TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung *Einführung in die Informatik 2 für Ingenieure (MSE)*

Alexander van Renen (renen@in.tum.de)
http://db.in.tum.de/teaching/ss16/ei2/

Lösungen zu Blatt 9

Tool zum Üben der relationalen Algebra: http://www-db.in.tum.de/~muehe/ira/.
SQL-Schnittstelle: http://hyper-db.com/interface.html.

Aufgabe 1: (Zur Wiederholung)

Formulieren Sie die folgenden Anfragen auf dem Universitätsschema in **Relationenalgebra**:

Geben Sie Paare von *Studenten*(-Namen) an, die sich aus der *Vorlesung* Grundzüge kennen.

Lösung 1

$$
\begin{aligned}
R \;:=\; & \Pi_{\text{s1.Name, s2.Name}} \\
& (\sigma_{\text{Titel='Grundzüge'}}(\text{Vorlesungen}) \bowtie_{\text{VorlNr=h1.VorlNr}} \\
& (\rho_{\text{s1}}(\text{Studenten}) \bowtie_{\text{s1.MatrNr=h1.MatrNr} \,\wedge \text{s1.MatrNr} \neq \text{s2.MatrNr}} \\
& (\rho_{\text{h1}}(\text{hören})) \bowtie_{\text{h1.VorlNr=h2.VorlNr}} (\rho_{\text{h2}}(\text{hören})) \\
& \bowtie_{\text{s2.MatrNr=h2.MatrNr}} (\rho_{\text{s2}}(\text{Studenten}))))
\end{aligned}
$$

Formulieren Sie folgende Anfragen auf dem Universitätsschema in **SQL**:

Finden Sie die *Studenten*, die *Vorlesungen* hören, die auch Fichte hört.

Lösung 1

```
select distinct s1.Name, s1.MatrNr
from Studenten s1, Studenten s2, hoeren h1, hoeren h2
where s1.MatrNr = h1.MatrNr
and s1.MatrNr != s2.MatrNr
and s2.MatrNr = h2.MatrNr
and h1.VorlNr = h2.VorlNr
and s2.Name = 'Fichte';
```

Aufgabe 2: SQL

Formulieren Sie folgende Anfrage auf dem Universitätsschema in SQL.

Lösung 2

(a) Finden Sie die Namen der *Studenten*, die in keiner *Prüfung* eine bessere Note als 3.0 hatten.

```
SELECT s.Name, s.MatrNr
FROM Studenten s
WHERE NOT EXISTS (SELECT *
                  FROM pruefen p
                  WHERE p.MatrNr = s.MatrNr
                  AND p.Note < 3.0);
```

(b) Alle Studenten müssen ab sofort alle Vorlesungen von Sokrates hören. Formulieren Sie einen SQL-Befehl (insert statement), der diese Operation ausführt.

```
INSERT INTO hoeren
(SELECT s.MatrNr, v.VorlNr
FROM Studenten s, Vorlesungen v, Professoren p
WHERE p.Name = 'Sokrates' AND p.PersNr = v.gelesenVon
AND (s.MatrNr, v.VorlNr) NOT IN (SELECT * FROM hoeren));
```

Aufgabe 3: SQL

Formulieren Sie die folgenden Anfragen auf dem bekannten Universitätsschema in SQL.

Lösung 3

(a) Bestimmen Sie das durchschnittliche Semester der Studenten der Universität.

```
select avg(semester*1.0) from studenten;
```

(b) Bestimmen Sie das durchschnittliche Semester der Studenten, die mindestens eine Vorlesung bei Sokrates hören. Beachten Sie, dass Sie das Semester von Studenten, die mehr als eine Vorlesung bei Sokrates hören, nicht doppelt zählen dürfen.

```
with vorlesungen_von_sokrates as (
  select * from vorlesungen v, professoren p where v.
     gelesenVon = p.persnr and p.name = 'Sokrates'
), studenten_von_sokrates as (
  select * from studenten s where exists (select * from
     hoeren h, vorlesungen_von_sokrates v where h.matrnr = s.
     matrnr and v.vorlnr = h.vorlnr)
)
select avg(semester*1.0) from studenten_von_sokrates
```

Man beachte, dass die Formulierung mittels `WHERE EXISTS` für die Elimination von Duplikaten sorgt, d.h. ein Student, der 3 Vorlesungen von Sokrates hört kommt nur einmal in Studenten_von_sokrates vor, was gewünscht ist. Alternativ kann man studenten_von_sokrates formulieren als:

```
select DISTINCT s.* from studenten s, hoeren h,
   vorlesungen_von_sokrates v where h.matrnr = s.matrnr and v
   .vorlnr = h.vorlnr
```

(c) Bestimmen Sie, wie viele Vorlesungen im Schnitt pro Student gehört werden. Beachten Sie, dass Studenten, die keine Vorlesung hören, in das Ergebnis einfließen müssen.

```
select hcount/(scount*1.0) from
(select count(*) as hcount from hoeren) h,
(select count(*) as scount from studenten) s
```

Aufgabe 4: SQL

Formulieren Sie folgende Anfrage auf dem Universitätsschema in SQL:
Finden Sie die *Studenten*, die **alle** Vorlesungen gehört haben.

Lösung 4

```
SELECT s.Name, s.MatrNr
FROM Studenten s
WHERE NOT EXISTS (SELECT *
                  FROM Vorlesungen v
                  WHERE NOT EXISTS (SELECT *
                                    FROM hoeren h
                                    WHERE h.VorlNr = v.VorlNr
                                    AND h.MatrNr = s.MatrNr));
```