Universität Leipzig, Institut für Informatik
Abteilung Datenbanken
Prof. Dr. E. Rahm, Dr. D. Sosna, Dr. A. Thor, D. Aumüller

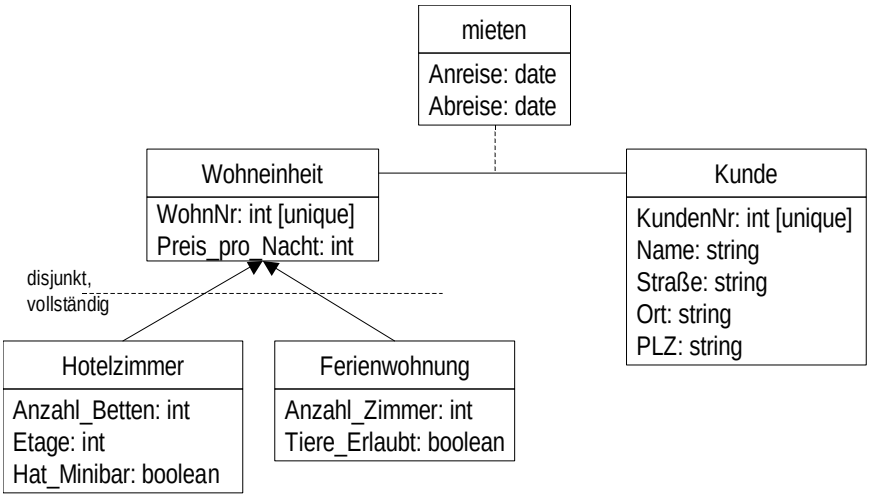
Besprechung ab 08.06.2009

Datenbanksysteme II

SS 2009 – Übungsblatt 4

Der Besitzer einer Ferienanlage möchte die Informationen, wann welche Kunden die von ihm angebotenen Wohneinheiten belegen, in einer relationalen Datenbank speichern und mit einem Java-Programm darauf zugreifen.

Dazu hat er folgendes UML-Diagramm erstellt:



1. Aufgabe: Konvertierung UML-RM

Erstellen Sie aus dem UML-Diagramm ein relationales Modell. Benennen Sie die Relationen entsprechend nach den UML-Klassennamen. Verwenden Sie für die Spezialisierung die vertikale Partitionierung.

2. Aufgabe: Datendefinition mit SQL, Löschregeln

Geben Sie die SQL-Anweisung zur Definition eines Datenbankschemas gemäß des relationalen Modells an. Dabei sollen beim Löschen einer Wohneinheit auch alle Informationen, welche Kunden sie gemietet haben, entfernt werden.

3. Aufgabe: Integritätsbedingungen, Trigger

Definieren Sie mittels SQL einen Trigger zur Wahrung der Integritätsbedingung, dass zu keiner Zeit einr Doppelbelegung vorliegt. Ein Einfügen (INSERT) eines neuen Datensatzes in die Relation *mieten* ist daher nur zulässig, wenn kein anderer Datensatz für die Wohneinheit

vorliegt, dessen An- oder Abreisetag zwischen An- und Abreisetag des neuen Datensatzes liegen.

4. Aufgabe: SQL-Änderungsoperationen

Geben Sie die notwendigen SQL-Änderungsoperationen (INSERT) an, um folgenden Sachverhalt in einer leeren Datenbank zu speichern:

Der Kunde (*Nr* = 7) Hans Müller (*Name*) aus Hauptstraße 5 (*Straße*) in 04109 Leipzig (*PLZ* und *Ort*) mietet vom 01.07.2009 (*Anreise*) bis 15.07.2009 (*Abreise*) die Ferienwohnung mit der *WohnNr* = 9. Die Wohnung kostet 50€ pro Nacht, hat 3 Zimmer und Tiere sind nicht erlaubt.

Welche Beschränkungen hinsichtlich der Reihenfolge der SQL-Operationen gibt es?

5. Aufgabe: SQL-Anfragen

Formulieren Sie eine Anfrage mittels SQL, die für jede Wohneinheitsnummer die Anzahl der Kunden angibt, welche die Wohneinheit gemietet haben. Sortieren Sie das Ergebnis absteigend nach der Anzahl der Vermietungen. Stellen Sie sicher, dass auch Wohneinheiten, die bisher nicht vermietet wurden, im Ergebnis auftreten (mit Vermietungsanzahl = 0).

6. Aufgabe: JDBC

Skizzieren Sie eine Java-Funktion unter Verwendung von JDBC, die nach Eingabe einer Wohneinheitsnummer alle zugehörigen Informationen (inklusive der Nummern der Kunden, welche die Wohneinheit gemietet haben) ausgibt. Diskutieren Sie, welchen Einfluss die Wahl der Partitionierung in Aufgabe 1 auf die Java-Funktion hat.

7. Aufgabe: SQLJ

Skizzieren Sie eine Java-Funktion unter Verwendung von SQLJ, die nach Eingabe einer Kundennummer alle zugehörigen Informationen zum Kunden (ohne Vermietungen) ausgibt.

8. Aufgabe: Normalisierung

Konvertieren Sie das Datenbankschema in die 3. Normalform vor dem Hintergrund, dass aus der Postleitzahl (PLZ) eines Kunden der Ort bereits eindeutig bestimmt ist. Welche Vor- und Nachteile ergeben sich aus der durchgeführten Schematransformation?

9. Aufgabe: View-Definition

Definieren Sie auf dem normalisierten Schema von Aufgabe 8 eine View *Kunde_unnormalisiert*, deren Struktur und Inhalt der ursprünglichen Relation *Kunde* aus Aufgabe 1 entspricht.

10. Aufgabe: Schemaevolution

Welche Änderungen müssen am Programm von Aufgabe 7 gemacht werden, damit es mit dem normalisierten Schema arbeiten kann. Inwieweit kann die in Aufgabe 9 definierte View dazu verwendet werden.