relationale Datenbanken

... ois was vermutlich zur Klausur kommt. https://github.com/eisenwinter/fh-hgb-stuff / MIT

Jan Caspar, Aktualisiert 19. Mai 2017

Grundlagen

Eine Datenbank hat die Aufgabe Daten effizient zu verwalten. Das DBS - Datenbanksystem - hat das Ziel gängige Probleme bei der Nutzung von Daten vorzubeugen. Diese Aspekete sind

Integrität Mehrfachspeicherung (Redundanz), Widersprüchlichen (Konsistenz), Aufbau

Vertraulichkeit Schutz vor unberechtigtem Zugriff (unberechtigte Personen können nicht zugreifen; bestimmte Personen dürfen nur einen Ausschnitt der Daten sehen)

Verfügbarkeit Handling von Fehlerfällen (Stromausfall, Systemcrash)

Dies stellt das DBMS - Datenbankmanagmentsystem - der Datenbank sicher. Das DBMS verwaltet die strukturierte Datenbasis (Datenbank) des DBS und ist ein fixer Bestandteil jeder Datenbank. Aus den drei Problemen ergeben sich die drei Kernaufgaben rund um Die Daten: **Integrität, Vertraulichkeit, Verfügbarkeit**.

Integrität

statische Integrität

- **Integritätsbedingungen** (Domänen-,Entitäts-, referentielle, Benutzerdef.) sichergestellt durch **Schema**
- **Widerspruchsfreiheit/Redundanzfreiheit** sichergestellt durch **Normalisierung**

dynamische Integrität

- **Ablaufintegrität** sichergestellt durch Transaktionskonzepte (Atomarität, Isolation)

Verfügbarkeit

Anwedung- und DB-System unabhängig

- Standardisierte Sprache : **SQL**
- Standardisiertes Datenmodell: **Relationenmodell**

Änderungsresistent

- **Logische** (Sichten) und **physische** Datenunabhängigkeit.

Clientunabhängig

- **APIs** (z.B. ODBC, JDBC)
- **UIs** (text-basiert, graphisch)

Performanz

- Abfragen: **Indizierung**, **Anfrageoptimierung**, **Denormalisierung**
- Einfügen: genau Seitenverkehrt

Ausfallsicher

- Recovery
- Replikation
- Durability

Skalierbar

- Datenmenge (**Big Data**), **Verteilung** (horizontal / vertikal)
- Benutzermenge (**Big Users**), **Replikation**

Vertraulichkeit

Daten

- Sichten
- DB-Verschlüsselung

Personen

- Authentifizierung
- Authorisierung

Weiterführend wird nur relationale Datenbanken behandelt. Dieses Konzept von E.F. Codd bereits 1970 entwickelt und stellt alle Daten in Tabellen da. Diese Tabellen bilden das Schema.

Datenbankentwurf

Entwurfsschritte

Phase	Modell	Ergebnis	Beispiel
1 Anforderungsanalyse	Requirements Engineering	Pflichtenheft (Use-Cases, ...)	ANF 5.4: Eine Vorlesung wird durch eine eindeutige Vorlesungsnummer identifiziert.
2 Konzeptueller Entwurf	Entity Relationship (UML, ...)	Konzeptuelles DB-Schema	Das ER Diagramm.
3 Logischer Entwurf	Datenbankmodell	Logisches DB-Schema	Vorlesung{VorlNr, Titel, ...} Student{MatrNr, Name, ...}
4 Physischer Entwurf	Data Definition Language	Physisches DB-Schema	CREATE TABLE Vorlesung(VorlNr NUMBER(10) PRIMARY KEY, Titel VARCHAR(20), ...);

ER

Begriff	Beduetung
Entity	Zu repräsentierende Informationseinheit.
Entitätstyp	Gruppierung von Entities mit gleichen Eigenschaften (= Entitätsmengen)
Beziehungstyp	Gruppierung von Beziehungen zwischen Entities
Attribut	Datenwertige Eigenschaft eines Entity oder einer Beziehung
Schlüssel	Identifizierende Eigenschaft von Entities, sind eine minimale Menge von Attributen
Kardinalitäten	Einschränkung von Beziehungstypen bezüglich der mehrfachen Teilnahme von Entities an der Beziehung
Grad/Stelligkeit	Anzahl der an einem Beziehungstyp beteiligten Entitätstypen
Funktionale Beziehung	Beziehungstyp mit Funktionseigenschaft
is-a-Beziehung	Spezialisierung von Entitätstypen
Rollen	Beschreibung wie die an einer Beziehung beteiligten Entities sich verhalten
Schwache Entities	Entities, deren Existenz von einer anderen, übergeordneten Entity abhängen und die durch eine Kombination mit dem Schlüssel der übergeordneten Entity identifizierbar sind

Relationenmodell > logischer EntwurfW

Übertragen des ER in die Tuppelform (Professor: {[PersNr:integer, Name:string, Rang:string]}). Primärschlüssel unterstreichen, Fremdschlüssel gestrichelt unterstreichen.

