在上一章中对在线考试系统进行需求分析，确定了整个系统的架构。本章将介绍系统的详细设计，包括数据库的详细设计以及一些关键模块的详细设计。

## 数据库详细设计

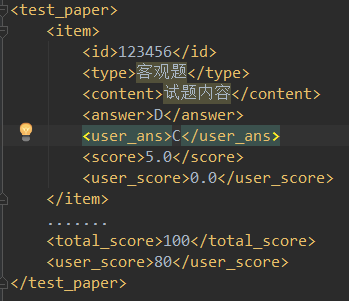
通过第三章对系统的整体进行设计和数据库的需求分析，我们确定采用MySQL数据库存储本系统涉及到的表格。管理员信息表 ADMIN\_TABLE，教师信息表 TEACHER\_TABLE，学生信息表 STU\_TABLE，知识点表KPOINT\_TABLE，题目表 QUESTION\_TABLE，试卷表TESTPAPER\_TABLE，试卷模板表 PAPERMODULE\_TABLE，课程表SUBJECT\_TABLE。由于篇幅的限制，下面只给出一些关键表的详细结构。

1. 题目表QUESTION\_TABLE、知识点KPOINT\_TABLE，课程表SUBJECT\_TABLE这三个表格都是由教师进行维护，每个题目都有所属课程的ID以及所要考察知识点的ID。表XXX是题目表的详细字段，每个题目都有唯一的标识ID，同时记录了录入该题目的创建者ID和录入时间，以及用于描述题目的一些字段，例如难度、内容、类型等。创建者在录入一个题目的同时，需要指定该题目的正确答案。对于客观题，题目答案中只需要直接给出答案内容，而对于客观题，需要给出测试数据以及标准代码。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| ID | BIGINT | 50 | 否 | 题目ID号 |
| CREATORID | BIGINT | 50 | 否 | 创建者ID |
| CREATEDATE | DATE |  | 否 | 创建时间 |
| DIFFICULITY | VARCHAR | 10 | 否 | 题目难度 |
| CONTENT | VARCHAR | 5000 | 否 | 题目内容 |
| TYPE | VARCHAR | 5 | 否 | 题目类型 |
| KPOINTID | BIGINT | 50 | 否 | 知识点ID |
| SUBJECTID | BIGINT | 20 | 否 | 课程ID |
| ANSWER | VARCHAR | 5000 | 否 | 题目答案 |
| SCORE | VARCHAR | 50 | 否 | 参考分值 |

1. 试卷表 TESTPAPER\_TABLE用于存放考卷信息，是根据预先设定的考试模板生成的，其中MODULEID为对应试卷模板ID号，ISCHECK表征试卷是否已经打分，其中TEST\_RESULT和TEST\_CONETNT字段采用XML的格式存储考卷内容以及考生答题后的试卷，格式如下。。所示。<test\_paper></test\_paper>为整个试卷，<item></item>用于标识一道题目，其中id对应的题目表中的ID号，type为题目的类型，content为题干，answer为题目的答案而user\_ans为考生给出的答案，score为改题目的分值，user\_score为考生得分。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段长度 | 是否为空 | 描述 |
| ID | VARCHAR | 50 | 否 | 试卷ID号 |
| TITLE | VARCHAR | 50 | 否 | 试卷名称 |
| STATE | VARCHAR | 10 | 否 | 是否有效 |
| SUBJECTID | VARCHAR | 20 | 否 | 课程ID |
| MODULEID | VARCHAR | 20 | 否 | 试卷模板ID |
| STUDENTID | VARCHAR | 20 | 否 | 考生ID |
| TEST\_RESULT | VARCHAR | 100000 | 否 | 提交试卷内容 |
| ISCHECK | VARCHAR | 10 | 是 | 是否已经评阅 |
| SOCRE | VARCHAR | 20 | 否 | 考试成绩 |



