



成	
绩	

廣東工業大學

## 数据库课程设计

课程名称 数据库系统

题目名称 电子资源管理系统

学生学院 计算机学院

专业班级 软件工程 1 班

学 号 3216004762

学生姓名 叶嘉怡

指导教师 左亚尧

2018 年 12 月



类别	评价标准					分数比例 (%)	成绩
课程设 计报告	论文结构包含： 1、 相关技术介绍 2、 需求分析 3、 概念结构 4、 逻辑结构 5、 物理结构 6、 数据库实施、系统测试方案和测试报告 7、 系统的主要功能和使用说明、系统安装说明。 要求论文完整、内容详细，格式规范。					30	
程序	1、 系统运行正确，界面友好； 2、 功能完善：有增、删、改、查功能；有基本的统计、报表功能； 有多表连接查询、自身连接查询、字符串匹配查询、模糊查 询、分组查询等； 3、 工作量充分； 4、 系统实现技术的难度和创新性。					30	
数据 库设 计	1、 E-R 图设计正确； 2、 数据库逻辑结构设计规范化； 3、 数据库物理结构设计合理。 4、 安全性设计：用到使用用户标识与鉴别，自主存取控制等技 术； 5、 完整性设计：用到各种完整性技术，例如触发子、约束等实现 实体完整性，参照完整性和各种用户定义的完整性。					40	
总评成绩	<input type="checkbox"/> 优	<input type="checkbox"/> 良	<input type="checkbox"/> 中	<input type="checkbox"/> 及格	<input type="checkbox"/> 不及格	总分	



# 目录

1 引言 .....	7
1.1 问题定义.....	7
1.2 设计背景.....	7
2 需求分析.....	9
2.1 功能需求.....	9
2.2 信息需求.....	9
2.3 逻辑模型.....	9
2.3.1 数据流图.....	9
2.3.1 数据字典.....	12
3 概念结构设计.....	15
3.1 部分实体图和 E-R 图 .....	15
3.1.1 实体图.....	15
3.1.2 完整的 E-R 图 .....	16
3.2 功能模块图.....	17
4 逻辑结构设计.....	19
4.1 关系模型.....	19
4.2 规范化分析.....	19
4.3 数据表设计.....	20
4.4 用户子模式.....	23
4.5 安全性设计.....	23
4.6 完整性设计.....	23
4.7 完整性实现.....	24
5 物理结构设计.....	25
4.1 存取方法.....	25
4.2 数据库存储结构.....	25
4.3 系统配置.....	25
6 数据库实施.....	27
6.1 创建数据库及数据对象.....	27
6.2 数据库备份和恢复方案.....	30
6.2.1 数据库备份方案.....	30
6.2.2 数据库恢复方案.....	31
7 详细设计及实现.....	33
7.1 模块详细设计.....	33
7.1.1 注册模块.....	33
7.1.2 登录模块.....	33
7.1.3 登出模块.....	34
7.1.4 上传文件模块.....	34
7.1.5 创建目录模块.....	35
7.1.6 新建/编辑文件模块.....	35
7.1.7 删除文件模块.....	36
7.1.8 文件分类模块.....	37
7.1.9 分享文件模块.....	38

7.1.10 查找所有子节点模块.....	39
7.2 系统实现界面.....	40
8 运行与维护.....	45
8.1 模块运行测试.....	45
8.1.1 获取某用户所有文件.....	45
8.1.2 获取用户磁盘使用情况.....	45
8.1.3 获取某用户目录树.....	46
8.1.4 上传文件.....	46
8.1.5 获取某用户回收站.....	47
8.2 维护.....	47
9 总结 .....	49
参考资料.....	49

# 1 引言

## 1.1 问题定义

随着计算机的日益普及和网络的发展,人们已经进入了信息时代,亦或是数字化时代。数据库的应用范围越来越广,数据库应用的功能也越来越强,因此编写管理信息系统应用程序也显得尤为重要,在强调管理,强调信息的现代社会中它变得越来越普及。

互联网就是一个免费分享的社区,在基于信息共享的理念上,如何快速、准确的获得信息也成为人们关注的关键问题。现今已存在许多电子资源管理系统,如网盘等,但是随着系统成熟,充斥着大量广告,也丢失了原有的简洁易用的特性。通过此次的数据库课程设计,能实现一个拥有基本功能的电子资源管理系统,延续其直观预览、分组管理、稳定安全等特性,同时能够向用户提供良好的交互性。熟悉背后的数据库设计方法,提高分析问题、解决问题和实践能力。

## 1.2 设计背景

本电子资源管理系统顺应时代背景,采用 B/S 架构设计,即浏览器和服务器架构模式。它是随着 Internet 技术的兴起,对 C/S 架构的一种变化或者改进的架构。在这种架构下,用户工作界面是通过浏览器来实现,极少部分事务逻辑在前端(Browser)实现,但是主要事务逻辑在服务器端(Server)实现,B/S 架构是 WEB 兴起后的一种网络架构模式,WEB 浏览器是客户端最主要的应用软件。这种模式统一了客户端,将系统功能实现的核心部分集中到服务器上,简化了系统的开发、维护和使用。

数据库管理系统 DBMS 选择 MySQL,MySQL 是一个关系型数据库管理系统,所使用的 SQL 语言是用于访问数据库的最常用标准化语言。由于其体积小、速度快、总体拥有成本低,一般中小型网站的开发都选择 MySQL 作为网站数据库。

对于应用开发环境和工具,选择 VSCode 完成前端页面的开发,使用 html,css,JavaScript 等语言,选择 PyCharm 作为后台开发的 IDE,语言选择 python,并使用开源的 Django 框架辅助。

本电子资源管理系统是根据一个局域网内共享数据资料和信息交互的需求为基础的一个项目开发,类似于一个网盘系统,可使团队间方便数据资料的共享和传输。





## 2 需求分析

### 2.1 功能需求

该电子资源管理系统应该具有如下基本功能：

- (1) 用户管理。在该管理模块中包含用户登录，信息验证和用户注册。管理员有权限可以管理所有普通用户以及所有文件和文件夹，有权更改用户权限。
- (2) 文件管理。即对文件和文件夹的管理，在该管理模块中包含文件创建，文件删除，文件编辑，文件共享等功能。普通用户只能够对自己的文件进行管理，而管理员拥有能管理所有用户文件的权限。
- (3) 目录管理。该功能模块实现掩护上传的文件在系统中已目录列表的形式显示在文件管理页面中，让用户更加方便的对自己的文件进行管理。
- (4) 消息管理。管理员有权限可以发送所有用户均可收到的系统消息。

### 2.2 信息需求

- 1) 用户信息：用户名，用户 id，用户密码，用户邮箱，用户权限级别，用户磁盘容量。
- 2) 文件信息：文件名，文件 id，文件类型，文件创建者，文件创建日期，文件大小，文件存储路径。
- 3) 文件夹信息：文件夹 id、文件夹名称、文件夹创建者 id、文件夹创建时间。
- 4) 用户文件信息：用户 id、文件 id、所在文件夹 id、拥有此文件日期、文件状态、权限（是否可读写）。
- 5) 用户文件夹信息：用户 id、文件夹 id、所在文件夹 id、拥有此文件夹日期、文件夹状态、权限（是否可读写）。

### 2.3 逻辑模型

#### 2.3.1 数据流图

数据流图如下。

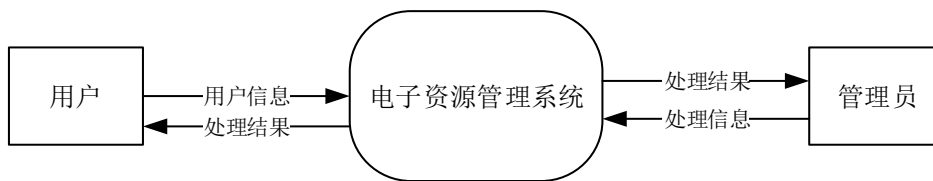


图 2-1 电子资源管理系统数据流图首层

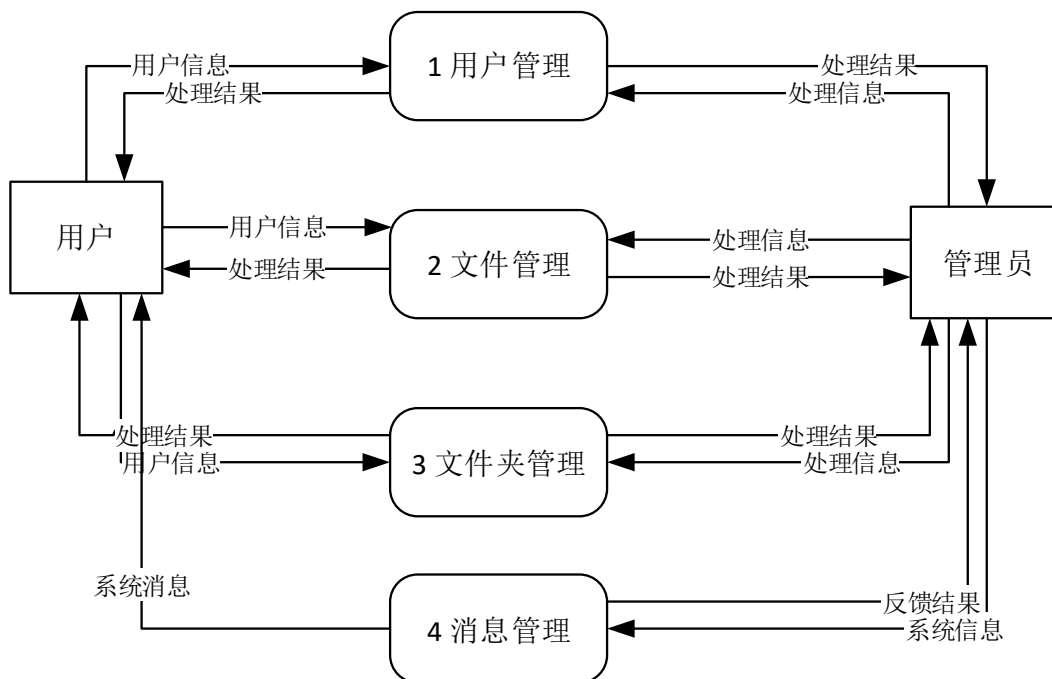
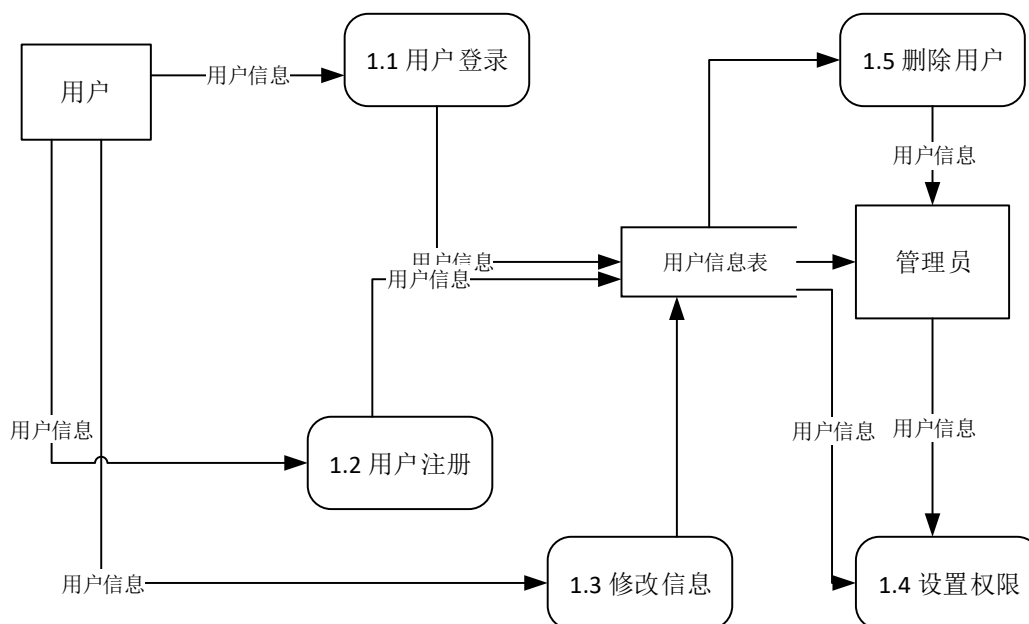


