豌豆射手

数据库设计说明书

小组成员：

郑廷健（221600137）、余嘉宸（221701117）、王泽宇（221701135）

李舸（221701216）、何瑞晨（221701239）、伍燕文（221701315）

唐涔（221701334）、洪惠强（221701419）、温杰（021700531）

目 录

1. 引言

本项目介绍了该软件——豌豆射手。豌豆射手是一个方便简短记录自己对书影音看法与评价的平台，利用休闲娱乐时间来了解并且分享用户所喜爱的书籍，影视以及音乐的软件项目。在这个项目当中用户可以发表自己有关任何作品的想法，见解，发布到网络上与其他相关的爱好者一起交流讨论，让用户可以更好的接触网络上受大众热爱的一些事物，并且积极的参与到讨论当中，从而更加融入到互联网当中。

该文档详尽说明了这一软件产品的E-R分析设计部分、数据库表结构设计、系统安全和健壮性、权限设计说明部分。

关键词：**E-R 分析、权限设计、系统安全**

1.1 项目概述

1.1.1 项目内容

豌豆射手是一个充分利用休闲娱乐时间来了解并且分享用户所喜爱的书籍，影视以及音乐的软件项目。

1.1.2 项目具体

在这个项目当中用户可以发表自己有关任何作品的想法，见解，发布到网络上与其他相关的爱好者一起交流讨论，让用户可以更好的接触网络上受大众热爱的一些事物，并且积极的参与到讨论当中，从而更加融入到互联网当中。

1.3 附加功能

①该软件项目提供各种相关资源，减少用户在网络上寻找热点话题的相关资源，让用户可以更快的加入到讨论当中，可以更好地发表自己的看法。

②项目提供个人主页的功能，用户可以分享自己的日常点滴到主页当中，可以上传资源分享，可以转载收藏其他用户的随笔以便观看，也可以关注其他用户，时刻了解他们的想法作品。

③项目还根据不同作品进行分块，用户可以选择自己所管兴趣的内容，参与其中进行讨论，节约用户寻找话题的时间，增加针对性，让用户可以更容易找到志同道合的朋友。

1.2 目的

数据库设计说明书的编制目的是对数据库中使用的所有标识、逻辑结构和物理结构做出具体的设计规定。数据库的表结构设计是整个项目开发中一个非常重要的环节,一个良好的数据库设计，可以提高开发效率，方便系统维护，并且为以后项目功能的扩展留下余地。我们通过书写这份文档说明，从各方面进行网上订餐系统的数据库设计规划，用它指导该系统在数据库各方面的内容，为系统开发的程序员、系统分析员提供基准文档。我们也希望通过写数据设计说明书，规范数据名称、数据范围、数据代码等。这份文档是项目小组共同作战的基础，有了开发规范、程序模块之间和项目成员之间的接口规则、数据方式，大家就有了共同的工作语言、共同的工作平台，使整个软件开发工作可以协调有序地进行。

