



**本科毕业论文**

|  |
| --- |
| **基于MongoDB的网盘系统的设计与实现** |
|  |

**罗广威**

**201231000417**

|  |  |
| --- | --- |
| 指导教师 | **陈湘骥 讲师** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学院名称 |  | **软件学院** | 专业名称 |  | **软件工程** |
| 论文提交日期 |  | 2016年4月 | 论文答辩日期 |  | 2016年5月 |

摘 要

近年来，随着信息化的进程，人们身边的大量的事物被数字化，产生了大量的，各种类型的数据文件。过去，人们常常将需要保存的重要文件保存在软盘，磁带等存储介质，随着科技的发展，软盘等存储介质逐渐被容量更大，速度更快的硬盘取代。时至今日，随着互联网的高速发展和人们接触文件的终端越来来越多元化,特别是移动终端和公共网络的发展，使人们对存储文件便捷性的要求越来越高。网盘的诞生满足了人们的这一需求，使人们可以随时随地存储和下载文件。基于上述背景，设计与实现了基于MongoDB的网盘系统，为用户提供了一个简单便捷的网络存储方式，也为管理员提供了一个系统管理平台。

本文首先是对相关技术的介绍，首先是对MongoDB的说明，说明用基于MongoDB搭建网盘系统的优势和不足之处，再介绍系统使用的MVC框架Yii 2.0，介绍该框架的相关特性，最后介绍本系统的前端框架Bootstrap。接着是阐述本系统的需求分析和系统设计。本系统分为权限管理模块，用户文件管理模块和管理员模块三大模块，针对每个模块中的主要业务进行用例分析。在此基础上进行数据库的设计和搭建，根据各自特点，主要的业务逻辑使用的是MySQL数据库，文件存储使用的是MongoDB。最后阐述了搭建系统的具体实现，介绍Yii 2.0框架的具体配置方法和相关辅助技术的配置方法，以及对最后实现结果和系统使用方法的简单介绍。

关键词：网盘 MongoDB Yii 2.0

目 录

[1 引言 1](#_Toc445746052)

[1.1 选题背景 1](#_Toc445746053)

[1.2 国内网盘的现状 2](#_Toc445746054)

[1.3 主要内容 2](#_Toc445746055)

[2 相关技术介绍 3](#_Toc445746056)

[2.1 NoSQL和MongoDB技术介绍 3](#_Toc445746057)

[2.1.1 NoSQL数据库简介 3](#_Toc445746058)

[2.1.2 MongoDB数据库简介 3](#_Toc445746059)

[2.1.3 GridFS存储文件 4](#_Toc445746060)

[2.2 Yii Framework框架介绍 5](#_Toc445746061)

[2.2.1 Yii框架简介 5](#_Toc445746062)

[2.2.2 Yii框架的应用结构 5](#_Toc445746063)

[2.3 Highcharts简介 6](#_Toc445746064)

[2.4 开发环境和运行环境 7](#_Toc445746065)

[2.4.1 开发环境PhpStorm 7](#_Toc445746066)

[2.4.2 运行环境Apache 7](#_Toc445746067)

[2.4.3 MySQL数据库 7](#_Toc445746068)

[2.4.4 浏览器 7](#_Toc445746069)

[3 系统需求分析与设计 8](#_Toc445746070)

[3.1 网盘系统需求分析 8](#_Toc445746071)

[3.1.1 用例图 8](#_Toc445746072)

[3.1.2 时序图 10](#_Toc445746073)

[3.2 系统设计 13](#_Toc445746074)

[3.2.1 类图 13](#_Toc445746075)

[3.2.2 MongoDB数据库设计 13](#_Toc445746076)

[3.2.3 MySQL数据库设计 14](#_Toc445746077)

[3.3 模块设计 19](#_Toc445746078)

[3.3.1 系统模块图 19](#_Toc445746079)

[3.3.2 权限管理模块 20](#_Toc445746080)

[3.3.3 用户文件管理模块 20](#_Toc445746081)

[3.3.4 管理员模块 20](#_Toc445746082)

# 1 引言

## 1.1 选题背景

随着互联网和公共网络的发展，人们对通过网络进行数据存储这种便捷的存储方式的

需求，各大公司也纷纷推出了自己的网盘产品，比如百度云，360网盘等。网盘，又称网络硬盘、网络U盘，是一种基于网络的在线存储服务。网盘向用户提供文件的存储、共享、访问、备份等文档管理功能。（彭晓红，2013）。网盘的诞生，向广大用户的数据存储和数据共享提供了一个崭新的方式，它的移动性、安全性和便捷性是传统的存储方式所不能比拟的。

