hdjd课程设计（论文）任务书

**软 件**  学 院 专 业班

一、课程设计(论文)题目

二、课程设计(论文)工作自 2021年 12月 20日起至 2021 年 12 月 31日止

三、课程设计(论文) 地点: 软件测试实验室

四、课程设计(论文)内容要求：

**1．本课程设计的目的**

（1）巩固和加深对数据库基本知识的理解，提高综合运用课程知识的能力。

（2）使学生巩固所学的理论基础知识的理解，掌握数据库设计的全过程及技

术与方法。

1. 培养学生编制软件文档及开发应用系统的能力，提高学生独立分析问题、解决问题的能力，锻炼和加强学生的动手能力。使学生掌握使用各种计算机资料和有关

参考资料。

1. 通过对多门课程的综合实践，能够运用软件工程学方法为指导，java为开发

语言，数据库系统原理设计数据库，进行数据库应用项目的设计与编程。通过综合课

程设计加强对课堂学习内容的理解，提高对理解《Java程序设计》及《数据库系统原

理》课程的联系性及多门课程的综合应用能力。

**2．课程设计的任务及要求**

（1）任务要求

① 开发语言：Java语言；

② DBMS：ms sql server、MySql、Oracle；

③ 数据库设计与实施

根据选题任务要求，收集并查询相关文献资料，明确系统需求；通过对系统的功

能分析和数据分析进行系统的需求分析设计，概念结构设计、逻辑结构设计、物理结构设计，以及数据库的实施。

④编写具有良好交互界面的数据库应用程序；

**（2）创新要求：**

在基本要求达到后，可进行创新设计，如完善的功能、友好的人机界面*。*

**（3）课程设计论文编写要求**

①要按照书稿的规格打印与写课程设计报告书；

②报告包括目录、绪论、正文、小结、参考文献、附录等；

③课程设计报告装订按学校的统一要求完成；

**3.课程设计进度安排**

**内容 天数 地点**

准备与动员 1 实验室

选题及可行性分析 2 实验室、图书馆

需求分析 1 实验室

设计实现 4 实验室

文档撰写 1 图书馆、实验室

答辩 1 实验室

学生签名：

2021 年 12 月31日

**课程设计(论文)评审意见**

（1）平时表现（20分）： 优（ ）、良（ ）、中（ ）、一般（ ）、差（ ）；

（2）设计报告（20分）： 优（ ）、良（ ）、中（ ）、一般（ ）、差（ ）；

（3）系统功能设计（20分）：优（ ）、良（ ）、中（ ）、一般（ ）、差（ ）；

（4）编码质量（30分）： 优（ ）、良（ ）、中（ ）、一般（ ）、差（ ）；

（5）答辩（10分） 优（ ）、良（ ）、中（ ）、一般（ ）、差（ ）；

（6）**任何抄袭成绩一律归零；文档格式不规范降档**

评阅人： 职称：

2022年 1月 1日

目 录

[1. 系统需求分析阶段 1](#_Toc91802404)

[1.1 引言 1](#_Toc91802405)

[1.2 目标与任务 1](#_Toc91802406)

[1.2.1 需求分析阶段的目标 1](#_Toc91802407)

[1.2.2 需求分析阶段的任务 1](#_Toc91802408)

[1.3 需求分析阶段成果 3](#_Toc91802409)

[1.3.1用例图 3](#_Toc91802410)

[1.3.2数据流图 7](#_Toc91802411)

[1.3.3数字字典 8](#_Toc91802412)

[2. 概念设计阶段 12](#_Toc91802413)

[2.1 引言 12](#_Toc91802414)

[2.2 概念模型设计 12](#_Toc91802415)

[2.2.1 实体属性图 12](#_Toc91802416)

[2.2.2局部分E-R图 13](#_Toc91802417)

[2.2.3 总E-R图 15](#_Toc91802418)

[3．逻辑设计阶段 16](#_Toc91802419)

[3.1逻辑设计的任务和目标 16](#_Toc91802420)

[3.2将E-R图转换为关系模型 16](#_Toc91802421)

[3.3模型优化 17](#_Toc91802422)

[3.4数据库模式定义 17](#_Toc91802423)

[3.5用户子模式设计 21](#_Toc91802424)

[3.6数据库的逻辑结构定义 22](#_Toc91802425)

[4．物理设计阶段 26](#_Toc91802426)

[4.1物理设计阶段的目标与任务 26](#_Toc91802427)

[4.2存取方式 26](#_Toc91802428)

[4.3存储结构 26](#_Toc91802429)

[5． 数据库实施阶段 27](#_Toc91802430)

[5.1建立数据库 27](#_Toc91802431)

[5.2 建立数据表 27](#_Toc91802432)

[5.2.1 员工基础信息表的建立 27](#_Toc91802433)

[5.2.2 广告投放表的建立 28](#_Toc91802434)

[5.2.3 广告基础信息表的建立 29](#_Toc91802435)

[5.2.4 栏目基础信息表的建立 30](#_Toc91802436)

[5.2.5 调度单审计表的建立 30](#_Toc91802437)

[5.2.6 调度单表的建立 31](#_Toc91802438)

[5.2.7 演播室基础信息表的建立 32](#_Toc91802439)

[5.3 数据入库 33](#_Toc91802440)

[6 系统详细设计与实现 35](#_Toc91802441)

[6.1 总体设计结构 35](#_Toc91802442)

[6.1.1运行环境 35](#_Toc91802443)

[6.1.2基本处理模块 35](#_Toc91802444)

[6.1.3 模块结构 36](#_Toc91802445)

[6.2 登入模块设计（用自己具体模块名称代替，下同） 36](#_Toc91802446)

[6.2.1 模块算法描述 36](#_Toc91802447)

[6.2.2 模块功能介绍及界面截图 37](#_Toc91802448)

[6.2.3 模块核心代码 37](#_Toc91802449)

[6.3 员工信息管理模块设计 40](#_Toc91802450)

[6.3.1 模块算法描述 40](#_Toc91802451)

[6.3.2 模块功能介绍及界面截图 40](#_Toc91802452)

[6.3.3 模块核心代码 43](#_Toc91802453)

[主窗口： 43](#_Toc91802454)

[DAO层接口实现类代码： 45](#_Toc91802455)

[Service层接口实现类代码： 47](#_Toc91802456)

[Model对象代码： 48](#_Toc91802457)

[6.4 演播厅信息管理模块设计 52](#_Toc91802458)

[6.4.1 模块算法描述 52](#_Toc91802459)

[6.4.2 模块功能介绍及界面截图 52](#_Toc91802460)

[6.4.3 模块核心代码 53](#_Toc91802461)

[6.5 广告信息管理模块设计 61](#_Toc91802462)

[6.5.1 模块算法描述 61](#_Toc91802463)

[6.5.2 模块功能介绍及界面截图 62](#_Toc91802464)

[6.5.3 模块核心代码 62](#_Toc91802465)

[6.6 调度单信息管理模块设计 70](#_Toc91802466)

[6.6.1 模块算法描述 70](#_Toc91802467)

[6.6.2 模块功能介绍及界面截图 71](#_Toc91802468)

[6.6.3 模块核心代码 71](#_Toc91802469)

[6.7 栏目信息管理模块设计 79](#_Toc91802470)

[6.7.1 模块算法描述 79](#_Toc91802471)

[6.7.2 模块功能介绍及界面截图 80](#_Toc91802472)

[6.7.3 模块核心代码 80](#_Toc91802473)

[7 总结 89](#_Toc91802474)

[附录1 数据库JDBC连接接口与DAO层部署代码 89](#_Toc91802475)

[com/LuoZhiHao/controller/DatabaseController.java： 89](#_Toc91802476)

[com/LuoZhiHao/utils/Database.java： 90](#_Toc91802477)

[8 **参考文献** 92](#_Toc91802478)

# 1. 系统需求分析阶段

## 1.1 引言

某电视台需要开发一套信息管理系统，以方便对全台的员工、栏目、广告和演播厅等进行管理。

## 1.2 目标与任务

### 1.2.1 需求分析阶段的目标

了解电视台关于全台员工、栏目、广告和演播厅等内容的现状，并且通过实地调查和问答－记录的方式了解宿舍管理的工作业务流程，并记录和处理相关的数据。

### 1.2.2 需求分析阶段的任务

（1）处理对象：

系统将要处理的对象有：员工基本信息、栏目基本信息、广告基本信息、演播厅基本信息、调度单信息、调度单审计信息，各个对象包括信息如下所示：

* 员工基本信息(User)：  
  包括员工编号、员工姓名、员工职位、员工性别、员工的出生日期、员工的电话、住址、员工登入系统的密码。
* 栏目基本信息(Column)：  
  包括栏目编号、栏目名称、播出时间和播出时长。
* 广告基本信息(Advertisement)：  
  包括广告编号、广告价格。
* 演播厅基本信息(Television\_Room)：  
  包括演播厅的房间号、房间面积大小。
* 调度单信息(Dispatch\_List)：  
  包括调度单号、被协调的栏目编号、演播厅房间号、对应的员工编号。
* 调度单审计信息(Dispatch\_List\_Audit)：  
  包括调度单审计编号、调度单编号、调度单审计状态、审计人员编号。

（2）处理功能要求

系统主要完成的功能有以下几点：

1. 员工基本信息的增加、删除、修改和查询。
2. 栏目基本信息的增加、删除、修改和查询。
3. 广告基本信息的增加、删除、修改和查询。
4. 演播厅基本信息的增加、删除、修改和查询。
5. 调度单信息的增加、删除、修改和查询。
6. 调度单的通过审批和记录。

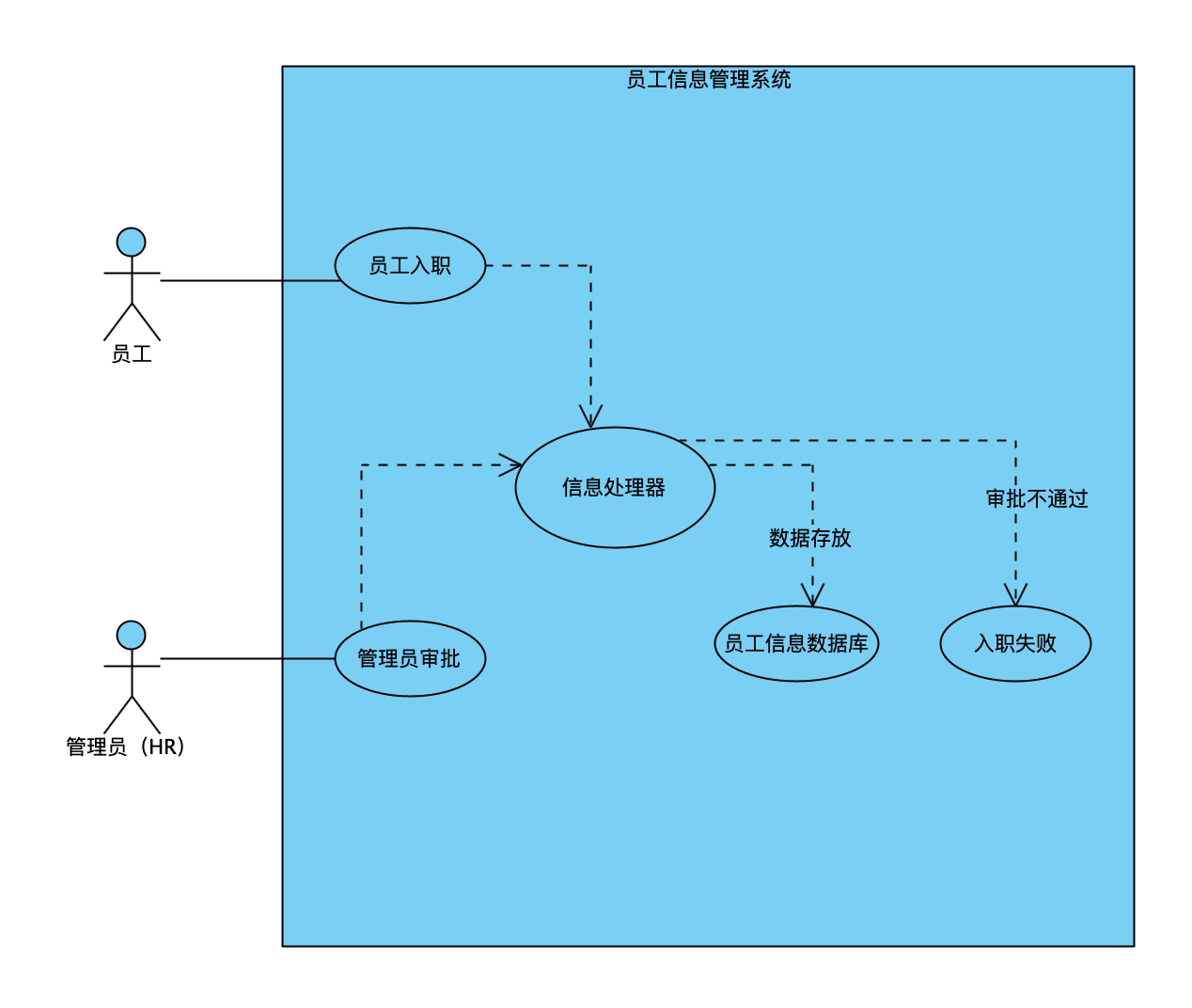
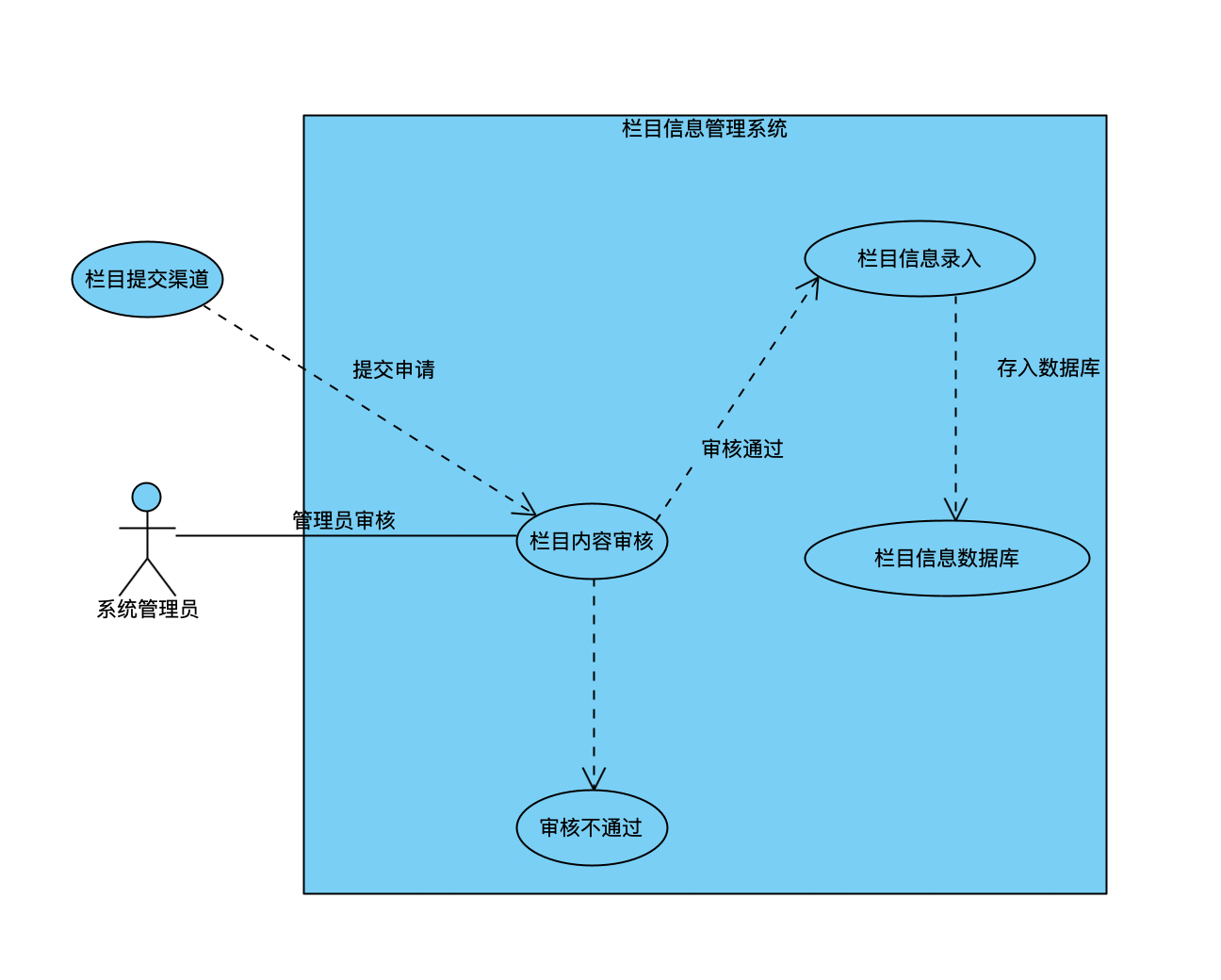
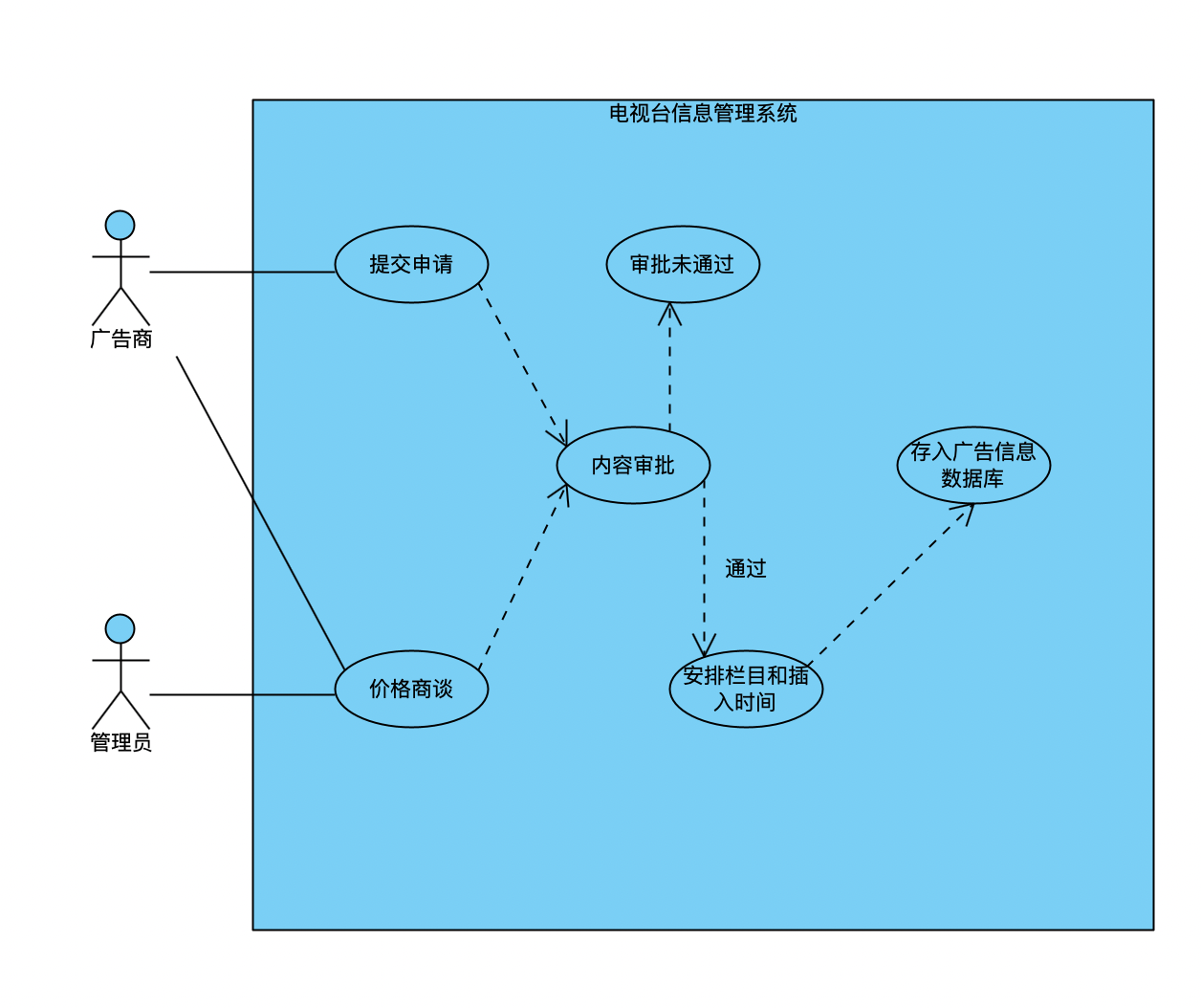
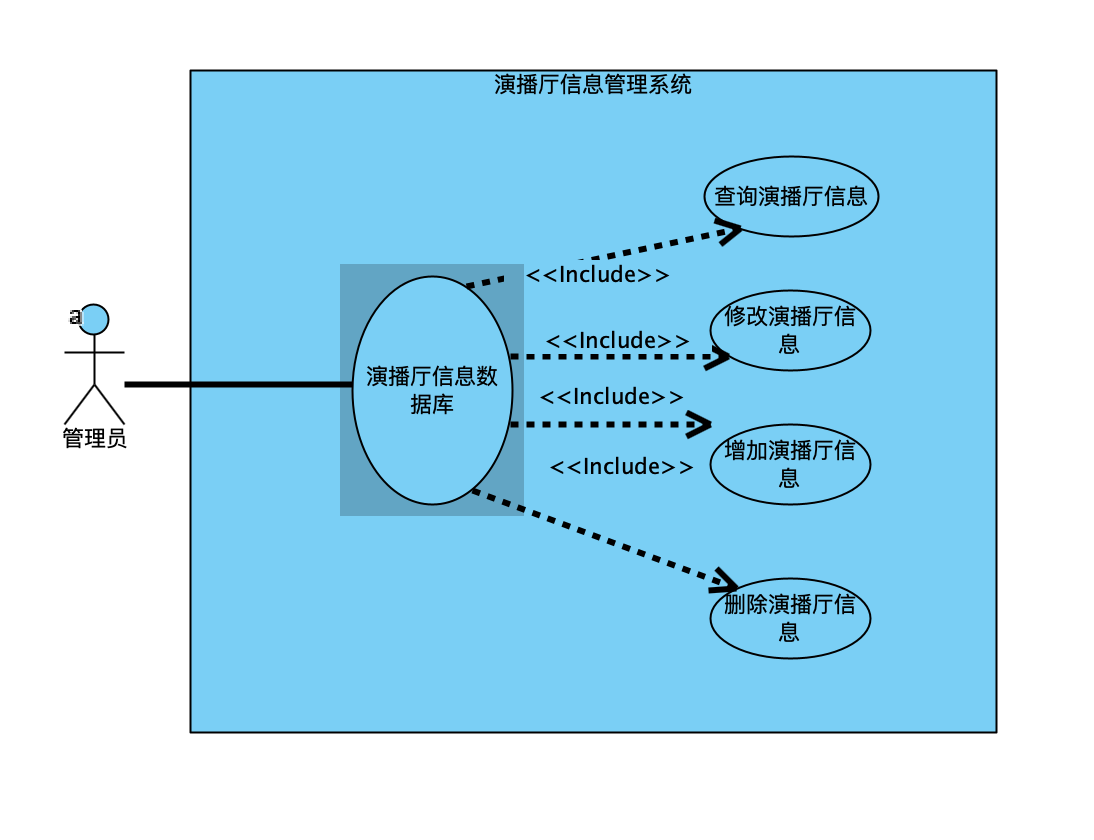
（3）安全性和完整性要求

