Data Interface Design for Java-Based Mysql Database

Hongxia Liu

Department of Electronic Information Engineering, Handan Polytechnic College , Handan, 056001, China

Keywords: ORM component, middleware, SQL (Structured Query Language), encapsulation

Abstract: This essay designs and realizes a Java-based lightweight ORM component-Nemo: Nemo is one Java-based lightweight ORM component, a middleware which can realize the seamless connection between Java language and SQL (Structured Query Language) of data access. The function it realizes is to form an auto-mapping between object and relational database. The operation of class can automatically generate SQL statement and replace the process of traditional way of writing SQL statement, which will reduce programmers' coding workload to a great extent. Besides, the SQL statement and be encapsulated in the class and realize the reusage of SQL statement.

Introduction

ORM is short for Object/Relational Mapping. Currently, most enterprises adopt MVC mode as the main system architecture mode for application system design that is Model-View-Control. Early in 1998, Scott W Ambler has written an essay on the detailed design of ORM, where he put forward the Class Type Architecture idea and designed a persistence layer framework based on it. Later, according to this framework, Artem Rudoy developed an open-source project, then researches relevant to ORM became hotspot. Many manufacturers and open-source communities provide persistence realization frameworks, such as iBatis, Apache OJB, Hibernate, Cayenne, Toplink, jRelationalFramework, Jaxor. Apeche OJB, Hibernate and iBatis is widely used at present, with active development team and warm attention from all communities. Particularly, Hibernate is now almost the real standard of ORM framework. The publishing of Hibernate3 version realized the repair and improvement of bugs. However, all researches on above Object/Relational Mapping haven't been approved whether the mapping is complete. Due to the difference between relational mode and object mode, it is very difficult to completely realize Object/Relational Mapping.

Compared with foreign countries, the domestic ORM research is still on its first step. By research on three common persistence ways including embedded SQL, closed coupling and robust persistence technology, Yiqing Qin illustrated their characteristics and compared their advantages and disadvantages. He suggested developers should consider their own situation to decide whether to choose a simpler realization way and bear the risk of maintenance and expansion or choose a complex realization expansion way (persistence layer) which can be easily maintained and transplanted.

Demand Analysis on ORM Component (Nemo)

Function Demand of Nemo; Nowadays, many domestic programmers pay more attention on the persistence of program. Persistence means to synchronously save data in database or some storage devices, and the insert, delete, select and update of data in the database should be conducted through persistence layer. But ORM component connecting to persistence is heavyweight, and all the operation will affect the database, so it is not applicable to some smaller projects.

Performance Demand of Nemo; In terms of performance, users input in java program, so they should know about the class names, interface names, methods and parameters of the methods defined by the component, so that they can use the right way to input correct parameters. This component is only specific to ORACLE database, so users only need to lay-in the tables on a machine with ORACLE database, and no need to open ORACLE service when conducting detailed

operation.

Function Design of Nemo Component

General Design of Nemo: The design idea of Nemo is that users conduct relevant operations by interface and depart completely from detailed classes. What programmers need to know is the corresponding interface instead of detailed realizing procedures, which can eliminate much unnecessary trouble.

Detailed Design and Realization of Nemo Component: The core function of this component is to write SQL statement by Java, and auto-generate SQL statement for users. During the design, we realize these functions through the inheritance of interfaces layer by layer. Interfaces define methods, while methods will be realized in sub-classes. Methods to operate SQL statement includes Select, Insert, Delete and Update, also batch processing in PL/SQL, so we need to create corresponding statement objects. In the classes generating SQL, “from” is used in every operation, so it is defined in interface, and all class of statement only need to inherit this interface, which will simplify code to a great extent.

