

Datenbanksysteme

herausgegeben von der Fachschaft Mathematik/Informatik, umbrochen von Sabine

Aufgabe 1 – Multiple-Choice (8 Punkte)

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 strenge Zweiphasen-Sperrprotokoll (Strict 2PL) stellt Serialisierbarkeit sicher.	<input type="checkbox"/>	<input type="checkbox"/>
Das strenge Zweiphasen-Sperrprotokoll stellt sicher, dass keine Lost Updates auftreten.	<input type="checkbox"/>	<input type="checkbox"/>
Das strenge Zweiphasen-Sperrprotokoll stellt sicher, dass keine Dirty Reads auftreten.	<input type="checkbox"/>	<input type="checkbox"/>
Das strenge Zweiphasen-Sperrprotokoll stellt sicher, dass keine Non-Repeatable Reads auftreten.	<input type="checkbox"/>	<input type="checkbox"/>
Das Zweiphasen-Sperrprotokoll ist im verteilten Fall nicht anwendbar, weil die Anordnung der Transaktionen auf unterschiedlichen Knoten im Allgemeinen unterschiedlich sein kann.	<input type="checkbox"/>	<input type="checkbox"/>
Das CAP Theorem besagt, dass hohe Verfügbarkeit und Konsistenz des Datenbestands i. Allg. unvereinbar sind, wenn Netzwerkpartitionen möglich sind.	<input type="checkbox"/>	<input type="checkbox"/>
'Netzpartition' bedeutet, dass mindestens ein Knoten nicht verfügbar ist.	<input type="checkbox"/>	<input type="checkbox"/>
'Eventual Consistency' bedeutet, dass irgendwann alle Lesezugriffe auf ein Datenobjekt den gleichen Wert zurückliefern werden, wenn ab einem gewissen Zeitpunkt keine Updates mehr stattfinden.	<input type="checkbox"/>	<input type="checkbox"/>
'Liveness' bedeutet, dass irgendwann in der Zukunft eine bestimmte Eigenschaft erfüllt sein muss.	<input type="checkbox"/>	<input type="checkbox"/>
Relationale Datenbanktechnologie kommt zur Anwendung, um Datenbasiskonsistenz sicherzustellen.	<input type="checkbox"/>	<input type="checkbox"/>
Der Sorted Merge Join ist ein Operator der physischen Ebene.	<input type="checkbox"/>	<input type="checkbox"/>
Gegeben sind zwei Relationen R1 und R2. Das Attribut A kommt in beiden Relationen vor, weitere gemeinsame Attribute dieser beiden Relationen gibt es aber nicht. Angenommen, alle Tupel in R1 und R2 haben den gleichen Wert für dieses Attribut. Dann ist die Komplexität des Nested Loop Joins O(n1 * n2). (n1 – Größe von R1, n2 – Größe von R2).	<input type="checkbox"/>	<input type="checkbox"/>
Gegeben sind zwei Relationen R1 und R2. Das Attribut A kommt in beiden Relationen vor, weitere gemeinsame Attribute dieser beiden Relationen gibt es aber nicht. Angenommen, alle Tupel in R1 und R2 haben den gleichen Wert für dieses Attribut. Dann ist die Komplexität des Sorted Merge Joins O(n1 * n2). (n1 – Größe von R1, n2 – Größe von R2).	<input type="checkbox"/>	<input type="checkbox"/>

