UNIVERSITÄT DES SAARLANDES
Fachrichtung 6.2 - Informatik
Lehrstuhl Prof. Weikum

# Informationssysteme SS 2002

## Übung 4 Beispiellösung

## Aufgabe 2: Abbildung von SQL auf TRK und RA

Geben Sie für die folgenden SQL-Anfragen auf der Musikdatenbank äquivalente Formulierungen in der Relationenalgebra und dem sicheren Tupelrelationenkalkül an.

a)
```
Select  D.DiskTitel
From    Disk D
Where   D.DiskID In
            ( Select  M.DiskID From  Musikstück M
              Where M.Titel = 'I love you'
                And  M.DiskID In
                        ( Select  I.DiskID    From Interpret I
                          Where  I.Instrument = 'Triangel'
                            And I.StückID = M.StückID ) )
```

Die intuitive Bedeutung der Anfrage ist:
Auf welchen CDs ist ein Stück "I love you" mit einer Triangel zu hören?

RA: $\pi$[DiskTitel]
((Disk) |×| $\sigma$[Titel = 'I love you'](Musikstück) |×| $\sigma$[Instrument='Triangel'](Interpret))

TRK: $\{d.DiskTitel \mid d \in Disk \wedge$
$\exists m (m \in Musikstück \wedge m.Titel = 'I love you' \wedge m.DiskID = d.DiskID \wedge$
$\exists i (i \in Interpret \wedge i.Instrument = 'Triangel' \wedge i.DiskID = d.DiskID \wedge$
$i.StückID = m.StückID))\}$

b)
```
Select  D.DiskTitel
From    Disk D
Where   D.Preis < 20
   And  Exists
            ( Select  *
              From   Interpret I
              Where  I.Instrument = 'Sitar'
                And  I.DiskID = D.DiskID )
   And  Not Exists
            ( Select  *
              From  Musikstück M
              WhereM.Länge > 5
                And  M.DiskID = D.DiskID )
```

Die intuitive Bedeutung der Anfrage ist:
Auf welchen CDs unter 20 DM ist eine Sitar zu hören und kein einziges Stück länger als 5 Minuten?

RA: $\pi$[DiskTitel] ( $\sigma$[Preis < 20] (Disk) |×| $\sigma$[Instrument = 'Sitar'] (Interpret) –
$\sigma$[Länge > 5] (Disk |×| Musikstück))

TRK: {d.DiskTitel | d ∈ Disk ∧ d.Preis < 20 ∧
∃i (i ∈ Interpret ∧ i.Instrument = 'Sitar' ∧ i.DiskID = d.DiskID) ∧
¬∃m (m ∈ Musikstück ∧ m.Länge > 5 ∧ m.DiskID = d.DiskID)}
ist unsicher, also

c) Select D.DiskTitel
From Disk D
Where D.DiskID = All ( Select M.DiskID
From Musikstück M, Person P, Interpret I, Autor A
Where (M.StückID = I.StückID Or M.StückID = A.StückID)
And (P.PID = I.PID Or P.PID = A.PID)
And P.Nationalität = 'Luxemburg' )

Die intuitive Bedeutung der Anfrage ist:
Gibt es eine einzige CD, an der ein Luxemburger beteiligt ist (als Interpret oder Komponist oder Dirigent oder ...)?
Falls ja, gib diese CD aus.
Falls es mehrere CDs mit Luxemburgern gibt, gib die leere Menge aus.
Falls es überhaupt keine CDs mit Luxemburgern gibt (das Ergebnis der Subquery also leer ist), gib alle CDs aus.

RA: M := Musikstück;
I := Interpret;
A := Autor;
P := Person;

$\pi$[DiskTitel] (
$\pi$[DiskID] (Disk)
–
$\pi$[Disk.DiskID] ($\sigma$[M.DiskID ≠ Disk.DiskID]
($\sigma$[Nationalität = ‚Luxemburg' ∧
(M.StückID=I.StückID ∨ M.StückID=A.StückID) ∧
(P.PID =I.PID ∨ P.PID=A.PID)]
(M |×| P |×| I |×| A |×| Disk)))
)

TRK: {d.DiskTitel | d ∈ Disk ∧ ∀m (m ∈ Musikstück ⇒
∃m ∃p ∃i ∃a (m ∈ Musikstück ∧ p ∈ Person ∧ i ∈ Interpret ∧ a ∈ Autor ∧
p.Nationalität = 'Luxemburg' ∧
(p.PID = i.PID ∨ p.PID = a.PID) ∧
(m.StückID = i.StückID ∨ m.StückID = a.StückID)))}

Formale Herleitung der Semantik
(Dieser formale Teil wurde bei der Lösung eigentlich nicht erwartet
und dient hier der zusätzlichen Illustration.)

a1)
sql2trc [Select...From D Where D.DiskID In
(Select...From M Where M.Titel=...And M.DiskID In
(Select...From I Where I.Instrument=...And I.StückID=...))]

