DAT151
Database and Unix System Management

Spring 2018

Assignment 5

Obligatory assignment. Deadline: Thursday 15.02.2018

The report should include all necessary commands to complete the tasks, printout from the system, explanation of what is done, the result and explanation of the result.

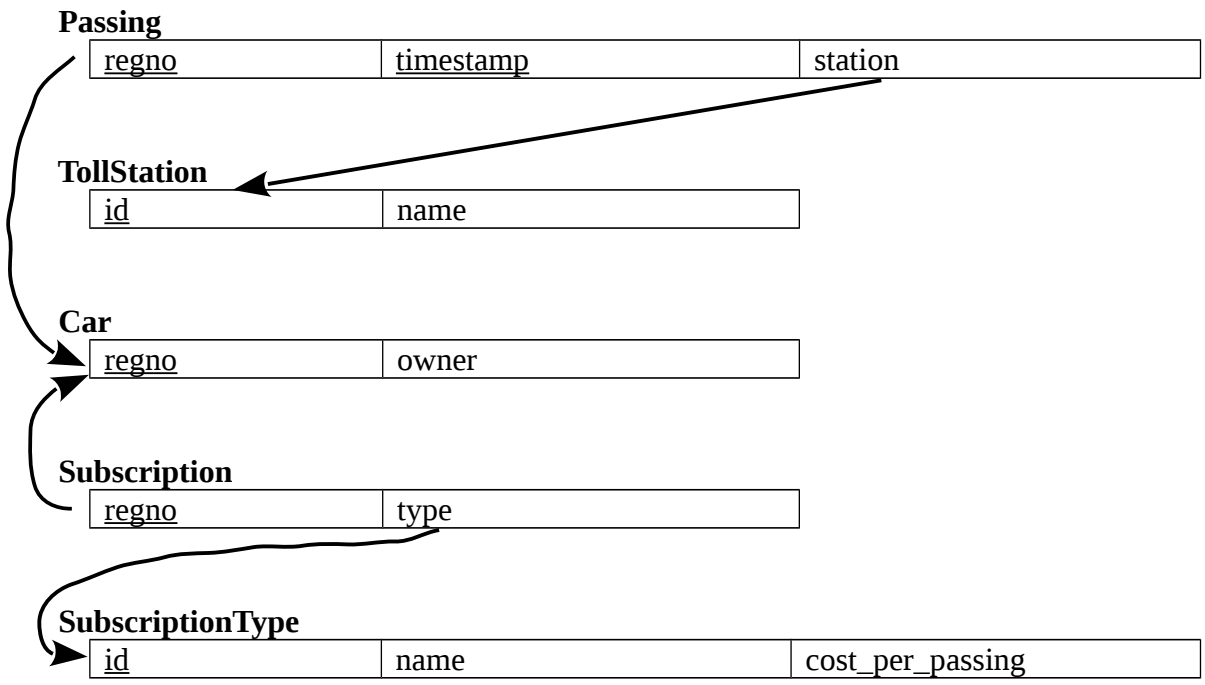
Remember to include the names of the group members on the front page of the report. The report can be in English or Norwegian. The report should be handed in via it's learning.

The assignment should be accomplished in groups of three or four students.

Lecturers:
Bjarte Kileng, Bjarte.Kileng@hib.no, room E418
Maksim Melnik Storetvedt, Maxim.Storetvedt@hvl.no, room E506
Faustin Ahishakiye, Faustin.Ahishakiye@hvl.no, room E506
Violet Ka I Pun, Violet.Ka.I.Pun@hvl.no

Task 1: Normal Forms

In this and the following tasks, you will optimize a database for the execution of a given set of queries. Make sure to read the entire assignment before you start. The following logical database schema is given:



An EER Diagram of the above for MySQL Workbench can be found in the attached file *car_passing.mwb*. Note that the file is only for reducing possible confusions, and you still need to edit it to satisfy the requirements.

Answer the following (explain why, not just yes/no):

1. Is this schema in 1NF?
2. Is it in 2NF?
3. Is it in 3NF?

Task 2: Create a physical schema and a test environment

Unless explicitly required, all work must be done as a normal database user (i.e. not root).

You will now implement the schema from the previous task in your MariaDB database and fill it with test data before we in the next task will try to optimize the performance.

1. Implement the schema from the previous task in your MariaDB database.
2. Fill the database with test data. We need some amount of test data before we can start tuning the performance. As a minimum, INSERT the following data into the database:
   - 10 rows in **TollStation**,
   - 6 rows in **SubscriptionType**,
   - 3,000 rows in **Car** and **Subscription**, and
   - 100,000 rows in **Passing**.

   You may need to create a routine (e.g. a procedure) to fill the data in the database.

   Reference: http://dev.mysql.com/doc/refman/5.7/en/stored-routines.html.

Task 3: Performance

Unless explicitly required, all work must be done as a normal database user (i.e. not root).

Consider the following SQL queries:

a) Retrieve a list over car owners and time of passing:

```
SELECT SQL_NO_CACHE c.owner, p.timestamp
  FROM Car c, Passing p WHERE p.regno = c.regno;
```

b) Show how much each car owner has used on toll booths in total:

```
SELECT SQL_NO_CACHE c.owner, Sum(t.cost_per_passing)
  FROM Car c, Passing p, Subscrition s, SubscriptionType t
  WHERE c.regno = s.regno AND p.regno = c.regno
    AND s.type = t.id GROUP BY c.owner;
```

c) Show all the car owner that passed the toll station in Sandviken:

```
SELECT SQL_NO_CACHE c.owner FROM Car c WHERE c.regno
  IN (SELECT p.regno FROM Passing p, TollStation s
  WHERE p.station = s.id AND s.name LIKE 'Sandviken' );
```

First, carry out the following and document the input and output:

- Turn on the profiling, run the queries, and then use SHOW PROFILE(S) to see the statistics.
- Use EXPLAIN to see how indexes, JOINs, and subqueries are handled by the optimizer.

Now, for each of the queries, try to improve performance by the following (restore to the original schema after each test):

1. Denormalization:

   Describe the denormalization you do, and explain what normal forms it violates. You will need to implement mechanisms to make sure that any redundancy is maintained correctly without any risk of loss of data integrity.

2. Using indexes:

   Create indexes and specify index hints (use index, ignore index, force index, ...), if necessary, to influence the execution of the queries.

Use profiling and EXPLAIN to show that your optimization improves performance. If not, or if there is no need to do any optimization in a case, then simply write down the reasons.

Last changed, 23.01.2018 (BK)