Normalisierung

Zweck: Vermeidung/Verringerung von **Redundanz** (Mehrfachspeicherung), um potentielle Anomalien/Inkonsistenzen zu verhindern

Zu Grunde liegendes Prinzip: **Funktionale Abhängigkeiten** in den Daten werden analysiert

Vorgehen:

- Aufteilung der Attribute (Tabellenspalten) in mehrere Relationen (Tabellen) gemäß den Normalisierungsregeln (auf Basis der enthaltenen funktionalen Abhängigkeiten), so dass eine Form entsteht, die keine vermeidbaren Redundanzen mehr enthält
- U**nterschiedliche "Güteklassen"** (je nach Grad an erlaubter/enthaltener "Restredundanz"): **1.-3. Normalform (Boyce Codd Normalform)**
- Zerlegungsalgorithmen müssen zwei Kriterien erfüllen:
  - **Abhängigkeitstreue**: alle funktionalen Abhängigkeiten die auf der ursprünglichen Relation gelten, müssen auch auf der Zerlegung noch gelten
  - **Verbundtreue**: ursprüngliche Relation muss rekonstruierbar sein

1. Normalform

Erste Normalform liegt vor, wenn jedes Attribut der Relation einen atomaren Wertebereich hat, d.h., weder mehrere Attributswerte eines Tupels zusammenfasst (Spaltenübergreifend) noch Attributswerte mehrerer Tupel zusammenfasst (Zeilenübergreifend). (**Einfach**: nur primitive Werte!)

2. Normalform

Eine Relation ist in der zweiten Normalform, wenn die erste Normalform vorliegt und kein Nichtschlüsselattribut funktional abhängig von einer echten Teilmenge eines Schlüsselkandidaten ist. (**Einfach**: kein zusammengestzter Schlüssel!)

3. Normalform

Die dritte Normalform ist erreicht, wenn sich das Relationenschema in 2NF befindet, und kein Nichtschlüsselattribut von einem anderen Nichtschlüsselattribut funktional abhängig ist (**Einfach**: nichts doppelt Eintragen, Schlüssel!)

SQL

SQL besteht aus

Datendefinitionssprache (DDL) Erstellen, Ändern und Löschen von Datenbankobjekten (Tabellen, etc.)

Datenmanipulationssprache (DML) Einfügen, Ändern und Löschen von Daten

Anfrage (Query)-Sprache (DRL) Abfragen von Daten (dies ist der komplexeste Teil von SQL)

Kontrollsprache (DCL) Verwalten von Zugriffsrechte auf Datenbankobjekte, Transaktionskontrolle, etc.

DDL Beispiel

```sql
CREATE TABLE Student (
   MatrNr INTEGER PRIMARY KEY,
   Name VARCHAR (30) NOT NULL,
   Semester INTEGER
   CONSTRAINT Check_Semester CHECK (Semester BETWEEN 1 AND 13)
);
-- SCHEMA EVOLUTION
--neue Spalten hinzugefügt
ALTER TABLE Professor ADD (Titel VARCHAR(30));
--vorhandene Spalten geändert
ALTER TABLE Professoren MODIFY(Titel VARCHAR(40));
--Spalte löschen
ALTER TABLE Professor DROP COLUMN Titel;
```

DML Beispiel

```sql
--INSERT
INSERT INTO Professor (PersNr, Name, Rang) VALUES (2140, 'Altmann', 'C4');
--DELETE
DELETE FROM Student WHERE MatrNr = 24002;
--UPDATE
UPDATE Student SET Semester = Semester + 1 WHERE MatrNr = 24002;
```