	RICHTIG	FALSCH
Join gehört zu den blockierenden Operatoren.	<input type="checkbox"/>	<input type="checkbox"/>
Equi-Depth Histogramme haben die Eigenschaft, dass die Anzahl der Datenobjekte pro Bucket für alle Buckets (annähernd) gleich ist.	<input type="checkbox"/>	<input type="checkbox"/>
Wenn das Schreibquorum (W) klein ist, wird das Schreiben tendenziell weniger aufwändig und dadurch schneller.	<input type="checkbox"/>	<input type="checkbox"/>
Ein Schreibquorum besteht aus mindestens zwei Knoten.	<input type="checkbox"/>	<input type="checkbox"/>
Bei Chord ist die Anzahl der Knoten unabhängig von der Größe des Schlüsselraums.	<input type="checkbox"/>	<input type="checkbox"/>
Die Chord Finger Table bewirkt, dass der Suchaufwand logarithmisch mit der Anzahl der verwalteten Datenobjekte wächst (und nicht mehr linear).	<input type="checkbox"/>	<input type="checkbox"/>
Die Anzahl der Chord-Knoten, die Replikate eines Datenobjekts speichern, ist frei wählbar.	<input type="checkbox"/>	<input type="checkbox"/>
Dass eine Hash-Funktion Anwendungsschlüssel auf (Chord-)Systemschlüssel abbildet, führt zu einem reduzierten Suchaufwand (logarithmisch anstelle von linear).	<input type="checkbox"/>	<input type="checkbox"/>
Die Implementierung der 'Add to Cart'-Operation erfordert sowohl lesenden als auch schreibenden Zugriff.	<input type="checkbox"/>	<input type="checkbox"/>
Eine sinnvolle Möglichkeit, unterschiedliche Cart-Versionen im Fall einer Inkonsistenz zusammenzuführen, ist die Vereinigung.	<input type="checkbox"/>	<input type="checkbox"/>
Die Größe des Chord-Schlüsselraums ist proportional zur Anzahl der Chord-Knoten.	<input type="checkbox"/>	<input type="checkbox"/>

Aufgabe 2 – ER-Modellierung (10 Punkte)

Das folgenden Szenario beschreibt Fußballturniere. Modellieren Sie ein ER-Diagramm, das die folgenden Beziehungen darstellt:

- Eine Mannschaft besteht aus 23 Spielern (inkl. Ersatzspieler).
- Eine Mannschaft kommt aus einem bestimmten Land.
- Bei einem Spiel gibt es eine Gast- und eine Heimmannschaft.
- Ein Spiel wird an einem bestimmten Datum in einem bestimmten Stadion ausgetragen. Dabei ist es möglich, dass bis zu drei Spiele pro Tag in dem gleichen Stadion stattfinden.
- In einem Spiel kann ein Schiedsrichter maximal eine Verwarnung (Gelbe Karte) an Spieler vergeben.
- Ein Schiedsrichter kommt aus einem bestimmten Land.

Verwenden Sie Standardkardinalitäten.

Hinweis: Es ist nicht gefordert, dass Sie Attribute im ER-Modell darstellen.

Aufgabe 3 – Funktionale Abhängigkeiten (14 Punkte)

- a) Gegeben sei eine Relation $R(A, B, C, D)$ mit den funktionalen Abhängigkeiten:

$$\begin{aligned} F = & \{ A B C \rightarrow D \\ & D \rightarrow B C \} \end{aligned}$$

Alle Attribute der Relation sind atomar.

- a1) Bestimmen Sie alle möglichen Schlüssel von R . Begründen Sie! (1 Punkt)
- a2) In welcher höchsten Normalform liegt R vor? Begründen Sie! (1 Punkt)
- a3) Führen Sie eine BCNF-Zerlegung nach dem aus der Vorlesung bekannten Dekompositionsalgorithmus durch. Geben Sie an, welche funktionale Abhängigkeit Sie gewählt haben und begründen Sie kurz. Geben Sie die beiden Relationen R_1 und R_2 nach dem ersten Zerlegungsschritt an. (2 Punkte)
- a4) Ist die Zerlegung aus a3) abhängigkeitstreu? Begründen Sie! (1 Punkt)

(5 Punkte)

- b) Gegeben seien die drei Relationen *Person*, *Fahrzeug* und *Kind*. Der Schlüssel ist jeweils unterstrichen. Weiterhin gegeben ist jeweils ein Ausschnitt ihrer Einträge in der Datenbasis:

Person(PerName, *KindName*, *FahrzgNr*)

PERNAME	KINDNAME	FAHRZGNR
Thomas	Timmy	KA-AE-503
Thomas	Tina	KA-AE-503
Thomas	Timmy	S-IT-042
Thomas	Tina	S-IT-042
...

Fahrzeug(*FahrzgNr*, *Marke*, *Baujahr*)

FAHRZGNR	MARKE	BAUJAHR
...
KA-AE-503	VW	1999
...
S-IT-042	Audi	2010
...

Kind(*KindName*, *Hobby*, *Verein*)

KINDNAME	HOBBY	VEREIN
...
Timmy	Lesen	Fußball
Timmy	Lesen	Tennis
...
Tina	Tanzen	Tennis
Tina	Singen	Tennis
...

