Kennesaw State University

College of Science and Mathematics

Department of Computer Science

Database Systems 3310 Section 01

*Database Implementation and Revision Report*

Austin Hickman

[ahickma2@students.kennesaw.edu](mailto:ahickma2@students.kennesaw.edu)

6/20/13

* The Purpose

The purpose of this project was, given a set of tables, to identify both functional and multivalued dependencies and to suggest a new reconstruction of the tables in such a manner as to comply with BCNF and 4NF. Also, it was necessary to include a way to implement our new structure in an updatable database with a user friendly GUI interface to allow a user to modify and add to the existing database that was implemented (updatable).

* Part I: Identifying Dependencies and Reconfiguring the Tables

PT\_SESSION (Trainer, Phone, Email, Fee, ClientLastName, ClientFirstName, ClientPhone, ClientEmail, Date, Time)

CLUB\_MEMBERSHIP (ClientNumber, ClientLastName, ClientFirstName, ClientPhone, ClientEmail, MembershipType, EndingDate, Street, City, State, Zip)

CLASS (ClassName, Trainer, StartDate, EndDate, Cost)

Given the 3 tables above it’s easy to see that there needs to be some degree of separation among them. Here is my interpretation of the tables in the order they appear:

1. **Possible Multivalued Dependencies:**

PT\_SESSION

* Trainer ->-> Phone
* ClientName ->-> ClientPhone

CLUB\_MEMBERSHIP

* ClientNumber ->-> ClientPhone
* ClientName ->-> ClientPhone

CLASS

* I could not find any multivalued dependencies here

1. **Possible Functional Dependencies:**

PT\_SESSION

* Trainer -> Email
* Trainer -> Fee
* ClientLastName -> ClientEmail
* ClientFirstName -> ClientEmail

CLUB\_MEMBERSHIP

* ClientNumber -> ClientLastName
* ClientNumber -> ClientFirstName
* ClientNumber -> ClientPhone [This one is very arguable, but I included it for completeness]
* ClientNumber -> ClientEmail
* ClientNumber -> MembershipType
* ClientNumber -> EndingDate
* ClientNumber -> Street
* ClientNumber -> City
* ClientNumber -> State
* ClientNumber -> Zip

CLASS

Since CLASS doesn’t have any clear and distinct separate relations…I just combined some to make a few possible choices…

* (ClassName,Trainer,StartDate) -> Cost
* (ClassName, Trainer, StartDate) -> EndDate
* ClassName -> Trainer

1. **BCNF and 4NF**

PT\_SESSION

* I would say it is neither BCNF or 4NF because the table has 3 different subjects…the personal trainer, the trainer’s client, and the actual personal training session and its associated cost. It is not in 4NF because based on what I stated above it has two multivalued dependencies. Additionally, it would be hard to find a good primary key out of the columns provided in the table. ClientEmail (arguably) could be good enough to use, but the real problem lies with the ClientName which is not exclusive enough to base the table on. It should have a dedicated SessionID.

CLUB\_MEMBERSHIP

* I would say this table is in BCNF (roughly). By definition, a table is in BCNF if every determinant is a candidate key. The ClientNumber cloumn is the determinant for the rest of the table. Likewise, the tuples you can make out of (ClientLastName, ClientFirstName) determines the ClientNumber. I do not, howeve, think it is in 4NF because it has 2 multivalued dependencies I identified which breaks the definition of 4NF.

CLASS

* I would say CLASS is in both BCNF and in 4NF. There are no multivalued dependancies.

1. I’m going to break this section into two parts. The **first** part will show what I did. The **second** part will show what I could have possibly done but did not decide to do initially when I did this project, for better or for worse.

Part I: Table Changes I Made

* I split the 3 tables provided into 6 different tables:
  + TRAINER
    - TrainerID (Primary key)
    - TrainerLastName
    - TrainerFirstName
    - TrainerEmail
    - TrainerPhone
  + CLIENT
    - ClientNumber (Primary key)
    - ClientLastName
    - ClientfirstName
    - ClientPhone
    - ClientEmail
    - ClientState
    - ClientCity
    - ClientAddress
    - ClientZipCode
  + CLIENT\_MEMBERSHIP
    - MembershipID (Primary key)
    - ClientNumber (Foreign key to CLIENT.ClientNumber)
    - MembershipType
    - MembershipStartDate
    - MembershipendDate
  + TRAINING\_SESSION
    - SessionID (Primary key)
    - ClientNumber (Foreign key to Client.ClientNumber)
    - TrainerID
    - TrainingStartDate
    - TrainingEndDate
    - TrainingFee

