NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

LABORATORY MANUAL

CZ2007: Introduction to Databases

Software Lab 2 (Location: N4-01c-06)

or

Software Lab 3 (Location: N4-B1c-14)

Implementation of a Database Application

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

1. OBJECTIVES

Upon completion of the assignment, the student should be able to:

a. Construct an entity-relationship model at a conceptual level
b. Map the model into a schema of a relational DBMS
c. Implement the given schema on a relational DBMS
d. Use a database language (SQL) for manipulating and updating data

2. LABORATORY

This is a group assignment. Each group consists of **four to five** members from your laboratory group. You have the choice of selecting your group members. However, all the names within your group must be given to the laboratory technician(s) during your first laboratory session. Name lists with respect to each laboratory group are available in the "Public Folders" towards the beginning of the semester.

Note that the laboratory will start from the **third week** of the semester onwards and that you might need more than the mentioned five sessions for the actual implementation. You are also encouraged to **start early** with your assignment (as soon as the topic is covered in the lectures).

Attendance is taken for all supervised laboratory sessions. It is the responsibility of each student to sign-in at the beginning of each session. **Failing to sign-in for the first, third, or fifth lab session may result in a F grade for the respective assessment.**

3. INTRODUCTION

The assignment covers the portion of the course concerning data modelling, database design and implementation from the user's viewpoint. Thus, the assignment involves modelling as well as implementation aspects of the database course.

The overall aim of the laboratory is to develop an application based on a given data model using a given database management system. This exercise will bring you through a crucial first part of the life cycle of a database application. It is assumed that the data analysis has been performed. Note that this manual provides you with more information than is required for the first laboratory session; e.g., not all constraints can be modelled in the beginning but are included at a later implementation stage. In contrast you might require additional information for the understanding of the application. Proceed by stating your assumptions in written form and / or ask your laboratory supervisor.

4. DESCRIPTION OF THE ASSIGNMENT

The description of the application is given in the appendices. This includes the back-ground and general requirements of the application, conceptual information about the system and its users as well as a list of queries that must be fulfilled as a minimum.

Note that teamwork is required. Every team has to submit one solution. **No individual submission will be accepted.**

4.1 First Laboratory Session: Creating an ER Diagram

Appendix A gives conceptual information about the project obtained after a partial sys-tem analysis was performed. Based on the appendices, construct a suitable ER diagram. Analyse the cardinality of relationships, the usage of weak entity sets, choice of entity sets etc. and compare them with alternative solutions. The laboratory technicians will provide the necessary information at the beginning of the lab session.

You need to submit the followings at latest **three working days** after the first laboratory session.

- A hard copy of your ER diagram and written discussion of your solution (maximum one page), which highlights the reasons for the chosen design.

4.2 Second Laboratory Session: Finalization of the ER Diagram

There is no submission for the second laboratory session. In this lab, each group should finalize their database design based on the feedback received from their lab supervisor. Please note that the second laboratory session is a free-access session; i.e., attendance is not compulsory (but recommended in case the group has questions).

4.3 Third Laboratory Session: Generation of Normalized Database Schema

In this lab you must ensure that the database is at least in 3NF. Follow the general guide-lines covered during the lectures and tutorials to produce suitable normalized relations. For each relation, the key(s), primary key, and functional dependencies must be specified. If a relation is generated due to normalization of an original relation, then the normalization steps must be presented.

You need to submit the followings at latest **three working days** after the third laboratory session.

- A hardcopy of the normalized database schema and FDs associated with each relation. If a relation created from the ER diagram violates the 3NF form then this should be highlighted along with the decomposed normalized relations. Note that for this lab no SQL code should be submitted. Hence, the structure of your solution shall be similar to the following example:

  R1(A, B, C, D)

  Keys: AB, AD
  Primary Key: AB
  FDs: AB → CD, A → D
  The relation is in 3NF.

4.4 Fourth Laboratory Session: Implementation of the database schema

There is no submission for the third laboratory session. In this lab, the finalized database schema must be implemented using SQL DDL commands. Your implementation should clearly incorporate the primary and foreign keys, data types, integrity constraints, value-based and tuple-based constraints. Solve the implementation by using the MS SQL Server software.

Please note that the fourth laboratory session is a free-access session, i.e., attendance is not compulsory (but recommended in case the group has questions).

4.5 Fifth Laboratory Session: Final demonstration

The fifth session is the final assessment of your implementation. The implementation obtained from the previous laboratory session has to be extended by incorporating necessary triggers and additional constraints. In addition, you have to formulate the SQL statements for the sample queries in Appendix B and triggers in Appendix C.