1.3 文档约定

标题	字体	字号	是否加粗
标题 1	宋体	三号	加粗
标题 2	宋体	小三号	加粗
标题 3	宋体	小四号	不加粗
正文	宋体	五号	不加粗

1.4 预期的读者和阅读建议

本文档面向多种读者对象：

（1） 项目经理：根据该文档了解产品数据流向过程；

（2） 开发人员：了解系统的数据库设计，进行业务设计；

（3） 测试人员：根据文档的数据库设计，对产品进行功能性测试，以及非功能性测试

1.5 术语

术语	解释与描述
WebApp	基于 Web 的系统和应用，其作用是向广大的最终用户发布一组复杂的内容和功能

1.6 外部设计

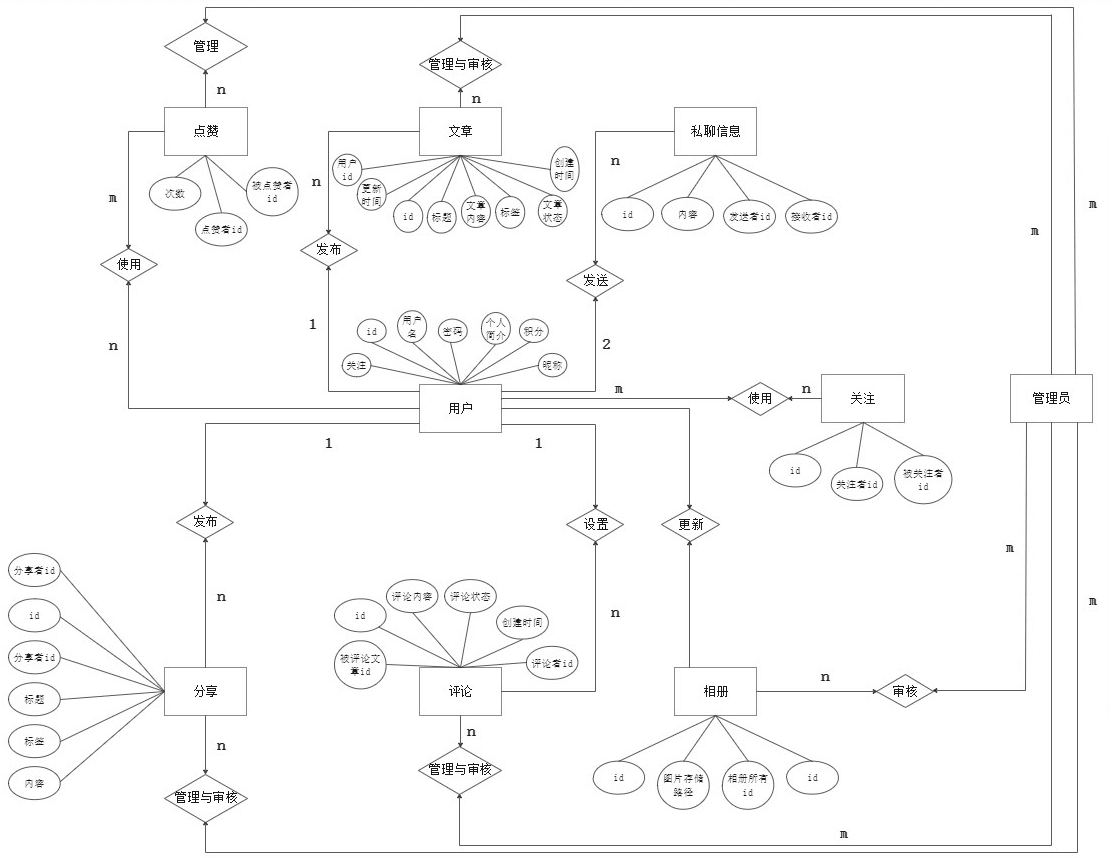
数据库：mysql1.8

数据库名称：peashooter

所有的数据库命名都是以模块的缩写加上具体表的英文词汇组成，这样能够统一数据库表的命名，也能够更好的规范数据库表命名。

2. E-R 分析

2.1 E-R 图

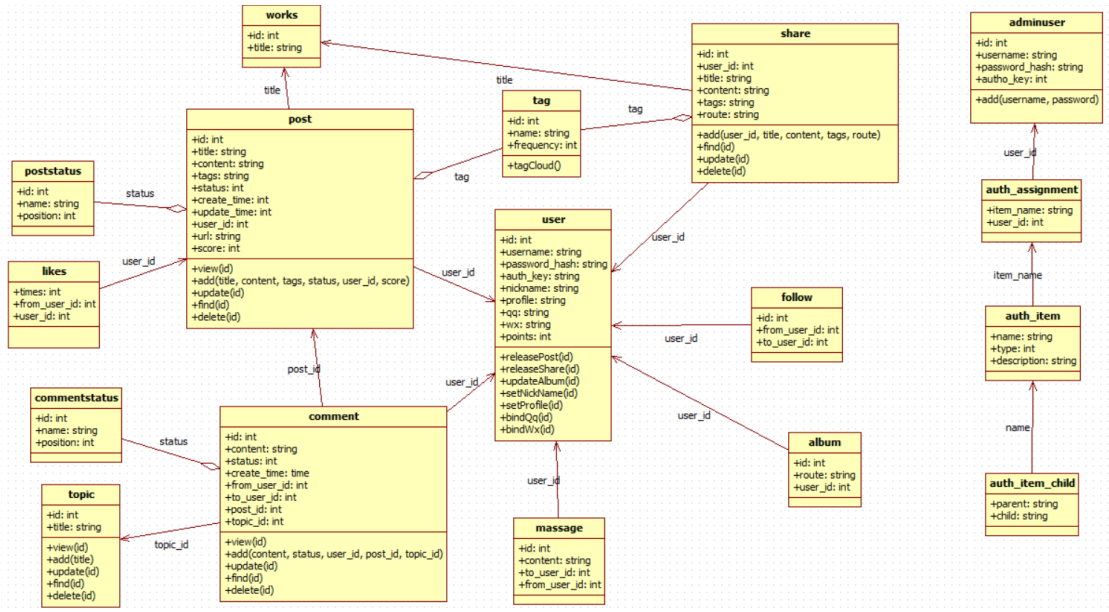


2.2 E-R 分析

本次 E-R 设计存在 3 种一般性约束：一对一约束（联系）、一对多约束（联系）和多对多约束（联系），它们用来描述实体集之间的数量约束。上图描绘了整个系统的 E-R 图，分为很多模块，包括点赞模块、文章模块、用户模块、关注模块、相册模块、评论模块、分享模块。每个模块都描述实体的属性以及属性之间的联系。大多数都是多对多约束，所以中间类就显得十分重要，用于关联两个有多对多关联的实体。

