

# Datenbanksysteme



herausgegeben von der Fachschaft Mathematik/Informatik, umbrochen von Sabine

## Aufgabe 1 – Multiple-Choice und ER-Modellierung (15 Punkte)

- a) Geben Sie für folgende 24 Aussagen an, ob die Aussage korrekt oder falsch ist.

*Hinweis: Für jede korrekte Antwort erhalten Sie  $\frac{1}{3}$  Punkt, für jede falsche Antwort wird Ihnen  $\frac{1}{3}$  Punkt abgezogen. Die Aufgabe wird mit mindestens null Punkten bewertet.*

	RICHTIG	FALSCH
Das Cursor-Konzept ist sowohl Bestandteil von JDBC als auch von Embedded SQL.	<input type="checkbox"/>	<input type="checkbox"/>
Mit JDBC werden Fehler im SQL-Statement stets erst zur Laufzeit erkannt.	<input type="checkbox"/>	<input type="checkbox"/>
JDBC standardisiert unter anderem, wie man durch das Anfrageergebnis navigiert.	<input type="checkbox"/>	<input type="checkbox"/>
JDBC hat die Eigenschaft, dass auf dem Client kein DBMS-spezifischer Code laufen muss.	<input type="checkbox"/>	<input type="checkbox"/>
JDBC bietet die Möglichkeit, auf Schema-Informationen zu einem Anfrageergebnis zuzugreifen.	<input type="checkbox"/>	<input type="checkbox"/>
Embedded SQL bietet die Möglichkeit, Syntaxfehler im SQL-Statement zum Übersetzungszeitpunkt zu erkennen.	<input type="checkbox"/>	<input type="checkbox"/>
Embedded SQL bietet die Möglichkeit, Schema-spezifische Fehler im SQL-Statement zum Übersetzungszeitpunkt zu erkennen.	<input type="checkbox"/>	<input type="checkbox"/>
Embedded SQL bietet die Möglichkeit, zum Übersetzungszeitpunkt zu erkennen, dass der Typ einer Variablen in der Host-Sprache und der eines Attributs der Relation nicht übereinstimmen.	<input type="checkbox"/>	<input type="checkbox"/>
Gespeicherte Prozeduren bieten die Möglichkeit zu erkennen, dass der Typ einer Variablen in einer Prozedur und der eines Attributs der Relation nicht übereinstimmen.	<input type="checkbox"/>	<input type="checkbox"/>
Ein Vorübersetzer für Embedded SQL übersetzt die eingebetteten Statements nach SQL.	<input type="checkbox"/>	<input type="checkbox"/>
Ein Vorübersetzer für Embedded SQL übersetzt die eingebetteten Statements in Stored-Procedure Aufrufe.	<input type="checkbox"/>	<input type="checkbox"/>
Wenn man mit einer JPA (Java Persistence API) arbeitet, bleibt dem Anwendungsentwickler das Formulieren deklarativer Anfragen (in SQL bzw. SQL-artigen Sprachen) erspart.	<input type="checkbox"/>	<input type="checkbox"/>
Gespeicherte Prozeduren können die üblichen aus imperativen Programmiersprachen bekannten Kontrollstrukturen enthalten.	<input type="checkbox"/>	<input type="checkbox"/>