虽然随着科技和工艺技术的进步，硬盘的容量大小，体积，存储速度和稳定性等各方面特性都取得飞跃的进步，固态硬盘的诞生和普及更是把数据存储推向了一个新的时代，大有取代机械硬盘的趋势。但是，网盘所满足的需求不是存储介质的各项指标的进步所能够满足，相较于传统的存储方式，把数据存入硬盘、光盘等，将数据存入网盘具有以下几点存储方式。

第一，网盘比传统存储方式更方便。虽然移动硬盘的体积越做越小，但是随身携带总有不方便的时候，而网盘只有连接互联网就等于随身携带了一块硬盘，不会给用户带来任何负担，并且随着公共网络的发展，使用网盘变得越来越方便。

第二，空间比传统存储方式更大。现在人们随身携带的更多的是U盘，而U盘的空间有限，而网盘的用户存储空间与硬盘相当，有过之而不及。

第三，网盘比传统存储方式更安全。无论是移动硬盘还是U盘，都会有丢失的风险，而网盘没有这种风险。任何存储介质都会有出故障的几率，而商业性的网盘都会有完备的备份方案和加密方案，所以数据也更安全。

正是因为网络存储具有以上的种种优势,越来越多人开始使用网盘作为自己临时存储,备份和分享资料的方式。在市场上的网盘产品行行色色，虽然各有特点，但是都离不开基础的核心文件操作功能。作为一个传统存储方式的替代者，补充者，网盘系统首先要做到传统的存储方式能够满足的功能，可以供用户便捷地上传、下载文件资料，能够让用户方便地组织和管理文件资料。在此基础上，充分发挥网盘的互联网属性，用户能够方便地进行文件分享。最后做好网盘系统的后台服务，做好安全和备份工作，使网盘系统能够稳定安全地运行。如果能够完成这些基本要求，让用户能够方便、安心地使用网盘，就能够在市场上有一定的竞争力。

## 1.2 国内网盘的现状

2012年，中国企业网盘市场规模约为1500-2000万元，2014年，中国企业网盘市场规模约为5000-6000万元。根据市场规模估算，2012-2014年市场增持绿始终报纸在70%以上（CCID，2015）。目前，国内市场上的主流网盘有：华为（数据银行DBank）、新浪（微盘）、金山（快盘）、115网盘（优蛋）、迅雷（随身盘）、360（云盘）、腾讯QQ。（彭晓辉，2013）。

上述网盘的容量基本上都超过了1T，可以满足用户日常的使用需求，360云盘的容量达到了36T，用户还可以通过付费来扩充容量。在其他方面，百度云网盘的分享功能较为突出，金山快盘因为自身的技术优势，在线上编辑文档比较方便。虽然各大公司的网盘各有自己的优势，但是基础功能上基本一致，都能够做到上传、下载和文件组织功能，有的产品在某些功能上需要用户安装软件来支持，例如上传下载大文件，文件夹等。就浏览器的客户端而已，都能让用户方便的使用。

本论文基于上述背景，设计与实现了基于MongoDB搭建网盘系统的解决方案。

## 1.3 主要内容

本文主要提供一个基于MongoDB搭建网盘系统的解决方案，阐述网盘系统的分析，设计与实现的过程，本文的章节内容的结构如下：

第一部分主要是阐述基于MongoDB的网盘系统的选题背景和目前市场上相关产品现状的简单概述。

第二部分是关于本系统所使用的框架，数据库以及有所涉及的相关技术和系统开发平台和运行环境的介绍。

第三部分主要是关于网盘系统具体的分析与设计过程，包括功能模块，用例图，类设计以及数据库设计。

第四部分主要介绍系统的具体实现，主要功能设计以及相关配置项的介绍。

第六部分是网盘系统的界面和功能展示部分。

# 2 相关技术介绍

## 2.1 NoSQL和MongoDB技术介绍

### 2.1.1 NoSQL数据库简介

NoSQL数据库，泛指非关系型的数据库，NoSQL是Not Only SQL的简写，其含义是“不仅是结构化查询”，是不同于传统的关系型数据库的数据库管理系统的统称（陈明，2013）。NoSQL都普遍存在以下的一些共同特征：

