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

Übung zur Vorlesung *Grundlagen: Datenbanken* im WS18/19

Moritz Sichert, Lukas Vogel (gdb@in.tum.de)
https://db.in.tum.de/teaching/ws1819/grundlagen/

Blatt Nr. 04

Tool zum Üben von SQL-Anfragen: https://hyper-db.com/interface.html.

Hausaufgabe 1

Gegeben seien die beiden Relationen $R : \{[a_1, \ldots, a_n]\}$ und $S : \{[b_1, \ldots, b_m]\}$. Geben Sie die folgenden Ausdrücke im Tupel- und Domänenkalkül an:

a) $Q_1 := R \bowtie_{a_1=b_1} S$

b) $Q_2 := R \rtimes_{a_1=b_1} S$

c) $Q_3 := R \ltimes_{a_1=b_1} S$

d) $Q_4 := R \triangleleft_{a_1=b_1} S$

Lösung:

Bitte beachten Sie, dass in dieser Aufgabe ausschließlich allgemeine *Theta*joins ($\bowtie_\Theta, \rtimes_\Theta, \ldots$) verwendet werden. Gemäß Definition werden somit alle Attribute der beiden Eingaberelationen in die Ausgaberelation projiziert, einschließlich der Attribute, welche in der Joinbedingung enthalten sind. Lediglich bei *natürlichen* Joins, wo implizit eine Gleichheitsbedingung für alle gleichnamigen Attribute erfüllt sein muss, werden gleichnamige Attribute nicht doppelt in die Ausgaberelation projiziert. Siehe hierzu auch Folie Kapitel 3, "Andere Join-Arten".

a) $Q_1 := R \bowtie_{a_1=b_1} S$

Formulierung im Tupelkalkül

$$Q_1 := \{[r.a_1, \ldots, r.a_n, s.b_1, \ldots, s.b_m] \mid r \in R \ \wedge \ s \in S \ \wedge \ r.a_1 = s.b_1\}$$

Da der Joinoperator Tupel aus verschiedenen Relationen verbindet, müssen für die Ergebnismenge neue Tupel mithilfe des Tupelkonstruktors konstruiert werden: $[\text{attribut}_1 : \text{wert}_1, \ldots, \text{attribut}_n : \text{wert}_n]$. Die oben verwendete Tupelkonstruktion $[r.a_1, \ldots]$ ist eine verkürzte Schreibweise für $[a_1 : r.a_1, \ldots]$ und kann verwendet werden, wenn der Attibutname im neuen Tupel unverändert bleibt.

Im Falle des Thetajoins kann auch die Tupelkonkatenation $t_1 \circ t_2$ verwendet werden:

$$Q_1 := \{r \circ s \mid r \in R \ \wedge \ s \in S \ \wedge \ r.a_1 = s.b_1\}$$

Formulierung im Domänenkalkül

$$Q_1 := \{[a_1, \ldots, a_n, b_1, \ldots, b_m] \mid [a_1, \ldots, a_n] \in R \wedge [b_1, \ldots, b_m] \in S \wedge a_1 = b_1\}$$

oder

$$Q_1 := \{[a_1, \ldots, a_n, b_1 : a_1, b_2, \ldots, b_m] \mid [a_1, \ldots, a_n] \in R \wedge [a_1, b_2, \ldots, b_m] \in S\}$$

b) $Q_2 := R \bowtie_{a_1=b_1} S$

Formulierung im Tupelkalkül

$$Q_2 := Q_1 \cup \{[r.a_1, \ldots, r.a_n, b_1 : \text{null}, \ldots, b_m : \text{null}] \mid \ r \in R \wedge \nexists\ s \in S(r.a_1 = s.b_1)\}$$

Formulierung im Domänenkalkül

$$Q_2 := Q_1 \cup \{[a_1, \ldots, a_n, b_1 : \text{null}, \ldots, b_m : \text{null}] \mid [a_1, \ldots, a_n] \in R \wedge \nexists\ c_2, \ldots, c_m([a_1, c_2, \ldots, c_m] \in S)\}$$

c) $Q_3 := R \rtimes_{a_1=b_1} S$

Formulierung im Tupelkalkül

$$Q_3 := \{s \mid \ s \in S \ \wedge \exists\ r \in R(r.a_1 = s.b_1)\}$$

Formulierung im Domänenkalkül

$$Q_3 := \{[b_1, \ldots, b_m] \mid [b_1, \ldots, b_m] \in S \wedge \exists\ a_2, \ldots, a_n([b_1, a_2, \ldots, a_n] \in R)\}$$

d) $Q_4 := R \ltimes_{a_1=b_1} S$

Formulierung im Tupelkalkül

$$Q_4 := \{s \mid \ s \in S \wedge \nexists\ r \in R(r.a_1 = s.b_1)\}$$

Formulierung im Domänenkalkül

$$Q_4 := \{[b_1, \ldots, b_m] \mid [b_1, \ldots, b_m] \in S \wedge \nexists\ a_2, \ldots, a_n([b_1, a_2, \ldots, a_n] \in R)\}$$

Hausaufgabe 2

Finden Sie die *Professoren*, deren sämtliche *Vorlesungen* nur auf selbst gelesenen (direkten) Vorgängern aufbauen. Formulieren Sie die Anfrage im Tupel- oder Domänenkalkül.

Lösung: Gesucht sind die Professoren, deren sämtliche Vorlesungen nur auf selbst gelesenen Vorgängern aufbauen. Damit sind im Ergebnis auch Professoren enthalten, die keine Vorlesungen oder nur Vorlesungen ohne direkte Vorgänger lesen.