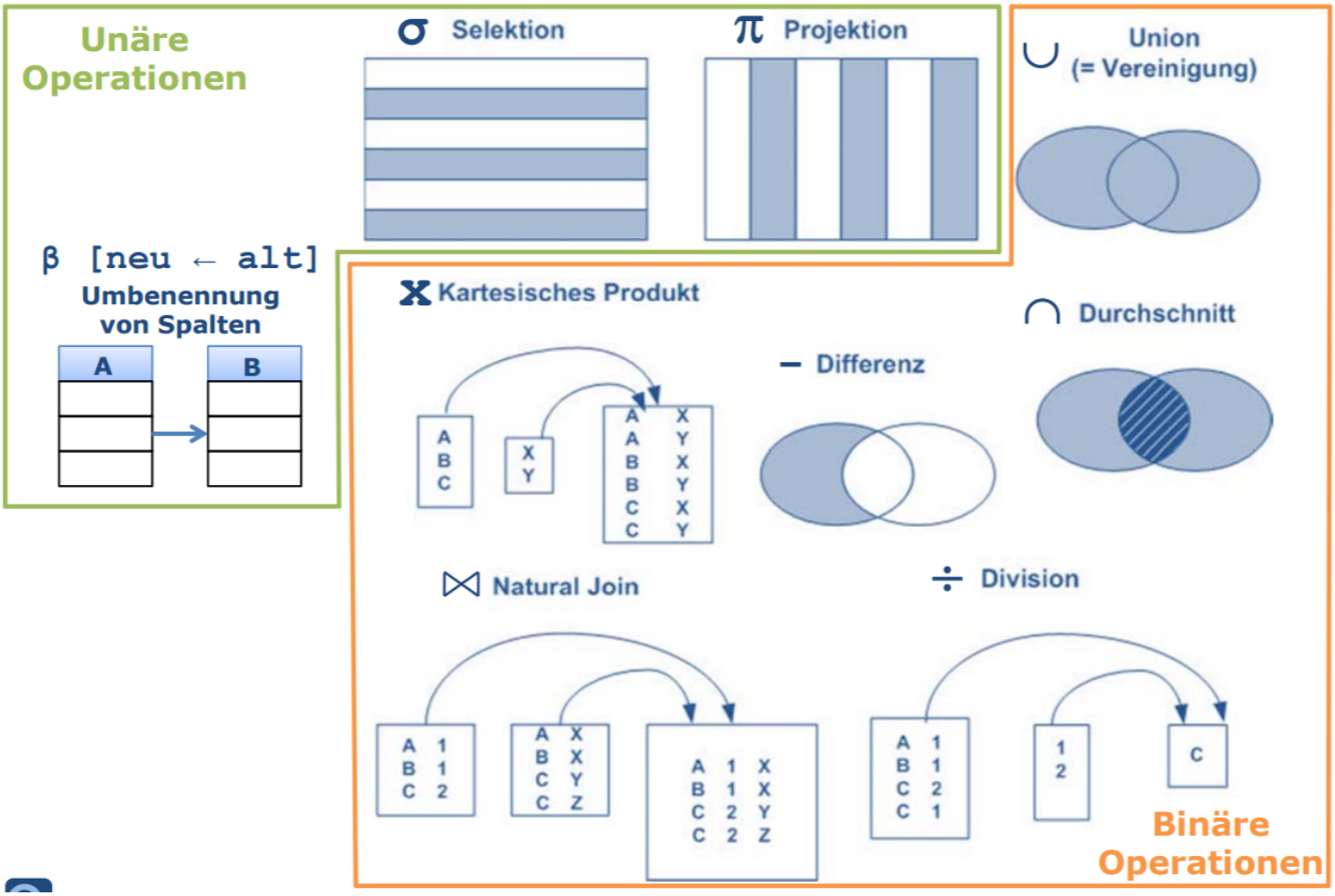
DRL Beispiel

```sql
--SELECT STATEMENTS
SELECT  * FROM departments;
```

DCL Beispiel

```sql
CREATE USER xx IDENTIFIED BY 'test'
```

Relationale Algebra



β Erlaubt die Umbenennung von Spalten einer Tabelle

π (Projektion) Erlaubt die Modifikation (Spalten entfernen, Spalten umkodieren) von Tupeln einer Tabelle

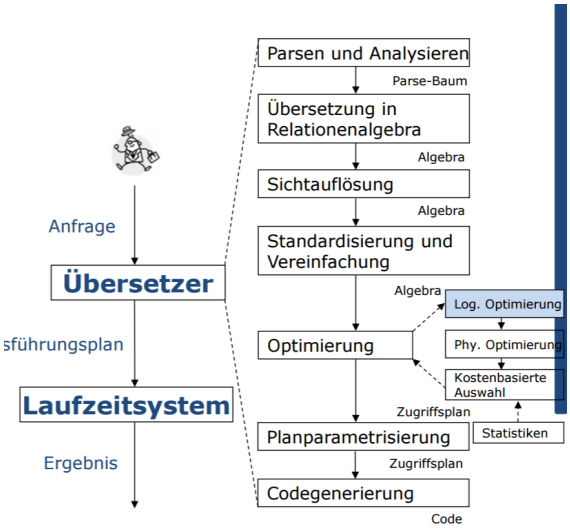
σ (Selektion) Erlaubt die Reduktion von Tupeln einer Tabelle

Mengenoperationen (Union, Differenz, Durchschnitt) Erlauben die Berechnung einer neuen Tupelmenge auf Basis von zwei Tabellen

$\times$ Karthesisches Produkt, $\bowtie$ Natural Join Erlauben die Modifikation von Tupeln auf Basis von zwei Tabellen (Tupel von verschiedenen Tabellen werden verknüpft)

Aggregation Erlaubt das Zusammenführen von mehreren Tupeln zu neuen Tupeln (Anmerkung: Aggregation gibt es in Relationenalgebra nicht!)

