# SQLite mit Android

SQLite ist eine Programmbibliothek, die ein relationales Datenbanksystem enthält.

Um SQLite zu verwenden benötigt man die Android Developer Tools (ADT) oder wie in meinem Falle die Entwicklungsumgebung Eclipse und das ADT-Plugin. Des Weiteren benötigt man ein mobiles Gerät (z.B. Handy/Tablet) oder einen Android Emulator um das Programm auszuführen.

In diesem Tutorial wird beispielhaft eine kleine App erstellt, die eine Datenbank erstellt, welche den Autoren und den Titel eines Buches enthält.

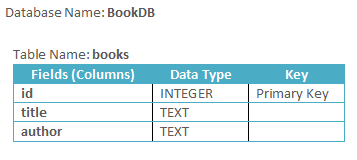
**( 1 ) Neue Android Application erstellen**

* **File** >> **New** >> **Android Application**
* App-Namen eingeben: **SQLite App**
* Projekt-Namen eingeben: **android-sqlite**
* Package: **com.example.android\_sqlite**
* Default-Einstellungen bestätigen, **Next** klicken bis man **Finish** erreicht.

**( 2 ) Daten Model Design “Tabellenstruktur”**

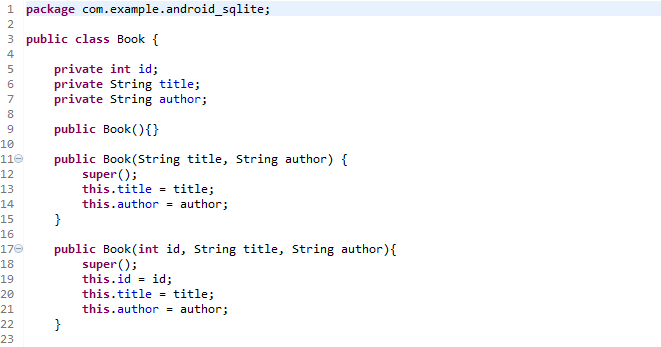
Folgendes soll erstellt werden:

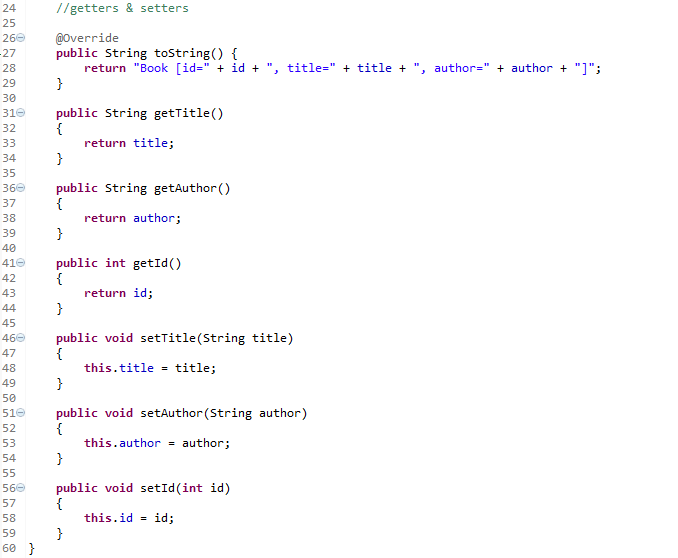
* Datenbank Instanz**:** “**BookDB**“.
* Tabelle: “**books**” mit 3 Spalten **id**, **title** & **author**

[](http://hmkcode.com/wp-content/uploads/2013/09/android-sqlite-table-books.png)

**( 3 ) Objekt Model “Book.java”**

* Erstellen einer Java Bean Klasse: **Book.java**





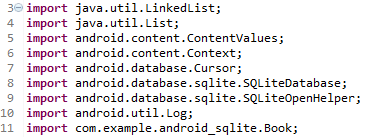
**( 4 ) extends SQLiteOpenHelper**

Die empfohlene Methode eine neue SQLite-Datenbank zu erstellen, ist eine Subklasse von **SQLiteOpenHelper** zu erstellen und die **onCreate()** Methode zu überschrieben. In dieser kann man ein SQLite-Kommando ausführen um Tabellen in der Datenbank zu erstellen.