图 2-2 电子资源管理系统数据流图一层



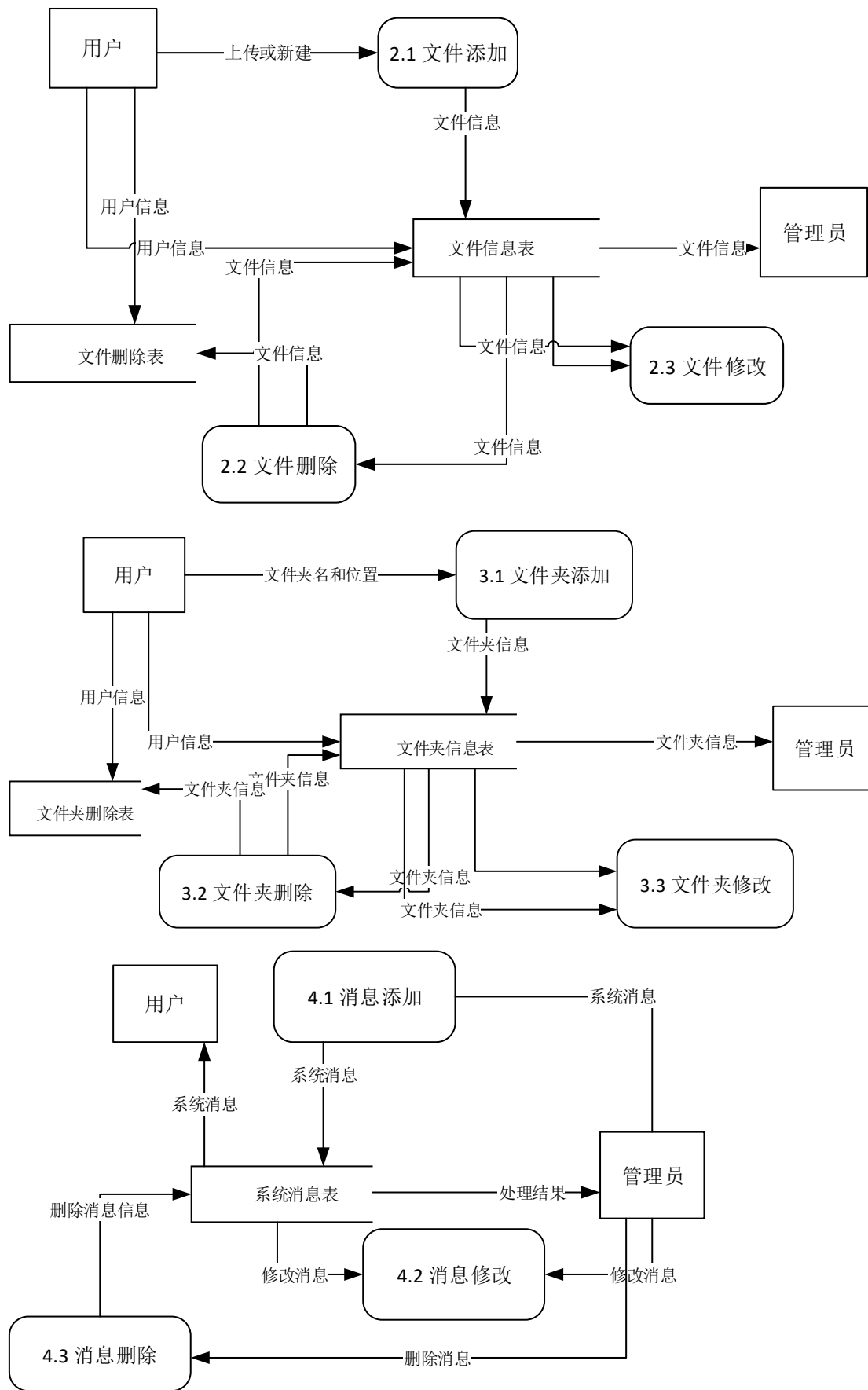


图 2-3 电子资源管理系统数据流图二层

## 2.3.1 数据字典

### 1、数据项

用户名、用户 id、密码、邮箱、磁盘容量、用户权限级别、用户权限 id、权限名称、文件 id、文件名称、类别、文件拓展名（类别）、文件创建者 id、文件创建时间、文件大小、文件存储路径、文件夹 id、文件夹名称、文件夹创建者 id、所在目录 id、文件夹创建时间、拥有此文件日期、文件状态、删除此文件日期、加好友日期、消息 id、消息内容、消息日期。