第一不需要预定于模式：不需要事先定义数据模式和预定义表结构。数据中的每条记录可以有各不相同的属性和格式。插入记录的时候，不需要事先预定义它们的模式。

第二无共享结构：相对于将所有数据存储的存储区域网络中的全共享架构。NoSQL往往将数据划分后存储在各个本地服务器上。为了提高系统的性能，从本地磁盘读取数据的性能往往好于通过网络传输读取数据的性能。

第三弹性可扩展：在系统运行的时候，可以动态增加或者删除节点。不需要停机维护，数据可以自动迁移。

第四分区：相对于将数据存放在同一个节点，NoSQL数据库需要将数据进行分区，将记录分散在多个节点上面。并且通常分区的同时还有做复制。这样既提高了并行性能，又保证没有单点失效的问题。

第五异步复制：和RAID（Redundant Arrays of Independent Disks）存储系统不一样的地方是，NoSQL中的复制通常是基于日志的异步复制。这样，数据就可以尽快地写入一个节点，而不会被网络传输引起延迟。缺点是并不总能确保一致性，这样的方式在出现故障的时候，可能会丢失少量的数据。

第六BASE（Basically Available Soft state Eventually consistent）：相对于事物严格的ACID（Atomicity Consistency Isolation Durability）特性，NoSQL数据库保证的是BASE特性，BASE主要强调基本的可用性，是最终一致性和软事务。

NoSQL数据库主要有四大分类，分别是键值（Key-Value）存储数据库，列存储数据库，文档型数据库和图形（Graph）数据库。

### 2.1.2 MongoDB数据库简介

MongoDB是一款强大、灵活的，且易于扩展的通用型数据库。它能扩展出非常多的功能，如二级索引（secondary index）、范围查询（range query）、排序、聚合（aggregation），以及地理空间索引（geospatial index）。MongoDB作为一款NoSQL的数据库，它具有以下几个特点：

（1）易于使用：MongoDB是一个面向文档（document-oriented）的数据库，而不是关系型数据库。不采用关系模型主要是为了获取更好的扩展性。

（2）易于扩展：MongoDB的设计采用横向扩展。面向文档的数据模型使它很容易地在多台服务器之间进行分割，它能自动处理跨集群的数据和负载，自动重新分配文档，以及将用户请求路由到正确的机器上。

（3）丰富的功能：MongoDB除了能够创建、读取、更新和删除数据外，还扩展了一系列功能，如索引（indexing），聚合（aggregation），文件存储（file storage）等。

（4）卓越的性能：MongoDB能够对文档进行动态填充（dynamic padding），也能够预分配数据文件以利用额外的空间来获取稳定的性能。

### 2.1.3 GridFS存储文件

GridFS是MongoDB的一种存储机制，用来存储大型二进制文件。相比用文件系统来存储文件，使用MongoDB具有以下几点优势：

（1）GridFS会自动平衡已有的复制或者为MongoDB设置的自动分片，所以对文件存储做故障转移和横向扩展会更容易。

（2）GridFS可以避免在其一些文件系统可能会遇到的问题。例如，在GridFS中，如果在同一个目录存储大量的文件是没有任何问题的。

（3）在GridFS中，文件存储的集中度会比较高，因为MongoDB是以2GB为单位来分配数据文件的。

（4）数据维护相对容易。使用传统的文件系统必须自行进行备份、复制和扩展。GridFS实现了分布式文件系统的功能，可以方便地进行扩展。

使用GridFS作为文件存储方式以有以下两个缺点：

（1）GridFS的性能比较低：从MongoDB中访问文件，速度比直接从文件系统中访问文件慢。

（2）如果要修改GridFS上的文档，只能先将已有的文档删除，然后再讲整个文档重新保存。MongoDB将文件作为多个文档进行存储，所以它无法在同一时间对文件中的所有块加锁。

综上所述，GridFS非常适合用来存储一些不常改变但是经常需要连续访问的大文件。所以结合关系关系数据库和NoSQL数据库的各自优点和不足，决定使用MongoDB作为网盘系统的文件存储方式，使用MySQL数据库存储对事务要求高的数据。

## 2.2 Yii Framework框架介绍

### 2.2.1 Yii框架简介

Yii是一个高性能,基于组件的PHP（Hypertext Preprocessor）框架，用于快速开发现代Web应用程序。Yii是一个通用的Web编程框架，既可以用来开发各种PHP构建的Web应用。因为基于组件的框架结构和设计精巧的缓存支持，它特别适合开发大型应用，如门户网站、社区、CMS（Content Management System）内容管理系统、电子商务项目和RESTful（Representational State Transfer） Web服务等。