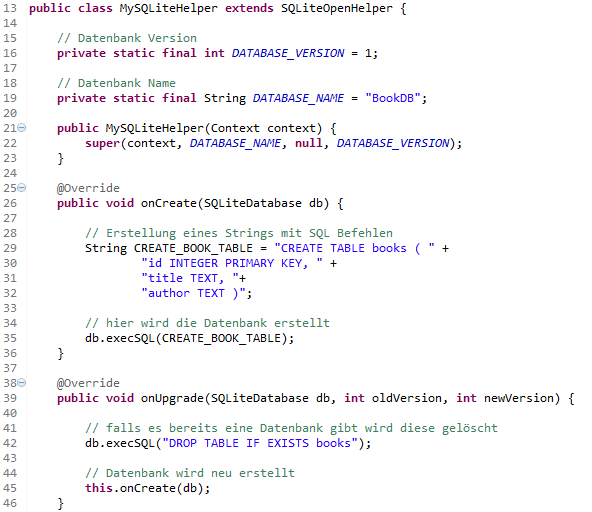
Folgende Schritte sind notwendig:

* Erstellen einer neuen Klasse **MySQLiteHelper**die von **SQLiteOpenHelper** erbt.
* **MySQLiteHelper Konstruktor**muss den **super** Klassen Konstruktor aufrufen.
* Überschreiben der **onCreate**() Methode um die Tabelle(n) zu erstellen.
* Überschreiben der **onUpgrade**() Methode alte Tabellen zu löschen und neue zu erstellen.

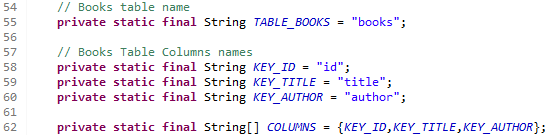
Zunächst einmal müssen diverse Klasssen importiert und eingebunden werden:



Dann wird die Datenbank definiert (Name, Version) und die beiden Methoden überschrieben:

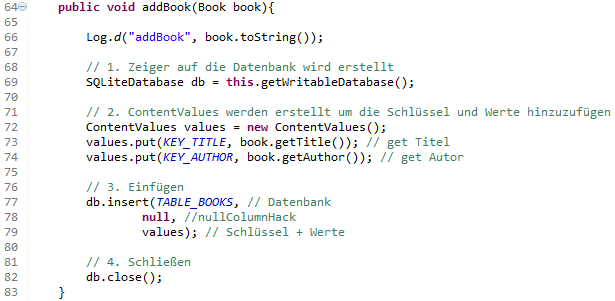


Dazu noch einige weitere Definitionen bezüglich des Aufbaus der Datenbank:

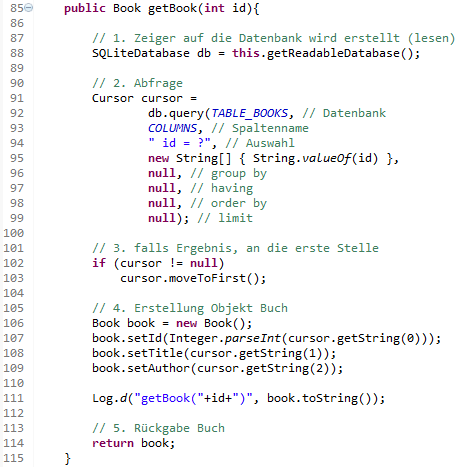


Nun können die gewünschten Datenbank-Operationen implementiert werden. Hierbei werden die berühmten **CRUD**-Operationen benutzt (**C**reate "add", **R**ead "get", **U**pdate, **D**elete) sowie eine Methode um sich alle Bücher ausgeben zu lassen.

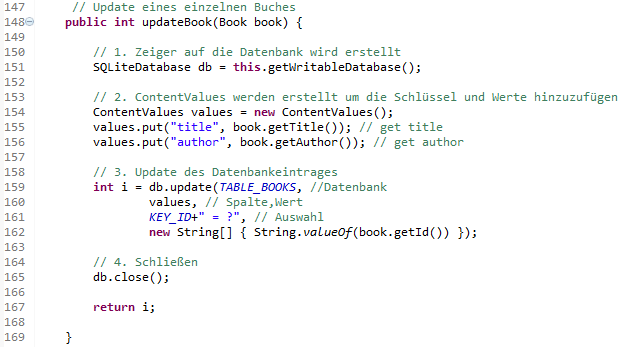
Methode **addBook** zum Hinzufügen eines Buches:



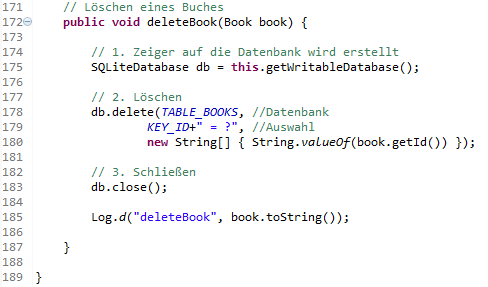
Methode **getBook** zum Anzeigen eines Buches:



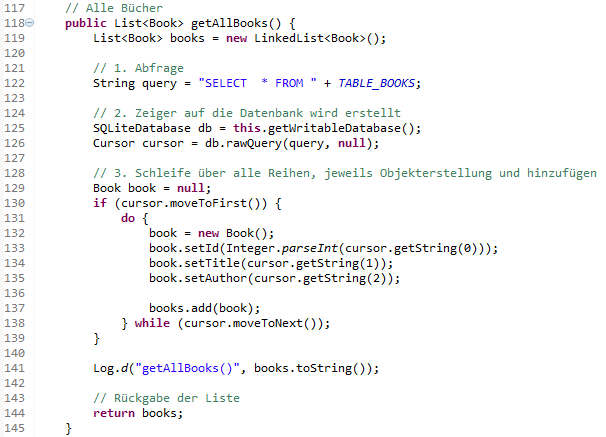
Methode **updateBook** zum Aktualisieren eines Buches:



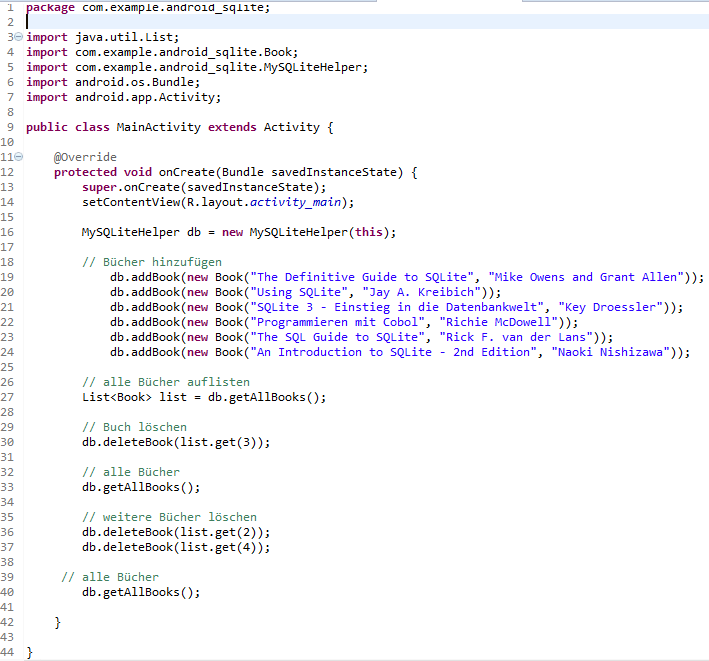
Methode **deleteBook** zum Löschen eines Buches:



Methode **getAllBooks** zum Anzeigen aller Bücher:



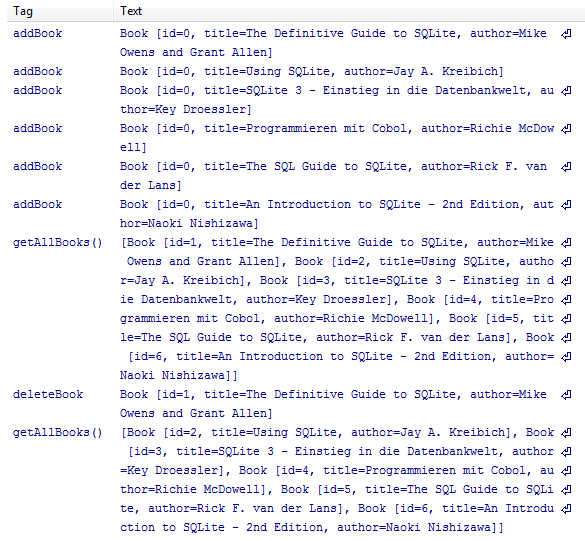
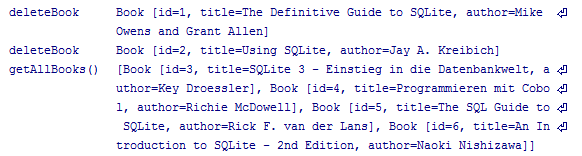
**( 5 ) Benutzen der MySQLiteHelper in der Klasse MainActivity**



**In der neuen Klasse MainActivity werden die beiden anderen Klassen Book und MySQLiteHelper eingebunden und die gewünschten Operationen implementiert. Hier zum Beispiel werden zunächst 6 Bücher zur Datenbank hinzugefügt. Dann wird die komplette Datenbank einmal ausgegeben. Danach wird das Buch mit dem Index 3 (also das vierte) gelöscht und wieder die komplette Datenbank ausgegeben. Es werden nochmal zwei Bücher gelöscht und schließlich noch einmal die Datenbank, die nun noch 3 Bücher enthält, ausgegeben.**

**Die Ergebnisse der Datenbank-Operationen kann man im View LogCat nachverfolgen.**

**• Window >> Show View >> Other >> Android >> LogCat**

An diesem einfachen Beispiel kann man gut erkennen, dass die Implementierung einer Datenbank bei Android kein Hexenwerk ist und man durch wenige Methoden zahlreiche Möglichkeiten hat mit einer Datenbank zu interagieren.

Ich hoffe euch hat mein Tutorial gefallen und ihr probiert es selbst einmal aus.