The connection of databases is realized by corresponding data package. These packages will inherit and invoke with each other, while classes of packages can also inherit and invoke. We can easily find out relation between packages and classes from the class diagram. As shown in Fig 1:

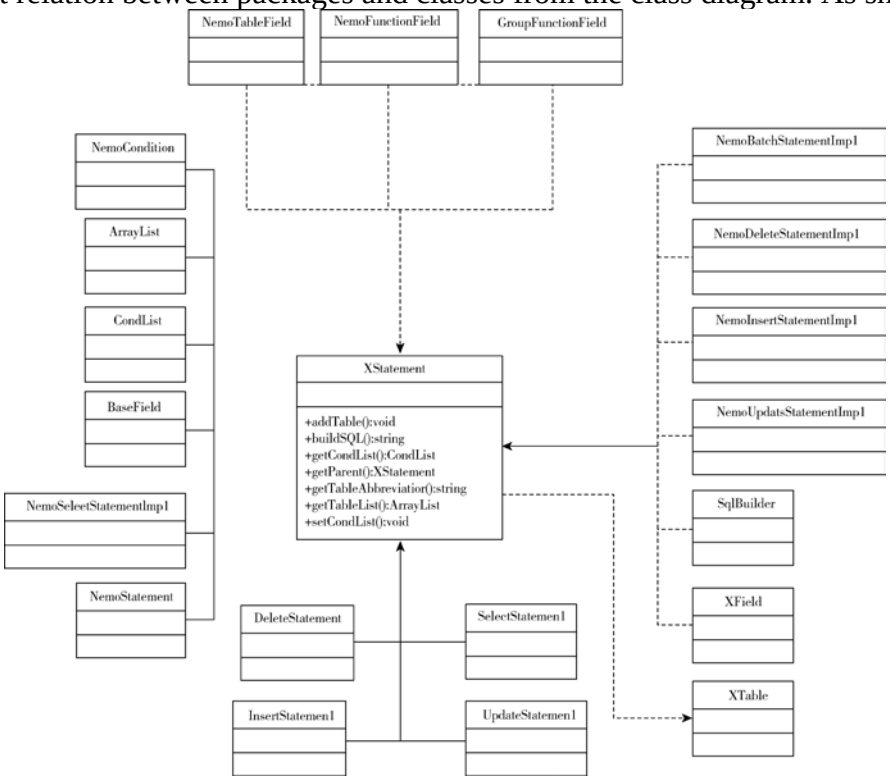


Fig 1. Nemo Class Diagram

Nemo Database Connection and Table Mapping

General Design: The design of component is mainly based on Oracle database, a very huge database. All Oracle users know that it is not easy to run Oracle on their machines. Another benefit of this design is that when developing software, if the needed tables are mapped to corresponding packages, users can finish the database operation without database service. It will increase the running speed and working efficiency without database installation in the machine, which will also save cost and optimize resource allocation. The process of creating cn.liaoningjusoft.nemo.bridge package and connecting it with database is also the process of realizing SQL statement by Java language. There is only one class NemoTablesGenerator.java underneath. Let us see the class diagram for NemoTablesGenerator.java in detail as shown in Fig 2.