安全性先通过视图机制，不同的用户只能访问系统授权的视图，这样可提供系统数据一定程度上的安全性，再通过用户授权机制，欲用户登陆来识别用户级别，根据这个级别来分配用户权限，达到数据更高层次的安全保密功能。

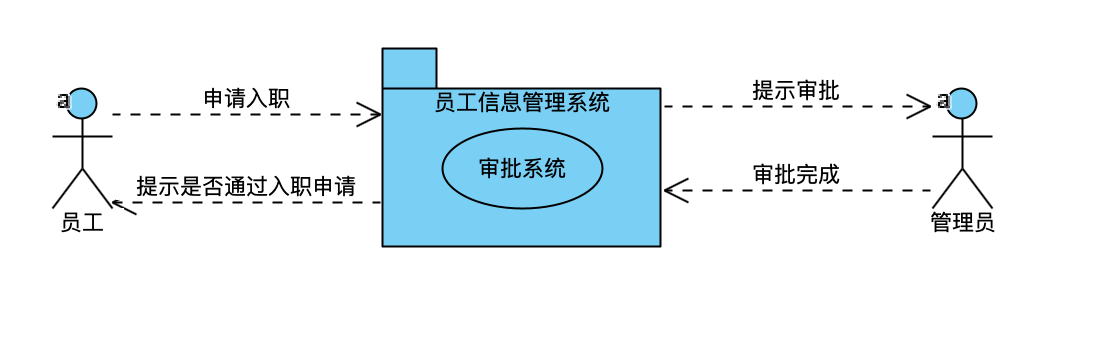
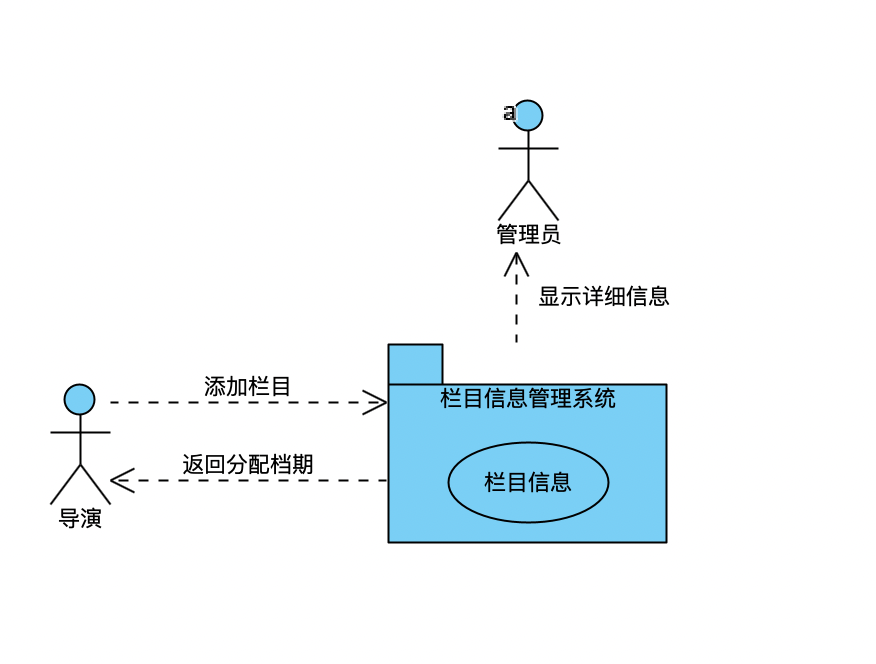
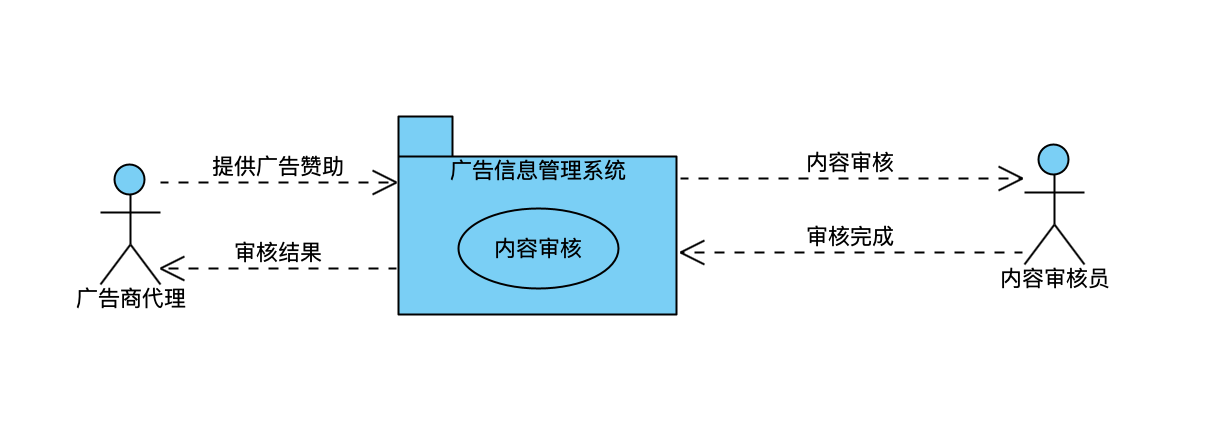
完整性要求用于描述员工基本信息、栏目基本信息、广告基本信息、演播厅基本信息、调度单信息、调度单审计信息中数据项能否为null，以及一些用户自定义完整性（符合实际要求），详细完整性要求见于系统的逻辑设计阶段。

## 1.3 需求分析阶段成果

### 1.3.1用例图

1. 员工入职审批流程图
2. 栏目管理用例图：
3. 广告入驻基本用例图：  
   
4. 演播厅基本信息处理用例图：  
   
5. 调度单处理用例图：  
   

### 1.3.2数据流图

1. 员工入职审批数据流程图：  
   
2. 栏目管理用例图：  
   
3. 广告入驻基本用例图：  
   

### 1.3.3数字字典

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段备注** |
| tv | user | uid | int unsigned | NULL | 否 | 员工编号 |
| tv | user | uname | varchar(10) | NULL | 否 | 员工姓名 |
| tv | user | upassword | varchar(32) | NULL | 否 | 员工密码 |
| tv | user | uposition | varchar(12) | NULL | 否 | 员工职位 |
| tv | user | uaddress | varchar(100) | NULL | 是 | 员工住址 |
| tv | user | ubirthday | datetime | NULL | 是 | 员工生日 |
| tv | user | ugender | varchar(2) | NULL | 是 | 员工性别 |
| tv | user | utel | varchar(15) | NULL | 是 | 员工电话 |
| tv | tv\_room | tvrid | int unsigned | NULL | 否 | 演播厅房间号 |
| tv | tv\_room | tvrsize | double(10,2) | NULL | 是 | 演播厅大小 |
| tv | dispatch\_list | cid | int unsigned | NULL | 否 | 被协调栏目编号 |
| tv | dispatch\_list | did | int unsigned | NULL | 否 | 调度单号 |
| tv | dispatch\_list | tvrid | int unsigned | NULL | 否 | 演播厅房间号 |
| tv | dispatch\_list | uid | int unsigned | NULL | 否 | 员工编号 |
| tv | dispatch\_audit | dlaid | int unsigned | NULL | 否 | 调度单审计编号 |
| tv | dispatch\_audit | dlid | int unsigned | NULL | 否 | 调度单编号 |
| tv | dispatch\_audit | dlstatus | varchar(10) | 未审批 | 否 | 调度单状态 |
| tv | dispatch\_audit | dluid | int unsigned | NULL | 否 | 审计人员名称 |
| tv | column | cduration | int | NULL | 否 | 播出时长(秒) |
| tv | column | cid | int unsigned | NULL | 否 | 栏目编号 |
| tv | column | cname | varchar(50) | NULL | 否 | 栏目名称 |
| tv | column | ctime | datetime | NULL | 否 | 播出时间 |
| tv | advertisement | aid | int unsigned | NULL | 否 | 广告编号 |
| tv | advertisement | aprice | decimal(10,2) | NULL | 否 | 广告价格 |
| tv | ad\_dispatch | adid | int unsigned | NULL | 否 | 广告投放编号 |
| tv | ad\_dispatch | aid | int unsigned | NULL | 否 | 广告编号 |
| tv | ad\_dispatch | cid | int unsigned | NULL | 否 | 栏目编号 |

# 2. 概念设计阶段

## 2.1 引言

概念设计阶段主要是将需求分析阶段得到的用户需求抽象为信息结构(概念模型)的过程,它是整个数据库设计的关键，包括概念模型设计和新系统流程两个阶段。

## 2.2 概念模型设计

### 2.2.1 实体属性图

（论文中***图*** 的画法，注意以下几点：1 图画图尽量选用visio等画图工具，否则打印出来图的效果很不清晰。2 ***图的标题 和序号***要写在***图的下方***，***居中***，用***5号黑体字***。3 ***图中的文字***采用***5号宋体字***。4 ***每章***的所有图采用***顺序编号***，如图2-1 ，图2-2等）