= {d.DiskTitel | d in D and exists m ( m in M and d.DiskID=m.DiskID and
sql2trc' [M.Titel=... And
M.DiskID In (Select...From I Where I.Instrument=...And I.StückID=...)])}

= {d.DiskTitel | d in D and exists m ( m in M and d.DiskID=m.DiskID and m.Titel=... and
exists i (i in I and m.DiskID=i.DiskID
and sql2trc'[I.Instrument=...And I.StückID=...]))}

= {d.DiskTitel | d in D and exists m ( m in M and d.DiskID=m.DiskID and m.Titel=... and
exists i (i in I and m.DiskID=i.DiskID and i.Instrument=... and i.StückID=...))}

a2)
sql2ra [Select...From D Where D.DiskID In
(Select...From M Where M.Titel=...And M.DiskID In
(Select...From I Where I.Instrument=...And I.StückID=...))]

= pi[D.DiskTitel] (sql2ra'[Where D.DiskID In
(Select...From M Where M.Titel=...And M.DiskID In
(Select...From I Where I.Instrument=...And I.StückID=...))] (D)

= pi[D.DiskTitel] (sql2ra'[Where M.Titel=... And M.DiskID In
(Select...From I Where I.Instrument=...And I.StückID=...)]
(sigma[D.DiskID=M.DiskID] (D x M)))

= pi[D.DiskTitel] (sigma[M.Titel=...]
(sql2ra'[Where I.Instrument=... And I.StückID=...]
(sigma[M.DiskID=I.DiskID] (sigma[D.DiskID=M.DiskID] (D x M) x I))))

= pi[D.DiskTitel] (sigma[M.Titel=...]
(sigma[I.Instrument=... and I.StückID=...]
(sigma[M.DiskID=I.DiskID] (sigma[D.DiskID=M.DiskID] (D x M) x I))))

b1)
sql2trc [Select...From D Where D.Preis<...And
Exists (Select...From I Where I.Instrument=...And I.DiskID=...) And
Not Exists (Select...From M Where M.Länge>...And M.DiskID=...)]

= {d.DiskTitel | d in D and d.Preis<... and
exists i (i in I and i.instrument=... and i.DiskID=...) and
not exists m (m in M and m.Länge>... and m.DiskId=...)}

b2)
sql2ra [Select...From D Where D.Preis<...And
Exists (Select...From I Where I.Instrument=...And I.DiskID=...) And

```
                Not Exists (Select...From M Where M.Länge>...And M.DiskID=...)]

= pi[D.DiskTitel] (sql2ra'[D.Preis<... And
                Exists (Select...From I Where I.Instrument=...And I.DiskID=...) And
                Not Exists (Select...From M Where M.Länge>...And M.DiskID=...)] (D))

= pi[D.DiskTitel] (sigma[D.Preis<...](D) intersect
            sql2ra'[Exists (Select...From I Where I.Instrument=...And I.DiskID=...) And
                Not Exists (Select...From M Where M.Länge>...And M.DiskID=...)] (D))

= pi[D.DiskTitel] (sigma[D.Preis<...](D) intersect
            sql2ra'[I.Instrument=...And I.DiskID=...And
                Not Exists (Select...From M Where M.Länge>...And M.DiskID=...)] (D x I))

= pi[D.DiskTitel] (sigma[D.Preis<...](D) intersect
             pi[sch(D)](sigma[I.Instrument=...and I.DiskID=...] (D x I)) intersect
             pi[sch(D)]((D x I) minus
                    pi[sch(D x I)](sigma[M.Länge>...And M.DiskID=...](D x I x M))))


c1)
sql2trc [Select...From D Where D.DiskID =All (Select M.DiskID From M,P,I,A Where cond)]

= {d.DiskTitel | d in D and forall m
            (m in M => (d.DiskID=m.DiskID and
                    exists m,p,i,a
                        (m in M and p in P and i in I and a in A and sql2trc'[cond])))}

mit
sql2trc'[cond] =  (m.StückD=i.StückID or M.StückID=a.StückID) and
                  (p.PID=i.PID or p.PID=a.PID) and p.Nationalität='Luxemburg'



c2)
sql2ra [Select...From D Where D.DiskID =All (Select M.DiskID From M,P,I,A Where cond)]

= pi[D.DiskTitel] ( sql2ra'[=All (Select M.DiskID From M,P,I,A Where cond] (D))

= pi[D.DiskTitel] ( sql2ra-[<>Any (Select M.DiskID From M,P,I,A Where cond] (D))

= pi[D.DiskTitel] ( D minus
              pi[sch(D)] (sql2ra'[<>Any (Select M.DiskID From M,P,I,A Where cond](D)) )

= pi[D.DiskTitel] ( D minus
              pi[sch(D)] (sql2ra'[cond] (sigma[D.DiskID<>M.DiskID](D x M x P x I x A))) )


mit
sql2ra'[cond](expr) = sigma[(M.StückID=I.StückID or M.StückID=A.StückID) and
                                (P.PID=I.PID or P.PID=A.PID) and P.Nationalität='Luxemburg'] (expr)
```

