装

订

线

同濟大學
TONGJI UNIVERSITY

“Java 语言学习平台”项目报告

学　　院：软 件 学 院

专　　业：软件工程专业

学生姓名：王 诗 腾

学　　号：2052995

联系方式：17879683118

上课时间：周二 三、四节

任课教师：贺 向 东

2022 年 12 月 10 日

目 录

1 项目概述

本项目通过服务器端、客户端、管理员端的交互，以 MySQL 为数据库平台、Java Swing 为前端交互平台，结合推荐算法、对象数组化传输等技术，合理分工、紧密合作，成功完成了一个基于知识和习题数据的 java 语言学习平台，达成了各项要求与功能，是一个较为成熟的软件项目。

1.1 项目整体功能概述

总体而言，本项目具备以下功能，课程所要求的基本功能包含在内：

1. **客户端**

   (a) **自主学习功能**　用户在客户端可以自主按需学习知识库的内容，默认以图片的形式展示知识。目前，知识库共分为十一个大章节、若干小章节。

   (b) **章节练习功能**　用户学习完每一章节后，都可以对应地练习本章习题。默认服务器发送六道选择题、六道填空题、六道判断题、两道编程题。

   (c) **综合复习（推荐）功能**　根据用户学习过程记录反映出来的数据，动态生成综合复习的若干习题，用来巩固此前学习的知识。习题的生成（推荐）算法基于用户在这些章节中刷题的正确率和章节的重要性等因素。

   (d) **学习评价功能**　每次做完题后，客户端将显示学习评价功能，包括正确答案、做题时间、正确率等信息。需要指出的是，填空题和编程题被视为主观题，将由用户自主评判答案是否正确。同时这些学习评价信息将上传至服务器数据库。

   (e) **学习、复习推荐功能**　十一个章节间，有着知识依赖和学习上的先后关系（在图论上可称为拓扑关系），每次学完一章后，将根据目前学习状况来推荐可以学习的下一章节，同时考察拓扑关系中上一章的学习情况，决定是否推荐复习上一章节。

2. **管理员**

   (a) **习题库增删查改功能**　这里所述的习题库，包含所有题目的章节号、题型、题号、题干描述等信息，题目选项和答案等信息在数据库其他表和其他界面中体现。管理员可以对任何题目进行查阅和删除、对其题干描述进行更新，还可以插入新的题目。

   (b) **各题型题目其他维护功能**　出于数据库的主外键依赖关系和功能拆分的理念，我们将各个题型的非题干部分单独设置页面进行维护，可以对这些字段进行查询和更新。但题目本身的增加和删除仍旧在上一功能里实现。

(c) **知识库文件的增删查改功能**　可以查看和更新知识库中的图像文件，同时可以对每一小节的文件进行尾插和尾删功能，但不能从中间直接进行删除和插入。

3. **服务器**

客户端和管理员的每项功能，都**必须**配合**服务器**的相应响应和数据支持才能实现。

(a) **知识库文件发送功能**　本项目中，知识库文件以图像的方式存储在服务器中。服务器发送知识架构给客户端，然后根据客户端调取章节的需求，逐张以字节流的方式发送给客户端。

(b) **习题库信息发送功能**　根据客户端拉取题目的需求，将若干题目的章节号、题型、题号、题干、答案等信息整合为 byte 流，发给客户端进行显示。

(c) **学习过程记录功能**　客户端每次做完习题之后，都会将数据发送给服务器，服务器再转储至数据库，完成学习过程记录。

(d) **历史学习情况发送功能**　为了配合客户端做学习章节推荐、复习章节推荐、综合复习习题推荐等功能，服务器将查询出指定用户的特定学习情况发送给客户端，通过算法决定如何推荐。

装

订

线

1.2 个人分工

我们本着合理分工、紧密合作的原则，将上述功能点按照结合程度进行了分工，其中笔者主要负责**数据库的实施、管理员窗体的设计和服务器的构建**，实现了 1.1.2 和 1.1.3 节所述的各项功能，具体如下：

1. **知识库和习题库的数据准备与处理**
2. **数据库的设计与实施**
3. **服务器与客户端的信息交互**
4. **服务器与管理员端的信息交互**
5. **服务器向数据库查询的 SQL 语句设计**
6. **管理员端的 GUI 设计**
7. **管理员端对数据库的增删查改功能设计**

1.3 项目使用的主要技术和特色

笔者在进行本项目前，已有开发数据库项目、Java 可视化编程、C++ 网络通信项目开发、Python 机器学习等学习研究经历，笔者将现有知识同项目实际情况相结合，同时边做边学、以做促学，主要使用了以下技术进行开发和辅助开发：

1. **数据处理和数据库实施**
   - Python 正则表达式
   - Python 网络爬虫
   - MySQL 关系数据库的设计与实施
2. **Java 服务器构建**
   - 基于 JDBC 的数据库访问技术
   - 基于 Socket 的连接通信程序设计
   - 多线程技术
   - 基于 Serializable 接口的 Java 对象传输技术
3. **Java 管理员窗体设计**
   - 基于 javax.swing 的控件设计
   - 基于 java.awt.event 的事件和监听器模式
   - 基于 javax.swing.table.AbstractTableModel 的数据表格样式设计

经过精心研究、缜密部署、严谨编程，本项目具备如下几个特色：

1. **程序稳定，健壮性高**　项目具备周密的异常处理，经过广泛的边界条件和极端情况测试，例如查询信息为空、信息输入不合规范、客户端强制断开等情况，程序均能保持正常状态，不会出现崩溃、卡死等错误状态。
2. **风格简洁、功能全面**　项目摒弃了眼花缭乱的界面设计，力图用最少的控件实现全面的功能，例如管理员窗口没有另外增加更新题目按钮，而是允许用户在表格中直接编辑。项目的增删查改功能一应俱全，同时遵守了数据库设计规范和数据库的完整性，保障了安全性和便捷性。
3. **代码规范、注释详尽**　项目源文件使用了规范、安全、简洁的代码风格，类名、方法名、变量名命名规范、见名知义。每个方法和类都配有详尽的注释，便于代码的理解和维护。