	RICHTIG	FALSCH
Gespeicherte Prozeduren können SQL-Anfragen enthalten.	<input type="checkbox"/>	<input type="checkbox"/>
Herkömmliche eindimensionale Indexstrukturen können die Auswertung mehrdimensionaler Bereichsanfragen beschleunigen.	<input type="checkbox"/>	<input type="checkbox"/>
Der kd-Baum ist nicht balanciert.	<input type="checkbox"/>	<input type="checkbox"/>
Die Ausschnitte des Datenraums, die Geschwisterknoten im kd-Baum entsprechen, sind immer überlappungsfrei.	<input type="checkbox"/>	<input type="checkbox"/>
Das (in der Vorlesung vorgestellte) optimale Verfahren zum Finden des nächsten Nachbarn mit Hilfe räumlicher Indexstrukturen ist ein Beispiel für Tiefensuche.	<input type="checkbox"/>	<input type="checkbox"/>
Das (in der Vorlesung vorgestellte) optimale Verfahren zum Finden des nächsten Nachbarn mit Hilfe räumlicher Indexstrukturen ist ein Beispiel für Breitensuche.	<input type="checkbox"/>	<input type="checkbox"/>
Das Ergebnis des DBSCAN-Algorithmus ist unabhängig von der Reihenfolge, in der die Objekte betrachtet werden.	<input type="checkbox"/>	<input type="checkbox"/>
Die Komplexität von DBSCAN ist logarithmisch in der Anzahl der Datenobjekte.	<input type="checkbox"/>	<input type="checkbox"/>
Die Zahl der Cluster, die der DBSCAN-Algorithmus zurückliefert, ist ein Parameter dieses Algorithmus.	<input type="checkbox"/>	<input type="checkbox"/>
In hochdimensionalen Merkmalsräumen wächst der erwartete Abstand zweier Datenobjekte (jedes für sich zufällig gewählt, Datenobjekte sind im Raum gleichverteilt) mit der Anzahl der Datenobjekte.	<input type="checkbox"/>	<input type="checkbox"/>
In hochdimensionalen Merkmalsräumen wächst die Anzahl der Blätter einer räumlichen Indexstruktur mit der Anzahl der Dimensionen.	<input type="checkbox"/>	<input type="checkbox"/>

(8 Punkte)

b) Modellieren Sie ein ER-Diagramm für eine Autovermietung mit Reparaturwerkstätten nach folgenden Vorgaben:

- Ein Kunde mietet ein Auto indem er einen Mietvertrag abschließt. Die Adresse des Kunden wird als strukturiertes Attribut festgehalten: Straße, Hausnummer, PLZ, Stadt. Ein Kunde kann beliebig viele Autos mieten. Weiterhin können Autos auch von verschiedenen Kunden gemietet werden. Optional kann zu einer Vermietung ein Navigationsgerät gebucht werden.
- Jeder Mietvertrag gilt für einen Mietzeitraum. Zu jedem Mietvertrag muss mindestens eine Versicherung abgeschlossen werden, vier weitere Versicherungen (z. B. Vollkasko) sind möglich. Zu jeder abgeschlossenen Versicherung wird eine Selbstbeteiligung vereinbart.
- Bei Beschädigungen eines Autos wird ein Reparaturauftrag an eine Werkstatt erteilt. In einer Werkstatt arbeiten beliebig viele KFZ-Mechaniker (aber mindestens einer). Jede Werkstatt hat verschiedene Ersatzteiltypen vorrätig.

Verwenden Sie zur Modellierung die Standardkardinalität.

(7 Punkte)

## Aufgabe 2 – Rel. Algebra/Funktionale Abhängigkeiten (15 Punkte)

Gegeben sei die Definition des Verbund-Operators (Join:  $\bowtie$ ) aus der Vorlesung.

$$r_1 \bowtie r_2 := \{ t \mid t(R_1 \cup R_2) \wedge [\forall i \in \{1, 2\} \exists t_i \in r_i : t_i = t(R_i)] \}$$

$r_1, r_2$  sind hierbei Relationen mit den Attributmengen  $R_1$  bzw.  $R_2$ .  $t$  stellt ein Tupel der Relation dar.  $t(R)$  ist ein Tupel eingeschränkt auf die Attributmenge  $R$ . Gegeben seien folgende Relationen  $r_1$  und  $r_2$  mit den Attributmengen  $R_1 = \{A, B\}$  und  $R_2 = \{B', C\}$ .

	A	B
$r_1:$	$a_1$	$b_2$
	$a_2$	$b_1$

	B'	C
$r_2:$	$b_1$	$c_2$
	$b_2$	$c_1$
	$b_3$	$c_2$

Die Wertebereiche der Attribute sind gegeben als:

$$\text{dom}(A) = \{a_1, a_2\}, \text{dom}(B) = \{b_1, b_2, b_3\}, \text{dom}(B') = \{b_1, b_2, b_3\}, \text{dom}(C) = \{c_1, c_2, c_3\}.$$

a) Bestimmen Sie folgende Relationen:

- $r_1 \bowtie r_2$
- $r_1 \bowtie \beta_{B \leftarrow B'} r_2$
- $r_3 := \{t \mid t(R_2)\}$

(3 Punkte)

b) Zeigen Sie folgende Eigenschaften für den Verbund-Operator  $\bowtie$ :

- $\bowtie$  ist kommutativ
- $\bowtie$  ist assoziativ

(2 Punkte)

c) Zeigen Sie, dass die Menge  $\Omega' := \{\pi, \sigma, \bowtie, \beta, \cup, \cap, -\}$  nicht unabhängig ist.

