

Anforderungsanalyse

Welche Vorteile bietet eine Datenbank gegenüber Excel?

Eine Datenbank kann viel schneller und praktischer mit großen Datenmengen arbeiten. Außerdem sind für übliche Datenbanken mit vielen verschiedenen Systemen einbindbar. Excel dagegen nutzt ein proprietäres Dateiformat, welche von externer Software nicht einfach verarbeitet werden kann. Außerdem kann die Datenbank Zugriffsverwaltung, womit der Zugriff auf Benutzer oder Benutzergruppen beschränkt werden kann. Die Datenintegrität ist einfacher sichergestellt und es kommt nicht zu redundanten Daten. Es ist möglich auf eine Datenbank gleichzeitig von mehreren Nutzern zu verwenden.

Wie legt man eine neue Datenbank mit SQL an?

SQL stellt den Befehl `CREATE DATABASE 'NAME';` zur Verfügung.

Machen Sie ein Beispiel für das Anlegen einer Tabelle in SQL:

```
CREATE TABLE `Hunderassen` (  
    `id` INT NOT NULL AUTO_INCREMENT,  
    `name` TEXT NOT NULL,  
    PRIMARY KEY (`id`)  
);
```

Welche Aufgabe besitzt der Primärschlüssel?

Der Primärschlüssel identifiziert eine Zeile in einer Tabelle eindeutig. Er muss daher einmalig sein. Ein Primärschlüssel kann zudem aus mehreren Spalten zusammengesetzt werden.

Beispiel ist in der Aufgabe oben.

Wofür verwendet man ALTER TABLE in SQL?

`ALTER TABLE` wird verwendet um die Struktur und die Eigenschaften einer Tabelle zu bearbeiten.

Beispiel für die Anlegung eines Fremdschlüssels in SQL:

```
CREATE TABLE `hund` (  
    `id` INT NOT NULL AUTO_INCREMENT,  
    `rasse` INT NOT NULL,  
    PRIMARY KEY (`id`),  
    CONSTRAINT `FK__hunderassen` FOREIGN KEY (`rasse`) REFERENCES `hunderassen` (`id`) ON  
    UPDATE NO ACTION ON DELETE NO ACTION  
);
```

Erklären Sie, wie man einer Tabelle neue Datensätze hinzufügt.

