Linnéuniversitetet
Kalmar Växjö

Report

Final Project - Assigment3

2DV513

Author: Johan Eriksson
Semester: Autumn 2021
Email: Je222zx@student.lnu.se

Table of Contents

Introduction

The following report describes the features and results of the final database project in the course 2dv513. For the assignment any programming language could be used and also any type of relational database management. The choice used for programming language is Java and the relational database choice is MySQL.

Project Idea

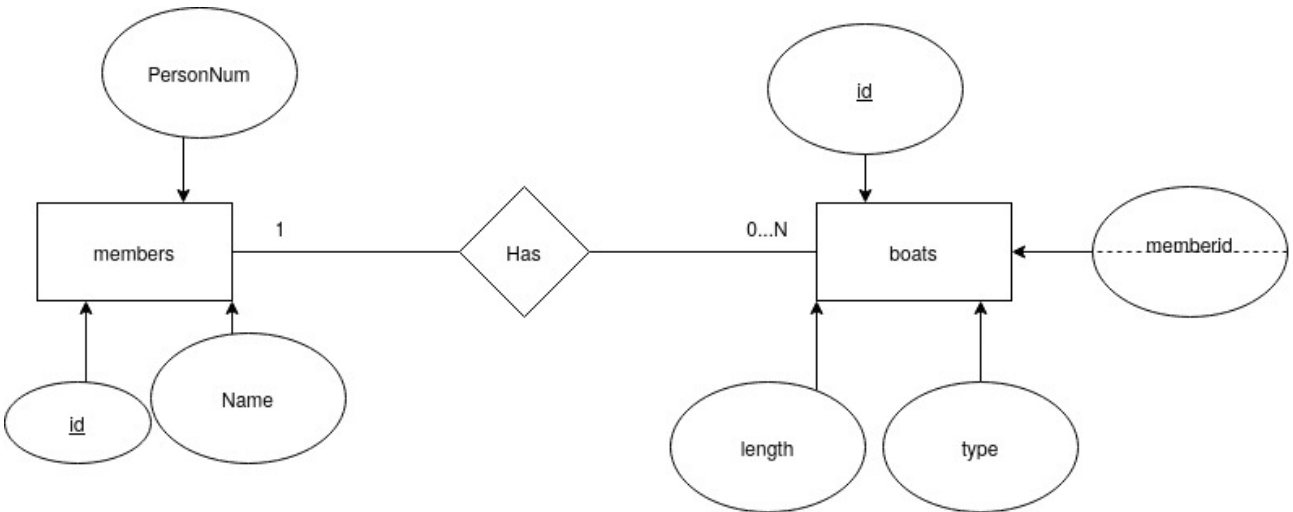
Living next to the see there is a boat club. The boat club need a way of keeping track of their members and which boats each member own. The boat club already has a local desktop computer in the reception that they should be able to use for the purpose. To solve the issue a system for management the members and their boats was created where the administrator of the boat club can add members and then add boats to the members. To make sure that is only the administrator (authorised user) that can make changes to the system a prompt for log-in is created. The main user will be the boat club and then specifically the administrative staff.

If a registered member or boat has wrong information the information about the member or boat needs to be changed it will be possible to do so. The information about the boats will be about the boat model and the length and the boat will be connected to the boat owner. The information about the owner will contain personal number, a member id and a name. The boats should also have a id, so that the system in the future could be developed so that the boat easily can switch owner however this will not be a feature of the first version of the system.

Feature of the system:

- The administrator should be able to add members
- The administrator should be able to edit members information
- The administrator should be able to remove members
- The administrator should be able to add boats to the members
- The administrator should be able to edit the boat information
- The administrator should be able to remove boats from members

Schema Design



One note to be made is that the tables are missing the ”s” when the system was build so members is “member” and “boats” is boat, however since each table can have more then one member or boat the “s” is still in the E-R Model. These typos was seen when the system was almost finished so therefore it was not being changed.

Design in SQL

The connection to the database was created by providing a user, password and which database schema it should connect to and the database URL. The unicode part of the URL for the time was needed for some reason and with out that a connection was not established.

```
public class SqlOperator {
    private static Connection connection;
    private static final String user = "root";
    private static final String password = "R1!-You_logged_in";
    private static final String databseSchema = "BoatClub";
    private static final String databaseURL = "jdbc:mysql://localhost:3306/" + databseSchema
            + "?useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC";

    public static Connection getConnection() throws SQLException {
        if (connection == null) {
            connection = DriverManager.getConnection(databaseURL, user, password);
        }
        return connection;
    }
```

For the design of the project three tables are needed. The first one to make a log-in for the administrator, to make more difficult for unauthorised persons to make the changes. The two other ones for handling the members and the boats. The members table will store information about the member such as personal number, member id, name. The boats will store the boat model, length and boat id.

