

Prof. Dr. Guido Moerkotte

Email: moer@db.informatik.uni-mannheim.de

Alexander Böhm

B6, 29, Raum C0.08

68131 Mannheim

Telefon: (0621) 181-2585

Email: alex@pi3.informatik.uni-mannheim.de

Datenbanksysteme 1
Frühjahrs-/Sommersemester 20083. Übungsblatt
12. März 2008

Aufgabe 1

Eine Firma verwaltet ihre Mitarbeiter und Produktionsdaten mit Hilfe eines Datenbanksystems. Aus dem ER-Modell (des letzten Übungsblatts) wurde folgendes, relationales Schema erstellt.

- Standort(Name, geleitetVon)
- Mitarbeiter(PersNr, Name, arbeitetAnStandort)
- Zeitarbeiter(PersNr, Firma)
- Festangestellter(PersNr, EinstellungsDatum)
- Maschine(InventarNr, Hersteller)
- Schicht(Tag, Startzeit, geleitetVon, Bericht)
- Charge(Tag, Startzeit, Nummer, Stück)
- arbeitetAnMaschine(PersNr, Tag, StartZeit, InventarNr)

Geben sie Anfragen in SQL an, mit deren Hilfe die folgenden Ergebnisse ermittelt werden können.

Aufgabe 1 a)

Die Gesamtanzahl aller Mitarbeiter der Firma.

Lösung

```
select count(*)  
from mitarbeiter;
```

Aufgabe 1 b)

Die Namen aller Zeitarbeiter, die am Standort Walldorf arbeiten, aufsteigend sortiert.

Lösung

```
select m.name
from Zeitarbeiter z, Standort s, Mitarbeiter m
where z.PersNr = m.PersNr and m.arbeitetAnStandort = s.name
and s.Name='Walldorf'
order by m.name asc;
```

Aufgabe 1 c)

Alle Maschinen, an denen noch nie ein Festangestellter gearbeitet hat.

Lösung

```
select m.*
from Maschine m
where not exists(select * from Festangestellter f, arbeitetAnMaschine am
where f.persNr = am.persNr and am.InventarNr = m.InventarNr);
```

Aufgabe 1 d)

Die Namen aller Standorte, die mehr Zeitarbeiter als Festangestellte haben.

Lösung

```
select s.name
from Standort s
where
(select count(f.*)
from Festangestellter f, Mitarbeiter m1
where s.Name=m1.arbeitetAnStandort and m1.PersNr=f.PersNr)
<
(select count(z.*)
from Zeitarbeiter z, Mitarbeiter m2
where s.Name=m2.arbeitetAnStandort and m2.PersNr=z.PersNr);
```

Aufgabe 1 e)

Die Namen der Mitarbeiter der fleissigsten Schicht (also die Schicht, die in ihren Chargen die grösste Stückzahl produziert hat).

Lösung

```

create view outputProSchicht as (
  select c.tag, c.startzeit, sum(c.stueck)
  from charge c, schicht s
  where c.Tag = s.Tag and c.startzeit = s.startzeit
  group by c.tag, c.startzeit
);

select m.name
from mitarbeiter m, outputProSchicht s1, arbeitetAnMaschine am
where m.PersNr = am.PersNr and am.Tag = s1.tag and am.Startzeit = s1.startzeit
and not exists (select 1 from outputProSchicht s2 where s2.sum > s1.sum);

(drop view outputProSchicht;)

```

Aufgabe 1 f)

Für jeden Mitarbeiter soll die Produktivität geschätzt werden. Hierfür soll durchschnittliche Stückzahl, die von den Schichten, an denen der Arbeiter beteiligt war, produziert wurde als Kenngrösse genommen werden. Mitarbeiter, die noch nie an einer Maschine gearbeitet haben sollen mit einem Schnitt von 0 in der Statistik auftauchen.

Lösung

```

select result.name, avg(result.stueck) as schnitt
from (select m.Name, sum(c.Stueck) as stueck
      from Mitarbeiter m, arbeitetAnMaschine am, Charge c
      where m.PersNr=am.PersNr and c.Tag=am.Tag and c.Startzeit=am.Startzeit
      group by m.Name, c.Tag, c.Startzeit) as result
group by result.name
union
select m.name, 0 as schnitt
from mitarbeiter m
where not exists (select 1
                  from arbeitetAnMaschine am
                  where am.persnr = m.persnr);

```

Aufgabe 2

Gegeben sei das aus der Vorlesung bekannte Uni-Datenbankschema mit den folgenden Relationen:

- Professoren(PersNr, Name, Rang, Raum)
- Studenten(MatNr, Name, Semester)
- Vorlesungen(VorlNr, Titel, SWS, gelesenVon)

- Assistenten(PersNr, Name, Fachgebiet, Boss)
- voraussetzten(Vorgänger, Nachfolger)
- hören(MatNr, VorlNr)
- prüfen(MatNr, VorlNr, PersNr, Note)

Geben sie Anfragen in SQL an, mit deren Hilfe die folgenden Ergebnisse ermittelt werden können.