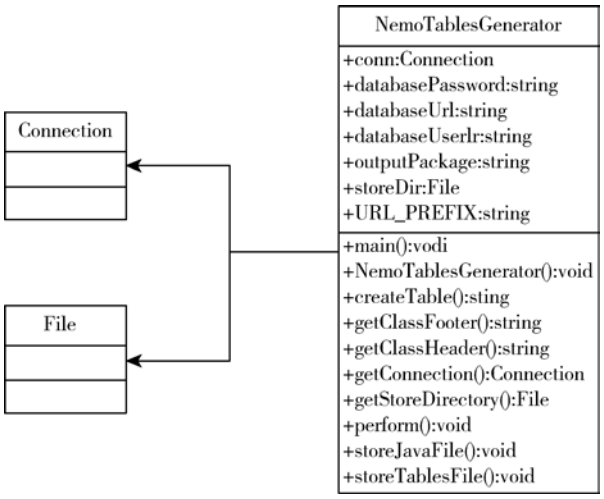
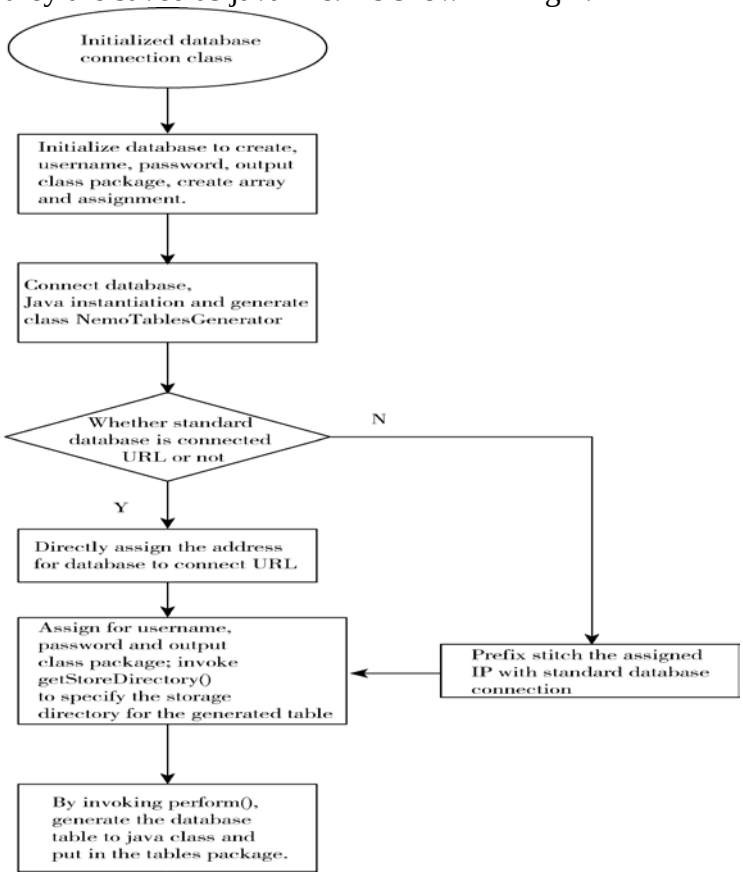


Fig 2. Class Diagram for NemoTablesGenerator.java

Database Connection: Hereunder is the detailed method to connect database and invoke to Java class. Firstly, define the object, and define array in the main function, then by manual assigning, write out the IP of the connecting computer, username and password of logging oracle and the package name. By invoking getStoreDirectory(), we can specify the storage directory for the generated table. By invoking perform (), we can generate the selected computer users' oracle table to JAVA and put in table package. The flow is shown as Fig 3.

Table Mapping: Now we see how the class generating TABLES.java file is defined. This class is used to generate TABLES.java file, including which are all the table names of connected users, and they are saved as java file. As shown in Fig 4.



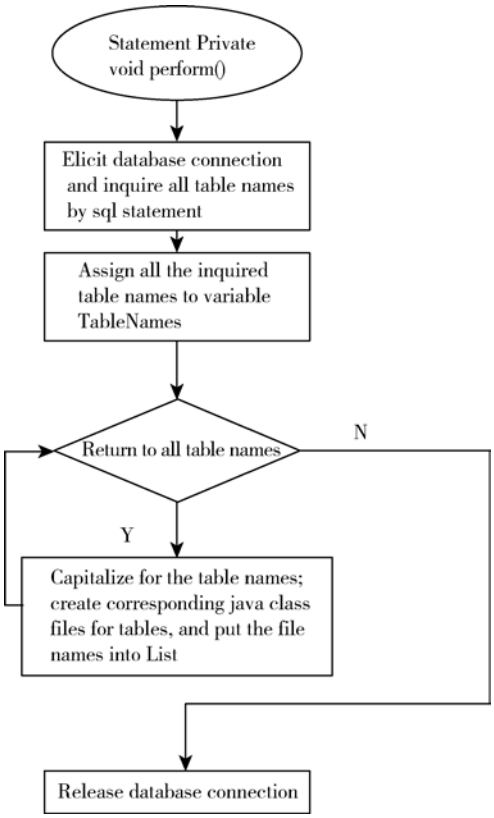


Fig 3. Flow Diagrams for Database Connection Fig 4. Table Mapping Flow

Nemo Interface Design and Realization

Java language provides a way of interface. Interface defines needs but realizes it by class that inherits this interface, which has increased the readability for program. Program cannot be without

interface. Nemo generates and operates database by invoking sub class through interface and returns to parent class. These classes are for users, and users write java by themselves and automatically generate SQL statement by interface. Classes provided for users are as shown in Table 1:

