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

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

Christoph Anneser (anneser@in.tum.de), Simon Ellmann (ellmann@in.tum.de)

`http://db.in.tum.de/teaching/ss24/ei2/`

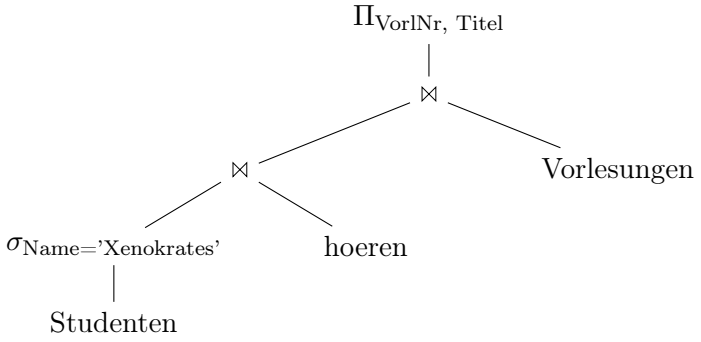
Lösungen zu Blatt 9

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

Aufgabe 1: Relationenalgebra I

Formulieren Sie die folgenden Anfragen auf dem Universitätsschema in Relationenalgebra.

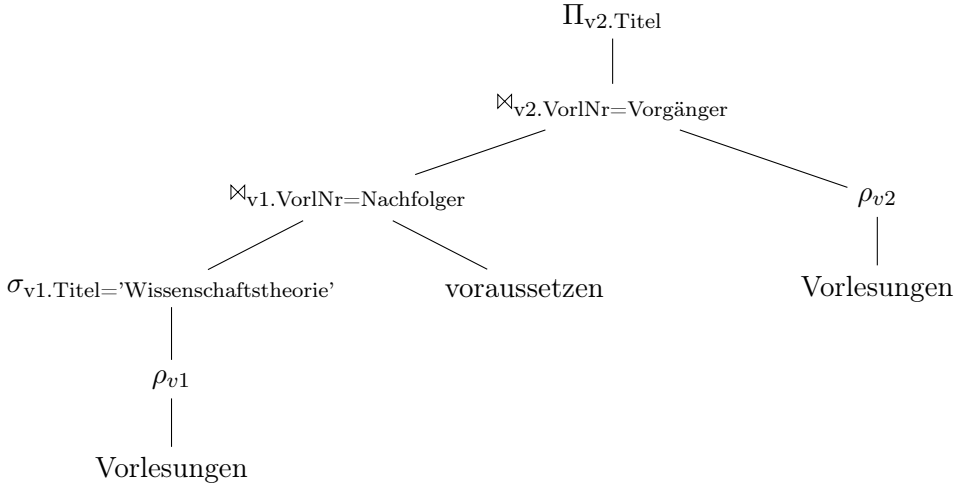
(a) Geben Sie alle *Vorlesungen* an, die der *Student* Xenokrates gehört hat.



Alternativ:

$$R := \Pi_{\text{VorlNr, Titel}}(\text{Vorlesungen} \bowtie (\text{hören} \bowtie (\sigma_{\text{Name='Xenokrates'}}(\text{Studenten}))))$$

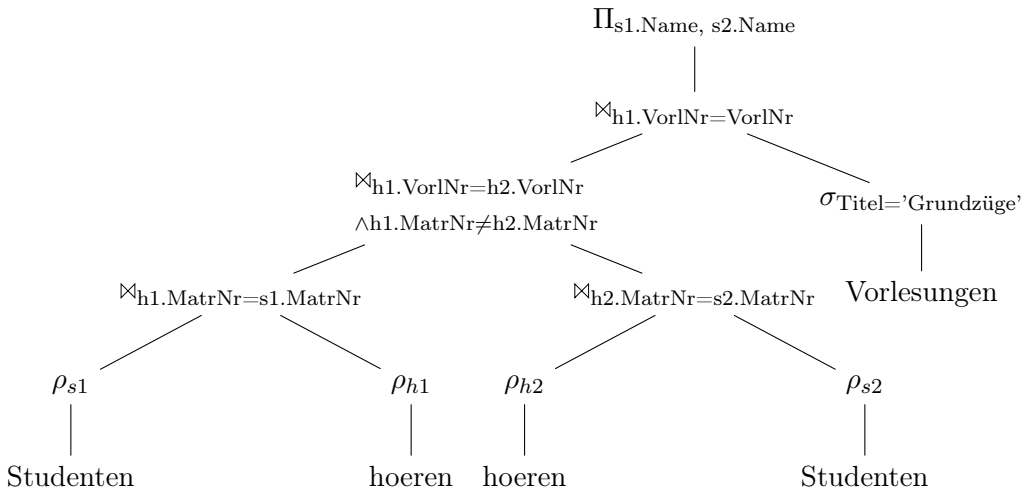
(b) Geben Sie die Titel der direkten Voraussetzungen für die *Vorlesung* Wissenschaftstheorie an.



Alternativ:

$$R := \Pi_{\text{v2.Titel}}(\rho_{\text{v2}}(\text{Vorlesungen}) \bowtie_{\text{v2.VorlNr=Vorgänger}} (\text{voraussetzen} \bowtie_{\text{v1.VorlNr=Nachfolger}} (\sigma_{\text{v1.Titel='Wissenschaftstheorie'}}(\rho_{\text{v1}}(\text{Vorlesungen})))))$$

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



Alternativ:

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

Aufgabe 2: SQL I

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

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

```
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 3: SQL II

(a) Finden Sie die *Studenten*, die Sokrates aus *Vorlesung(en)* kennen.

```
select s.Name, s.MatrNr
from Studenten s, hoeren h, Vorlesungen v, Professoren p
where s.MatrNr = h.MatrNr
and h.VorlNr = v.VorlNr
and v.gelesenVon = p.PersNr
and p.Name = 'Sokrates';
```

`DISTINCT` wäre nett, um Duplikate zu unterdrücken ist aber nicht explizit in der Aufgabe gefordert.

(b) Finden Sie die *Assistenten* von *Professoren*, die den Studenten Fichte unterrichtet haben – z.B. als potentielle Betreuer seiner Diplomarbeit.

```
select a.Name, a.PersNr
from Assistenten a, Vorlesungen v, hoeren h, Studenten s
where a.Boss = v.gelesenVon
and v.VorlNr = h.VorlNr
and h.MatrNr = s.MatrNr
and s.Name = 'Fichte';
```

(c) Geben Sie die Namen der *Professoren* an, die Xenokrates aus *Vorlesungen* kennt.

```
select p.PersNr, p.Name
from Professoren p, hoeren h, Vorlesungen v, Studenten s
where p.PersNr = v.gelesenVon
and v.VorlNr = h.VorlNr
and h.MatrNr = s.MatrNr
and s.Name = 'Xenokrates';
```

(d) Welche *Vorlesungen* werden von *Studenten* im Grundstudium (1.-4. Semester) gehört? Geben Sie die Titel dieser *Vorlesungen* an.

```
select v.Titel
from Vorlesungen v, hoeren h, Studenten s
where v.VorlNr = h.VorlNr
and h.MatrNr = s.MatrNr
and s.Semester between 1 and 4;
```

Aufgabe 4: SQL III – DML

Formulieren Sie folgende Anfrage auf dem Universitätsschema in SQL.

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));
```