

SQL

- Einführung
- SQL Grundlagen
 - Datenbank und Tabellen erstellen
 - SQL Datentypen
 - Daten einfügen (INSERT)
 - Daten selektieren (SELECT)
 - Sortieren (ORDER BY)
 - DISTINCT
 - Auswahl eingrenzen (WHERE)
 - Daten aktualisieren (UPDATE)
 - Daten löschen (DELETE)
- SQL Fortgeschritten
 - Tabellen zusammenfügen (JOIN)
 - SQL Funktionen

Was ist SQL?

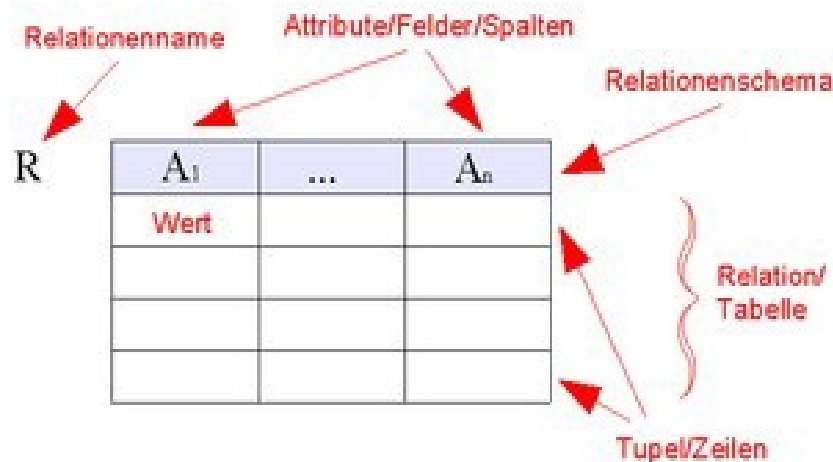
- SQL steht für *Structured Query Language* und mit ihr kann man Daten aus einer Datenbank
 - selektieren,
 - eintragen,
 - aktualisieren und
 - Löschen.
- Zudem kann man mit SQL Datenbank-Tabellen
 - erstellen,
 - aktualisieren und
 - Löschen.
- Um eine existierende Datenbank zu benutzen, braucht du SQL

Tabellen

- Jede Datenbank hat mindestens eine Tabelle
- Eine Tabelle besteht dabei aus einer Liste von Spalten (Attribute) und Typen sowie aus einigen Verwaltungseigenschaften wie Primärschlüssel
- In Tabellen werden Daten gespeichert
 - zum Beispiel in der Tabelle „teilnehmer“ werden alle teilnehmer-relevanten Daten gespeichert.
- **Tabellennormalisierung**

Spalten und Zeilen

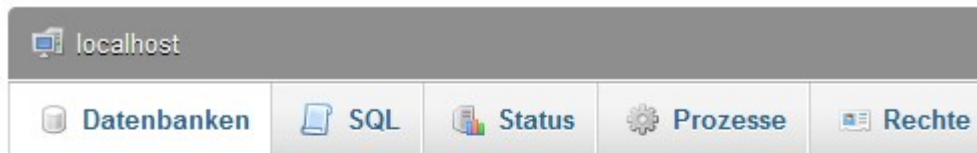
- Jede Tabelle besteht aus einer Menge aus Spalten und Zeilen.
- Jeder Datensatz (Tupel) ordnet jeder Spalte der Tabelle einen Wert zu.



Primary Key (Primärschlüssel) und Auto Increment

- Der Primary Key ist ein eindeutiger Schlüssel, der jede Zeile einer Tabelle eindeutig identifiziert, z. B. ID der Tabelle *teilnehmer*.
- Auto Increment
 - wird auf ganze Zahlen angewendet
 - beim eintragen einer neuen Zeile wird Primärschlüssel automatisch gesetzt
 - nach jedem INSERT erhöht er sich
- Anhand des Primary Keys und des Foreign Keys (Fremdschlüssel) innerhalb der Tabellen kann man Daten mittels JOINS bei einer Abfrage verknüpfen.