2 功能需求分析与设计

2.1 数据准备与清洗

本项目的实施依赖于完整、格式良好、基数较大的数据，笔者将在本节中简述笔者所做的相关工作。

本项目的难点之一就在于，并没有格式良好的数据供我们直接使用。笔者经过精心筛选，挑选了《Java 程序设计实用教程》（朱战立沈伟编著）作为知识库的数据来源，挑选了《JAVA 面向对象程序设计习题集》（主编姚骏屏）作为习题库的数据来源。

笔者先将《Java 程序设计实用教程》的 PDF 版书籍转化为图片集，再利用 Python 编写脚本，将图片裁剪为 1000×1250 的大小，同时将每张图片按规则命名，完成知识库的数据准备。

习题库的数据准备更为复杂。经过反复尝试，笔者首先将《JAVA 面向对象程序设计习题集》转化为文本文档格式，然后将各章的习题及其选项、答案分别导入两个不同的文本文档中。将习题集对应的问题和答案分别手动导入数据库是一个工作量极大、几乎不可能完成的工作，笔者编写了若干 Python 脚本，分别生成了所有题目、四种题型的插入语句。这里用到的技术主要是正则表达式。笔者观察到每道题都以“数字、+ 题干”的方式显现，其中选择题还有四个“选项、+ 答案”的题支，根据这个特征，笔者设计了正则表达式语句，将章节、题型、题号、题干、选项等信息提取出来，组合成正则表达式。

由于转化而来的 txt 格式文档常有错误，笔者又设计了若干正则表达式来捕捉并修复这些错误。例如，填空题中的下划线“_____”常常缺失形成空白，笔者需准确捕捉哪些空白占据的是填空的位置、哪些是单词间句子间正常的空白，然后予以下划线补充。

此外，习题中涉及到代码的部分往往格式较差，不具备 Java 代码规范的缩进格式。为此笔者在互联网上找到了某 java 代码在线格式化美化工具（http://tools.jb51.net/code/java_format）通过 Python 爬虫脚本，将习题库中代码批量上传至该格式化网站，进行格式化后再返回。