**\*I probably should have used TrainerID as a primary key here and as a foreign key to TRAINER.TrainerID, but I did not when I implemented the project.**

* + CLUB\_CLASSES
    - ClassID (Primary key)
    - TrainerID (Foreign key to TRAINER.TrainerID)
    - ClassName
    - ClassStartDate
    - ClassEndDate
    - ClassCost
  + CLIENT\_ENROLLMENT
    - ClientNumber (Primary ; Foreign key to CLIENT.ClientNumber)
    - TrainerID (foreign key to TRAINER.TrainerID)
    - ClassesSubscribed
    - AmountPaid

Part II: Another Possible Way (maybe) to Do This Project upon Further Reflection

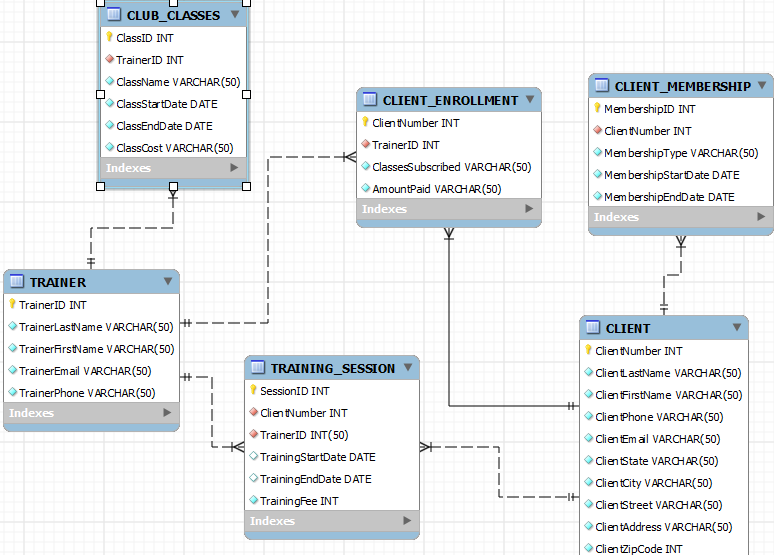
* I split the 3 tables into 7 this time
  + TRAINER
    - TrainerID (Primary key)
    - TrainerLastName
    - TrainerFirstName
    - TrainerEmail
    - TrainerPhone
  + CLIENT
    - ClientNumber (Primary key)
    - ClientLastName
    - ClientfirstName
    - ClientPhone
    - ClientEmail
  + CLIENT\_ADDRESS
    - ClientNumber (Foreign key to CLIENT.ClientNumber)
    - ClientState
    - ClientCity
    - ClientAddress
    - ClientZipcode

\***Here the reason I broke the table into two is because to truly be in BCNF it needed to be broken into the 2 tables CLIENT\_ADDRESS and CLIENT.**

* + CLIENT\_MEMBERSHIP
    - MembershipID (Primary key)
    - ClientNumber (Foreign key to CLIENT.ClientNumber)
    - MembershipType
    - MembershipStartDate
    - MembershipendDate
  + TRAINING\_SESSION
    - TrainerID (Primary; Foreign key to TRAINER.TrainerID)
    - ClientNumber (Foreign key to Client.ClientNumber)
    - TrainingStartDate
    - TrainingEndDate
    - TrainingFee
  + CLUB\_CLASSES
    - ClassID (Primary key)
    - TrainerID (Foreign key to TRAINER.TrainerID)
    - ClassName
    - ClassStartDate
    - ClassEndDate
    - ClassCost
  + CLIENT\_ENROLLMENT
    - ClientNumber (Primary ; Foreign key to CLIENT.ClientNumber)
    - TrainerID (foreign key to TRAINER.TrainerID)
    - ClassesSubscribed
    - AmountPaid