Eine Person kann mehrere Kinder und/oder Fahrzeuge besitzen. Ein Kind kann mehrere Hobbies haben und/oder mehreren Sportvereinen angehören.

- b1) Was ist eine mehrwertige Abhängigkeit (multi-valued dependency - MVD)? (Formale Definition oder anschauliche Erklärung) (1 Punkt)
- b2) Welche nicht trivialen MVDs kommen in *Person*, *Kind* und *Fahrzeug* vor? (2 Punkte)

(3 Punkte)

- c) Gegeben sei eine Relation $R(A, B, C)$. Für welche funktionalen Abhängigkeiten (FDs) und Zerlegungen von R in R' und R'' ist die Zerlegung abhängigkeitstreu und/oder verbundtreu?

Füllen Sie hierfür die Tabelle *auf dem Lösungsblatt* aus. Richtige Kreuze werden mit 0,5 Punkten bewertet, bei falschen Kreuzen werden 0,5 Punkte abgezogen. Insgesamt wird die Aufgabe mit mindestens 0 Punkten bewertet.

(3 Punkte)

FDs	ZERLEGUNG	ABHÄNGIGKEITSTREU		VERBUNDTREU	
		JA	NEIN	JA	NEIN
$A \rightarrow B, C \rightarrow B$	$R'(\underline{A}, B), R''(B, \underline{C})$	■	■	■	■
$A \rightarrow B, B \rightarrow C$	$R'(\underline{A}, B), R''(\underline{B}, C)$	■	■	■	■
$A \rightarrow B, B \rightarrow C, A \rightarrow C$	$R'(\underline{A}, B), R''(\underline{B}, C)$	■	■	■	■

- d) F_1 und F_2 sind zwei Mengen von funktionalen Abhängigkeiten zur gleichen Relation $R(A, B, C)$. Ihre Hüllen sind F_1^+ bzw. F_2^+ , ihre Schlüssel sind S_1 bzw. S_2 . Widerlegen Sie durch ein Gegenbeispiel unter Nutzung von F_1 und F_2 jeweils die Allgemeingültigkeit der Implikationen Imp1 und Imp2.

$$F_1^+ = F_2^+ \Rightarrow F_1 = F_2 \quad (\text{Imp1})$$

$$S_1 = S_2 \Rightarrow F_1^+ = F_2^+ \quad (\text{Imp2})$$

(3 Punkte)

Aufgabe 4 – SQL (13 Punkte)

Für die Klassifizierung und Bewertung verschiedener Orte der Stadt hat das Rathaus eine App entwickelt, die ein SQL-basiertes Datenbanksystem nutzt. Dort seien Orte, Experten und Rezensionen durch folgende Relationen modelliert:

```

CREATE TABLE Ort
(
    id NUMBER NOT NULL,
    typ CHAR NOT NULL,
    name VARCHAR2(20) NOT NULL,
    x-Koordinate NUMBER NOT NULL,
    y-Koordinate NUMBER NOT NULL,
    CONSTRAINT ort_pk PRIMARY KEY (id)
);

CREATE TABLE Experte
(
    id NUMBER NOT NULL,
    name VARCHAR2(20) NOT NULL,
    quote NUMBER NOT NULL,
    CONSTRAINT experte_pk PRIMARY KEY (id)
);

CREATE TABLE Rezension
(
    e_id NUMBER NOT NULL,
    o_id NUMBER NOT NULL,
    note NUMBER NOT NULL
);

```

Die Inhalte der Relationen sind wie folgt:

Ort				
id	typ	name	x-Koordinate	y-Koordinate
1	'R'	'Ciao Bello'	75	100
2	'B'	'Jambolambo'	50	50
3	'C'	'Café chez Tintin'	25	50

Experte		
id	name	quote
1	'Bob'	3.5
2	'Jack'	3
3	'Alice'	5

Rezension		
e_id	o_id	note
1	1	3
1	2	2
1	3	5
2	1	5
2	2	3
2	3	4
3	1	3
3	2	2
3	3	5

Hinweise:

- note bezeichnet die Note (ganzzahlig, von 1 bis 5 einschließlich), die ein Experte einem Ort gegeben hat.
- quote ist die Quote der Experten (Gleitkommazahl), die das Rathaus selbst für jeden Experten berechnet.
- typ ist der Typ eines Ortes. 'R' steht für Restaurant, 'B' für Bar und 'C' für Café.
- Die x-Koordinate und y-Koordinate (ganzzahlig, von -200 bis 200) sind die vereinfachten geographischen Koordinaten eines Ortes. Der Wert der x-Koordinate steigt von links nach rechts und der Wert der y-Koordinate steigt von unten nach oben.

