SQL Exam No. 2 25.04.2024

For each of the following tasks, write appropriate SQL commands. We expect solutions in the form of one file containing **the content** of SQL commands, not the results of the query. Any queries with incorrect syntax will not be checked. Check your solution, for example, using `\i file.sql`! You can send the file multiple times; only the latest version will be checked. Remember that before executing a database-modifying operation, you can issue the command `BEGIN;` and then `ROLLBACK;` (if you want to revert all changes) or `COMMIT;` (if you want to save them).

Load the file `jjit.sql` into your database. Send your solutions through the form at `https://dbserv.stud.ii/`. Do this as often as possible! All data on your computers is erased after a restart. In case of any problems, be sure to contact the course instructor before restarting your computer.

The format of the first line of the solution: `-- group-firstname-lastname`, where `group` are the initials of the instructor leading your group (`jotop`/`mabi`/`plg`/`pwi`), e.g., `pwi-Jan-Kowalski`. The required format of the entire solution file:

```
-- group-firstname-lastname
-- Task 1
<query>

-- Task 2
<query>
...
```

Task 1 (5 points) Find companies that have remote job offers (at least one) with salary paid in PLN, but such that none of these offers allows for a permanent employment contract (value `permanent` in `employment_details.type`). As a result, output the id and name of the company along with the number of such offers. Sort the results in descending order by the number of offers, and then alphabetically by company name. Remote job offers are those that have the `remote` attribute set to `true`.

When counting offers, note that they may have more than one employment type; do not count the same offer multiple times.

The reference query returns 255 rows, including the tuple (19645, Inventively Ltd, 10).

Solution

```
SELECT c.id, c.name, count(DISTINCT o.id)
-- DISTINCT to avoid counting the same offer with multiple types: b2b and contracts
FROM company c JOIN
     offer o ON c.id = o.company_id JOIN
     employment_details e ON o.id = e.offer_id
WHERE
```

```
      o.remote AND
      e.currency = 'pln' AND
      c.id NOT IN (
         SELECT c.id
         FROM company c JOIN
              offer o ON c.id = o.company_id JOIN
              employment_details e ON o.id = e.offer_id
         WHERE e.type = 'permanent' AND
               o.remote AND
               e.currency = 'pln'
      )
GROUP BY c.id
ORDER BY 3 DESC, 1;
```

Task 2 (5 points) Consider job offers in Wrocław that allow for permanent employment contracts (value `permanent` in `employment_details.type`) with a positive minimum salary in PLN. We want to know the list of important skills in Wrocław, ie., such that each of the aforementioned offers requiring a certain skill guarantees a salary above the average calculated from the minimum salaries for those offers. For each such skill, return its name and the minimum salary occurring in those offers that require it.

Order the result first in descending order by the second column, and then alphabetically by company name.

The reference query returns 142 tuples, and the range of lower salary thresholds for the specified skills is from 14000 PLN to 36000 PLN.

Solution

```
WITH skill_min_salary AS (
  SELECT s.name, min(e.salary_from) AS min_salary
  FROM skill s JOIN
      offer o ON s.offer_id = o.id JOIN
      employment_details e ON o.id = e.offer_id
  WHERE
      e.salary_from > 0 AND
      e.currency = 'pln' AND
      e.type = 'permanent' AND
      o.city = 'Wrocław'
  GROUP BY s.name
)
SELECT name, min_salary FROM skill_min_salary
WHERE min_salary > (
    SELECT avg(e.salary_from)
```

```
    FROM offer o JOIN
         employment_details e ON o.id = e.offer_id
    WHERE
        e.salary_from > 0 AND
        e.currency = 'pln' AND
        e.type = 'permanent' AND
        o.city = 'Wrocław'
)
ORDER BY 2 DESC, 1;
```

Task 3 (4 points) We want to delete all companies that have more than 200 job offers, and we want to do it using a single command starting with `DELETE FROM company`. In the current state of the database, such a command would fail, so first replace existing constraints with new ones so that the offers, their details, and skills, referring to the deleted companies are automatically deleted as well.

When testing, remember to use `BEGIN` and `ROLLBACK`!

The reference query removes 3 tuples from the `company` table.

Solution

```
-- \d to check existing constraints

ALTER TABLE offer DROP CONSTRAINT offer_company_id_fkey;
ALTER TABLE employment_details DROP CONSTRAINT employment_details_offer_id_fkey;
ALTER TABLE skill DROP CONSTRAINT skill_offer_id_f
ALTER TABLE offer ADD FOREIGN KEY (company_id)
             REFERENCES company(id) ON DELETE CASCADE;
ALTER TABLE skill ADD FOREIGN KEY (offer_id)
             REFERENCES offer(id) ON DELETE CASCADE;
ALTER TABLE employment_details ADD FOREIGN KEY (offer_id)
             REFERENCES offer(id) ON DELETE CASCADE;

DELETE FROM company WHERE
company.id in (
    SELECT company_id
    FROM offer
    GROUP BY
        company_id
    HAVING count(*)>200
);
```