### 2、数据存储

#### 1) 用户信息

别名：用户

定义：用户信息={用户名+用户 id+密码+邮箱+磁盘容量+用户权限级别+用户权限 id}

来源：用户注册

#### 2) 文件信息

别名：文件

定义：文件信息={文件 id+文件名称+类别+文件拓展名（类别）+文件创建者 id+文件创建时间+文件大小+文件存储路径}

来源：新建或上传文件

#### 3) 文件夹信息

别名：文件夹

定义：文件夹信息={文件夹 id+文件夹名称+文件夹创建者 id+文件夹创建时间}

来源：新建文件夹

#### 4) 用户文件信息

别名：用户文件

定义：用户文件信息={用户 id+文件 id+拥有此文件日期+所在目录 id+文件状态+权限}

来源：新建或上传文件或被共享文件

#### 5) 用户文件夹信息

别名：文件夹

定义：用户文件夹信息={用户 id+文件 id+拥有此文件日期+所在目录+文件状态+权限}

来源：新建文件夹或被共享文件夹

### 6) 系统消息信息

别名：系统消息

定义：系统消息={消息 id+发送此系统消息的管理员 id+消息内容}

来源：管理员发送系统消息

## **3、数据处理：**

### 1) 用户信息处理

描述：查看用户信息、修改用户权限

输入数据流：用户信息

输出数据流：用户信息

### 1) 文件(夹)管理

描述：对文件(夹)的增删查改

输入数据流：文件信息、文件夹信息、用户信息

输出数据流：文件信息、文件夹信息、用户信息



## 3 概念结构设计

### 3.1 部分实体图和 E-R 图

#### 3.1.1 实体图

##### 1. 用户实体及其属性

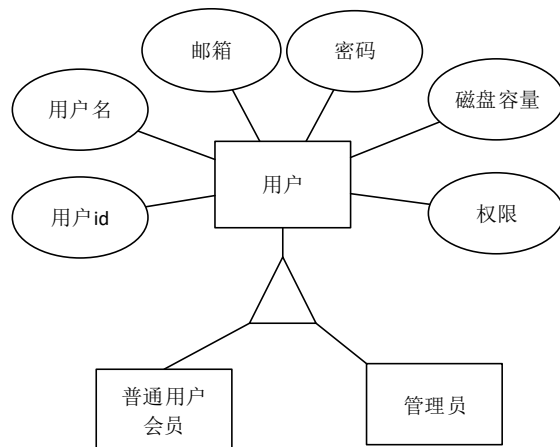


图 3-1 用户实体图

##### 2. 文件实体及其属性

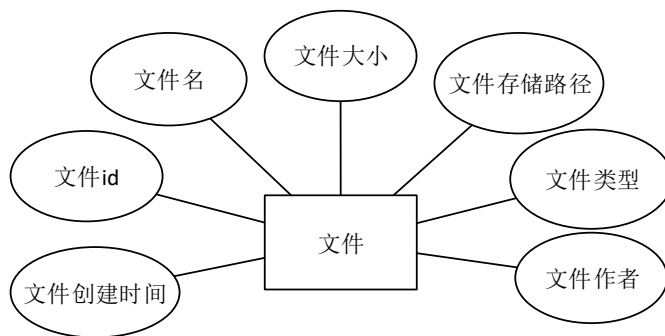


图 3-2 文件实体图

##### 3. 文件夹实体及其属性

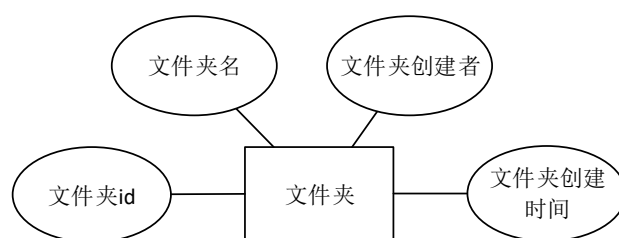


图 3-3 文件夹实体图

### 3. 消息实体及其属性

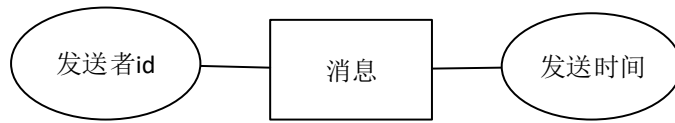


图 3-3 消息实体图

#### 3.1.2 完整的 E-R 图

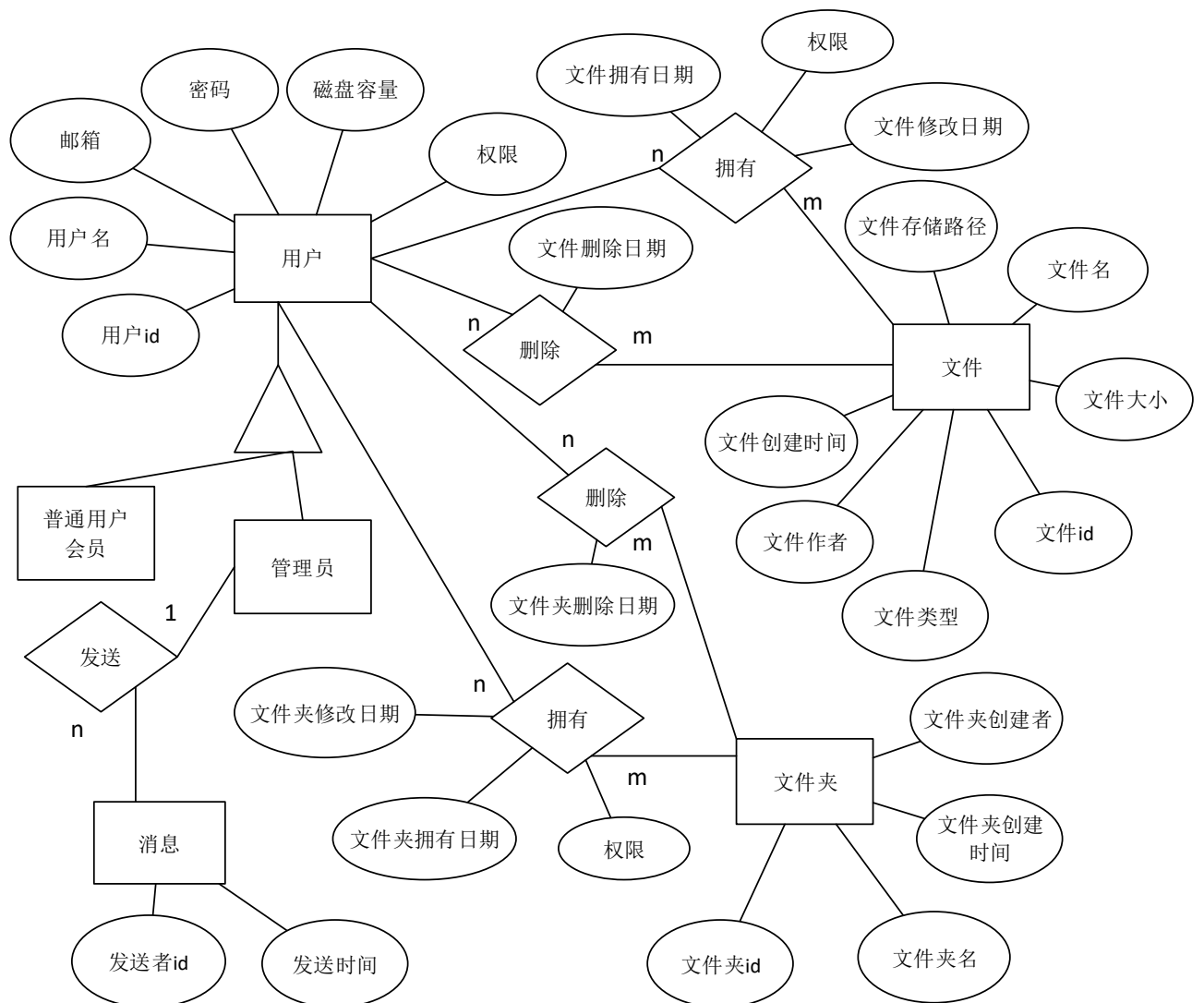


图 3-4 完整的实体联系图



3.2 类图

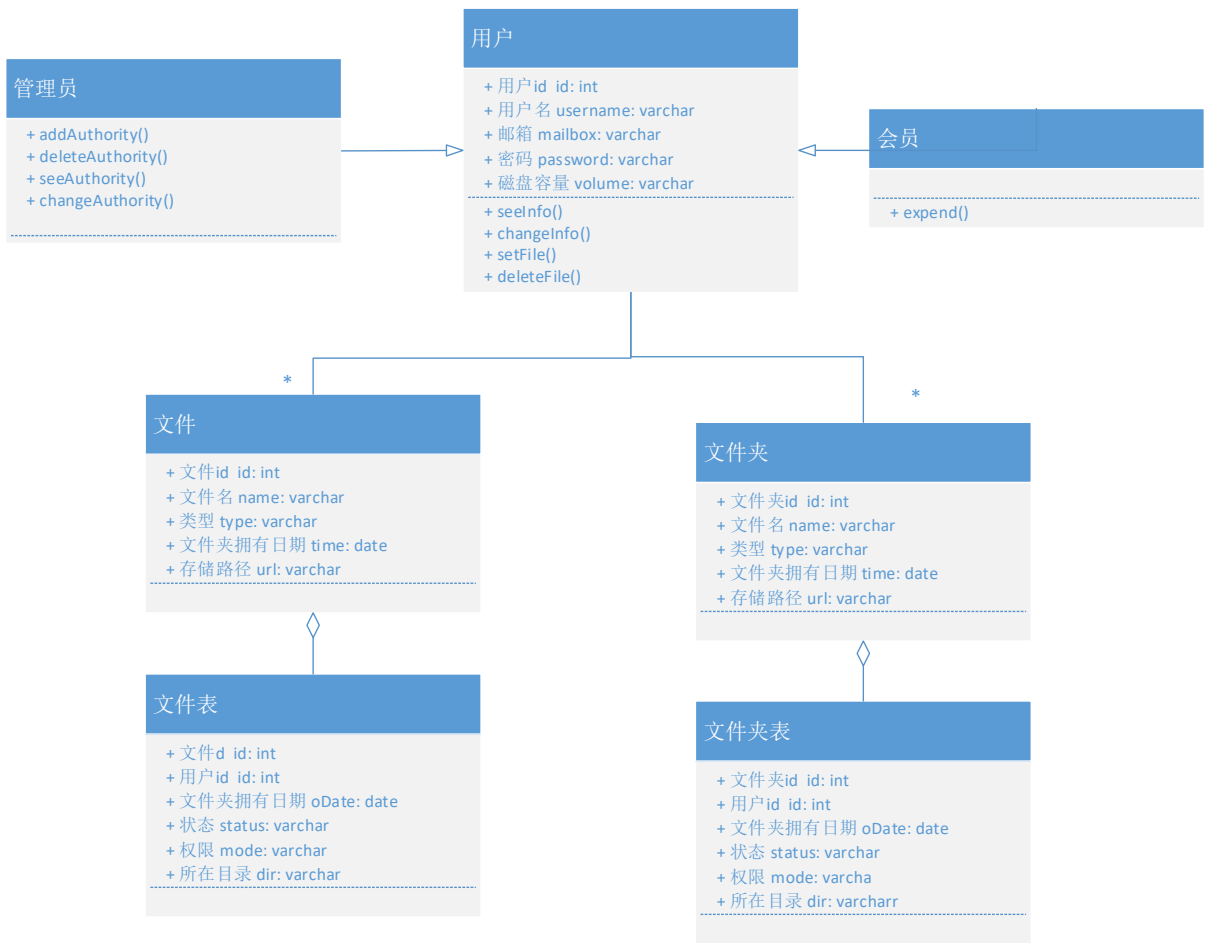


图 3-5 电子资源管理系统类图

### 3.3 功能模块图

系统的功能模块图如下。

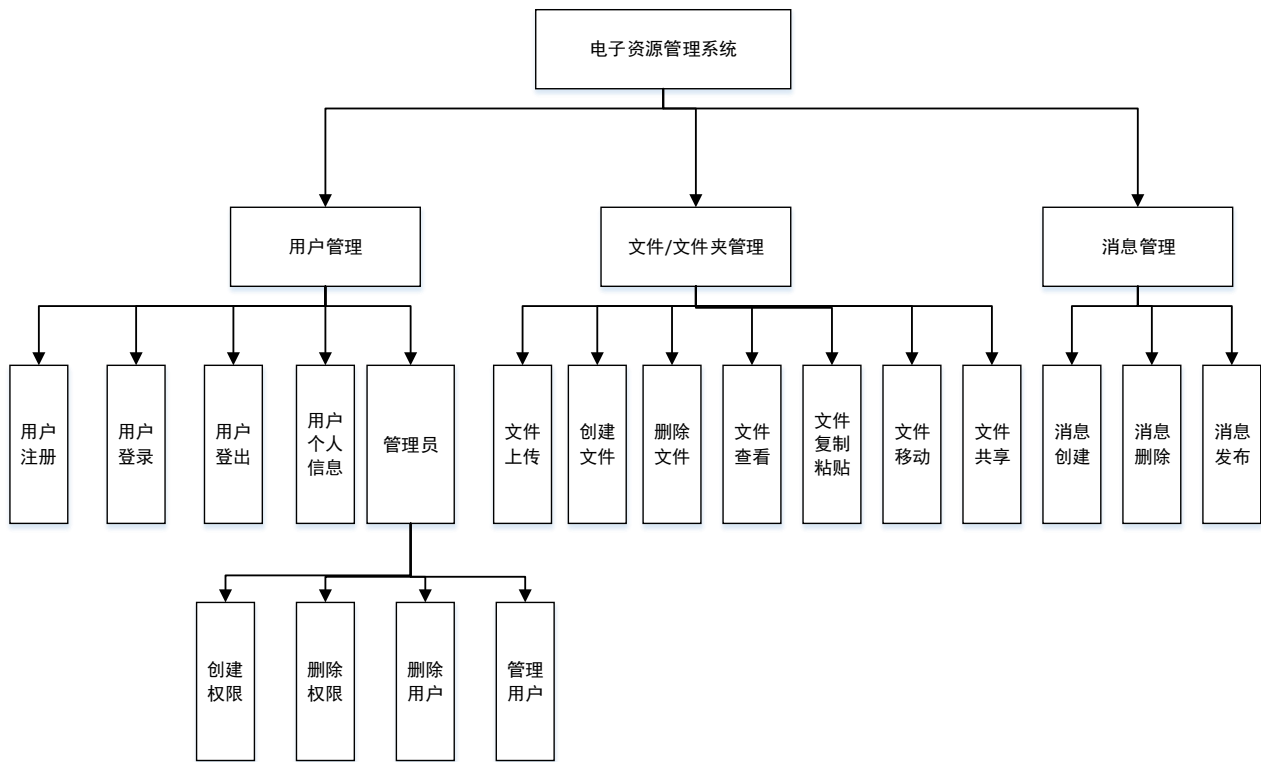


图 3-6 电子资源管理系统功能模块图

## 4 逻辑结构设计

### 4.1 关系模型

根据需求规格说明书中的 E-R 模型，实体转化为如下关系模式：

- 1) 用户信息 (用户 id、密码、邮箱、用户名、磁盘容量、用户权限级别)
- 2) 用户权限 (用户权限 id、权限名称)
- 3) 文件信息 (文件 id、文件名称、类别、文件拓展名 (类别)、文件创建者 id、文件创建时间、文件大小、文件存储路径)
- 4) 文件夹信息 (文件夹 id、文件夹名称、文件夹创建者 id、文件夹创建时间)
- 5) 用户-文件 (用户 id、文件 id、所在文件夹 id、拥有此文件日期、文件状态、权限)
- 6) 用户-文件夹 (用户 id、文件夹 id、所在文件夹 id、拥有此文件夹日期、文件夹状态、权限)
- 7) 文件删除 (用户 id、文件 id、文件删除日期)
- 8) 文件夹删除 (用户 id、文件夹 id、文件夹删除日期)
- 9) 系统消息 (消息 id、发送此系统消息的管理员 id、消息内容)
- 10) 用户-好友 (用户 id、好友 id、加好友日期)
- 11) 文件权限 (文件或文件夹权限 id、权限名)

### 4.2 规范化分析

经过对初始关系模式的规范化分析和处理，以下关系模式中已经不存在部分函数依赖和传递函数依赖，已经达到 3NF 模式。(粗体下划线为主码，斜体为外码)

- 1) sysite\_user (**id**, password, mailbox, name, volume, mode)
- 2) sysite\_usermode (**id**, uthority)
- 3) sysite\_file (**id**, name, type, author, time, size, url)
- 4) sysite\_folder (**id**, name, author, time)
- 5) sysite\_userfile (*userId*, *fileId*, dir, oDate, status, mode)
- 6) sysite\_userfolder (*userId*, *folderId*, dir, oDate, status, mode)

- 7) sysite\_deletedfile (userId, fileId, dDate)
- 8) sysite\_deletedfolder (userId, folderId, dDate)
- 9) sysite\_message (id, sender, content)
- 10) sysite\_userfriend (userId, friendId, time)
- 11) sysite\_mode (id, authority)

### 4.3 数据表设计

- 1) 用户信息表: 用户 id、密码、邮箱、用户名、磁盘容量、用户权限级别。

字段名	类型	长度	主键	外键	是否为空	说明
id	int	11	是	否	不允许	用户 id
password	varchar	30	否	否	不允许	密码
mailbox	varchar	30	否	否	允许	邮箱
name	varchar	255	否	否	不允许	用户名
volume	varchar	255	否	否	允许	磁盘容量
level	varchar	10	否	否	不允许	用户权限级别

- 2) 用户权限表: 用户权限 id、权限名。

字段名	类型	长度	主键	外键	是否为空	说明
id	int	11	是	否	不允许	权限 id
authority	varchar	5	否	否	不允许	权限名

- 3) 文件信息表: 文件 id、文件名称、类别、文件拓展名 (类别)、文件创建者 id、文件创建时间、文件大小、文件存储路径。

字段名	类型	长度	主键	外键	是否为空	说明
id	int	20	是	否	不允许	文件 id
name	varchar	20	否	否	不允许	文件名称
type	varchar	10	否	否	不允许	文件拓展名 (类别)
author	int	20	否	是	不允许	文件创建者 id

time	date	/	否	否	不允许	文件创建时间
size	varchar	10	否	否	不允许	文件大小
url	varchar	250	否	否	不允许	文件存储路径