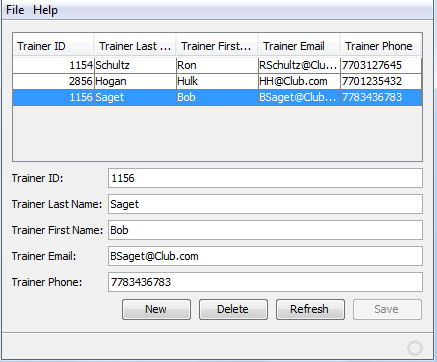
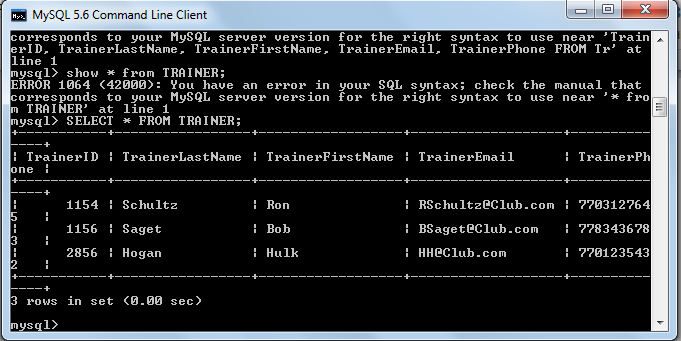
For either design the referential integrity is enforced by the system of foreign keys within them. For example, you could not delete CLIENT\_ENROLLMENT if there is information in the column ClientNumber of table CLIENT. Likewise, if you try to delete a certain row, say TrainerID row in CLUB\_CLASS the link between CLUB\_CLASS’s TrainerID and TRAINER’s TrainerID will both need to be updated to make the database consistent. Otherwise, you would have many inconsistent ID references if someone was deleted. It is generally a bad idea to have a description or reference to something that no longer exists or applies within your database.

* Part II: Database Design

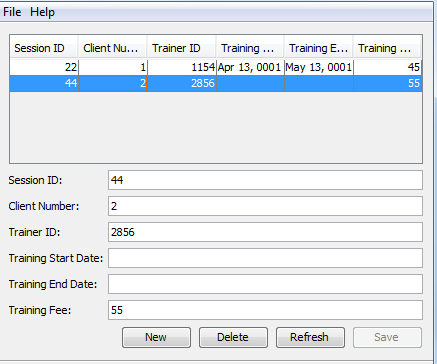
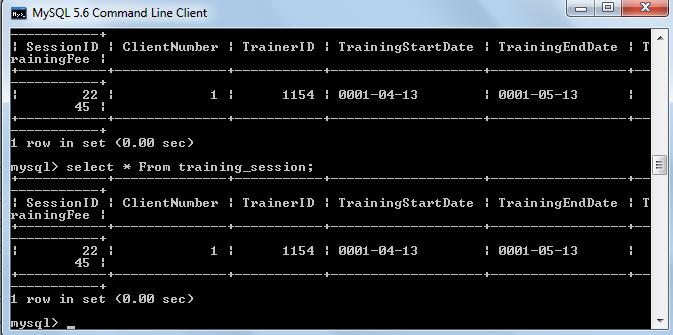


* Database Implementation
  + Requirements to reconstruct the project:
* MySQL and its command line client
* NetBeans Version 7.0.1 or Earlier (Reason why is discussed later)
* The MYSQL driver connector
* Used with Windows 7 x64

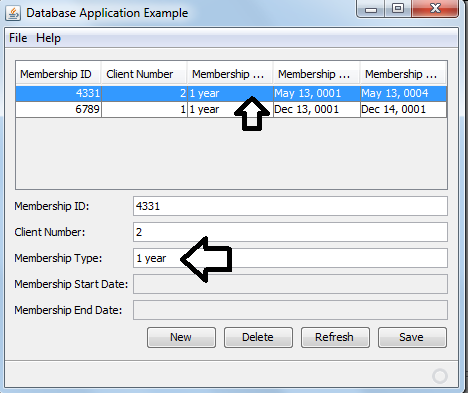
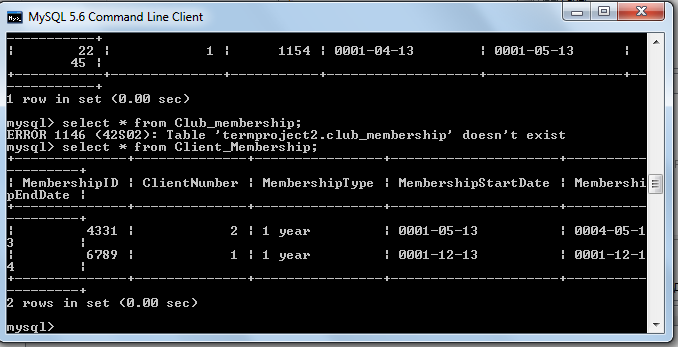
1. Insertion

