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

Übung zur Vorlesung *Grundlagen: Datenbanken* im WS17/18

Harald Lang, Linnea Passing (gdb@in.tum.de)
http://www-db.in.tum.de/teaching/ws1718/grundlagen/

Blatt Nr. 06

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

Hausaufgabe 1

„Fleißige Studenten“: Formulieren Sie eine SQL-Anfrage, um die Studenten zu ermitteln, die mehr SWS belegt haben als der Durchschnitt. Berücksichtigen Sie dabei auch Totalverweigerer, die gar keine Vorlesungen hören.

Lösung:

Folgende SQL-Anfrage ermittelt die fleißigen Studenten:

```
select s.*
from Studenten s
where s.MatrNr in
  (select h.MatrNr
   from hoeren h join Vorlesungen v on h.VorlNr = v.VorlNr
   group by h.MatrNr
   having sum(SWS) >
      (select sum(cast(SWS as decimal(5,2)))/count(
         distinct(s2.MatrNr))
       from Studenten s2
          left outer join hoeren h2 on h2.MatrNr = s2.
             MatrNr
          left outer join Vorlesungen v2 on v2.VorlNr = h2
             .VorlNr));
```

Durch die Verwendung von **with** und **case** wird die Anfrage übersichtlicher:

```
with GesamtSWS as (
  select sum(cast(SWS as decimal(5,2))) as AnzSWS
  from hoeren h2, Vorlesungen v2
  where v2.VorlNr = h2.VorlNr
),
GesamtStudenten as (
  select count(MatrNr) as AnzStudenten
  from Studenten
)
select s.*
from Studenten s
where s.MatrNr in (
  select h.MatrNr
  from hoeren h join Vorlesungen v on h.VorlNr = v.VorlNr
  group by h.MatrNr
  having sum(SWS) > (select AnzSWS / AnzStudenten
                     from GesamtSWS, GesamtStudenten));
```

Alternativ:

```
with SWSProStudent as (
select s.MatrNr,
  cast((case when sum(v.SWS) is null
             then 0 else sum(v.SWS)
        end) as real) as AnzSWS
 from Studenten s
   left outer join hoeren h on s.MatrNr = h.MatrNr
   left outer join Vorlesungen v on h.VorlNr = v.VorlNr
 group by s.MatrNr
)

select s.*
from Studenten s
where s.MatrNr in (select sws.MatrNr
                   from SWSProStudent sws
                   where sws.AnzSWS > (select avg(AnzSWS)
                                       from SWSProStudent)
                                          );
```

Hausaufgabe 2

Was bringt der Vorlesungsbesuch? Finden Sie heraus, ob es für Prüfungen von Vorteil ist, die jeweiligen Vorlesungen auch gehört zu haben. Ermitteln Sie dazu die Durchschnittsnote der Prüfungen, zu denen die Studenten die Vorlesungen nicht gehört haben und die Durchschnittsnote der Prüfungen, zu denen sie die Vorlesungen gehört haben. - Formulieren Sie Ihre Antwort in SQL.

Lösung:

Diese Anfrage lässt sich auf zwei Arten beantworten. Zum einen kann ermittelt werden, wie das Verhältnis der Prüfungen für jede Vorlesung aussieht. Eine mögliche Formulierung hierfür ist folgende:

```
select ngehoert.VorlNr, ngehoert.ds, gehoert.ds
from (select p.VorlNr, avg(p.Note) as ds
      from pruefen p
      where not exists( select *
                        from hoeren h
                        where h.MatrNr = p.MatrNr
                        and h.VorlNr = p.VorlNr)
      group by p.VorlNr) ngehoert,
     (select p.VorlNr, avg(p.Note) as ds
      from pruefen p
      where p.VorlNr in (select h.VorlNr
                         from hoeren h
                         where h.MatrNr = p.MatrNr)
      group by p.VorlNr) gehoert
where ngehoert.VorlNr = gehoert.VorlNr;
```

Alternativ kann aber auch bestimmt werden, wie das Verhältnis der Prüfungsleistungen von gehörten zu nicht gehörten Vorlesungen sich insgesamt darstellt.

```
create view nichtgehoert as
  select avg(Note) as DnoteVLNichtGehoert
  from  pruefen p
  where not exists (select *
```

```
                    from hoeren h
                    where h.VorlNr = p.VorlNr
                      and h.MatrNr = p.MatrNr);

create view gehoert as
  select avg(Note) as DnoteVLGehoert
  from pruefen p
  where exists (select *
                from hoeren h
                where h.VorlNr = p.VorlNr
                  and h.MatrNr = p.MatrNr);
```

Da beide Views aus jeweils nur einem Wert bestehen, ergibt sich das Resultat der Anfrage als Kreuzprodukt:

```
select *
from nichtgehoert, gehoert;
```

Hausaufgabe 3

Welche Studenten haben alle Vorlesungen, die sie haben prüfen lassen, auch tatsächlich vorher gehört?

Lösung:

Die Anforderung, dass die Studenten im Anfrage-Ergebnis alle Vorlesungen, die sie haben prüfen lassen auch tatsächlich gehört haben, lässt sich umschreiben zu: „Es darf keine Vorlesung geben, die geprüft wurde, zu der es aber keinen Eintrag in *hoeren* gibt."

```
select s.*
from Studenten s
where not exists (select * from pruefen p
                  where s.MatrNr = p.MatrNr
                    and not exists (select *
                                    from hoeren h
                                    where h.MatrNr = s.
                                       MatrNr
                                      and h.VorlNr = p.
                                         VorlNr));
```

Hausaufgabe 4

„Frühestes Semester": Formulieren Sie eine SQL-Anfrage, um das Semester zu ermitteln in dem die Vorlesung "Der Wiener Kreis" frühestens gehört werden kann. Testen Sie die Anfrage auch mit anderen Vorlesungen, insbesondere mit „Logik".

Lösung:

```
with recursive voraussetzenRek(vorlnr, counter) as (
(
  select v.vorlnr, 1 as counter
    from vorlesungen v
   where v.titel = 'Der Wiener Kreis'
)
union
(
  select vs.vorgaenger, vsr.counter + 1 as counter
```

```
      from voraussetzenRek vsr, voraussetzen vs
     where vsr.vorlnr=vs.nachfolger
  )
)
select max(counter) as fruehestesSemester
  from voraussetzenRek;
```