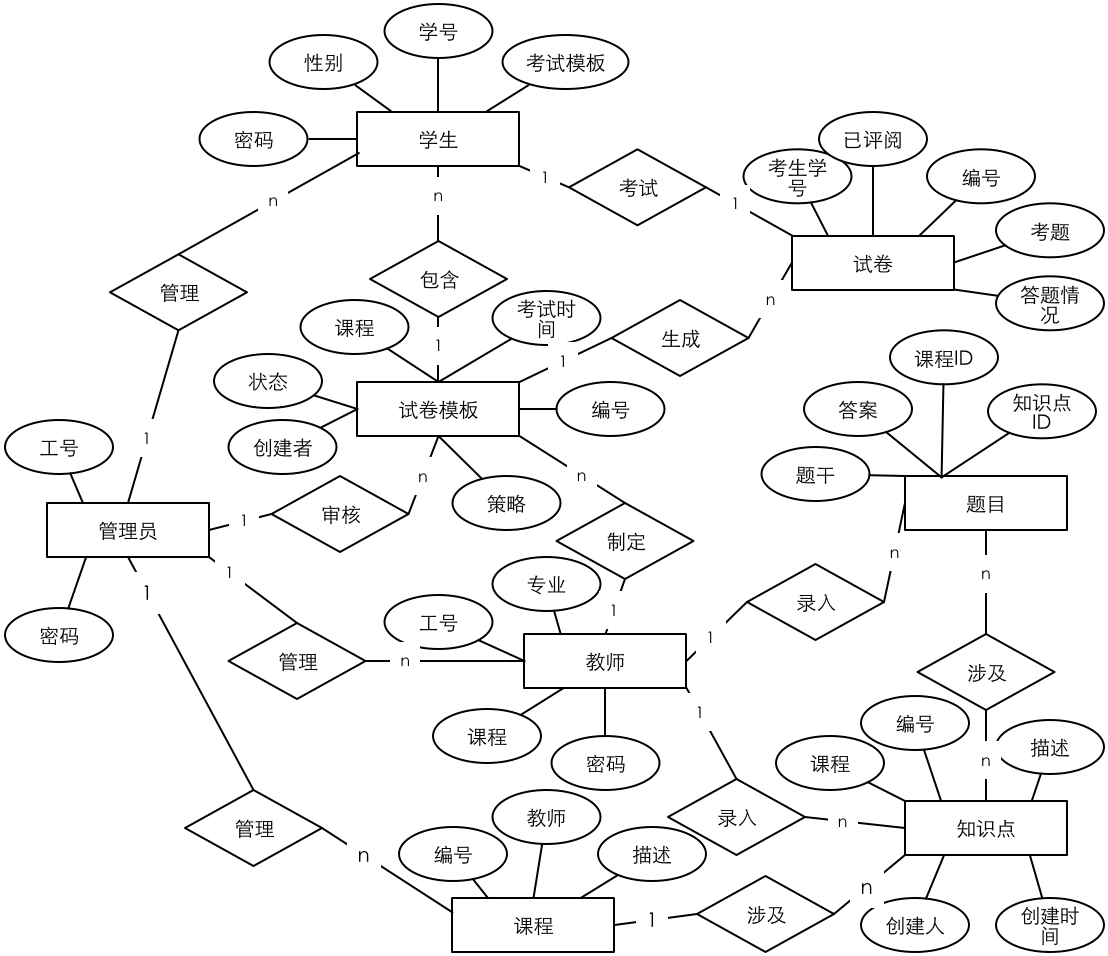
1. 试卷模板表 。教师制定的生成试卷的策略，POLICYCONTENT为组卷策略描述，仍采用XML格式进行描述。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段长度 | 是否为空 | 描述 |
| ID | VARCHAR | 50 | 否 | 模板ID号 |
| CREATORID | VARCHAR | 100 | 否 | 创建者ID |
| CREATEDATE | DATE | 7 | 否 | 创建时间 |
| TITLE | VARCHAR | 50 | 否 | 试卷名称 |
| STATE | VARCHAR | 10 | 否 | 是否审核通过 |
| SUBJECTID | VARCHAR | 20 | 否 | 课程ID |
| POLICYCONTENT | VARCHAR | 500 | 是 | 组卷策略 |

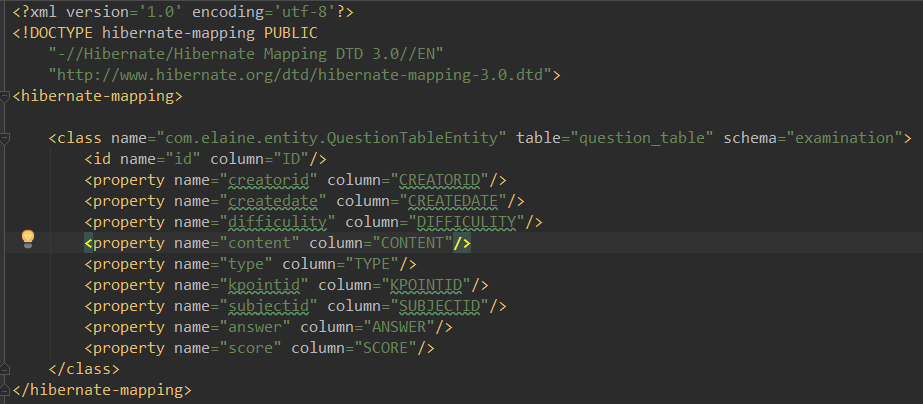
4.学生信息表，由学生自己通过学号注册或者管理员代为注册，并设定在线考试系统的登录密码。其中EXAM\_MODULE默认为null，只有当教师或者管理员将对应考试模板ID号添加到该字段，考生才能够参加对应的考试。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段长度 | 是否为空 | 描述 |
| ID | VARCHAR | 50 | 否 | 考生ID号 |
| PWD | VARCHAR | 50 | 是 | 密码 |
| NAME | VARCHAR | 50 | 否 | 考生姓名 |
| MAJOR | VARCHAR | 50 | 否 | 专业 |
| GRADE | VARCHAR | 50 | 否 | 年级 |
| GENDER | VARCHAR | 10 | 否 | 性别 |
| BIRTHDAY | DATE | 7 | 是 | 出生日期 |
| ID\_NUMBER | VARCHAR | 20 | 是 | 身份证号 |
| EMAIL | VARCHAR | 50 | 是 | 电子邮箱 |
| PHONE\_NO | VARCHAR | 50 | 是 | 联系电话 |
| PHOTO | VARCHAR | 200 | 否 | 电子照片地址 |
| EXAM\_MODULE | VARCHAR | 50 | 是 | 考试模板ID |