通过上述方法，笔者准备与清洗了近 300 张知识库图像，和近 4500 余道题目和答案，数据总量丰富、范围全面、格式规范，顺利完成了项目前期所必须做的工作。

2.2 数据库设计

数据库是本项目的基石。数据库的设计既要符合项目的功能需求，又要符合数据库技术的规范。

2.2.1 数据库的概念模式设计

经过对项目要求的细心揣摩和成员间的充分讨论，笔者做出数据库的 E—R 模型图如下（为方便显示，把 E-R 分为两张图分别显示）：

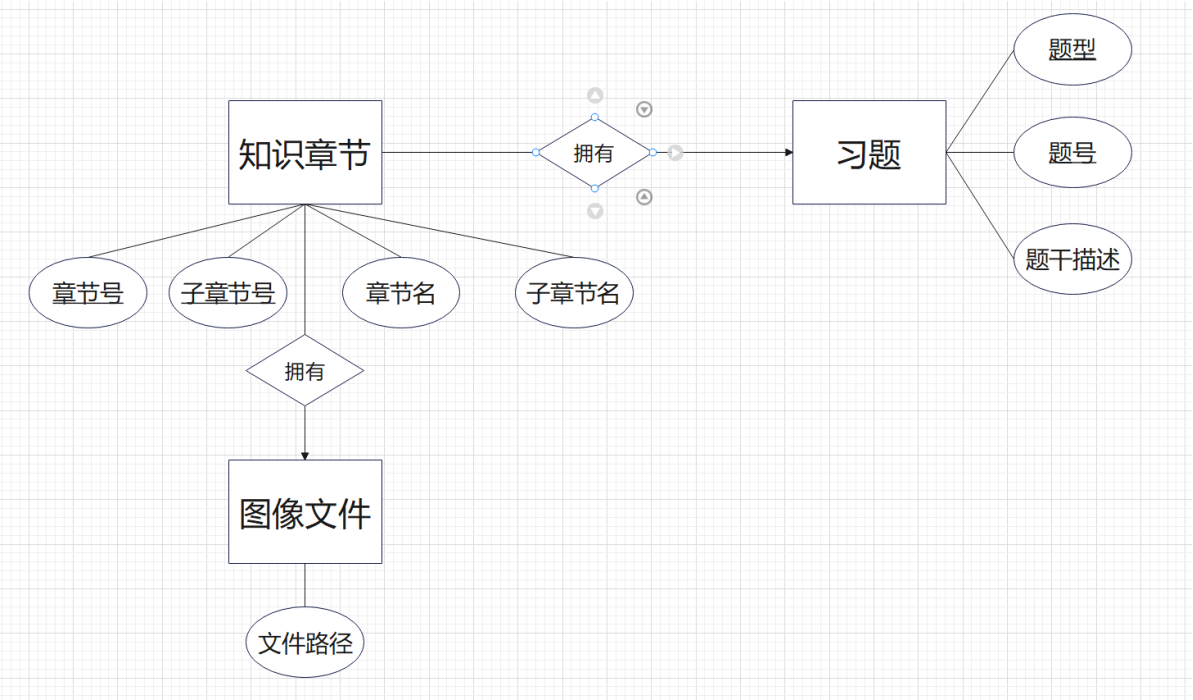

Figure 1: 知识库与习题库间的概念模式

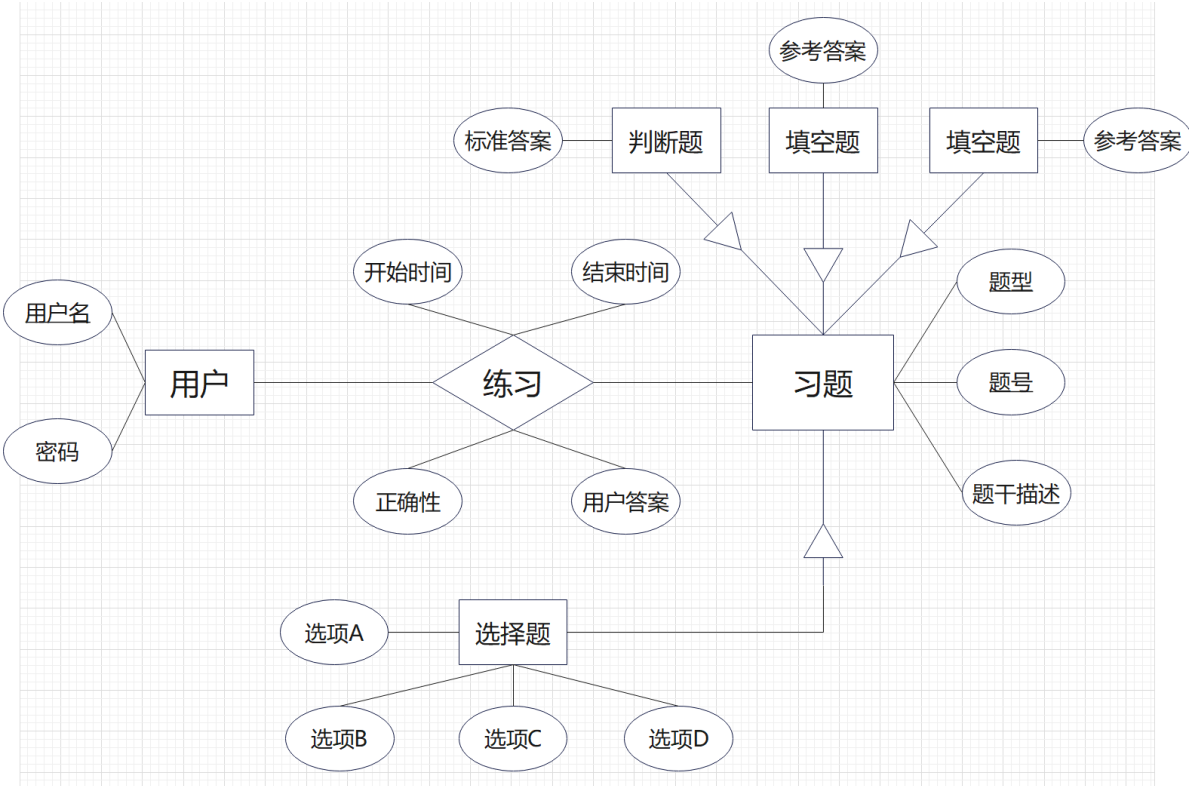

Figure 2: 用户与习题库间的概念模式

装 订 线

2.2.2 数据库的逻辑模式设计

- USERS 表（用户信息表）

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
USERNAME	用户名	char	10,not null		是
USERPASSWORD	密码	char	20,not null		

- KNOWLEDGE 表（知识框架表）

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null		是
SUBCHAPTER	子章节号	int	not null	是	
CHAPTERNAME	章节名称	int	40，not null		
SUBCHAPTERNAME	子章节名称	char	90,not null		

- CHAPTERFILES 表（知识章节文件表）-依赖于 KNOWLEDGE 表

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null	REFERENCES KNOWLEDGE	是
SUBCHAPTER	子章节号	int	not null		是
FILEPATH	文件路径	char	100, not null		是

- QUESTIONS 表（题目框架表）-依赖于 KNOWLEDGE 表

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null	REFERENCES KNOWLEDGE	是
TYPE	题型	char	(’ 选择题’,’ 填空题’, ’ 判断题’,’ 编程题’)		是
NUMBER	题号	int	not null		是

- ANSWERS 表（做题记录表）-依赖于 QUESTIONS、USERS 表

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
time	解题时间	TIMESTAMP	not null		是
timelength	解题时长	TIMESTAMP	not null		是
answer	用户解答	char	200,not null		
judge	对错与否	char	(’RIGHT’,’WRONG’)		
CHAPTER	章节号	int	not null	REFERENCES QUESTIONS	是
TYPE	题型	char	(’ 选择题’,’ 填空题’, ’ 判断题’,’ 编程题’)	REFERENCES QUESTIONS	是
NUMBER	题号	int	not null	REFERENCES QUESTIONS	是
USERNAME	做题人	char	10,not null	REFERENCES USERS	是

- selectquestions 表（选择题表）-QUESTIONS 的子类

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null	REFERENCES QUESTIONS	是
TYPE	题型	char	(’ 选择题’,’ 填空题’, ’ 判断题’,’ 编程题’)	REFERENCES QUESTIONS	是
NUMBER	题号	int	not null	REFERENCES QUESTIONS	是
sa	选项 a	char	255，not null		
sb	选项 b	char	255，not null		
sc	选项 c	char	255，not null		
sd	选项 d	char	255，not null		
answer	标准答案	char	’ A’,’ B’,’ C’,’ D’		

- fillquestions 表（填空题表）-QUESTIONS 的子类

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null	REFERENCES QUESTIONS	是
TYPE	题型	char	(’ 选择题’,’ 填空题’, ’ 判断题’,’ 编程题’)	REFERENCES QUESTIONS	是
NUMBER	题号	int	not null	REFERENCES QUESTIONS	是
answer	标准答案	char	255,not null		