4) 文件夹信息表: 文件夹 id、文件夹名称、文件夹创建者 id、文件夹创建时间。

字段名	类型	长度	主键	外键	是否为空	说明
id	int	20	是	否	不允许	用户 id
name	varchar	20	否	否	不允许	文件名称
author	varchar	20	否	是	不允许	文件创建者 id
time	date	/	否	否	不允许	文件创建时间

5) 用户文件表: 用户 id、文件 id、所在文件夹 id、拥有此文件日期、文件状态、权限 (是否可读写)。

字段名	类型	长度	主键	外键	是否为空	说明
fileId	int	11	是	是	不允许	文件 id
userId	int	11	是	是	不允许	用户 id
dir	int	11	否	是	不允许	所在文件夹 id
oDate	date	10	否	否	不允许	拥有此文件日期
stauts	varchar	/	否	否	不允许	文件状态
mode	varchar	5	否	否	不允许	权限

6) 用户文件夹表: 用户 id、文件夹 id、所在文件夹 id、拥有此文件夹日期、文件夹状态、权限 (是否可读写)。

字段名	类型	长度	主键	外键	是否为空	说明
folderId	int	11	是	是	不允许	文件夹 id
userId	int	11	是	是	不允许	用户 id
dir	int	11	否	否	不允许	所在文件夹 id
oDate	date	10	否	否	不允许	拥有此文件夹日期
stauts	varchar	/	否	否	不允许	文件夹状态

mode	varchar	5	否	否	不允许	权限
------	---------	---	---	---	-----	----

7) 文件删除表: 文件 id、用户 id、删除日期。

字段名	类型	长度	主键	外键	是否为空	说明
fileId	int	11	是	是	不允许	文件 id
userId	int	11	是	是	不允许	用户 id
Ddate	date	/	否	否	不允许	删除日期

8) 文件夹删除表: 文件夹 id、用户 id、删除日期。

字段名	类型	长度	主键	外键	是否为空	说明
folderId	int	11	是	是	不允许	文件夹 id
userId	int	11	是	是	不允许	用户 id
Ddate	date	/	否	否	不允许	删除日期

9) 系统消息表: 消息 id、发送此系统消息的管理员 id、消息内容。

字段名	类型	长度	主键	外键	是否为空	说明
id	int	20	是	否	不允许	消息 id
sender	varchar	20	否	是	不允许	发送消息管理员 id
content	varchar	250	否	否	不允许	系统消息的内容

10) 好友表: 用户 id、好友 id、加好友日期

字段名	类型	长度	主键	外键	是否为空	说明
userId	int	20	是	否	不允许	用户 id
friendId	varchar	20	否	是	不允许	好友 id
time	varchar	250	否	否	不允许	加好友日期

11) 文件权限表: 文件或文件夹权限 id、权限名。

字段名	类型	长度	主键	外键	是否为空	说明
id	int	11	是	否	不允许	权限 id

authority	varchar	5	否	否	不允许	权限名
-----------	---------	---	---	---	-----	-----

## 4.4 用户子模式

为用户建立以下视图便于操作。

- 1) 用户（用户 id、用户名、磁盘容量、用户权限级别）
- 2) 用户-文件（用户 id、文件 id、所在文件夹 id、拥有此文件日期、文件状态、权限）
- 3) 用户-文件夹（用户 id、文件夹 id、所在文件夹 id、拥有此文件夹日期、文件夹状态、权限）
- 4) 用户-好友（用户 id、好友 id、加好友日期）
- 5) 系统消息（消息 id、发送此系统消息的管理员 id、消息内容）

视图编号	视图名称	说明
V-1	V1_user	查询用户的基本信息
V-2	V2_userfile	查询用户所拥有文件的基本信息
V-3	V3_userfolder	查询用户所拥有文件夹的基本信息
V-4	V4_userfriends	查询用户好友
V-5	V5_message	查询系统消息

## 4.5 安全性设计

1. 用户标识与鉴别的基本方法，每次登陆时先识别用户身份，根据其身份确定权限再应用可执行的相应操作。
2. 确保只授权给有资格的用户访问数据库的权限，即存取控制。本系统采用自主存取控制（DAC）方法，同一用户对于不同的数据对象有不同的存取权限，不同的用户对同一对象也有不同的权限。

## 4.6 完整性设计

1. 实体完整性：通过设置各个表的主键都已实现。
2. 参照完整性：通过设置各个表的外键约束都已实现，违约处理采取默认策略即拒绝执行

(NO ACTION)。

3. 用户定义完整性：在某些表的某些属性上设置'NOT NULL'不允许空值。

## 4.7 完整性实现

### 1) 触发器的定义

定义一个 BEFORE 行级触发器，为用户表 sysite\_user 定义完整性规则“会员的磁盘存储容量 volume 不得低于 20000kb，如果低于 20000kb，自动改为 20000kb”。

```
CREATE TRIGGER Insert_Or_Update_Volume
    BEFORE INSERT OR UPDATE ON sysite_user
    /*触发事件是插入或更新操作*/
    FOR EACH ROW                                /*行级触发器*/
    AS BEGIN                                    /*定义触发动作体，是 PL/SQL 过程块*/
        IF (new.mode='2') AND (new.volume < 20000) THEN
            new.volume :=20000;
        END IF;
    END;
```

### 2) 存储过程

定义一个存储过程，在向数据库表 sysite\_user 插入数据时通过唯一索引去重。

```
BEGIN
    DECLARE v_volume INT DEFAULT 10000;
    DECLARE id INT(11) UNSIGNED;
    SET id = IF(LENGTH(TRIM(id))>0,CAST(id AS SIGNED),NULL);
    INSERT INTO sysite_user (name, password, mailbox, mode, volume)
    VALUES (%s,%s,%s,%s,%s)
    on duplicate key update
    name = %s, password = %s, mailbox = %s, mode = %s, volume = %s,
END
```



## 5 物理结构设计

### 4.1 存取方法

#### 1. 聚簇存取方法

由于用户文件表和文件表和用户表，用户文件夹表和文件表和用户表，需要经常进行连接操作，如果分散存放在多个不同物理块，操作效率会很低，所以考虑使用聚簇存取的方法来提高连接操作的效率，即把这几个表在磁盘空间允许的情况下存放在连续的物理块中。这样相当于把多个关系按“预连接”的形式存放，从而大大提高链接操作的效率。

#### 2. 建立索引

考虑在用户 id，文件 id 及文件夹 id 上建立索引，因为这些属性经常出现在连接操作的连接条件中出现，建立索引虽然会占用一定的空间，而且系统要为维护索引付出代价，但是为了提高效率还是值得的。

### 4.2 数据库存储结构

本系统将日志文件与数据库对象（表、索引等）放在不同的磁盘上，以改进系统性能。

### 4.3 系统配置

初始的部分系统配置如下（基于 MySQL）：

#### 1. `max_connect_errors = 10`

即 `mysqld` 线程没重新启动过，一台物理服务器只要连接 异常中断累计超过 10 次，就再也无法连接上 `mysqld` 服务。

#### 2. `interactive_timeout = 172800`

此参数是处于交互状态连接的活动被服务器端强制关闭而等待的时间。

#### 3. `innodb_buffer_pool_size`

开辟一片内存用于缓存 InnoDB 引擎表的数据和索引。



## 6 数据库实施

### 6.1 创建数据库及数据对象

#### 1. 数据库建立

```
CREATE DATABASE `file_system`;
```

#### 2. 数据表创建

1) 用户信息表 sysite\_user

```
CREATE TABLE `sysite_user` (  
    `id` int(11) NOT NULL AUTO_INCREMENT,  
    `mailbox` varchar(30) DEFAULT NULL,  
    `password` varchar(30) NOT NULL,  
    `name` varchar(255) DEFAULT NULL,  
    `volume` varchar(255) NOT NULL DEFAULT '10000',  
    `level` varchar(10) NOT NULL,  
    PRIMARY KEY (`id`)  
)
```

2) 用户权限表 sysite\_usermode

```
CREATE TABLE `sysite_usermode` (  
    `authority` varchar(5) NOT NULL,  
    `id` int(11) NOT NULL,  
    PRIMARY KEY (`id`),  
    FOREIGN KEY (`id`) REFERENCES `sysite_user`(`id`)  
)
```

3) 文件信息表 sysite\_file

```
CREATE TABLE `sysite_file` (  
    `id` int(11) NOT NULL AUTO_INCREMENT,
```

```
`name` varchar(20) NOT NULL,  
`type` varchar(10) NOT NULL,  
`author` int(20) NOT NULL,  
`time` date NOT NULL,  
`size` varchar(10) NOT NULL,  
`url` varchar(250) NOT NULL,  
PRIMARY KEY (`id`)  
)
```

4) 文件夹信息表 sysite\_folder

```
CREATE TABLE `sysite_folder` (  
  `id` int(20) NOT NULL AUTO_INCREMENT,  
  `name` varchar(20) NOT NULL,  
  `author` varchar(20) NOT NULL,  
  `time` date NOT NULL,  
  PRIMARY KEY (`id`)  
)
```

5) 用户文件表 sysite\_fileuser

```
CREATE TABLE `sysite_fileuser` (  
  `userId` int(11) NOT NULL,  
  `fileId` int(11) NOT NULL,  
  `dir` int(11) NOT NULL,  
  `status` varchar(10) NOT NULL DEFAULT '0',  
  `oDate` date NOT NULL,  
  `mode` varchar(5) NOT NULL DEFAULT '0',  
  PRIMARY KEY (`userId`, `fileId`)  
  FOREIGN KEY (`userId`) REFERENCES `sysite_file`(`id`)  
  FOREIGN KEY (`fileId`) REFERENCES `sysite_folder`(`id`)  
)
```

6) 用户文件夹表 sysite\_folderuse

```
CREATE TABLE `sysite_folderuser` (  
  `userId` int(11) NOT NULL,  
  `folderId` int(11) NOT NULL,  
  `dir` int(11) NOT NULL,  
  `status` varchar(10) NOT NULL DEFAULT '0',  
  `oDate` date DEFAULT NULL,  
  `mode` varchar(5) NOT NULL DEFAULT '0',  
  PRIMARY KEY (`userId`,`folderId`)  
  FOREIGN KEY (`userId`) REFERENCES `sysite_file`(`id`)  
  FOREIGN KEY (`folderId`) REFERENCES `sysite_folder`(`id`)  
)
```

7) 文件删除表 sysite\_deletedfile

```
CREATE TABLE `sysite_deletedfile` (  
  `userId` int(11) NOT NULL,  
  `fileId` int(11) NOT NULL,  
  `Ddate` date NOT NULL,  
  PRIMARY KEY (`userId`,`fileId`)  
  FOREIGN KEY (`userId`) REFERENCES `sysite_file`(`id`)  
  FOREIGN KEY (`fileId`) REFERENCES `sysite_folder`(`id`)  
)
```

8) 文件夹删除表 sysite\_deletedfolder

```
CREATE TABLE `sysite_deletedfolder` (  
  `userId` int(11) NOT NULL,  
  `folderId` int(11) NOT NULL,  
  `Ddate` date NOT NULL,  
  PRIMARY KEY (`userId`,`folderId`)
```

```
FOREIGN KEY (`userId`) REFERENCES `sysite_file`(`id`)
FOREIGN KEY (`folderId`) REFERENCES `sysite_folder`(`id`)
)
```

#### 9) 系统消息表

```
CREATE TABLE `sysite_message` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `sender` int(11) DEFAULT NULL,
  `content` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
)
```

#### 10) 用户好友表

```
CREATE TABLE `sysite_userfriend` (
  `userId` int(11) NOT NULL,
  `friendId` int(11) NOT NULL,
  `time` date NOT NULL,
)
```