Table 1 Nemo Interface

Package name	Class name	Type	Function
cn.liaoningjusoft.nemo.user	CondList	Class	Condition list for one statement, which is needed for the statement execution.
	SqlBuilder	Abstract Class	Actual main body for SQL creation, which is needed for user realization by inheriting this class.
	XField	Interface	Represent for any kind of fields
	XStatement	Interface	Represent for any kind of statements
	XTable	Interface	Represent for any kind of tables
cn.liaoningjusoft.neme.user.statment	DeleteSatement	Interface	Delete statement interface
	InsertStatement	Interface	Insert statement interface
	SelectStatement	Interface	Select statement interface
	UpdadeStatement	Interface	Update statement interface

The database connection is extremely critical and very limited but precious resource, which is very previous in many web application programs. In terms of the flexibility and robustness, even the performance indicator of the whole program, how to manage the database connection will take great effects. Aiming at such kind of problems, database connection pool comes into being. It provides 7 methods, and its main function is to represent one field.

setFieldName(String) is used to set field name in string type. getFieldName() is used to acquire the set field name. The same methodology for buildSQL(). setStatement(Xstatement) is used to set Statement object. getStatement() is used to acquire Statement object. setTable(Xtable) is used to set table name. getTable() is used to acquire table name. The method provided by this interface mainly focuses on Select and Update. The inserting field is the corresponding updating content in database. As shown in Fig 5:

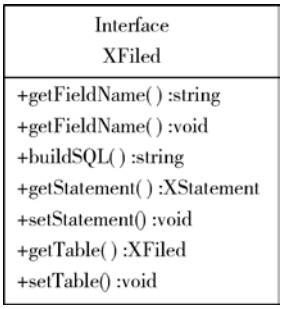


Fig 7. Class Diagram for XField Interface

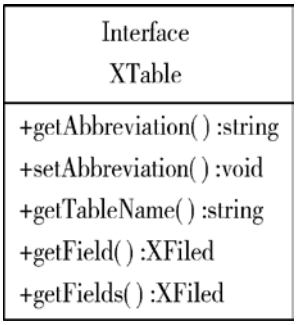


Fig 8. Class Diagram for XField Interface

To deal with table, getAbbreviation() is used to acquire alias. setAbbreviation() is used to set alias for generated table. getTableName() is used to acquire table name. getField(String) can acquire the field name in string type. getFields() is used to acquire all fields in the table.

As shown in Fig 8:

Conclusion

Nemo is a Java-based ORM component. After testing, codes can run normally and the interface design can reach the basic requirement. Basically, its design is successful. Its characteristic is to release programmers from large number of SQL coding. Its main advantages are summarized as below:

When used under regular work, it is easy to operate. Users can conduct database operation without being familiar with SQL statement. The generation of database language can be realized through the relation between object and method.

No need to install database. There is no need to install the huge database system during the programming stage, which can save the resource space and improve the efficiency.

The most important is that this component realizes the reusability. No matter how complicated the code is, as long as it is written for once, it can be reused by invoking. This is also the initial purpose and idea source for the component design.

Reference

[1] Martin Flower. Patterns of Enterprise Application Architecture [M]. China Machine Press, 2004

[2] Yiqing Qin. Research on Common Method for Object Persistence. (2003)Journal of Beijing Information S&T University, 3:20-24.

[3] Tiancai Liang, Youguo Pi. (2005) ORM Design Mode for J2EE Data Persistence Layer [J]. Shenzhen Institute of Information Technology, 6:15-19.

[4] Xiaoxiang Zhang. Java Employment Training Tutorial [M]. Tsinghua University Press, 2003.9

[5] Qin Xia, Xiaogang Cao, Yong Tang. (2005) Head First Hibernate[M]. Publishing House Of Electronics Industry，6:20-186.

[6] Shaofang Yang. (2005) Go Deep Into J2EE Technology [M]. Science Press, 5:167-172.

[7] Zhuo Zhou, Shuping Tao.(2005) Application of Persistence Object Technology in Software Development [J]. Ningxia Engineering Technology, 2: 187-190.

[8] Ivor Horton, Xiaolei Pan, Junbo Yu, Dan Wang(translators).(2006) Beginning JAVA2 [M]. China Machine Press, 248-260