Abfrageabarbeitung



Definitionen

Datenbanksystem (DBS)

Ein Datenbanksystem (DBS) besteht aus einer oder mehreren Datenbanken (DB), einem Data Dictionary (DD) und einem Datenbankmanagementsystem (DBMS).

Datenbankverwaltungssystem (DBMS)

Ein Datenbankverwaltungssystem (Database Management System) bezeichnet die Gesamtheit der Programme zum Zugriff auf die im Datenbanksystem (DBS) gespeicherten Daten, zur Konsistenzkontrolle und zur Datendefinition.

Datenbank (DB)

Die in einem Datenbanksystem gespeicherten Daten werden als Datenbank bezeichnet.

Data Dictionary (DD)

Ein Data Dictionary (DD) enthält die Daten, die den Datenbestand beschreiben

Datenmodell

Ein Datenbanksystem (DBS) basiert auf einem Datenmodell. Das Datenmodell stellt die
Konzepte zur Modellierung der realen Welt zur Verfügung und besteht aus:

- Datendefinitionssprache (DDL)
- Datenmanipulationssprache (DML)
  - Abfragesprache (Query Language)
  - Befehle zum Einfügen, Löschen und Verändern der Daten

Datenbankschema (Intension)

Ein Datenbankschema legt die Struktur der Daten fest. (Metadaten: Daten über Datenbank)

Datenbankausprägung (Extension)

Die Datenbankausprägung stellt den momentan gültigen Zustand der Datenbank dar.

Redundanz

Redundanz ist das mehrfache Vorhandensein ein und derselben Informationen.

Konsistenz

Konsistenz bezeichnet in der Informatik allgemein die Widerspruchsfreiheit von Daten.

Datenintegrität

Unter Datenintegrität versteht man die logische Korrektheit, Gültigkeit und Genauigkeit von Daten in einer Datenbank.

Funktionale Abhängigkeit

Innerhalb einer Relation bestimmt eine Attributmenge den Wert einer anderen Attributmenge.

- Eine funktionale Abhängigkeit zwischen zwei Attributen A und B (bzw. Attributsmengen) liegt vor, falls
  - immer dann wenn Attribut A (z.B. SV.-Nr.) den Wert a aufweist, das Attribut B (z.B. Name oder Wohnort) den Wert b aufweist, d.h., man bei Kenntnis des Wertes des Attributes A, EINDEUTIG auf den Wert des Attributes B schließen kann, d.h., dieser damit festgelegt ist
- Daraus lässt sich schließen, dass wenn Attribut A den Wert a mehrmalig in einer Relation annehmen kann, automatisch Redundanz entsteht, da jedes Mal derselbe Wert b für das Attribut B resultiert

Partielle Abhängigkeit

Attributmenge ist funktional schon von einem Teil des Schlüssels abhängig.

Transitive Abhängigkeit

Vom Schlüssel abhängige Attributmenge bestimmt andere Attributmenge funktional

Anomalien

Anomalien bezeichnen in relationalen Datenbanken Fehlverhalten der Datenbank. Anomalien entstehen durch einen fehlerhaften oder inkorrekten Datenbankentwurf.

Löschanomalie

Bei der Löschanomalie kann es passieren, dass man durch das bewusste Löschen eines Datensatzes, unbewusst Informationen verliert, die man später wieder gebraucht hätte.

Einfügeanomalie

Liegt ein fehlerhaftes Datenbankdesign vor, kann es bei der Einfügeanomalie passieren das Daten gar nicht gespeichert/angenommen werden, wenn beispielsweise für den Primärschlüssel kein Wert eingegeben wird, oder es führt bei einer nicht vollständigen Eingabe von Daten zu Inkonsistenz.

Änderungsanomalie

Bei der Änderungsanomalie werden gleiche Attribute eines Datensatzes nicht automatisch geändert. So entsteht eine Inkonsistenz der Daten. Man muss per Hand alle Einträge aktualisieren, es darf kein Fehler unterlaufen ansonsten führt es zur Inkonsistenz.