#### 11) 文件权限表

```
CREATE TABLE `sysite_mode` (
  `authority` varchar(5) NOT NULL,
  `id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (`id`) REFERENCES `sysite_file`(`id`)
)
```

## 6.2 数据库备份和恢复方案

### 6.2.1 数据库备份方案

由于本系统选用的DBMS为MySQL,查阅资料得到在MySQL下备份的集中常用工具。由于本系统设计的数据量比较少,所以计划采用 `mysqldump` 工具,每周执行一次全量备份,

每天执行一次增量备份。备份计划如下：

1. 将数据和备份放在不同的磁盘设备上，异机或异地备份存储较为理想。
2. 周期性地对备份的数据进行还原测试。
3. 每次灾难恢复后都应该立即做一次完全备份。
4. 针对不同规模或级别的数据量，要定制好备份策略。
5. 二进制日志应该跟数据文件在不同磁盘上，并周期性地备份好二进制日志文件。

### 6.2.2 数据库恢复方案

每周进行一次恢复测试，主要在测试机上进行，模拟某个时间点主机数据全部丢失，要求恢复到丢失时间点的所有数据，先进行全备恢复，然后根据 `mysqldump` 恢复到最近时间点，具体恢复步骤如下：

1. 停止 MySQL 服务器。
2. 记录服务器的配置和文件权限。
3. 将数据从备份移到 MySQL 数据目录；其执行方式依赖于工具。
4. 改变配置和文件权限。
5. 以限制访问模式重启服务器；`mysqld` 的 `--skip-networking` 选项可跳过网络功能。
6. 载入逻辑备份，检查和重放二进制日志。
7. 检查已经还原的数据。
8. 重新以完全访问模式重启服务器。





# 7 详细设计及实现

## 7.1 模块详细设计

### 7.1.1 注册模块

```
@csrf_exempt
def register(request):
    if request.method == 'POST':
        name = request.POST.get('username')
        mailbox = request.POST.get('email')
        password = request.POST.get('password')
        db = MysqlHelper()
        sql = "INSERT INTO sysite_user (name, password, mailbox) VALUES (%s, %s, %s)"
        params = [name, password, mailbox]
        res = db.insert(sql, params)
        data = 'success'
    return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')
```

### 7.1.2 登录模块

```
@csrf_exempt
def login(request):
    if request.method == 'GET':
        return render(request, 'static/html/login.html')
    else:
        username = request.POST.get('username')
        password = request.POST.get('password')
        db = MysqlHelper()
        sql = "SELECT * FROM sysite_user WHERE name = %s and password = %s"
        params = [username, password]
        res = db.fetchone(sql, params)
        if res:
            data = {'result': 'success', 'level': res[5]}
            request.session['id'] = res[0]
            request.session['name'] = res[3]
            request.session['level'] = res[5]
        else:
            data = "error"
    return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')
```

### 7.1.3 登出模块

```
@csrf_exempt
def logout(request):
    del request.session["id"]
    del request.session["name"]
    return render(request, 'static/html/login.html')
```

### 7.1.4 上传文件模块

```
@csrf_exempt
def upload(request):
    base_url = 'media/' + str(request.session['id']) + '/'
    if not os.path.exists(base_url):
        os.makedirs(base_url)
    if request.method == 'POST':
        file = request.FILES.get('file') # 获取文件对象，包括文件名文件大小和文件内容
        dir = request.POST.get('dir')
        fileName = file.name
        prefix = re.findall(r'(.+?)\.', fileName)[0] # 正则表达式获取前缀
        type = os.path.splitext(fileName)[-1][1:] # python 获取后缀
        date = time.strftime('%Y-%m-%d', time.localtime(time.time()))
        # 写入文件
        with open(base_url + file.name, 'wb+') as f:
            for chunk in file.chunks():
                f.write(chunk)
        size = formatSize(file.size)
        db = MySQLHelper()
        sql1 = "INSERT INTO sysite_file (name, type, author, time, size, url) VALUES"
        (%s, %s, %s, %s, %s, %s)"
        params = [prefix, type, request.session['id'], date, size, base_url+file.name]
        res = db.insert(sql1, params)
        db = MySQLHelper()
        sql3 = "SELECT max(id) from sysite_file"
        new_id = db.fetchone(sql3, [])[0]
        # 插入关联表
        db = MySQLHelper()
        sql2 = "INSERT INTO sysite_fileuser (userId, fileId, dir, oDate) VALUES (%s, %s, %s, %s)"
        params = [request.session['id'], new_id, dir, date]
        res2 = db.insert(sql2, params)
        data = { "result": "success", "detail": { "name": prefix, "type": type, "owner": request.session['id'], "time":
date, "size": file.size} }
        return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')
```

### 7.1.5 创建目录模块

```
@csrf_exempt
def newFolder(request):
    if request.method == 'POST':
        name = request.POST.get('name')
        owner = request.POST.get('owner')
        dir = request.POST.get('parent_dir')
        date = time.strftime('%Y-%m-%d', time.localtime(time.time()))
        db = MysqlHelper()
        sql1 = "INSERT INTO sysite_folder (name, author, time) VALUES (%s, %s, %s)"
        params = [name, request.session['id'], date]
        res = db.insert(sql1, params)
        db = MysqlHelper()
        sql3 = "SELECT max(id) from sysite_folder"
        new_id = db.fetchone(sql3, [])[0]
        # 插入关联表
        db = MysqlHelper()
        sql2 = "INSERT INTO sysite_folderuser (userId, folderId, dir, oDate) VALUES (%s, %s, %s, %s)"
        params = [request.session['id'], new_id, dir, date]
        res2 = db.insert(sql2, params)
        data = {"result": "success",
                "detail": {"name": name, "type": type, "owner": request.session['id'], "time": date}}
        return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')
```

### 7.1.6 新建/编辑文件模块

```
@csrf_exempt
def newOrEdit(request):
    base_url = 'media/' + str(request.session['id']) + '/'
    if request.method == 'POST':
        id = request.POST.get('id')
        name = request.POST.get('name')
        content = request.POST.get('content')
        dir = request.POST.get("dir")
        date = time.strftime('%Y-%m-%d', time.localtime(time.time()))
        db = MysqlHelper()
        sql = "SELECT * FROM sysite_file WHERE id = %s"
        params = [id]
        res = db.fetchone(sql, params)
        url = base_url + name + '.txt'
        # 编辑则重新写入文件，新建则新建文件
        with open(base_url + name + '.txt', 'w+') as f:
```

```

        f.write(content)
size = getFileSize(base_url + name + '.txt')
if res:  # 编辑原有文件，需要改变大小
    db = MysqlHelper()
    sql3 = "UPDATE sysite_file SET name = %s, size = %s WHERE id = %s"
    db.update(sql3, [name, size, id])
    data = {"result": "success", "message": "编辑成功"}
else:  # 新建 txt 文件
    db = MysqlHelper()
    sql1 = "INSERT INTO sysite_file (name, type, author, time, size, url) VALUES
(%s, %s, %s, %s, %s, %s)"
    params = [name, "txt", request.session['id'], date, size, url]
    res = db.insert(sql1, params)
    db = MysqlHelper()
    sql3 = "SELECT max(id) from sysite_file"
    new_id = db.fetchone(sql3, [])[0]
    db = MysqlHelper()
    sql2 = "INSERT INTO sysite_fileuser (userId, fileId, dir, oDate) VALUES (%s, %s, %s, %s)"
    params = [request.session['id'], new_id, dir, date]
    res2 = db.insert(sql2, params)
    data = {"result": "success", "message": "新建成功"}
return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')

```

### 7.1.7 删除文件模块

```

@csrf_exempt
def delete(request):
    if request.method == 'POST':
        folderList = json.loads(request.POST.get('folderList'))
        fileList = json.loads(request.POST.get('fileList'))
        date = time.strftime("%Y-%m-%d", time.localtime(time.time()))
        if folderList:
            for i, j in enumerate(folderList):  # 所有文件夹的子节点也要一并删除
                list1, list2 = getChildren(request.session["id"], j)
                if list1:
                    for k in list1:
                        folderList.append(k)
                if list2:
                    for k in list2:
                        fileList.append(k)
            for i, v in enumerate(folderList):
                db = MysqlHelper()
                sql1 = "INSERT INTO sysite_deletedfolder (userId, folderId, Ddate) VALUES
(%s, %s, %s)"

```

```

        res1 = db.insert(sql1, [request.session["id"], v, date])
        db = MysqlHelper()
        sql2 = "UPDATE sysite_folderuser SET status = %s WHERE folderId = %s"
        db.update(sql2, [1, int(v)])

    if fileList:
        for i, v in enumerate(fileList):
            db = MysqlHelper()
            sql = "INSERT INTO sysite_deletedfile (userId, fileId, Ddate) VALUES (%s, %s, %s)"
            res = db.insert(sql, [request.session["id"], v, date])
            db = MysqlHelper()
            sql3 = "UPDATE sysite_fileuser SET status = %s WHERE fileId = %s"
            db.update(sql3, [1, int(v)])

    data = 'success'
    return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')

```

### 7.1.8 文件分类模块

```

# 获取所有文件类型
@csrf_exempt
def getType(request):
    if request.method == 'GET':
        db = MysqlHelper()
        sql = "SELECT type, count(*) FROM sysite_file, sysite_fileuser WHERE userId = %s and fileId = %s GROUP BY type"
        res = db.fetchall(sql, [request.session["id"]])
        if res:
            data = {"result": "success", "message": {}}
            for i, v in enumerate(res):
                data["message"][i] = {"type": v[0], "count": v[1]}
            print(data)
        else:
            data = {"result": "success", "message": "None"}
    return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')

# 根据类型获取某用户文件
@csrf_exempt
def getFileByType(request):
    if request.method == 'GET':
        type = request.GET.get("type")
        if type == "All":
            db = MysqlHelper()
            sql = "SELECT sysite_file.id, sysite_file.name, type, author, time, size , url, " \
                  "sysite_user.name, authority " \
                  "FROM sysite_file, sysite_fileuser, sysite_user, sysite_mode " \

```

```

        "WHERE status != 1 and userId = %s and sysite_file.id = sysite_fileuser.fileId " \
        "and sysite_file.author = sysite_user.id and sysite_mode.id = sysite_fileuser.mode"
    res = db.fetchall(sql, [request.session["id"]])
else:
    db = MysqlHelper()
    sql = "SELECT sysite_file.id, sysite_file.name, type, author, time, size , url, " \
        "sysite_user.name, authority " \
        "FROM sysite_file, sysite_fileuser, sysite_user, sysite_mode " \
        "WHERE status != 1 and userId = %s and sysite_file.id = sysite_fileuser.fileId" \
        "and type = %s and sysite_file.author = sysite_user.id and sysite_mode.id = " \
        "sysite_fileuser.mode "
    res = db.fetchall(sql, [request.session["id"], type])
if res:
    data = {"result": "success", "message": {}}
    for i, v in enumerate(res):
        data["message"][i] = {'id': v[0], 'name': v[1], 'type': v[2], 'author': v[7], 'time':
            v[4].isoformat(), 'size': v[5], 'url': v[6], 'mode': v[8], 'authorId': v[3]}
else:
    data = {"result": "None"}
return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')

```

### 7.1.9 分享文件模块

```

@csrf_exempt
def share(request):
    if request.method == 'POST':
        shareTo = request.POST.get('shareTo')
        folderList = json.loads(request.POST.get('folderList'))
        fileList = json.loads(request.POST.get('fileList'))
        date = time.strftime('%Y-%m-%d', time.localtime(time.time()))
        if folderList:
            for i, j in enumerate(folderList):
                # 所有文件夹的子节点也要一并分享
                list1, list2 = getChildren(request.session["id"], j)
                if list1:
                    for k in list1:
                        folderList.append(k)
                if list2:
                    for k in list2:
                        fileList.append(k)
        print("folder:", folderList)
        for i, v in enumerate(folderList):
            db = MysqlHelper()
            sql1 = "INSERT INTO sysite_folderuser (userId, folderId, dir, oDate) VALUES

