Author: Yang Zheng

Now the project uses a JDBC-based data-access layer. There are a lot of duplicated codes in each dao class for database access. If any change is made to the database (such as the vendor of the database system, the IP address for connection and the username and password) in the future, each dao class will have to be modified. Moreover the java programmer for the project needs to be familiar with the details of the database systems and write a lot of SQL statements. While if we use hibernate to deal with the database access, we can manage the configuration for accessing database in one place and use the object-oriented way to interact with the database so that the programmer will write much less SQL statements and make changes and extensions more easily.

In order to transform the JDBC-based database access layer into the hibernate-based one, I need to first read the database design documents and understand the function of each entity and the relationship between them. Then set up the configuration and mapping for hibernate in the project, extract the duplicated JDBC-based data access code in each dao class and some domain classes, create a super class for the common operations in each dao classes change the JDBC-based database access operations to hibernate-based ones and modify the domain class to satisfy hibernate requirements.