上面介绍了主要的数据表，为了更加全面的了解整体的数据结构，我们采用实体-关系图(Entity-Relationship Diagram)的方式进行描述，如。。所示。



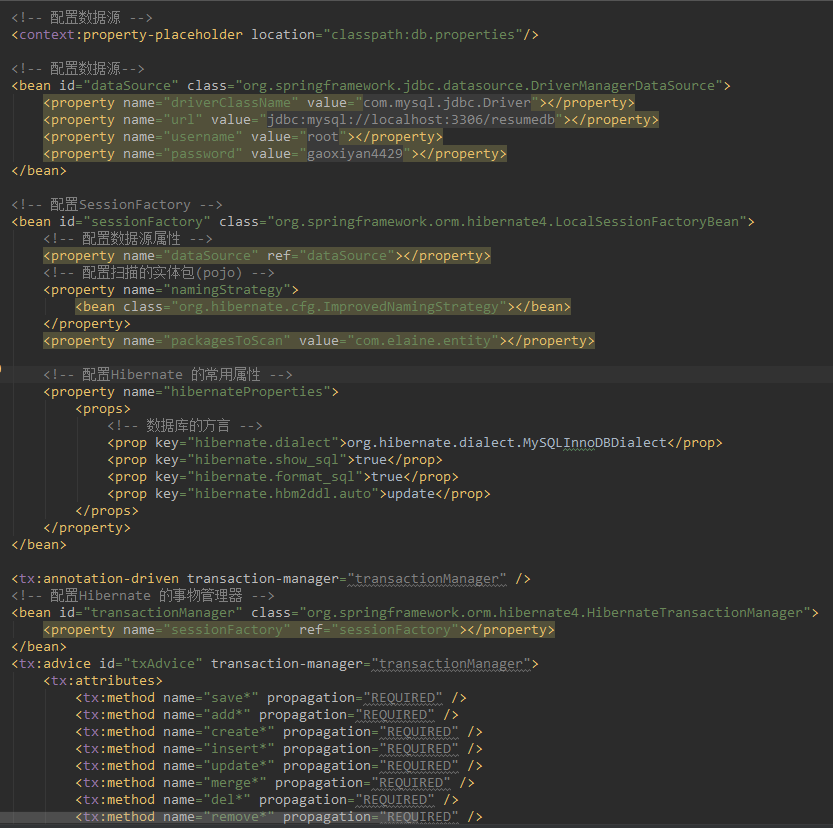
本文采用内Hibernate实现数据层的持久化，建立实体类与数据表之间的映射关系，从而避免开发人员采用JDBC的方式直接操作数据库，从而保证整个系统遵从面向对象的方式进行开发。Hibernate通过hbm.xml格式的文件将POLO类映射到数据库的表格中，不需要编写任何额外的代码。IntelliJ IDEA支持将数据库中的表格直接转化为POLO类，大大减少了开发的工作量。POLO类对应数据库表格，每个POLO类的实例对应数据库中的一行记录，POLO类的属性对应数据库表的字段。图。。就是QuestionEntity与QUESTION\_TABLE的映射文件question.hbm.xml。这个文件通过class name与table建立联系，并将POJO类的属性一一与table的字段对应，包括名称以及数据类型。之后对数据库中数据的操作就可以简单通过操作POJO类来实现。



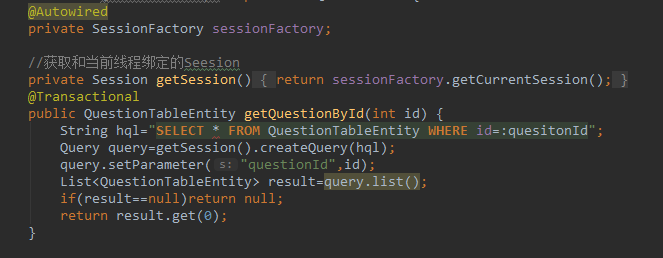
单纯使用Hibernate进行事务处理，每次对数据的操作都需要经历8个步骤。首先需要获取Configuration对象，接下来通过Configuration对象创建sessionFactory对象，并通过sessionFactory对象创建session，然后通过session开启事务，进行一些数据持久化操作，处理结束后需要提交事务，如果发生异常需要进行事务回滚，最终关闭session，对应代码如下：