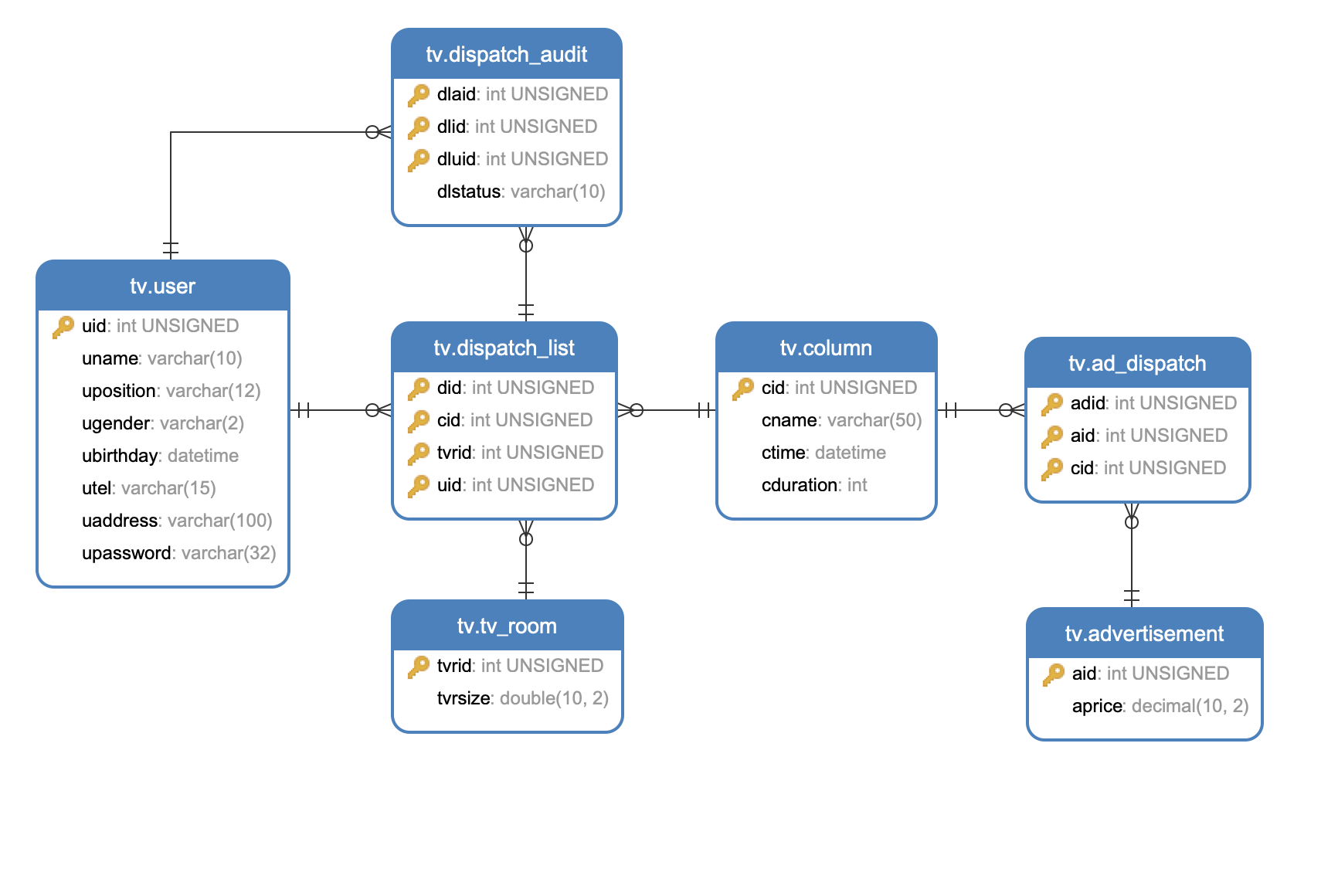
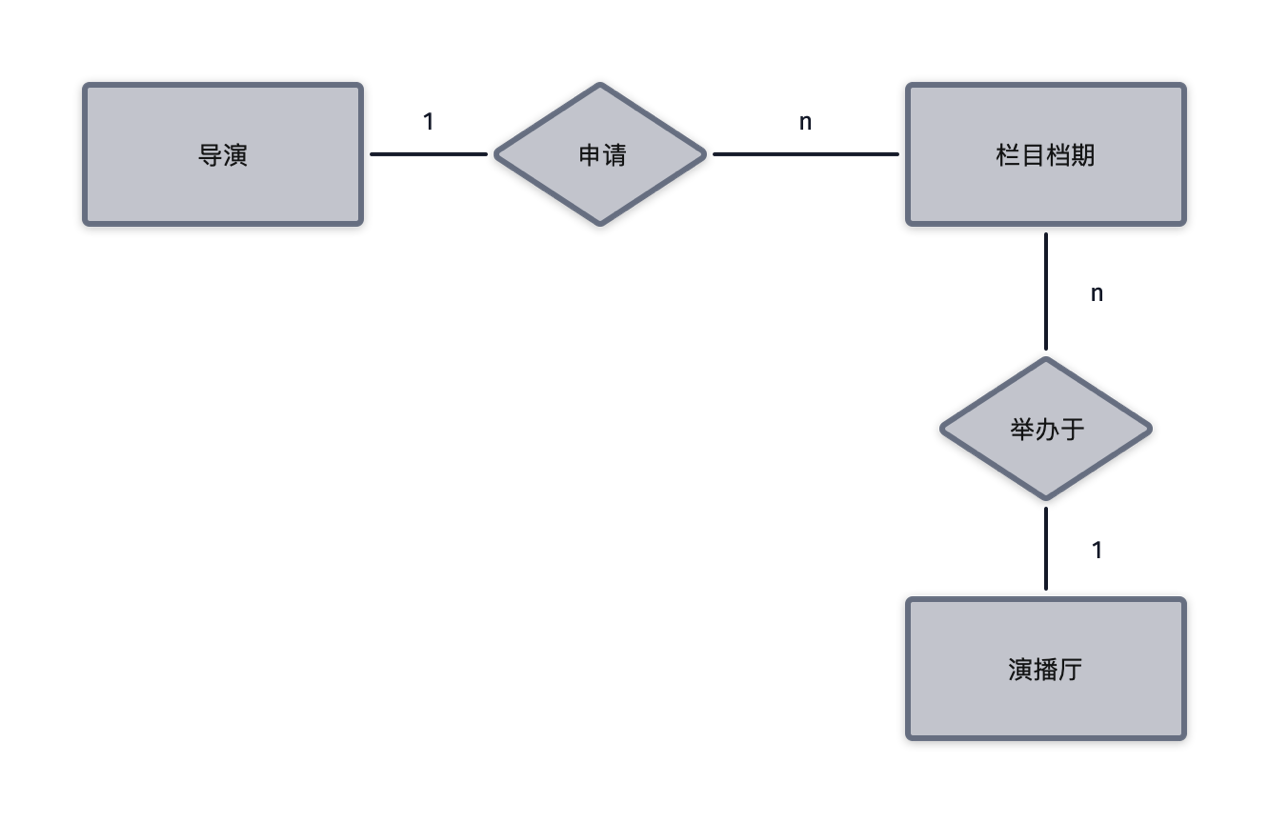
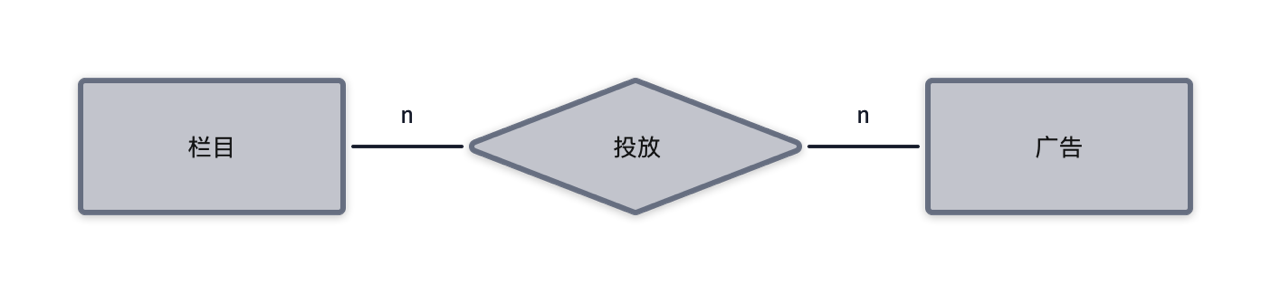
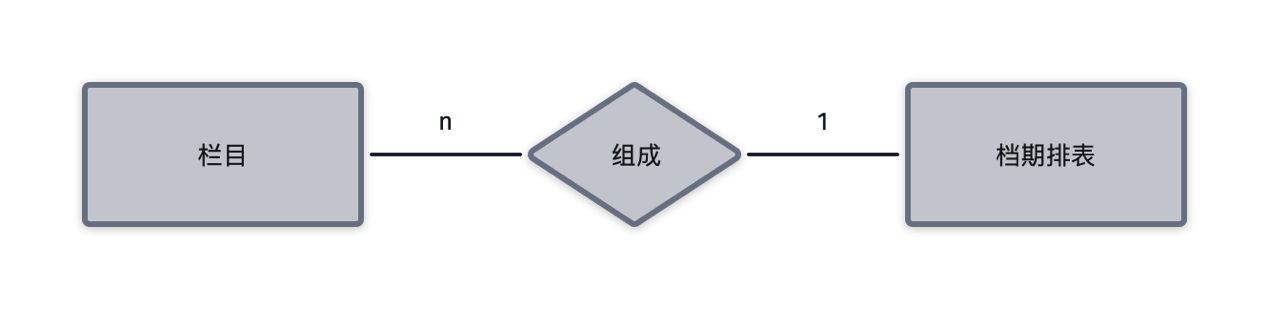


图2-1 实体联系图

### 2.2.2局部分E-R图

1. 导演申请档期E-R图  
   
2. 广告投放E-R图  
   
3. 栏目档期E-R图  
   

### 2.2.3 总E-R图

图2\_2 系统总E-R图

# 3．逻辑设计阶段

## 3.1逻辑设计的任务和目标

以上的概念设计阶段是独立于任何一种数据模型的，但是逻辑设计阶段就与选用的DBMS产品发生关系了，系统逻辑设计的任务就是将概念设计阶段设计好的基本E-R图转换为选用DBMS产品所支持的数据模型相符合的逻辑结构。具体内容包括数据组织（将E-R图转换成关系模型、模型优化、数据库模式定义、用户子模式设计）、数据处理（画出系统功能模块图）两大任务

## 3.2将E-R图转换为关系模型

将总E-R图中的1对1、1对多、多对多关系分别转化为关系模型，如下所示：

员工：user(uid, uname, uposition, ugender, ubirthday, utel, uaddress, upassword);

演播厅：tv\_room(tvrid, tvrsize);

栏目：column(cid, cname, ctime, cduration);

调度单：dispatch\_list(did, cid, tvrid, uid);

调度单审计表：dispatch\_audit(dlaid, dlid, dluid, dlstatus);

广告投递表： ad\_dispatch(adid, aid, cid);

广告：advertisement(aid, aprice);

## 3.3模型优化

关系模型中存在部分冗余，但能提高查询效率和使用效率，故不用继续优化。

## 3.4数据库模式定义

（论文中***表*** 的画法， ***表的标题 和序号***要写在***表的下方***，***居中***，用***5号黑体***。3 ***图中的文字***采用***5号宋体字***。4 ***每章***的所有表采用***顺序编号***，如图2-1 ，图2-2等）

**表3\_1 数据库模式定义表**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | 逻辑结构(基本表)定义 | | | | | | 完整性和安全性 | | |
| T-1 | ad\_dispatch | | | | | | 详见表3-2 | | |
| T-2 | advertisement | | | | | | 详见表3-3 | | |
| T-3 | column | | | | | | 详见表3-4 | | |
| T-4 | dispatch\_audit | | | | | | 详见表3-5 | | |
| T-5 | dispatch\_list | | | | | | 详见表3-6 | | |
| T-6 | tv\_room | | | | | | 详见表3-7 | | |
| T-7 | user | | | | | | 详见表3-8 | | |
| **数据库名** | | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | | **字段说明** |
| tv | | ad\_dispatch | adid | int unsigned | NULL | 否 | | 广告投放编号 |
| tv | | ad\_dispatch | aid | int unsigned | NULL | 否 | | 广告编号 |
| tv | | ad\_dispatch | cid | int unsigned | NULL | 否 | | 栏目编号 |

**表3-2** ad\_dispatch**定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | advertisement | aid | int unsigned | NULL | 否 | 广告编号 |
| tv | advertisement | aprice | decimal(10,2) | NULL | 否 | 广告价格 |

**表3-3** advertisement**定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | column | cduration | int | NULL | 否 | 播出时长(秒) |
| tv | column | cid | int unsigned | NULL | 否 | 栏目编号 |
| tv | column | cname | varchar(50) | NULL | 否 | 栏目名称 |
| tv | column | ctime | datetime | NULL | 否 | 播出时间 |

**表3-4** column**定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | dispatch\_audit | dlaid | int unsigned | NULL | 否 | 调度单审计编号 |
| tv | dispatch\_audit | dlid | int unsigned | NULL | 否 | 调度单编号 |
| tv | dispatch\_audit | dlstatus | varchar(10) | 未审批 | 否 | 调度单状态 |
| tv | dispatch\_audit | dluid | int unsigned | NULL | 否 | 审计人员名称 |

**表3-5** dispatch\_audit**定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | dispatch\_list | cid | int unsigned | NULL | 否 | 被协调栏目编号 |
| tv | dispatch\_list | did | int unsigned | NULL | 否 | 调度单号 |
| tv | dispatch\_list | tvrid | int unsigned | NULL | 否 | 演播厅房间号 |
| tv | dispatch\_list | uid | int unsigned | NULL | 否 | 员工编号 |

**表3-6** dispatch\_list**定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | tv\_room | tvrid | int unsigned | NULL | 否 | 演播厅房间号 |
| tv | tv\_room | tvrsize | double(10,2) | NULL | 是 | 演播厅大小 |

**表3-7** tv\_room**定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | user | uid | int unsigned | NULL | 否 | 员工编号 |
| tv | user | uname | varchar(10) | NULL | 否 | 员工姓名 |
| tv | user | upassword | varchar(32) | NULL | 否 | 员工密码 |
| tv | user | uposition | varchar(12) | NULL | 否 | 员工职位 |
| tv | user | uaddress | varchar(100) | NULL | 是 | 员工住址 |
| tv | user | ubirthday | datetime | NULL | 是 | 员工生日 |
| tv | user | ugender | varchar(2) | NULL | 是 | 员工性别 |
| tv | user | utel | varchar(15) | NULL | 是 | 员工电话 |

**表3-8** user**定义表**

## 3.5用户子模式设计

表3-9 用户子模式设计(View)列表

| 编号 | 用户子模式(View) | 作用(共性：提供数据保密和安全保护机制) |
| --- | --- | --- |
| V – 1 | ad\_dispatchView | 便于查询和修改广告投放信息 |
| V – 2 | advertisementView | 便于查询和修改广告信息 |
| V – 3 | columnView | 便于查询和修改栏目信息 |
| V – 4 | dispatch\_auditView | 便于查询和修改调度审核信息 |
| V – 5 | dispatch\_listView | 便于查询和修改调度单审核 |
| V – 6 | tv\_roomView | 便于查询和修改演播厅信息 |
| V – 7 | userView | 便于查询和修改员工好基本信息 |

## 3.6数据库的逻辑结构定义

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | ad\_dispatch | adid | int unsigned | NULL | 否 | 广告投放编号 |
| tv | ad\_dispatch | aid | int unsigned | NULL | 否 | 广告编号 |
| tv | ad\_dispatch | cid | int unsigned | NULL | 否 | 栏目编号 |

**表3-10** ad\_dispatch**逻辑结构定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | advertisement | aid | int unsigned | NULL | 否 | 广告编号 |
| tv | advertisement | aprice | decimal(10,2) | NULL | 否 | 广告价格 |

**表3-11** advertisement**逻辑结构定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | column | cduration | int | NULL | 否 | 播出时长(秒) |
| tv | column | cid | int unsigned | NULL | 否 | 栏目编号 |
| tv | column | cname | varchar(50) | NULL | 否 | 栏目名称 |
| tv | column | ctime | datetime | NULL | 否 | 播出时间 |

**表3-12** column**逻辑结构定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | dispatch\_audit | dlaid | int unsigned | NULL | 否 | 调度单审计编号 |
| tv | dispatch\_audit | dlid | int unsigned | NULL | 否 | 调度单编号 |
| tv | dispatch\_audit | dlstatus | varchar(10) | 未审批 | 否 | 调度单状态 |
| tv | dispatch\_audit | dluid | int unsigned | NULL | 否 | 审计人员名称 |

**表3-13** dispatch\_audit**逻辑结构定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | dispatch\_list | cid | int unsigned | NULL | 否 | 被协调栏目编号 |
| tv | dispatch\_list | did | int unsigned | NULL | 否 | 调度单号 |
| tv | dispatch\_list | tvrid | int unsigned | NULL | 否 | 演播厅房间号 |
| tv | dispatch\_list | uid | int unsigned | NULL | 否 | 员工编号 |

**表3-14** dispatch\_list**逻辑结构定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | tv\_room | tvrid | int unsigned | NULL | 否 | 演播厅房间号 |
| tv | tv\_room | tvrsize | double(10,2) | NULL | 是 | 演播厅大小 |

**表3-15** tv\_room**逻辑结构定义表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据库名** | **表名** | **字段名** | **数据类型** | **默认值** | **是否允许为空** | **字段说明** |
| tv | user | uid | int unsigned | NULL | 否 | 员工编号 |
| tv | user | uname | varchar(10) | NULL | 否 | 员工姓名 |
| tv | user | upassword | varchar(32) | NULL | 否 | 员工密码 |
| tv | user | uposition | varchar(12) | NULL | 否 | 员工职位 |
| tv | user | uaddress | varchar(100) | NULL | 是 | 员工住址 |
| tv | user | ubirthday | datetime | NULL | 是 | 员工生日 |
| tv | user | ugender | varchar(2) | NULL | 是 | 员工性别 |
| tv | user | utel | varchar(15) | NULL | 是 | 员工电话 |

**表3-16** user**逻辑结构定义表**

# 4．物理设计阶段

## 4.1物理设计阶段的目标与任务

数据库的物理设计就是为逻辑数据模型选取一个最合适应用要求的物理结构的过程，在这个阶段中要完成两大任务：

（1）确定数据库的物理结构，在关系数据库中主要是存取方法和存储结构；

（2）对物理结构进行评价，评价的重点是时间和空间效率。

## 4.2存取方式

Hash索引存取方法

## 4.3存储结构

根据系统应用情况将数据分为易变部分和稳定部分，将两部分分开存放。将员工基础信息表存放在某一磁盘中，其数据较为稳定。将栏目、演播厅、广告信息等存放在另一性能较好的磁盘中方便读取。

# 数据库实施阶段

## 5.1建立数据库

**create database** tv;  
**SET NAMES** utf8mb4;  
**SET** FOREIGN\_KEY\_CHECKS = 0;

## 5.2 建立数据表

### 5.2.1 员工基础信息表的建立

-- ----------------------------

-- Table structure for user

-- ----------------------------

DROP TABLE IF EXISTS `user`;

CREATE TABLE `user` (

`uid` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '员工编号',

`uname` varchar(10) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '员工姓名',

`uposition` varchar(12) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '员工职位',

`ugender` varchar(2) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '员工性别',

`ubirthday` datetime DEFAULT NULL COMMENT '员工生日',

`utel` varchar(15) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '员工电话',

`uaddress` varchar(100) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '员工住址',

`upassword` varchar(32) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '员工密码',

PRIMARY KEY (`uid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

### 5.2.2 广告投放表的建立

-- ----------------------------

-- Table structure for ad\_dispatch

-- ----------------------------

DROP TABLE IF EXISTS `ad\_dispatch`;

CREATE TABLE `ad\_dispatch` (

`adid` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '广告投放编号',

`aid` int unsigned NOT NULL COMMENT '广告编号',

`cid` int unsigned NOT NULL COMMENT '栏目编号',

PRIMARY KEY (`adid`,`aid`,`cid`),

KEY `aid1` (`aid`),

KEY `cid1` (`cid`),

CONSTRAINT `aid1` FOREIGN KEY (`aid`) REFERENCES `advertisement` (`aid`),

CONSTRAINT `cid1` FOREIGN KEY (`cid`) REFERENCES `column` (`cid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

### 5.2.3 广告基础信息表的建立

-- ----------------------------

-- Table structure for advertisement

-- ----------------------------

DROP TABLE IF EXISTS `advertisement`;

CREATE TABLE `advertisement` (

`aid` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '广告编号',

`aprice` decimal(10,2) NOT NULL COMMENT '广告价格',

PRIMARY KEY (`aid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

### 5.2.4 栏目基础信息表的建立

-- ----------------------------

-- Table structure for column

-- ----------------------------

DROP TABLE IF EXISTS `column`;

CREATE TABLE `column` (

`cid` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '栏目编号',

`cname` varchar(50) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '栏目名称',

`ctime` datetime NOT NULL COMMENT '播出时间',

`cduration` int NOT NULL COMMENT '播出时长(秒)',

PRIMARY KEY (`cid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

### 5.2.5 调度单审计表的建立

-- ----------------------------

-- Table structure for dispatch\_audit

-- ----------------------------

DROP TABLE IF EXISTS `dispatch\_audit`;

CREATE TABLE `dispatch\_audit` (

`dlaid` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '调度单审计编号',

`dlid` int unsigned NOT NULL COMMENT '调度单编号',

`dluid` int unsigned NOT NULL COMMENT '审计人员名称',

`dlstatus` varchar(10) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL DEFAULT '未审批' COMMENT '调度单状态',