Yii框架有以下几个特点：

（1）Yii实现了MVC（Model-View-Controller）设计模式并基于该模式组织代码。

（2）Yii的代码较为简洁，便于学习理解和快速编程。

（3）Yii是一个全栈框架，并且对关系型和NoSQL数据库都有很好的支持。

（4）Yii易于扩展。可以让使用者自定义或替换几乎任何一处的核心代码。

（5）Yii具有较高的性能。

### 2.2.2 Yii框架的应用结构

Yii框架使用MVC设计模式来组织。模型代表数据、业务逻辑和规则，视图用来展示模型的输出；控制器接受请求并且将其转换为模型和视图命令。Yii框架应用的静态结构示意图如图1所示。除了MVC，Yii应用还包括以下几个部分：

（1）入口脚本：终端用户能直接访问的PHP脚本，负责启动一个请求处理周期。

（2）应用：能够全局范围访问的对象，管理协调组件来完成请求。

（3）应用组件：在应用中注册的对象，提供不同的功能来完成请求。

（4）模块：包含完整MVC结构的独立包，一个应用可以由多个模块组建。

（5）过滤器：控制器在处理请求之前或之后需要出发执行的代码。

（6）小部件：可嵌入到视图中的对象，可包含控制器逻辑，可被不同视图重复调用。

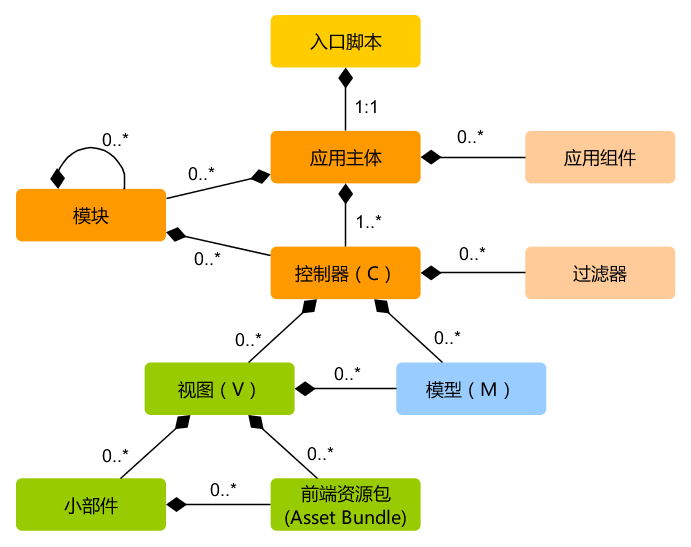


图1 Yii应用的静态结构示意图

## 2.3 Highcharts简介

HighCharts是一个用纯JavaScript编写的一个图表库，能够很简单编辑的在web网站或者是web应用程序添加具有交互性的图表，并且免费提供给个人学习、个人网站和非商业用途使用。Highcharts支持的图表类型有曲线图、区域图、柱状图、饼状图、散状点图和综合图表。

Highcharts具有以下几点主要特性：

（1）兼容性：Highcharts采用纯JavaScript编写，兼容当今大部分的浏览器，包括Safari、IE和火狐等。

（2）不受语言约束：Highcharts可以在大多是的WEB开发中使用，并且对个人用户免费，支持ASP（Active Server Page），PHP，JAVA，.NET等多种语言中使用。

（3）导出：表格可导出为PDF、PNG、JPG、SVG等格式。

## 2.4 开发环境和运行环境

### 2.4.1 开发环境PhpStorm

PhpStorm是由JetBrains公司开发的一款商业的PHP集成开发工具，它是一个轻量级且便捷的PHP IDE（Integrated Development Environment），PhpStorm能够让用户对其编码进行调整，运行单元测试或者提供可视化的调试功能。

### 2.4.2 运行环境Apache

Apache HTTP Server是Apache软件基金会的一个开放源码的网页服务器，可以在大多数计算机操作系统中运行，由于其多平台和安全性被广泛使用，是最流行的Web服务器端软件之一。它快速、可靠并且可通过简单的API（Application Programming Interface）扩展，将Perl、Python等解释器编译到服务器中。

