Aufgabe 1

Es sein eine Datenmenge mit Kundendaten. Im Vergleich von DBMS und Dateisystem ergeben sich für letzters folgende Nachteile:

- **1.** Redundanz: Doppelte Kundendaten brauchen doppelten Platz, zudem besteht bei einem Dateisystem die Gefahr, dass nur ein Eintrag verändert wird, wodurch nicht nur doppelte, sondern auch fehlerhafte Daten entstehen.
  Inkonsistenz: Stürzt das System bei der Aktualisierung von Kundenadressen ab, so könnte das Feld oder auch die Datei korrupt sein.
- **2.** Datenzugriff: Beim reinen Dateisystem kann ein bestimmter Kunde nicht einfach so gesucht werden (deklarativ), sondern es muss auch bekannt sein, wo genau die Informationen abgespeichert sind.
- **3.** Atomizität: Jede Transaktion, z.B. von Gehältern muss atomar (*atomic*) sein, also so tiefem Niveau, dass sie ohne Probleme klar definiert und wiederherstellbar ist. Ist dies nicht der Fall, wie bei einer Datei, kann die Datenbank-Konsistenz nicht garantiert werden.

Als Nachteile von DBMS kann man erstens den erhöhten Aufwand bei der Einarbeitung und beim Zugriff sehen, da z.B. eine Datei viel einfacher und schneller aufgesetzt werden kann, und auch der Zugriff viel einfach und schneller möglich ist. Zweitens bedingt eine Speicherung von Daten in einer Datenbank ein vorherige Erstellung von Relationen, bevor überhaupt ein Einfügen von Daten möglich ist. Die Tabellen/Relationen müssen geplant, verknüpft und erstellt werden.

Aufgabe 2

- **1.** Eine DDL, eine *Data Definition Language*, erlaubt es, neue Instanzen zu kreieren und Strukturen einer Datenbank festzulegen. Die DML hingegen, wie der Begriff *Data Manipulation Language* schon aussagt, erlaubt es, Daten zu abzufragen und zu manipulieren. Auch die Relationenalgebra kann in diesem Bereich aufgezählt werden.
- **2.** Ein Schema einer Datenbank kann als Struktur oder Aufbau beschrieben werden. Es können Tabellen und Relationen, Schlüssel und Domänen beschrieben werden, ohne jedoch auf den tatsächlichen Inhalt einzugehen. Eine Instanz wiederum ist eine konkret existierende Datenbank, inklusive der Daten, also der Werte, also mit einer Anzahl eingetragener Datensätze/Tupel.

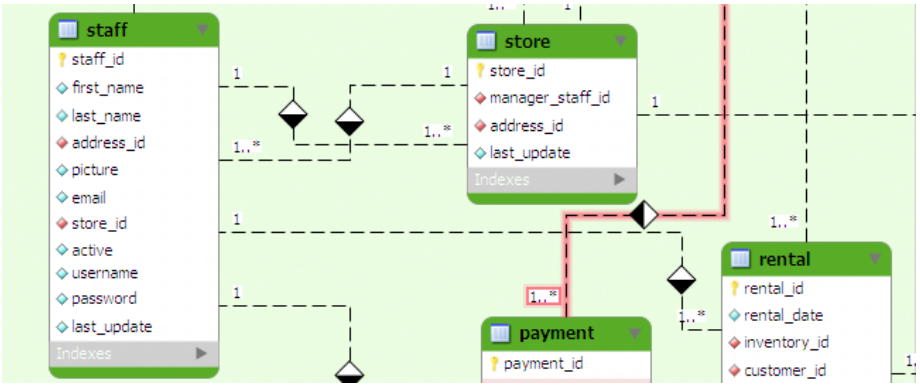

Abbildung 1: Ein Beispiel eines Datenbank-**Schemas**, als Beispiel eine MySQL-Implementation

Aufgabe 3

Das Prinzip der Konsistenz wurde hier missachtet. Heutige DBMS müssen in der Regel garantieren, dass eine Datenbank immer konsistent bleibt. Dass heisst, wenn zwischen bestimmten Operationen die Anwendung oder Architektur abstürzen sollte, muss immer ein konsistenter Zustand wiederhergestellt werden können. Dies kann der vor der Transaktion sein, oder, falls die Transaktion wiederhergestellt werden kann, der Zustand der Datenbank nach der Transaktion.

Aufgabe 4

Listing 1: Pseudocode zu Aufgabe 4

```
1  array a[0]
2  int temp
3
4  update()
5    temp <= a[0]
6    a[0] <= 0
7    a[1] <= a[1] + temp
8
9  repair()
10   int sum <= 0
11   for int i <= 0 to 99
12     sum <= sum + a[i]
13
14   if sum = 100 //--> Tabelle ist konsistent
15     return 0;
16
17   if sum < 100 //--> Zuweisung ist nicht erfolgt
18     a[1] <= a[1] + temp
19     return 0;
```

Aufgabe 5

Total aller Mengen ist die Summe des kartesischen Produkts (*cross product*), also $| A \times B |$. Dies ist in diesem Fall gleichbedeutend mit $n \cdot m$. Als Begründung sei anzuführen, dass diese Operation alle möglichen Relationen abdeckt, da jedes Element mit jedem anderen kombiniert wird.