PRIMARY KEY (`dlaid`,`dlid`,`dluid`) USING BTREE,

KEY `dlid` (`dlid`),

KEY `dluid` (`dluid`),

CONSTRAINT `dispatch\_audit\_ibfk\_1` FOREIGN KEY (`dlid`) REFERENCES `dispatch\_list` (`did`),

CONSTRAINT `dispatch\_audit\_ibfk\_2` FOREIGN KEY (`dluid`) REFERENCES `user` (`uid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

### 5.2.6 调度单表的建立

-- ----------------------------

-- Table structure for dispatch\_list

-- ----------------------------

DROP TABLE IF EXISTS `dispatch\_list`;

CREATE TABLE `dispatch\_list` (

`did` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '调度单号',

`cid` int unsigned NOT NULL COMMENT '被协调栏目编号',

`tvrid` int unsigned NOT NULL COMMENT '演播厅房间号',

`uid` int unsigned NOT NULL COMMENT '员工编号',

PRIMARY KEY (`did`,`cid`,`tvrid`,`uid`),

KEY `cid` (`cid`),

KEY `tvrid` (`tvrid`),

KEY `uid` (`uid`),

KEY `did` (`did`),

CONSTRAINT `cid` FOREIGN KEY (`cid`) REFERENCES `column` (`cid`),

CONSTRAINT `tvrid` FOREIGN KEY (`tvrid`) REFERENCES `tv\_room` (`tvrid`),

CONSTRAINT `uid` FOREIGN KEY (`uid`) REFERENCES `user` (`uid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

### 5.2.7 演播室基础信息表的建立

-- ----------------------------

-- Table structure for tv\_room

-- ----------------------------

DROP TABLE IF EXISTS `tv\_room`;

CREATE TABLE `tv\_room` (

`tvrid` int unsigned NOT NULL COMMENT '演播厅房间号',

`tvrsize` double(10,2) DEFAULT NULL COMMENT '演播厅大小',

PRIMARY KEY (`tvrid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8\_unicode\_ci;

## 5.3 数据入库

INSERT INTO `user` (`uid`, `uname`, `uposition`, `ugender`, `ubirthday`, `utel`, `uaddress`, `upassword`) VALUES (1, 'Alan', '普通员工', '男', '2000-02-21 00:00:00', '15370129382', '江西省南昌市', '123');

INSERT INTO `user` (`uid`, `uname`, `uposition`, `ugender`, `ubirthday`, `utel`, `uaddress`, `upassword`) VALUES (2, 'Bob', '面试官', '男', '2000-01-01 00:00:00', '15920201948', '江西省萍乡市', '123');

INSERT INTO `user` (`uid`, `uname`, `uposition`, `ugender`, `ubirthday`, `utel`, `uaddress`, `upassword`) VALUES (3, 'Lisa', '首席执行官', '女', '2001-03-18 00:00:00', '15770779257', '江西省赣州市', '123');

INSERT INTO `tv\_room` (`tvrid`, `tvrsize`) VALUES (100, 150.00);

INSERT INTO `tv\_room` (`tvrid`, `tvrsize`) VALUES (101, 160.50);

INSERT INTO `tv\_room` (`tvrid`, `tvrsize`) VALUES (103, 100.60);

INSERT INTO `column` (`cid`, `cname`, `ctime`, `cduration`) VALUES (300, '非诚勿扰', '2021-12-30 21:08:26', 3600);

INSERT INTO `column` (`cid`, `cname`, `ctime`, `cduration`) VALUES (301, '非常6+1', '2021-12-23 21:09:03', 4800);

INSERT INTO `column` (`cid`, `cname`, `ctime`, `cduration`) VALUES (302, '家庭幽默录像', '2021-12-07 21:09:26', 3600);

INSERT INTO `advertisement` (`aid`, `aprice`) VALUES (200, 800.00);

INSERT INTO `advertisement` (`aid`, `aprice`) VALUES (201, 400.50);

INSERT INTO `advertisement` (`aid`, `aprice`) VALUES (202, 500.00);

INSERT INTO `ad\_dispatch` (`adid`, `aid`, `cid`) VALUES (400, 200, 302);

INSERT INTO `ad\_dispatch` (`adid`, `aid`, `cid`) VALUES (401, 201, 300);

INSERT INTO `ad\_dispatch` (`adid`, `aid`, `cid`) VALUES (402, 202, 301);

INSERT INTO `dispatch\_list` (`did`, `cid`, `tvrid`, `uid`) VALUES (1, 300, 101, 1);

INSERT INTO `dispatch\_list` (`did`, `cid`, `tvrid`, `uid`) VALUES (2, 301, 103, 3);

INSERT INTO `dispatch\_list` (`did`, `cid`, `tvrid`, `uid`) VALUES (3, 302, 100, 2);

INSERT INTO `dispatch\_audit` (`dlaid`, `dlid`, `dluid`, `dlstatus`) VALUES (500, 1, 1, '未审批');

INSERT INTO `dispatch\_audit` (`dlaid`, `dlid`, `dluid`, `dlstatus`) VALUES (501, 2, 1, '未审批');

INSERT INTO `dispatch\_audit` (`dlaid`, `dlid`, `dluid`, `dlstatus`) VALUES (502, 3, 2, '已通过');

INSERT INTO `dispatch\_audit` (`dlaid`, `dlid`, `dluid`, `dlstatus`) VALUES (503, 3, 1, '未通过');

6 系统详细设计与实现

（系统设计详细设计部分，如果是多人合作的，每个人按照模块划分，一下只写自己完成的部分。）

### 6.1 总体设计结构

### 6.1.1运行环境

CPU: Intel core I7-8700

频率: 3.2 GHz

内存(RAM): 16.0 GB

操作系统: Windows 10

数据库: Mysql 8.0.21

开发工具包: JDK 1.8 version 1.8.0\_271

开发平台: IntelliJ IDEA 2020.2.3 x64

### 6.1.2基本处理模块

* 员工信息处理模块
* 栏目信息处理模块
* 演播厅信息处理模块
* 广告处理模块
* 调度单处理模块

### 6.1.3 模块结构

（用一览表及框图的形式说明本系统的系统元素（各层模块、子程序、公用程序等）的划分，扼要说明每个系统元素的[标识符](http://baike.baidu.com/view/390932.htm)和功能，分层次地给出各元素之间的控制与被控制关系。）

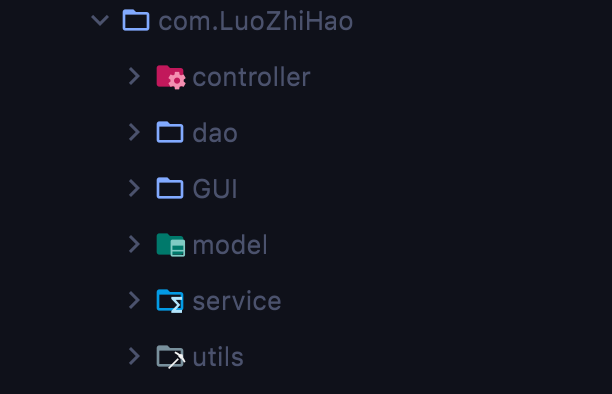


图6-1 项目模块结构

分别为：

1. 控制层（Controller）
2. 数据库连接层（DAO）
3. 图形界面层（GUI）
4. 对象模型定义（Model）
5. 服务层（Service）
6. 核心控制层（Utils）

## 6.2 登入模块设计（用自己具体模块名称代替，下同）

### 6.2.1 模块算法描述

通过Java Swing创建一个登入窗口 JFrame，定义面板 JPanel，在登入窗口中布置登入提示信息和信息输入栏，以及登入按钮和退出按钮，其中将用户输入的数据通过Swing组件获取，再从Servcice层和Dao层进行判断用户输入的账号密码和数据库中存放的账号密码进行对比，如果对比成功则返回登入成功，否则失败。

### 6.2.2 模块功能介绍及界面截图

运行界面如下：



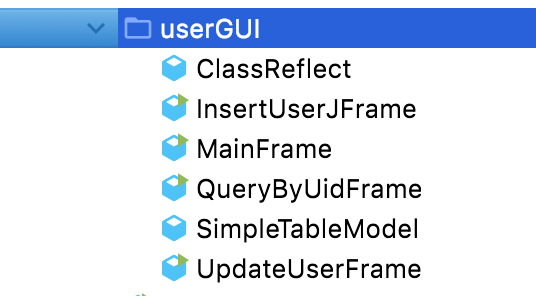
### 6.2.3 模块核心代码

实现代码如下：

**private** JPanel **contentPane**;  
**private** JTextField **userTel**;  
**private** JPasswordField **userPassword**;  
  
**public** Login(){  
 setTitle(**"电视台信息管理系统"**);  
 setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
 setBounds(200,200,450,250);  
 CardLayout cardLayout = **new** CardLayout();  
  
 **contentPane** = **new** JPanel();  
 **contentPane**.setBorder(**new** EmptyBorder(5,5,5,5));  
 setContentPane(**contentPane**);  
 **contentPane**.setLayout(cardLayout);  
  
 JPanel userPanel = **new** JPanel();  
 **contentPane**.add(userPanel);  
 userPanel.setLayout(**null**);  
  
 **userTel** = **new** JTextField();  
 **userTel**.setBounds(148,55,122,21);  
 userPanel.add(**userTel**);  
 **userTel**.setColumns(10);  
  
 **userPassword** = **new** JPasswordField();  
 **userPassword**.setBounds(148,96,122,21);  
 userPanel.add(**userPassword**);  
  
 JButton userButton1 =**new** JButton(**"登入"**);  
 userButton1.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 userLoginActionPerformed(e);  
 }  
 });  
 userButton1.setBounds(72,159,93,23);  
 userPanel.add(userButton1);  
  
 JButton userButton2 =**new** JButton(**"退出系统"**);  
 userButton2.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 System.***out***.println(**"成功退出系统"**);  
 setVisible(**false**);  
 dispose();  
 }  
 });  
 userButton2.setBounds(220,159,93,23);  
 userPanel.add(userButton2);  
  
  
 JLabel label = **new** JLabel(**"用户名："**);  
 label.setBounds(72,58,54,15);  
 userPanel.add(label);  
  
 JLabel label1 = **new** JLabel(**"密码："**);  
 label1.setBounds(72,99,54,15);  
 userPanel.add(label1);  
  
}  
  
**private void** userLoginActionPerformed(ActionEvent e) {  
 String uTel = **userTel**.getText();  
 String uPassword = **userPassword**.getText();  
 UserServiceImpl userService = **new** UserServiceImpl();  
 **try** {  
 **if** (userService.certifyUser(uTel,uPassword)){  
 JOptionPane.*showMessageDialog*(**this**,**"登入成功！\n欢迎用户："** + **userTel**.getText() +  
 **" 登入系统。"**,**"登入成功"**,JOptionPane.***INFORMATION\_MESSAGE***);  
 MainFrame mainFrame = **new** MainFrame();  
 mainFrame.setBounds(600,400,800,600);  
 mainFrame.setVisible(**true**);  
 System.***out***.println(**"用户"**+uTel +**"登入成功"**);  
 System.***out***.println(**"登入成功，正在关闭登入页面。"**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
 }**else**{  
 JOptionPane.*showMessageDialog*(**this**,**"登入失败，请检查手机号或密码是否正确！"**,**"登入失败"**,JOptionPane.***ERROR\_MESSAGE***);  
 System.***out***.println(**"登入失败请重新登入。"**);  
 }  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(**this**,**"登入失败，系统内部错误！"**,**"登入失败"**,JOptionPane.***ERROR\_MESSAGE***);  
 }  
}

## 6.3 员工信息管理模块设计

### 6.3.1 模块算法描述



该模块中先在GUI的user GUI包中创建主窗口，以用来显示从数据库读取出来的员工的详细信息，展现在面板中，并且提供滚轮滑块，以便阅读更多行数据。

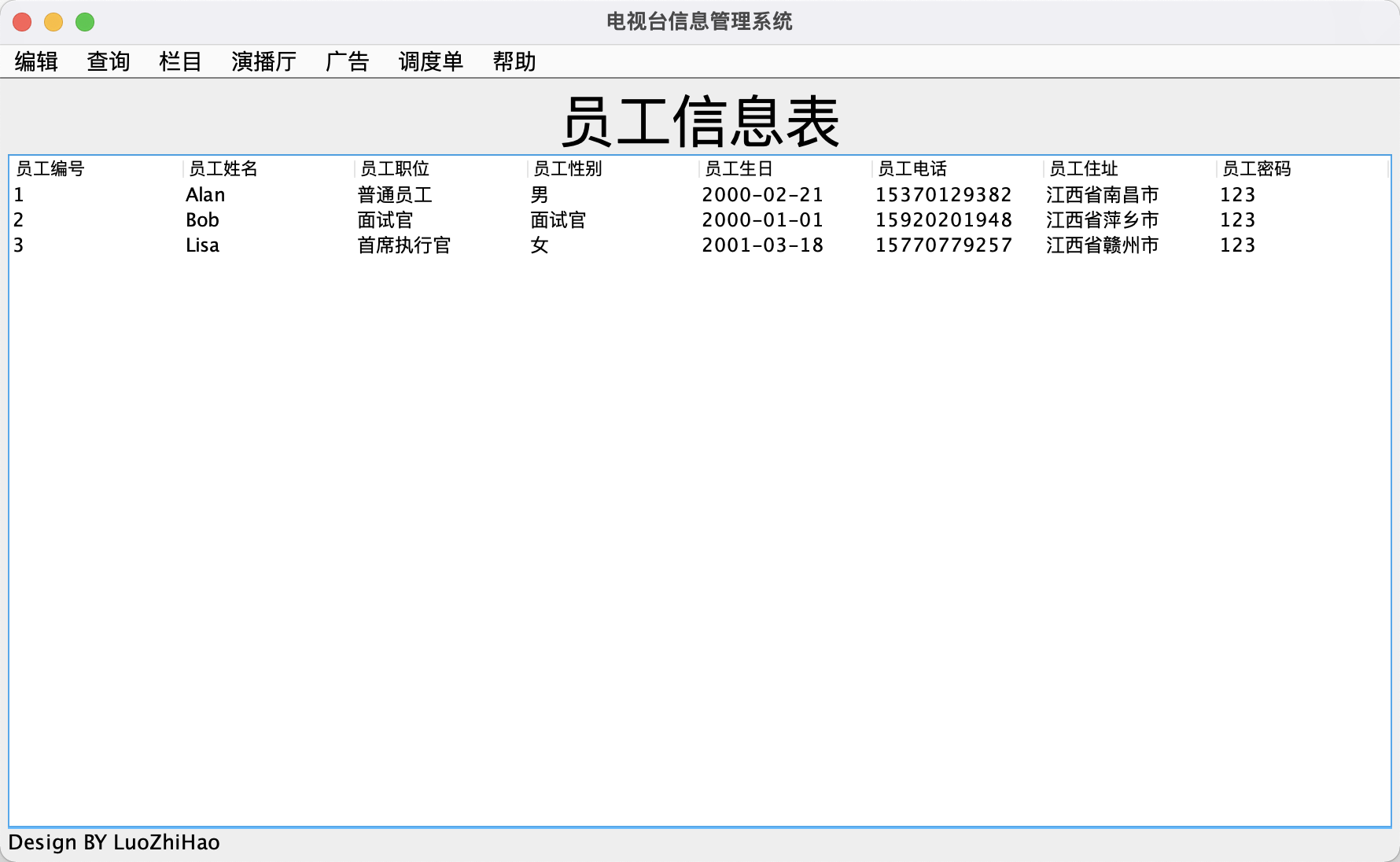
在主窗口添加菜单栏 JMenuBar来添加按钮以提供增删改查（CRUD）的操作。

