[1引言 2](#_Toc521464958)

[1.1编写目的 2](#_Toc521464959)

[1.2背景 2](#_Toc521464960)

[1.3定义 2](#_Toc521464961)

[1.4参考资料 2](#_Toc521464962)

[2总体设计 2](#_Toc521464963)

[2.1需求规定 2](#_Toc521464964)

[2.2运行环境 2](#_Toc521464965)

[2.3基本设计概念和处理流程 3](#_Toc521464966)

[2.4结构 3](#_Toc521464967)

[2.5功能器求与程序的关系 3](#_Toc521464968)

[2.6人工处理过程 3](#_Toc521464969)

[2.7尚未问决的问题 3](#_Toc521464970)

[3接口设计 3](#_Toc521464971)

[3.1用户接口 3](#_Toc521464972)

[3.2外部接口 3](#_Toc521464973)

[3.3内部接口 4](#_Toc521464974)

[4运行设计 4](#_Toc521464975)

[4.1运行模块组合 4](#_Toc521464976)

[4.2运行控制 4](#_Toc521464977)

[4.3运行时间 4](#_Toc521464978)

[5系统数据结构设计 4](#_Toc521464979)

[5.1逻辑结构设计要点 4](#_Toc521464980)

[5.2物理结构设计要点 4](#_Toc521464981)

[5.3数据结构与程序的关系 4](#_Toc521464982)

[6系统出错处理设计 5](#_Toc521464983)

[6.1出错信息 5](#_Toc521464984)

[6.2补救措施 5](#_Toc521464985)

[6.3系统维护设计 5](#_Toc521464986)

**概要设计说明书**

# 1引言

## 1.1编写目的

编写本文档是不仅是为了下一步系统详细设计打好基础而且是作为详细设计的重要依据。设计内容主要对系统的体系架构组成、系统内部交互、功能职责分配、系统外部接口及物理部署等。本文档的预期读者为项目组全体成员，以及日后维护本系统的开发人员。

## 1.2背景

本软件系统名称：学校宿舍管理系统；

本项目的任务提出者：黄志标；

开发者：黄志标、邬晓东、黄豪东；

用户：学校宿舍管理人员；

## 1.3定义

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
| 数据字典 | 指对数据的数据项、 数据结构、数据流、数据存储、处理逻辑、 外部实体等进行定义和描述, 其目的是对数据流程图中的各个元素做出详细的说明。 |
| spring | 一个开放源代码的设计层面框架,解决的是业务逻辑层和其他各层的松耦合问题，因此它将面向接口的编程思想贯穿整个系统应用。 |
| mybatis | 一款优秀的持久层框架，它支持定制化 SQL、存储过程以及高级映射。MyBatis 可以使用简单的 XML 或注解来配置和映射原生信息，将接口和 Java 的 POJOs(Plain Old Java Objects,普通的 Java对象)映射成数据库中的记录。 |
| spring MVC | Spring 框架提供的构建 Web 应用程序的全功能 MVC 模块,Spring MVC 分离了控制器、模型对象、过滤器以及处理程序对象的角色。 |
| Redis | 一个开源的使用ANSI C语言编写、支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API。 |

