Exercise for *Database System Concepts for Non-Computer Scientist* im WiSe 19/20

Alexander van Renen (renen@in.tum.de)
http://db.in.tum.de/teaching/ws1920/DBSandere/?lang=en

Sheet 09

Exercise 1

Write a query that determines the kind of degree a student is pursuing. In our database, we assume that this can be deduced from the student's semester in the following way: A student who has not reached her 7th semester yet is still considered a "bachelor student". Once in the 7th semester, she should be categorized as a "master student". Starting in the 11th semester, we label her as a "phd student".

Solution:

```
select s.studNr, s.name,
   case
      when s.semester < 7 then 'bachelor student'
      when s.semester < 11 then 'master student'
      else 'phd student'
   end) as degree
from Students s
```

Exercise 2

Answer the following questions on our university database using SQL:

a) Calculate how many lectures each student is attending. Students who do not attend any lecture should be included in the result as well (*attend_count* = 0). This time, use outer joins.

b) Figure out how many students are attending each lecture. Lectures which are not attended by any student should be included in the result as well (*attend_count* = 0). Again, use outer joins.

Solution:

a)
```
select s.studNr, s.name, count(a.studNr)
from Students s left outer join attend a
                     on s.studnr = a.studnr
group by s.studNr, s.name
```

b)
```
select l.lectureNr, l.title, count(a.lectureNr)
from Lectures l left outer join attend a on l.
   lectureNr = a.lectureNr
group by l.lectureNr, l.title
```

Exercise 3

„Busy Students": Write a SQL query to find all students that have more weekly hours in total than the average student has. Also consider students that do not attend any lecture.

Solution:

The following query determines the „busy students“:

```
select s.*
from Students s
where s.studNr in
  (select a.studNr
   from attend a, Lectures l
   where a.lectureNr = l.lectureNr
   group by a.studNr
   having sum(weeklyHours) >
      (select sum(cast(weeklyHours as decimal(5,2)))
              / count(distinct(s2.studNr))
       from Students s2
          left outer join attend a2
                      on a2.studNr = s2.studNr
          left outer join Lectures l2
                      on l2.lectureNr = a2.lectureNr));
```

Exercise 4

Find those students who have attended all lectures that they wrote a test in.

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.“

The requirement that students in the query result should have attended all lectures that they were tested in, can be rephrased as follow: “For a given student, there should be no test/exam, that has no entry in *attend*”.

This can then be translated into sql easily.

```
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)
                            );
```

This query is an example of a “for all query” where the counting-based technique can not be applied. The reason is that we can not simply count the number of attended lectures, because we need to make sure that the attended lectures match the ones that were tested.