Case Study

The assignments for lab work are based on the Taxi database. Don't worry it is quite easy to understand as you can see in figure 1.

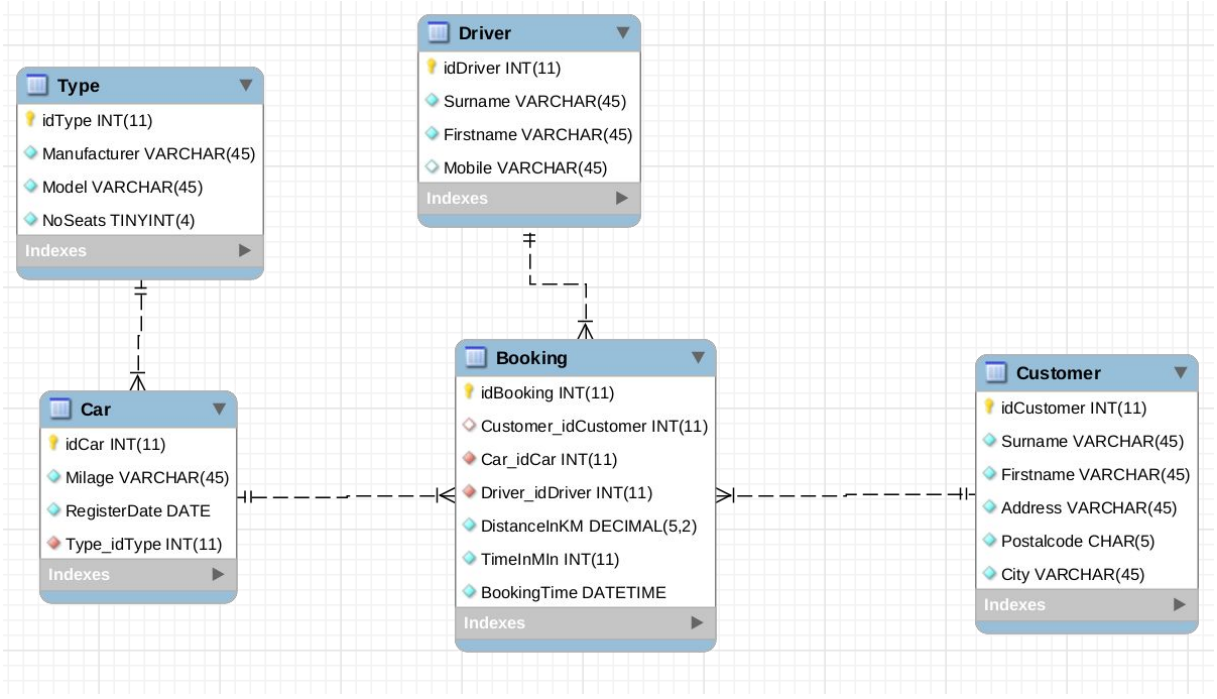

figure 1: Lab Work Data Model

Preparation

In order to be able to solve the tasks in the assignment the database has to be imported in your VM. This can be done by firstly downloading the SQL file from Moodle, then create a database called ***userXX_taxi*** and afterwards importing it using PHPMyAdmin or MySQL Workbench.

Solution Format

Please hand in an SQL file containing your solution for each of the assignments. The document should contain your name(s) and matriculation number(s). Additionally, upload a PDF version of your SQL file so that it can be commented in Moodle.

Assignments

For each of the following assignments the relevant SQL code has to be documented and described.

1. Create a stored procedure ***generate_driver_statistics*** that creates a table (if not exists) that stores *Year, Week, idDriver* and the *number of bookings*. Furthermore, the procedure should compute the data (using an appropriate aggregate function) for a specified week and store it into the newly created table. As a parameter the ***year*** and the ***week of the year*** have to be handed over to the procedure.
   Hints:
   - It makes sense to use MySQLs Date and Time Functions.
   - Make use of INSERT INTO…SELECT.
   - Choose Week 22 and year 2019 for testing.

2. Create an event ***exec_driver_statistics*** that executes the procedure described in assignment 1 every Monday at 0:00 am. The event should start at October the 28th 2019.
   Hints:
   - The function *NOW()* returns the current timestamp.

3. Create a function ***customer_class*** that groups a given customer into the categories A, B or C depending on the distance she has travelled so far. The classes are as follows:
   - 'A' customers have a total distance that is at least 85% of the maximum distance
   - 'B' customers have a distance that is above 40% of the average distance of all customers.
   - 'C' customers neither fall into category 'A' nor into 'B'.

   Hints:

   - In order to calculate the relevant figures the following view may be used:

```
CREATE VIEW travel_statistic AS(
      SELECT idCustomer, Surname, Firstname,
      SUM(DistanceInKM) AS SumDistance
      FROM Customer c JOIN Booking b ON
      b.Customer_idCustomer = c.idCustomer
      GROUP BY idCustomer
);
```

   - The standard aggregation functions AVG() and MAX() might help.