- judgequestions 表（判断题表）-QUESTIONS 的子类

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null	REFERENCES QUESTIONS	是
TYPE	题型	char	(’ 选择题’,’ 填空题’, ’ 判断题’,’ 编程题’)	REFERENCES QUESTIONS	是
NUMBER	题号	int	not null	REFERENCES QUESTIONS	是
answer	标准答案	char	(’Y’,’N’)		

- programquestions 表（编程题表）-QUESTIONS 的子类

字段名	解释	数据类型	数据类型约束	外键约束	是否主键
CHAPTER	章节号	int	not null	REFERENCES QUESTIONS	是
TYPE	题型	char	(’ 选择题’,’ 填空题’, ’ 判断题’,’ 编程题’)	REFERENCES QUESTIONS	是
NUMBER	题号	int	not null	REFERENCES QUESTIONS	是
answer	参考答案	text	not null		

2.3 服务器连接数据库和多线程

项目中所有对数据的访问和维护需求，都通过 JDBC 访问和维护数据库来实现。为此服务器构建的第一步，就是连接数据库、启动相关服务。

本项目作为一个学习平台，在实际应用中面临着与多个客户端交互的需求。为此需要利用多线程技术，开放服务器的多个端口，每个用户在尝试连接服务器时，将逐端口试探，直到遇到第一个空闲端口，进行连接。客户端与服务器连接上后，所有操作将在一个独立的线程里进行，不同线程共享一份数据库，但彼此的操作互不影响。客户端断开服务器连接后，服务器的该端口被释放，继续等待下一个客户端连接。

2.4 服务器与客户端、管理员端的信息交互

服务器与客户端、管理员端的信息交互，实际上是如下图所示的关系：

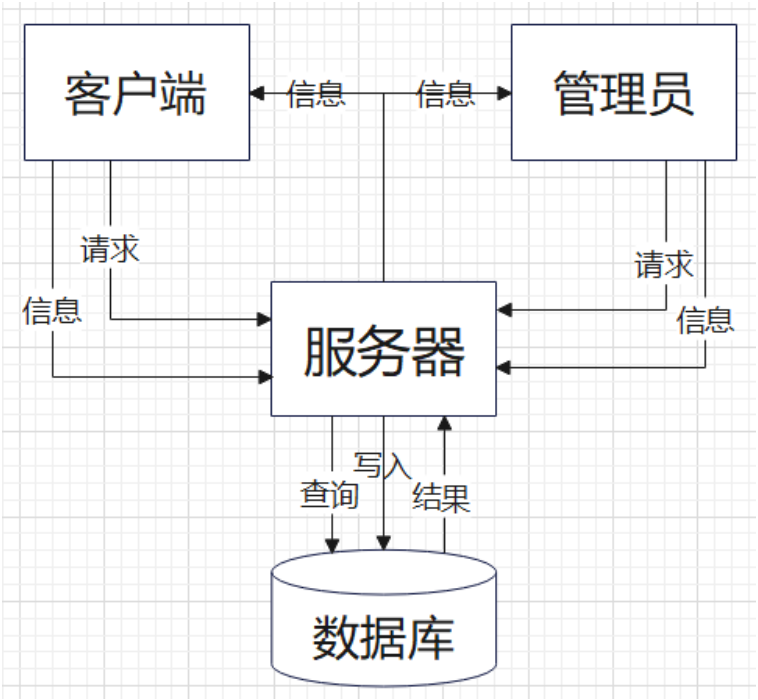

Figure 3: 信息交互模式

如图，客户端和管理员每次向服务器请求获取信息或者请求传送信息之前，都必须先往数据流里向服务器发送一个请求字符串；服务器解析这个请求字符串，来决定是准备向数据库中查询信息并发送给下游端口，还是准备接收数据并写入数据库。

本项目利用 Java Socket 网络通信技术进行信息交互。笔者从互联网的“请求-响应”机制获取灵感，规定每次客户端、管理员端与服务器进行信息交互前发的请求字符串，都以“’ 请求头’——’

参数 1'——' 参数 2'.....” 的形式，向服务器发送请求。服务器将请求字符串根据'-' 字符分隔，根据请求头的内容，做出相应的相应。

信息交互的数据多种多样，有图像、题目、章节名称等等，且它们的数目也不尽相同，必须制作出相对统一的接口，方便数据传输。为此，我们设计如下的类：

```
1 public class DataByte implements Serializable
2 {
3     private static final long serialVersionUID = 6529685098267757690L;
4     public Vector<Vector<String>> vector=new Vector<Vector<String>>();
5 }
```

该类具有一个类 ID 用以标识自定义类，同时继承 Serializable 接口，使其可以被 ObjectOutputStream 转换为字节流，同时也可以通过 ObjectInputStream 再将其解析为对象。它还包含一个 Vector<Vector<String> >，传输信息时，只需要将要传输的信息组装成二维 String 数组，再将放入 DataByte 对象，并将其转化为 byte 数组，即可进行传输。接收信息时，只需将接收的 byte 数组重新转化为 DataByte 对象，再提取其中的二维 vector 即可。

装

订

线

2.5 管理员窗体的功能与设计

管理员窗体提供了可视化操作数据库的功能。笔者在窗体设置了“章节图片、习题集、选择题、判断题、填空题、编程题”六个菜单项，点击每个菜单项后都将进入对应的界面。其中章节图片界面中提供了对知识库图片的增删查改功能。但值得注意的是，对于一个知识体系、特别是一本书而言，并不适合从任意一个位置进行删除或者增加，这样可能导致知识体系不连贯的问题。笔者采取了折衷的方案，即管理员可以对知识库进行尾插和尾删。