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) Das Rathaus möchte wissen, ob Orte von einem bestimmten Typ besser bewertet werden als Orte anderer Typen. Formulieren Sie eine SQL-Anfrage, die die durchschnittliche Note für jeden Typ der Orte ausgibt.

Ergebnis bei gegebenem Datenbankinhalt:

('R', 3.67), ('B', 2.33), ('C', 4.67).

(1,5 Punkte)

- b) Ein Experte möchte immer als Erster Orte bewerten, insbesondere Bars. Daher möchte er zuerst herausfinden, welche Bars noch keine Rezension haben. Formulieren Sie eine SQL-Anfrage, die den Namen aller Bars ausgibt, die noch keine Rezension haben.

(2,5 Punkte)

- c) Das Rathaus hat entschieden, faule und/oder unprofessionelle Experten sowie deren Rezensionen aus der Datenbank zu löschen. Ein Experte wird als faul bezeichnet, wenn er weniger als zwei Rezensionen gemacht hat. Ein Experte wird als unprofessionell bezeichnet, wenn er eine Quote kleiner als 3 hat. Formulieren Sie die SQL-Anfragen, die diese Experten und deren Rezensionen löschen.

(5 Punkte)

- d) Das Rathaus möchte den Nutzern nützliche Listen mit ausgewählten Orten anbieten. Als Beispiel soll eine Liste mit den besten Orten aus der Südstadt geliefert werden. Erstellen Sie eine Sicht namens 'Suedstadt', die alle Orte (alle Spalten der Relation 'Ort') ausgibt, die in der Südstadt liegen, und keine Note schlechter als 3 erhalten haben.

Hinweise: Ein Ort befindet sich in der Südstadt, wenn er vollständig innerhalb des Rechtecks liegt, das folgendermaßen spezifiziert ist: seine Ecke unten links hat die Koordinaten (x_l, y_l), seine Ecke rechts oben hat die Koordinaten (X_r, Y_r).

(3 Punkte)

- e) Das Rathaus möchte zusätzliche Informationen über den durchschnittlichen Preis der Angebote jedes Ortes einfügen. Erweitern Sie das Schema der Relation 'Ort' um eine neue Spalte 'Durchschnittspreis'. Für die schon bestehenden Orte soll der Durchschnittspreis 10 betragen.

Hinweise: Lösungen der vorherigen Teilaufgaben müssen Sie nicht entsprechend anpassen.

(1 Punkt)

Aufgabe 5 – Histories (15 Punkte)

Gegeben seien die folgenden Transaktionen:

$$T_1 = \begin{matrix} r_1[x] & r_1[y] & w_1[x] & c_1 \\ 11r & 12r & 13w & 14c \end{matrix}$$

$$T_2 = \begin{matrix} w_2[x] & w_2[y] & r_2[z] & w_2[z] & c_2 \\ 21w & 22w & 23r & 24w & 25c \end{matrix}$$

$$T_3 = \begin{matrix} r_3[y] & r_3[x] & w_3[z] & w_3[y] & c_3 \\ 31r & 32r & 33w & 34w & 35c \end{matrix}$$

Hinweis: Die Bezeichner unterhalb der Operationen sollen im Rahmen der Lösung verwendet werden, um einen genauen Bezug auf bestimmte Operationen nehmen zu können. Die Bezeichner folgen dabei dem Schema: 1. Ziffer entsprechend Transaktionsindex, 2. Ziffer entsprechend Position in Transaktion; die 3. Ziffer dient als Hilfe, um den Operationstyp zu erkennen.

- a) Ermitteln Sie die Menge aller konfigurierenden Operationspaare. Zur Erinnerung: Zwei Operationen sind dann konfigurierend, wenn die Operationen aus unterschiedlichen Transaktionen stammen, der Zugriff auf die selbe Variable erfolgt und eine der beiden Operationen ein Schreibzugriff ist. Verwenden Sie zur Angabe der Lösungsmenge die oben eingeführten Bezeichner.