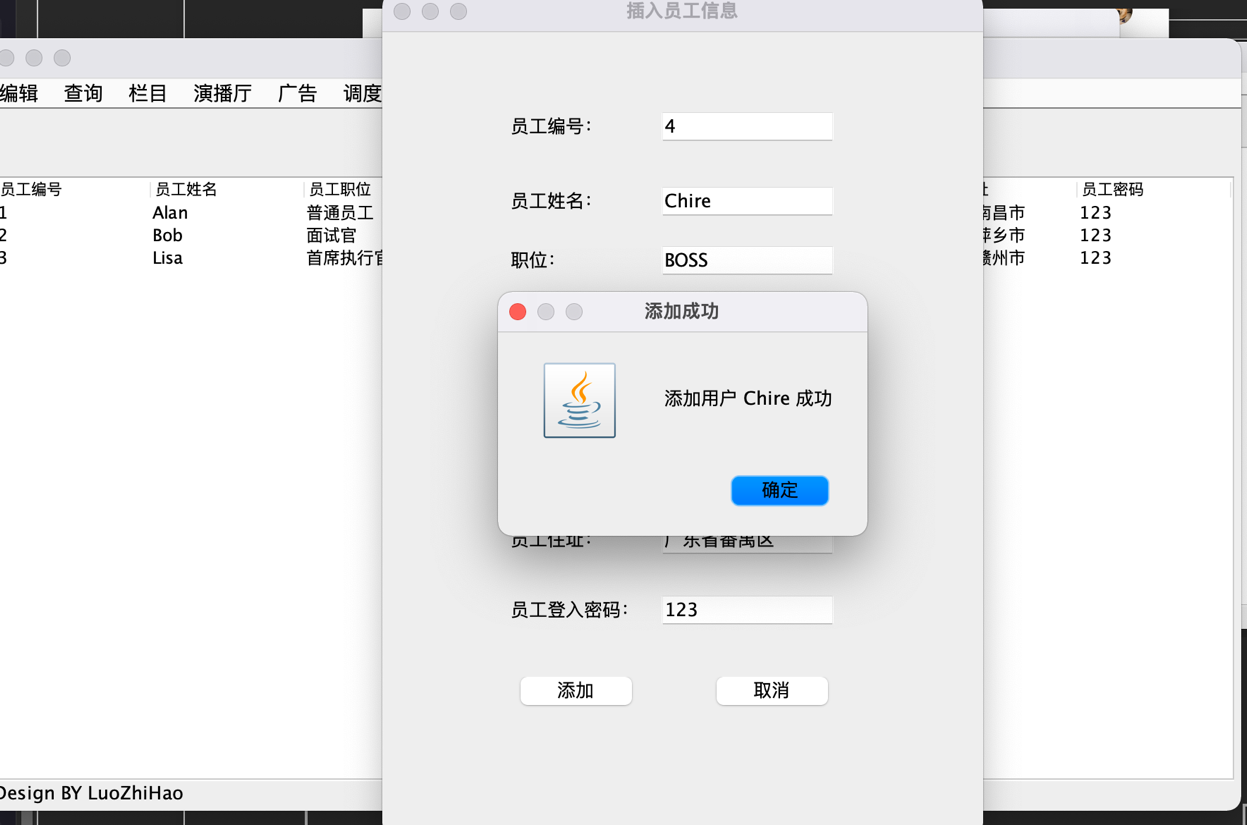
在user GUI包中分别创建其他的操作窗口，提供可视化的增删改查。

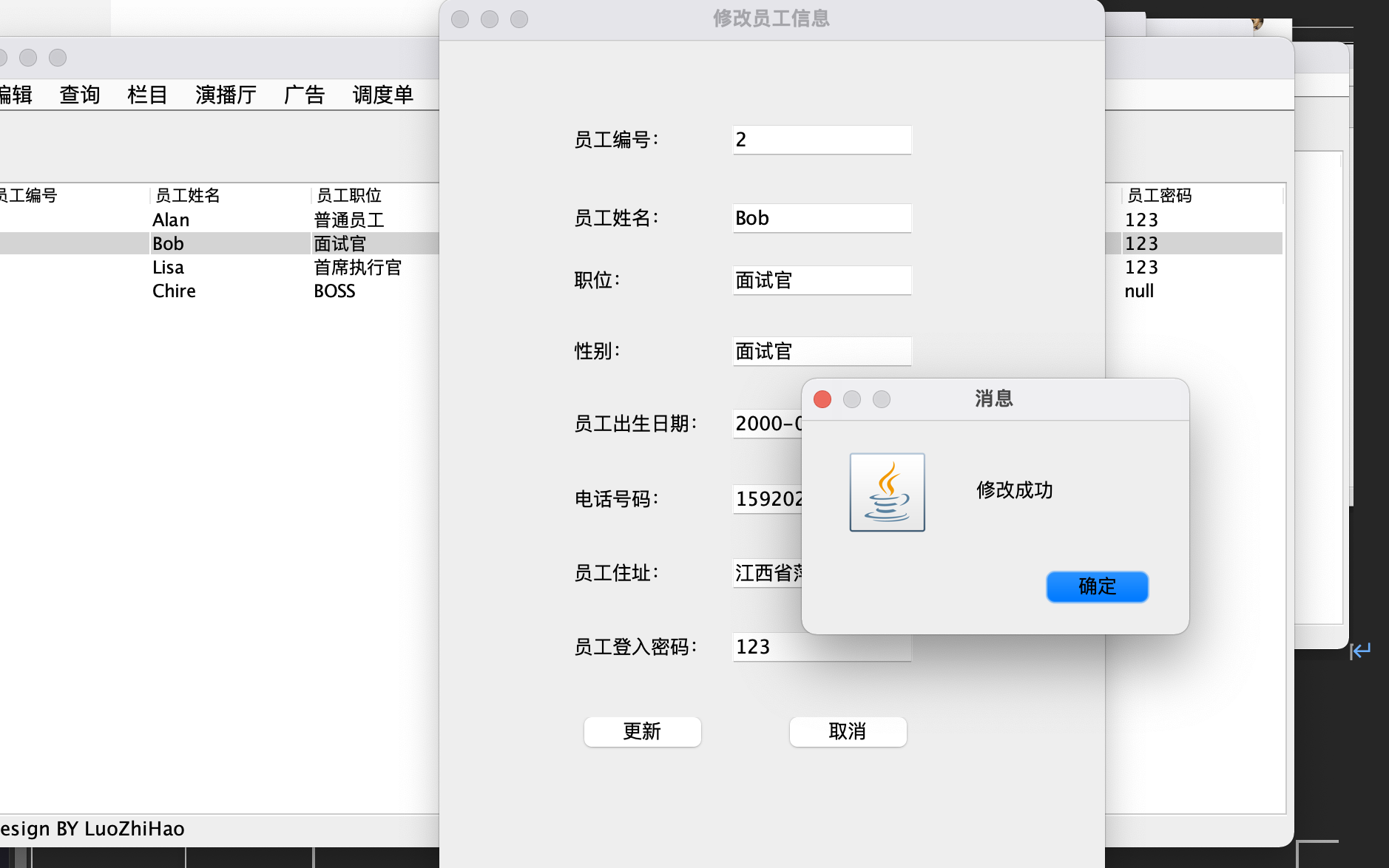
其中的数据来源于DAO层中的数据库持久层。

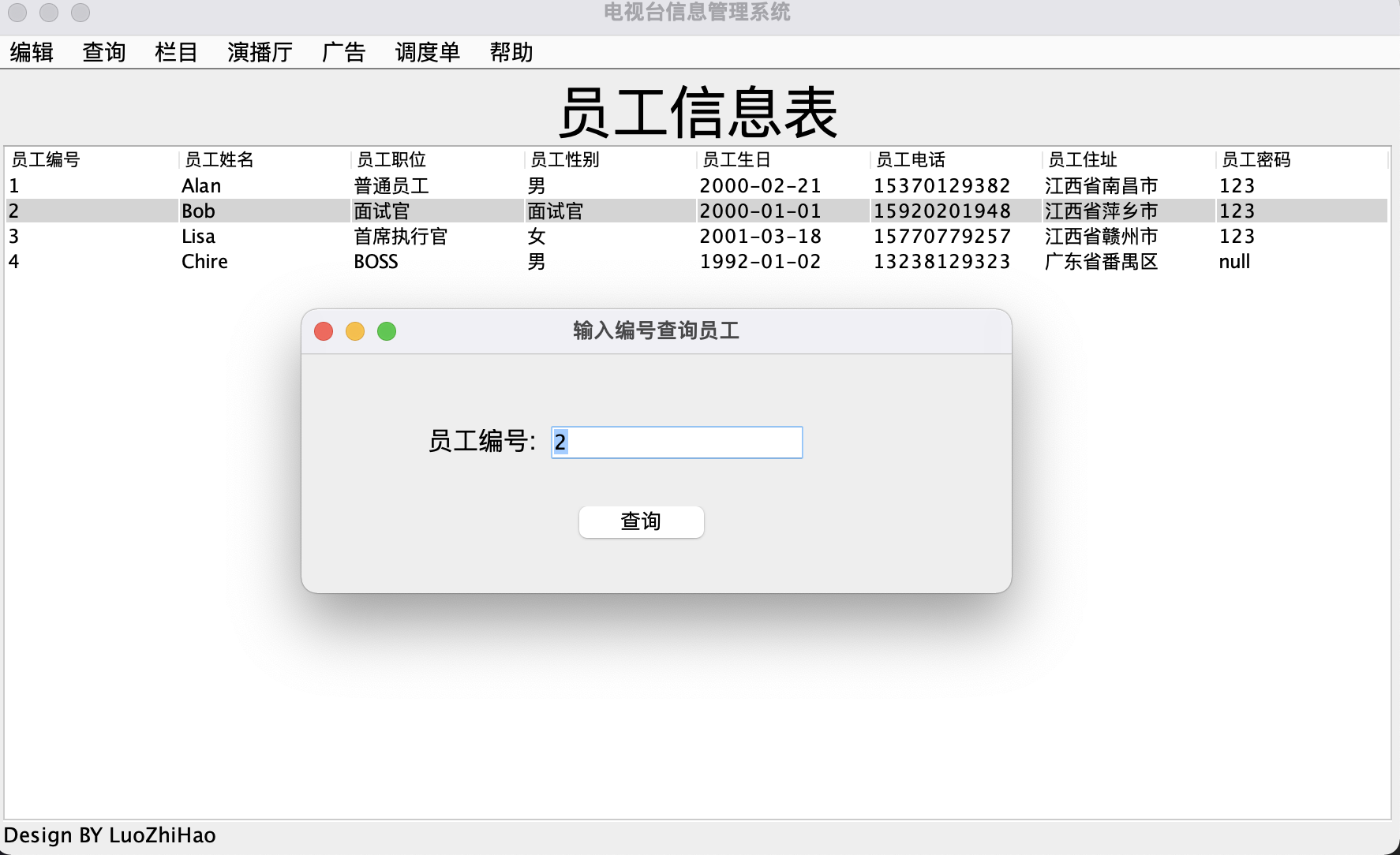
### 6.3.2 模块功能介绍及界面截图

运行结果截图：









### 6.3.3 模块核心代码

#### 主窗口：

**private** JPanel **contentPane**;  
**private** JTextField **userTel**;  
**private** JPasswordField **userPassword**;  
  
**public** Login(){  
 setTitle(**"电视台信息管理系统"**);  
 setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
 setBounds(200,200,450,250);  
 CardLayout cardLayout = **new** CardLayout();  
  
 **contentPane** = **new** JPanel();  
 **contentPane**.setBorder(**new** EmptyBorder(5,5,5,5));  
 setContentPane(**contentPane**);  
 **contentPane**.setLayout(cardLayout);  
  
 JPanel userPanel = **new** JPanel();  
 **contentPane**.add(userPanel);  
 userPanel.setLayout(**null**);  
  
 **userTel** = **new** JTextField();  
 **userTel**.setBounds(148,55,122,21);  
 userPanel.add(**userTel**);  
 **userTel**.setColumns(10);  
  
 **userPassword** = **new** JPasswordField();  
 **userPassword**.setBounds(148,96,122,21);  
 userPanel.add(**userPassword**);  
  
 JButton userButton1 =**new** JButton(**"登入"**);  
 userButton1.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 userLoginActionPerformed(e);  
 }  
 });  
 userButton1.setBounds(72,159,93,23);  
 userPanel.add(userButton1);  
  
 JButton userButton2 =**new** JButton(**"退出系统"**);  
 userButton2.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 System.***out***.println(**"成功退出系统"**);  
 setVisible(**false**);  
 dispose();  
 }  
 });  
 userButton2.setBounds(220,159,93,23);  
 userPanel.add(userButton2);  
  
  
 JLabel label = **new** JLabel(**"用户名："**);  
 label.setBounds(72,58,54,15);  
 userPanel.add(label);  
  
 JLabel label1 = **new** JLabel(**"密码："**);  
 label1.setBounds(72,99,54,15);  
 userPanel.add(label1);  
  
}  
  
**private void** userLoginActionPerformed(ActionEvent e) {  
 String uTel = **userTel**.getText();  
 String uPassword = **userPassword**.getText();  
 UserServiceImpl userService = **new** UserServiceImpl();  
 **try** {  
 **if** (userService.certifyUser(uTel,uPassword)){  
 JOptionPane.*showMessageDialog*(**this**,**"登入成功！\n欢迎用户："** + **userTel**.getText() +  
 **" 登入系统。"**,**"登入成功"**,JOptionPane.***INFORMATION\_MESSAGE***);  
 MainFrame mainFrame = **new** MainFrame();  
 mainFrame.setBounds(600,400,800,600);  
 mainFrame.setVisible(**true**);  
 System.***out***.println(**"用户"**+uTel +**"登入成功"**);  
 System.***out***.println(**"登入成功，正在关闭登入页面。"**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
 }**else**{  
 JOptionPane.*showMessageDialog*(**this**,**"登入失败，请检查手机号或密码是否正确！"**,**"登入失败"**,JOptionPane.***ERROR\_MESSAGE***);  
 System.***out***.println(**"登入失败请重新登入。"**);  
 }  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(**this**,**"登入失败，系统内部错误！"**,**"登入失败"**,JOptionPane.***ERROR\_MESSAGE***);  
 }  
}

#### DAO层接口实现类代码：

SimpleDateFormat **simpleDateFormat** = **new** SimpleDateFormat(**"yyyy-MM-dd"**);  
 @Override  
 **public** List<User> getAllUser() **throws** Exception {  
 List<User> users = **new** ArrayList<>();  
 String sql = **"SELECT \* FROM `user`"**;  
 ResultSet resultSet = Database.*query*(sql);  
 **while** (resultSet.next()){  
 User user = **new** User();  
 user.setuId(resultSet.getInt(**"uid"**));  
 user.setuName(resultSet.getString(**"uname"**));  
 user.setuPosition(resultSet.getString(**"uposition"**));  
 user.setuGender(resultSet.getString(**"ugender"**));  
 Date date = **new** Date(resultSet.getDate(**"ubirthday"**).getTime());  
 user.setuBirthday(**simpleDateFormat**.format(date));  
 user.setuTel(resultSet.getString(**"utel"**));  
 user.setuAddress(resultSet.getString(**"uaddress"**));  
 user.setuPassword(resultSet.getString(**"upassword"**));  
 System.***out***.println(user);  
 users.add(user);  
 }  
 Database.*releaseAll*();  
*// users.add(new User(1,"Alan","HR","男","2000-02-21","15770779257","江西省南昌市","123"));  
// users.add(new User(2,"Bob","HR","男","2000-02-22","15770779258","江西省南昌市","123"));* **return** users;  
 }  
  
 @Override  
 **public boolean** delUserById(Integer uId) **throws** Exception {  
 String sql = **"DELETE FROM `user` where uid = "**+ uId +**" ;"**;  
 **return** Database.*update*(sql) == 0?**false**:**true**;  
 }  
  
 @Override  
 **public boolean** addUser(User user) **throws** Exception {  
 String sql = **"INSERT INTO `user` (`uid`, `uname`, `uposition`, `ugender`, `ubirthday`, `utel`, `uaddress`, `upassword`) "** +  
 **"VALUES ("** + user.getuId() +  
 **", '"** +user.getuName() +  
 **"', '"** +user.getuPosition() +  
 **"', '"** + user.getuGender() +  
 **"', '"** + **new** java.sql.Date(**simpleDateFormat**.parse(user.getuBirthday()).getTime()) +  
 **"', '"** + user.getuTel() +  
 **"', '"** + user.getuAddress() +  
 **"', '"** + user.getuPassword() +  
 **"');"** ;  
 **int** sign = Database.*update*(sql);  
 Database.*releaseAll*();  
 **return** sign != 0;  
 }  
  
 @Override  
 **public boolean** updateUser(User user) **throws** Exception {  
 String sql = **"UPDATE `user` SET `uid` = '"** + user.getuId() +  
 **"', `uname` = '"** + user.getuName() +  
 **"', `uposition` = '"** + user.getuPosition() +  
 **"', `ugender` = '"** + user.getuGender() +  
 **"', `ubirthday` = '"** + **new** java.sql.Date(**simpleDateFormat**.parse(user.getuBirthday()).getTime()) +  
 **"', `utel` = '"** + user.getuTel() +  
 **"', `uaddress` = '"** + user.getuAddress() +  
 **"', `upassword`= '"** + user.getuPassword() +  
 **"' WHERE `uid` = "** + user.getuId() +  
 **" ;"**;  
 **int** sign = Database.*update*(sql);  
 Database.*releaseAll*();  
 **return** sign != 0;  
 }

#### Service层接口实现类代码：

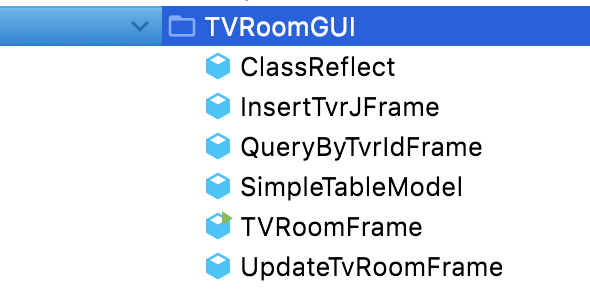
UserDaoImpl userDao = new UserDaoImpl();  
  
@Override  
public boolean certifyUser(String uTel, String uPassword) throws Exception {  
  
 return true;  
}  
  
@Override  
public List<User> getAllUser() throws Exception {  
 return userDao.getAllUser();  
  
}  
  
@Override  
public boolean delUserById(Integer uId) throws Exception {  
 return userDao.delUserById(uId);  
}  
  
@Override  
public boolean addUser(User user) throws Exception {  
 return userDao.addUser(user);  
}  
  
@Override  
public boolean updateUser(User user) throws Exception {  
 return userDao.updateUser(user);  
}

#### Model对象代码：

**public class** User {  
 *//员工编号* **private** Integer **uId**;  
 *//员工姓名* **private** String **uName**;  
 *//员工职位* **private** String **uPosition**;  
 *//员工性别* **private** String **uGender**;  
 *//员工生日* **private** String **uBirthday**;  
 *//员工电话* **private** String **uTel**;  
 *//员工住址* **private** String **uAddress**;  
 *//员工的登入密码* **private** String **uPassword**;  
  
 **public** User() {  
 }  
  
 **public** User(Integer uid, String uname, String uposition, String ugender, String ubirthday, String utel, String uaddress, String upassword) {  
 **this**.**uId** = uid;  
 **this**.**uName** = uname;  
 **this**.**uPosition** = uposition;  
 **this**.**uGender** = ugender;  
 **this**.**uBirthday** = ubirthday;  
 **this**.**uTel** = utel;  
 **this**.**uAddress** = uaddress;  
 **this**.**uPassword** = upassword;  
 }  
  