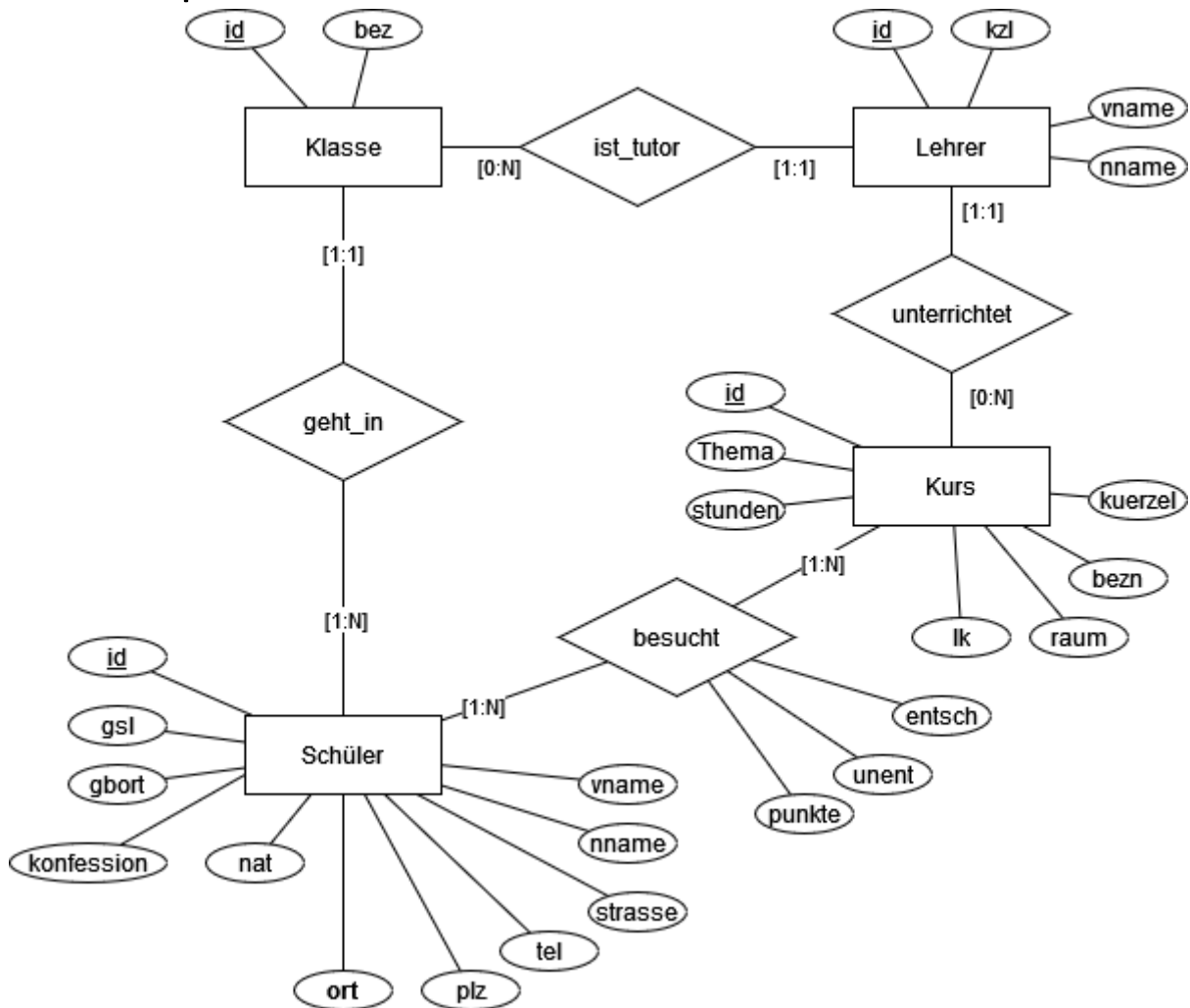
```
INSERT INTO `hunderassen` (`name`) VALUES ('Dogge');  
INSERT INTO `hunderassen` (`name`) VALUES ('Tamaskan');
```

Zeigen Sie wie bestehende Daten gelöscht oder Verändert werden können.

Bestehende Daten können mit dem SQL-Befehl **UPDATE** verändert und mit dem Befehl **DELETE** gelöscht werden.

```
UPDATE `hunderassen` SET `name` = 'Husky' WHERE `name` = 'Tamaskan';  
DELETE FROM `hunderassen` WHERE name = "Dogge";
```

Konzeptioneller Entwurf



ID:

Wir haben uns dafür entschieden, an alle Entitätstypen das Attribut ID als Primärschlüssel zu definieren, da hiermit die Benennung der Fremdschlüssel einfach und einheitlich ist.

Attribute an der Relation besucht:

Jeder Schüler hat für jeden seiner Kurse sowohl individuelle Punkte, als auch Fehlzeiten. Dies begründet außerdem die N-zu-M-Beziehung

Kardinalitäten:

Da ein Schüler in genau eine Klasse geht, eine Klasse jedoch aus mehreren Schülern besteht, wurde sich bei der geht_in Beziehung für eine 1-zu-N-Beziehung entschieden.

Jede Klasse hat einen Tutor zugewiesen, allerdings haben nicht alle Lehrer ein Tutorium, deshalb wird hier auf eine 0-zu-N-Beziehung gesetzt. Dies gilt auch für die unterrichtet-Beziehung.

Logischer Entwurf

1&2.NF

Klasse(id, bezeichnung, tutor_id#)

Schueler(id, nachname, vorname, geschlecht, geburtsort, konfession, nationalitaet, ort, plz, tel, strasse, klasse_id#)

Lehrer(id, kuerzel, nachname, vorname)

Kurs(id, thema, stunden, kuerzel, bezeichnung, raum, lkgk, lehrer_id#)

Schuler_besucht_kurs(schueler_id#, kurs_id#, punkte, unentschuldigt, entschuldigt)

3.NF

Klasse(id, bezeichnung, tutor_id#)

Schueler(id, nachname, vorname, geschlecht, geb_ort_id#, konfession, nationalitaet, ort_id#, tel, strasse, klasse_id#)

Lehrer(id, kuerzel, nachname, vorname)

