SQL Exam No. 1 18.04.2024

For each of the following tasks, write appropriate SQL commands. We expect solutions in the form of a file containing **SQL command texts**, not their results. Any syntactically incorrect queries will not be graded — 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 `BEGIN;` command, and then `ROLLBACK;` (if you want to undo all changes) or `COMMIT;` (if you want to save them).

Load the file `jjit.sql` into your database. Send your solutions via the form at `https://dbserv.stud.ii/`. Do it as often as possible! All data on your computer 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: `-- mabi-firstname-lastname`. The required format of the entire solution file:

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

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

Task 1 (2 points) List all the cities where the company 'Siepomaga.pl' offers job opportunities. Sort the results alphabetically and ensure that the returned rows are unique.

The reference query returns 5 tuples.

Solution

```
SELECT DISTINCT o.city
FROM offer o JOIN
     company c ON c.id=o.company_id
WHERE
     c.name='Siepomaga.pl'
ORDER BY 1;
```

Task 2 (2 points) Provide the names of companies and job positions offered in Warsaw, where one of the requirements is knowledge of the Kotlin language (according to `skill.name`), with the importance of this skill below the value of 5 (`skill.value`). Keep in mind that the skill name may be written in mixed case (lowercase, uppercase), and the word 'Kotlin' may only be part of the skill name. Sort the results alphabetically first by company name, and then by job position.

The reference query returns 45 tuples.

Solution

```
SELECT c.name, o.title
FROM skill s JOIN
     offer o ON (s.offer_id=o.id) JOIN
     company c ON (o.company_id = c.id)
WHERE s.name ILIKE '%kotlin%' AND
      s.value<5 AND
      o.city = 'Warszawa'
ORDER BY 1, 2;
```

Task 3 (2 points) Provide the names of companies (without repetitions) that do not offer any remote job offers (`offer.remote`). Sort the results alphabetically by company name.

The reference query returns 458 tuples.

Solution

```
SELECT DISTINCT c.name
FROM company c LEFT JOIN
     offer o ON (c.id = o.company_id AND o.remote)
WHERE o.id is NULL
ORDER BY 1;

-- or

SELECT DISTINCT c.name
FROM company c
WHERE c.id NOT IN
  (
     SELECT company_id FROM offer WHERE remote
  )
ORDER BY 1;
```

Task 4 (4 points) Some people are annoyed that there are companies that do not provide salary ranges in their advertisements, or the provided range is very wide. Let's try to investigate this phenomenon. Consider only offers containing salary ranges for B2B contracts published on September 1, 2023, stating the salary in Polish złoty (PLN), where the lower threshold is nonzero. List tuples consisting of: company name `company.name`, job title `offer.title`, experience level `offer.experience_level`, minimum salary `employment_type.salary_from`, maximum salary `employment_type.salary_to`, the range between the maximum and the minimum, and what fraction of the minimum salary this range constitutes. Sort the results by the last value in ascending order, and then alphabetically by company name. Round the percentages and the ranges to the nearest integer (see the documentation on mathematical functions).

Ensure that the returned rows are unique.

Solution

```
SELECT DISTINCT c.name, o.title, o.experience_level,
       e.salary_from, e.salary_to,
       round(e.salary_to - e.salary_from) AS diff,
       round(100*(e.salary_to-e.salary_from)
                  /e.salary_from) AS "%"
FROM offer o JOIN
     company c ON c.id = o.company_id JOIN
     employment_details e ON e.offer_id= o.id
WHERE e.salary_from>0 AND
      e.currency='pln' AND
      e.type = 'b2b' AND
      DATE(o.published_at) = '2023-09-01'::date
ORDER BY 7,1;
```

The reference query returns 485 tuples, the smallest percentage is 0, and the largest exceeds 100%.

Task 5 (3 points) Create a table `salaries_abroad` with a schema identical to `employment_details`. Add a column `country_code` of type `text` to it. Ensure that `salaries_abroad` has a primary key and a foreign key analogous to `employment_details`.

You can perform the task using more than one command.

Task 6 (1 point) Insert into the `salaries_abroad` table all the tuples from `employment_details` that concern job offers outside Poland, don't forget about the country code. *The reference query returns 276 tuples.*

Solution

```
CREATE TABLE salaries_abroad(LIKE employment_details);
ALTER TABLE salaries_abroad ADD COLUMN country_code text;

ALTER TABLE salaries_abroad ADD PRIMARY KEY (offer_id, type);
ALTER TABLE salaries_abroad ADD FOREIGN KEY (offer_id) REFERENCES offer(id);

INSERT INTO salaries_abroad
   SELECT e.*, o.country_code
   FROM employment_details e JOIN
```

```sql
     offer o ON e.offer_id = o.id
WHERE country_code != 'PL';
```