```

```

(%s, %s, %s, %s)"
        res1 = db.insert(sql1, [shareTo, v, shareTo, date])
    if fileList:
        for i, v in enumerate(fileList):
            db = MysqlHelper()
            sql = "INSERT INTO sysite_fileuser (userId, fileId, dir, oDate) VALUES
(%s, %s, %s, %s)"
            res = db.insert(sql, [shareTo, v, shareTo, date])
        data = 'success'
    return HttpResponse(json.dumps(data, cls=JSONEncoder), content_type='application/json')

```

### 7.1.10 查找所有子节点模块

```

def getChildren(id, pid):
    list1 = []
    list2 = []
    db = MysqlHelper()
    sql = "select folderId from (select t1.folderId, if(find_in_set(dir, @pids) > 0, @pids := concat(@pids, ',',
        folderId), 0) " \
        "as ischild from (select folderId, dir from sysite_folderuser t where t.status = 0 and t.userId
        = %s " \ "order by dir, folderId) t1, (select @pids := %s) t2) t3 where ischild != 0" \
    params = [id, pid]
    res = db.fetchall(sql, params)
    if res:
        for i in res:
            list1.append(i[0])
    db = MysqlHelper()
    sql1 = "select fileId from (select t1.fileId, if(find_in_set(dir, @pids) > 0, @pids := concat(@pids, ',',
        fileId), 0)
        "as ischild from (select fileId, dir from sysite_fileuser t where t.status = 0 and t.userId = %s
        order by dir, fileId) t1, " \
        "(select @pids := %s) t2) t3 where ischild != 0"
    params = [id, pid]
    res1 = db.fetchall(sql1, params)
    if res1:
        for i in res1:
            list2.append(i[0])
    print("list2:", list2)
    return list1, list2