需要说明的是，笔者把习题库的增删查改操作分散在了五个界面，一方面是出于拆解功能、方便操作的考虑，另一方面是为了与数据库的关系模式相匹配。在现实世界里，习题之间的共性是都具备题型、题号、题干，都对应着相应的知识章节；但各种类型的题目都具备各自的共性，例如选择题有四个选项、判断题的答案只能是“T”或“F”，编程题的参考答案占据多行等等。为此在数据库的设计里，我们将其分为习题表和四个题型对应的表，分别存放题号题干等信息和参考答案等信息。与数据库对应的，管理员窗体也分为这几个界面，分别操作题干等信息和参考答案等信息。

与此同时，由于各个题型对应的关系实体外键依赖于习题这一父类主体，为此笔者只在“习题集”界面设置了删除和插入按钮，同时操作习题表和对应题型的表。在管理员窗体里，我们以表格展示查询到的题目信息，管理员可自由在表格里对非主键字段进行更改，表格将监听改动事件，并将相应信息反馈给服务器，完成数据库的更新。

3 功能的简介与实现

本节中，笔者将集中展示服务器和管理员端的功能实现，不再赘述数据库的构建相关内容。（数据准备和数据库的实现可参见上文）

3.1 数据库连接和多线程

由于笔者使用的 MySQL 为 8.0 版本，为此需要先向服务器项目导入 mysql-connector-java-8.0.16.jar 外部库，之后设置如下的常量，用以连接华为云上数据库：

```
static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
static final String DB_URL = "jdbc:mysql://localhost:3306/test?useSSL=false&
allowPublicKeyRetrieval=true&serverTimezone=UTC&useUnicode=true&characterEncoding=UTF-8";
static final String USER = "debian-sys-maint";
static final String PASS = "ONPEVtZlkY6cA8V0";
static Connection conn = null;     static Statement stmt = null;
```

其中 conn 是数据库的连接器，在构造函数里进行连接：

```
Class.forName(JDBC_DRIVER);
conn = DriverManager.getConnection(DB_URL,USER,PASS);
```

stmt 是数据库的句柄对象，每次需要操纵数据库时进行创建，用完后应该关闭。

```
GreetingServer.stmt = GreetingServer.conn.createStatement();
GreetingServer.stmt.close();
```

我们为客户端开放 20 个线程，分别对应端口 6000-6019，为管理员开放 7000 端口：

```
Thread manager = new GreetingServer(7000); manager.start();
for(int i = 6000;i<=6019;i++) {
    Thread t = new GreetingServer(i); t.start();}
```

对于每个线程，都在线程的 run() 函数里进行连接-交互-关闭的循环，除非强制关闭服务器，否则该循环永远运行下去，保证了客户端连接的连续性：

```
public void run(){
    while(true){
        server = serverSocket.accept();
        dealClient(serverIn, in,out);    //根据下游端口的请求字符串来处理事务
        server.close();
    }
}
```

3.2 服务器与客户端交互的相关功能实现

3.2.1 登录与注册功能

该功能的实现在 LoginRegister.Java 文件中。

- **登录功能**

方法名	Login
参数	name：用户名 password：密码
功能	处理客户端的登录请求
返回值	-1：查无此人 -2：数据库错误 0：密码不匹配 1：登陆成功

关键语句：

```
1  sql = "SELECT * FROM users where USERNAME = '"+name+"'";
2  String clientName = rs.getString("USERNAME");
3  String clientPassword = rs.getString("USERPASSWORD");
4  if(!password.equals(clientPassword))    return 0;    //密码不匹配
5  return 1;  //登陆成功
```

说明：根据用户名字段查询数据库，若查得密码与用户输入的密码一致，则成功登录。

- **注册功能**

方法名	Register
参数	name：用户名 password：密码
功能	处理客户端的注册请求
返回值	-1：用户名重复 -2：数据库错误 0：未成功注册 1：登陆成功

关键语句：

```
1  String sqlString = "select * from users where USERNAME = '"+name+"'";
2  if(rs.next())    return -1;  //用户名重复
3  else {
4      sqlString = "insert into users (USERNAME,USERPASSWORD)
5          values('"+name+"','"+password+"')";
6      int resultStat = GreetingServer.stmt.executeUpdate(sqlString);
7      if(resultStat>0)     return 1;    //注册成功
8  }
```

说明：根据用户名字段查询数据库，若存在相同用户名，则不允许注册；否则在数据库和输入密码合法的情况下，注册成功。

3.2.2 图像文件传输功能

该功能的实现在 FileIO.Java 文件中。

- **传输某小节所有图片功能**

方法名	sendImage
参数	chapter：章节号 subChapter：子章节号
功能	将某小节的所有图片转化为二维数组返回
返回值	imageBytes：包含了某小节所有图片的 byte 二维数组

关键语句：

```
String pathString = "/root/JavaProject/";     //图片总文件夹
sql = "select FILEPATH from chapterfiles where CHAPTER = "
    +chapter+" AND SUBCHAPTER = "+subChapter;
try (FileInputStream fileInputStream = new FileInputStream(new File(pngString));) {
    imageBytes[index] = new byte[fileInputStream.available()];
    fileInputStream.read(imageBytes[index]);     //写入图片
}
```

说明：根据章节号、子章节号，在数据库里找到对应图片的地址，打开对应图片文件并转化为 byte 数组进行返回。

- **传输章节名功能**

方法名	sendChapterNames
参数	无
功能	查询数据库，将所有章节子章节名转化为 byte 数组返回
返回值	bytes：包含了所有章节子章节名的 byte 数组

关键语句：

```
sql = "select CHAPTERNAME,SUBCHAPTERNAME from knowledge";
DataByte shaper= new DataByte();
while(rs.next())
{
    vector = new Vector<String>();
    vector.add(rs.getString(1));
    vector.add(rs.getString(2));
    shaper.vector.add(vector);
}
bytes =SendQuestions.getBytesFromObject(shaper);
```