# Aufgabe 2: SQL Abfragen

Betrachten Sie die vereinfachte Universitätsdatenbank mit Informationen über Fachbereiche, Dozenten, Lehrangebote, Studenten und Prüfungen. Das Schema der Datenbank (mit Beispielausprägungen) ist unten aufgeführt (Primärschlüssel sind unterstrichen):

**Departments** (DName, Chair):

| DName | Chair |
|---|---|
| Computer Science | Bob Smith |
| Electrical Engineering | John Miller |
| ... | ... |

**Teachers** (TName, Office, Phone, DName):

| TName | Office | Phone | DName |
|---|---|---|---|
| Bob Smith | 122 | 4819 | Computer Science |
| Mary Taylor | 245 | 4716 | Computer Science |
| John Miller | 312 | 223322 | Electrical Engineering |
| Mike Franklin | 444 | 4545 | Electrical Engineering |
| ... | ... | ... | ... |

**Courses** (CNo, Title, Semester, Room, Schedule, Lecturer):

| CNo | Title | Semester | Room | Schedule | Lecturer |
|---|---|---|---|---|---|
| 1 | Database Systems | Summer 2001 | 322 | ... | Mary Taylor |
| 2 | Wireless Communication | Winter 2001 | 455 | ... | John Miller |
| 3 | Wireless Communication | Summer 2002 | 455 | ... | Mike Franklin |
| ... | ... | ... | ... | ... | ... |

**Students** (SNo, SName, Address, Major, Minor)

| SNo | SName | Address | Major | Minor |
|---|---|---|---|---|
| 1001 | David Chang | ... | Computer Science | Psychology |
| 1002 | Sunita Singh | ... | Electrical Engineering | Computer Science |
| 1003 | Joe Doe | ... | Electrical Engineering | Physics |
| ... | ... | ... | ... | ... |

**Exams** (SNo, CNo, EDate, Grade):

| Sno | CNo | EDate | Grade |
|---|---|---|---|
| 1001 | 1 | 27 July 2001 | 1.7 |
| 1002 | 2 | 15 July 2001 | 2.0 |
| 1002 | 1 | 28 July 2001 | 1.3 |
| 1003 | 3 | NULL | NULL |
| ... | ... | ... | ... |

Das Attribut *Chair* der Relation *Department* ist ein Fremdschlüssel bzgl. *Teacher.TName*; *Courses.Lecturer* ist Fremdschlüssel bzgl. *Teacher.TName*; in der Relation *Students* sind *Major* (Hauptfach) und *Minor* (Nebenfach) Fremdschlüssel bzgl. *Department.DName*.