```

7.2 系统实现界面

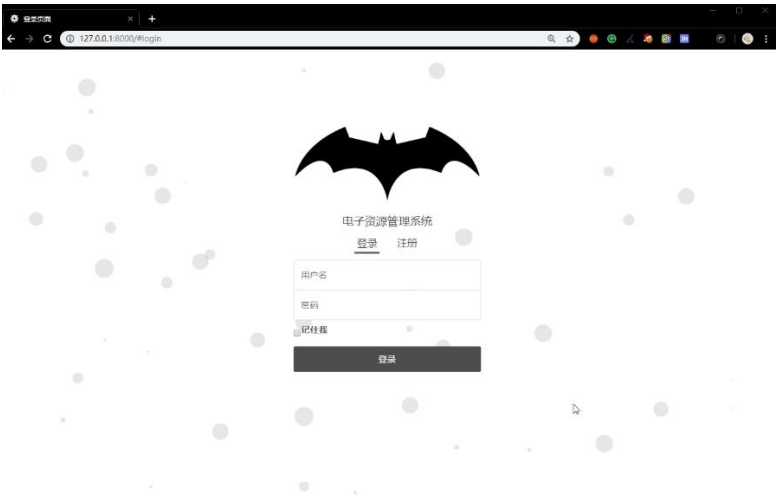


图 7-1 用户登录页面

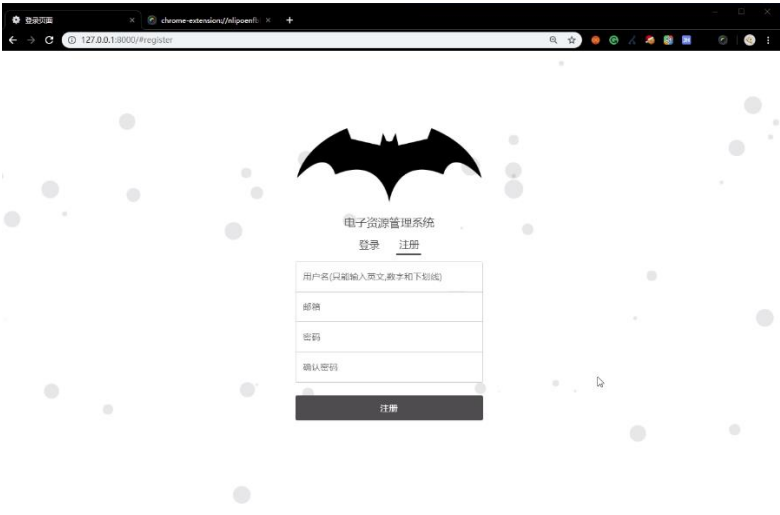


图 7-2 用户注册页面

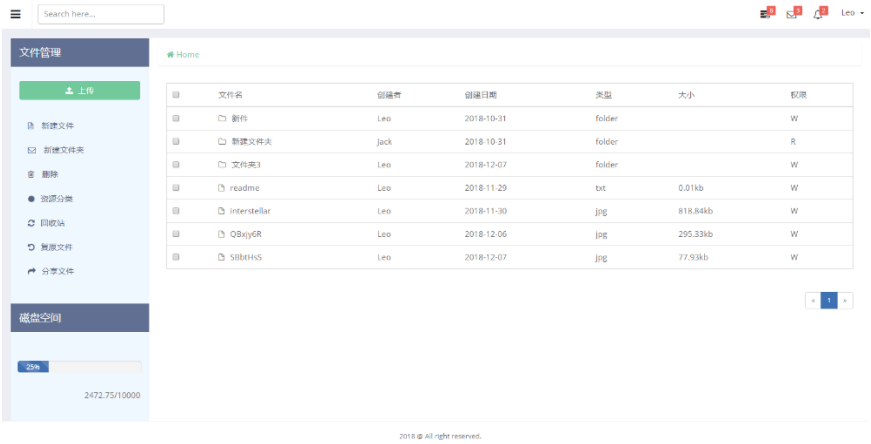


图 7-3 用户主页面



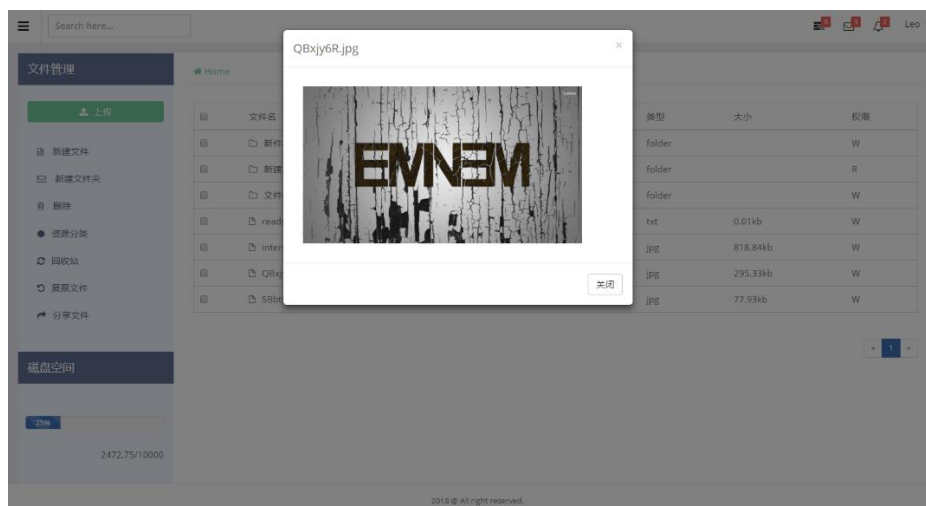


图 7-4 查看图片文件

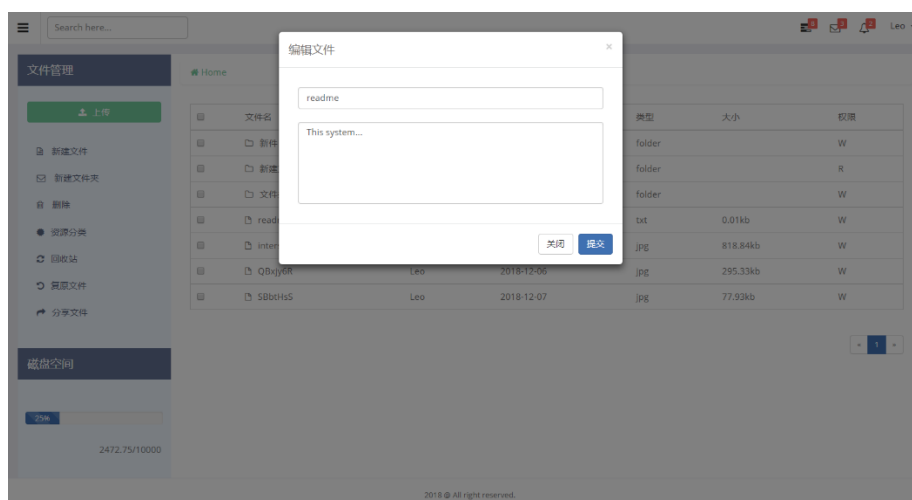


图 7-5 查看或编辑文本文件

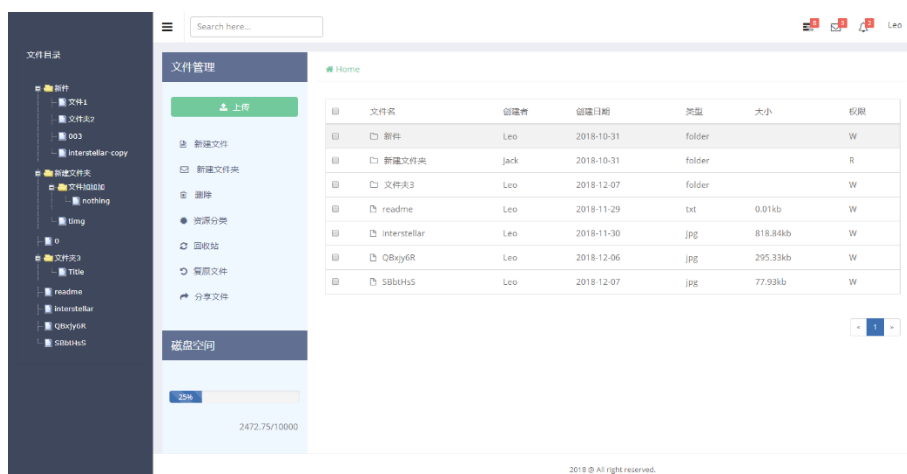


图 7-6 查看目录

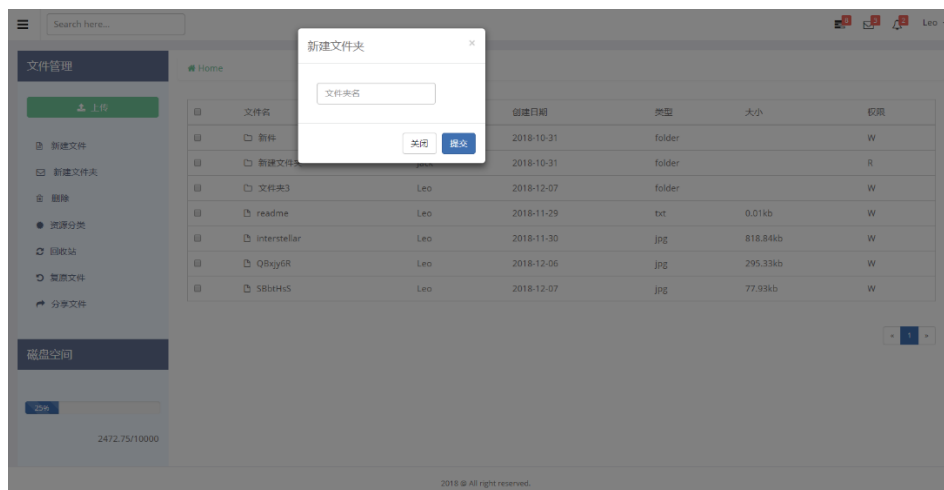


图 7-7 新建文件夹

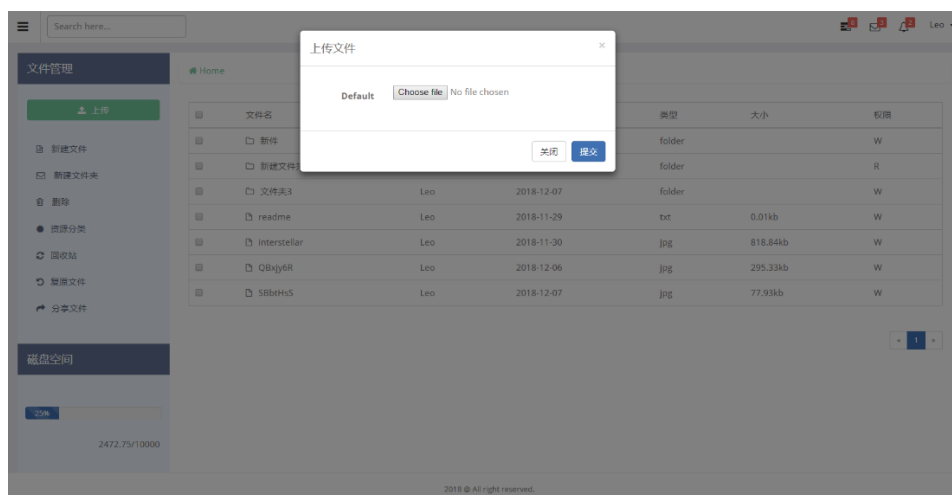


图 7-8 上传文件

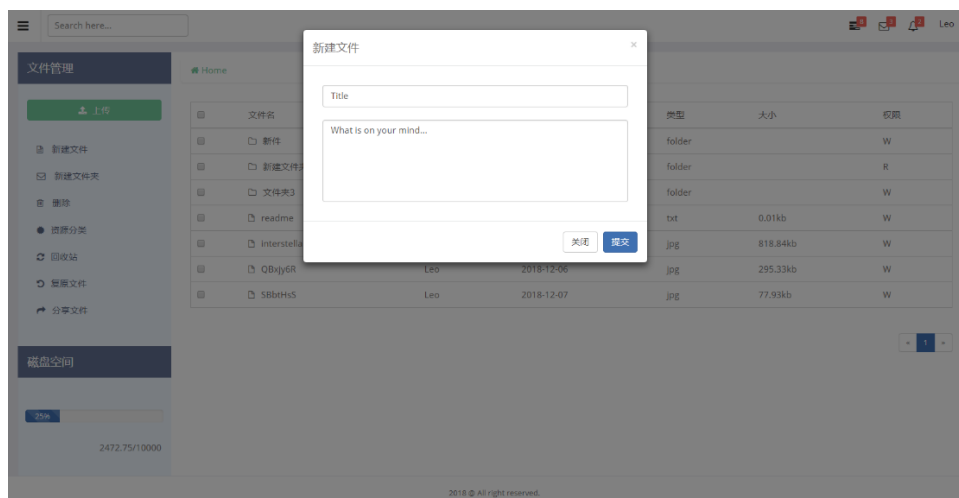


图 7-9 新建文本文件

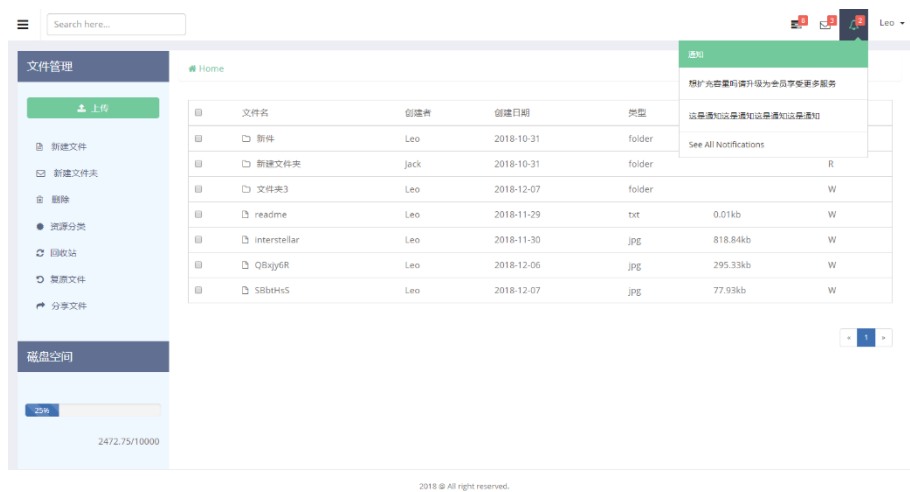


图 7-10 查看系统消息

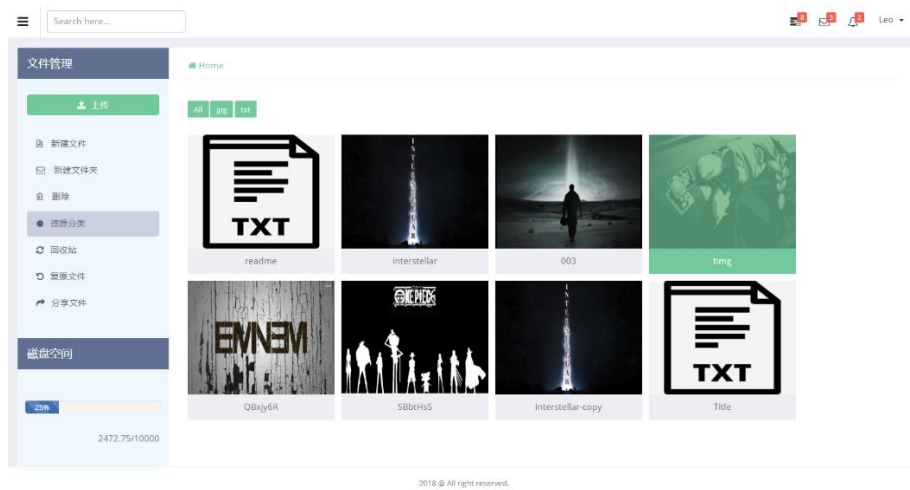


图 7-11 文件资源分类

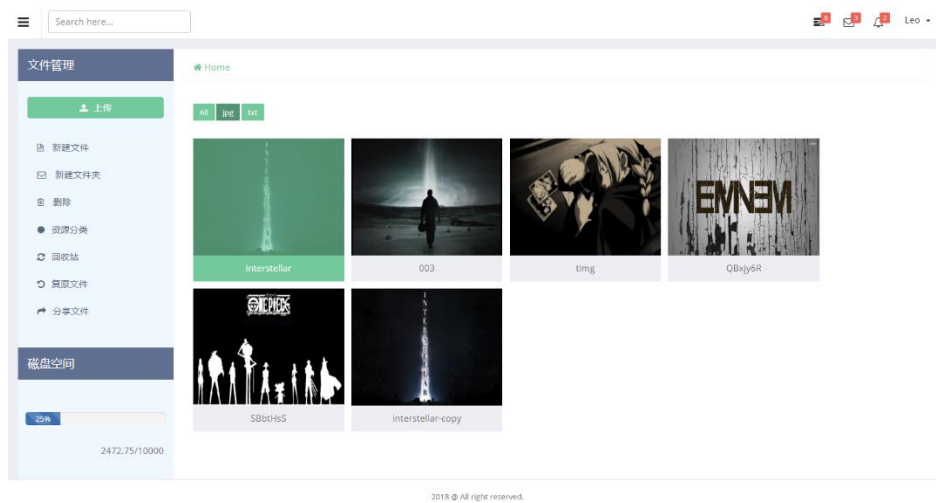


图 7-12 查看所有图片

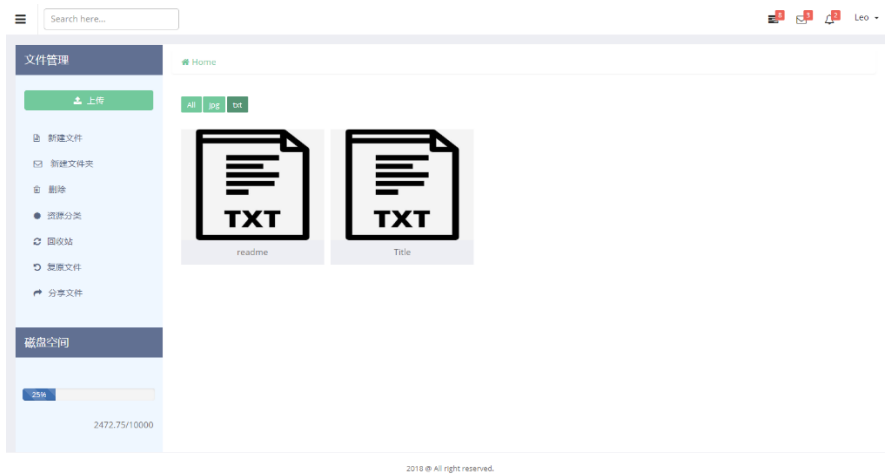


图 7-13 查看所有文本

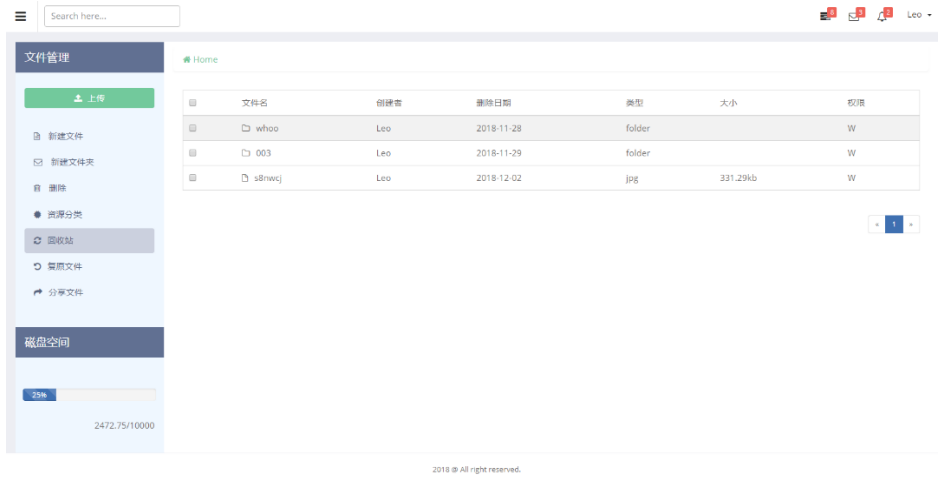


图 7-14 查看回收站

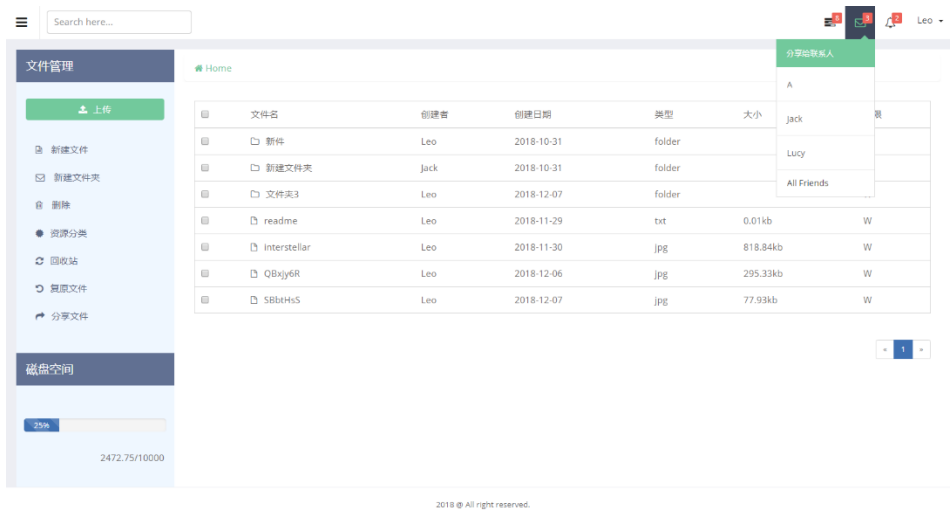


图 7-15 查找联系人分享文件

## 8 运行与维护

### 8.1 模块运行测试

#### 8.1.1 获取某用户所有文件

接口 URL : /getFile?dir=3

经测试，测试结果符合预期，接口返回数据如下图所示。

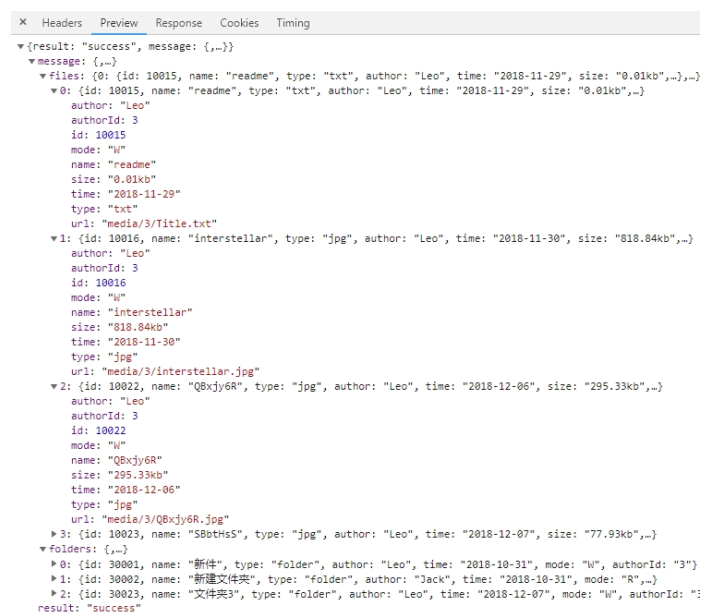


图 8-1 获取某用户所有文件信息

#### 8.1.2 获取用户磁盘使用情况

接口 URL : /getVolume

经测试，测试结果符合预期，接口返回数据如下图所示。

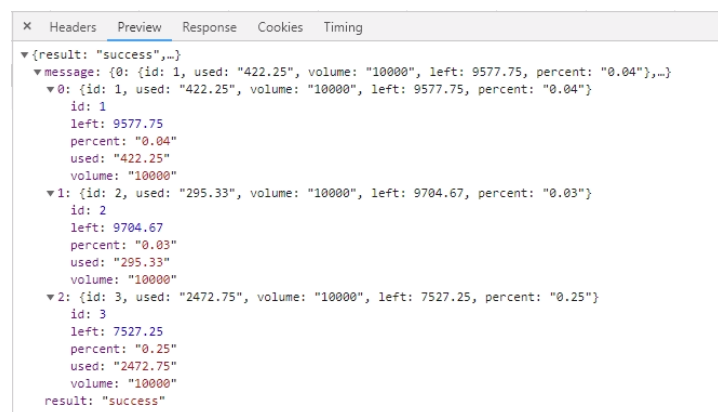


图 8-2 获取用户磁盘使用情况

### 8.1.3 获取某用户目录树

接口 URL : /getNode

经测试，测试结果符合预期，接口返回数据如下图所示。

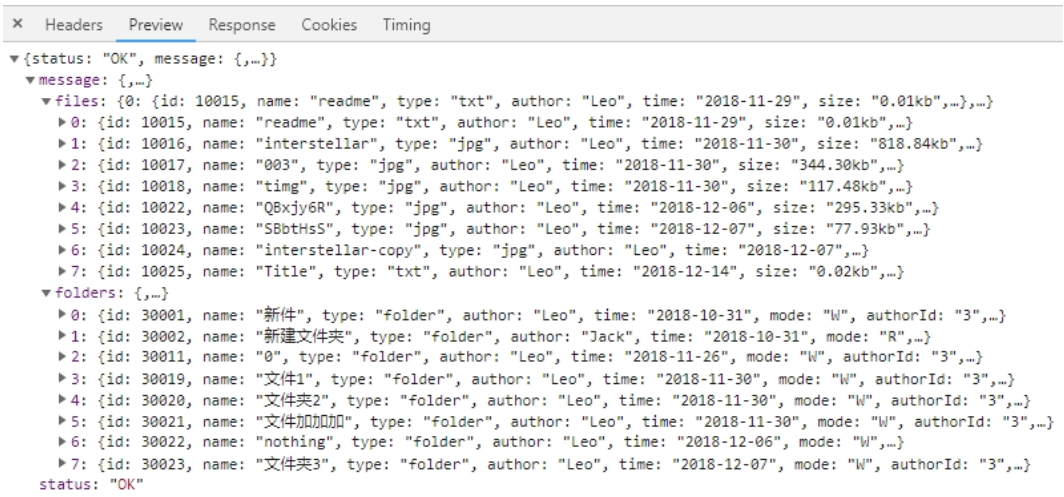


图 8-3 获取某用户目录树

### 8.1.4 上传文件

接口 URL : /upload

经测试，测试结果符合预期，接口返回数据如下图所示。

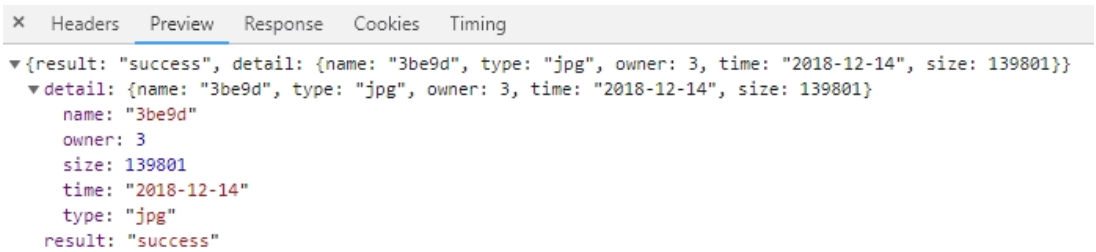


图 8-4 文件上传

界面显示如下图所示。

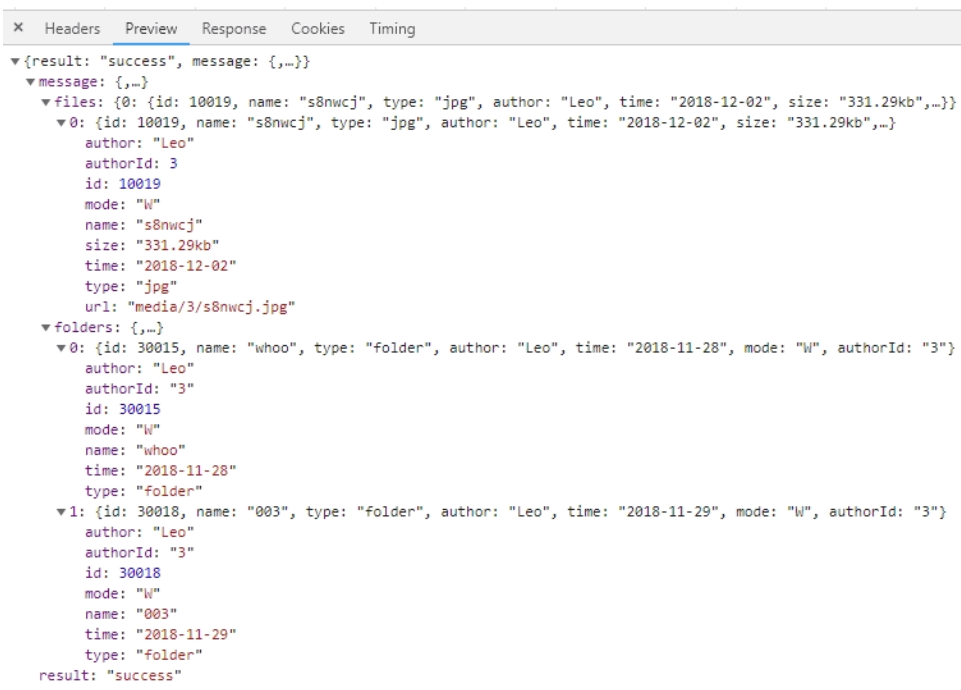