### 2.4.3 MySQL数据库

MySQL是一个关系型数据库管理系统，由瑞典MySQLAB公司开发，目前属于Oracle旗下公司。MySQL是最流行的关系型数据库管理系统之一，在Web应用方面MySQL是最好的RDBMS（Relational Database Management System）应用软件之一。MySQL所使用的SQL语言是用于访问数据库的最常用标准化语言。MySQL与PHP和Apache搭配可组成良好的开发运行环境。

### 2.4.4 浏览器

Mozilla Firefox，火狐浏览器，是一个自由及开放源代码网页浏览器，使用Gecko排版引擎，支持多种操作系统，如Windows、Mac OS X及GNU/Linux等。

Google Chrome是由Google开发的一款设计简单、高效的Web浏览器。Google Chrome的特点是简洁、快捷、Google Chrome支持多标签浏览，每个标签页面都在独立的“沙箱”内运行，在提高安全性的同时，一个标签页面的崩溃也不会导致其他标签页面被关闭。此外，Google Chrome基于更强大的JavaScript V8引擎，这是当前Web浏览器所无法实现的。

# 3 系统需求分析与设计

## 3.1 网盘系统需求分析

### 3.1.1 用例图

用例：用例是系统、子系统或类和外部的参与者（actor）交互的动作序列的说明，包括可选的动作序列和会出现异常的动作序列。

参与者：参与者是指系统以外的，需要使用系统或者与系统交互的东西，包括人、设备、外部系统等。

用例间的关系存在以下三种：

（1）泛化关系：泛化（generalization）代表一般与特殊的关系。在泛化关系中，子用例继承了父用例的行为和含义，子用例也可以增加新的行为和含义或者覆盖父用例中的行为和含义。

（2）包含关系：包含（include）关系指的是两个用例之间的关系，其中一个用例（称作基本用例，base use case）的行为包含了另一个用例（称作包含用例，inclusion use case）的行为。

（3）扩展关系：扩展（extend）关系的基本含义与泛化关系类似。但在扩展关系中，对于扩展用例（extension use case）有更多的规则限制，即基本用例必须声明若干“扩展点”（extension point），而扩展用例只能在这些扩展点上增加新的行为和含义。

用例图（use case diagram）：用例图是显示一组用例、参与者以及它们之间关系的图，主要用于对系统，子系统或类的功能行为进行建模。

下图2展示了用户模块的用例图，包括一个参与者以及以下几个用例。登录系统、注册账号、上传文件、下载文件、删除文件、管理文件（重命名、复制、剪切、粘贴、创建文件夹）、查看文件、查看用户信息。

下图3展示了管理员模块的用例图，包括以下几个主要用例。登录管理系统、查看系统运行信息、查看账户信息、设置用户状态、设置文件状态。



图2 用户用例图



图3 管理员用例图

### 3.1.2 时序图

时序图，也称作顺序图，是一种UML（Unified Modeling Language）交互图。Rumbaugh对顺序图的定义是：顺序图是显示对象之间交互的图，这些对象是按照时间顺序排列的。特别地，顺序图中显示的是参与交互的对象及对象之间消息交互的顺序。顺序图中还有以下几个要素：角色，对象，生命线，激活期和消息。

（1）角色（Actor）：系统角色，可以是人或者其它系统，子系统。

（2）对象（Object）：对象代表时序图中的对象在交互中所扮演的角色，位于时序图顶部和对象代表类的角色。

（3）生命线（Lifeline）：生命线代表时序图中的对象在一段时期内的存在。时序图中每个对象和底部中心都有一条垂直的虚线，这就是对象的生命线，对象之间的消息存在于两条虚线间。

（4）激活期（Activation）：激活期代表时序图中的对象执行一项操作的时期，在时序图中每条生命线上的窄的矩形代表活动期。

（5）消息（Message）：消息是定义交互和协作中交换信息的类，用于对实体间的通信内容建模，信息用户在实体间传递信息。允许实体请求其他的服务，类角色通过发送和接受信息进行通信。



图4 上传文件时序图

图4所示为用户上传文件用例的时序图，用户点击上传文件，页面打开文件选择窗口，用户选择文件，点击上传提交。浏览器提交表单，上传文件。文件控制器接受请求，调用FileService保存上传文件，FileService创建UserFile并写入MongoDB，若写入成功创建FileRecord并写入MySQL。MySQL写入成功后返回上传成功信息。



