Persistence on Score Management of Japanese-Language Proficiency Test Based on NHibernate

Fengjuan Liu

Teaching and Research Institute of Foreign Languages, Bohai University, Jinzhou, P.R. China

751076834@qq.com

Keywords: NHibernate; Japanese-Language proficiency test; score management; persistence

Abstract. The Japanese language proficiency test is the largest Japanese examinations in the world. This paper based on the NHibernate technology of .NET platform for the score management system to conduct the persistence layer design, to provide technical support for software development. Firstly, research on NHibernate architecture, to clarify the relationship between the various parts; then, in accordance with the score management needed information for database logical structure design; finally, conduct persistent design for logical structure according to NHibernate configuration steps. Contents of this paper solves the key technical issues in the process of software development, developers can fully in accordance with the object model to manipulate the database, and laid the foundation for rapid software development.

Introduction

The Japanese-Language Proficiency Test (JLPT) is established, a more complete examination and evaluation system, by the Japan International Exchange Foundation and Japan International Education Support Association in 1984, in the same year started and implemented in the relevant countries and regions. The exam in the mother tongue is not Japanese people as objects, test their Japanese ability. Test includes four levels: one, two, three, four grade, one grade is approximately equivalent to the level of professional Japanese 3-4 grade in Chinese university. Each level of the test was divided into three parts: words and vocabulary, listening, reading and grammar [1,2]. According to passing score of comprehensive scores and individual scores to assess whether the test is passed.

Persistence is the data saved to the powered down storage device for future use. Persistence is the logical level at the system used to implement data persistence a relatively independent fields, is the data in the database to store, retrieve, update, and delete a set of classes and components [3]. In order to improve the information level of test information management, the development of score management system, persistence is the basis work for system development. This paper based on the NHibernate technology of .NET platform for the score management system to carry on the persistence layer design, to provide technical support for the software development.

NHibernate Architecture

Object-Relation Mapping (ORM) is the original operating directly in the database evolved into the operating of the class properties and methods, it is to solve the contradiction that is a mismatch between object-oriented programming and relational databases, and the resulting technology, programmer can use a very simple way to the object in the program automatically persist to the relational database. NHibernate is a popular O/RM framework in the field of.NET, NHibernate originated in Hibernate. Hibernate is an object-relational mapping framework based on JAVA platform, due to hibernate achieved great success, Microsoft is using its basic idea developed NHibernate, is based on object-relational mapping framework for .NET platform [4]. NHibernate frame architecture is shown in Fig. 1.

As can be seen from Fig. 1, NHibernate is between Database and Application, provide persistent object for the program application to the database service. NHibernate separates Database and Application, so that application portability is achieved. The main elements of Fig. 2 are described as

follows: persistence operations with three query mechanism, namely HQL Query, QBC Query and Native SQL Query; configuration file is mainly used to configure some global parameters; mapping file to class as the core element, specified the mapping between classes and database table, sub elements specified the mapping between class attribute and fields of database table; session and the session factory is a connecting link when the runs between the oriented object model and relational model; transaction processing to achieve the underlying ADO.NET, ODBC, OLEDB related API encapsulates.

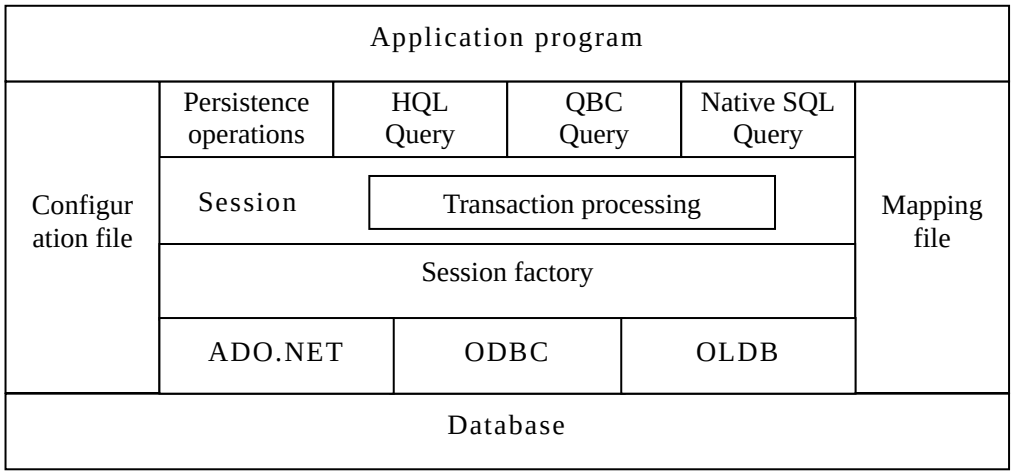

Fig. 1. NHibernate FrameworkArchitecture

Database Logic Structure Design

According to the Japanese language proficiency test scores need to manage the information, based on SQL Server database management system for database logical structure design, the results is shown in Table 1. Table named "JLPTScoreManagement".