- Starte deinen xampp Server (mysql und Apache)
- Rufe die URL-Adresse '<http://localhost>' auf => localhost = Servername
- Gehe zu phpMyAdmin
- Klick auf *Datenbanken*
- Im Eingabefeld unter 'Neue Datenbank anlegen' *tutorial* eintragen
- Auf *Anlegen* klicken

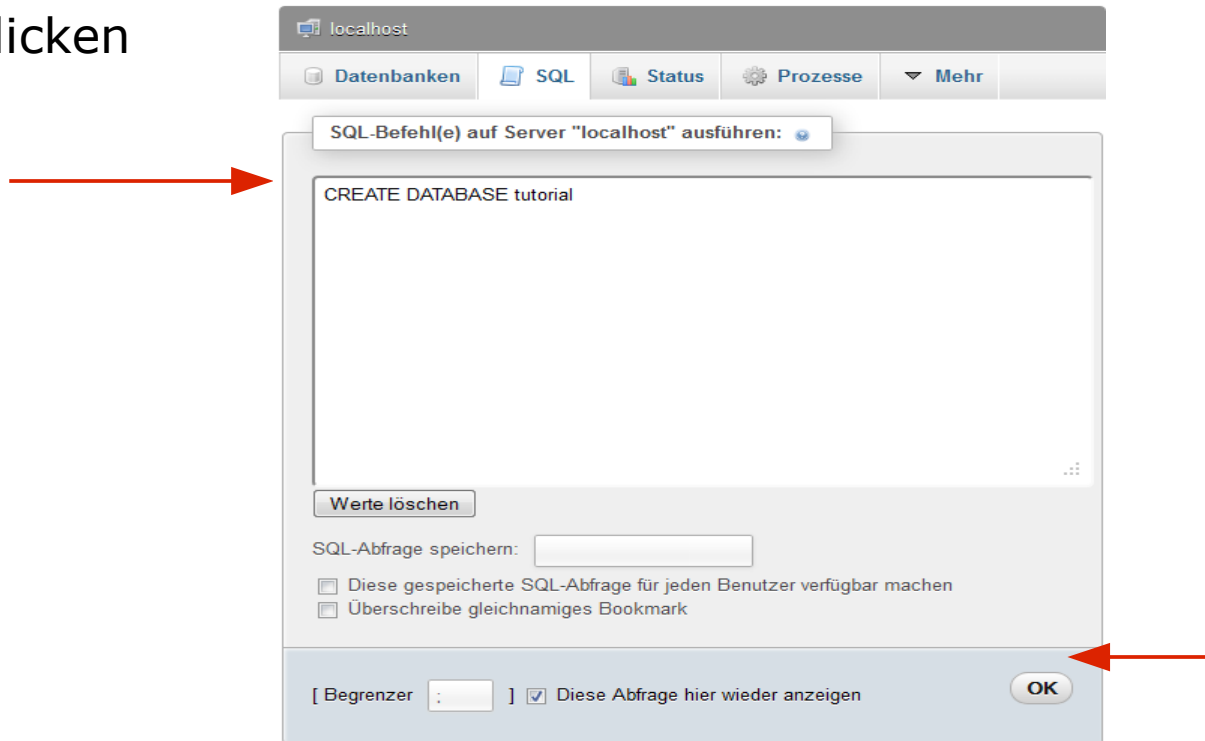


Datenbanken



SQL Grundlagen – Datenbank erstellen II

- Starte deinen xampp Server (mysql und Apache)
- Gehe zu phpMyAdmin
- Klick auf *SQL*
- Im Textarea 'CREATE DATABASE tutorial' eintragen
- Auf *OK* klicken



SQL Grundlagen – Tabellen erstellen

Mit dem CREATE Befehl werden Tabellen erstellt

Syntax

```
CREATE TABLE [IF NOT EXISTS] tabellen_name (  
  Spalte1 datentyp1 [ DEFAULT standardert1 | NULL |  
    NOT NULL ] [ AUTO_INCREMENT ],  
  ...  
  SpalteX datentypX [ DEFAULT standardertX | NULL |  
    NOT NULL ],  
  PRIMARY KEY (Spalte),  
  [ INDEX schluesselname (Spalte),]  
  [ FOREIGN KEY (Spalte) REFERENCES tabellen_name  
    (Spalte) [ ON UPDATE referenzoption] [ ON  
    DELETE ] ..]  
  ) [ENGINE=InnoDB/MyISAM] [CHARSET=CHARSET];
```