 **public** Integer getuId() {  
 **return uId**;  
 }  
  
 **public void** setuId(Integer uId) {  
 **this**.**uId** = uId;  
 }  
  
 **public** String getuName() {  
 **return uName**;  
 }  
  
 **public void** setuName(String uName) {  
 **this**.**uName** = uName;  
 }  
  
 **public** String getuPosition() {  
 **return uPosition**;  
 }  
  
 **public void** setuPosition(String uPosition) {  
 **this**.**uPosition** = uPosition;  
 }  
  
 **public** String getuGender() {  
 **return uGender**;  
 }  
  
 **public void** setuGender(String uGender) {  
 **this**.**uGender** = uGender;  
 }  
  
 **public** String getuBirthday() {  
 **return uBirthday**;  
 }  
  
 **public void** setuBirthday(String uBirthday) {  
 **this**.**uBirthday** = uBirthday;  
 }  
  
 **public** String getuTel() {  
 **return uTel**;  
 }  
  
 **public void** setuTel(String uTel) {  
 **this**.**uTel** = uTel;  
 }  
  
 **public** String getuAddress() {  
 **return uAddress**;  
 }  
  
 **public void** setuAddress(String uAddress) {  
 **this**.**uAddress** = uAddress;  
 }  
  
 **public** String getuPassword() {  
 **return uPassword**;  
 }  
  
 **public void** setuPassword(String uPassword) {  
 **this**.**uPassword** = uPassword;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "User{"** +  
 **"uid="** + **uId** +  
 **", uname='"** + **uName** + **'\''** +  
 **", uposition='"** + **uPosition** + **'\''** +  
 **", ugender='"** + **uGender** + **'\''** +  
 **", ubirthday="** + **uBirthday** +  
 **", utel='"** + **uTel** + **'\''** +  
 **", uaddress='"** + **uAddress** + **'\''** +  
 **", upassword='"** + **uPassword** + **'\''** +  
 **'}'**;  
 }  
}

## 6.4 演播厅信息管理模块设计

### 6.4.1 模块算法描述

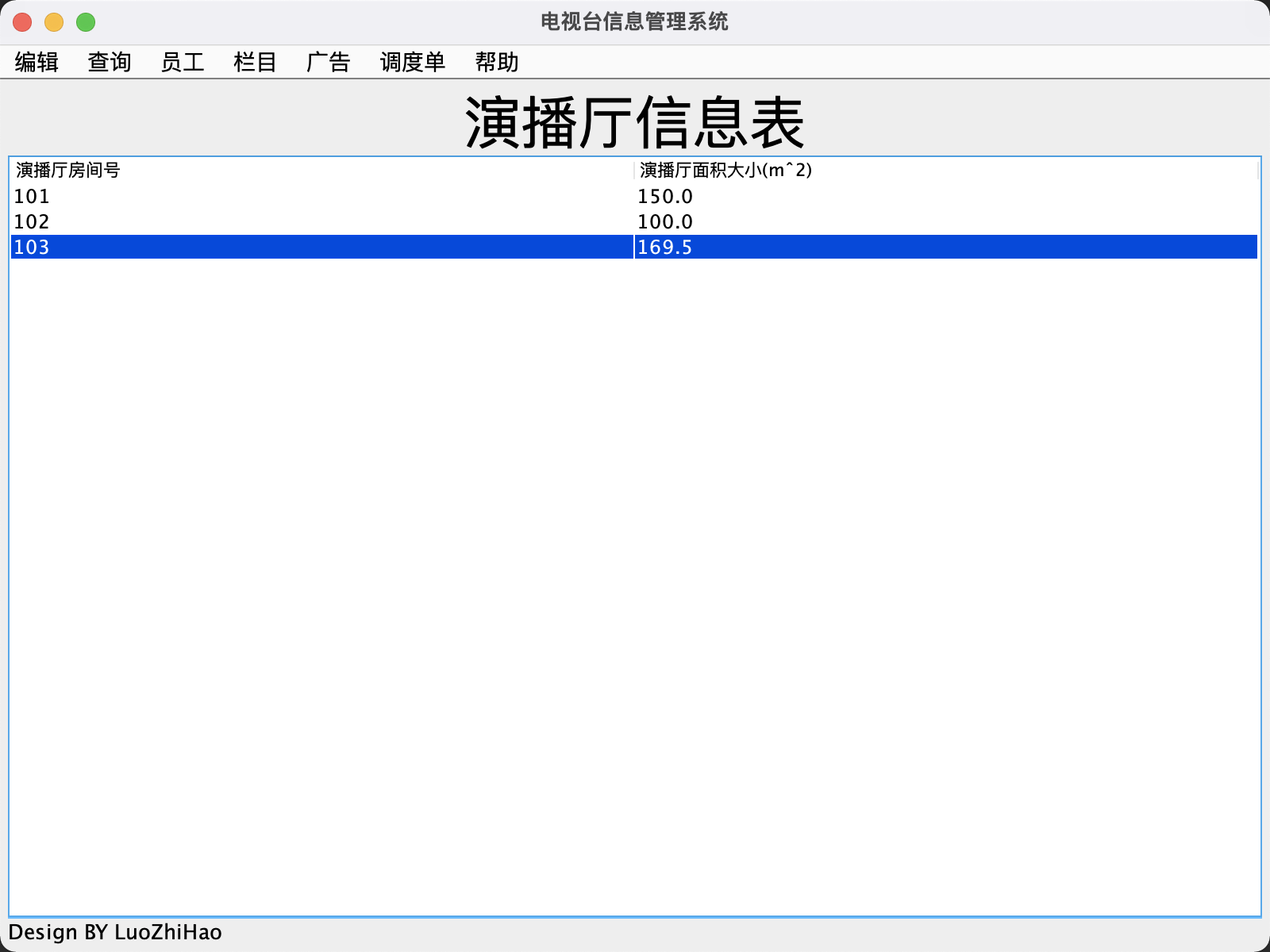


演播厅信息管理模块中，先在GUI的user GUI包中创建主窗口，以用来显示从数据库读取出来的员工的详细信息，展现在面板中，并且提供滚轮滑块，以便阅读更多行数据。在主窗口添加菜单栏 JMenuBar来添加按钮以提供增删改查（CRUD）的操作。在user GUI包中分别创建其他的操作窗口，提供可视化的增删改查。

其中的数据来源于DAO层中的数据库持久层。

### 6.4.2 模块功能介绍及界面截图

运行结果截图：

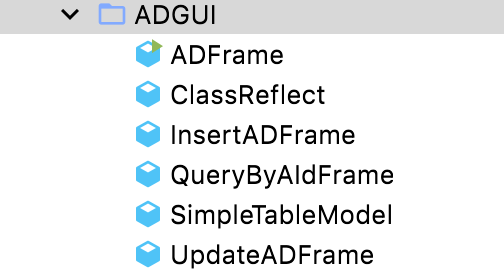


### 6.4.3 模块核心代码

**private** JPanel **contentPane**;  
**private** List<String> **tvrCols**;  
**private** List<TelevisionRoom> **tvrRows**;  
**private** TVRoomServiceImpl **tvRoomService**;  
**private** SimpleTableModel<TelevisionRoom> **simpleTableModel**;  
**private** JTable **jTable**;  
  
**public** TVRoomFrame(){  
 setTitle(**"电视台信息管理系统"**);  
 setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
 setBounds(600,400,800,600);  
 **contentPane** = **new** JPanel();  
 **contentPane**.setBorder(**new** EmptyBorder(5,5,5,5));  
 **contentPane**.setLayout(**new** BorderLayout(0,0));  
 setContentPane(**contentPane**);  
 setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);  
  
 **tvrCols** = **new** ArrayList<>();  
 **tvrCols**.add(**"演播厅房间号"**);  
 **tvrCols**.add(**"演播厅面积大小(mˆ2)"**);  
 **tvRoomService** = **new** TVRoomServiceImpl();  
 **try** {  
 **tvrRows** = **tvRoomService**.getAllTvRoom();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel** = **new** SimpleTableModel<TelevisionRoom>(**tvrCols**, **tvrRows**);  
  
 JLabel jLabel = **new** JLabel();  
 jLabel.setText(**"演播厅信息表"**);  
 jLabel.setFont(**new** Font(**"Lucida Grande"**, Font.***PLAIN***,36));  
 jLabel.setHorizontalAlignment(SwingConstants.***CENTER***);  
 getContentPane().add(jLabel,BorderLayout.***PAGE\_START***);  
 JScrollPane jScrollPane = **new** JScrollPane();  
 **jTable** = **new** JTable();  
  
 **jTable**.setModel(**simpleTableModel**);  
 jScrollPane.setViewportView(**jTable**);  
 getContentPane().add(jScrollPane, BorderLayout.***CENTER***);  
  
 JLabel jLabel2 = **new** JLabel();  
 jLabel2.setText(**"Design BY LuoZhiHao "**);  
 getContentPane().add(jLabel2, BorderLayout.***PAGE\_END***);  
  
 JMenuBar jMenuBar = **new** JMenuBar();  
  
 JMenu edit = **new** JMenu();  
 edit.setText(**"编辑"**);  
 edit.setToolTipText(**""**);  
 JMenuItem insert = **new** JMenuItem();  
 insert.setText(**"插入数据"**);  
 insert.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 insertUserActionPerformed(e);  
 }  
 });  
 edit.add(insert);  
  
 JMenuItem update = **new** JMenuItem();  
 update.setText(**"修改数据"**);  
 update.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 updateUserActionPerformed(evt);  
 }  
 });  
 edit.add(update);  
  
 JMenuItem refresh = **new** JMenuItem();  
 refresh.setText(**"刷新"**);  
 refresh.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 refreshActionPerformed(evt);  
 }  
 });  
 edit.add(refresh);  
  
 JMenuItem delete = **new** JMenuItem(**"删除"**);  
 delete.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 delUserActionPerformed(e);  
 }  
 });  
 edit.add(delete);  
 jMenuBar.add(edit);  
  
 JMenu query = **new** JMenu();  
 query.setText(**"查询"**);  
 JMenuItem queryAll = **new** JMenuItem(**"查询所有演播厅数据"**);  
 queryAll.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryAllActionPerformed(e);  
 }  
 });  
 query.add(queryAll);  
 JMenuItem queryByUid = **new** JMenuItem(**"通过编号查询"**);  
 query.add(queryByUid);  
 queryByUid.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryByUidActionPerformed(e);  
 }  
 });  
 jMenuBar.add(query);  
  
 JMenu User = **new** JMenu(**"员工"**);  
 JMenuItem userQuery = **new** JMenuItem(**"员工管理"**);  
 userQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showUserActionPerformed(e);  
 }  
 });  
 User.add(userQuery);  
 jMenuBar.add(User);  
  
 JMenu Column = **new** JMenu(**"栏目"**);  
 JMenuItem columnQuery = **new** JMenuItem(**"栏目管理"**);  
 columnQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showColumnsActionPerformed(e);  
 }  
 });  
 Column.add(columnQuery);  
 jMenuBar.add(Column);  
  
 JMenu AD = **new** JMenu(**"广告"**);  
 JMenuItem adQuery = **new** JMenuItem(**"广告管理"**);  
 adQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showADActionPerformed(e);  
 }  
 });  
 AD.add(adQuery);  
 jMenuBar.add(AD);  
  
 JMenu DP = **new** JMenu(**"调度单"**);  
 JMenuItem dpQuery = **new** JMenuItem(**"调度单管理"**);  
 dpQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showDPActionPerformed(e);  
 }  
 });  
 DP.add(dpQuery);  
 jMenuBar.add(DP);  
  
 JMenu help = **new** JMenu(**"帮助"**);  
 JMenuItem about = **new** JMenuItem(**"关于"**);  
 about.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 aboutActionPerformed(e);  
 }  
  
 **private void** aboutActionPerformed(ActionEvent e) {  
 JOptionPane.*showMessageDialog*(**contentPane**,**"欢迎使用电视台信息管理系统"**,**"by LuoZhiHao"**,JOptionPane.***PLAIN\_MESSAGE***);  
 }  
 });  
 help.add(about);  
 jMenuBar.add(help);  
 setJMenuBar(jMenuBar);  
 pack();  
}  
  
*//转到调度单管理界面***private void** showDPActionPerformed(ActionEvent e) {  
 DispatchFrame dispatchFrame = **new** DispatchFrame();  
 dispatchFrame.setBounds(600,400,800,600);  
 dispatchFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showUserActionPerformed(ActionEvent e) {  
 MainFrame mainFrame = **new** MainFrame();  
 mainFrame.setBounds(600,400,800,600);  
 mainFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showADActionPerformed(ActionEvent e) {  
 ADFrame adFrame = **new** ADFrame();  
 adFrame.setBounds(600,400,800,600);  
 adFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showColumnsActionPerformed(ActionEvent e) {  
 ColumnFrame columnFrame = **new** ColumnFrame();  
 columnFrame.setBounds(600,400,800,600);  
 columnFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
  
}  
  
**private void** queryByUidActionPerformed(ActionEvent e) {  
 **try** {  
 **tvrRows** = **tvRoomService**.getAllTvRoom();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**tvrRows**);  
 **new** QueryByTvrIdFrame(**jTable**).setVisible(**true**);  
}  
  
**private void** queryAllActionPerformed(ActionEvent e) {  
 **try** {  
 **tvrRows** = **tvRoomService**.getAllTvRoom();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**tvrRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** delUserActionPerformed(ActionEvent e) {  
 TelevisionRoom tvr = **tvrRows**.get(**jTable**.getSelectedRow());  
 **try** {  
 **tvRoomService**.delTVRoomById(tvr.getTvrId());  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"删除失败，系统错误"**);  
 }  
 **try** {  
 **tvrRows** = **tvRoomService**.getAllTvRoom();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"查询失败，系统错误"**);  
 }  
 **simpleTableModel**.setRows(**tvrRows**);  
 **simpleTableModel**.fireTableDataChanged();  
 JOptionPane.*showMessageDialog*(**this**,**"删除演播厅"**+tvr.getTvrId() +**"信息成功！"**,**"删除成功"**,JOptionPane.***INFORMATION\_MESSAGE***);  
}  
  
**private void** refreshActionPerformed(ActionEvent evt){  
 **try** {  
 **tvrRows** = **tvRoomService**.getAllTvRoom();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**tvrRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** updateUserActionPerformed(ActionEvent evt){  
 **try** {  
 **tvrRows** = **tvRoomService**.getAllTvRoom();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**tvrRows**);  
 **if** (**jTable**.getSelectedRow() == -1){  
 JOptionPane.*showMessageDialog*(**this**,**"请先用鼠标选中数据"**);  
 }  
 TelevisionRoom tvr = **tvrRows**.get(**jTable**.getSelectedRow());  
 **new** UpdateTvRoomFrame(tvr).setVisible(**true**);  
}  
  
**private void** insertUserActionPerformed(ActionEvent e) {  
 **new** InsertTvrJFrame().setVisible(**true**);  
}

## 6.5 广告信息管理模块设计

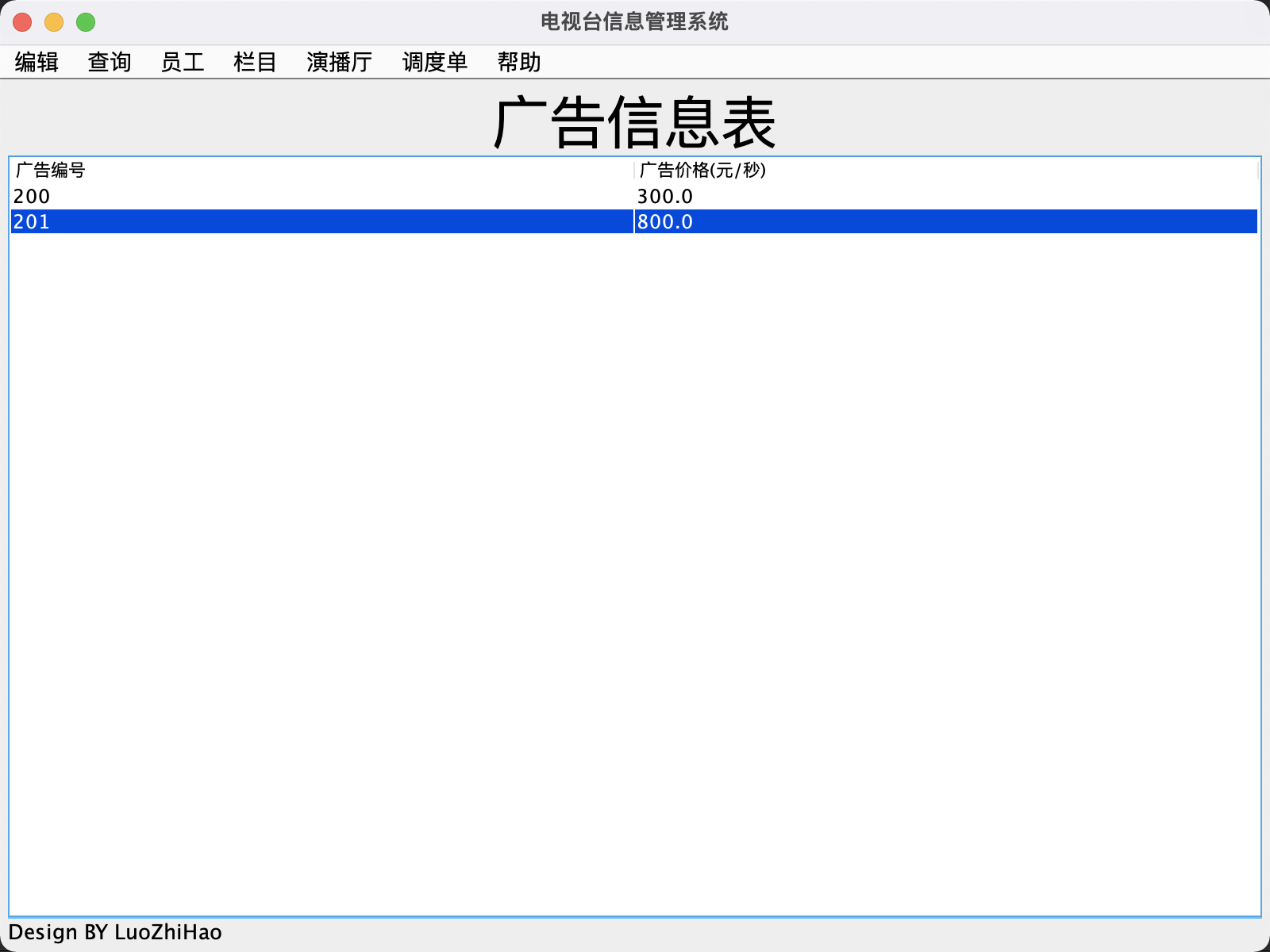
### 6.5.1 模块算法描述



广告信息管理模块中，先在GUI的user GUI包中创建主窗口，以用来显示从数据库读取出来的员工的详细信息，展现在面板中，并且提供滚轮滑块，以便阅读更多行数据。在主窗口添加菜单栏 JMenuBar来添加按钮以提供增删改查（CRUD）的操作。在user GUI包中分别创建其他的操作窗口，提供可视化的增删改查。

