

**Databases Project**

Business Informatics – Data Science

Students: Philipp Becht

Pascal Schmidt

Simon Wrigg

Course: WWI19DSB

Lecturer: Petko Rutesic

**Project Description**

The goal of the project was to foster the basic skills to design and implement a relational database. First, we had to prepare a preliminary specification, which contains the general setup of our database. The group decided to design a database for a gym-company, which keeps track of their members, trainers, locations, and more.

Based on the preliminary specification, we created an ER-Diagram, that represents all relations between the entities as well as important indications, such as the foreign key indication.

**Preliminary Specification**

The gym keeps track of each members first and last name, birthdate, email-address, sex and the chosen subscription. Furthermore, it contains the start date and end date of contract of each member, as well as the gym-location and, if chosen, also the ID of a personal trainer. We allocate each member an unique Member\_id.

For the gym’s trainers, the database manages similarly to the member table some personal data such as first name, last name, birthdate, sex and email-address. We allocate each trainer an unique Trainer\_id. Some trainers are responsible for different courses with a Course\_id. Each trainer is based in a specific gym location with an unique Base\_id.

Each offered course has a specific Course\_id. The database contains information about the course-name, when the course takes place and its duration. Also, every course takes place in a specific room marked with a Room\_id.

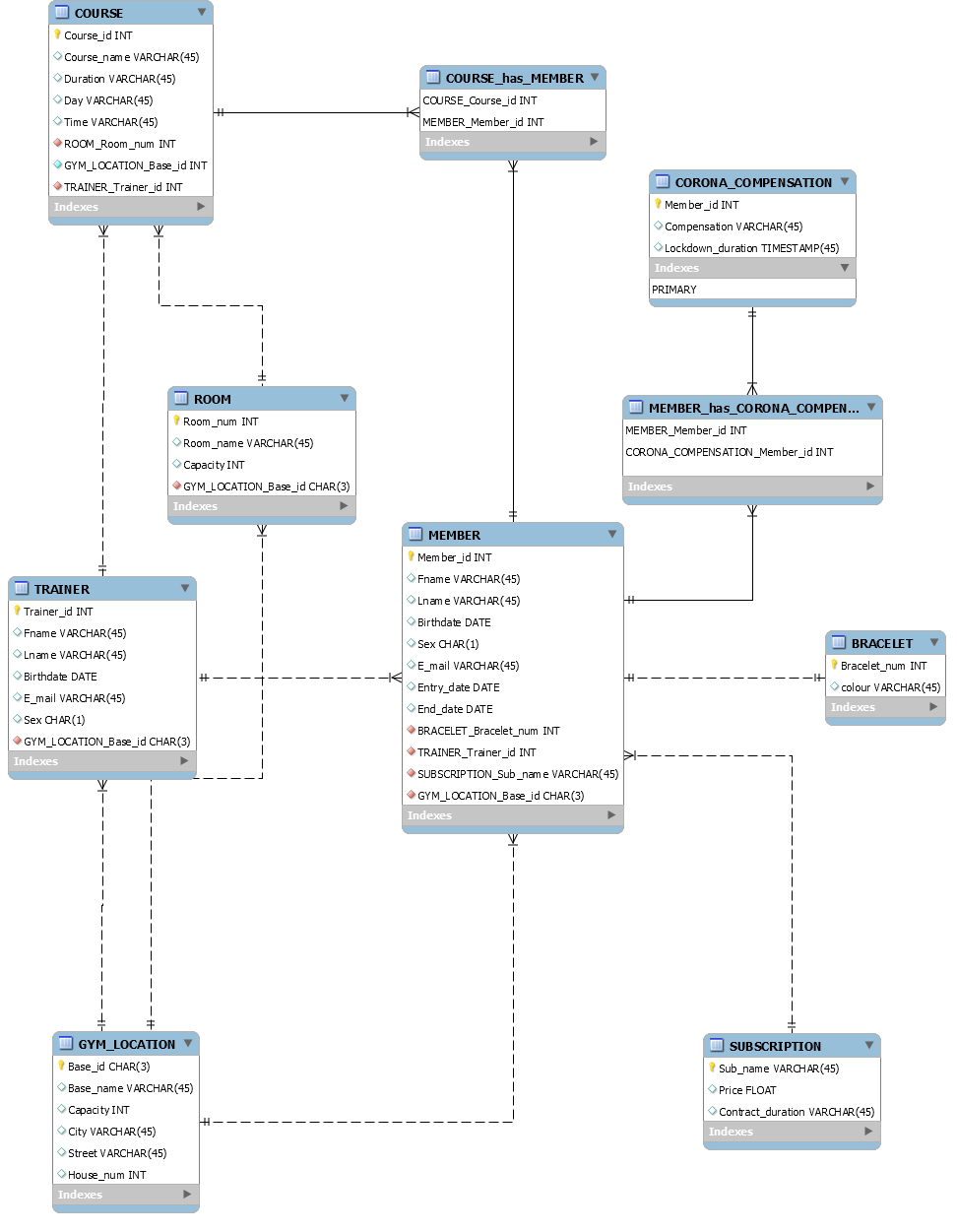
In the fitness center, the members can choose between different subscriptions. Therefore, a table keeps track of the subscription name, its monthly price as well as the duration of the contract (e.g. 1 year). Depending on the certain subscription, members have access to specific rooms in the gyms (e.g. spa). Therefore, the assigned Room\_ids are stored in the subscription table.