## 1.4参考资料

1. 《计算机软件文档编制规范》 GB-T8567-2006
2. 张海藩主编,《软件工程导论》,清华大学出版社

# 2总体设计

## 2.1需求规定

**1.主要用途：**

hostel（学校宿舍管理系统）可以帮助学校管理人员更好的管理学生住宿问题，可以了解每栋宿舍楼的住宿情况，甚至可以知道某个学生住在那间宿舍。

1. **主要功能：**

2.1 通过角色控制可以实现权限分配，每个角色都有相应的管理权限

2.2 学校宿舍管理员可以直观的了解每栋宿舍楼、宿舍的住宿情况

2.3 可以管理学生的入住与迁出，以及宿舍变动

2.4 实现学生，宿舍楼，宿舍，用户的增删改查，角色只实现查询功能

1. **输入输出要求**

学生

---输入的数据 : 学号，姓名，性别，电话号码，学院，班级，入学时间

---输出数据 : 学生信息, 入住宿舍信息, 修改密码信息。

宿舍楼

---输入的数据 : 宿舍楼名称，层数，每层间数，每间住多少人

---输出数据 : 宿舍楼信息, 该宿舍楼的所有宿舍信息

宿舍

---输入的数据 : 房间号

---输出数据 : 宿舍基本信息，该宿舍住的所有学生信息

用户

---输入的数据 : 用户名，联系电话，密码，性别，角色ID

---输出数据 : 用户基本信息，角色信息

角色

---输入的数据 : 无

---输出数据 : 角色信息

## 2.2运行环境

|  |  |
| --- | --- |
| 网络环境： | 局域网 |
| 硬件平台： | 开发用PC数台（P4 3.0以上CPU，40G以上硬盘，8G内存）； |
| 测试平台： | 服务端：  无  客户端：  个人PC,系统:Centos 7.3 |
| 操作系统平台： | 个人PC Win10/Centos 7.3；服务器 Centos 7.3 |
| 数据库平台： | Mysql5 |
| Web服务： | Tomcat8.0 |
| 内存数据库： | Redis |

## 2.3基本设计概念和处理流程

mysql

Redis

JavaWeb

用户A

用户B

用户.....

## 

## 2.4结构

## 

角色管理

角色查询

学生管理

学生查询

添加学生

修改学生信息

管理学生的入住状态

学校宿舍管理系统

用户管理

用户登陆、注销

修改密码

变更角色（权限）

用户查询

更新用户信息

宿舍楼管理

添加

软删除

查询

修改

宿舍管理

添加

修改

查询

软删除

## 2.5功能需求与程序的关系

本条用一张如下的矩阵图说明各项功能需求的实现同各块程序的分配关系：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 程序1 | 程序2 | …… | 程序n |
| 功能需求1 | √ |  |  |  |
| 功能需求2 |  | √ |  |  |
| …… |  |  |  |  |
| 功能需求n |  | √ |  | √ |

## 2.6人工处理过程

添加入住学生(添加)须手工录入

## 2.7尚未解决的问题

在当前的网络环境中,面临这各种的威胁, 病毒、密码、蠕虫的攻击, 系统可能受攻击,在安全性方面不够高。

# 3接口设计

## 3.1用户接口

学生信息系统采用的浏览器与服务器的形式,系统通过网页的形式为不同身份的用户提供录界面, 不同身份的用户通过自身的身份验证登录系统,实现自己权限范围内的各种操作.

## 3.2外部接口

学生信息管理系统, 以游览器与服务器的形式, 通过查询数据库服务器,查询相关信息并通过鼠标、键盘、显示器输入与输出相关信息。

## 3.3内部接口

学生信息管理系统的内部接口有:

角色管理，学生管理，宿舍楼管理，宿舍管理，用户管理.

# 4运行设计

## 4.1运行模块组合

<1>角色管理模块单独运行，只实现了查询功能.

<2>用户管理模块运行时与角色信息数据库关联，超级管理员可以修改用户的角色.

<3>宿舍楼信息管理模块运行时与宿舍信息模块关联，相应的管理员可以对宿舍楼，以及宿舍进行管理.

<4>学生管理模块运行与宿舍信息关联，可以通过宿舍查询该宿舍的所有入住学生信息，当然也可以通过主键，学号查询学生信息.