Formulieren Sie folgende Anfragen in SQL :

a) Wie heisst der Dekan (*Chair*) des Fachbereichs (*Department*) mit der besten Durchschnittsnote, berechnet über alle Kurse des Sommersemesters 2002?

```
SELECT D1.Chair
FROM Departments D1, Teachers T1, Courses C1, Exams E1
WHERE
   C1.Lecturer = T1.TName AND
   C1.CNo = E1.CNo AND
```

```
   D1.DName = T1.DName AND
   C1.Semester = 'Summer 2001' AND
   E2.Grade IS NOT NULL
GROUP BY D1.DName, D1.Chair
HAVING AVG (E1.Grade) >= ALL
    (
     SELECT AVG (E2.Grade)
     FROM Teachers T2, Exams E2, Courses C2
     WHERE
        C2.CNo = E2.CNo AND
        C2.Lecturer = T2.TName AND
        C2.Semester = 'Summer 2001' AND
        E2.Grade IS NOT NULL
     GROUP BY T2.DName
)
```

b) Welche Studenten sind im Nebenfach (*Minor*) ständig erfolgreicher als im Hauptfach (*Major*)? Auszugeben sind die Namen solcher Studenten sowie deren Haupt- und Nebenfach.

```
SELECT S.SName, S.Major, S.Minor
FROM Students S
WHERE S.SNo IN
(
SELECT S1.SNo
FROM Students S1, Exams E1, Courses C1, Teachers T1
WHERE
   E1.SNo = S1.SNo AND                          All Minor-Grades, grouped by Student
   E1.CNo = C1.CNo AND
   T1.TName = C1.Lecturer AND
   S1.Minor = T1.DName AND
   E1.Grade IS NOT NULL
GROUP BY S1.SNo
HAVING MAX (E1.Grade) <
(
  SELECT MIN (E2.Grade)
  FROM Students S2, Exams E2, Courses C2, Teachers T2
  WHERE
     E2.SNo = S2.SNo AND                        Min(Major-Grade) of the same student
     E2.CNo = C2.CNo AND                                    > Max (Minor-Grade)
     E2.Grade IS NOT NULL AND
     T2.TName = C2.Lecturer AND
     S2.Major = T2.DName AND
     S1.SNo = S2.SNo
))
```

```
SELECT SName, Major, Minor
FROM
(
SELECT S.SNo, S.SName, S.Major, S.Minor
FROM Students S, Exams E, Courses C, Teachers T
WHERE
   E.SNo = S.SNo AND                            All Minor-Grades, grouped by Student
   E.CNo = C.CNo AND
   T.TName = C.Lecturer AND
   S.Minor = T.DName AND
   E.Grade IS NOT NULL
GROUP BY S.SNo, S.SName, S.Major, S.Minor
HAVING MAX (E.Grade) <
(
```

```
  SELECT MIN (E1.Grade)
  FROM Students S1, Exams E1, Courses C1, Teachers T1
  WHERE
     E1.SNo = S1.SNo AND                    Min(Major-Grade) of the same student
     E1.CNo = C1.CNo                                        > Max (Minor-Grade)
     T1.TName = C1.Lecturer AND
     S1.Major = T1.DName AND
     S1.SNo = S.SNo AND
     E1.Grade IS NOT NULL
))
```

zur besseren Lesbarkeit könnte man zusätzlich zwei Views (*min (Major)* and *max (Minor)* hinzufügen :

```
CREATE VIEW S_MAJOR AS
SELECT S.SNo AS SNo, MIN (E.Grade) AS SMin
FROM Students S, Exams E, Courses C, Teachers T
WHERE
   S.SNo = E.SNo AND                              Best Major-Grade per Student
   E.CNo = C.CNo AND
   E.Grade IS NOT NULL AND
   T.TName = C.Lecturer AND
   S.Major = T.DName
GROUP BY S.SNo

CREATE VIEW S_MINOR AS
SELECT S.SNo AS SNo, MAX (E.Grade) AS SMax
FROM Students S, Exams E, Courses C, Teachers T
WHERE
   S.SNo = E.SNo AND                              Worst Minor-Grade per Student
   E.CNo = C.CNo AND
   T.Tname = C.Lecturer AND
   E.Grade IS NOT NULL AND
   S.Minor = T.DName
GROUP BY S.Sno
```

Die Anfrage lautet:

```
SELECT SName, Major, Minor
FROM Students
WHERE SNo IN
(  SELECT Minor.SNo
   FROM S_Major Major, S_Minor Minor
   WHERE
      Minor.SNo = Major.SNo AND
      Minor.SMax < Major.SMin
)
```