说明：将查询结果转化为 n×2 的 vector 对象（2 代表章节名、子章节名），用一个 DataByte 对象安放并转化为 byte[] 数组。

3.2.3 某章习题传输功能

该功能的实现在 SendQuestions.Java 文件中。

- **某章习题传输功能**

方法名	sendQuestionsByChapter
参数	chapter：章节号
功能	将客户端要求的某章习题，先转化为 Serializable 对象再转化为 byte 数组，最后按题型组装为二维数组
返回值	questionsBytes：包含四种题型的 byte[4][]

关键语句：

```
String selectString = "select * from("+
"select questions.CHAPTER ,knowledge.CHAPTERNAME,questions.NUMBER,
questions.questiondescribe,selectquestions.sa,selectquestions.sb,
selectquestions.sc,selectquestions.sd,selectquestions.answer "+
"from questions ,knowledge,selectquestions "+
"where questions.CHAPTER = "+chapter+" and knowledge.CHAPTER = questions.CHAPTER
and questions.TYPE = '选择题' and selectquestions.CHAPTER = questions.CHAPTER and
 selectquestions.NUMBER = questions.NUMBER "+
"order by rand() limit 6)as tmp ORDER BY tmp.NUMBER;";
```

说明：这是查询选择题相关信息的 sql 语句，包括章节号、章节名、题号、题干、四个选项、答案九个字段。其他题型的 sql 语句与之类似，不再赘述。

- **传输综合复习习题功能**

方法名	sendReviewQuestions
参数	v：一个 11×2 的 vector 数组，包含了章节号和题型
功能	提供客户端所需的综合复习习题。按照客户端要求的章节号和题型，先转化为 Serializable 对象，再转化为 byte 数组进行返回
返回值	bytes：包含用户综合复习所需的若干题目信息

关键语句：与上一功能类似，不再赘述。

说明：客户端传的参数为不定长的二维 Vector，只需遍历该 Vector，逐题检索客户端要求的某章节某题型题目，同样组装成二维 Vector 转化为 byte[] 传输即可。

装

订

线

3.2.4 答题记录交互功能

该功能的实现在 AnswersIO.Java 文件中。客户端做完习题后，需要向服务器反馈做题情况；客户端进行推荐复习和综合复习推荐题目时，又需要服务器向其传送各个该用户在习题集上各个章节的正确率，以此来决定推荐内容。

- **用户历史答题正确率传输功能**

方法名	sendTopicInfo
参数	userName：用户名
功能	为客户端推荐学习路径和综合复习功能提供数据， 包括用户在某一章节的近期第 1-15、16-30、31-45 和历史做过的所有题目的正确率。 共有十一章，记录进 11×4 的二维 vector 并转化为 byte 数组
返回值	bytes 一维数组：包含上述信息转化而来的 byte 数组

说明：本函数中关键语句较多，整体思路是，客户端每次做完一章习题，都会推荐复习章节，依据的是本章节的在逻辑关系上的上一节（即有知识依赖关系的上一节）的答题正确率，这里我们取近期第 1-15、16-30、31-45 道题的加权正确率之和；用户要求综合复习时，将会推荐给他曾经学过的章节的若干习题，每章分配的习题数目根据的是他在此前该章节上总体的正确率和该章节的重要性权重占比。

- **录入答题记录功能**

方法名	addAnswers
参数	info：某用户某次做题后的做题记录数组，包括用户名、题型、题号、答案等信息
功能	将上述信息录入数据库
返回值	无

关键语句：录入数据库语句，不再赘述。

3.3 数据库与管理员交互的相关功能实现

3.3.1 某张图片交互功能

该功能的实现在 FileIO.Java 文件中。

- **传输某小节含有图像总数功能**

方法名	imageNum
参数	chapter：章节号 subChapter：子章节号
功能	计算某小节含有的图片总数
返回值	len：某小节含有的图片总数

说明：之所以要计算每章含有的图片总数，是因为管理员是逐张查看图片的，必须反馈给管理员图片总数，才能确定管理员要查看哪一张图片。

- **传输某张给定图片功能**

方法名	sendSelectImage
参数	chapter 章节号，subChapter：子章节号 number：图片号
功能	将某张指定的图片转化为 byte 数组返回
返回值	imageBytes：包含了所有章节子章节名的 byte 数组

说明：与 sendImage 功能类似，不再赘述。

- **更新某张特定的图像功能**

方法名	updateImage
参数	chapter 章节号，subChapter：子章节号 number：图片号 bytes：新图像的 byte 数组
功能	更新某张特定的图像
返回值	无

关键语句：

```
1 fileOutputStream = new FileOutputStream(pngFile);
2 fileOutputStream.write(bytes);      //往图片文件里写数据
```

说明：服务器接收到管理员传来的 byte 数组，直接将其转化为图像并写入在原文件位置即可。

- **在某子章节尾插图像功能**

方法名	addImage
参数	chapter 章节号，subChapter：子章节号 number：图片号 bytes：新图像的 byte 数组
功能	向某一子章节后尾插图片
返回值	无

关键代码：与更新图像类似，不再赘述。

说明：服务器接收到管理员传来的 byte 数组，直接将其转化为图像，并在子章节图像文件夹新建一份图像文件，写入即可，其序号（命名）递增。同时在数据库里加上相应记录。

- **在某子章节尾删图像功能**

方法名	deleteImage
参数	chapter 章节号，subChapter：子章节号 number：图片号
功能	删除某子章节最后一张图像
返回值	ok1&&ok2：是否删除成功

关键代码：与更新图像类似，不再赘述。

说明：服务器在某子章节图像文件夹删除最后一个文件，同时在数据库里删去相应记录。

3.3.2 习题查询功能

本功能的实现在 Manager.Java 文件中。

- **题头题面查询功能**

方法名	questionsQuery
参数	chapter：章节名 type：题型
功能	查询某章节某题型的所有题目信息，包括章节名、题型、题号、题干描述信息
返回值	questionsBytes：一维数组，包含上述信息转化而来的 byte 数组

