UNIVERSITÄT DES SAARLANDES
Fachbereich 14 – Informatik
Lehrstuhl Prof. Weikum

Datenbanksysteme WS 1998/99

Erste Teilklausur — Beispiellösung

Aufgabe 1: Transformation RA ⇔ TRK (3+3=6 Punkte)

Betrachten Sie folgenden Teil des aus den Übungen bekannten Schemas der Musikdatenbank:

Disk	(DiskID, DiskTitel, Preis)
Musikstück	(DiskID, StückID, Titel, Länge)
Person	(PID, Name, Nationalität)
Interpret	(PID, DiskID, StückID, Instrument)

Die unterstrichenen Attribute bilden den Primärschlüssel.

a) Transformieren Sie den folgenden Ausdruck im Tupel-Relationenkalkül in einen äquivalenten Ausdruck der **Relationenalgebra**:

{ d.DiskTitel | d ∈ Disk ∧
∀ m: ((m ∈ Musikstück ∧ m.DiskId=d.DiskId) ⇒ m.Länge > 10)}

Lösung:

$\pi_{DiskTitel}(Disk \bowtie (\pi_{DiskID}(Musikstück) - \pi_{DiskID}(\sigma_{Länge \leq 10}(Musikstück))))$
(Annahme: Es existieren keine Disks ohne Musikstücke.)

Achtung: Der Ausdruck

(*) $\pi_{DiskTitel}(Disk) - \pi_{DiskTitel}(\sigma_{Länge \leq 10}(Musikstück))$,

ist keine korrekte Lösung, da *DiskTitel* kein Schlüssel ist. Es kann beispielsweise vorkommen, daß zwei Disks mit dem Titel *Best of* existieren und genau eine der beiden Disks nur Stücke enthält, die länger als 10 (Sekunden) sind. Im Ergebnis muß dann der Titel *Best of* enthalten sein, was bei (*) jedoch nicht der Fall wäre.

b) Transformieren Sie den folgenden Ausdruck der Relationenalgebra in einen äquivalenten Ausdruck im **Tupel-Relationenkalkül**:

$\pi_{PID,DiskId,StückId}(\text{Interpret}) \div \pi_{DiskId,StückId}(\sigma_{Länge>10}(\text{Musikstück}))$

Lösung:
Bem.1: Das Ergebnis obiger Division sind die PID's der Interpreten, die an **allen** Musikstücken, **deren Länge > 10 ist**, mitgewirkt haben. Dies leistet auch folgender TRK-Ausdruck :

$$\{p.PID \mid p \in Interpret \wedge \forall m : ((m \in Musikstück \wedge m.Länge > 10) \Rightarrow (\exists p2 : p2 \in Interpret \wedge p2.PID = p.PID \wedge p2.DiskID = m.DiskID \wedge p2.StückID = m.StückID))\}$$

Bem. 2: Dagegen ist folgender TRK-Ausdruck bei näherer Betrachtung **falsch**:

$$\{p.PID \mid p \in Interpret \wedge \forall m : ((m \in Musikstück \wedge m.Länge > 10) \Rightarrow (m.DiskID = p.DiskID \wedge m.StückID = p.StückID))\}$$

Sobald mehr als zwei verschiedene Musikstücke existieren, kann kein p aus Interpret die Bedingung für alle m erfüllen.

Aufgabe 2: SQL (3+3=6 Punkte)

Gegeben sei das folgende Schema einer Bibliotheksdatenbank:

Bücher	(ISBN, Kategorie, Titel, AnzahlExemplare)
Leser	(LeserID, Name)
Ausleihe	(LeserID, ISBN, vonDatum, bisDatum)

Die unterstrichenen Attribute bilden den Primärschlüssel. Das Attribut *bisDatum* enthält das Rückgabedatum und ist auf *NULL* gesetzt, solange der Leser das Buch noch nicht zurückgegeben hat.
Formulieren Sie die folgenden Anfragen in SQL.

a) Die Namen der Leser, die bisher noch nie ein Buch der Kategorie *Theorie* ausgeliehen haben.

