population"},
                    null,
                    null,
                    null,
                    null,
                    DatabaseHelper.COUNTRY_NAME);
        ...
    }
    ...
}
```