The ids, both member id and boat id, uses auto increment to make the system handle that part so that the administrator does not need to came up with a random number. The ids will also therefore be set to primary keys.

To avoid non real values all of the columns has the not null constrain.

The Tables created was used by using the following schema:

CREATE DATABASE BoatClub;

USE BoatClub;

CREATE TABLE member(

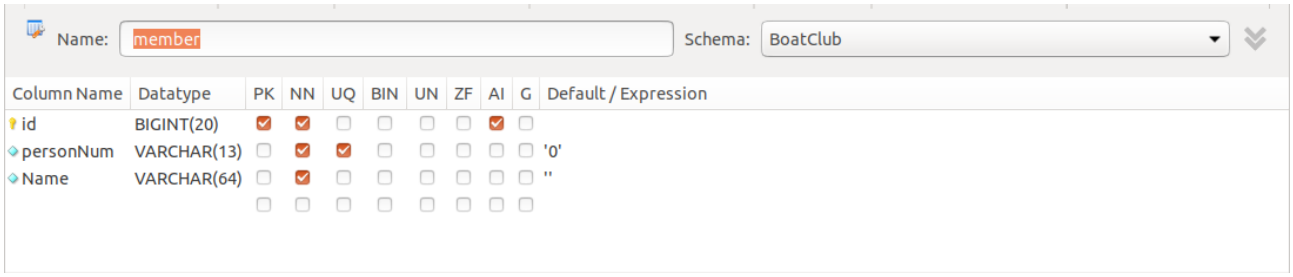
`id` bigInt NOT NULL AUTO_INCREMENT,

`personNum` int NOT NULL DEFAULT 0,

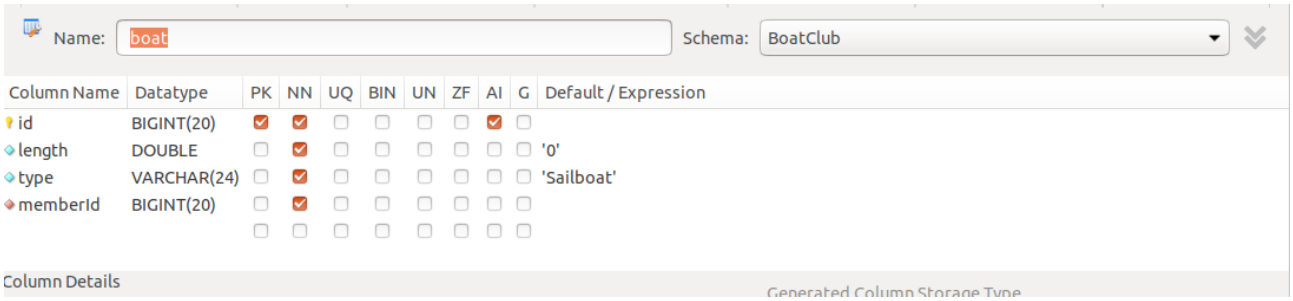
`Name` varchar(64) NOT NULL DEFAULT "",

PRIMARY KEY(id)

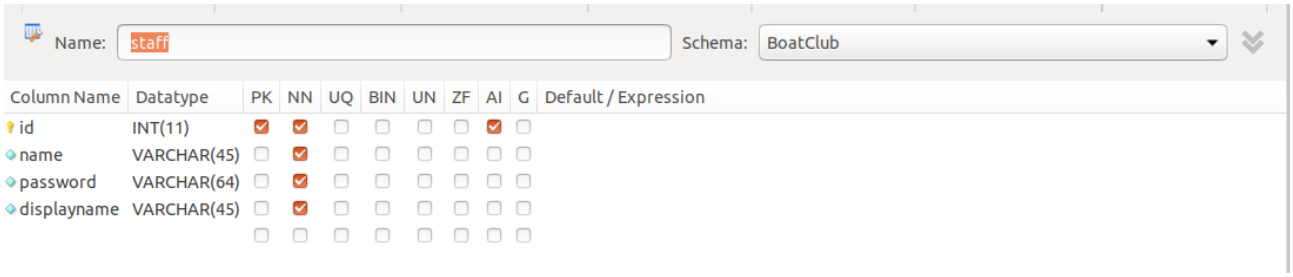
```
);
```




```
CREATE TABLE boat(
        `id` bigInt NOT NULL AUTO_INCREMENT,
        `length` double NOT NULL DEFAULT 0,
        `type` varchar(24) NOT NULL DEFAULT "Sailboat",
  `memberId` bigint NOT NULL,
  PRIMARY KEY(id),
        FOREIGN KEY(memberId) REFERENCES member(id)
);
```




```
CREATE TABLE `staff` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `password` varchar(64) NOT NULL,
  `displayname` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
);
```