关键代码：

```
String sql = "select number,questiondescribe from questions
    where chapter = "+chapter+" and type = '"+type+"';";
```

说明：本功能对应着管理员界面“习题集”查询功能，只提供题号题干等信息。

- **选项答案查询功能** 以选择题的查询为例，其他三种题型的实现大体一致。

方法名	selectQuestionsQuery
参数	chapter：章节名
功能	查询某章选择题的信息，包括题号、四个选项、答案字段（题干等信息可用 questionsQuery 方法查询）
返回值	questionsBytes：一维数组，包含上述信息转化而来的 byte 数组

关键代码：

```
String sql = "select number,sa,sb,sc,sd,answer
    from selectquestions where chapter = "+chapter+";";
while(rs.next())
{
    vector = new Vector<String>();
    vector.add(rs.getString(1));
    vector.add(rs.getString(2));
    vector.add(rs.getString(3));
    vector.add(rs.getString(4));
    vector.add(rs.getString(5));
    vector.add(rs.getString(6));
    shaper.vector.add(vector);
}
```

说明：本功能对应着管理员界面“选择题”查询功能，只提供题号、选项、答案等信息。

3.3.3 习题删除功能

本功能的实现在 Manager.Java 文件中。

方法名	deleteQuestion
参数	chapter：章节名：type 题型 number：题号
功能	删除某给定的题目，在 questions 表和对应的题型表中都进行删除
返回值	result>0：是否删除成功

关键代码：

```
if(type.equals("选择题"))sheetName = "selectquestions";
if(type.equals("判断题"))sheetName = "judgequestions";
if(type.equals("填空题"))sheetName = "fillquestions";
if(type.equals("编程题"))sheetName = "programquestions";
String sql = "delete from "+sheetName+" where CHAPTER = "+chapter+
    " and TYPE = '"+type+"' and NUMBER = "+number+";";
GreetingServer.stmt = GreetingServer.conn.createStatement();
GreetingServer.stmt.executeUpdate(sql);
sql = "delete from questions where CHAPTER = "+chapter+
    " and TYPE = '"+type+"' and NUMBER = "+number+";";
int result = GreetingServer.stmt.executeUpdate(sql);
System.out.println("delete"+result);
GreetingServer.stmt.close();
return result>0;
```

说明:由于在数据库中,各题型对应的习题表格 ("xxxxquestions") 外键依赖于习题集表格 ("questions")，因此进行习题删除的时候，应该现在对应题型表中删除，再在习题集总表中删除。

3.3.4 习题更新功能

本功能的实现在 Manager.Java 文件中。

- **题干信息更新功能**

方法名	updateQuestions
参数	chapter：章节名 type：题型 number：题号 describe：题干描述
功能	更新某给定的题目（题干），在 questions 表进行更新
返回值	result>0：是否更新成功

关键代码：

```
String sql = "UPDATE questions set questiondescribe = '"+describe+
    "' where CHAPTER = "+chapter+" and TYPE = '"+type+"' and NUMBER = "+number+";";
```

说明：管理员在“习题集”界面的数据表格里可以直接进行题干的改动，服务器接收到改动信息后，会在数据库里执行 update 语句。

- **选项答案更新功能** 以选择题的更新为例，其他三种题型的实现大体一致。

方法名	updateSelectQuestions
参数	chapter：章节号 number：题号 columnNum：字段号 contain：更新内容
功能	更新某章某道选择题的信息，包括四个选项、答案字段 （题干信息更新可用 updateQuestions 方法更新）
返回值	result>0：是否更新成功

关键代码：

```
1 if(columnNum.equals("1"))columnName = "sa";
2 if(columnNum.equals("2"))columnName = "sb";
3 if(columnNum.equals("3"))columnName = "sc";
4 if(columnNum.equals("4"))columnName = "sd";
5 if(columnNum.equals("5"))columnName = "answer";
6 String sql = String.format("UPDATE selectquestions SET %s = '%s'
7     where chapter = %s and number=%s;",columnName,contain,chapter,number);
```

说明:本功能对应着管理员界面“选择题”更新功能,只可更新四个选项和答案信息。column-Num 指示服务器更新哪一个字段，由管理员界面表格上的监听器给出。

3.3.5 习题添加功能

本功能的实现在 Manager.Java 文件中。

方法名	addQuestions
参数	sql1：在某题型表中的插入 sql 语句 sql2：在 questions 表中的插入 sql 语句
功能	插入一道新题目，在对应的题型表和 questions 表中同时插入
返回值	result>0：是否插入成功

关键代码：

```
1 int ok1,ok2;
2 try {
3     ok1 = GreetingServer.stmt.executeUpdate(sql1);
4     ok2 = GreetingServer.stmt.executeUpdate(sql2);
5 } catch (Exception e) {
6     return false;
7 }
8 return (ok1>0)&&(ok2>0);
```

说明：为了编程方便，本功能的 sql 语句由服务器在确认添加习题时给出，同样由于主外键依赖原因，先在某题型表中进行插入，再在 questions 总表中插入。

3.4 管理员界面的功能实现举例

管理员界面实现了对知识库图片的增删查改、对习题库的增删查改等操作，功能较为繁杂，实现方法之间互有重叠，笔者将在 doc/项目说明书里结合图示进行展示。在本章中，笔者仅简要介绍几个重点功能。

3.4.1 图像的 I/O 和显示

本功能在 FilePanel.java 中实现。在 FilePanel 界面中，设置了如下控件：

```
JComboBox<String>chapterBox;          //章节名选择框
JComboBox<String>subChapterBox;       //子章节名选择框
JComboBox<Integer>imageNumBox;        //图片序号选择框
JButton queryButton;                  //查询按钮
JButton updateButton;                 //更新按钮
JButton addButton;                    //尾插按钮
JButton deleteButton;                 //尾删按钮
JScrollPane scrollPane;               //滚轮面板
```