Formulierung im Tupelkalkül

$$\{p \mid p \in \text{Professoren}$$
$$\wedge \neg\exists\ v_1 \in \text{Vorlesungen}(v_1.\text{gelesenVon}=p.\text{PersNr}$$
$$\wedge \exists\ o \in \text{voraussetzen}(v_1.\text{VorlNr}=o.\text{Nachfolger}$$
$$\wedge \exists\ v_2 \in \text{Vorlesungen}(o.\text{Vorgänger}=v_2.\text{VorlNr}$$
$$\wedge v_2.\text{gelesenVon} \neq p.\text{PersNr})))\}$$

Formulierung im Domänenkalkül

$$\{[p,n] \mid \exists\ rg,ra\ ([p,n,rg,ra] \in \text{Professoren} \wedge ($$
$$\forall\ na,t1,sws1\ ([na,t1,sws1,p] \in \text{Vorlesungen} \Rightarrow ($$
$$\forall\ vo\ ([vo,na] \in \text{voraussetzen}) \Rightarrow$$
$$\exists\ t2,sws2\ ([vo,t2,sws2,p] \in \text{Vorlesungen})))))\}$$

Hausaufgabe 3

Formulieren Sie folgende Anfragen auf dem bekannten Universitätsschema in SQL. Geben Sie alle Ergebnisse duplikatfrei aus.

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

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

(c) Finden Sie die *Assistenten* von *Professoren*, die den Studenten Carnap unterrichtet haben – z.B. als potentielle Betreuer seiner Bachelorarbeit.

(d) Geben Sie die Namen der *Professoren* an, die Theophrastos aus *Vorlesungen* kennt.

(e) Welche *Vorlesungen* werden von *Studenten* im Bachelorstudium (1. – 6. Semester) gehört? Geben Sie die Titel dieser *Vorlesungen* an.

(f) Bestimmen Sie für jede Vorlesung wie viele Studenten diese hören. Geben Sie auch Vorlesungen ohne Hörer aus. Sortieren Sie das Ergebnis absteigend nach Anzahl der Hörer.

Lösung:

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

```
select distinct 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';
```

(b) 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';
```

(c) Finden Sie die *Assistenten* von *Professoren*, die den Studenten Carnap unterrichtet haben – z.B. als potentielle Betreuer seiner Bachelorarbeit.

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

(d) Geben Sie die Namen der *Professoren* an, die Theophrastos aus *Vorlesungen* kennt.

```
select distinct 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 ='Theophrastos';
```

(e) Welche *Vorlesungen* werden von *Studenten* im Bachelorstudium (1. – 6. Semester) gehört? Geben Sie die Titel dieser *Vorlesungen* an.

```
select distinct 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 6;
```

(f) Bestimmen Sie für jede Vorlesung wie viele Studenten diese hören. Geben Sie auch Vorlesungen ohne Hörer aus. Sortieren Sie das Ergebnis absteigend nach Anzahl der Hörer.

```
select v.VorlNr, v.Titel, count(h.MatrNr) as hoerer
from
  Vorlesungen v left outer join
  hoeren h on (v.VorlNr = h.VorlNr)
group by v.VorlNr, v.Titel
order by hoerer desc;
```

Hausaufgabe 4

Füren Sie die folgenden Änderungen am Datenbestand des bekannten Universitätsschemas in SQL aus. Stellen Sie sicher, dass Ihre SQL-Statements mit jeder beliebigen Ausprägung des Schemas funktionieren.

a) Alle Professoren, die den Rang C3 haben, werden auf den Rang C4 befördert. Setzen Sie dazu den Rang aller C3-Professoren auf C4.

b) Die Planetenbewegungen sind vollständig erforscht. Löschen Sie alle Assistenten mit diesem Fachgebiet.

c) Eine neue Vorlesung mit dem Namen „Grundlagen: Datenbanken“ mit der Nummer 5278 soll erstellt werden. Die Vorlesung wird von der Professorin Curie gehalten und hat die Vorlesung „Logik“ als Voraussetzung. Sie soll 4 SWS umfassen. Tragen Sie den Studenten mit der Matrikelnummer 28106 als Hörer der Vorlesung ein. Erstellen Sie alle notwendigen SQL-Statements.

Lösung:

a) Alle Professoren, die den Rang C3 haben, werden auf den Rang C4 befördert. Setzen Sie dazu den Rang aller C3-Professoren auf C4.

```
update Professoren set Rang = 'C4' where Rang = 'C3';
```

b) Die Planetenbewegungen sind vollständig erforscht. Löschen Sie alle Assistenten mit diesem Fachgebiet.

```
delete
from Assistenten
where Fachgebiet = 'Planetenbewegung';
```

c) Eine neue Vorlesung mit dem Namen „Grundlagen: Datenbanken“ mit der Nummer 5278 soll erstellt werden. Die Vorlesung wird von der Professorin Curie gehalten und hat die Vorlesung „Logik“ als Voraussetzung. Sie soll 4 SWS umfassen. Tragen Sie den Studenten mit der Matrikelnummer 28106 als Hörer der Vorlesung ein. Erstellen Sie alle notwendigen SQL-Statements.

```
insert into Vorlesungen
select 5278, 'Grundlagen:␣Datenbanken', 4, PersNr
from Professoren
where Name = 'Curie';

insert into voraussetzen
select VorlNr, 5278
from Vorlesungen
where Titel = 'Logik';

insert into hoeren values (28106, 5278);
```