Anmerkung: Hier werden zunächst die Befehle mit ihren häufigsten Varianten vorgestellt. Die vollständigen Befehle bzw. die Befehle mit allen Möglichkeiten finden Sie am Ende des Dokumentes in Form von Railroad-Diagrammen.

**Tabellen erstellen**

# CREATE TABLE

CREATE TABLE tabellenname (

spaltenname\_1 <DATENTYP> <Einschränkung>,

spaltenname\_2 <DATENTYP> <Einschränkung>,

… <DATENTYP> <Einschränkung>,

<WEITERE EINSCHRTÄNKUNG>,

……

)

Erstellt eine neue Tabelle mit den angegebenen Spalten. Jede Spalte benötigt einen Datentyp, der angibt, welche Daten in dieser Spalte eingetragen werden können. Neben dem Datentyp können noch Einschränkungen hinzugefügt werden. Jede Tabelle muss eine Spalte "id" mit dem Datentyp INTEGER und der Einschränkung PRIMARY KEY haben, damit jeder Zeile eine eindeutige Identifikationsnummer zugewiesen werden kann, die automatisch hochgezählt wird.

#### Datentypen in SQLite

SQLite verfügt nur über eine eingeschränkte Anzahl an Datentypen.

|  |  |
| --- | --- |
| **Datentyp** | **Erläuterung** |
| INTEGER | Ganzzahl |
| REAL | Gleitkommazahl |
| TEXT | Text |
| BLOB | Hält Daten genau so, wie diese eingegeben wurden. Ideal z.B. für Dateien oder Bilder. |

In SQLite gibt es keinen Datentyp für Datumswerte. Diese speichern wir als Text nach ISO8601. Die Syntax hierzu ist YYYY-MM-DD, z.B. 2009-11-13.

#### EINSCHRÄNKUNG

| **Einschränkung** | **Erklärung** |
| --- | --- |
| UNIQUE | Jeder Wert dieser Spalte darf nur einmal vorkommen. |
| PRIMARY KEY | Definiert die Werte einer Spalte als Primärschlüssel, anhand derer jede Zeile eindeutig identifizierbar wird. Sie sind automatisch UNIQUE. |
| FOREIGN KEY | Definiert die Werte einer Spalte als Fremdschlüssel. |
| NOT NULL | Legt fest, dass die Werte einer Spalte nicht leer sein dürfen. |
| DEFAULT <wert> | Legt einen Standartwert für die Spalte fest, wenn kein Wert eingetragen wird. |

### Beispiel 1:

CREATE TABLE ´lehrer´ (

´id´ INTEGER PRIMARY KEY,

´vorname´ TEXT,

´nachname´ TEXT

)

Mit diesem Befehl wird eine Tabelle mit dem Namen *lehrer* angelegt, die zwei Spalten als TEXT sowie eine Spalte id mit dem Datentyp INTEGER und der Einschränkung PRIMARY KEY hat.

#### Ergebnis:

Tabelle: *lehrer*

|  |  |  |
| --- | --- | --- |
| id | vorname | Nachname |
|  |  |  |

Die neu erstellte Tabelle hat noch keine Einträge, diese können mit dem INSERT INTO-Befehl eingefügt werden.

### Beispiel 2:

CREATE TABLE ´klassen´ (

´id´ INTEGER PRIMARY KEY,

´name´ TEXT,

´jahrgangsstufe´ INTEGER,

´lehrer\_ID´ INTEGER

FOREIGN KEY (lehrer\_id) REFERENCES lehrer(id)

)

Mit diesem Befehl wird eine Tabelle mit dem Namen *klassen* angelegt, die mehrere Spalten als TEXT und als INTEGER hat sowie eine Spalte id, mit der Einschränkung PRIMARY KEY. Am Ende des Befehls wird eine Referenz auf eine zweite Tabelle *klassen*, auf die Spalte *id* gesetzt.

Dieser Verweis verbindet eindeutig Einträge, die über mehrere Tabellen hinweg verteilt sind. Beispiel: eine Klasse ist genau einem bestimmten Lehrer zugeordnet.

#### Ergebnis:

Tabelle 2: *klassen*

|  |  |  |  |
| --- | --- | --- | --- |
| id | Name | jahrgangsstufe | Lehrer\_id |
|  |  |  |  |

**Daten in Tabellen einfügen**

# INSERT (INTO)

INSERT INTO tabellenname (spaltenname\_1, spaltenname\_2,…., spaltenname\_n)

VALUES (<wert\_1>, <wert\_2>, ….<wert\_n> )

Mit dem INSERT INTO Befehl wird eine neue Zeile einer Tabelle hinzugefügt. Neben dem Tabellennamen müssen alle zu befüllenden Spaltennamen der Tabelle in Klammern dahinter eingetragen werden. Nur die Spalte "id", die automatisch hochgezählt wird und einzigartig sein muss, wird nicht mit aufgeführt. Bei VALUES müssen die Werte in der richtigen Reihenfolge, passend der Reihenfolge der Spaltennamen, eingetragen werden.