(3 Punkte)

d) Gegeben sei die Relation  $R(A, B, C, D, E)$  mit der folgenden Menge an funktionalen Abhängigkeiten:

$$\begin{aligned} F = & \{ A \rightarrow E \\ & AC \rightarrow D, \\ & D \rightarrow BE, \\ & E \rightarrow B, \\ & C \rightarrow BE \} \end{aligned}$$

Der einzige Schlüssel von  $R$  ist  $\{AC\}$  und die höchste Normalform von  $R$  ist die 1NF. Geben Sie eine minimale, verbundtreue, abhängigkeitserhaltende Zerlegung von  $R$  an, welche in dritter Normalform ist. Benutzen Sie für das Ergebnis Ihrer Zerlegung die Notation:

$$S = \{(R_i, K_i), \dots, (R_n, K_n)\}$$

wobei  $R_i$  für ein Relationenschema und  $K_i$  für die entsprechende Schlüsselmenge stehen. Verwenden Sie zur Berechnung das aus der Vorlesung bekannte Syntheseverfahren. Geben Sie für jede Umformung das Zwischenergebnis und abschließend die Zerlegung von  $R$  an, die sich aus der Anwendung des Syntheseverfahrens ergibt.

(4 Punkte)

e) Gegeben sei die Relation:

*Institut(MitarName, MitarbeiterID, Adresse, ProjektName, ProjektID, ProjektManager)*

Die Relation enthält Informationen über Angestellte und Projekte, an denen die Angestellten arbeiten. Es gelten die folgenden funktionalen Abhängigkeiten:

$$F = \{ \text{MitarID} \rightarrow \text{MitarName Adresse}, \\ \text{ProjektID} \rightarrow \text{ProjektName ProjektManager} \}$$

Geben Sie anhand der Relation *Institut* jeweils ein Beispiel für jede der drei in der Vorlesung vorgestellten Anomalien an.

(3 Punkte)

### Aufgabe 3 – SQL (15 Punkte)

Um vom Gewerbeamt eine Verlängerung der Betriebserlaubnis zu erhalten, hat der Geschäftsführer des italienischen Restaurants 'Ciao Bello' seine Datenhaltung auf ein SQL-basiertes Datenbanksystem umgestellt. Dort seien Pizzen, Zutaten und deren Zusammensetzung durch folgende Relationen modelliert:

```
CREATE TABLE Pizza
(
    id NUMBER NOT NULL,
    name VARCHAR2(20) NOT NULL,
    CONSTRAINT pizza_pk PRIMARY KEY (id)
);

CREATE TABLE Zutat
(
    id NUMBER NOT NULL,
    name VARCHAR2(20) NOT NULL,
    kosten NUMBER NOT NULL,
    vorrat NUMBER NOT NULL,
    CONSTRAINT zutat_pk PRIMARY KEY (id),
);

CREATE TABLE Pizzazutat
(
    p_id NUMBER NOT NULL,
    z_id NUMBER NOT NULL
);
```

**Hinweise:** kosten bezeichnet den Einkaufspreis in Euro für eine Zutat, vorrat deren im Restaurant gelagerte Menge.

Die Inhalte der Relationen sind wie folgt:

Pizza	
id	name
1	'Tonno'
2	'Prosciutto'
3	'Prosciutto e Funghi'

Zutat			
id	name	kosten	vorrat
1	'Tomaten'	0.5	100
2	'Käse'	1	100
3	'Thunfisch'	3	10
4	'Schinken'	2	20
5	'Pilze'	2	10

Pizzazutat	
p_id	z_id
1	1
1	2
1	3
2	1
2	2
2	4
3	1
3	2
3	4
3	5
1	1
182	0

Alle Aufgaben lassen sich mit SQL-Konstrukten lösen, die in der Vorlesung vorgestellt wurden. Die Lösungen müssen dem SQL-Standard folgen und unabhängig vom Datenbankinhalt sein.