SQL Server provides a variety of data types; the system uses five kinds of data types, namely: Decimal, precision decimal type. Storage space occupied by the data based on the integer digits and decimal places to determine, on both sides of the decimal point total storage 38; Smallint, short integer type. Each value occupies storage space of two bytes, storage data range is less than Int type, greater than Tinyint type; Datetime, date time type. Datetime data is composed of the effective date and time. The Datetime type takes up 8 bytes of storage; Varchar, variable-length character type. The character data is composed of the arbitrarily combination by letters, symbols and figures. Data length of Varchar types does not exceed 8KB; Text, text types. All data is ASCII characters, and and the data type is selected when the length is over 8KB in general.

Table 1. Logical structure on score management of JLPT

No	Field name	Data type	Byte	No	Field name	Data type	Byte
1	PrimaryKey	Decimal	10	12	ArrivalTime	Datetime	8
2	TicketNumber	Varchar	20	13	ExaminationTime	Datetime	8
3	ExaminationGrade	Varchar	2	14	LanguageScore	Smallint	2
4	ExamineeName	Varchar	30	15	LanguageStandard	Smallint	2
5	CertificatesNumber	Varchar	20	16	ReadingScore	Smallint	2
6	ExamPlaceName	Varchar	50	17	ReadingStandard	Smallint	2
7	ExamPlaceCode	Varchar	5	18	ListeningScore	Smallint	2
8	ExamAreaName	Varchar	20	19	ListeningStandard	Smallint	2
9	ExamAreaCode	Varchar	4	20	TotalScore	Smallint	2
10	ExaminationAddress	Varchar	200	21	TotalStandard	Smallint	2
11	TrafficRoutes	Varchar	200	22	Remarks	Text	

Step and Instance of Persistence

According to NHibernate configuration steps, the logic structure of Table 1 is persistent [5-8].

(1) The reference component. At least four components are cited, such as NHibernate.dll, Log4net.dll, Iesi.Collections.dll, and Castle.DynamicProxy.dll. Wherein, NHibernate.dll is the core assemblies, Log4net.dll is logging record assembly, Iesi.Collections.dll is the collection framework assembly, and Castle.DynameicProxy.dll is the control reverse assembly.

(2) Configure NHibernate. Configuration information is used to specify the manner in which access to the database. According to the configuration information, NHibernate dynamically create data access objects and interact with the database. There are two configuration methods: one is, to be configured in the application configuration file. The other is, the individual using the configuration file for configuration.

(3) Create the persisted .Net class. NHibernate by reflection of object properties to work, so it is needed to add a object properties of persistence. Code is as follows:

```
using System;
namespace NHibernate.Examples.ScoreManagement
{public class JLPTScoreManagement
    {
  // object properties
  private Decimal PrimaryKey;
  private String TicketNumber;
  // omitted 9 Varchar type fields
  private Datetime ArrivalTime;
  private Datetime ExaminationTime;
  private Int16 LanguageScore;
  private Int16 LanguageStandard;
  private StringClob Remarks;
  // omitted 6 Smallint type fields
  // attribute function
  public Decimal PrimaryKey
  {   get {return PrimaryKey;}
  set {PrimaryKey=value; }
  }
  public String TicketNumber
  {   get {return TicketNumber;}
  set {TicketNumber=value; }
  }
  // omitted 6 Varchar type field functions
  public Datetime ArrivalTime
  {   get {return ArrivalTime;}
  set {ArrivalTime=value; }
  }
  public Datetime ExaminationTime
  {   get {return ExaminationTime;}
  set {ExaminationTime=value; }
  }
  public Int16 LanguageScore
  {   get {return LanguageScore;}
  set {LanguageScore=value; }
  }
  public Int16 LanguageStandard
  {   get {return LanguageStandard;}
  set {LanguageStandard=value; }
  }
  //omitted 6 Smallint type field functions
  public StringClob Remarks
  {   get {return Remarks;}
  set {Remarks=value; }
  }
}   }
```