## 4.2运行控制

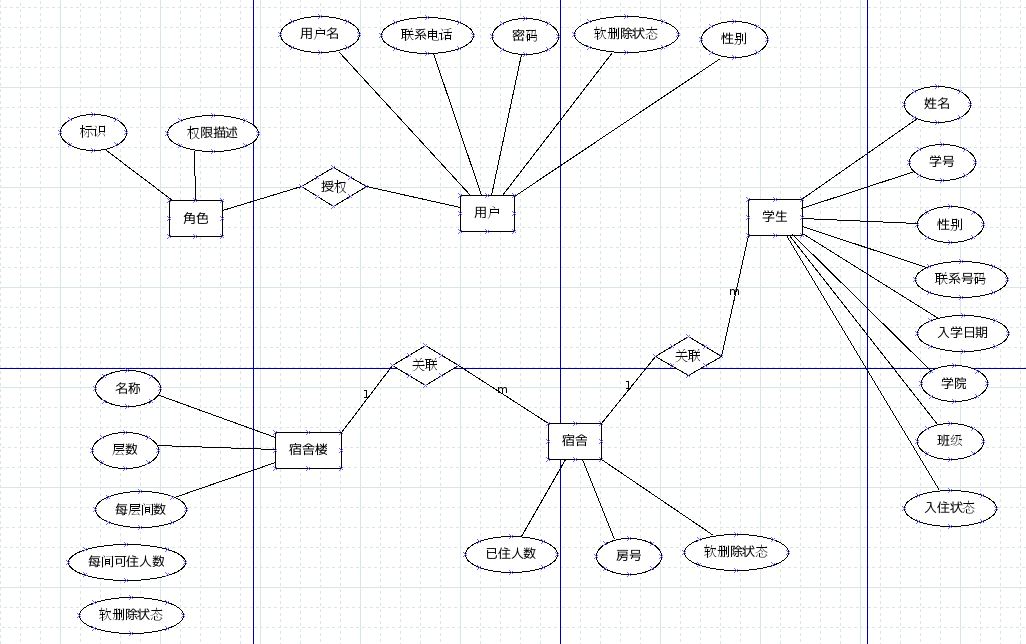
|  |  |  |
| --- | --- | --- |
| 角色管理 | 查询角色信息 | 只允许拥有root角色的用户查询 |
| 用户管理 | 查询，修改所有用户信息 | 只允许拥有root角色的用户查看 |
| 修改用户删除的状态 | 只允许拥有root角色的用户修改 |
| 修改用户自身的密码 | 用户登陆后才可以修改登陆密码 |
| 修改用户角色 | 只允许拥有root角色的用户修改 |
| 修改，查询用户自己的基本信息 | 登陆后才可以操作 |
| 宿舍楼管理 | 添加，修改，软删除 | 只允许拥有root或administrator角色的用户操作 |
| 查询 | 拥有root，administrator，guest角色的用户登陆后都可以查看 |
| 宿舍管理 | 添加，修改，软删除 | 只允许拥有root或administrator角色的用户操作 |
| 查询 | 拥有root，administrator，guest角色的用户登陆后都可以查看 |
| 学生管理 | 添加，修改，软删除 | 只允许拥有root或administrator角色的用户操作 |
| 查询 | 拥有root，administrator，guest角色的用户登陆后都可以查看 |

## 4.3运行时间

根据不同的硬件环境与操作系统, 系统的运行时间不确定, 当硬件达到要求配置时, 运行的时间很短。 系统运行查询占用的资源与时间最多,查询操作要与数据库的信息进行匹配,当数据库中的信息很多时,系统有一定的延迟。

# 5系统数据结构设计

## 5.1逻辑结构设计要点



## 

## 5.2物理结构设计要点

以下个模块访问数据库的方法和保密条件:

|  |  |  |  |
| --- | --- | --- | --- |
| 模块 | 功能 | 访问数据库方法 | 保密条件 |
| 登录模块 | 登录 | 直接访问 | 帐号与密码 |
| 角色管理 | 查看角色信息 | 查询数据库信息并显示 | 登陆 |
| 学生管理 | 查看学生信息 | 查询数据库信息并显示 | 登陆 |
| 宿舍楼管理 | 查看宿舍楼信息 | 查询数据库信息并显示 | 登陆 |
| 宿舍管理 | 查看宿舍信息 | 查询数据库信息并显示 | 登陆 |
| 用户管理 | 查看用户信息 | 查询数据库信息并显示 | 登陆 |
| 密码管理 | 用户密码管理 | 查询密码数据库 | 帐号与密码 |

## 5.3数据结构与程序的关系

以下是学校宿舍管理系统中所涉及到的信息表:

学生表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 属性名 | 字段名称 | 数据类型 | 长度 | 备注 |
| 1 | 主键 | stu\_id | int | 11 | 主键 |
| 2 | 学号 | stu\_number | int | 10 | 不空 |
| 3 | 姓名 | name | varchar | 10 | 不空 |
| 4 | 性别 | sex | tinyint | 1 | 不空 |
| 5 | 电话 | phone | varchar | 11 | 不空 |
| 6 | 学院 | major | varchar | 15 | 不空 |
| 7 | 班级 | class\_num | int | 5 | 不空 |
| 8 | 入学年份 | year | int | 4 | 不空 |
| 9 | 创建日期 | create\_time | timestamp |  | 不空 |
| 10 | 宿舍主键 | dormitory\_id | int | 11 | 不空 |
| 11 | 入住状态 | active | tinyint | 1 | 不空 |

