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

***Database System Concepts for Non-Computer Scientist* - WiSe 24/25**

Alice Rey (rey@in.tum.de)

http://db.in.tum.de/teaching/ws2425/DBSandere/?lang=en

Sheet 05

Exercise 1

Answer the following questions on our university database using SQL:

a) Figure out the average semester of all students.

b) What is the average semester of students that are not attending any lecture?

c) Determine the average semester of students that attend at least one lecture of *Sokrates*.

d) Calculate how many lectures students are attending on average. Students who do not attend any lecture should be reflected in the result as well. If you get stuck, see hints: [1] [2]

e) Calculate how many lectures each student is attending. Students who do not attend any lecture should be included in the result as well ($attend\_count = 0$).

[1] Remember that the from clause is optional ('select 1.0 / 2.0;' is a valid query).

[2] Remember that you can use sub-queries in the select clause.

Solution:

a) Figure out the average semester of all students.

```
select avg(semester) from students;
```

b) What is the average semester of students that are not attending any lecture?

```
select avg(semester)
from students s
where not exists (
  select *
  from attend a
  where s.studnr = a.studnr)
```

Or:

```
select avg(semester)
from students s
where s.studnr not in (
  select a.studnr
  from attend a)
```

c) Determine the average semester of students that attend at least one lecture of *Sokrates*.

```
select avg(semester)
from students s
where exists (
  select *
  from attend a, lectures l, professors p
  where s.studnr = a.studnr
  and a.lecturenr = l.lecturenr
  and l.given_by = p.persnr
  and p.name = 'Sokrates')
```

In this query we need to make sure that each student is only counted once, even if she is attending two lectures by *Sokrates*. In our solution, the use of *exists* takes care of this. However, we could have also used *distinct* in combination with a sub-query:

```
select avg(semester)
from (select distinct s.*
      from Students s, attend a, lectures l, professors p
      where s.studnr = a.studnr
        and a.lecturenr = l.lecturenr
        and l.given_by = p.persnr
        and p.name = 'Sokrates')
```

d) Calculate how many lectures students are attending on average. Students who do not attend any lecture should be reflected in the result as well.

```
select attend_count/(student_count*1.000)
from (select count(*) as attend_count from attend) a,
     (select count(*) as student_count from students) s
```

Or:

```
select attend_count / cast(student_count as numeric(10,4))
from (select count(*) as attend_count from attend) a,
     (select count(*) as student_count from students) s
```

Or:

```
select ((select count(*) from attend) * 1.000)/ (select count(*)
    from students)
```

e) [Bonus] 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).

In this exercise we have to make sure to include students that do not attend any lecture.

```
select s.studnr, s.name, (select count(*) from attend a where a.
    studnr = s.studnr)
from students s;
```

Another possible solution would be to use *union*. We first calculate the number of attended lectures for each student that does attend a lecture. Then we create a query that produces the student number, student name and a zero for all students that do not attend a lecture. We then simply combine the two results using the *union* operator. Note, however, that it is important to only allow students that do not attend any lecture in the second sub-query. Otherwise, duplicates would be possible.

```
  (select s.studnr, s.name, count(*)
  from students s, attend a
  where s.studnr = a.studnr
  group by s.studnr, s.name)
union
  (select s.studnr, s.name, 0
  from students s
  where not exists (select * from attend a where a.studnr = s.
      studnr))
```

A similar approach that takes care of duplicates in a different way is shown in the following query. Here we do not avoid duplicates, but filter them out in a second step, instead.

```
select x.studnr, x.name, sum(x.cnt)
from
  ((
    select s2.studnr, s2.name, count(*) as cnt
    from students s2, attend a
    where s2.studnr = a.studnr
    group by s2.studnr, s2.name
  )
  union
  (
    select s1.studnr, s1.name, 0 as cnt
    from students s1
  )) x
group by x.studnr, x.name
```

As should be clear from this exercise, there are many different ways how a query can be written. As a rule of thumb, shorter queries are often better, because these are easier to understand. That holds for everyone involved: you yourself (when proof-reading your queries in the exam), other people (who read your queries and need to understand them) and the database (which has to execute your queries in an efficient manner).

Exercise 2

Answer the following questions on our university database using SQL:

a) Each assistants works in a certain area. Figure out how many assistants work in each area.

b) Determine the number of areas a professor is interested in. A professor is interested in an area if one of their assistants works in this area. Professors who do not have an assistant do not need to be included in the result.

c) Include the professors without assistants in the result.

Solution:

a) Each assistants works in a certain area. Figure out how many assistants work in each area.

```
select a.area, count(*)
from Assistants a
group by a.area
```

b) Determine the numbe r of areas a professor is interested in. A professor is interested in an area if one of their assistants works in this area. Professors who do not have an assistant do not need to be included in the result.

```
select p.persNr, p.name, count(distinct a.area)
from Professors p, Assistants a
where p.persNr = a.boss
group by p.persNr, p.name
```

c) Include the professors without assistants in the result.

```
(select p.persNr, p.name, count(distinct a.area)
 from Professors p, Assistants a
 where p.persNr = a.boss
 group by p.persNr, p.name)
union
(select p.persNr, p.name, 0
 from Professors p
 where p.persNr not in (select a.boss from Assistants a))
```

An alternative solution would be to use a subquery instead

```
select p.persNr, p.name, (select count(*) from assistants a where
     a.boss = p.persNr)
from Professors p
```