Nun führe folgenden SQL-Code innerhalb der Datenbank „tutorial“ aus:

```
CREATE TABLE IF NOT EXISTS `teilnehmer` (  
  `id` INT(10) NOT NULL AUTO_INCREMENT,  
  `vorname` VARCHAR(60) NOT NULL,  
  `nachname` VARCHAR(60) NOT NULL,  
  `geburtsdatum` DATE NOT NULL,  
  `strasse` VARCHAR(60) NOT NULL,  
  `plz` VARCHAR(5),  
  `ort` VARCHAR(30) NOT NULL,  
  PRIMARY KEY (`id`)  
  ) ENGINE=INNODB;
```

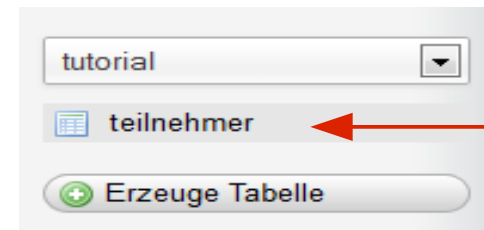


Tabelle erstellt

Datentypen: Jeder Spalte wird einen Typ zugeordnet

- Numerische Datentypen:
 - Integer:
 - Ganzzahlwerte
 - TINYINT, SMALLINT, MEDIUMINT, INT/INTEGER, BIGINT
 - Floating-Point
 - Fließkommazahlen: FLOAT, DOUBLE
 - Fixed-Point
 - Festkommazahlen: DECIMAL
 - Bit
- Zeichenbasierte Datentypen: Char, Varchar, Enum, Text (TINY, MEDIUM, LONG)
- Binary
- Zeitbezogene Datentypen: DATE, TIMESTAMP, DATETIME, TIME

SQL Grundlagen – Daten einfügen (INSERT)

Mit dem INSERT Befehl werden Daten in die Datenbank eingetragen.

Syntax

INSERT INTO tabellen_name (spalte1, spalte2, spalte3, ...)

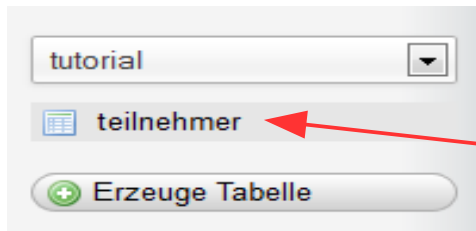
VALUES ('Wert1', 'Wert2', 'Wert3', ...)[, ('Wert1', 'Wert2', 'Wert3', ...)]...

Nun führe folgenden SQL-Code innerhalb der Datenbank „tutorial“ aus:

```
INSERT INTO `teilnehmer` (`id`, `vorname`, `nachname`, `geburtsdatum`, `strasse`, `plz`, `ort`)
VALUES
(NULL, 'Sara', 'Müller', '1992-03-03', 'Musterstraße 12', '04159', 'Leipzig'),
(NULL, 'Susane', 'Braun', '1990-07-15', 'Saagengasse 13', '01257', 'Dresden'),
(NULL, 'Felix', 'Schulze', '1991-10-20', 'Blumenweg 22', '04109', 'Leipzig'),
(NULL, 'Maria', 'Bauer', '1993-01-23', 'Kirchenstrasse 50', '01324', 'Dresden'),
(NULL, 'Jan', 'Meier', '1992-12-11', 'Daten Allee 7', '12683', 'Berlin'),
(NULL, 'David', 'Klein', '1992-08-17', 'Grüner Weg 2', '12487', 'Berlin');
```

SQL Grundlagen – Daten einfügen (INSERT)

Klicke auf „teilnehmer“, um die eingefügten Daten zu selektieren.



Klick hier

← T →					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	3	Felix	Schulze	1991-10-20	Blumenweg 22	04109	Leipzig
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

Mit dem SELECT Befehl kann man Daten aus der Datenbank selektieren.