Kurs(id, thema, stunden, bezeichnung_id#, raum, lkgk, lehrer_id#)

Schuler_besucht_kurs(schueler_id#, kurs_id#, punkte, unentschuldigt, entschuldigt)

Ort(id, name, plz)

Kursbezeichnung(id, bezeichnung, kuerzel)

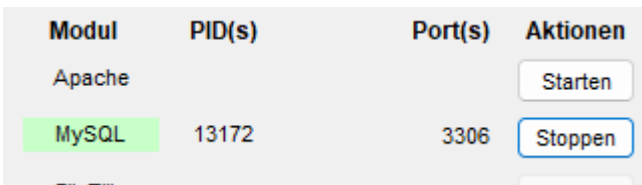
Begründungen:


Es wurde sich dafür entschieden, den Ort in eine eigene Tabelle auszulagern. Dies lässt sich begründen, da sich der Ortsname auf die PLZ bezieht und mit dem Geburtsort zudem redundant vorkommt.

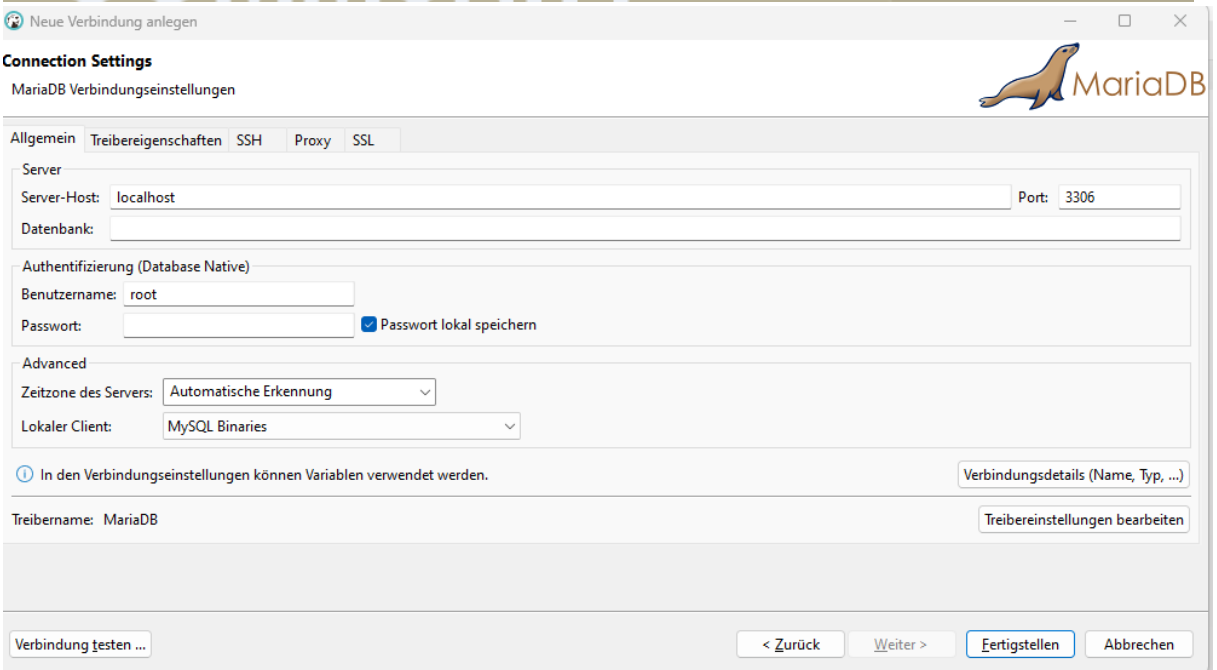
Da das Kurskürzel von der Kursbezeichnung abhängig ist, wurde dies in eine extra Tabelle ausgelagert.

Des Weiteren wurden alle Fremdschlüssel ergänzt, welche für die Umsetzung des Relationalen Modells aus dem ER-Modell benötigt werden.