其中的数据来源于DAO层中的数据库持久层。

### 6.5.2 模块功能介绍及界面截图

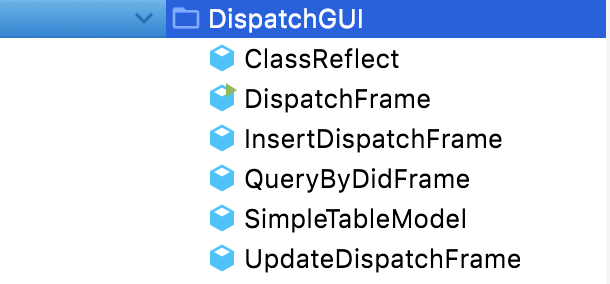


### 6.5.3 模块核心代码

**private** JPanel **contentPane**;  
**private** List<String> **adCols**;  
**private** List<Advertisement> **adRows**;  
**private** ADServiceImpl **adService**;  
**private** SimpleTableModel<Advertisement> **simpleTableModel**;  
**private** JTable **jTable**;  
  
**public** ADFrame(){  
 setTitle(**"电视台信息管理系统"**);  
 setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
 setBounds(600,400,800,600);  
 **contentPane** = **new** JPanel();  
 **contentPane**.setBorder(**new** EmptyBorder(5,5,5,5));  
 **contentPane**.setLayout(**new** BorderLayout(0,0));  
 setContentPane(**contentPane**);  
 setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);  
  
 **adCols** = **new** ArrayList<>();  
 **adCols**.add(**"广告编号"**);  
 **adCols**.add(**"广告价格(元/秒)"**);  
 **adService** = **new** ADServiceImpl();  
 **try** {  
 **adRows** = **adService**.getAllAD();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel** = **new** SimpleTableModel<Advertisement>(**adCols**, **adRows**);  
  
 JLabel jLabel = **new** JLabel();  
 jLabel.setText(**"广告信息表"**);  
 jLabel.setFont(**new** Font(**"Lucida Grande"**, Font.***PLAIN***,36));  
 jLabel.setHorizontalAlignment(SwingConstants.***CENTER***);  
 getContentPane().add(jLabel,BorderLayout.***PAGE\_START***);  
 JScrollPane jScrollPane = **new** JScrollPane();  
 **jTable** = **new** JTable();  
  
 **jTable**.setModel(**simpleTableModel**);  
 jScrollPane.setViewportView(**jTable**);  
 getContentPane().add(jScrollPane, BorderLayout.***CENTER***);  
  
 JLabel jLabel2 = **new** JLabel();  
 jLabel2.setText(**"Design BY LuoZhiHao "**);  
 getContentPane().add(jLabel2, BorderLayout.***PAGE\_END***);  
  
 JMenuBar jMenuBar = **new** JMenuBar();  
  
 JMenu edit = **new** JMenu();  
 edit.setText(**"编辑"**);  
 edit.setToolTipText(**""**);  
 JMenuItem insert = **new** JMenuItem();  
 insert.setText(**"插入数据"**);  
 insert.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 insertUserActionPerformed(e);  
 }  
 });  
 edit.add(insert);  
  
 JMenuItem update = **new** JMenuItem();  
 update.setText(**"修改数据"**);  
 update.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 updateUserActionPerformed(evt);  
 }  
 });  
 edit.add(update);  
  
 JMenuItem refresh = **new** JMenuItem();  
 refresh.setText(**"刷新"**);  
 refresh.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 refreshActionPerformed(evt);  
 }  
 });  
 edit.add(refresh);  
  
 JMenuItem delete = **new** JMenuItem(**"删除"**);  
 delete.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 delUserActionPerformed(e);  
 }  
 });  
 edit.add(delete);  
 jMenuBar.add(edit);  
  
 JMenu query = **new** JMenu();  
 query.setText(**"查询"**);  
 JMenuItem queryAll = **new** JMenuItem(**"查询所有演播厅数据"**);  
 queryAll.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryAllActionPerformed(e);  
 }  
 });  
 query.add(queryAll);  
 JMenuItem queryByUid = **new** JMenuItem(**"通过编号查询"**);  
 query.add(queryByUid);  
 queryByUid.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryByUidActionPerformed(e);  
 }  
 });  
 jMenuBar.add(query);  
  
 JMenu User = **new** JMenu(**"员工"**);  
 JMenuItem userQuery = **new** JMenuItem(**"员工管理"**);  
 userQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showUserActionPerformed(e);  
 }  
 });  
 User.add(userQuery);  
 jMenuBar.add(User);  
  
 JMenu Column = **new** JMenu(**"栏目"**);  
 JMenuItem columnQuery = **new** JMenuItem(**"栏目管理"**);  
 columnQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showColumnsActionPerformed(e);  
 }  
 });  
 Column.add(columnQuery);  
 jMenuBar.add(Column);  
  
 JMenu TVR = **new** JMenu(**"演播厅"**);  
 JMenuItem tvrQuery = **new** JMenuItem(**"演播厅管理"**);  
 tvrQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showTVRActionPerformed(e);  
 }  
 });  
 TVR.add(tvrQuery);  
 jMenuBar.add(TVR);  
  
 JMenu DP = **new** JMenu(**"调度单"**);  
 JMenuItem dpQuery = **new** JMenuItem(**"调度单管理"**);  
 dpQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showDPActionPerformed(e);  
 }  
 });  
 DP.add(dpQuery);  
 jMenuBar.add(DP);  
  
 JMenu help = **new** JMenu(**"帮助"**);  
 JMenuItem about = **new** JMenuItem(**"关于"**);  
 about.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 aboutActionPerformed(e);  
 }  
  
 **private void** aboutActionPerformed(ActionEvent e) {  
 JOptionPane.*showMessageDialog*(**contentPane**,**"欢迎使用电视台信息管理系统"**,**"by LuoZhiHao"**,JOptionPane.***PLAIN\_MESSAGE***);  
 }  
 });  
 help.add(about);  
 jMenuBar.add(help);  
 setJMenuBar(jMenuBar);  
 pack();  
}  
  
*//转到调度单管理界面***private void** showDPActionPerformed(ActionEvent e) {  
 DispatchFrame dispatchFrame = **new** DispatchFrame();  
 dispatchFrame.setBounds(600,400,800,600);  
 dispatchFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showTVRActionPerformed(ActionEvent e) {  
 TVRoomFrame tvRoomFrame = **new** TVRoomFrame();  
 tvRoomFrame.setBounds(600,400,800,600);  
 tvRoomFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showUserActionPerformed(ActionEvent e) {  
 MainFrame mainFrame = **new** MainFrame();  
 mainFrame.setBounds(600,400,800,600);  
 mainFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showColumnsActionPerformed(ActionEvent e) {  
 ColumnFrame columnFrame = **new** ColumnFrame();  
 columnFrame.setBounds(600,400,800,600);  
 columnFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
  
}  
  
**private void** queryByUidActionPerformed(ActionEvent e) {  
 **try** {  
 **adRows** = **adService**.getAllAD();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**adRows**);  
 **new** QueryByAIdFrame(**jTable**).setVisible(**true**);  
}  
  
**private void** queryAllActionPerformed(ActionEvent e) {  
 **try** {  
 **adRows** = **adService**.getAllAD();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**adRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** delUserActionPerformed(ActionEvent e) {  
 Advertisement advertisement = **adRows**.get(**jTable**.getSelectedRow());  
 **try** {  
 **adService**.delADById(advertisement.getaId());  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"删除失败，系统错误"**);  
 }  
 **try** {  
 **adRows** = **adService**.getAllAD();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"查询失败，系统错误"**);  
 }  
 **simpleTableModel**.setRows(**adRows**);  
 **simpleTableModel**.fireTableDataChanged();  
 JOptionPane.*showMessageDialog*(**this**,**"删除演播厅"**+advertisement.getaId() +**"信息成功！"**,**"删除成功"**,JOptionPane.***INFORMATION\_MESSAGE***);  
}  
  
**private void** refreshActionPerformed(ActionEvent evt){  
 **try** {  
 **adRows** = **adService**.getAllAD();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**adRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** updateUserActionPerformed(ActionEvent evt){  
 **try** {  
 **adRows** = **adService**.getAllAD();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**adRows**);  
 **if** (**jTable**.getSelectedRow() == -1){  
 JOptionPane.*showMessageDialog*(**this**,**"请先用鼠标选中数据"**);  
 }  
 Advertisement advertisement = **adRows**.get(**jTable**.getSelectedRow());  
 **new** UpdateADFrame(advertisement).setVisible(**true**);  
}  
  
**private void** insertUserActionPerformed(ActionEvent e) {  
 **new** InsertADFrame().setVisible(**true**);  
}

## 6.6 调度单信息管理模块设计

### 6.6.1 模块算法描述



调度单信息管理模块中，先在GUI的user GUI包中创建主窗口，以用来显示从数据库读取出来的员工的详细信息，展现在面板中，并且提供滚轮滑块，以便阅读更多行数据。在主窗口添加菜单栏 JMenuBar来添加按钮以提供增删改查（CRUD）的操作。在user GUI包中分别创建其他的操作窗口，提供可视化的增删改查。

其中的数据来源于DAO层中的数据库持久层。

### 6.6.2 模块功能介绍及界面截图



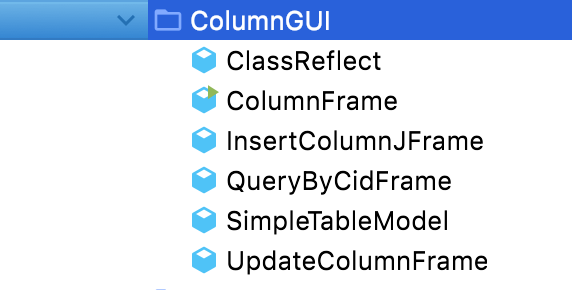
### 6.6.3 模块核心代码

实现主窗口的核心代码：

**private** JPanel **contentPane**;  
**private** List<String> **dpCols**;  
**private** List<DispatchList> **dpRows**;  
**private** DispatchServiceImpl **dispatchService**;  
**private** SimpleTableModel<DispatchList> **simpleTableModel**;  
**private** JTable **jTable**;  
  
**public** DispatchFrame(){  
 setTitle(**"电视台信息管理系统"**);  
 setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
 setBounds(600,400,800,600);  
 **contentPane** = **new** JPanel();  
 **contentPane**.setBorder(**new** EmptyBorder(5,5,5,5));  
 **contentPane**.setLayout(**new** BorderLayout(0,0));  
 setContentPane(**contentPane**);  
 setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);  
  
 **dpCols** = **new** ArrayList<>();  
 **dpCols**.add(**"调度单编号"**);  
 **dpCols**.add(**"栏目编号"**);  
 **dpCols**.add(**"演播厅编号"**);  
 **dpCols**.add(**"场协员工编号"**);  
 **dispatchService** = **new** DispatchServiceImpl();  
 **try** {  
 **dpRows** = **dispatchService**.getAlldispatch();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel** = **new** SimpleTableModel<DispatchList>(**dpCols**, **dpRows**);  
  
 JLabel jLabel = **new** JLabel();  
 jLabel.setText(**"栏目信息表"**);  
 jLabel.setFont(**new** Font(**"Lucida Grande"**, Font.***PLAIN***,36));  
 jLabel.setHorizontalAlignment(SwingConstants.***CENTER***);  
 getContentPane().add(jLabel,BorderLayout.***PAGE\_START***);  
 JScrollPane jScrollPane = **new** JScrollPane();  
 **jTable** = **new** JTable();  
  
 **jTable**.setModel(**simpleTableModel**);  
 jScrollPane.setViewportView(**jTable**);  
 getContentPane().add(jScrollPane, BorderLayout.***CENTER***);  
  
 JLabel jLabel2 = **new** JLabel();  
 jLabel2.setText(**"Design BY LuoZhiHao "**);  
 getContentPane().add(jLabel2, BorderLayout.***PAGE\_END***);  
  
 JMenuBar jMenuBar = **new** JMenuBar();  
  
 JMenu edit = **new** JMenu();  
 edit.setText(**"编辑"**);  
 edit.setToolTipText(**""**);  
 JMenuItem insert = **new** JMenuItem();  
 insert.setText(**"插入数据"**);  
 insert.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 insertUserActionPerformed(e);  
 }  
 });  
 edit.add(insert);  
  
 JMenuItem update = **new** JMenuItem();  
 update.setText(**"修改数据"**);  
 update.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 updateUserActionPerformed(evt);  
 }  
 });  
 edit.add(update);  
  
 JMenuItem refresh = **new** JMenuItem();  
 refresh.setText(**"刷新"**);  
 refresh.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 refreshActionPerformed(evt);  
 }  
 });  
 edit.add(refresh);  
  
 JMenuItem delete = **new** JMenuItem(**"删除"**);  
 delete.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 delUserActionPerformed(e);  
 }  
 });  
 edit.add(delete);  
 jMenuBar.add(edit);  
  
 JMenu query = **new** JMenu();  
 query.setText(**"查询"**);  
 JMenuItem queryAll = **new** JMenuItem(**"查询所有栏目数据"**);  
 queryAll.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryAllActionPerformed(e);  
 }  
 });  
 query.add(queryAll);  
 JMenuItem queryByUid = **new** JMenuItem(**"通过编号查询"**);  
 query.add(queryByUid);  
 queryByUid.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryByUidActionPerformed(e);  
 }  
 });  
 jMenuBar.add(query);  
  
 JMenu User = **new** JMenu(**"员工"**);  
 JMenuItem userQuery = **new** JMenuItem(**"员工管理"**);  
 userQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showUserActionPerformed(e);  
 }  
 });  
 User.add(userQuery);  
 jMenuBar.add(User);  
  
 JMenu Column = **new** JMenu(**"栏目"**);  
 JMenuItem columnQuery = **new** JMenuItem(**"栏目管理"**);  
 columnQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showColumnsActionPerformed(e);  
 }  
 });  
 Column.add(columnQuery);  
 jMenuBar.add(Column);  
  
 JMenu TVR = **new** JMenu(**"演播厅"**);  
 JMenuItem tvrQuery = **new** JMenuItem(**"演播厅管理"**);  
 tvrQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showTVRActionPerformed(e);  
 }  
 });  
 TVR.add(tvrQuery);  
 jMenuBar.add(TVR);  
  
 JMenu AD = **new** JMenu(**"广告"**);  
 JMenuItem adQuery = **new** JMenuItem(**"广告管理"**);  
 adQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showADActionPerformed(e);  
 }  
 });  
 AD.add(adQuery);  
 jMenuBar.add(AD);  
  
 JMenu help = **new** JMenu(**"帮助"**);  
 JMenuItem about = **new** JMenuItem(**"关于"**);  
 about.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 aboutActionPerformed(e);  
 }  
  
 **private void** aboutActionPerformed(ActionEvent e) {  
 JOptionPane.*showMessageDialog*(**contentPane**,**"欢迎使用电视台信息管理系统"**,**"by LuoZhiHao"**,JOptionPane.***PLAIN\_MESSAGE***);  
 }  
 });  
 help.add(about);  
 jMenuBar.add(help);  
 setJMenuBar(jMenuBar);  
 pack();  
}  
  
*//转到演播厅管理界面***private void** showTVRActionPerformed(ActionEvent e) {  
 TVRoomFrame tvRoomFrame = **new** TVRoomFrame();  
 tvRoomFrame.setBounds(600,400,800,600);  
 tvRoomFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showUserActionPerformed(ActionEvent e) {  
 MainFrame mainFrame = **new** MainFrame();  
 mainFrame.setBounds(600,400,800,600);  
 mainFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showADActionPerformed(ActionEvent e) {  
 ADFrame adFrame = **new** ADFrame();  
 adFrame.setBounds(600,400,800,600);  
 adFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showColumnsActionPerformed(ActionEvent e) {  
 ColumnFrame columnFrame = **new** ColumnFrame();  
 columnFrame.setBounds(600,400,800,600);  
 columnFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
  
}  
  
**private void** queryByUidActionPerformed(ActionEvent e) {  
 **try** {  
 **dpRows** = **dispatchService**.getAlldispatch();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**dpRows**);  
 **new** QueryByDidFrame(**jTable**).setVisible(**true**);  
}  
  
**private void** queryAllActionPerformed(ActionEvent e) {  
 **try** {  
 **dpRows** = **dispatchService**.getAlldispatch();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**dpRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** delUserActionPerformed(ActionEvent e) {  
 DispatchList dispatchList = **dpRows**.get(**jTable**.getSelectedRow());  
 **try** {  
 **dispatchService**.delDispatchById(dispatchList.getcId());  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"删除失败，系统错误"**);  
 }  
 **try** {  
 **dpRows** = **dispatchService**.getAlldispatch();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"查询失败，系统错误"**);  
 }  
 **simpleTableModel**.setRows(**dpRows**);  
 **simpleTableModel**.fireTableDataChanged();  
 JOptionPane.*showMessageDialog*(**this**,**"删除调度单"**+dispatchList.getdId() +**"信息成功！"**,**"删除成功"**,JOptionPane.***INFORMATION\_MESSAGE***);  
}  
  
**private void** refreshActionPerformed(ActionEvent evt){  
 **try** {  
 **dpRows** = **dispatchService**.getAlldispatch();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**dpRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** updateUserActionPerformed(ActionEvent evt){  
 **try** {  
 **dpRows** = **dispatchService**.getAlldispatch();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**dpRows**);  
 **if** (**jTable**.getSelectedRow() == -1){  
 JOptionPane.*showMessageDialog*(**this**,**"请先用鼠标选中数据"**);  
 }  
 DispatchList dispatchList = **dpRows**.get(**jTable**.getSelectedRow());  
 **new** UpdateDispatchFrame(dispatchList).setVisible(**true**);  
}  
  