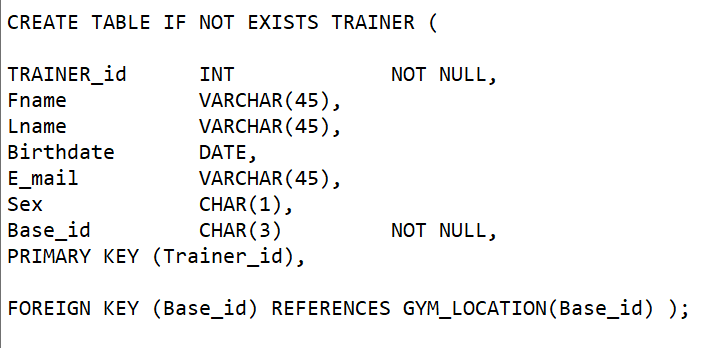
As already mentioned, each room has a unique Room\_id as well as a room name. In the gyms are different rooms, some rooms are used for courses, others for e.g. spa or sauna. Every room has an indication for its capacity, e.g. a maximum amount of 10 people.

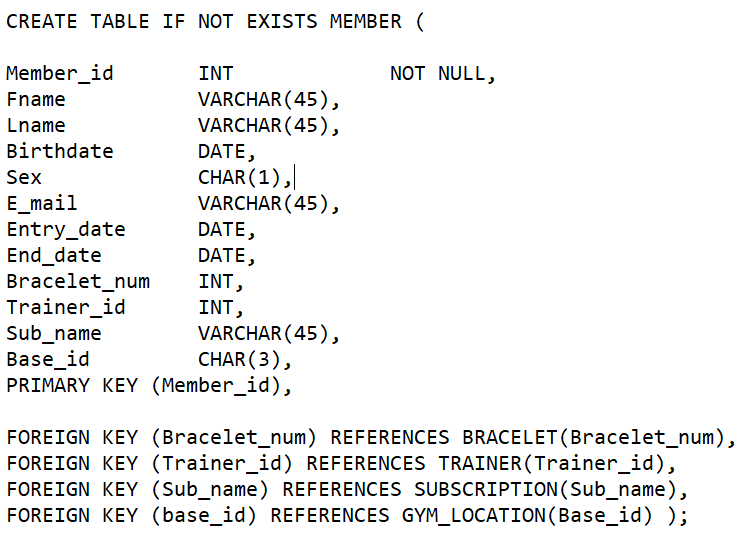
Every gym of the company chain is stored in a location-table, where the Base\_id, the gym’s name, its address and also its capacity are represented.

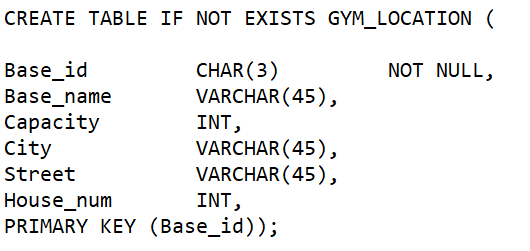
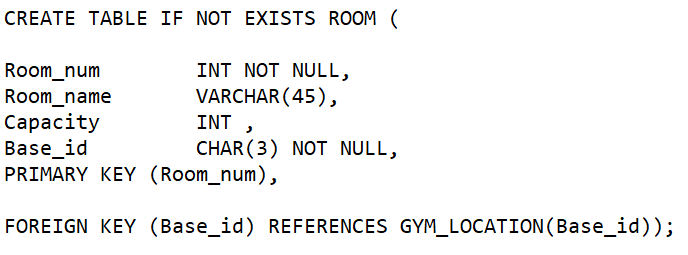
Due to the Corona Crisis, all the gyms are currently closed. That’s the reason why the gym is offering compensations to the members. Therefore, the compensation (e.g. a protein shake flatrate or a guest card) are stored in the compensation-table. For those members who have chosen a compensation, the Member\_id is linked to the compensation. Moreover, the table contains the period of time for which the compensation is chosen.

**ER- Diagram**



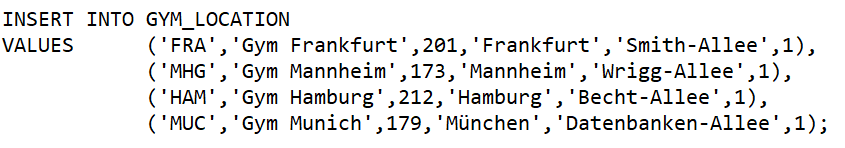
**Create Tables (Examples) – All commands are attached to this documentation.**

