Project: Database System and Applications

Choose an exercise from the two exercises below, follow the instructions to complete the project.

# Letting Agency

A database needs to be developed for a letting agency in a big city. The agency rents out properties to tenants on behalf of the landlords.

1. The company employs staff. Every member of staff has a unique staff ID. In addition they have a name, a function (like secretary, accountant, etc.), and a monthly salary.
2. In order to locate properties, the agency has split the city into different areas. Each area has a name, by which it can be uniquely identified. For each area, there is a member of staff who is responsible for the properties in the area.
3. For each property that the agency manages there is a property ID by which it can be identified. Moreover, the agency wants to record the address, the type of the property (like ﬂat, house, garage, etc.), the monthly rate, and the deposit that has to be paid by a tenant. Each property is located in some area.
4. Each property is of a specific type, for instance, a ﬂat, a house, or a garage. The possible types are specified by the company. Each possible type needs to be recorded, also if there is no property of that type in the database.
5. Each property is owned by a single landlord. A landlord, however, may own more than one property. A landlord is registered with name, address, and phone number. Each landlord can be uniquely identified by their name and address.
6. A tenant is registered with a tenant ID, a current address, and a phone number.
7. A tenant rents a property by signing a lease. The lease specifies the start date, i.e., the time when the contract begins, and the duration of the lease. A lease is always related to a specific property. For a given property, there cannot be two leases with the same start date.

# Movie Database

Suppose you are going to develop a movie website like IMDB, and now you try to build a movie database. The database will contain information about movies, actors, directors, and comments.

1. Each movie has an unique ID, a title, a publishing year, a director, a description, several writers and several producers. A director can direct many movies, but a movie can have only one director.
2. There are different genres of movies, and a movie may belong to multiple genres. For example, Spiderman is an Action movie and also an Adventure movie.
3. For each movie, the website will also show the cast, i.e. the characters in the movies and the corresponding actors.
4. For each movie, the website will show the full crew. For example, the makeup artists, set designers, and so on.
5. A user should sign up to comment on the movies. When a user signs up, the name and email of the user are required, and a user id will be generated.
6. A movie may have multiple comments, and the website will show the a title, the author, posting time and text for each comment.
7. A user can rate a comment if he or she think the comment is helpful, and the rating is from 1 to 5, the higher the better.
8. Based on the rates of users, an average rate will be calculated for each movie.

Step 1: Data Requirements and Conceptual Design

The goal of the first part of your project is to

1. identify a collection of sample queries,
2. formulate data requirements for your application,
3. draw an ER-diagram that captures these requirements.

In the following, we will describe the single steps in more detail.

# Sample Queries

Specify **ten** example queries that you think are of interest in the domain of your application.

You need criteria to decide what to include into the data requirements and what not. A good starting point for developing a database is to identify typical queries that users will pose—either directly or via applications. These queries will be used to direct the formulation of data requirements and to check them for completeness. Write up the queries in plain English. Do not be concerned whether they can actually be expressed in a query language.

# Data Requirements and ER Diagram

In reality, the formulation of requirements and the design of the diagram go hand in hand. So, do both in parallel.

Your application should have a size that makes it nontrivial. As a guideline, aim at a diagram of **10 (entities + relationships)**. The minimum size should be **8 (entities + relationships)**.

Step 2: Constructing and Populating a Database

The goal of the second part of your project is to translate the ER-diagram of your project into a relational schema; implement the schema as a MySQL database; and fill your database with data.

# Translation into Relational Schema

Write up a relational schema that is equivalent to your ER diagram. Make sure that you translate not only entities but also relationships and that you also capture, whenever possible, the integrity constraints of your ER diagram.

Explain brieﬂy how the various entities and relations are expressed in the relational schema. Whenever there was a choice for the translation, explain the reasons for the choice you made.

During the translation you may find out that the relational schema resulting from the translation does not make a lot of sense. In this case, revise your ER diagram and the data requirements. It is imperative that all documents are consistent with each other.

# Implementation of the Schema in PostgreSQL

Write an SQL script with the commands to create the tables of your database. An SQL script is simply a file with SQL commands.

The CREATE TABLE statements must specify:

1. appropriate types for the attributes;
2. the primary key;
3. constraints such as NOT NULL and UNIQUE whenever appropriate;
4. default values where appropriate;
5. FOREIGN KEY constraints, together with the policy for reacting to changes (remember that the default is ON DELETE NO ACTION)

Write comments into the script that explain the rationale behind the definition of your constraints. Comments in SQL can be written in two ways:

# Loading Data

Load data into the database. Write a second script that contains INSERT statements for your relations. You may have to pay attention to insert records in such an order that the foreign key constraints are not violated.

Part 3: Querying a Database

The goal of the third part of your project is to query your database using the SQL query language

# Requirement Queries

As the first step in your requirement collection you have specified 10 queries from the domain of your application. Now, try to express them in SQL. It may be the case that some of your queries were too ambitious so that they cannot be captured by SQL. In that case explain why you think the query cannot be written as SQL query; identify other queries that can retrieve useful data for your original query and describe how that data has to be processed further to obtain answers for the original query.

# New Queries Over the Database

In addition to the queries stemming from the requirements, write new queries over your database schema that are interesting in your application domain. For each query, first write in English what it is supposed to do, and then give the SQL code.

Part 4: Database Access from a Programming Language

The goal of the fourth part of your project is to create an application program in Java or Python (or other language) by which a user can access the database.

There is no need for a graphical user interface or anything fancy. A simple command line interface will do. If you write Python program, PyMySQL is a good package to use.

**After you complete the project, submit you project report via email to liuxiping@jxufe.edu.cn. The format of the report can be found in next page.**

Database System and Applications

Project Report

Student ID: Name:

1. Introduction
2. Data Requirements and Conceptual Design
3. Database Construction and Population
4. Querying the Database
5. Database Access from Programs