**Beispiel**

Tabelle1: schueler

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | klasse\_id | fehltage |
| 1 | Paul | Weber | 2000-07-12 | FI101 | 25 |
| 2 | Marie | Müller | 2003-11-19 | FI102 | 5 |
| 3 | Mahmud | Al’hisse | 2003-05-30 | FI102 | 2 |
| 4 | Sabrina | Otto | 2004-05-19 | FI102 | 6 |
| 5 | Ayse | Ünül | 2004-12-12 | FI103 | 0 |

**Befehl:**

INSERT INTO schueler (vorname, nachname, geb\_Datum, klasse\_id)

VALUES (´Ullrich´, ´Gscheuder´, ´2006-07-05´, ´FI103´)

**Ergebnis:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | klasse\_id | fehltage |
| 1 | Paul | Weber | 2000-07-12 | FI101 | 25 |
| 2 | Marie | Müller | 2003-11-19 | FI102 | 5 |
| 3 | Mahmud | Al’hisse | 2003-05-30 | FI102 | 2 |
| 4 | Sabrina | Otto | 2004-05-19 | FI102 | 6 |
| 5 | Ayse | Ünül | 2004-12-12 | FI103 | 0 |
| 6 | Ullrich | Gcheuder | 2006-07-05 | FI103 |  |

**Inhalte in Tabellen löschen**

**DELETE**

Mit dem DELETE-Befehl werden alle Einträge einer Tabelle gelöscht.

DELETE FROM tabellenname

**Inhalte ändern**

**UPDATE**

Mit dem UPDATE-Befehl können alle oder einzelne Einträge einer Tabelle verändert werden.

UPDATE tabellenname SET spaltenname\_1 = <neuerWert>

Oder

UPDATE tabellenname SET spaltenname\_1 = <neuerWert>

WHERE spaltenname <Operator> <Wert>

**Beispiel**

Tabelle: schueler

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | klasse\_id | fehltage |
| 1 | Paul | Weber | 2000-07-12 | FI101 | 25 |
| 2 | Marie | Müller | 2003-11-19 | FI102 | 5 |
| 3 | Mahmud | Al’hisse | 2003-05-30 | FI102 | 2 |
| 4 | Sabrina | Otto | 2004-05-19 | FI102 | 6 |
| 5 | Ayse | Ünül | 2004-12-12 | FI103 | 0 |

**Befehl:**

UPDATE schueler SET fehltage = 0

Mit diesem Befehl werden die Fehltage aller Schüler auf 0 zurückgesetzt. Dies könnte am Ende eines Schuljahres sinnvoll sein.

**Ergebnis:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | klasse\_id | fehltage |
| 1 | Paul | Weber | 2000-07-12 | FI101 | 0 |
| 2 | Marie | Müller | 2003-11-19 | FI102 | 0 |
| 3 | Mahmud | Al’hisse | 2003-05-30 | FI102 | 0 |
| 4 | Sabrina | Otto | 2004-05-19 | FI102 | 0 |
| 5 | Ayse | Ünül | 2004-12-12 | FI103 | 0 |

Oder, wenn der Wert eines bestimmten Datensatzes verändert werden soll

UPDATE schueler SET klasse\_id = FI103

WHERE klasse\_id = 2

Mit diesem Befehl werden alle Schüler\*innen die, die klasse\_id = 2 haben in die 3. Klasse versetzt (klasse\_id = 3).

Ergebnis:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | klasse\_id | fehltage |
| 1 | Paul | Weber | 2000-07-12 | 1 | 0 |
| 2 | Marie | Müller | 2003-11-19 | 3 | 0 |
| 3 | Mahmud | Al’hisse | 2003-05-30 | 3 | 0 |
| 4 | Sabrina | Otto | 2004-05-19 | 3 | 0 |
| 5 | Ayse | Ünül | 2004-12-12 | 3 | 0 |

**Tabellenstruktur ändern**

ALTER TABLE

Den ALTER Befehl gibt es in etlichen Varianten. Ich nenne hier erst einmal die am häufigsten gebrauchten.

Mit dem nachfolgenden Befehl wird der Tabelle eine neue Spalte hinzugefügt.

ALTER TABLE tabelle ADD COLUMN ´spalte´ DATENTYP

Beispiel:

Tabelle Schüler:

|  |  |  |  |
| --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum |
|  |  |  |  |

ALTER TABLE schueler ADD COLUMN ´geburtsort´ TEXT

Ergebnis:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | Geburtsort |
|  |  |  |  |  |

Mit dem nachfolgenden Befehl wird eine **Spalte gelöscht**.

ALTER TABLE tabelle DROP COLUMN spalte

Beispiel:

Tabelle Schüler:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | vorname | nachname | geb\_Datum | Geburtsort |
|  |  |  |  |  |

**Befehl:**

ALTER TABLE schueler DROP COLUMN geb\_Datum

**Ergebnis:**

|  |  |  |  |
| --- | --- | --- | --- |
| id | vorname | nachname | Geburtsort |
|  |  |  |  |

**Tabellen/ Tabellenspalte umbenennen**

Der nachfolgende Befehlt benennt eine Tabelle um

ALTER TABLE tabelle RENAME TO ´tabellenname\_neu´

Der nachfolgende Befehlt benennt eine Spalte in der Tabelle um

ALTER TABLE tabelle RENAME spaltenname TO ´tabellenname\_neu´

**Beispiel 1:**

Tabelle: lehrer

**Befehl:**

ALTER TABLE lehrer RENAME TO ´lehrkraefte´

Ergebnis: Dieser Befehl nennt die Tabellen „lehrer“ in „lehrkraefte“ um

**Beispiel 2:**

Tabelle: schueler

ALTER TABLE schueler RENAME vorname TO ´rufname´

Dieser Befehl benennt die Spalte *vorname* in der Tabelle *schueler* in *rufname* um.

**Ergebnis:**

Tabelle: schueler

|  |  |  |  |
| --- | --- | --- | --- |
| id | rufname | nachname | Geburtsort |
|  |  |  |  |

Quelle: verändert nach: <https://sqlverine.org/docs/SQLite%20Befehle/intro>

Ein Bild, das Reihe, Diagramm, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Ein Bild, das Diagramm, Text, Screenshot, Muster enthält.

Automatisch generierte Beschreibung

Quelle: https://www.sqlite.org/syntaxdiagrams.html