图 8-5 上传文件后界面提示上传成功

## 8.1.5 获取某用户回收站

接口 URL : /getDeleted

经测试，测试结果符合预期，接口返回数据如下图所示。



```
{result: "success", message: {files: [{id: 10019, name: "s8nwcj", type: "jpg", author: "Leo", time: "2018-12-02", size: "331.29kb",...}], folders: [{id: 30015, name: "whoo", type: "folder", author: "Leo", time: "2018-11-28", mode: "W", authorId: "3"}, {id: 30018, name: "003", type: "folder", author: "Leo", time: "2018-11-29", mode: "W", authorId: "3"}]}}
```

图 8-6 获取某用户回收站

## 8.2 维护

经过上面数据的初步载入和应用程序的调试，测试应用程序的功能满足现阶段设计要求。

一下是进一步运行和维护的计划。

1. 按照 6.2 节的设计执行数据库的备份和恢复方案，根据实际情况调整数据库的转储与恢复计划。
2. 在数据库运行过程中，当应用环境变化，安全性的要求发生变化时，系统中用户的密级也会改变，数据库管理员不断修正以满足用户要求。
3. 在数据库运行过程中，监督系统运行，对监测数据进行分析，找出改进系统性能的方法。
4. 数据库运行一段时间后，可能由于记录不断增删改，数据库的物理存储情况变坏，降低数据的存取效率，使数据库性能下降，数据库管理员要对数据库进行重组或部分重组。





## 9 总结

在完成此数据库课程设计的过程中，经过需求分析，系统详细设计，数据库初步设计和优化，本系统已经达到基本的性能需求等需求，而且有易于操作的用户界面，且经过初步测试，运行情况良好。

当然，在完成此电子资源管理系统的过程中，也遇到了不少问题并查阅了资料解决，收获很多，比如在数据库表的设计上，达到 4NF 并不一定就比 3NF 好，因为把表分解了，意味着查询可能需要另外的拼接，降低了查询的效率，如果查询非常频繁，效率其实是不高的，但是也可以创建视图，创建索引，或使用聚簇的方式来弥补查询效率低的不足。另外，在做这个系统之前，我还没有遇到过把树形结构存储在数据库中的情况，而文件就需要这种形式的存储，在查找某节点的所有子节点的问题上，也遇到了 SQL 语句难以设计的问题，最后查找资料了解到使用 MySQL 中的内置函数 FIND\_IN\_SET() 加上嵌套查询等操作可以解决这个问题。

总之，通过此课程设计，让我对整个系统数据库的设计流程从了解到亲自实践而更加熟悉，对数据库的增删查改等操作也更加熟练，同时对 SQL 有了更深的理解，还熟悉了 MySQL 这个数据库管理系统 DBMS 的使用，可以说收获很多。

## 参考资料

[1] 数据库系统概论 王珊 萨师煊 编著 高等教育出版社

[2] 数据库系统概念 Database System Concepts, 6E

(美)Abraham Silberschatz/(美)Henry F. Korth/(美)S. Sudarshan

编著 高等教育出版社