Datenbanken

Seminararbeit

Einführung in das wissenschaftliche Arbeiten

vorgelegt von: Christian Lechner

Gutachter: Dr. Georg Moser

Studienbereich: Informatik

Datum: 6. Juni 2013

Inhaltsverzeichnis

1 Einführung in Datenbanken

In diesem Kapitel soll die Notwendigkeit von Datenbanken bzw. Datenbanksystemen aufgezeigt und deren grundlegender Aufbau erklärt werden.

1.1 Motivation

Die Vorteile eines Datenbanksystems gegenüber einem Dateisystem sind vielfältig, so ist z. B. die sequentielle Suche in Dateien bei großen Datenmengen ineffizient, möchten mehrere Benutzer die gleiche Datei editieren, kommt es zu Konflikten. Soll eine Änderung auf zwei Dateien nur gemeinsam oder gar nicht sichtbar sein z. B. bei einer Überweisung (Abbuchung und Gutschrift), dann stößt ein Dateisystem schnell an seine Grenzen. Ein Systemabsturz stellt auch ein großes Problem dar, oftmals sind die Datensicherungen mehrere Stunden bzw. Tage alt. Ein weiterer Nachteil eines Dateisystems sind die unflexiblen Zugangsbeschränkungen, ein Benutzer erhält Zugriff auf die komplette Datei oder dieser wird komplett verweigert. Um die genannten Probleme zu umgehen verwendet man ein Datenbanksystem. [Specht 2012]

1.2 Definitionen

Umgangssprachlich wird oft der Begriff *Datenbank* für ein *Datenbanksystem* (DBS) verwendet. Dabei ist eine Datenbank eine konkrete Anwendung mit konkreten Daten z. B. Flugbuchungen oder Zugauskünfte, diese läuft immer auf einem Datenbanksystem. Ein DBS stellt die Speicherstrukturen, Zugriffsoperationen, Datenverwaltung und Schutzmechanismen zur Verfügung. [Specht 2012]

1.3 Architektur eines DBS

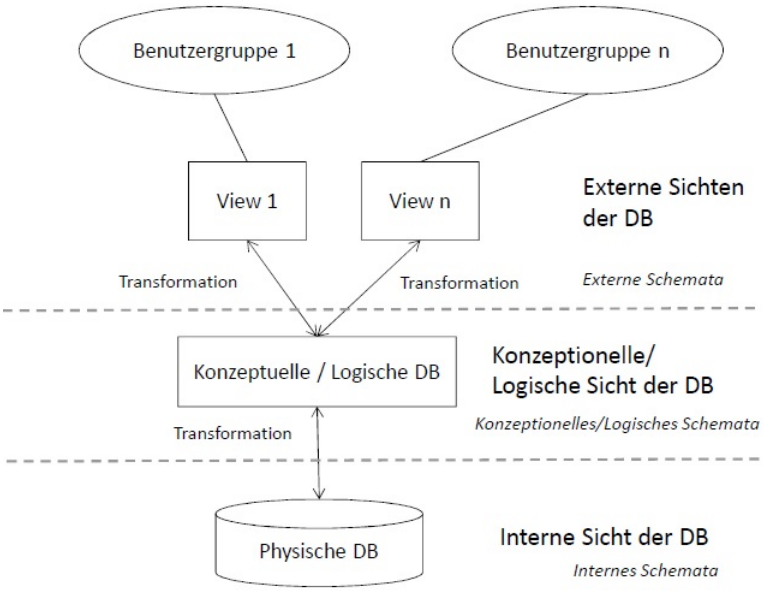

Abbildung 1.1: Architektur eines DBS

Abbildung 1.1 zeigt das 3-Schichten-Modell eines DBS, d.h. das System wird auf unterschiedlichen Abstraktionsniveaus gesehen. Die physische (interne) Schicht beinhaltet die konkrete Implementation der Datenbank als Verbund von Feldern, Strukturen, Dateien, Indizes und Hashtabellen. Die konzeptuelle (logische) Schicht modelliert den gesamten Anwendungsbereich und stellt Daten einheitlich ohne Redundanz dar. Die Implementierung dieser erfolgt mittels DDL[1]. Die View (externe) Schicht repräsentiert eine spezielle Sicht einer bestimmten Benutzergruppe auf die Daten. Datenbankabfragen und Aktualisierungen werden mittels DRL[2] und DML[3] implementiert. [Specht 2012]

1.4 Zusammenfassung

Ein DBS soll also Daten sicher und langlebig speichern und imstande sein große Datenbestände effizient zu verwalten.

[1] Data Definition Language: beschreibt Datenstrukturen einer Datenbank
[2] Data Retrieval Language: Abfrageoperationen
[3] Data Manipulation Language: Einfüge-, Aktualisierungs und Löschoperationen

2 Modellierung einer relationalen Datenbank

Dieses Kapitel soll die Modellierung eines Beispiels aus der realen Welt als relationale Datenbank aufzeigen. Eine relationale Datenbank ist eine Sammlung von Tabellen mit Attributen und Beziehungen untereinander.

