Exercise for *Database System Concepts for Non-Computer Scientist* im WiSe 18/19

Alexander van Renen (renen@in.tum.de)
http://db.in.tum.de/teaching/ws1819/DBSandere/?lang=en

Sheet 10

Exercise 1

„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));
```

By using the **with** construct or **case**, we can write a query that is much easier to read. First with **with**:

```
with TotalWeeklyHours as (
  select sum(cast(weeklyHours as decimal(5,2))) as
     CountWeeklyHours
  from attend a, Lectures l
  where l.lectureNr = a.lectureNr
),
TotalStudents as (
  select count(studNr) as CountStudents
  from 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 CountWeeklyHours / CountStudents
                   from TotalWeeklyHours, TotalStudents));
```

And here with **case**:

```
with WeeklyHoursPerStudent as (
select s.studNr,
  cast((case when sum(l.weeklyHours) is null
             then 0 else sum(l.weeklyHours)
        end) as real) as CountWeeklyHours
 from Students s
   left outer join attend a on s.studNr = a.studNr
   left outer join Lectures l on a.lectureNr = l.lectureNr
 group by s.studNr
)

select s.*
from Students s
where s.studNr in (select weeklyHours.studNr
                   from WeeklyHoursPerStudent weeklyHours
                   where weeklyHours.CountWeeklyHours
                         > (select avg(CountWeeklyHours)
                            from WeeklyHoursPerStudent));
```

Exercise 2

ExamPoints			
StudName	ExerciseId	PossiblePoints	Score
Bond	1	10	4
Bond	2	10	10
Bond	3	11	4
Maier	1	10	4
Maier	2	10	2
Maier	3	11	3

Create a **view** in SQL for the *ExamResult*, which should look like the following for our example instantiation:

ExamResult				
Name	PossiblePoints	Score	Ratio	Passed
Bond	31	18	0,580645	yes
Maier	31	9	0,290323	no

An exam should be graded as passed if at least 50% of the possible points where scored.

[Bonus] Create the underlying table for *ExamPoints* and think about what the **primary key** should be.

Solution:

```
create table ExamPoints(studName varchar not null,
                        exerciseId int not null,
                        possiblePoints int not null,
                        score int not null,
                        primary key(studName,
                           exerciseId));
insert into ExamPoints values
   (’Bond’, 1, 10, 4), (’Bond’, 2, 10, 10),
   (’Bond’, 3, 11, 4), (’Maier’, 1, 10, 4),
   (’Maier’, 2, 10, 2), (’Maier’, 3, 11, 3);

create view ExamResult (Name, PossiblePoints, Score,
   Ratio, Passed) as (
select e.Name, sum(e.PossiblePoints) as PossiblePoints,
    sum(e.Score) as Score,
(cast (sum(e.Score) as float))/sum(e.PossiblePoints) as
    Ratio,
(case when (cast (sum(e.Score) as float))/sum(e.
   PossiblePoints) >= 0.5 then ’yea’ else ’no’ end) as
   Passed
from ExamPoints e
group by e.Name);
```