图5 管理用户时序图

图5所示为管理员设置用户用例时序图。管理员首先需要点击查询需要设置的用户账户，浏览器提交表单。管理员控制器接受管理员请求，并且调用UserService查找用户。

UserService创建User查找用户，如果存在则返回该用户信息。页面显示该用户账户信息，管理员如需修改则修改用户信息，点击提交。管理员控制器接受请求，并调用UserService设置用户。User修改该用户信息并保存，并返回结果。



图6 系统类图

## 3.2 系统设计

### 3.2.1 类图

类图描述了系统中各类对象以及它们之间的各种关系。在面向对象的建模中，建立是最基本的任务。类图中包括以下几个要素：

（1）类（class）：类是对拥有相同属性和操作的一组对象的统一抽象描述。

（2）属性：属性是用来描述对象性质的一个数据项。

（3）操作：操作是类的对象被要求提供的服务的规约。

根据对系统主要用例的分析，归纳得出系统类图，如图6所示，系统共包含11个类，分别是：用户（User），管理员（Admin），用户控件（Disk），文件记录（FileRecord），文件（File），登录日志（LoginLog），分享码（ShareCode），下载日志（DownlogLog），删除日志（RemoveLog）文件管理记录（FileManageLog），用户管理记录（UserManageLog）。

### 3.2.2 MongoDB数据库设计

根据NoSQL数据库和关系数据库各自的特点，本网盘的文件存储是基于MongoDB，即使用MongoDB的GridFS存储大文件，使用关系数据库MySQL存储其他数据，利用MongoDB存储管理文件的优势和MySQL在事务和SQL查询方面的优势。

MongoDB是一个面向文档的数据库，而文档就是键值对的一个有序集。MongoDB中集合就是一组文档。如果将MongoDB中的一个文档比喻成关系数据库中的一行，那么一个集合就相当于一张表。多个文档组成集合，而多个集合可以组成数据库。在概念上，MongoDB的文档与JavaScript中的对象相近，因而可以认为它类似于JSON。

GridFS的理念是：将大文件分割为多个比较大的块，将每个块作为独立的文件进行存储。由于MongoDB支持在文档中存储二进制数据，所以可以将块存储的开销降到非常低。除了将文件的每一个块单独存储之外，还有一个文档用于将这些块组织在一起并存储该文件的云信息。

本网盘使用数据库webdb存储文件，分割的文件块使用的集合是files.chunks。files.chunks的文档结构为：

{

“\_id”：ObjectId（“…”），

“n”：0，

“data”：BinData（“…”），

“files\_id”：ObjectId（“…”）

}

“\_id”存储文档唯一标识。“files\_id”存储块所属文件的元信息。“n”存储块在文件中的相对位置。“data”存储块所包含的二进制数据。

每个文件的元信息保存在集合files.files，其文档结构如下：

{

“\_id”：ObjectId（“…”），

“filename”：“string”，

“filetype”：“string”，

“length”：NumberLong（），

“chunkSize”：NumberLong()，

“uploadDate”：ISODate（），

“md5”：“string”

}

其中主要键值得含义为：“\_id”是文件的唯一id，“length”存储文件所包含的字节数，“chunkSize”表示组成文件的每个块的大小，单位为字节，“uploadDate”存储文件被上传到GridFS的日期。“md5”表示文件内容的md5校验值，这个值有服务器端计算得到。

### 3.2.3 MySQL数据库设计

根据系统分析得出数据库ER图如图7所示。

系统共有7各实体，8组关系。其中用户与操作。其中用户与登录日志和管理员与登录日志之间皆为1对N的关系。用户与分享码和文件与分享码之间为1对N的关系。用户与文件项之间为1对N的关系。管理员与用户管理日志和管理员与文件管理日志皆为1对N的关系。用户与用户空间为1对1的关系。



图7 系统ER图

根据ER图及分析出的类设计，共建立以下数据表：

表1 用户表user

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| user\_id | 用户id | varchar | PK | NO |
| user\_email | 用户邮箱 | text |  | NO |
| user\_name | 用户名 | varchar |  | NO |
| user\_password | 用户密码 | varchar |  | NO |
| create\_date | 申请时间 | datetime |  | NO |
| state | 账户状态 | char |  | NO |

表1位用户表，共有6个字段，其中user\_id为主键，user\_email为用户登录账户，不允许重复。

表2 管理员表admin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| admin\_id | 管理员id | varchar | PK | NO |
| admin\_account | 管理员账户 | varchar |  | NO |
| admin\_password | 管理员密码 | varchar |  | NO |
| create\_date | 创建时间 | datetime |  | NO |