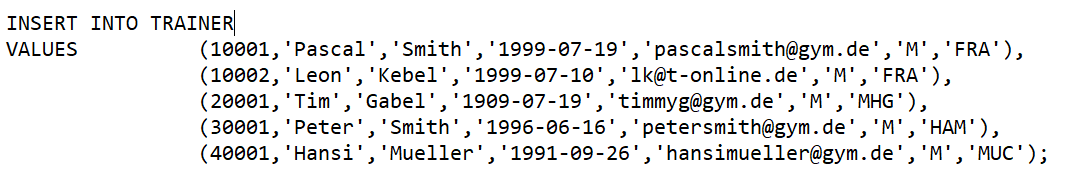


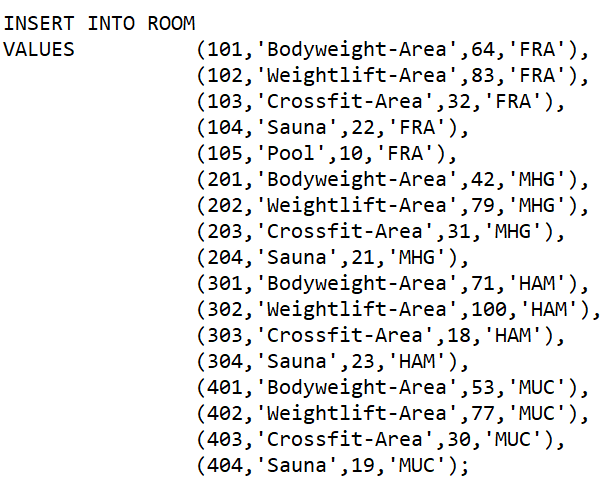


**Insert values (Examples) – All commands are attached to this documentation.**

As we have created the database only once, we only had to insert the values at one stage. This is the reason why we didn´t write “INSERT IF NOT EXISTS”.





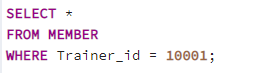


**SQL Queries**

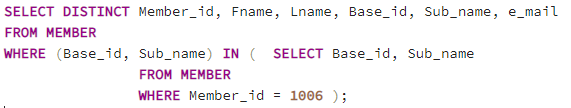
1. Retrieve the first and last name of those members, who have a trainer



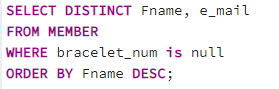
1. Retrieve all information of the members, who have the same trainer (identified by ID: 10001)



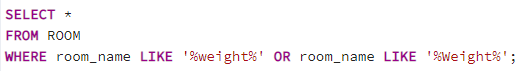
1. Retrieve the Member-ID, first & last name, base-ID, subscription type and e-mail address of those members, who work out at the same location and who have the same subscription as Member 1006



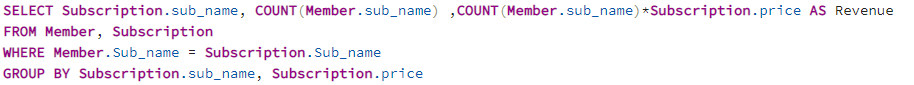
1. Retrieve the first name and the e-mail address of those customers who do not have a bracelet yet and order them descending.



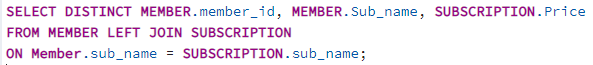
1. Retrieve the room number, the room name, the courses´ capacity and the base, in which the courses take place that contain the word “weight”.



1. Provide an overview of the gyms´ revenues for each subscription type.

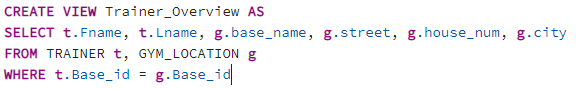


1. Retrieve the Member-ID, their subscription and the monthly fare of each customer.



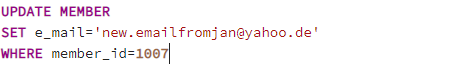
**View**

Create a view, to get an overview of all the trainers and their location. Provide some detailed information of their base.



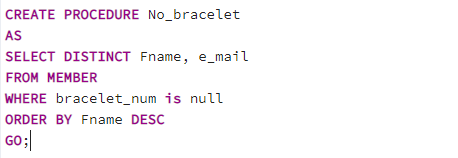
**Transaction**

Update the e-mail address of the member (with member\_id: 1007) to “new.emailfromjan@yahoo.de”.



**Stored Procedure**

Procedure, that extracts all Members, who don´t have a bracelet.





**Database Normalization**

The database relations should satisfy at least 3NF.

Our database meets the requirements of 1NF, as all attributes are atomic (e.g. The addresses of the gym locations are subdivided in city, street, housenumber). What is more, there are no relations within other relations in our database.

Moreover, our relation schema is in the second normal form, due to the fact that it is in 1NF and every non-key attribute is full functionally dependent on any candidate key of our relation. Therefore, we created separate tables (e.g. for Trainer and Course).

Lastly, our database is also in 3NF because it is in 2NF and furthermore there is no non-key attribute of our relation schema which is transitively dependent on a candidate key. (e.g. We created a Bracelet relation, so that the bracelet\_num (which is a key attribute) and the bracelet colour (which is a non-key attribute) are not part of the Member relation).