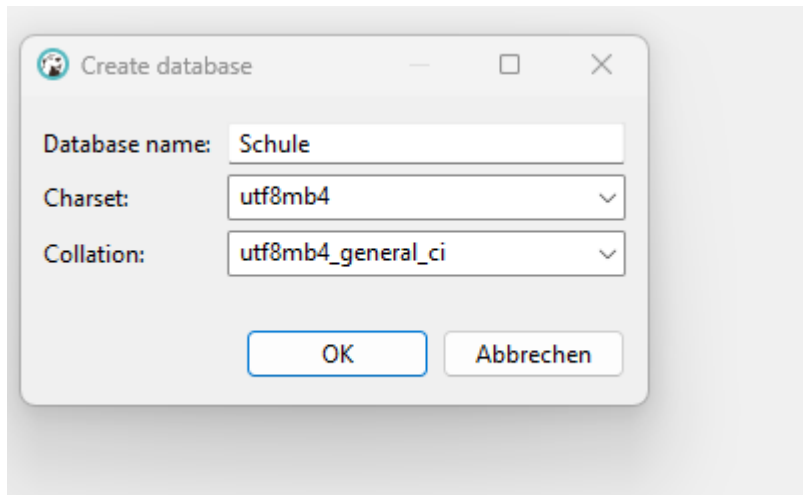
Physischer Entwurf

- 

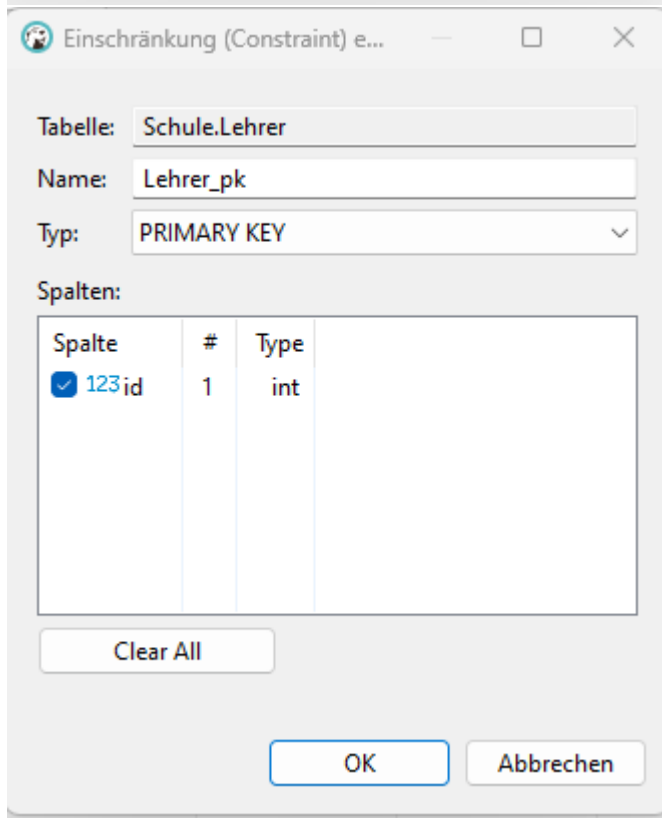
Modul	PID(s)	Port(s)	Aktionen
Apache			<button>Starten</button>
MySQL	13172	3306	<button>Stoppen</button>
- 

The screenshot shows the DBeaver application window. On the left, there's a sidebar with a cartoon animal. The main area displays 'EDITION COMMUNITY'. On the right, the 'DBeaver' logo is visible, along with the text 'Version' and 'Free Universal Database Manager'.
- 