Aufgabe 2 a)

Die Namen aller Assistenten, die für Kopernikus arbeiten.

Lösung

```
select a.name
from assistenten a, professoren p
where a.boss = p.persNr and p.name = 'Kopernikus';
```

Aufgabe 2 b)

Die Namen aller Studenten, die einen Notendurchschnitt von mindestens 2.0 haben.

Lösung

```
select s.name
from studenten s, pruefen p
where s.MatrNr = p.MatrNr
group by s.matrnr, s.name
having avg(p.note)<=2.0;
```

Aufgabe 2 c)

Die Titel und Teilnehmeranzahl aller Vorlesungen, absteigend nach Teilnehmerzahl sortiert.

Lösung

```
select v.titel , count(*) as teilnehmerZahl
from vorlesungen v, hoeren h
where v.VorlNr = h.VorlNr
group by v.titel
order by teilnehmerZahl desc;
```

Aufgabe 2 d)

Der Professor, der die meisten Prüfungen abgehalten hat.

Lösung

```
with pruefProf(PersNr, Name, Anzahl) as(
select prof.persnr, prof.name, count(*) as Anzahl
from professoren prof, pruefen pruef
where prof.PersNr=pruef.PersNr
group by prof.persnr, prof.name)

select p.name
from pruefProf p
where p.Anzahl = (select max(p2.Anzahl) from pruefProf p2);
```

Aufgabe 2 e)

Die Namen der Studenten, die kürzer als 3 Semester studieren, aber schon mehr als drei Prüfungen bestanden haben (Note besser als 4).

Lösung

```
select s.name
from studenten s, pruefen p
where s.semester<3 and s.MatrNr=p.MatrNr and p.note<=4.0
group by s.name, s.matnr
having count(*)>3;
```

Aufgabe 2 f)

Die Namen der Studenten, die länger als 5 Semester studieren, aber weniger als drei Prüfungen bestanden haben (Note besser als 4).

Lösung

```
select s.name
from studenten s, pruefen p
where s.semester>5 and s.MatrNr=p.MatrNr and p.note<=4.0
group by s.name, s.matnr
having count(*)<3
union
select s2.name
```

from studenten s2
where s2.semester>5 **and not exists**(select 1 **from** pruefen p2 **where** p2.MatrNr=s2.MatrNr)

Aufgabe 3

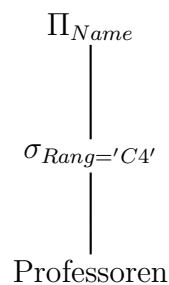
Finden Sie für die folgenden Aufgaben heraus welches Ergebnis der Anfragende vermutlich haben wollte. Überführen Sie die SQL-Anfragen anschliessend in einen (kanonischen) Operatorbaum.

Aufgabe 3 a)

```
select distinct p.name  
from Professoren p  
where p.Rang = 'C4';
```

Lösung

Ergebnis: Die Namen aller Professoren, die einen C4-Rang haben.



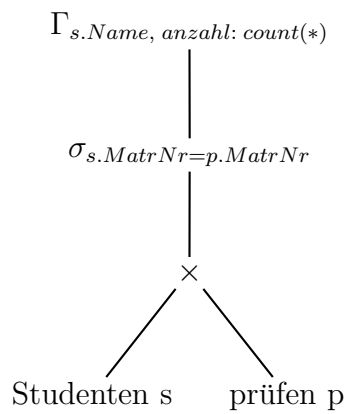
$\Pi_{Name}(\sigma_{Rang='C4'}(Professoren))$

Aufgabe 3 b)

```
select distinct s.Name, count(*) as Anzahl  
from Studenten s, pruefen p  
where s.MatrNr=p.MatrNr  
group by s.Name;
```

Lösung

Ergebnis: Die Namen und Anzahl der bisher abgelegten Prüfungen für alle Studenten, die mindestens eine Prüfung haben.



Aufgabe 3 c)

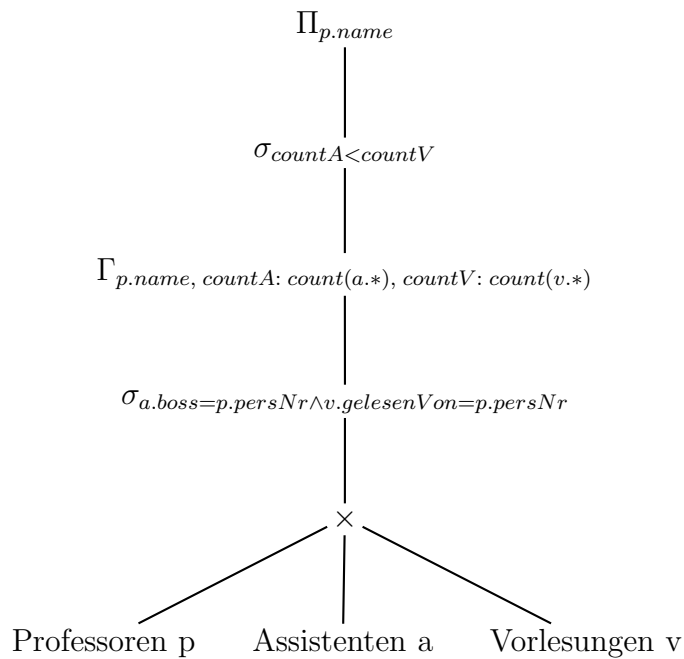
```

select distinct p.name
from professoren p
where
  (select count(*) as countA from assistenten where boss=p.PersNr)
< (select count(*) as countV from vorlesungen where gelesenVon=p.PersNr)

