Spickzettel SQL · MariaDB

Grundlagen · Datenbank · Tabellen · Daten ändern · Hilfe

Holger Sebastião · www.sebastiao.org
Version 1.0 · 2026-10-25

Begriffe — relationale Datenbank

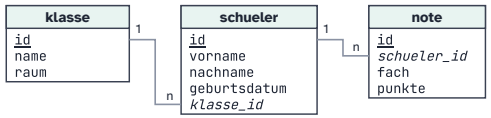
Tabelle schueler

id	vorname	nachname	geburtsdatum	klasse_id
1	Ali	Yilmaz	2008-03-14	1
2	Lea	Schmitt	2007-11-02	2
3	Tom	Becker	2008-06-30	1

Primär-schlüssel — *gelb: eine Zeile = ein Datensatz, jede Spalte = ein Attribut (Feld)* — *Fremd-schlüssel → klasse.id*

Primärschlüssel	macht jede Zeile eindeutig, nie doppelt, nie leer
Fremdschlüssel	verweist auf den Primärschlüssel einer anderen Tabelle
DBMS	das Programm, das die Datenbank verwaltet: hier MariaDB

Unsere Beispiel-Datenbank — schule



unterstrichen = Primärschlüssel, *kursiv* = Fremdschlüssel. Eine Klasse hat viele Schüler, ein Schüler hat viele Noten.

Schreibregeln

- Jeder Befehl endet mit `;`
- Befehlswörter groß (`SELECT`), Namen klein: so liest man es leichter.
- Text und Datum in einfachen Hochkommas: `'Ali'`, `'2008-03-14'`
- Kommentar: `-- bis Zeilenende` (Leerzeichen nach `--`)
- Namen ohne Umlaute und Leerzeichen: `schueler`, `klasse_id`

Wo schreibe ich SQL?

- **phpMyAdmin** (z. B. mit XAMPP): Datenbank wählen → Reiter **SQL** → Befehl eingeben → OK.
- **Kommandozeile:** anmelden mit `mariadb -u root -p` (bei XAMPP: `mysql -u root -p`), beenden mit `exit`.

Datenbank anlegen

```
CREATE DATABASE schule
  CHARACTER SET utf8mb4
  COLLATE utf8mb4_german2_ci;
USE schule;   -- ab jetzt in schule arbeiten
```

`SHOW DATABASES;`	alle Datenbanken
`SHOW TABLES;`	alle Tabellen in der Datenbank
`DESCRIBE schueler;`	Spalten und Datentypen
`DROP DATABASE schule;`	löscht **alles**, ohne Nachfrage

`utf8mb4` speichert alle Zeichen (ä, ß, Emojis). `german2_ci` sortiert deutsch, `ci`: Groß/klein egal.

Datentypen

TYP	FÜR · BEISPIEL
`INT`	ganze Zahl: `42`
`DECIMAL(6,2)`	Geld, 6 Stellen, 2 nach dem Komma: `1234.50`
`DOUBLE`	Kommazahl (Messwert): `3.14`
`VARCHAR(50)`	Text bis 50 Zeichen: `'Ali'`
`CHAR(5)`	Text, immer genau 5: `'63450'`
`TEXT`	langer Text
`DATE`	Datum: `'2026-10-25'`
`TIME`	Uhrzeit: `'07:45:00'`
`DATETIME`	Datum und Uhrzeit: `'2026-10-25 07:45:00'`
`BOOLEAN`	wahr/falsch, gespeichert als 1/0: `TRUE`

Kommazahlen mit **Punkt**: `3.5`. Datum immer `'JJJJ-MM-TT'`, nicht `'25.10.2026'`.

Postleitzahl und Telefon sind Text, keine Zahl: sonst fällt die führende 0 weg.

NULL heißt: kein Wert

`NULL` ist **nicht** `0` und nicht `''` (leerer Text), sondern „unbekannt“. Felder ohne `NOT NULL` dürfen leer bleiben. Prüfen mit `IS NULL` (Seite 2).

Tabelle anlegen — CREATE TABLE

```
CREATE TABLE klasse (
  id INT PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(10) NOT NULL UNIQUE,
  raum VARCHAR(10)
);

CREATE TABLE schueler (
  id INT PRIMARY KEY AUTO_INCREMENT,
  vorname VARCHAR(50) NOT NULL,
  nachname VARCHAR(50) NOT NULL,
  geburtsdatum DATE,
  klasse_id INT,
  FOREIGN KEY (klasse_id)
    REFERENCES klasse(id)
);
```

`AUTO_INCREMENT`	MariaDB zählt die id selbst hoch: 1, 2, 3 …
`NOT NULL`	Pflichtfeld, darf nicht leer sein
`UNIQUE`	kein Wert doppelt
`DEFAULT 0`	Vorgabe, wenn nichts angegeben
`FOREIGN KEY`	Fremdschlüssel: nur ids, die es in `klasse` gibt

Reihenfolge: erst `klasse`, dann `schueler` anlegen.