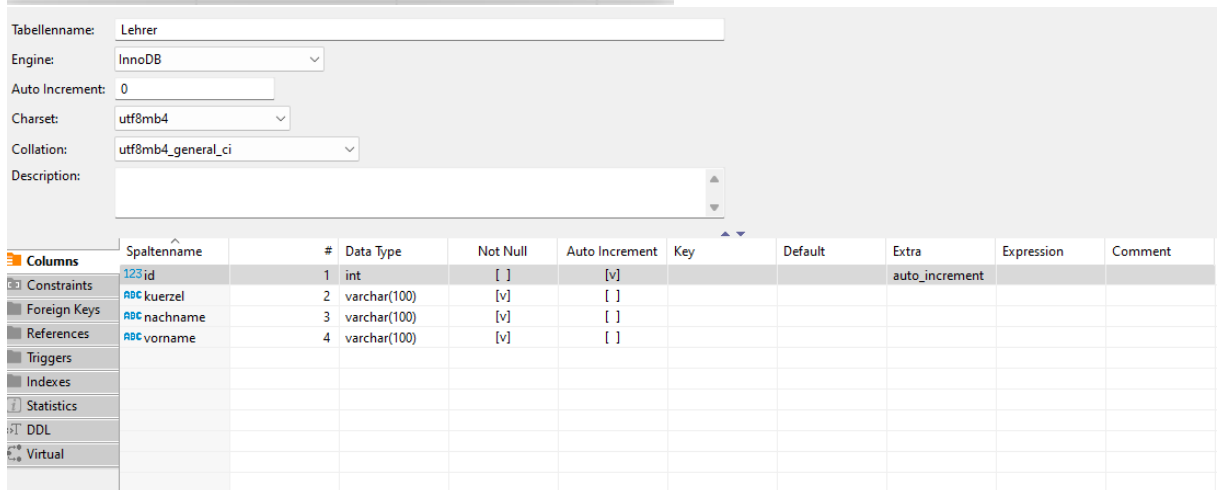
The screenshot shows the 'Connection Settings' dialog box for MariaDB. The 'Allgemein' tab is selected. The 'Server' section shows 'Server-Host: localhost' and 'Port: 3306'. The 'Datenbank:' field is empty. The 'Authentifizierung (Database Native)' section shows 'Benutzername: root' and a password field with the checkbox 'Passwort lokal speichern' checked. The 'Advanced' section shows 'Zeitzone des Servers: Automatische Erkennung' and 'Lokaler Client: MySQL Binaries'. At the bottom, there's a 'Verbindung testen ...' button and navigation buttons: '< Zurück', 'Weiter >', 'Fertigstellen', and 'Abbrechen'.



4.



5.