3　表结构设计

3.1 类图



描述：

Works:作品类

　属性包括 id、作品标题。

Post:文章类

　属性包括 id、标题、文章内容、标签、文章状态、创建时间、更新时间、用户 id、评分。操作包括增删改查、显示。

Poststatus:文章状态类

　属性包括 id、属性名、属性值。为文章类提供文章状态。

Likes：点赞类

　属性包括次数、点赞者 id、被点赞者 id。

User：用户类

　属性包括 id、用户名、密码 hash 值（随机生成用于加密）、密码（加密后）、昵称、个人简介、qq 绑定信息、微信绑定信息、积分。操作包括增删改查、发布文章、发布分享、更新相册、更新昵称、更新个人简介、绑定 qq、绑定微信。

Comment：评论类

　属性包括 id、评论内容、评论状态、创建时间、评论者 id、被评论的文章 id、被评论的用户 id、被评论的话题 id。操作包括增删改查、显示。

Commentstatus：评论状态类

　属性包括 id、状态名、状态值。

Topic:话题类

　属性包括 id、话题标题。操作包括增删改查、显示。

Massage：私聊消息类

　属性包括 id、内容、发送者 id、接受者 id。

Tag：标签类

属性包括 id、标签名、标签出现次数。操作包括标签云显示。

Share：分享类

属性包括 id、分享者 id、标题、内容、标签、分享文件的储存路径。操作包括增删改查。

Follow：关注类

属性包括 id、关注者 id、被关注者 id。

Album：相册类

属性包括 id、图片存储路径、相册所有者 id。

Adminuser：后台管理员类

属性包括 id、管理员账号、密码 hash 值（随机生成用于加密）、密码（加密后）。

Auth_assignment：权限管理类

属性包括权限名、管理员 id。

Auth_item：权限分配类

属性包括权限名、权限值、权限描述。

Auth_item_child：权限分配子类

属性包括父权限名、子权限名。用于判断权限的上下级关系。

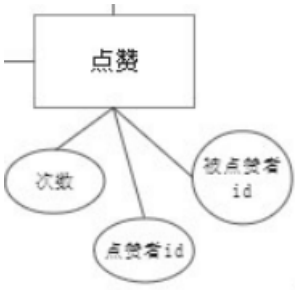
根据项目的需求可划分为如下描述的类图。用户类以 id 自增方式存储在用户表中，包含属性有用户名，MD5 密码，昵称以及积分等，操作包括设置昵称，发布评论，关注，绑定其他平台账号。评论管理类，用户可通过发布分享，以及评论分享，评论类中还有自己调用自己，是个自循环类，同样，包括管理员账户管理类。其中包括发布的分享的状态，以及评论的状态，管理员赋予权限设置以管理后台数据，以及产品运行的过程不法用户的恶意操作，做及时的屏蔽账户，以及普通用户忘记密码等需要后台介入的操作。

3.2 概念结构设计

3.2.1 实体与属性的定义

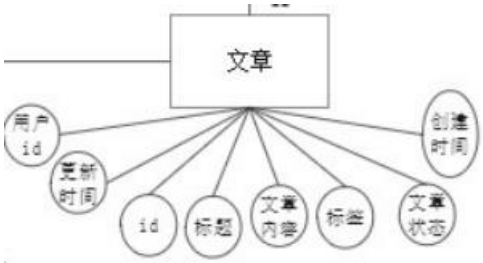
点赞模块

（自身 id，被点赞文章 id，点赞数）



文章模块

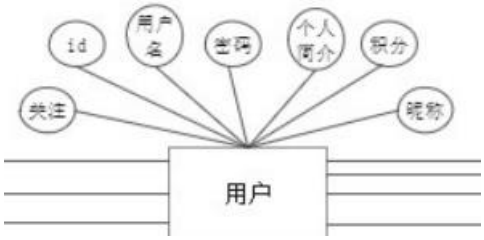
文章信息（自身 id，自增 id，文章标题，文章内容，标签，文章状态，创建时间，更新时间，用户 id，作品 url，作品评分(0-10)）



