



Datenbanken und SQL

(Woche 2 - Tag 2)

Agenda

ALTER TABLE -> Data Definition Language (Fortsetzung)

- Definition
- Motivation
- Syntax
 - ADD
 - Nichtschlüssel-Attribut
 - Fremdschlüssel
 - DROP
 - Nichtschlüssel-Attribut
 - Fremdschlüssel
 - CHANGE
 - RENAME

DDL (Fortsetzung)

ALTER TABLE -> Definition

- „ALTER TABLE“ ist der **Oberbegriff** für eine ganze Reihe von Befehlen mit deren Hilfe Tabellen (nachträglich) verändert werden können.
- Da auf diese Weise die Struktur (bzw. Definition) des Datenbankschemas verändert wird, zählen diese Befehle zur **DDL** (Data Definition Language).
- Folgende **Veränderungen** werden wir kennen lernen:
 - Hinzufügen eines neuen Attributs
 - Ändern eines bestehenden Attributs
 - Löschen eines bestehenden Attributs
 - Ändern eines Tabellennamens

ALTER TABLE -> Motivation

- Falls die mittels „CREATE TABLE“ erstellten Tabellen fehlerhaft sind, können wir sie mit Hilfe von „ALTER TABLE“ entsprechend **korrigieren**.
- Ferner könnte es notwendig werden, ein ursprünglich korrekt erstelltes Schema im Laufe der Zeit zu **aktualisieren**.
- Darüber hinaus werden wir Situationen kennen lernen, in denen (zumindest im Falle einer aktivierten Fremdschlüsselüberprüfung) ein Einsatz von „ALTER TABLE“ **unverzichtbar** ist.

ALTER TABLE

ALTER TABLE Tabellen-Name **ADD** Attribut-Name Definition

Um in der Tabelle „Kunde“ (z.B.) das **neue Attribut „Anrede“** hinzuzufügen, verwenden wir den folgenden Befehl:

```
ALTER TABLE Kunde ADD Anrede VARCHAR(255) NOT NULL ;
```

Der durch die **geschweifte Klammer** markierte Bereich entspricht **exakt jener Syntax**, die verwendet worden wäre, hätte man dieses Attribut bereits von Beginn an bei der Implementierung der Tabelle mittels „**CREATE TABLE** Kunde“ erzeugt.

ALTER TABLE Tabellen-Name ADD Fremdschlüssel-Attribut

Wir beweisen Phantasie ;-) und behaupten, dass wir jedem Produkt **höchstens 1** Kunden zuordnen können, der dieses Produkt erfunden hat. Da jeder Kunde auch **mehrere** Produkte erfunden haben könnte, bestünde dann eine **1:n-Beziehung** zwischen Kunde und Produkt. Entsprechend müsste zur Tabelle **Produkt** zunächst das Attribut **Kunde_ID** **hinzugefügt** werden:

```
ALTER TABLE Produkt ADD Kunde_ID INT(11);
```

Anschließend muss dann aber auch noch mitgeteilt werden, dass das Attribut **Kunde_ID** auf die Tabelle **Kunde** referenziert:

```
ALTER TABLE Produkt ADD FOREIGN KEY(Kunde_ID) REFERENCES Kunde(Kunde_ID);
```

Der durch die **geschweifte Klammer** markierte Bereich entspricht **erneut jener Syntax**, die verwendet worden wäre, hätte man diesen Fremdschlüssel-Constraint bereits von Beginn an bei der Implementierung der Tabelle mittels „**CREATE TABLE Produkt**“ erzeugt.

ALTER TABLE Tabellen-Name DROP Attribut-Name

Um (z.B.) in der Tabelle „Kunde“ das Attribut „Anrede“ zu löschen, verwenden wir den folgenden Befehl:

```
ALTER TABLE Kunde DROP Anrede;
```

ALTER TABLE Tabellen-Name DROP **einziges** Fremdschlüssel-Attribut

In der Tabelle *Abrechnung* existiert nur ein **einziges** Fremdschlüssel. Daher ist die **interne Namensgebung** in diesem Fall eindeutig:

```
ALTER TABLE Abrechnung DROP FOREIGN KEY Abrechnung_ibfk_1;
```

Nachdem wir dadurch die **Referenz** des Fremdschlüssels *Kunde_ID* (in „Abrechnungen“) auf den Primärschlüssel *Kunde_ID* (in „Kunde“) gelöscht haben, können wir nun - trotz aktivierter Fremdschlüsselüberprüfung – das Attribut **Kunde_ID** (in „Abrechnungen“) löschen :

```
ALTER TABLE Abrechnung DROP Kunde_ID;
```

Hinweis:

Die Namensgebung **TabellenName_ibfk_Nummer** ist eine individuelle Entscheidung des Entwicklers von MariaDB und keine allgemeingültige **Syntax** für alle SQL-Dialekte!

ALTER TABLE Tabellen-Name DROP einen (von mehreren) Fremdschlüsseln

Der Kunde hat weniger Phantasie ☺ und verlangt, dass der Fremdschlüssel „Kunde_ID“ aus der Tabelle Produkt gelöscht wird. Die Tabelle Produkt besitzt aber mittlerweile **mehrere** Fremdschlüssel (Hersteller_ID und Kunde_ID).