2.1 E/R-Modellierung

Das Entity/Relationship-Modell (E/R-Modell) ist ein grafisches Hilfsmittel zur Modellierung der Diskurswelt, d.h. zum Entwurf einer Datenbank unabhängig vom konkreten DBS. Die Grundidee dahinter ist, dass sich die reale Welt durch Objekte und Beziehungen zwischen diesen beschreiben lässt. Abbildung 2.1 zeigt eine solche Modellierung. Entitäten werden als Rechtecke, Attribute als Kreise und Beziehungen zwischen den Entitäten werden mit einer Raute symbolisiert. [Kemper u. Eickler 2006]

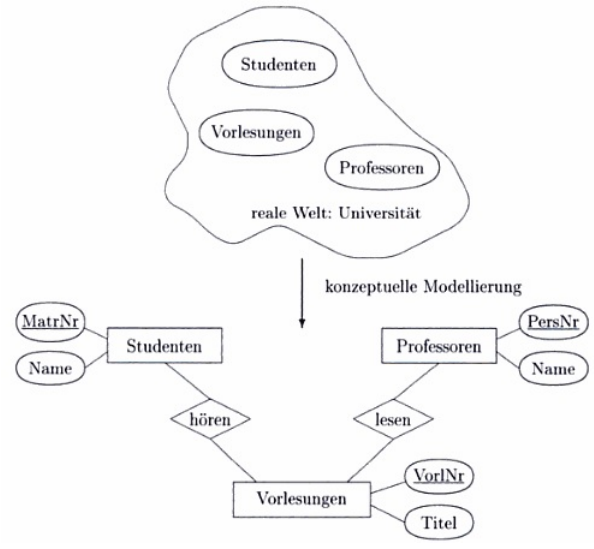

Abbildung 2.1: Konzeptuelle Modellierung einer Beispielanwendung

2.2 Abbildung auf Tabellen

Um nun das E/R-Modell in relationale Tabellen umzuwandeln führt man zuerst alle Entitäten mit ihren Attributen in eine Tabelle über. Konkret entspricht das folgenden Entitäten (Attributen): *Studenten* (*MatrNr*, *Name*), *Professoren* (*PersNr*, *Name*) und *Vorlesungen* (*VorlNr*, *Titel*). Die Beziehungen *hören* und *lesen* sind n:m Beziehungen, d.h. ein Student kann mehrere Vorlesungen besuchen und eine Vorlesung kann von mehreren Studenten besucht werden. Weitere mögliche Abbildungen sind 1:1, 1:n und n:1 Beziehungen. Tabelle 2.1 zeigt die Abbildung auf Tabellen von *Studenten* und *Vorlesungen* mit der Beziehung *hören*. Um einen Eintrag in einer Tabelle eindeutig identifizieren zu können, muss ein Primärschlüssel definiert werden (unterstrichen). [Kemper u. Eickler 2006]

Studenten	
MatrNr	Name
26120	Max Muster
26125	Martina Muster
...	...

hören	
MatrNr	VorlNr
26120	5022
26125	2033
...	...

Vorlesungen	
VorlNr	Titel
2033	Datenbanken
5022	Compilerbau
...	...

Tabelle 2.1: Relationen *Studenten*, *hören* und *Vorlesungen*

2.3 Regeln für eine gute E/R-Modellierung

Ein gut modelliertes E/R-Diagramm ist der Ausgangspunkt für eine gute Datenbank ohne Redundanzen und Konflikten. Im folgenden sind einige Punkte angeführt die unabdingbar für ein gutes E/R-Diagramm sind.

- Namenskonflikte von Attributen auflösen
- Eliminierung von redundanten Beziehungen
- mehrwertige Attribute in schwache Entitäten (können nur mit einer Beziehung bestehen) auflösen
- Vermeidung von redundanten Attributen
- Vermeidung von zusammengesetzten Schlüsseln in Entitäten [Specht 2012]

3 Abfragen auf relationale Datenbanken

Abfragen auf relationale Datenbanken werden mittels der *Structured Query Language* (SQL) ausgeführt. SQL ist eine deklarative Sprache, d.h. man beschreibt welche Daten man einfügen, aktualisieren, auslesen, etc. möchte. Die Art wie die Operationen ausgeführt werden interessiert hier nicht. Wie bereits in Kapitel 1.3 erwähnt, unterscheidet man drei Arten von SQL Befehlen: DDL zur Definition von Tabellen, DML zur Manipulation von Daten, DRL zum Auslesen von Daten. [Date u. Darwen 1998]

3.1 Beispiele für SQL Abfragen

Die folgenden Befehle beziehen sich auf das Beispiel aus Kapitel 2.

DDL-Beispiel: Anlegen der Tabelle *Studenten*: Matrikelnummer ist vom Typ *Integer* der Größe 10, Name vom Typ *String* (max. Länge 255), die Felder dürfen nicht undefiniert sein und der Schlüssel ist die Matrikelnummer.

```
CREATE TABLE Studenten (
 MatrNr int(10) NOT NULL,
 Name varchar(255) NOT NULL,
 PRIMARY KEY (MatrNr) )
```

DML-Beispiel: Daten einfügen

```
INSERT INTO Studenten VALUES ('26125', 'Martina Muster')
```

DRL-Beispiel: Daten auslesen (liefert Datensatz zu der Matrikelnummer 26125: Martina Muster)

```
SELECT * FROM Studenten WHERE MatrNr = '26125'
```

Literaturverzeichnis

[Date u. Darwen 1998] DATE, C. ; DARWEN, H.: *SQL. Der Standard.*. Bd. 3. Auflage. Addison-Wesley, 1998 3

[Kemper u. Eickler 2006] KEMPER, A. ; EICKLER, A.: *Datenbanksysteme: eine Einführung*. Bd. 8. Auflage. Oldenbourg, 2006 2.1, 2.2

[Specht 2012] SPECHT, Günther: Einführung und Überblick. In: *Skriptum zur Vorlesung Datenbanksysteme* (2012/2013). http://dbis-informatik.uibk.ac.at/files/ext/lehre/ws12-13/VO-DBS/Kapitel01.pdf 1.1, 1.2, 1.3, 2.3

Abbildungsverzeichnis

Tabellenverzeichnis