```

Lösung

Ergebnis: Die Namen aller Professoren, die mehr Vorlesungen halten als sie Mitarbeiter haben.



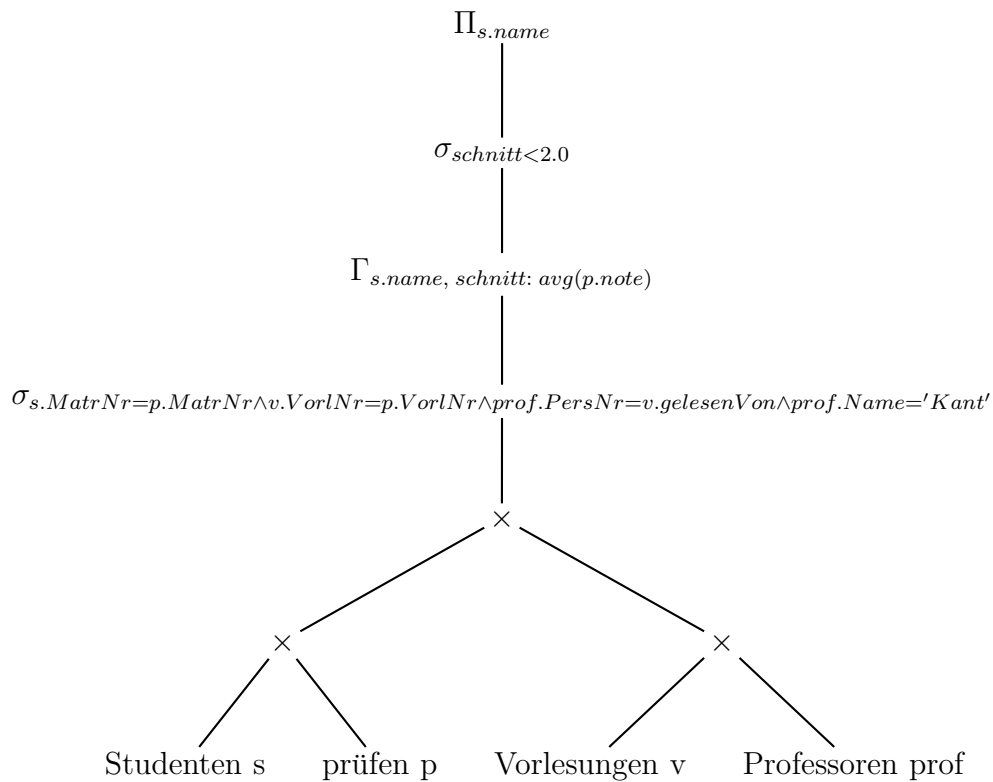
Aufgabe 3 d)

```

select distinct s.name
from studenten s, vorlesungen v, pruefen p, professoren prof
where s.MatrNr = p.MatrNr and v.VorlNr=p.VorlNr
       and prof.PersNr=v.gelesenVon and prof.Name='Kant'
group by s.name
having avg(p.note)<2.0
  
```

Lösung

Ergebnis: Die Namen aller Studenten, die sich von Prof. Kant haben prüfen lassen und in diesen Prüfungen einen Notendurchschnitt von besser als 2.0 erreicht haben.



Aufgabe 3 e)

select rang **from** professoren;

Lösung

Ergebnis: Für jeden Professor wird der Rang ausgegeben.
 Eine algebraische Darstellung ist nicht möglich, da das Ergebnis keine Menge ist.

Freiwillige Zusatzaufgabe 4

Installieren Sie ein relationales Datenbankmanagementsystem Ihrer Wahl auf Ihrem Computer (z.B. PostgreSQL).

Aufgabe 4 a)

Erstellen Sie für das relationale Schema in Aufgabe 1 zugehörige Tabellen mit Hilfe von SQL. Achten Sie hierbei auch auf die für die referentielle Integrität benötigten foreign

keys.

Lösung

Siehe Webseite.

Aufgabe 4 b)

Füllen Sie mit Hilfe von SQL die erzeugten Relationen mit Beispieldaten.

Lösung

Siehe Webseite.

Aufgabe 4 c)

Überprüfen Sie nun mit Hilfe Ihrer Datenbank die in Aufgabe 1 erarbeiteten Lösungen auf ihre Korrektheit.