宿舍楼表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 属性名 | 字段名称 | 数据类型 | 长度 | 备注 |
| 1 | 主键 | building\_id | int | 11 | 主键 |
| 2 | 宿舍楼名称 | building\_name | varchar | 10 | 不空 |
| 3 | 层数 | floors | int | 2 | 不空 |
| 4 | 每层间数 | rooms | int | 2 | 不空 |
| 5 | 每间可住人数 | lives | int | 2 | 不空 |
| 6 | 创建时间 | create\_time | timestamp |  | 不空 |
| 7 | 软删除状态 | active | tinyint | 2 | 不空 |

宿舍表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 属性名 | 字段名称 | 数据类型 | 长度 | 备注 |
| 1 | 主键 | dormitory\_id | int | 11 | 主键 |
| 2 | 房号 | dormitory\_number | int | 5 | 不空 |
| 3 | 软删除状态 | active | tinyint | 1 | 不空 |
| 4 | 宿舍楼主键 | building\_id | int | 11 | 不空 |
| 5 | 已住人数 | totals | tinyint | 2 | 不空 |

用户表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 属性名 | 字段名称 | 数据类型 | 长度 | 备注 |
| 1 | 主键 | user\_id | int | 11 | 主键 |
| 2 | 用户名 | user\_name | varchar | 10 | 不空 |
| 3 | 密码 | user\_pwd | varchar | 32 | 不空 |
| 4 | 性别 | user\_sex | tinyint | 1 | 不空 |
| 5 | 电话 | user\_phone | varchar | 11 | 不空 |
| 6 | 注册时间 | create\_time | timastamp |  | 不空 |
| 7 | 软删除状态 | active | tinyint | 1 | 不空 |
| 8 | 角色主键 | role\_id | int | 11 | 不空 |
| 9 | 用户审核状态 | user\_state | tinyint | 4 | 不空 |

角色表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 属性名 | 字段名称 | 数据类型 | 长度 | 备注 |
| 1 | 主键 | role\_id | int | 11 | 主键 |
| 2 | 角色标识 | symbol | varchar | 15 | 不空 |
| 3 | 权限描述 | description | varchar | 30 | 不空 |

# 6系统出错处理设计

## 6.1出错信息

用一览表的方式说朗每种可能的出错或故障情况出现时，系统输出信息的形式、含意及处理方法。

|  |  |  |  |
| --- | --- | --- | --- |
| 模块 | 出错原因 | 系统输出形式 | 处理方法 |
| 登录模块 | 登录时输入的帐号与密码不一致。 | 系统提示帐号与  密码错误。 | 弹出提示框 |
| 由于收网络因素影响,登录没有响应。 | 系统没有任何错  误提示信息。 | 刷新页面,重新登  录 |
| 角色管理 | 输入的查询条件有误 | 输出没有想要的  查询结果。 | 审查查询条件 |
| 学生管理 | 输入的查询条件有误 | 输出没有想要的  查询结果。 | 审查查询条件 |
| 宿舍楼管理 | 输入的查询条件有误 | 输出没有想要的  查询结果。 | 审查查询条件 |
| 宿舍管理 | 输入的查询条件有误 | 输出没有想要的  查询结果。 | 审查查询条件 |
| 用户管理 | 输入的查询条件有误 | 输出没有想要的  查询结果。 | 审查查询条件 |
| 密码管理 | 修改的密码两次不一致 | 系统提示两次密  码不一致 | 弹出提示框 |

## 6.2补救措施

1. 定期对系统进行更新,备份;

## 6.3系统维护设计

在整个系统的运行过程中, 系统的维护和管理是始终贯穿其中的,包括进行系统应用程序的维护、代码的维护、数据的备份与恢复、硬件设备维护。系统在运行时,也要随着环境的变化根据不同的需求及变化对系统进行必要的修改,使得系统功能更加完善。

系统维护工作贯穿于系统的整个运行过程中,包括:系统应用程序的维护、数据的维护、代码的维护、硬件设备维护,系统维护的重点是系统应用软件的维护工作。 而系统维护工作不应总是被动的等待用户提出要求后才进行,应进行主动的预防性维护。