Tabelle ändern und löschen

```
-- neue Spalte
ALTER TABLE schueler ADD email VARCHAR(100);
-- Datentyp ändern
ALTER TABLE schueler MODIFY email TEXT;
-- Spalte löschen
ALTER TABLE schueler DROP COLUMN email;
-- Tabelle samt Daten löschen
DROP TABLE note;
```

Beziehungen umsetzen

1:n Fremdschlüssel auf die n-Seite. **n:m** Zwischentabelle:

```
CREATE TABLE teilnahme (
  schueler_id INT,
  ag_id INT,
  PRIMARY KEY (schueler_id, ag_id),
  FOREIGN KEY (schueler_id)
    REFERENCES schueler(id),
  FOREIGN KEY (ag_id) REFERENCES ag(id)
);
```

Daten einfügen — INSERT

```
INSERT INTO klasse (name, raum)
VALUES ('11IT1', 'B204'),
       ('12IT1', 'B206');

INSERT INTO schueler (vorname, nachname,
  geburtsdatum, klasse_id)
VALUES ('Ali', 'Yilmaz', '2008-03-14', 1);
```

Die `id` weglassen: `AUTO_INCREMENT` vergibt sie. Spalten und Werte in **gleicher Reihenfolge**.

Daten ändern — UPDATE

```
UPDATE schueler
SET klasse_id = 2, nachname = 'Kaya'
WHERE id = 1;
```

Daten löschen — DELETE

```
DELETE FROM schueler
WHERE id = 3;
```

Ohne `WHERE` ändert `UPDATE` bzw. löscht `DELETE` **alle** Zeilen. Es gibt kein Rückgängig. Tipp: erst mit `SELECT … WHERE …` prüfen, welche Zeilen getroffen werden.

`DELETE FROM t;`	alle Zeilen weg, Tabelle bleibt
`TRUNCATE t;`	wie oben, id fängt wieder bei 1 an
`DROP TABLE t;`	Tabelle samt Aufbau weg

Typische Fehlermeldungen

`1064`	Tippfehler im Befehl: Komma, Klammer, `;` oder `'` prüfen. Die Meldung zeigt die Stelle **nach** dem Fehler.
`1452`	Fremdschlüssel zeigt auf eine id, die es nicht gibt
`1451`	Zeile wird noch von einer anderen Tabelle gebraucht: erst dort löschen
`1062`	Wert doppelt bei `PRIMARY KEY` oder `UNIQUE`
`1146`	Tabelle gibt es nicht: Name oder `USE` prüfen

Spickzettel SQL · MariaDB

Abfragen · Filtern · Auswerten · Verknüpfen · Hilfe

Holger Sebastião · www.sebastiao.org
Version 1.0 · 2026-10-25

Aufbau einer Abfrage — SELECT

```
SELECT   spalten          -- was?
FROM     tabelle          -- woher?
WHERE    bedingung        -- welche Zeilen?
GROUP BY spalte           -- Gruppen bilden
HAVING   bedingung        -- welche Gruppen?
ORDER BY spalte           -- sortieren
LIMIT    anzahl;          -- wie viele?
```

Nur `SELECT` und `FROM` sind Pflicht. Die **Reihenfolge** ist fest. Merkhilfe: **S**ie **F**ahren **W**ie **G**estörte **H**ühner **O**hne **L**icht.

Spalten auswählen

```
SELECT * FROM schueler;     -- alle Spalten

SELECT vorname, nachname
FROM schueler;

SELECT DISTINCT klasse_id   -- ohne Doppelte
FROM schueler;

SELECT nachname AS Name     -- Alias
FROM schueler;
```

Sortieren und begrenzen

```
SELECT vorname, nachname
FROM schueler
ORDER BY nachname ASC, vorname ASC
LIMIT 10;
```

`ASC` aufsteigend (A–Z, 1–9, Standard), `DESC` absteigend. Bei Gleichstand entscheidet die zweite Spalte. `LIMIT 10`: nur die ersten 10 Zeilen.