Syntax

```
SELECT [DISTINCT] * | Datenfelder | Berechnung  
[FROM tabellen_name[, tabellen_name2, tabellen_name3...]  
[WHERE Bedingung]  
[GROUP BY Spalte [HAVING Bedingung] ]  
[ORDER BY Spalte [ASC | DESC] ]  
[LIMIT [Start, ] Anzahl];
```

SQL Grundlagen - Daten selektieren (SELECT)

Nun führe folgenden SQL-Code innerhalb der Datenbank „tutorial“ aus:

```
SELECT *
```

```
FROM teilnehmer
```

<div><div><div></div><div></div><div></div></div></div>					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
<div><div><div></div></div></div>	<div><div><div></div></div></div>	Bearbeiten	<div><div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div></div></div>	Kopieren	<div><div><div></div></div></div>	Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
<div><div><div></div></div></div>	<div><div><div></div></div></div>	Bearbeiten	<div><div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div></div></div>	Kopieren	<div><div><div></div></div></div>	Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
<div><div><div></div></div></div>	<div><div><div></div></div></div>	Bearbeiten	<div><div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div></div></div>	Kopieren	<div><div><div></div></div></div>	Löschen	3	Felix	Schulze	1991-10-20	Blumenweg 22	04109	Leipzig
<div><div><div></div></div></div>	<div><div><div></div></div></div>	Bearbeiten	<div><div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div></div></div>	Kopieren	<div><div><div></div></div></div>	Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
<div><div><div></div></div></div>	<div><div><div></div></div></div>	Bearbeiten	<div><div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div></div></div>	Kopieren	<div><div><div></div></div></div>	Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
<div><div><div></div></div></div>	<div><div><div></div></div></div>	Bearbeiten	<div><div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div></div></div>	Kopieren	<div><div><div></div></div></div>	Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

SQL Grundlagen - Daten selektieren (SELECT)

Nur bestimmte Spalten selektieren:

```
SELECT vorname, nachname, geburtsdatum  
FROM teilnehmer
```

						vorname	nachname	geburtsdatum			
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Sara	Müller	1992-03-03
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Susane	Schmidt	1990-07-15
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Felix	Schultz	1991-10-20
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Maria	Bauer	1993-01-23
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Jan	Meier	1992-12-11
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	David	Klein	1992-08-17

SQL Grundlagen - Daten selektieren (SELECT)

Spaltennamen umbenennen mit „AS“:

SELECT vorname, nachname, geburtsdatum **AS** geburtstag
FROM teilnehmer

← T →

					vorname	nachname	geburtstag
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Sara	Müller	1992-03-03
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Susane	Schmidt	1990-07-15
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Felix	Schultz	1991-10-20
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Maria	Bauer	1993-01-23
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Jan	Meier	1992-12-11
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	David	Klein	1992-08-17

SQL Grundlagen - Daten selektieren (SELECT)

Mit ORDER BY kann man das Ergebnis einer Selektion auf- oder absteigend sortieren.

- ASC für aufsteigend (Voreinstellung)
- DESC für absteigend

SELECT vorname, nachname, geburtsdatum AS geburtstag
FROM teilnehmer

ORDER BY nachname **DESC**




















←T→					vorname	nachname	geburtstag
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Felix	Schultz	1991-10-20
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Susane	Schmidt	1990-07-15
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Sara	Müller	1992-03-03
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Jan	Meier	1992-12-11
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	David	Klein	1992-08-17
<input type="checkbox"/>	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Maria	Bauer	1993-01-23

Nachnamen absteigend sortiert












SQL Grundlagen - Daten selektieren (SELECT)

Wenn man eine Tabelle hat, in der viele Werte doppelt vorkommen, kann man mit dem Schlüsselwort **DISTINCT** die Selektion von doppelten Werten befreien.

```
SELECT ort  
FROM teilnehmer  
ORDER BY nachname DESC
```

←T→					ort
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Leipzig
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Dresden
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Leipzig
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Berlin
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Berlin
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Dresden

```
SELECT DISTINCT ort  
FROM teilnehmer  
ORDER BY nachname DESC
```

←T→					ort
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Dresden
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Leipzig
<input type="checkbox"/>		Bearbeiten		Direkt bearbeiten	 Kopieren  Löschen Berlin

SQL Grundlagen - Daten selektieren (SELECT)

Mit WHERE kann man das Ergebnis einer Selektion eingrenzen.

```
SELECT vorname, nachname, geburtsdatum AS geburtstag, ort  
FROM teilnehmer
```

```
WHERE ort = 'Leipzig'
```

```
ORDER BY nachname DESC
```

					vorname	nachname	geburtstag	ort				
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Felix	Schultz	1991-10-20	Leipzig
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	Sara	Müller	1992-03-03	Leipzig

SQL Grundlagen - Daten selektieren (SELECT)

