

## 2. Normalform (2NF)



Ein Relationstyp (Tabelle) befindet sich in der zweiten Normalform (2NF), wenn die erste Normalform erfüllt ist und für jeden Primärschlüssel eindeutige Attributwerte vorhanden sind.

Die 2. Normalform entfernt Redundanzen:



In der Tabelle in der 1NF tauchen zahlreiche Attributwerte mehrfach auf, z.B. Die Namen der Doktoren oder Hersteller.

### Grundsatz: Eine Entität - eine Tabelle

Wenn man den **Grundsatz** – den unser Sekretär mit seinem an die Tabellenkalkulation angelehnten Vorgehen bereits verletzt hat, den wir aber eigentlich kennen – befolgt: **Jede Entität bekommt ihre eigene Relation, die Attribute sind die Tabellenspalten<sup>1)</sup>** und die Attribute entsprechend der Miniwelt atomar wählt erhält man relativ direkt die 2NF.

Hier sieht man nochmal, dass die **Normalisierung beim Design der Datenbank stattfinden sollte**, nicht wie wir das hier machen, nachdem man bereits ein schlechtes Design implementiert hat.



#### (A1)

- Überführe die Inhalte der „Universaltabelle“ in der 1NF in drei Tabellen: doktoren, hersteller, produkte. Verteile die Attribute auf die Tabellen der Entitäten. Gehe wie folgt vor:
  1. Lege die drei Tabellen mit den entsprechenden Attributen an, jedoch zunächst ohne Surrogatschlüssel (keine id Spalte!
  2. Überführe die Inhalte aus der Universaltabelle in die jeweilige Tabelle, indem du das Ergebnis einer Abfrage direkt an eine INSERT INTO Statement weitergibst: `INSERT INTO doktoren SELECT DISTINCT name, vorname, strasse, plz, wohnort, telefon, fax FROM `zahnarztbedarf``. Du musst auf die Reihenfolge der Attribute bei der Abfrage achten, damit die Werte in die richtigen Felder eingefügt werden. Warum benötigt man das Schlüsselwort DISTINCT bei der Abfrage?
  3. Wenn alle Werte überführt sind, fügst du jeder Tabelle noch eine Feld id als Surrogatschlüssel hinzu (Primary-Key, Auto-Increment).
- Zwei Tabellenspalten der Universaltabelle können den Entitätstabellen nicht sinnvoll zugeordnet werden - bei einer geht keine wesentliche Information verloren, wenn man sie weglässt, bei der

anderen schon. Welche Attribute sind das? Lasse beide Attribute vorerst einfach aus - wir kommen später darauf zurück.

- Fügen nun den Datensatz für Frau Bohrgut ein aus den [Vorüberlegungen](#) ein - du siehst, nun ist ganz klar, welche Informationen in welche Tabelle eingefügt werden müssen.

### Lösungshinweise Tabellenstrukturen:



### SQL Befehle:

```
INSERT INTO doktoren
SELECT DISTINCT name, vorname, strasse, plz, wohnort, telefon, fax
FROM `zahnarztbedarf`
```

```
INSERT INTO hersteller
SELECT DISTINCT firma, f_strasse, f_ort, f_plz
FROM `zahnarztbedarf`
```

```
INSERT INTO produkte
SELECT DISTINCT produkt, preis, nummer
FROM `zahnarztbedarf`
```

### Ergebnis:



### (A2)

Was bei der Normalisierung nun dummerweise verloren gegangen ist, ist die Zuordnung der Bestellungen zu unseren Kunden, den Doktoren - deswegen hatte der Sekretär die Tabelle ja mal angelegt. Auch die Verbindung zwischen Herstellern und Produkten sucht man in den einzelnen Tabellen vergeblich?!

Um die Verknüpfung zwischen den Entitäten nun wieder herzustellen, wird eine weitere Tabelle angelegt, die lediglich die Verknüpfung zwischen den anderen Entitäten herstellt.

Erstelle eine Tabelle bestellungen mit den Feldern id (Primary-Key, Auto-Increment), produkt\_id, doktor\_id, hersteller\_id und anzahl.

Nun schauen wir in unserer Universaltafel nach den Bestellungen:



Die dritte Zeile sagt uns, dass Viktoria Einsenfaust drei Kneifzangen bestellt hat.

Um das in unserer bestellungen-Tabelle abzubilden, müssen wir folgendes eintragen:

| id | anzahl | produkt_id | doktor_id | hersteller_id |
|----|--------|------------|-----------|---------------|
| 1  | 3      | 3          | 3         | 3             |

(i) Was muss man für die Bestellung in der letzten Zeile der Universaltafel eintragen?

### Lösung

| id | anzahl | produkt_id | doktor_id | hersteller_id |
|----|--------|------------|-----------|---------------|
| 2  | 2      | 4          | 1         | 3             |

(ii) Komplettiere die Bestellungstabelle in deiner Übungsdatenbank und führe dann die folgende Abfrage aus:

```
SELECT * FROM hersteller, doktoren, produkte, bestellungen
WHERE bestellungen.produkt_id = produkte.id
AND bestellungen.doktor_id = doktoren.id
AND bestellungen.hersteller_id = hersteller.id
```

Erläutere, was bei dieser Abfrage passiert und interpretiere das Ergebnis.



Nun befinden sich alle Relationen unserer Datenbank in der 2. Normalform<sup>2)</sup>.

<sup>1)</sup>

Erinnerung: Relation ist nur ein anderer Name für Tabelle...

<sup>2)</sup>

in unserem speziellen Fall sogar zufällig bereits in der 3.

From:  
<https://wiki.qg-moessingen.de/> - QG Wiki

Permanent link:  
[https://wiki.qg-moessingen.de/faecher:informatik:oberstufe:datenbanken:normalisierung:2\\_normalform:start?rev=1606335891](https://wiki.qg-moessingen.de/faecher:informatik:oberstufe:datenbanken:normalisierung:2_normalform:start?rev=1606335891)

Last update: 25.11.2020 21:24