This session has two components. First, **at the beginning of the lab** a hardcopy of the schema implemented using the SQL DDL commands together with constraints and sample queries need to be submitted. Auto-generated relations are not permitted. Hence, the structure of your solution for the database schema definition shall be similar to the following example and written by yourselves:

```
CREATE TABLE name (
        attr1 datatype NOT NULL,
        attr2 datatype,
        …
        PRIMARY KEY (attr1),
        FOREIGN KEY (attr3) REFERENCES name(attr1)
                ON DELETE … ON UPDATE …,
);
```

The second component of this session involves **demonstration** of your system. All team members are required to contribute actively during the demonstration session. Additionally, the laboratory supervisor will ask individual questions. During the demo session, the evaluation shall be based on the following points:

- Implementation and execution of additional queries on the spot
- Answers on and understanding of the design and related issues
- Demonstration of the proper working of your implementation
- Additional effort in terms of implementation etc.
- Presentation quality

Note that your group might be required to begin the presentation at any time during the fifth laboratory session; i.e., one team will be asked to present at the beginning of the session. All applications should run on the provided hardware and software components of the Software Laboratory 2 using MS SQL Server.

APPENDIX A

User requirements about the DBMS

You are to design a database system for a Location Based Social Network website (You are welcome to name your website :-). The basic features of the website are as follows:

- Each user of the site is identified by an id and has a unique email address. Basic info of the user, including name, sex, birthday (year, month and date) and hometown are stored.
- Education and work experiences of the user can be included. For each education experience, the school name, class year, school type (graduate school, college, high school, and elementary school) are maintained; for each work experience, employer, position, city/town and time period (including a starting and ending time, in year and month) are maintained.
- User can write a message (a.k.a. status update), or send a message to another user (a.k.a. write on another user's wall). Any user can add comments to such a message. The text of message/comment and the timestamp are kept.
- A user can have friends who are also users of the site. Users can create named list to organize friends. There are two default lists "Friends" (which contains all friends), "News Feed" (which contains friends that the user will receive updates from).
- Each point of interest (e.g., a hawker center, https://foursquare.com/v/maxwell-road-hawker-centre/4b1f21a6f964a520432424e3) is identified by an id, and has unique coordinates. Basic info of a point of interest includes name, address, contact information, description, etc. Each point of interest may also be associated with several category labels (e.g., coffee shop)
- Suppose there is a collection of photos, each associated with text caption. A user can associate a photo from the collection to a point of interest. Suppose that a photo can be associated only once by a user to a point of interest.
- A user can check-in a point of interest, and the user can leave a shout (a short text) at check-in.
- A user can also leave a tip at a point of interest.

Note that Appendix A only gives the scope of the project, and understanding and defining user requirement is a necessary part of database design in practice. The provided information is not complete. Many aspects of the system's functions and details have been omitted. It is expected that the teams come up with their own solutions in case of inconsistencies or missing information. However, you have to keep track of these aspects and explain your assumptions if asked for the reasons. You can consider Facebook and FourSquare for references. Extensions to the implementation of the basis system are encouraged.

APPENDIX B

Queries

- Show news feed for a user: List the latest 5 messages, each of which is either a status update from this user, a wall message sent from this user to a friend in the "News Feed" list, or a wall message sent from a friend in the "News Feed" list to this user.

- Suggest friends to a user: select top 5 users of the site who has the largest number of mutual friends with this user and yet to be a friend.

- Show most connected users: list top 5 users of the site with the largest number of friends.

- Suggest points of interest to a user: select top 5 points of interest that are mostly visited by the user's friends, and have not yet been visited by the user.

- Find nearby Points of Interest of a given point of interest: Given a point of interest, find those points of interest such that they are within 10km of the given point of interest, their descriptions contain word "ipad", and their tips contain word "excellent".

- Show most active users: list top 5 users of the site who is most active (the degree of activity for a user is measured by the number of status updates from this user, plus the number of messages sent from this user to another user, plus the number of comments from this user, plus the number of check-in from this user, plus the number of tips from this user)

Design two queries that are not in the above list. They are evaluated based on the usefulness, complexity, and the interestingness.

Appendix C
Constraints & Triggers

- The two default friend lists cannot be deleted.

- The friend relationship has to be mutual, meaning, if user A is a friend of user B, B has to be a friend of A.

- When a new friend is added, it is placed automatically in the "Friends" and "News Feed" lists; when it is removed, it is deleted automatically from these lists.

- The database should automatically keep track of the top 10 most active users: (the degree of activity for a user is measured by the number of status updates from this user, plus the number of messages sent from this user to another user, plus the number of comments from this user)

- The maximum number of check-in of a user make at a point of interest per day is 10

Design two triggers that are not in the above list. They are evaluated based on the usefulness, complexity, and the interestingness.