Database Systems – Autumn 2023
Teacher: Matteo Lissandrini – Exercise Sheet: SQL
Thanks to: Katja Hose, Gabriela Montoya

1 Creating Tables

Consider the following relational schema, which obviously does not describe the standard situation at Aalborg University.

We assume that tutors are responsible for one or multiple study groups, students individually (not per group) hand in solutions for exercises and receive individual grades in terms of the number of achieved points per sheet. Some of the tutors are more experienced (senior) than others.

student: $\{[\ \underline{\text{sid: int}}$, firstname: string, lastname: string, semester: int, birthdate: date $]\}$

tutor: $\{[\ \underline{\text{tid: int}}$, firstname: string, lastname: string, issenior: boolean $]\}$

studygroup: $\{[\ \underline{\text{gid: int}}$, tid $\rightarrow$ tutor, weekday: string, room: string, starttime: time $]\}$

exercise: $\{[\ \underline{\text{eid: int}}$, maxpoints: int $]\}$

handsin: $\{[\ \underline{\text{sid} \rightarrow \text{student, eid} \rightarrow \text{exercisesheet}}$, achievedpoints: int $]\}$

member: $\{[\ \underline{\text{sid} \rightarrow \text{student, gid} \rightarrow \text{studygroup}}\ ]\}$

Please formulate appropriate SQL statements to create these 6 tables using:

- Sequence number generators whose values automatically increase when data is inserted
- Appropriate data types
- Primary and foreign keys

Hint:
There might be more than one possible data type for some of the attributes.
Consult `https://www.postgresql.org/docs/current/datatype.html`.

2 Querying Tables I

Translate the following queries into equivalent SQL statements that run on the tables created above.

1. Find the different last names of the students whose first name is "Helle".
2. Find all the different last names of students that end with 'sen'.
   Hint: you can test if attribute test ends with 'xyz' by using "`test LIKE '%xyz'`"
3. List the first and last names of the tutors that are senior.
4. Find the first and last names of all students who have study group on Wednesday or Friday.

3 Querying Tables II

Considering the tables created above, identify the missing information that should go into the boxes to make the queries compute the requested information.

1. Find all the ids of the study groups without any members.

```
SELECT SG.gid
FROM studygroup SG
EXCEPT
SELECT [box 1]
FROM [box 2]
WHERE [box 3];
```

2. Find the first and last names of all students that have "Helle" as tutor.

```
SELECT S.firstname, S.lastname
FROM student S, [box 1], tutor T, [box 2]
WHERE T.firstname = 'Helle' AND T.tid = [box 3]
  AND S.sid= [box 4] AND [box 5];
```

3. Find the IDs of all students born before 01.03.1998 that have the same first name as one of the tutors that are not seniors.

```
SELECT [box 1] sid
FROM student, [box 2]
WHERE [box 3] < '1998-03-01'
   AND [box 4] AND [box 5] ;
```

4 Manipulating Tables

1. Populate the above created tables by inserting at least one valid tuple per table. Do foreign key impose any restriction?
2. Delete all student tuples from table student for which the first name is 'Tom'.

5 Using PostgreSQL

Test your solutions to the previous exercises using PostgreSQL.

Hint:
you might want to have a look at
https://www.moodle.aau.dk/mod/page/view.php?id=1577917
("Suggestions for Installation and usage of Postgres for the Self-Study" in Moodle) and
https://www.postgresql.org/docs/13/datatype.html
(data types supported by PostgreSQL).

Exploration and Experimentation:
Test what happens in PostgreSQL when you insert 2 students without specifying their "sid", then insert a new student specifying as "sid" the value 3,
then try insert other 2 students without specifying their "sid".
What is the content of the table?

Test what happens when you use "char(n)" instead of "varchar(n)" with shorter strings,
Test how equality works when your strings contain spaces in those cases.