Lösung:

```
SELECT L.Name
FROM Leser L
WHERE NOT EXISTS( SELECT *
                  FROM Ausleihe A2, Buecher B2
                  WHERE B2.Kategorie="Theorie" AND
                        B2.ISBN=A2.ISBN AND
                        A2.LeserId=L.LeserId );
```

b) Die Namen der Leser mit den meisten aktuell ausgeliehenen *Theorie*-Büchern.
Hinweis: Sie dürfen hier auch *View*-Definitionen verwenden.

Lösung:

```
CREATE VIEW TheorieAusleihen( LeserId, AnzahlAusleihen ) AS
SELECT A.LeserId, COUNT(*)
FROM Ausleihe A, Buecher B
WHERE A.ISBN=B.ISBN AND
      B.Kategorie="Theorie" AND
      A.bisDatum IS NULL
GROUP BY LeserId;

SELECT L.Name
FROM Leser L, TheorieAusleihen T
WHERE L.LeserId=T.LeserId AND
      NOT EXISTS( SELECT *
                  FROM TheorieAusleihen T2
                  WHERE T2.AnzahlAusleihen > T.AnzahlAusleihen );
```

** alternativ: **

```
SELECT L.Name
FROM Ausleihe A, Buecher B, Leser L
WHERE A.ISBN=B.ISBN AND L.LeserId=A.LeserId
      B.Kategorie="Theorie" AND
      A.bisDatum IS NULL
GROUP BY L.LeserId, L.Name
HAVING COUNT(*) >= ALL( SELECT COUNT(*)
                        FROM Ausleihe A2, Buecher B2
                        WHERE A2.ISBN=B2.ISBN AND
                              B2.Kategorie="Theorie" AND
                              A2.bisDatum IS NULL
                              GROUP BY A2.LeserId
                       );
```

Aufgabe 3: Integritätssicherung (3+3=6 Punkte)

a) Sichern Sie mittels einer ASSERTION die folgende Integritätsbedingung in der Bibliotheksdatenbank aus Aufgabe 2:

> *Es dürfen zum aktuellen Zeitpunkt nie mehr Exemplare eines Buches ausgeliehen sein als vorhanden sind.*

Lösung:

```
CREATE ASSERTION MaxAusleihe CHECK
Buecher.AnzahlExemplare >=
(SELECT COUNT(*)
 FROM Ausleihen A
 WHERE A.ISBN=Buecher.ISBN AND A.bisDatum IS NULL)
```

** alternativ: **

```
CREATE ASSERTION MaxAusleihe CHECK
 NOT EXISTS(
```



```
            SELECT *
            FROM Buecher B
            WHERE B.AnzahlExemplare < ( SELECT COUNT(*)
                                        FROM Ausleihe A
                                        WHERE A.ISBN=B.ISBN AND
                                        A.bisDatum IS NULL )
            );
```

b) Sichern Sie mittels eines TRIGGERS die folgende Integritätsbedingung in der Bibliotheksdatenbank aus Aufgabe 2:

> *Leser, die 3 oder mehr Bücher seit über 60 Tagen ausgeliehen haben, dürfen keine Bücher mehr ausleihen.*

Lösung:

```
CREATE TRIGGER Sperre
BEFORE INSERT ON Ausleihen
REFERENCING NEW AS BNEW
FOR EACH ROW
WHEN( (SELECT COUNT(*)
       FROM Ausleihen A
       WHERE A.LeserId=BNew.LeserId AND
             A.vonDatum+60 < SYSDATE AND
             A.bisDatum IS NULL
       ) >=3
     )
( ROLLBACK WORK; )
```

HINWEIS: Die (Pseudo-) Konstante *SYSDATE* liefert das aktuelle Tagesdatum. Sie können die Datumsarithmetik von SQL direkt verwenden, also z.B. *SYSDATE+60* u.ä. schreiben.