Wenn man keinen bestimmten User selektieren möchte, sondern z. B. alle User deren *vorname* mit dem Buchstaben „S“ beginnt, kann man dafür das Schlüsselwort „LIKE“ mit einer Wildcard (%) nutzen:

```
SELECT vorname, nachname, geburtsdatum AS geburtstag  
FROM teilnehmer
```

```
WHERE vorname LIKE 'S%'
```

```
ORDER BY nachname DESC
```

←T→					vorname	nachname	geburtstag
	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Susane	Schmidt	1990-07-15
	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Sara	Müller	1992-03-03

SQL Grundlagen - Daten selektieren (SELECT)

Mit AND und OR kann man seine Auswahl noch verfeinern:

```
SELECT vorname, nachname, geburtsdatum AS geburtstag, ort  
FROM teilnehmer
```

```
WHERE vorname LIKE 'S%'
```

```
AND ort = 'Leipzig'
```

```
ORDER BY nachname DESC
```

A screenshot of a table editor interface. At the top left is a toolbar with icons for undo, redo, and a table structure icon. Below the toolbar are four buttons: 'Bearbeiten' (with a pencil icon), 'Direkt bearbeiten' (with a document icon), 'Kopieren' (with a copy icon), and 'Löschen' (with a red minus icon). To the right of these buttons is a table with four columns: 'vorname', 'nachname', 'geburtstag', and 'ort'. The first row of data contains the values 'Sara', 'Müller', '1992-03-03', and 'Leipzig'.

vorname	nachname	geburtstag	ort
Sara	Müller	1992-03-03	Leipzig

SQL Grundlagen - Daten selektieren (SELECT)

Mit AND und OR kann man seine Auswahl noch verfeinern:

```
SELECT vorname, nachname, geburtsdatum AS geburtstag, ort  
FROM teilnehmer
```

```
WHERE ort = 'Leipzig' OR ort = 'Dresden'
```

```
ORDER BY nachname DESC
```

					vorname	nachname	geburtstag	ort
	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Felix	Schultz	1991-10-20	Leipzig
	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Susane	Schmidt	1990-07-15	Dresden
	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Sara	Müller	1992-03-03	Leipzig
	 Bearbeiten	 Direkt bearbeiten	 Kopieren	 Löschen	Maria	Bauer	1993-01-23	Dresden

SQL Grundlagen - Daten aktualisieren (UPDATE)

- Mit der UPDATE-Anweisung kann einen oder auch mehrere Datensätze gleichzeitig aktualisieren.
- SET setzt die neuen Werte
- Die Auswahl der betreffenden Datensätze erfolgt dabei über eine WHERE-Anweisung.

Syntax

UPDATE tabellen_name

SET Spalte1 = wert1, ..., SpalteX = wertX

WHERE bedingung;

SQL Grundlagen - Daten aktualisieren (UPDATE)

Beispiel

					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	3	Felix	Schulze	1991-10-20	Blumenweg 22	04109	Leipzig
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

UPDATE teilnehmer

SET nachname = 'Schultz'

WHERE id = 3;

<div><div><div></div><div></div><div></div></div></div>					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
<div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div></div>	Bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Kopieren	<div><div><div></div><div></div><div></div></div></div>	Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
<div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div></div>	Bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Kopieren	<div><div><div></div><div></div><div></div></div></div>	Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
<div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div></div>	Bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Kopieren	<div><div><div></div><div></div><div></div></div></div>	Löschen	3	Felix	Schultz	1991-10-20	Blumenweg 22	04109	Leipzig
<div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div></div>	Bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Kopieren	<div><div><div></div><div></div><div></div></div></div>	Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
<div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div></div>	Bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Kopieren	<div><div><div></div><div></div><div></div></div></div>	Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
<div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div></div>	Bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Direkt bearbeiten	<div><div><div></div><div></div><div></div></div></div>	Kopieren	<div><div><div></div><div></div><div></div></div></div>	Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

Mit DELETE kann man Einträge aus der Datenbank löschen.