- a) Beim Erstellen der Relation 'Pizzazutat' war der ausführende Kellner in einer Stresssituation und hat leider zu viele Tupel eingetragen. Es sollen folgende Bedingungen erfüllt sein: 1) Jeder Eintrag von Pizza und Zutat ist eindeutig. 2) Es gibt nur Einträge von Pizzen bzw. Zutaten, die auch in den Relationen 'Pizza' bzw. 'Zutat' erfasst sind.

Formulieren Sie eine Folge von SQL-Anweisungen, die zunächst die aktuelle Datenbank entsprechend bereinigt und anschließend bewirkt, dass Fehler dieser Art nicht erneut geschehen können. Hinweis: Sie dürfen im Rahmen dieser Teilaufgabe explizite IDs verwenden.

(2,5 Punkte)

- b) Der Geschäftsführer fragt sich, welche Zutat *außer* Tomaten und Käse in den meisten Pizzen auf der Speisekarte vorkommt. Formulieren Sie eine SQL-Anfrage, die den Namen dieser Zutat(en) ausgibt.

**Hinweis:** Es kann sich um mehr als eine Zutat handeln.

*Ergebnis bei gegebenem Datenbankinhalt:*  
(‘Schinken’).

(3 Punkte)

- c) Der Geschäftsführer möchte die Preise für die Speisekarte automatisch kalkulieren. Erstellen Sie eine Sicht namens 'Pizzapreis', die den Namen und den Preis jeder Pizza ausgibt. Der Verkaufspreis berechnet sich als Summe der Kosten aller Zutaten, zuzüglich 2 Euro für Teig und Zubereitungsaufwand. Vergessen Sie nicht, zuletzt 19% Mehrwertsteuer aufzuschlagen, um unnötigen Ärger mit dem Finanzamt zu vermeiden.

**Hinweis:** Auf das Runden des entstehenden Preises dürfen Sie verzichten.

*Ergebnis von SELECT name, ROUND(preis, 2) FROM Pizzapreis bei Datenbankinhalt nach Teilaufgabe a):*

(‘Tonno’, 7.74), (‘Prosciutto’, 6.55), (‘Prosciutto e Funghi’, 8.93).

(2 Punkte)

- d) Der Geschäftsführer hat Beschwerden aufgrund immer gleicher Mittagsangebote erhalten. Er möchte allerdings keine anderen Zutaten einkaufen, um Ärger mit dem Gesundheitsamt wegen zusätzlicher Lagerungsvorschriften zu vermeiden. Formulieren Sie eine SQL-Anweisung, die alle Tripel von verschiedenen Zutaten zurückgibt, die noch nicht zusammen auf einer existierenden Pizza angeboten werden und deren Kosten zusammen weniger als 6 Euro betragen. Vermeiden Sie Duplikate.

**Hinweis:** Auch Pizzen ohne Tomaten oder Käse sind im Rahmen dieser Aufgabe zulässig.

*Ergebnis bei Datenbankinhalt nach Teilaufgabe a):*

(‘Tomaten’, ‘Thunfisch’, ‘Schinken’), (‘Tomaten’, ‘Thunfisch’, ‘Pilze’)

(3,5 Punkte)

- e) Der Geschäftsführer wünscht sich ein automatisches Bestellannahmesystem. In diesem Sinne beginnt er eine PL/SQL-Prozedur namens 'bestellePizza' zu schreiben, mit der die Bestellung einer beliebigen Menge n einer Pizza namens x abgewickelt werden soll. Dazu soll zunächst in der Variablen verfuegbar berechnet werden, wie viele Pizzen der gewünschten Art momentan hergestellt werden können. Falls die gewünschte Menge niedriger als die verfügbare Menge ist, sollen anschließend die vorrätigen Mengen in der Relation 'Zutat' entsprechend der benötigten Mengen für die bestellten Pizzen verringert werden. Ergänzen Sie die vorgegebene PL/SQL-Prozedur, damit sie das Gewünschte leistet.

