

Datenbanken

War, was und wie

engelbert gruber

02.04, 25.5.2020

2ND UPGRADE STUPID

INTRODUCTION

Warum: Datenbanken braucht man, wenn die Datenmenge so groß ist, dass sie nicht mehr in den Hauptspeicher passt. Das heisst bei heutigen Hauptspeicher Größen, es geht auch ohne Datenbank.

Warum trotzdem: Bei gleichzeitigem Zugriff von mehreren Anwendungen braucht es ein System, dass die Zugriffe reiht, die Datenkonsistenz gewährt.

Was: Wenn man von einer Datenbank redet ist das meist eine relationale Datenbank. Das ist eine Anzahl Tabellen, deren Inhalt über Beziehungen (Relationen) zusammenhängt. Zum Beispiel: Personen und Adressen und Telefonnummern.

Wie: siehe unten

WIE : Beispiel PIM

PIM Person Information System: Personen ihre Adressen und Telefonnummern.

Man kann eine einzelne Tabelle machen und darin alles eintragen:

Person1, Wohnadresse, Arbeitsadresse, Wohntel., Mobiltel., Arbeitstel.

Person2, Wohnadresse, Arbeitsadresse, Mobiltel1., Mobiltel2, Arbeitstel., Email

...

Wenn das in einer Tabellenkalkulation eingegeben wird, ist die Tabelle zu breit um überblickbar zu sein. Und die Möglichkeit von Mehrfacheinträgen Tel1, Tel2, ... macht die Lesbarkeit auch nicht einfacher.

Tabellen Entwurf

Tabelle: Person mit Name, Vorname, ...

Tabelle: Adresse mit Verweis-zur-Personen-Tabelle, Straße, Ort, PLZ, Wohn/Arbeit

Tabelle: Telefon mit Verweis-zur-Personen-Tabelle, Privat/Arbeit, Nummer

“Verweis-zur-Personen-Tabelle” ist etwas um die Adressentabellenzeile einer Personentabellenzeile zu zuordnen.



Hier ist die Adresse für die Person in der ersten Zeile in der zweiten Zeile von Adresse und die Person in der dritten Zeile hat zwei Adressen.

Meine mit mir getroffene Konvention ist jede Tabelle hat eine Zeilennummer, diese ist in der Tabelle eindeutig (logisch) und hat den Namen ID. Das ist im Datenbankjargon der

PRIMARY KEY. Der ID muss nicht mit der Zeilennummer übereinstimmen.

In der Tabelle Adresse gibt es dann eine Spalte mit dem Namen Person_ID dort steht die ID der Zeile in Person, das ist der FOREIGN KEY.

Adresse			
ID	Str	Ort	Person_ID
1	~	~	2
2	~	~	1
3	~	~	8
4	~	~	8

Person		
ID	Name	Vorname
1	~	~
2	~	~
8	~	~

Die Spalte ID gibt es sobald es eine Tabelle gibt und sie ist zur eindeutigen Identifikation der Zeile und sonst nichts, keine Sozialversicherungs- oder Mitgliedsnummer oder was auch immer.

GLOSSAR - BEGRIFFE

RELATIONALE DATENBANKEN

Die Bezeichnung relational bezieht sich auf den mathematischen Begriff, siehe wikipedia ... für mehr Details. Relation auf Deutsch Beziehung.

SQL Structured Query Language

[Wikipedia](#) beschreibt das recht gut:

Die Sprache basiert auf der [relationalen Algebra](#), ihre [Syntax](#) ist relativ einfach aufgebaut und [semantisch](#) an die englische Umgangssprache angelehnt.

Wichtig ist mir "ihre [Syntax](#) ist ... an die englische Umgangssprache angelehnt" das bedeutet, aus einer Zeit als man versuchte Computer so zu machen, dass sie Menschen verstehen.

SQL hat drei Teile

1. Data Query Language, zur Abfrage der Daten : SELECT
2. Data Manipulation Language, zum Ändern der Daten : INSERT, UPDATE, DELETE
3. Data Definition Language, zum Definieren der Datenstruktur: CREATE, ALTER, DROP
4. Und einen vierten Teil für andere Dinge, Rechteverwaltung, Speicherverwaltung,
...

Einfaches Englisch bedeutet um eine Tabelle zu Erzeugen “CREATE TABLE” und weil wir die Struktur nachher verwenden wollen geben wir ihr einen Namen, “CREATE TABLE Person”. Das definiert noch nichts, Personen haben Namen und ein Geburtsdatum. Und wir definieren eine PRIMARY KEY.

```
CREATE TABLE Person
(
  ID int PRIMARY KEY,
  Name VARCHAR(128),
  Firstname VARCHAR(128),
  Birthdate DATE
);
```

Dazu gibt es einiges zu Sagen. Aber wir lassen es dabei, dass SQL verlangt, dass man hinschreibt was man will.

Und dass SQL nur “eine” Abfragesprache ist, Relationen in verschiedenen Programmiersprachen aber auch anders formuliert werden können/könnten.

REFERENCES

1. https://de.wikipedia.org/wiki/Relationale_Datenbank
2. <https://de.wikipedia.org/wiki/SQL>