Exercise 1

Write a SQL statement to create a view that gives an overview of the difficulty of each lecture. The difficulty of a lecture is defined as the sum of the weekly hours of that lecture and its direct predecessors. In our example instantiation of the university schema, the following query on your view should yield the result (only partially shown):

```
select * from LectureDifficulties;
```

<table>
<thead>
<tr>
<th>lectureNr</th>
<th>title</th>
<th>difficulty</th>
</tr>
</thead>
<tbody>
<tr>
<td>5216</td>
<td>Bioethik</td>
<td>6</td>
</tr>
<tr>
<td>4630</td>
<td>Die 3 Kritiken</td>
<td>4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Solution:

```
create view LectureDifficulties(lectureNr, title, difficulty) as
(select l.lectureNr, l.titel, l.weeklyhours
 + (select (case when sum(l2.weeklyhours) is null then 0
       else sum(l2.weeklyhours) end)
       from Require r, Lectures l2
       where l.lectureNr = r.nachfolger
       and r.vorgaenger = l2.lectureNr)
from Lectures l);
```

Exercise 2

„Busy Students“: Find all students that have more weekly hours in total than the average student. Try to simplify the query using the with construct. (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 3**
Create a view in SQL for the ExamResult. An exam should be graded as passed if at least 50% of the possible points were scored. A view based on the example instantiation should look like the following (exercise continues on the next page):

<table>
<thead>
<tr>
<th>Name</th>
<th>PossiblePoints</th>
<th>Score</th>
<th>Ratio</th>
<th>Passed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bond</td>
<td>31</td>
<td>18</td>
<td>0.580645</td>
<td>yes</td>
</tr>
<tr>
<td>Maier</td>
<td>31</td>
<td>9</td>
<td>0.290323</td>
<td>no</td>
</tr>
</tbody>
</table>

Notes:

- Create the underlying table for ExamPoints and think about what the primary key should be.
- The ExamPoints relation does not exist in our web interface. If you want to try out your query, use your own database or you can try using the [https://www.db-fiddle.com/](https://www.db-fiddle.com/) website. Here is a template for this exercise [https://www.db-fiddle.com/f/m6a86jvHaGSTcxUHD2Lep9/0](https://www.db-fiddle.com/f/m6a86jvHaGSTcxUHD2Lep9/0). Note that any DDL and DML statements have to be entered on the left panel and DRL statements on the right.
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.StudName, 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.StudName;