1. Deletion

1. Update

* Part III Table Code

CREATE TABLE TRAINER(TrainerID INT NOT NULL, TrainerLastName VARCHAR(50) NOT NULL, TrainerFirstName VARCHAR(50) NOT NULL, TrainerEmail VARCHAR(50) NOT NULL, TrainerPhone VARCHAR(50) NOT NULL, PRIMARY KEY(TrainerID));

INSERT INTO TRAINER VALUES('1154','Schultz','Ron','RSchultz@Club.com','7703127645');

INSERT INTO TRAINER VALUES('2856','Hogan','Hulk','HH@Club.com','7701235432');

CREATE TABLE CLIENT(ClientNumber INT NOT NULL, ClientLastName VARCHAR(50) NOT NULL, ClientFirstName VARCHAR(50) NOT NULL, ClientPhone VARCHAR(50) NOT NULL, ClientEmail VARCHAR(50) NOT NULL, ClientState VARCHAR(50) NOT NULL, ClientCity VARCHAR(50) NOT NULL, ClientStreet VARCHAR(50) NOT NULL, ClientAddress VARCHAR(50) NOT NULL, ClientZipCode INT NOT NULL, PRIMARY KEY (ClientNumber));

INSERT INTO CLIENT VALUES ('1','Marget','Michael','7703399207','MM@gmail.com','Georgia','Kennesaw','Wending Way','312 Wending Way','30144');

INSERT INTO CLIENT VALUES ('2','Squarepants','Spongebob','7701274532','SS@gmail.com','Georgia','Kennesaw','Pineapple Under the Sea Way', '856 Pineapple Under the Seas Way','30122');

CREATE TABLE TRAINING\_SESSION(SessionID INT NOT NULL, ClientNumber INT NOT NULL, TrainerID INT(50) NOT NULL, TrainingStartDate date, TrainingEndDate date, TrainingFee INT NOT NULL, PRIMARY KEY(SessionID), FOREIGN KEY(TrainerID) REFERENCES TRAINER(TrainerID), FOREIGN KEY(ClientNumber) REFERENCES CLIENT(ClientNumber));

INSERT INTO TRAINING\_SESSION VALUES ('22','1','1154','1/4/13','1/5/13','45');

INSERT INTO TRAINING\_SESSION VALUES ('44','2','2856','1/8/13','1/9/13','55');

CREATE TABLE CLIENT\_MEMBERSHIP(MembershipID INT NOT NULL, ClientNumber INT NOT NULL, MembershipType VARCHAR(50) NOT NULL, MembershipStartDate DATE NOT NULL, MembershipEndDate DATE NOT NULL, PRIMARY KEY(MembershipID), FOREIGN KEY(ClientNumber) REFERENCES CLIENT(ClientNumber));

INSERT INTO CLIENT\_MEMBERSHIP VALUES ('6789','1','1 year','1/12/13','1/12/14');

INSERT INTO CLIENT\_MEMBERSHIP VALUES ('4331','2','3 months','1/05/13','4/05/13');

CREATE TABLE CLUB\_CLASSES(ClassID INT NOT NULL, TrainerID INT NOT NULL, ClassName VARCHAR(50) NOT NULL, ClassStartDate date NOT NULL, ClassEndDate date NOT NULL, ClassCost VARCHAR(50) NOT NULL, PRIMARY KEY(ClassID), FOREIGN KEY (TrainerID) REFERENCES TRAINER(TrainerID));

INSERT INTO CLUB\_CLASSES VALUES ('3501','1154','Yoga','1/3/13','1/5/13','100');

INSERT INTO CLUB\_CLASSES VALUES ('3502','2856','Pillate','1/3/13','2/5/13','50');

CREATE TABLE CLIENT\_ENROLLMENT(ClientNumber INT NOT NULL, TrainerID INT NOT NULL, ClassesSubscribed VARCHAR(50) NOT NULL, AmountPaid VARCHAR(50) NOT NULL, PRIMARY KEY(ClientNumber), FOREIGN KEY(ClientNumber) REFERENCES CLIENT(ClientNumber), FOREIGN KEY(TrainerID) REFERENCES TRAINER(TrainerID));

INSERT INTO CLIENT\_ENROLLMENT('1','1154','2', '150');

INSERT INTO CLIENT\_ENROLLMENT('2','2856','4', '500');