Um die interne Namensgebung ermitteln zu können, rufen wir zunächst den Befehl **SHOW CREATE TABLE Produkt** auf, worauf die folgende Ausgabe erscheinen wird. [Im Zweifel müssen Sie „Optionen“ anklicken und dort „vollständige Texte“ auswählen]

+ Optionen

Table	Create Table
produkt	<pre>CREATE TABLE `produkt` (`produkt_id` int(11) NOT NULL AUTO_INCREMENT, `hersteller_id` int(11) NOT NULL, `produkt_name` varchar(250) NOT NULL, `Euro_preis` float(11,2) NOT NULL, `kunde_id` int(11) DEFAULT NULL, PRIMARY KEY (`produkt_id`), KEY `hersteller_id` (`hersteller_id`), KEY `kunde_id` (`kunde_id`), CONSTRAINT `produkt_ibfk_1` FOREIGN KEY (`hersteller_id`) REFERENCES `hersteller` (`hersteller_id`), CONSTRAINT <u>produkt_ibfk_2</u> FOREIGN KEY (<u>kunde_id</u>) REFERENCES `kunde` (`kunde_id`)) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8mb4</pre>

Anschließend können wir in der Tabelle Produkt die Referenz von Kunde_ID auf die Tabelle „Kunde“ löschen:

ALTER TABLE Produkt DROP FOREIGN KEY Produkt_ibfk_2;

Erst durch die Aufhebung dieser Referenz können wir nun auch das Attribut Kunde_ID aus der Tabelle Produkt löschen:

ALTER TABLE Produkt DROP Kunde_ID;

ALTER TABLE Tabellen-Name **CHANGE** bisheriger Attribut-Name neuer Attribut-Name Definition

Um (z.B.) in der Tabelle „Kunde“ das Attribut „Vorname“ zu ändern, verwenden wir den folgenden Befehl:

```
ALTER TABLE Kunde CHANGE Vorname Kunde_Vorname VARCHAR(50) NOT NULL ;
```

Dies entspricht **auch diesmal jener Syntax**, die wir schon bei „CREATE TABLE Kunde“ hätten nutzen können.

Natürlich kann frei gewählt werden, **was** geändert werden soll. Insbesondere ist man also nicht verpflichtet, den Attribut-Namen zu ändern. Hierzu eine Beispiel:

```
ALTER TABLE Kunde CHANGE Vorname Vorname VARCHAR(100) NULL;
```

ALTER TABLE alter Tabellen-Name RENAME neuer Tabellen-Name

Um (z.B.) den Namen der Tabelle „Abrechnung“ durch die neue Bezeichnung „invoice“ zu überschreiben, verwenden wir den folgenden Befehl:

```
ALTER TABLE Abrechnung RENAME invoice;
```

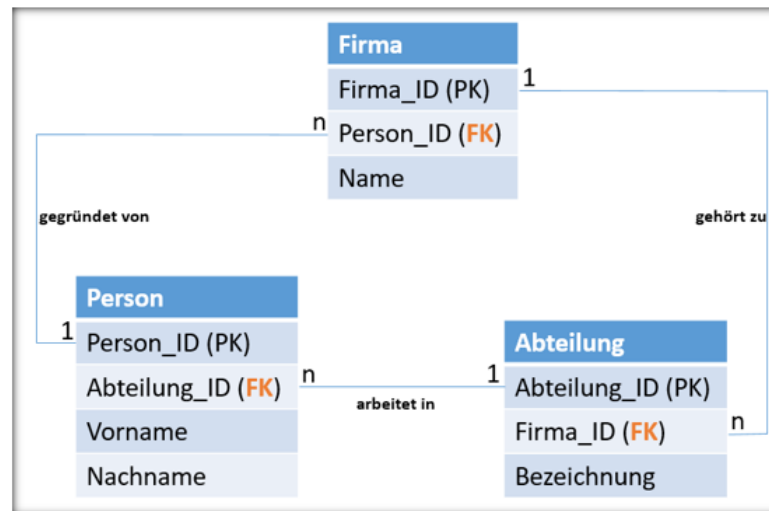
Hinweis:

Durch die Umbenennung wird auch die interne Bezeichnung des Fremdschlüssels angepasst:

Der Fremdschlüssel „Kunde_ID“ hieß bisher intern `abrechnung_ibfk_1`.
Nach der Umbenennung wird er intern nun entsprechend `invoice_ibfk_1` genannt.

Gemeinsame Übung („Live-Coding“) -> A_02_02_01

Aufgabe_02_02_01



Aufgabenstellung:

Implementieren Sie bitte das obige Schema. Achten Sie bitte darauf, dass **alle** Tabellen einen **Fremdschlüssel** besitzen!
(Hinweis: Diese Aufgabe ist ein Beispiel dafür, dass der Einsatz von ALTER TABLE im Einzelfall unvermeidbar ist)