每次 chapterBox 的内容发生变化时，都会向服务器查询对应的子章节名集合；每次 subChapterBox 的内容发生变化时，都会向服务器查询对应的图片编号集合。管理员可在三个选择框中选择要查询的图片，并点击 queryButton 向服务器发送请求，获得 byte 数组后转化为图像，然后在 scrollPane 显示。

点击 updateButton 时，将呼出一个文本选择框，默认只能选择 png 格式的图片。选中后，将该图片转化为字节流传给服务器。点击 addButton 时的反应与更新操作相似，但尾插成功后会刷新 imageNumBox，将新添加的图片序号加入进去。点击 deleteButton 删除成功后也会刷新 imageNumBox。

3.4.2 数据表格的设计和编辑操作

这里以“习题库”界面为例，其中含有一个表格，只有最后一列题干字段可以被编辑。swing 原生的 JTable 并不支持表格定制功能，为此我们定义 class QPModel extends AbstractTableModel 来为此界面定制需要的功能。

- 只有第三列可以被编辑, 重载 isCellEditable 方法实现

```
@Override
public boolean isCellEditable(int rowIndex, int columnIndex) {
    // 判断单元格是否可以编辑，只有第三列可以编辑
```

```
    if(columnIndex==3)return true;
    else return false;
}
```

- 为表格设计样式

```
table.getColumnModel().getColumn(0).setPreferredWidth(50);
table.getColumnModel().getColumn(1).setPreferredWidth(80);
table.getColumnModel().getColumn(2).setPreferredWidth(50);
table.getColumnModel().getColumn(3).setPreferredWidth(890-50-50-80);
scrollPane = new JScrollPane(table);
```

- 为表格添加修改事件监听器（AbstractTableModel 类提供的功能）

```
modelDemo.addTableModelListener(new TableModelListener() {
    public void tableChanged(TableModelEvent e) {
        int chapter = (int)chapterBox.getSelectedItem();
        String type = (String) typeBox.getSelectedItem();
        String number = (String)table.getValueAt(table.getSelectedRow(), 2);
        String describe = (String)table.getValueAt(table.getSelectedRow(), 3);
        //用户更新题目描述并确定更新
        int option=JOptionPane.showConfirmDialog(null, "您确定要更新吗？");
        if(option==0) {
            try {
                out.writeUTF("ManagerQuestionsUpdate-"+chapter+"-"+
                    type+"-"+number+"-"+describe);
                System.out.println(in.readBoolean());
            } catch (IOException e1) {
            e1.printStackTrace();
            }
        }
    }
});
```

4 项目总结

行笔至此，感慨万千。这门课是我跨学科选择的个性课程，是为了拓展知识、增强本领、锤炼技能而选择的课程。略出我意料的是，这门课程的小作业和大作业都极富有挑战性，要求的不只是单一语言、单一维度的编程能力，而更是一种工程思维、项目思维和多端协同、模块化编程的能力。

自十一月中旬以来，我和我的队友在本项目上花费了大量时间、倾注了不少心血，从无到有、从零到一，咬紧牙关、连续作战，终于如期完成了本次项目。在此期间我们遇到了不少困难挑战，从没有合适的数据来源，到毫无头绪的推荐算法，从没有前端构建经验，到部署云服务器时的异常频出..... 特别是 12 月以来，疫情在全国出现爆发，个人安危被摆在了首要的位置，在这种情况下，要静下心、定下身来做项目、做研究，殊为不易、难能可贵。

不管风吹浪打，胜似闲庭信步，没有良好格式的数据，就用正则表达式和爬虫造出来；想不通推荐算法，就结合其他学习平台的模式自己设计；没有前端构建经验，就自己摸索用 Java Swing 构建窗体；云端运行和本机运行状态迥异，就一行行调试、一行行摸索。世上无难事，只要肯登攀，经过一个时期的尝试与探索，项目的所有问题终告圆满解决，项目基本获得圆满成功。

Figure 4: 不管风吹浪打，胜似闲庭信步

通过本次项目，笔者锻炼了数据处理和数据库实施的能力，拓展了 Java 在数据库和网络通信方面的技术，进一步熟悉了 Java Swing 技术设计前端的流程。特别是笔者自己设计的客户端、管理员端与服务器间的“请求-响应”模式，虽然并不符合现在互联网上前后端交互的规范，但在笔者和队友都毫无前端知识储备的情况下，这样的模式约定也是一种被实践检验可行的探索，更重要的是，它将有助于我们将来理解前后端交互的技术，理解规范的技术模式为什么好、为什么要那样做。

本次项目的顺利完成，笔者要感谢贺向东老师，贺老师有着资深的 Java 开发经验，本着让同学们多学技术、增长才干的初衷，为我们布置了这样一个综合性项目，笔者作为软件工程专业的本科生，在此项目中深受启发。

笔者要感谢助教刘威老师，助教老师细心回答同学们的问题，认真、及时批改同学们的作业，尽职尽责、令人敬佩。

笔者要感谢我的队友，他在没有数据库基础、第一次学习 Java 的情况下，肯吃苦、肯钻研，凭借艰苦奋斗的精神和严谨扎实的专业基础，顺利完成了项目的客户端模块和相关算法，有力保障了项目的顺利完成。

笔者还要感谢我的爱人，她在这段忙碌动荡的日子里一直默默陪伴着我，给了我极大的鼓励与支持，使我有了完成项目和学业的不竭动力。

最后，笔者要感谢伟大的中国人民、伟大的祖国和伟大的中国共产党，党和人民审时度势、因时制宜，使我们走向了正常生活的光辉道路，也使笔者对未来充满信心。

踔厉奋发，勇毅前行，立德立言，无问西东。