Weekly exercises IN3020&4020 (2024)
Week 2: SQL Recap

Note: This exercise set is on paper, but feel free to experiment your queries with PostgreSQL. For your convenience, there is test data prepared for your experiments: `https://www-adm.uio.no/studier/emner/matnat/ifi/IN3020/v23/weekly_exercises/in3020_exercises_week2_test_data.sql`. Remember, however, that knowledge of the syntactic details of SQL will not be important during exam.

1 Basic SQL

Consider four relations with the following schemas (the domains and the meaning of the attributes are as expected):

Product [*manufacturer*, *model*, *type*]
PC [*model*, *speed*, *RAM*, *HDD*, *price*]
Laptop [*model*, *speed*, *RAM*, *HDD*, *screen*, *price*]
Printer [*model*, *colour*, *type*, *price*]

Write SQL queries that find the following:

1. the PC models that have speed at least 2.50;
2. the manufacturers that produce printers (i.e., products of type 'printer'), but not PC's;
3. the manufacturers that make laptops with a HDD of at least 500 GB;
4. the models and prices of all products made by manufacturer hp;
5. the manufacturers of at least two different computers (PCs or laptops); with speed at least 2.0 (hint: use a `WITH` statement in this query);
6. list the manufacturers with the number of black-and-white printer models and the number of colour printer models they produce; order by the number of colour printers (i.e., the result should be a table with 3 columns: *manufacturer*, *num_baw*, *num_colour*).

Give the examples of (states of) relations *Product* and *Laptop* so that

7. all their inner, left outer, right outer, and full outer joins on condition *Product.model* = *PC.model* are different;

8. all the joins are the same.

2 Queries in Lectures

Have another look at the recursive queries and queries with subqueries in the lectures and make sure you understand how they work (see slides 17, 24-26, 29 at `https://www.uio.no/studier/emner/matnat/ifi/IN3020/v24/lecture_slides/in3020_lecture_4.pdf`).

3 Subqueries in SQL

Assuming the schemas in Section 1, write two significantly different (but equivalent) SQL queries for each of the following, so that each of your queries should use at least one subquery with operators `EXISTS`, `IN`, `SOME`, `ALL`, `ANY`:

1. the manufacturers of laptops with speed at least 2.0;

2. the printers with the highest price;

3. the laptops whose speed is slower than that of the fastest PC;

4. the models (of PCs, laptops, and printers) with the lowest price;

5. the manufacturers of colour printers with the highest price.

4 Recursive SQL

1. Consider the following schema concerning scheduled flights for tomorrow (the domains and the meaning of the attributes are as expected):

$$\mathit{Flights}\,[\mathit{origin}, \mathit{destination}, \mathit{arrival_time}, \mathit{departure_time}]$$

Write a recursive SQL query that finds all pairs of cities (x, y) such that it is possible to get from x to y using a maximum of 5 connecting flights with at least one hour transfer time at each transfer.

2. Consider now the following schema concerning flight routes:

$$\mathit{Routes}\,[\mathit{origin}, \mathit{destination}]$$

Write a recursive SQL query that finds all pairs of cities (x, y) such that it is possible to get from x to y using a *minimum*(!) of 5 connecting flights (assuming that all routes are realised via frequent enough flights).

3(*). Do you think that there exists a recursion-free SQL query equivalent to the query in Question 1? In Question 2? Explain your answers.