**private void** insertUserActionPerformed(ActionEvent e) {  
 **new** InsertDispatchFrame().setVisible(**true**);  
}

## 6.7 栏目信息管理模块设计

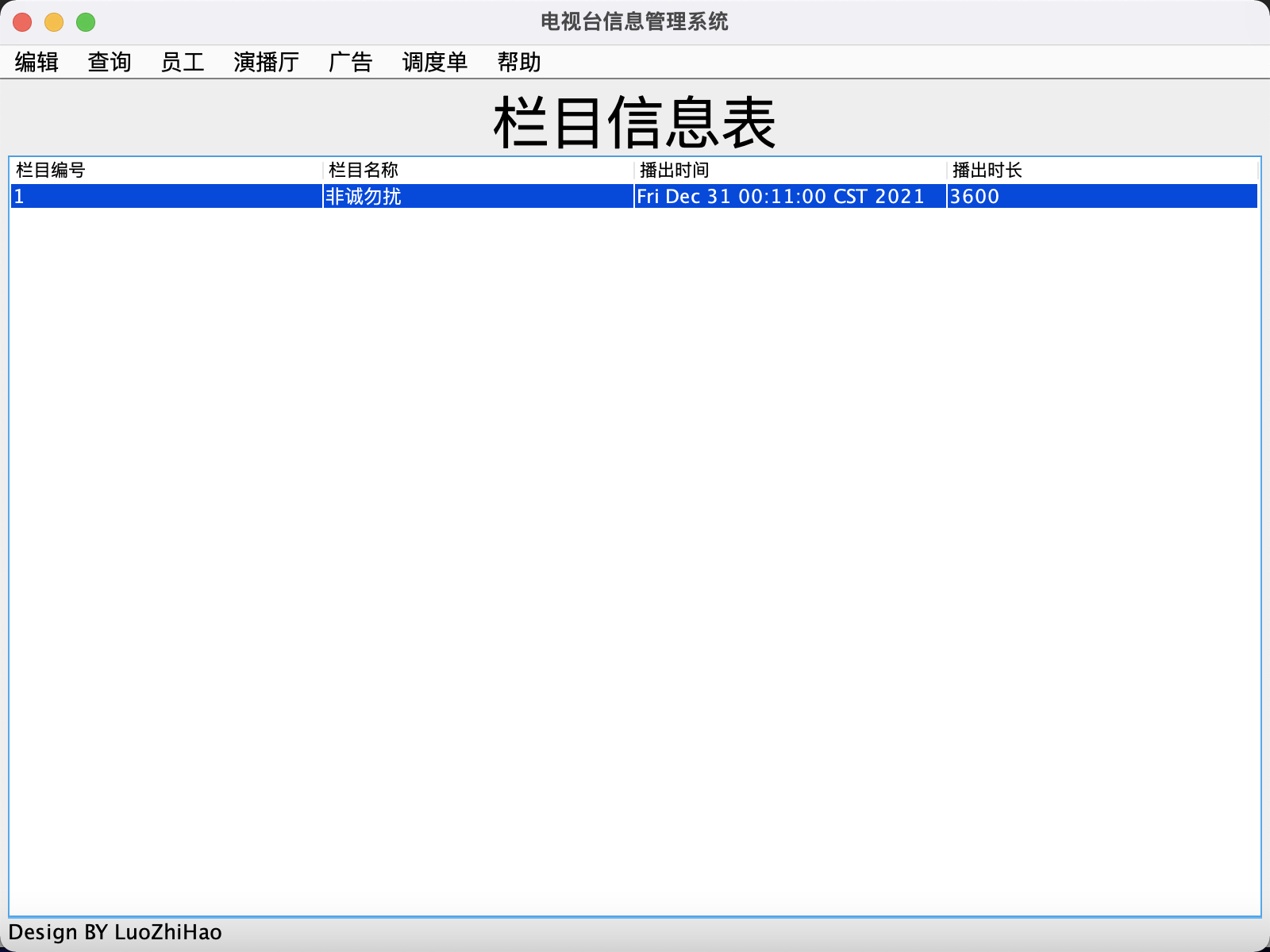
### 6.7.1 模块算法描述



栏目信息管理模块中，先在GUI的user GUI包中创建主窗口，以用来显示从数据库读取出来的员工的详细信息，展现在面板中，并且提供滚轮滑块，以便阅读更多行数据。在主窗口添加菜单栏 JMenuBar来添加按钮以提供增删改查（CRUD）的操作。在user GUI包中分别创建其他的操作窗口，提供可视化的增删改查。

其中的数据来源于DAO层中的数据库持久层

### 6.7.2 模块功能介绍及界面截图



### 6.7.3 模块核心代码

实现栏目信息管理的主窗口代码：

**private** JPanel **contentPane**;  
**private** List<String> **columnCols**;  
**private** List<Column> **columnRows**;  
**private** ColumnServiceImpl **columnService**;  
**private** SimpleTableModel<Column> **simpleTableModel**;  
**private** JTable **jTable**;  
  
**public** ColumnFrame(){  
 setTitle(**"电视台信息管理系统"**);  
 setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
 setBounds(600,400,800,600);  
 **contentPane** = **new** JPanel();  
 **contentPane**.setBorder(**new** EmptyBorder(5,5,5,5));  
 **contentPane**.setLayout(**new** BorderLayout(0,0));  
 setContentPane(**contentPane**);  
 setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);  
  
 **columnCols** = **new** ArrayList<>();  
 **columnCols**.add(**"栏目编号"**);  
 **columnCols**.add(**"栏目名称"**);  
 **columnCols**.add(**"播出时间"**);  
 **columnCols**.add(**"播出时长"**);  
 **columnService** = **new** ColumnServiceImpl();  
 **try** {  
 **columnRows** = **columnService**.getAllColumn();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel** = **new** SimpleTableModel<Column>(**columnCols**, **columnRows**);  
  
 JLabel jLabel = **new** JLabel();  
 jLabel.setText(**"栏目信息表"**);  
 jLabel.setFont(**new** Font(**"Lucida Grande"**, Font.***PLAIN***,36));  
 jLabel.setHorizontalAlignment(SwingConstants.***CENTER***);  
 getContentPane().add(jLabel,BorderLayout.***PAGE\_START***);  
 JScrollPane jScrollPane = **new** JScrollPane();  
 **jTable** = **new** JTable();  
  
 **jTable**.setModel(**simpleTableModel**);  
 jScrollPane.setViewportView(**jTable**);  
 getContentPane().add(jScrollPane, BorderLayout.***CENTER***);  
  
 JLabel jLabel2 = **new** JLabel();  
 jLabel2.setText(**"Design BY LuoZhiHao "**);  
 getContentPane().add(jLabel2, BorderLayout.***PAGE\_END***);  
  
 JMenuBar jMenuBar = **new** JMenuBar();  
  
 JMenu edit = **new** JMenu();  
 edit.setText(**"编辑"**);  
 edit.setToolTipText(**""**);  
 JMenuItem insert = **new** JMenuItem();  
 insert.setText(**"插入数据"**);  
 insert.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 insertUserActionPerformed(e);  
 }  
 });  
 edit.add(insert);  
  
 JMenuItem update = **new** JMenuItem();  
 update.setText(**"修改数据"**);  
 update.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 updateUserActionPerformed(evt);  
 }  
 });  
 edit.add(update);  
  
 JMenuItem refresh = **new** JMenuItem();  
 refresh.setText(**"刷新"**);  
 refresh.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent evt) {  
 refreshActionPerformed(evt);  
 }  
 });  
 edit.add(refresh);  
  
 JMenuItem delete = **new** JMenuItem(**"删除"**);  
 delete.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 delUserActionPerformed(e);  
 }  
 });  
 edit.add(delete);  
 jMenuBar.add(edit);  
  
 JMenu query = **new** JMenu();  
 query.setText(**"查询"**);  
 JMenuItem queryAll = **new** JMenuItem(**"查询所有栏目数据"**);  
 queryAll.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryAllActionPerformed(e);  
 }  
 });  
 query.add(queryAll);  
 JMenuItem queryByUid = **new** JMenuItem(**"通过编号查询"**);  
 query.add(queryByUid);  
 queryByUid.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 queryByUidActionPerformed(e);  
 }  
 });  
 jMenuBar.add(query);  
  
 JMenu User = **new** JMenu(**"员工"**);  
 JMenuItem userQuery = **new** JMenuItem(**"员工管理"**);  
 userQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showUserActionPerformed(e);  
 }  
 });  
 User.add(userQuery);  
 jMenuBar.add(User);  
  
 JMenu TVR = **new** JMenu(**"演播厅"**);  
 JMenuItem tvrQuery = **new** JMenuItem(**"演播厅管理"**);  
 tvrQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showTVRActionPerformed(e);  
 }  
 });  
 TVR.add(tvrQuery);  
 jMenuBar.add(TVR);  
  
 JMenu AD = **new** JMenu(**"广告"**);  
 JMenuItem adQuery = **new** JMenuItem(**"广告管理"**);  
 adQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showADActionPerformed(e);  
 }  
 });  
 AD.add(adQuery);  
 jMenuBar.add(AD);  
  
 JMenu DP = **new** JMenu(**"调度单"**);  
 JMenuItem dpQuery = **new** JMenuItem(**"调度单管理"**);  
 dpQuery.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 showDPActionPerformed(e);  
 }  
 });  
 DP.add(dpQuery);  
 jMenuBar.add(DP);  
  
 JMenu help = **new** JMenu(**"帮助"**);  
 JMenuItem about = **new** JMenuItem(**"关于"**);  
 about.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 aboutActionPerformed(e);  
 }  
  
 **private void** aboutActionPerformed(ActionEvent e) {  
 JOptionPane.*showMessageDialog*(**contentPane**,**"欢迎使用电视台信息管理系统"**,**"by LuoZhiHao"**,JOptionPane.***PLAIN\_MESSAGE***);  
 }  
 });  
 help.add(about);  
 jMenuBar.add(help);  
 setJMenuBar(jMenuBar);  
 pack();  
}  
  
*//转到调度单管理界面***private void** showDPActionPerformed(ActionEvent e) {  
 DispatchFrame dispatchFrame = **new** DispatchFrame();  
 dispatchFrame.setBounds(600,400,800,600);  
 dispatchFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showTVRActionPerformed(ActionEvent e) {  
 TVRoomFrame tvRoomFrame = **new** TVRoomFrame();  
 tvRoomFrame.setBounds(600,400,800,600);  
 tvRoomFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showUserActionPerformed(ActionEvent e) {  
 MainFrame mainFrame = **new** MainFrame();  
 mainFrame.setBounds(600,400,800,600);  
 mainFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
**private void** showADActionPerformed(ActionEvent e) {  
 ADFrame adFrame = **new** ADFrame();  
 adFrame.setBounds(600,400,800,600);  
 adFrame.setVisible(**true**);  
 **this**.setVisible(**false**);  
 **this**.dispose();  
}  
  
  
**private void** queryByUidActionPerformed(ActionEvent e) {  
 **try** {  
 **columnRows** = **columnService**.getAllColumn();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**columnRows**);  
 **new** QueryByCidFrame(**jTable**).setVisible(**true**);  
}  
  
**private void** queryAllActionPerformed(ActionEvent e) {  
 **try** {  
 **columnRows** = **columnService**.getAllColumn();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**columnRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** delUserActionPerformed(ActionEvent e) {  
 Column column = **columnRows**.get(**jTable**.getSelectedRow());  
 **try** {  
 **columnService**.delColumnById(column.getcId());  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"删除失败，系统错误"**);  
 }  
 **try** {  
 **columnRows** = **columnService**.getAllColumn();  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 System.***out***.println(**"查询失败，系统错误"**);  
 }  
 **simpleTableModel**.setRows(**columnRows**);  
 **simpleTableModel**.fireTableDataChanged();  
 JOptionPane.*showMessageDialog*(**this**,**"删除栏目"**+column.getcName() +**"信息成功！"**,**"删除成功"**,JOptionPane.***INFORMATION\_MESSAGE***);  
}  
  
**private void** refreshActionPerformed(ActionEvent evt){  
 **try** {  
 **columnRows** = **columnService**.getAllColumn();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**columnRows**);  
 **simpleTableModel**.fireTableDataChanged();  
}  
  
**private void** updateUserActionPerformed(ActionEvent evt){  
 **try** {  
 **columnRows** = **columnService**.getAllColumn();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **simpleTableModel**.setRows(**columnRows**);  
 **if** (**jTable**.getSelectedRow() == -1){  
 JOptionPane.*showMessageDialog*(**this**,**"请先用鼠标选中数据"**);  
 }  
 Column column = **columnRows**.get(**jTable**.getSelectedRow());  
 **new** UpdateColumnFrame(column).setVisible(**true**);  
}  
  
**private void** insertUserActionPerformed(ActionEvent e) {  
 **new** InsertColumnJFrame().setVisible(**true**);  
}

7 总结

开发一个应用项目需要团队的合作，多个人分别负责不同的模块，这样可以有效的减少错误的发生，这次项目为单独一个人完成，期间出现了大量不可抗拒的漏洞或者错误，很大一部分原因可能是需求分析和概念设计阶段没有做好有效的防护措施。在这次项目开发中吸收失败的经验和教训，在之后的应用项目开发中尽量减少类似的情况发生。

# 附录1 数据库JDBC连接接口与DAO层部署代码

## com/LuoZhiHao/controller/DatabaseController.java：

**public class** DatabaseController {  
 **private static** String *driver* = **"com.mysql.cj.jdbc.Driver"**;  
 **private static** String *url* = **"jdbc:mysql://localhost:3306/tv?characterEncoding=utf8&useSSL=false&serverTimezone = GMT"**;  
 **private static** String *username* = **"root"**;  
 **private static** String *password* = **"Qwe111.."**;  
 **private static** Connection *connection* = **null**;  
  
 *//获取数据库连接：Connection con = DatabaseController.getConnection;* **public static** Connection getConnection(){  
 **try** {  
 Class.*forName*(*driver*);  
 *connection* = DriverManager.*getConnection*(*url*,*username*,*password*);  
 } **catch** (SQLException e) {  
 e.printStackTrace();  
 System.***out***.println(**"数据库连接错误！"**);  
 } **catch** (ClassNotFoundException e) {  
 e.printStackTrace();  
 System.***out***.println(**"数据库连接驱动未找到！"**);  
 }  
 **return** *connection*;  
 }  
  
  
 **public static void** main(String[] args) {  
 *//测试数据库连接是否成功* Connection con = DatabaseController.*getConnection*();  
 System.***out***.println(**"数据库连接成功"**);  
 }  
}

## com/LuoZhiHao/utils/Database.java：

**public class** Database {  
 **private static** Connection *connection* = **null**;  
 **private static** Statement *statement* = **null**;  
 **private static** ResultSet *resultSet* = **null**;  
  
 **public static int** update(String sql) **throws** SQLException {  
 **int** count = 0;  
 *connection* = DatabaseController.*getConnection*();  
 *statement* = *connection*.createStatement();  
 count = *statement*.executeUpdate(sql);  
 *statement*.close();  
 *connection*.close();  
 **return** count;  
 }  
  
 **public static** ResultSet query(String sql) **throws** SQLException {  
 *connection* = DatabaseController.*getConnection*();  
 *statement* = *connection*.createStatement();  
 **return** *statement*.executeQuery(sql);  
 }  
  
 *//释放所有数据库连接* **public static void** releaseAll() **throws** SQLException {  
 **if**(*resultSet* != **null**){  
 *resultSet*.close();  
 *resultSet* = **null**;  
 }  
 **if** (*statement* != **null**){  
 *statement*.close();  
 *statement* = **null**;  
 }  
 **if** (*connection* != **null**){  
 *connection*.close();  
 *connection* = **null**;  
 }  
 System.***out***.println(**"已释放所有数据库连接！"**);  
 }  
}

8 参考文献

（参考文献书写注意以下几点：1 ***参考文献正文***内容用***5号宋体字***，书写格式按照国家标准GB7714-87规定。2要求参考文献数量在***10篇以上***）

参考文献是毕业设计中引用文献出处的目录表。参考文献一律放在文后，书写格式要按国家标准GB7714－87规定。

a. 专著、论文集、学位论文、报告

[序号] 著者. 文献题名[文献类型标识]. 出版地：出版者，出版年. 起止页码(任选).

[1] 刘国钧，陈绍业，王凤翥. 图书馆目录［M］. 北京：高等教育出版社，1957. 15-18.

[2] 辛希孟. 信息技术与信息服务国际研讨会论文集：A集［C］. 北京：中国社会科学出版社，1994.

[3] 张筑生. 微分半动力系统的不变集［D］. 北京：北京大学数学系数学研究所，1983.

[4] 冯西桥. 核反应堆压力管道与压力容器的LBB分析［R］. 北京：清华大学核能技术设计研究院，1997.

b. 期刊文章

[序号] 著者. 文献题名［J］. 刊名，年，卷(期)：起止页码.

[5] 何龄修. 读顾城《南明史》［J］. 中国史研究，1998，(3)：167-173.

[6] 金显贺，王昌长，王忠东，等. 一种用于在线检测局部放电的数字滤波技术［J］.清华大学学报(自然科学版)，1993，33(4)：62-67.

c. 电子文献

[序号] 著者. 电子文献题名［文献类型标识/载体类型标识］. 电子文献的出处或可获得地址，发表或更新日期/引用日期(任选).

[7]郭路. XML数据传输的安全加密［EB/OL］. http://www-900.ibm.com/developerWorks/cn/xml/ xmlb2b/index5.shtml, 2001-06/2001-10-04.

注1：参考文献类型

参考文献类型 专著 论文集 报纸文章 期刊文章 学位论文 报告 标准 专利

文献类型标识 M C N J D R S P

对于其他未说明的文献类型，建议采用单字母“Z”。

注2：电子文献类型

电子参考文献类型 数据库 计算机程序 电子公告

电子文献类型标识 DB CP EB

注3：电子载体类型

电子文献载体类型 磁带 磁盘 光盘 联机网络

电子载体类型标识 MT DK CD OL

如：[DB/OL]—联机网上数据库、[DB/MT]—磁带数据库、[M/CD]—光盘图书

[CP/DK]— 磁盘软件、[J/OL]— 网上期刊、[EB/OL]— 网上电子公告

在正文中引用参考文献：在引用处最后一个字的右上角，用方括号标明此序号。

如，“XML秘钥信息服务规范[1]”、“CAST对称秘钥算法[2],[4]”。