Syntax

DELETE FROM tabellen_name; - ganze Tabelle löschen

DELETE FROM tabellen_name
[**WHERE** Bedingung]

TRUNCATE [TABLE] tabellen_name; - Tabelle leeren

SQL Grundlagen - Daten löschen (DELETE)

Beispiel

					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	3	Felix	Schultz	1991-10-20	Blumenweg 22	04109	Leipzig
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

DELETE FROM teilnehmer

WHERE id = 3;

					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
		Bearbeiten		Direkt bearbeiten		Kopieren		Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

- Bei relationalen Datenbanken werden die Daten in der Regel auf mehrere, logisch zusammengehörige Tabellen verteilt.
- Eine wichtige Rolle spielen dabei Primär- und Fremdschlüssel, die die Verbindung von Datensätzen in mehreren Tabellen vereinfachen.
- Eine Tabellenreferenzierung heißt auch Join-Ausdruck.

Beziehungen zwischen Tabellen

Es gibt drei Grundtypen von Beziehungen:

- **1:1-Beziehung**

In einer 1:1-Beziehung ist jedem Datensatz in Tabelle A nur ein passender Datensatz in Tabelle B zugeordnet und umgekehrt. Zum Beispiel

- **1:n-Beziehung**

Eine 1:n-Beziehung ist der häufigste Beziehungstyp. In einer 1:n-Beziehung können einem Datensatz in Tabelle A mehrere passende Datensätze in Tabelle B zugeordnet sein, aber einem Datensatz in Tabelle B ist nie mehr als ein Datensatz in Tabelle A zugeordnet.

- **m:n-Beziehung**

In einer m:n-Beziehung können jedem Datensatz in Tabelle A mehrere passende Datensätze in Tabelle B zugeordnet sein und umgekehrt. Dies ist nur möglich, indem eine dritte Tabelle definiert wird (die als Verbindungstabelle bezeichnet wird), deren Primärschlüssel aus zwei Feldern besteht: den Fremdschlüsseln aus den Tabellen A und B.

1. Equi-Join (Inner Join)

Wird auf Gleichheit geprüft (=)

Syntax I

```
SELECT datenfelder  
FROM tabelle1  
INNER JOIN tabelle2 join_bedingung  
[WHERE]
```

join_bedingung:

```
ON bedingter_ausdruck | USING  
(Spalte1[, Spalte2,...])
```

Syntax II

```
SELECT datenfelder  
FROM tabelle1, tabelle2, ...  
WHERE tabelle1.datenfeld =  
        tabelle2.datenfeld  
[AND ..]
```

Beispiele: Equi-Join

Syntax I

```
SELECT *  
FROM teilnehmer  
INNER JOIN projekt_teilnehmer ON teilnehmer.id = projekt_teilnehmer.teilnehmer_id  
INNER JOIN projekt ON projekt_teilnehmer.projekt_id = projekt.id
```

Syntax II

```
SELECT *  
FROM teilnehmer, projekt, projekt_teilnehmer  
WHERE teilnehmer.id = projekt_teilnehmer.teilnehmer_id  
AND projekt_teilnehmer.projekt_id = projekt.id
```

2. Left-Outer-Join

LEFT JOIN funktioniert ähnlich wie INNER JOIN mit dem Unterschied, dass Einträge der linken Tabelle keine Verbindung zu den Daten der rechten Tabelle haben müssen, um selektiert zu werden.

Syntax

SELECT datenfelder

FROM tabelle1

LEFT [OUTER] JOIN tabelle2 **join_bedingung**

[**WHERE** ..]

join_bedingung:

ON bedingter_ausdruck | **USING** (Spalte1[, Spalte2,..])

3. Right-Outer-Join

RIGHT JOIN funktioniert genau wie LEFT JOIN, nur in diesem Fall ist alles umgedreht. Beim RIGHT JOIN werden die Einträge der rechten Tabelle selektiert, auch wenn keine Verbindung zu den Daten der linken Tabelle besteht.

Syntax

SELECT datenfelder

FROM tabelle1

RIGHT [OUTER] JOIN tabelle2 **join_bedingung**

[**WHERE** ..]

join_bedingung:

ON bedingter_ausdruck | **USING** (Spalte1[, Spalte2,..])

4. Self-Join

Für die Tabelle müssen zwei verschiedene Ersatznamen angegeben werden. Die Tabelle wird über den Befehl Inner-Join mit sich selbst verknüpft.