**public void** getPersistence(){  
 Configuration configuration=**new** Configuration().configure();  
 SessionFactory sessionFactory=configuration.buildSessionFactory();  
 Session session=**null**;  
 **try**{  
 session=sessionFactory.openSession();  
 session.beginTransaction();  
 *//数据持久化操作* ......  
 session.getTransaction().commit();  
 }**catch**(Exception e){  
 e.printStackTrace();  
 *//进行事务回滚* session.getTransaction().rollback();  
 }**finally**{  
 **if**(session!=**null**){  
 **if**(session.isOpen()){  
 session.close()  
 }  
 }  
 }  
}

而用Spring对Hibernate进行管理时会大大简化这个步骤，Spring提供了HibernateTemplate类，这个类对Session等进行了封装，并且会自动提交事务，不需要开发者手动开启事务和提交事务，只需要将对象进行save操作即可。图X是Spring对Hibernate的配置。



在Spring配置文件中加入Hibernate相关的配置之后，对数据持久化的操作如下图所示。只需要在使用sessionFactory之前对其进行类型声明，后续可以通过getCurrentSession得到session对象，然后进行数据库的相关操作，Spring会进行后续的处理。



## 系统功能模块详细设计

### 系统登录详细设计

系统登录是整个系统的入口，用户需要输入用户名和密码，点击登录按钮提交。如果提交的信息没有验证通过，则会提示用户名或者密码错误，用户需要重新输入用户名和密码；否则系统会根据用户的账号判定用户的身份，并跳转到对应的系统页面，详细流程如。。。所示。



### 学生在线考试详细设计

学生在线考试是整个系统的核心模块，考生在登录考试系统后，需要点击在线考试选项，系统会判断该学生是否具备考试资格。如果学生不具备考试资格，则直接退出；如果具备考试资格，则系统会根据试卷模板生成试卷并开始计时。如果在考试过程中，考生主动点击提交按钮，考试开始之后半个小时才能够提交试卷，否则继续答题。另外，到达考试时限后，系统会自动提交试卷结束考试，流程如。。所示。



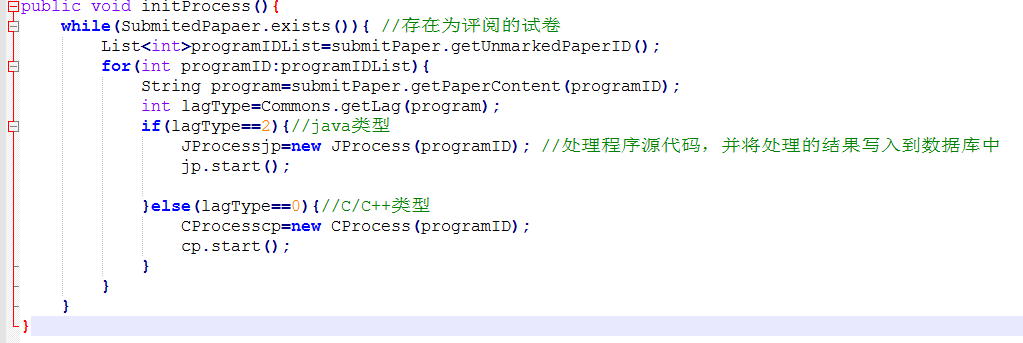
### 教师管理部分详细设计

## 系统关键模块设计

自动组卷

### 自动评阅试卷

在学生提交试卷后，系统会直接读取学生提交的试卷答题信息，编译运行检查结果并将结果写入到数据库中。在编程部分的试题进行评阅时，需要调用服务器上安装的编译器来编译考生提交的源代码，并保证在编译提交代码发生问题后，系统不会出现致命错误。为了达到该目的，需要引入一个守护线程，负责启动评阅试卷的线程，每个线程负责启动对应语言的编译器对程序进行编译和运行。守护线程负责每个一段时间从数据库中检查是否有未评阅的试卷，如果有则启动线程对提交的结果进行评阅。守护线程的关键代码如下图所示：



如果源程序编译之后能够运行并产生结果，则将产生的结果与标准答案进行对比。若与标准答案完全一致，则满分；若部分正确，则按照正确的比例给分；若得分为0，则将学生提交的代码与标准代码进行比对评分；如果程序未通过编译，无法产生运行结果，则转入程序修复。

C/C++评判线程

评判线程首先启动GCC编译器对源代码进行编译，然后指定测试数据文件作为编译后.exe文件的输入，并得到运行结果。