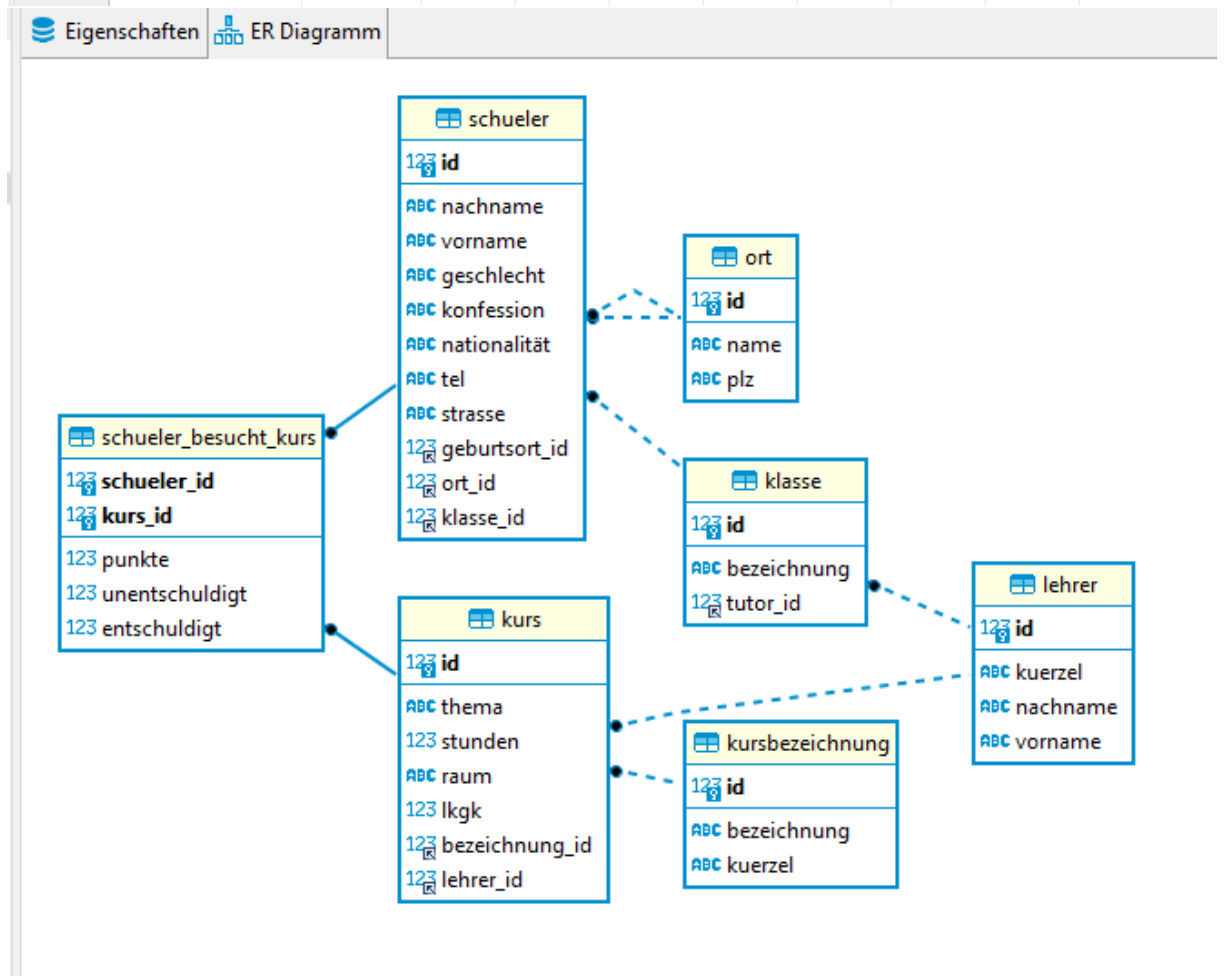


6.

Dokumentation „Eine Datenbank für das Max-Planck-Gymnasium
Projekt von Erik Liederbach und Nicholas Haag

Eigenschaften										
ER Diagramm										
Schema Name: schule			SQL Path:							
Default Charset: utf8mb4			Database size: 224K							
Default Collation: utf8mb4_general_ci										
Indexname	Tabelle	Indextyp	Ascending	Nullable	Unique	Extra	Cardinality	Comment		
> FK_lehrer	klasse	BTree			[]		0			
> PRIMARY	klasse	BTree			[v]		0			
> FK_kurs_kursbezeichnung	kurs	BTree			[]		0			
> FK_kurs_lehrer	kurs	BTree			[]		0			
> PRIMARY	kurs	BTree			[v]		0			
> PRIMARY	kursbezeichnung	BTree			[v]		0			
> PRIMARY	lehrer	BTree			[v]		0			
> PRIMARY	ort	BTree			[v]		0			
> FK_schueler_klasse	schueler	BTree			[]		0			
> FK_schueler_ort	schueler	BTree			[]		0			
> FK_schueler_ort_2	schueler	BTree			[]		0			
> PRIMARY	schueler	BTree			[v]		0			
> FK_schueler_besucht_kurs_kurs	schueler_besucht_kurs	BTree			[]		0			
> PRIMARY	schueler_besucht_kurs	BTree			[v]		0			

7.



8.

Dokumentation „Eine Datenbank für das Max-Planck-Gymnasium
Projekt von Erik Liederbach und Nicholas Haag

SQL:

```
INSERT INTO lehrer (kuerzel, nachname, vorname) VALUES ('MM', 'Mustermann', 'Max');
INSERT INTO klasse (bezeichnung, tutor_id) VALUES ('BG13b', 1), ('BG12a', 1), ('BG13c', 1), ('BG13d', 1), ('BG13a', 1);
INSERT INTO ort (`name`, plz) VALUES ('Ober-Rammstadt', '64372'), ('Darmstadt', '64285'), ('Langen', '63225'), ('Darmstadt', '64293'), ('Weiterstadt', '64331'), ('Pfungstadt', '64319'), ('Rödermark', '63322'), ('Seeheim-Jugenheim', '64342'), ('Erzhausen', '64390'), ('Dietzenbach', '63128'), ('Darmstadt', '64289'), ('Münster', '64839'), ('Darmstadt', NULL), ('Heydebreck-Cosel', NULL), ('Buenos Aires', NULL), ('Offenbach', NULL), ('Groß-Gerau', NULL);
INSERT INTO kursbezeichnung (bezeichnung, kuerzel) VALUES ('Mathematik', 'M'), ('Kunst', 'KU'), ('Sport', 'SPO'), ('Technikwissenschaft', 'TEWI'), ('Geschichte', 'G'), ('Technologie', 'TECH'), ('Ethik', 'ETHI');
INSERT INTO `schule`.`schueler` (`nachname`, `vorname`, `geschlecht`, `nationalität`, `tel`, `strasse`, `geburtsort_id`, `ort_id`, `klasse_id`) VALUES ('Schardt', 'Gina', 'W', 'D', '06154-53091', 'Langbeuneweg 26', '13', '1', '1'), ('Böttger', 'Mike', 'M', 'RK', 'D', '06151-33800', 'Landskronstraße 45', '13', '13', '2'), ('Dunn', 'Maren', 'W', 'RK', 'D', '06103-389890', 'Riedstraße 25', '14', '3', '1'), ('Maisch', 'Leo', 'M', 'EV', 'D', '06151-23475', 'Kölner Str. 39', '13', '13', '3'), ('Eckert', 'Frieder', 'W', 'EK', 'D', '06103-278671', 'Nordenstraße 75a', '3', '3', '3'), ('6', 'Weise', 'Andre', 'M', 'RK', 'D', '06150-98023', 'Meinzer Str. 20', '2', '5', '4'), ('Kainz', 'Willy', 'M', 'RK', 'D', '06157-37987', 'Adamstraße 5a', '8', '6', '5'), ('Fleming', 'Alisa', 'W', 'D', '06150-7225', 'Industriestraße 30', '13', '9', '4'), ('Schüssler', 'Beate', 'W', 'RK', 'D', '06151-8247', 'Wiesenstraße 46', '2', '5', '2');
```