SQL queries

The desktop application uses the following queries:

1. **SELECT * FROM memberBoats**
    a) Query for getting members and their boats. This uses a view that was created:
        i. **CREATE OR REPLACE VIEW memberBoats AS SELECT member.id, member.name, member.personNum, boat.id AS 'boatid', boat.type, boat.length, boat.memberId FROM member LEFT JOIN boat ON member.id=boat.memberid**
2. **INSERT INTO member(`personNum`,`Name`) VALUES(?,?)**
    a) adding a new member and the arguments passed is peronNum and Name and this is marked by the question marks.
3. **UPDATE member SET `personNum`= ?, `Name` = ? WHERE id=**
    a) This query will update the member info about name and personal number for an existing member. The id that the query should match is passed in as a parameter to the method in Java (String.valueOf(member.getId())) that is used by the query
4. **DELETE FROM member WHERE id = ?**
    a) This Query deletes the selected member
5. **INSERT INTO boat(`length`,`type`,`memberId`) VALUES(?,?,?)**
    a) This query creates a new boat and uses the foreign key to connect it to the member
6. **UPDATE boat SET `length`= ?, `type` = ?, `memberId` = ? WHERE id=**
    a) This query will update the boat info about length and typy for an existing member. The id that the query should match is passed in as a parameter to the method in Java (String.valueOf(boat.getId())) that is used by the query
7. **DELETE FROM boat WHERE id = ?**
    a) This query deletes the boat from the member by using the id for the boat

8. **SELECT displayname FROM staff WHERE name = ? and password=?**
    a) This query is used for the administrator to log-in. A administrator was created with following properties:
        i. **INSERT INTO `BoatClub`.`staff`**

        (`name`,

        `password`,

        `displayname`)

        VALUES

        ('Erik',

        sha1('animal_23_Colors'),

        'Erik');
9. **SELECT SUM(members)/SUM(boats) AS ratio FROM ((SELECT 1 AS 'members', 0 AS 'boats' FROM member) UNION ALL (SELECT 0 AS 'members', 1 AS 'boats' FROM boat)) ratio**
    a) This query full-fill the criteria of using aggregation since SUM is counted as aggregation and it gives the ratio of boats to members
10. **SELECT id, SUM(COUNT) AS 'COUNT' FROM ((SELECT member.*, 1 AS 'COUNT' FROM member INNER JOIN boat ON member.id = boat.memberid) UNION ALL (SELECT member.*, 0 AS 'COUNT' FROM member LEFT JOIN boat ON member.id = boat.memberid)) A GROUP BY id;**
    a) This query full-fill the criteria by using join. The joins used is both INNER JOIN and LEFT JOIN. The information about how many boats the member with most boats have and least boats have is given.
11. SELECT * FROM member WHERE personalNum = ?
    a) Used to check so that a new created member has a unique personal number

Discussion

How the queries is used in the methods could be update due to bugs found in last minute – however, there are work around for the bugs found and that is to restart the program after adding a new member or a boat.

Link to YouTube presentation: https://youtu.be/SgfQlvuVNNc

Link to GitHub: https://github.com/je222zx/2dv513-Assigment3 (Also includes how to use the progam in the read me file.)

Appendix 1

Changelog

Student:	Task	Date
Johan	Adding the functions for Members	2022-03-25
Johan	Testing the functions for Members	2022-03-25
Johan	Adding functions for Boats	2022-03-26
Johan	Testing the functions for Boats	2022-03-26
Johan	Adding functions for statistics	2022-03-26
Johan	Testing the functions for statistics	2022-03-27
Johan	Setting up Git and upload the project	2022-03-27