Mathematisch-Naturwissenschaftliche Fakultät

Wilhelm-Schickard-Institut für Informatik

Datenbanksysteme · Prof. Dr. Grust

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Datenbanksysteme I

WS 2021/22

Torsten Grust, Christian Duta

Assignment #9

Submission Deadline: January 12, 2021 - 10:00

Please note that the response times in our forum will likely be longer during the semester break. Assignment #10 will be published on January 12, 2021. **We wish you a merry Christmas and a happy new year!**

Exercise 1: GROUP BY and HAVING (12 Points)

1. Consider this schema

   ```
   CREATE TABLE r(a int, b int);
   ```

   and the following two queries[1]:

   ```
   SELECT r.a, COUNT(*) AS c        SELECT r.a, COUNT(*) AS c
   FROM   r AS r                    FROM   r AS r
   WHERE  r.b = 4                   GROUP BY r.a
   GROUP BY r.a;                    HAVING EVERY(r.b = 4);
   ```

   Create an instance for `r` for which the two queries yield different results. Hand in the DML statements. **Note:** The `EVERY` aggregate function is synonymous to `bool_and` which was previously discussed in the lecture.

2. Consider the following query:

   ```
   SELECT r.a, MAX(r.b) AS m
   FROM   r AS r
   GROUP BY r.a
   HAVING COUNT(*) > 2;
   ```

   Write an equivalent query without `HAVING`.

[1] Various aggregate functions are explained at https://www.postgresql.org/docs/current/functions-aggregate.html.

3. On the university database[2], find all MSc students whose workload (= number of courses taken) exceeds two.
   Output: a table with two columns `name` of type `text` and `workload` of type `bigint`.

4. Another task on the university database: Find the names of all staff members whose courses are attended by less than five students or whose courses are entirely unattended. Assume that staff member names are unique. Students enrolled in more than one course count towards the attendance of *each* their courses. We propose the following faulty solution:

```
SELECT f.staff_name
FROM   enrolled AS e, class AS c, staff AS f
WHERE  e.enrolled_class_id = c.class_id
AND    c.class_staff_id    = f.staff_id
GROUP BY f.staff_name
HAVING COUNT(*) < 5;
```

   Run this query and explain why the query is incorrect. Suggest a correct alternative.

Exercise 2: Functional Dependencies (9 Points)

1. Consider the following *functional dependencies* on relation $R(A, B, C, D, E)$:

$$\begin{aligned} ABC &\to D \\ E &\to D \\ D &\to A \\ D &\to B \end{aligned}$$

   (a) **Calculate the cover** $\{C, E\}^+$ based on the given functional dependencies. List the **intermediate results** of each iteration of the cover algorithm and hand them in.

   (b) Is relation R in **Boyce-Codd normal form**? Explain briefly.

2. Consider the following functional dependencies on relation $R(A, B, C, D, E, F)$ below:

$$\begin{aligned} AB &\to C \\ B &\to DF \\ BE &\to F \\ A &\to F \end{aligned}$$

   (a) **Hand in the five rows below with all gaps filled in** with integer values. These values must not violate the given functional dependencies:

A	B	C	D	E	F
1	1	☐	☐	1	1
2	1	2	☐	3	☐
3	2	3	1	1	1
1	1	1	1	2	☐
4	2	2	1	1	1

   (b) Will the insertion of row (1, 1, 1, 2, 1, 1) into the above table violate the given functional dependencies? Explain briefly.

[2]The university database schema and instance were introduced in Assignment #6 and provided to you again as `uni.zip`.

Exercise 3: Boyce-Codd Normal Form (9 Points)

Table `delivery` stores information about the deliveries of a logistics company to their clients. A delivery is done by a driver using a vehicle. Clients are assigned a unique identifier and their address is stored. For each delivery, the table stores date, time and the delivery status, as well as identifier and name of the driver and the designation of the vehicle. A driver does not change the vehicle during the course of a day. A client will receive only one delivery per day, but a driver usually delivers to multiple clients per day.

delivery

clientid	address	date	time	status	driverid	drivername	vehicle
17	Sand 13, 72076 Tü	19.08.08	17:00	OK	4711	D. Unlop	MB 104
17	Sand 13, 72076 Tü	14.10.08	09:00	keiner da	1508	M. Ichelin	FT 8
17	Sand 13, 72076 Tü	29.10.08	11:00	abgelehnt	1508	M. Ichelin	MAN 37
19	Wilhelmstr. 5, 72076 Tü	02.04.08	08:30	OK	1508	M. Ichelin	FT 8
19	Wilhelmstr. 5, 72076 Tü	10.05.08	14:30	Reklamation	4711	D. Unlop	MAN 37
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

The following functional dependencies hold:

$$
\begin{aligned}
\text{clientid, date} &\rightarrow \text{address, time, status, driverid, drivername, vehicle}\\
\text{date, time, vehicle} &\rightarrow \text{clientid, address, status, driverid, drivername}\\
\text{date, time, driverid} &\rightarrow \text{clientid, address, status, drivername, vehicle}\\
\text{clientid} &\rightarrow \text{address}\\
\text{driverid} &\rightarrow \text{drivername}\\
\text{driverid, date} &\rightarrow \text{vehicle}
\end{aligned}
$$

This table (in 1NF) obviously contains quite a lot of redundancy, leading to anomalies. Construct an equivalent schema in Boyce-Codd Normal Form using the decomposition algorithm (slide 25, chapter 10 "Functional Dependencies").

1. Hand in the complete split tree (as shown in the lecture video) produced by function `split` in plain text format. Make sure to include
   - the FD along which the next split is performed,
   - the two relation schemata (keys underlined) resulting from each split,
   - the set of FDs still applicable in the resulting relations.
2. Provide reasonable names for the BNCF relations at the leaves of the split tree.