SQL Funktionen bieten die Möglichkeit Rechenoperationen auf den selektierten Daten auszuführen. Einige Funktionen sind:

- AVG() – Durchschnittswert berechnen
- COUNT() – Zeilen zählen
- MAX() – den höchsten Wert einer Spalte selektieren
- MIN() – den niedrigsten Wert einer Spalte selektieren
- SUM() – Zeilenwerte summieren
- UCASE() – Werte einer Spalte in Großbuchstaben
- LCASE() – Wert einer Spalte in Kleinbuchstaben
- MID() – Zeichen extrahieren
- LENGTH() / CHAR_LENGTH – die Länge einer Zeichenkette einer Spalte
- ROUND() – selektierte Werte runden
- NOW() – aktuelles Datum und Zeit

HAVING ist eine Bedingung, die auf aggregierte Werte angewendet werden kann. Die WHERE Bedingung kann zum Beispiel auf gruppierte Werte (GROUP BY) nicht angewendet werden, dafür muss man HAVING verwenden.

Syntax

```
SELECT spalten_name, aggregations_funktion(spalten_name)
FROM tabellen_name
GROUP BY spalten_name
HAVING aggregations_funktion(spalten_name) operator wert
```

Beispiel

<div><div></div><div></div><div></div></div>					id	vorname	nachname	geburtsdatum	strasse	plz	ort				
<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	Bearbeiten	<div><div></div><div></div><div></div></div>	Direkt bearbeiten	<div><div></div><div></div><div></div></div>	Kopieren	<div><div></div><div></div><div></div></div>	Löschen	1	Sara	Müller	1992-03-03	Musterstraße 12	04159	Leipzig
<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	Bearbeiten	<div><div></div><div></div><div></div></div>	Direkt bearbeiten	<div><div></div><div></div><div></div></div>	Kopieren	<div><div></div><div></div><div></div></div>	Löschen	2	Susane	Schmidt	1990-07-15	Saagengasse 13	01257	Dresden
<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	Bearbeiten	<div><div></div><div></div><div></div></div>	Direkt bearbeiten	<div><div></div><div></div><div></div></div>	Kopieren	<div><div></div><div></div><div></div></div>	Löschen	4	Maria	Bauer	1993-01-23	Kirchenstrasse 50	01324	Dresden
<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	Bearbeiten	<div><div></div><div></div><div></div></div>	Direkt bearbeiten	<div><div></div><div></div><div></div></div>	Kopieren	<div><div></div><div></div><div></div></div>	Löschen	5	Jan	Meier	1992-12-11	Daten Allee 7	12683	Berlin
<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	Bearbeiten	<div><div></div><div></div><div></div></div>	Direkt bearbeiten	<div><div></div><div></div><div></div></div>	Kopieren	<div><div></div><div></div><div></div></div>	Löschen	6	David	Klein	1992-08-17	Grüner Weg 2	12487	Berlin

```
SELECT ort, COUNT(ort) AS anzahl_orte
```

```
FROM teilnehmer
```

```
GROUP BY ort
```

```
HAVING anzahl_orte > 1
```

ort	anzahl_orte
Berlin	2
Dresden	2

Übungen

```
SELECT YEAR(NOW()) - YEAR(geburtsdatum)
FROM teilnehmer
```

```
SELECT CHAR_LENGTH(vorname)
FROM teilnehmer
```

```
SELECT UCASE(vorname)
FROM teilnehmer
```

```
SELECT LCASE(vorname)
FROM teilnehmer
```

Übungen

```
SELECT MID(vorname, 1, 3)
```

```
FROM teilnehmer
```

```
SELECT MAX(YEAR(NOW()) - YEAR(geburtsdatum))
```

```
FROM teilnehmer
```

```
SELECT MIN(YEAR(NOW()) - YEAR(geburtsdatum))
```

```
FROM teilnehmer
```

```
SELECT AVG(YEAR(NOW()) - YEAR(geburtsdatum))
```

```
FROM teilnehmer
```

```
SELECT SUM(YEAR(NOW()) - YEAR(geburtsdatum))
```

```
FROM teilnehmer
```

Übungen

Equi-Join und COUNT()

```
SELECT COUNT(teilnehmer.id), projekt.name  
FROM teilnehmer, projekt, projekt_teilnehmer  
WHERE teilnehmer.id = projekt_teilnehmer.teilnehmer_id  
AND projekt_teilnehmer.projekt_id = projekt.id  
GROUP BY projekt.id
```

<http://dev.mysql.com/doc/refman/5.1/de/data-types.html>

<http://dev.mysql.com/doc/refman/5.1/de/join.html>

<http://dev.mysql.com/doc/refman/5.1/de/functions.html>