Testen

Seite 1:

```
SELECT nachname, vorname FROM schueler ORDER BY nachname, vorname;
SELECT nachname, vorname FROM schueler WHERE telefon IS NULL ORDER BY nachname;
SELECT nachname, vorname FROM schueler WHERE CAST(ort AS INT) > 1 AND CAST(ort AS INT) < 8 ORDER BY nachname;
SELECT nachname, vorname FROM schueler WHERE CAST(ort AS INT) < 1 OR CAST(ort AS INT) > 8 ORDER BY nachname;
SELECT lname, lvorname FROM lehrer WHERE lname LIKE "D%";
SELECT COUNT(*) AS Zahl FROM schueler;
SELECT COUNT(*) AS Zahl FROM schueler WHERE geschlecht = "w";
SELECT COUNT(geschlecht) AS Weiblich, (SELECT COUNT(geschlecht) FROM schueler WHERE geschlecht = "m") AS Maennlich FROM schueler WHERE geschlecht = "w";
SELECT vorname, nachname FROM schueler WHERE geburtsdatum < (CURRENT_DATE - INTERVAL 18 YEAR);
SELECT vorname, nachname FROM schueler WHERE `nationalität` != "D";
SELECT DISTINCT geburtsort FROM schueler;
SELECT AVG(notenp) AS Durchschnitt FROM schueler_kurs;
SELECT SUM(anzstdt) AS Summe FROM kurs WHERE raum = "C306";
SELECT raum, SUM(anzstdt) AS Summe FROM kurs GROUP BY raum ORDER BY Summe DESC;
SELECT raum, SUM(anzstdt) AS Summe FROM kurs GROUP BY raum HAVING Summe < 10 ;
SELECT nachname, vorname, geburtsdatum FROM schueler ORDER BY geburtsdatum ASC LIMIT 1;
```