(4 Punkte)

- b) Basierend auf den drei Transaktionen seien folgende Histories definiert:

$$H_1 = 21w\ 11r\ 22w\ 23r\ 24w\ 31r\ 32r\ 33w\ 34w\ 25c\ 35c\ 12r\ 13w\ 14c$$

$$H_2 = 11r\ 31r\ 12r\ 32r\ 13w\ 21w\ 22w\ 23r\ 33w\ 34w\ 24w\ 25c\ 35c\ 14c$$

Alle Konfliktpaare, d. h. konfigurierenden Operationspaare (siehe auch Aufgabenteil a)), sollen konkret für beide Histories analysiert werden. Dafür soll in den Lösungsblättern je eine Tabelle für H_1 und H_2 ausgefüllt werden. Hierbei gibt es für jedes Konfliktpaar eine Zeile in der Lösungstabelle. Gehen Sie beim Ausfüllen jeder Tabelle wie folgt vor:

- In der ersten Spalte notieren Sie das Konfliktpaar und zwar in der Reihenfolge wie die beiden Operationen in der jeweiligen History auftreten. *Beispiel:* (21w, 11r) bei H_1 bzw. (11r, 21w) bei H_2 .
- Geben Sie in der zweiten Spalte an, welche Transaktionsreihenfolge durch das Konfliktpaar impliziert wird. Verwenden Sie die Notation $T_i < T_j$ um auszudrücken, dass T_i vor T_j erfolgt. *Beispiel:* Das Konfliktpaar (21w, 11r) bei H_1 impliziert $T_2 < T_1$.
- In der dritten Spalte soll angegeben werden, ob das Konfliktpaar gegebenenfalls zu einer *reads-from* oder *overwrite* Beziehung führt. Für die Einträge in dieser Spalte soll daher eine der drei folgenden Möglichkeiten verwendet werden: "rf" für eine *reads-from* Beziehung, "ow" für ein *Overwrite* und "–" falls weder eine *reads-from* noch eine *overwrite* Beziehung vorliegt. *Beispiel:* Bei H_1 wäre für das Konfliktpaar (21w, 11r) "rf" einzutragen, da eine *reads-from* Beziehung vorliegt.
- Zum Ausfüllen der drei letzten Spalten sollen die drei Rücksetzbarkeitsbedingungen *rücksetzbar* (RC), *kaskadenfrei-rücksetzbar* (ACA) und *strikt-rücksetzbar* (STRICT) überprüft werden. Die Spalten müssen nur dann ausgefüllt werden, wenn eine Überprüfung zutreffend ist. Das heißt für eine *reads-from* Beziehung müssen die Bedingungen RC und ACA überprüft werden; für *overwrite* Beziehungen entsprechend die STRICT Spalte. Lassen Sie die Felder einfach unausgefüllt, wenn eine Überprüfung nicht zutreffend ist. Benutzen Sie die Symbole "✓", wenn eine Bedingung erfüllt ist und "–" für eine verletzte Bedingung. *Beispiel:* Bei H_1 handelt es sich beim Konfliktpaar (21w, 11r) um eine *reads-from* Beziehung. Daher sind RC und ACA zu überprüfen.

(6 Punkte)

Lösungstabelle für H_1 :

Konfliktpaar	Transaktionsreihenfolge	reads-from oder overwrite	RC	ACA	STRICT
21w,11r	$T_2 < T_1$	rf	✓	-	

Lösungstabelle für H_2 :

Konfliktpaar	Transaktionsreihenfolge	reads-from oder overwrite	RC	ACA	STRICT

- c) Analysieren Sie die maximalen Rücksetzbarkeitsklassen von H_1 und H_2 . Sie dürfen für die Begründung Ihrer Lösung die Ergebnisse aus Aufgabenteil b) verwenden.

(2 Punkte)

- d) Konstruieren Sie je für H_1 und H_2 die Serialisierbarkeitsgraphen $SG(H_1)$ und $SG(H_2)$. Es ist dabei nicht notwendig die Kanten des Serialisierbarkeitsgraphen zu annotieren. Begründen Sie für jede History, ob sie serialisierbar ist.

(3 Punkte)