表2为管理员表，共有4个字段，其中admin\_id为主键，admin\_account为管理员登录账户，不允许重复，管理员由系统生成，不允许注册。

表3 用户空间表disk

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| disk\_id | 空间id | varchar | PK | NO |
| user\_id | 用户id | varchar | FK | NO |
| capacity | 空间大小 | double |  | NO |
| available\_size | 可用容量 | double |  | NO |
| create\_date | 创建日期 | datatime |  | NO |

表3为用户空间表，用于记录用户空间信息，其中disk\_id为主键，user\_id为外键，一个用户只能有一个空间。用户空间在用户注册账号时自动创建。

表4 文件项表file\_record

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| f\_record\_id | 文件项id | varchar | PK | NO |
| f\_record\_type | 文件项类型 | char |  | NO |
| file\_id | 文件id | varchar |  | YES |
| user\_id | 用户id | varchar | FK | NO |
| file\_name | 文件项名 | text |  | NO |
| extension | 文件扩展名 | varchar |  | YES |
| file\_type | 文件项类型 | varchar |  | NO |
| file\_size | 文件项大小 | double |  | NO |
| parent\_id | 父节点id | varchar |  | YES |
| upload\_date | 创建时间 | datetime |  | NO |
| state | 文件项状态 | char |  | NO |

表4为文件项纪录表，其中f\_record\_id为文件项id，文件项类型共有两种，分别为文件和文件夹。file\_id为MongoDB中文件的唯一id，可为多个文件项引用，若文件项为文件夹则该项为空。user\_id为外键，每个文件项只属于一个用户。file\_name为文件项名，若为文件则代表文件名，若为文件夹则表示文件夹名。extension为文件扩展名，若为文件夹则该项为空。file\_type为表单文件类型，若为文件夹则该项为字符串“folder”。file\_size为文件大小，若为文件夹则该项为0。parent\_id为父节点id，若为更目录则该项为空。用户注册账号时，系统自动创建f\_record\_id为user\_id的根目录。

表5 分享码表share\_code

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| code\_id | 分享码id | varchar | PK | NO |
| code | 分享码 | varchar |  | NO |
| user\_id | 用户id | varchar |  | NO |
| f\_record\_id | 文件项id | varchar |  | NO |
| create\_date | 创建时间 | datetime |  | NO |

表6 删除记录表remove\_log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| r\_log\_id | 字段id | varchar | PK | NO |
| f\_record\_id | 文件项id | varchar | FK | NO |
| remove\_date | 删除日期 | datetime |  | NO |

表7 下载记录表download\_log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| d\_log\_id | 字段id | varchar | PK | NO |
| user\_id | 用户id | varchar | FK | NO |
| file\_id | 文件id | varchar |  | NO |
| download\_date | 下载时间 | datetime |  | NO |

表8 登录记录表login\_log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| l\_log\_id | 字段id | varchar | PK | NO |
| person\_id | 登录者id | varchar | FK | NO |
| login\_date | 登录时间 | datatime |  | NO |
| login\_ip | 登录ip | varchar |  | YES |
| ip\_address | ip所在地 | varchar |  | YES |

表9 用户管理记录表user\_manage\_log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| um\_manage\_id | 字段id | varchar | PK | NO |
| um\_manage\_type | 操作类型 | char |  | NO |
| um\_manage\_info | 操作信息 | text |  | YES |
| user\_id | 用户id | varchar | FK | NO |
| admin\_id | 管理员id | varchar | FK | NO |
| create\_date | 创建时间 | datetime |  | NO |

表10 文件管理记录表file\_manage\_log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 含义 | 类型 | PK/FK | NULL |
| fm\_manage\_id | 字段id | varchar | PK | NO |
| fm\_manage\_type | 操作类型 | char |  | NO |
| fm\_manage\_info | 操作信息 | text |  | YES |
| file\_id | 文件id | varchar |  | NO |
| admin\_id | 管理员id | varchar | FK | NO |
| create\_date | 创建时间 | datetime |  | NO |

## 3.3 模块设计

### 3.3.1 系统模块图



图8 系统模块图

如图8所示，系统主要分为三大模块，分别为权限管理模块，用户文件管理模块和管理员模块。

### 3.3.2 权限管理模块

### 3.3.3 用户文件管理模块

### 3.3.4 管理员模块