Dokumentation „Eine Datenbank für das Max-Planck-Gymnasium
Projekt von Erik Liederbach und Nicholas Haag

```
SELECT s.vorname, s.nachname, kl.bezeichnung FROM schueler s JOIN klasse kl
ON kl.KLAID = s.Klasse;
SELECT s.vorname, s.nachname, kl.bezeichnung, l.lname, l.lvorname FROM
schueler s JOIN klasse kl ON kl.KLAID = s.Klasse JOIN lehrer AS l ON l.LID
= kl.LID;
SELECT k.bezeichnung, l.lname, l.lvorname, l.`kürzel` FROM klasse k JOIN
lehrer l ON k.LID = l.LID;
SELECT k.kursthema, (SELECT COUNT(*) FROM schueler_kurs WHERE KID = k.KID)
AS Anzahl FROM kurs k;
SELECT s.vorname, s.nachname, o.ort, s.strasse, kl.bezeichnung, l.lname,
l.lvorname FROM schueler s JOIN klasse kl ON kl.KLAID = s.Klasse JOIN
lehrer l ON l.LID = kl.LID JOIN ort o ON o.OID = s.ort WHERE s.SID = 3;
SELECT k.kursthema, sk.notenp FROM schueler_kurs sk JOIN kurs k ON sk.KID =
k.KID WHERE sk.SID = 3;
SELECT s.vorname, s.nachname, sk.entschuldigt, sk.unentschuldigt,
(sk.entschuldigt + sk.unentschuldigt) AS Fehlzeit, k.kursthema FROM
schueler s JOIN schueler_kurs sk ON s.SID = sk.SID JOIN kurs k ON sk.KID =
k.KID WHERE s.SID = 3;
SELECT s.vorname, s.nachname, SUM(sk.entschuldigt) AS entschuldigt,
SUM(sk.unentschuldigt) AS unentschuldigt, (SUM(sk.entschuldigt) +
SUM(sk.unentschuldigt)) AS Gesamt FROM schueler s JOIN schueler_kurs sk ON
s.SID = sk.SID GROUP BY s.SID;
SELECT s.SID, s.vorname, s.nachname, AVG(sk.notenp) AS Durchschnitt FROM
schueler s JOIN schueler_kurs sk ON sk.SID = s.SID WHERE vorname = "Leo"
AND s.nachname = "Maisch";
SELECT s.SID, s.vorname, s.nachname, AVG(sk.notenp) AS Durchschnitt FROM
schueler s JOIN schueler_kurs sk ON sk.SID = s.SID GROUP BY s.SID HAVING
Durchschnitt > (SELECT AVG(sk.notenp) FROM schueler_kurs sk);
SELECT AVG(sk.notenp) AS Durchschnitt FROM schueler_kurs sk WHERE sk.KID =
1;
SELECT s.vorname, s.nachname, sk.notenp FROM schueler s JOIN schueler_kurs
sk ON sk.SID = s.SID WHERE sk.KID = 1 AND sk.notenp > (SELECT
AVG(sk.notenp) AS Durchschnitt FROM schueler_kurs sk WHERE sk.KID = 1);
SELECT l.`kürzel`, l.lvorname, l.lname, kl.bezeichnung FROM lehrer l LEFT
JOIN klasse kl ON kl.KLAID = l.LID;
SELECT s.nachname, s.vorname, k.kursthema, sk.notenp FROM schueler s JOIN
schueler_kurs sk ON s.SID = sk.SID JOIN kurs k ON k.KID = sk.KID;
INSERT INTO lehrer (LID, lname, lvorname, `kürzel`) VALUES(((SELECT
MAX(l.LID) FROM lehrer l) + 1), "Nolde", "Emil", "NOL");
INSERT INTO `schule`.`schueler` (`vorname`, `nachname`, `geschlecht`,
`geburtsdatum`, `geburtsort`, `konfession`, `nationalität`, `ort`,
`strasse`, `Klasse`) VALUES ('Paul', 'Ditter', 'M', '2002-06-17 00:00:00',
'Niederlande', 'EV', 'NL', '001', 'Mainzer Str. 16', '2');
UPDATE lehrer SET lname = "Janson", `kürzel` = "JAN" WHERE LID = 8;
UPDATE schueler s SET s.geburtsdatum = s.geburtsdatum + INTERVAL 1 YEAR;
INSERT INTO schueler_kurs (KID, SID, notenp, entschuldigt, unentschuldigt)
VALUES (
(SELECT KID FROM kurs k WHERE k.kursthema = "Objektorientierte
Programmierung I"),
(SELECT SID FROM schueler s WHERE s.vorname = "Merve" AND s.nachname =
"Hirt"),
10, 0, 0)
ON DUPLICATE KEY UPDATE notenp = 10;
INSERT INTO schueler_kurs (KID, SID, notenp, entschuldigt, unentschuldigt)
VALUES (
(SELECT KID FROM kurs k WHERE k.kursthema = "Organische Chemie I"),
(SELECT SID FROM schueler s WHERE s.SID = 1),
15, 0, 0)
ON DUPLICATE KEY UPDATE notenp = 15;
INSERT INTO kurs (KID, kursthema, anzstdt, lkgk, FID) VALUES ((SELECT
(MAX(k.KID) + 1) FROM kurs k), "Projektmanagement", 3, 'G', 16);
```