(3 Punkte)

Hinweis: Die folgende Lösungsschablone dient nur als Illustration. Bitte verwenden Sie die Lösungsschablone auf den Lösungsblättern.

```
CREATE OR REPLACE PROCEDURE bestellePizza(x VARCHAR2, n NUMBER) IS
    verfuegbar NUMBER;
BEGIN
    --berechne, wie viele Pizzen des Typs x hergestellt werden können
    SELECT           INTO verfuegbar

    IF n <= verfuegbar THEN
        --passte vorrätige Mengen an

    ELSE
        DBMS_OUTPUT.PUT_LINE('Mi dispiace!');
    END IF;
END;
```

- f) Eine partielle Belegung von Pizzen ist im aktuellen Schema nicht vorgesehen. Um einen Wunsch des Stammgasts Martin zu befriedigen, möchte der Geschäftsführer allerdings eine Pizza 'Quattro Martini' anbieten. Die Pizza soll vollständig mit Tomaten und Käse belegt sein, zusätzlich allerdings zu je einem Viertel mit Thunfisch, Schinken, Pilzen und Pommes. Erweitern Sie das Schema so, dass es möglich wäre, solche Pizzen mit unterschiedlichen Mengen von Zutaten anzulegen. Für die schon bestehenden Pizzen soll die Menge jeder Zutat 1 betragen.

Hinweise: Die neue Pizza 'Quattro Martini' mit ihren Zutaten müssen Sie nicht explizit anlegen. Lösungen der vorherigen Teilaufgaben müssen Sie nicht entsprechend anpassen.

(1 Punkt)

## Aufgabe 4 – Histories (15 Punkte)

Gegeben seien die folgenden Transaktionen:

$$T_1 = r_1[y] r_1[x] r_1[y] r_1[z] c_1$$

$$T_2 = w_2[z] r_2[x] w_2[z] c_2$$

$$T_3 = r_3[z] w_3[z] w_3[y] r_3[z] c_3$$

Basierend auf diesen Transaktionen seien folgende Histories definiert:

$$H_1 = r_1[y] \quad w_2[z] \quad r_2[x] \quad r_3[z] \quad r_1[x] \quad w_2[z] \quad c_2 \quad w_3[z] \quad w_3[y] \quad r_3[z] \quad r_1[y] \quad c_3 \quad r_1[z] \quad c_1$$

$$H_2 = \begin{matrix} r_3[z] & w_2[z] & r_1[y] & w_3[z] & w_3[y] & r_3[z] & r_2[x] & c_3 & w_2[z] & r_1[x] & c_2 & r_1[y] & r_1[z] & c_1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \end{matrix}$$

Hinweis: Die Zahlen unter den Operationen von  $H_2$  entsprechen der Indizierung, die für Aufgaben Teil b) verwendet werden soll.

- a) Konstruieren Sie je für  $H_1$  und  $H_2$  die Serialisierbarkeitsgraphen  $SG(H_1)$  und  $SG(H_2)$ . Begründen Sie für jede History, ob sie serialisierbar ist.

(5 Punkte)

- b) Welches Paar von Operationen muss in  $H_2$  vertauscht werden, damit sich der Serialisierbarkeitsgraph von  $H_2$  ändert ohne die Transaktionen zu verändern? Mit anderen Worten: Suchen Sie die History  $H'_2$  die dadurch entsteht, dass zwei Operationen in  $H_2$  vertauscht werden und folgende Eigenschaften hat:

- Die Transaktionen in  $H'_2$  sollen identisch sein zu  $H_2$ , d. h. die Operationsreihenfolge innerhalb der Transaktionen muss erhalten bleiben. Als Beispiel: Für die Operationen von  $T_1$  in  $H'_2$  muss gelten:  $r_1[y]$  vor  $r_1[x]$  vor  $r_1[y]$  etc.
- Der Serialisierbarkeitsgraph  $SG(H'_2)$  von  $H'_2$  soll nicht identisch sein zu  $SG(H_2)$ .

Verwenden Sie die in der Aufgabenstellung gegebene Indizierung der Operationen, um das zu vertauschende Paar anzugeben. Zur Klarstellung: Vertauschen ist nicht identisch mit Verschieben. Beispielsweise ist nach der Vertauschung von Operation 1 mit 14 das  $c_1$  an erster Stelle und  $r_3[z]$  an Position 14.

(5 Punkte)

- c) Analysieren Sie die maximalen Rücksetzbarkeitsklassen von  $H_1$  und  $H_2$ .

(5 Punkte)