Nützliche Funktionen

`CONCAT(vorname, ' ', nachname)`	Text verbinden
`UPPER(name)` · `LOWER`	groß · klein
`LENGTH(name)`	Länge
`CURDATE()` · `NOW()`	heute · jetzt
`YEAR(geburtsdatum)`	nur das Jahr

```
SELECT vorname,
  TIMESTAMPDIFF(YEAR, geburtsdatum,
                CURDATE()) AS jahre
FROM schueler;     -- Alter in Jahren
```

Zeilen filtern — WHERE

```
SELECT vorname, nachname
FROM schueler
WHERE klasse_id = 1
  AND geburtsdatum >= '2008-01-01';
```

`= <>`	gleich, ungleich (auch `!=`)
`< > <= >=`	kleiner, größer …
`AND`	beide Bedingungen müssen stimmen
`OR`	mindestens eine muss stimmen
`NOT`	kehrt um
`BETWEEN 10 AND 15`	von 10 bis 15, beide Grenzen **mit**
`IN ('Mathe', 'Deutsch')`	einer aus der Liste
`IS NULL`	Feld ist leer
`IS NOT NULL`	Feld ist gefüllt

! `= NULL` findet **nie** etwas. Immer `IS NULL`.

AND vor OR: Bei gemischten Bedingungen Klammern setzen:

```
WHERE (fach = 'Mathe' OR fach = 'Deutsch')
  AND punkte < 5
```

Text suchen — LIKE

```
SELECT * FROM schueler
WHERE nachname LIKE 'Sch%';
```

`%`	beliebig viele Zeichen, auch keins
`_`	genau ein Zeichen
`'Sch%'`	fängt mit Sch an: Schmitt, Schulz
`'%mann'`	endet auf mann: Neumann
`'%e%'`	enthält irgendwo ein e
`'M_ier'`	Maier, Meier, Mayer nicht

Rechnen in der Abfrage

```
SELECT fach, punkte,
  ROUND((17 - punkte) / 3, 1) AS schulnote
FROM note;
```

Pro Zeile mit `+ - * /`. Die Tabelle bleibt gleich.

Zählen und rechnen

```
SELECT COUNT(*) AS anzahl
FROM schueler
WHERE klasse_id = 1;
```

`COUNT(*)`	Anzahl Zeilen
`COUNT(spalte)`	Anzahl gefüllter Felder (ohne NULL)
`SUM(punkte)`	Summe
`AVG(punkte)`	Durchschnitt
`MIN(punkte)`	kleinster Wert
`MAX(punkte)`	größter Wert
`ROUND(AVG(punkte), 1)`	auf 1 Stelle runden

Gruppen bilden — GROUP BY

```
SELECT fach, ROUND(AVG(punkte), 1) AS schnitt
FROM note
GROUP BY fach
HAVING AVG(punkte) >= 8
ORDER BY schnitt DESC;
```

FACH	SCHNITT
Informatik	11.4
Deutsch	9.2
Englisch	8.0

`GROUP BY fach`: eine Ergebniszeile **pro Fach**, die Funktion rechnet je Gruppe.

WHERE oder HAVING? `WHERE` filtert Zeilen **vor** dem Gruppieren, `HAVING` filtert Gruppen **danach**. Nur in `HAVING` dürfen `COUNT`, `AVG` … stehen.

! Jede Spalte im `SELECT` steht im `GROUP BY` oder in einer Funktion. MariaDB meldet sonst keinen Fehler, zeigt aber einen zufälligen Wert.

Unterabfrage

```
SELECT vorname, nachname
FROM schueler
WHERE id IN (SELECT schueler_id
             FROM note
             WHERE punkte = 15);
```

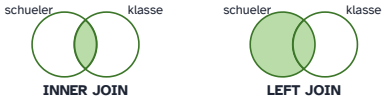
Die innere Abfrage läuft zuerst und liefert eine Liste von ids.

Tabellen verknüpfen — JOIN

```
SELECT schueler.vorname, klasse.name
FROM schueler
INNER JOIN klasse
  ON schueler.klasse_id = klasse.id;
```

VORNAME	NAME
Ali	11IT1
Lea	12IT1
Tom	11IT1

Nach `ON` steht immer: **Fremdschlüssel = Primärschlüssel**. `tabelle.spalte` sagt, aus welcher Tabelle die Spalte kommt.



INNER JOIN: nur Schüler, die eine Klasse haben.
LEFT JOIN: **alle** Schüler, ohne Klasse steht `NULL` bei `name`.

Drei Tabellen

```
SELECT klasse.name, schueler.nachname,
       note.fach, note.punkte
FROM note
INNER JOIN schueler
  ON note.schueler_id = schueler.id
INNER JOIN klasse
  ON schueler.klasse_id = klasse.id
WHERE klasse.name = '11IT1'
ORDER BY schueler.nachname;
```

Typische Fehler

- Komma vor `FROM`: `SELECT vorname, FROM`
- Text ohne Hochkommas: `WHERE fach = Mathe`
- `JOIN` ohne `ON`: jede Zeile mit jeder kombiniert.
- Spalte heißt in zwei Tabellen `id`: `tabelle.id` schreiben.
- Gesperrte Wörter als Namen, z. B. `alter`, `order`, `key`.