(4) Construction of the mapping file. It takes a way to get the NHibernate to know how to associate the data tables to the .Net class, which relies on the mapping file to complete. The easiest approach to manage is to write a mapping file for each class, if the name is YourObject.hbm.xml and put it and classes in the same directory, NHiberante operation is very simple. Mapping file is as follows:

```
<?xml version="1.0" encoding="utf-8" ?>
<hibernate-mapping xmlns="urn:nhibernate-mapping-2.0">
  <class name="NHibernate.Examples. ScoreManagement. JLPTScoreManagement, NHibernate.Examples" table=" JLPTScoreManagement ">
    <id name="PrimaryKey" column="PrimaryKey" type="Decimal" length="10">
      <generator class="assigned" /> </id>
    <property name="TicketNumber" type="String" length="20"/>
    // omitted 9 Varchar type fields
    <property name="ArrivalTime" type="DateTime"/>
    <property name="ExaminationTime" type="DateTime"/>
    <property name="LanguageScore" type="Int16"/>
    <property name="LanguageStandard" type="Int16"/>
    // omitted 6 Smallint type fields
    <property name="LanguageStandard" type="StringClob" length="2147483647"/>
    <property name="Remarks" type="StringClob" length="2147483647" />
  </class> </hibernate-mapping>
```

(5) Using NHibernate API for programming. Declared Factory of NHibernate in class; and loaded XML during system initialization, and create Factory file; use HSql query; compile specific insert, modify, delete, query statement. NHibernate API to access the database operations design divided into four steps: First, create a Configuration object via "new Configuration()", to resolve the mapping relationship between all .NET objects and the database; the second is through the "config.BuildSessionFactory()" to create a SessionFactory, through "factory.OpenSession()" method to obtain Session instance; the third is by calling the Session interface operation database "Save(), Update(), Delete(), Load(), Find()" and other methods to achieve the interaction between the object and relational database; Fourth, whether the transaction is executed successfully or fails, call the close() method to release resources by Session instance occupied.

Conclusion

The scope of examinee that participate in Japanese Language Proficiency Test is continue to expand, the test object is also increasing, in addition to test the Japanese level, there is for employment, upgrade, promotion, and others[9]. The requirements and suggestions for testing are also more and more. Score management system can provide examinee with convenient and efficient information service. Contents of this paper solve the key technical problems in the process of software development. NHibernate provides a more complete package for database structure, the database schema mapping for more complete object model that supports encapsulation and continuing mechanism, the function is more powerful, the flexibility is higher than the average of ORM. Based on NHibernate framework to design the persistence layer of Japanese language proficiency test score management system, developers can fully in accordance with the object model to manipulate the database, which lays the foundation for the rapid development of software.

References

[1] Q. Cheng, L. Liu, "On the International Japanese Proficiency Test examination tactics," Intelligence, vol. 14, no. 31, pp. 279, 2014.

[2] C. X. Li, "Countermeasures to deal with the Japanese ability test N2 language knowledge," Journal of Jiamusi Education Institute, vol. 30, no. 8, pp. 343-344, 2013.

[3] G. S.Deng, "Research and Application of SaaS management platform based NHibernate," Master's degree of Heefei University of Technology, 2013.

[4] M. Q. Wang, L. Q. Liu, H. Y. Wang, "Application of ORM Nhibernate in the exam administration system," Journal of Changchun University of Technology (Natural Science Edition), vol. 30, no. 3, pp. 352-356, 2009.

[5] Web developer, "NHibernate Quick entry," http://www.admin10000.com/document/2404.html, 2015-6-10.

[6] Z. Z. SunH. J. Y. Wei, Y. B. Qin, "Research and application of software architecture based on WCF and NHibernate," Computer & Digital Engineering, vol. 43, no. 4, pp. 591-595, 2015.

[7] J. F. Tong, "Data persistence on tourism resources management system based on NHibernate," Electronic Test, vol. 21, no. 20, pp. 12-13, 2014.

[8] J. G. Luo, E. X. Zhang, "Design and implementation of university laboratory information management system based on NHibernate," China Educational Technology & Equipment, vol. 28, no. 10, pp. 33-36, 2014.

[9] Baidu Encyclopedia, "The Japanese-Language Proficiency Test," http://baike.baidu.com/link?url=n8XBGwW6slEdEr9atsvNtj4-i8uhyWDdAooTOMqI7SQFAUt_6kxe5_nLqaDYYNZz4pt5VuX9n6bazhSvu-YiyK, 2015-6-10.