文章状态信息（自身 id，属性名，属性值）

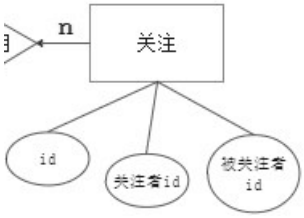
用户模块

（自增 id，用户名，密码 hash 值，随机值（校验码），昵称，个人简介，积分）



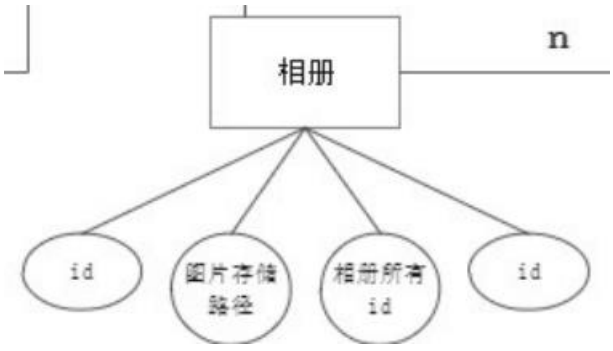
关注模块

（自增 id，关注者 id，被关注者 id）



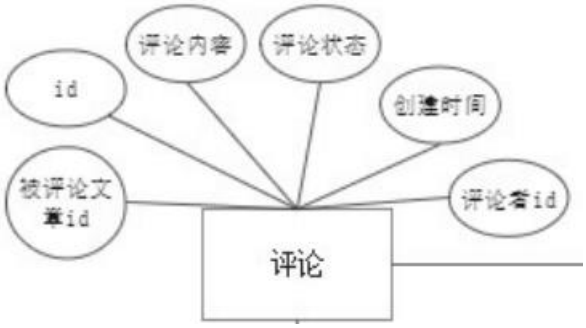
相册模块

（自增 id，所属用户 id，图片存储路径）



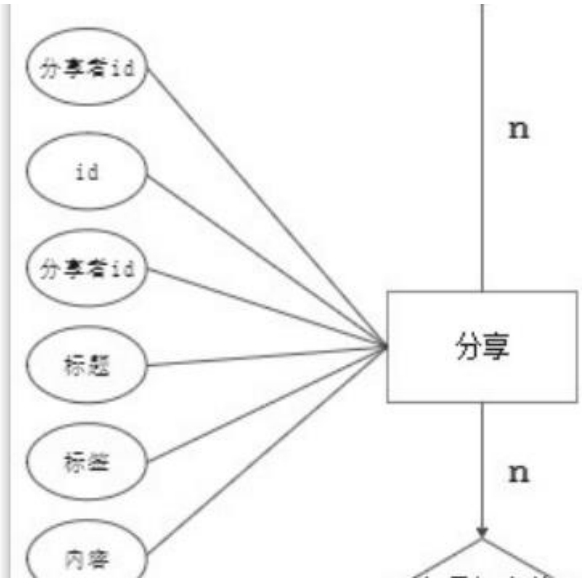
评论模块

评论信息（自增 id，评论内容，创建时间，评论者 id，被评论的用户 id，被评论的文章 id，被评论的话题 id）



评论状态信息（自增 id，状态名，状态值）

分享模块（自增 id，分享者 id，标题，内容，标签，分享文件的存储路径）



3.3 逻辑结构设计

3.3.1 模式

表结构设计

作品信息

表 work 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
title	varchar	128		是	作品标题

文章信息

表 post 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
title	varchar	128		是	文章标题

content	text			是	文章内容
tags	text			否	标签
status	int	4		否	文章状态
create_time	int	16		是	创建时间
update_time	int	16		是	更新时间
user_id	int	16		是	用户 id
url	text			是	作品 url
score	int	4		是	作品评分(0-10)

文章状态信息

表 poststatus 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4		否	自增 id
name	varchar	128		是	属性名
position				是	属性值

点赞信息

表 likes 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
user_id	int	4		是	点赞用户 id
post_id	int	4		是	被点赞的文章 id

用户信息

表 user 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
username	varchar	128		是	用户名
password_hash	varchar	256		是	密码 hash 值
auth_key	varchar	32		是	校验码
nickname	vachar	128		是	昵称
profile	text			否	个人简介
qq	varchar	128		否	qq 绑定信息
wx	varchar	128		否	微信绑定信息
points	int	4		是	积分

评论信息

表 comment 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
content	text			是	评论内容
status	int	4		是	评论状态
create_time	int	16		是	创建时间
from_user_id	int	4		是	评论者 id

to_user_id	int	4		否	被评论的用户 id
post_id	int	4		否	被评论的文章 id
topic_id	int	4		否	被评论的话题 id

评论状态信息

表 commentstatus 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
name				是	状态名
position				是	状态值

话题信息

表 topic 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
title	varchar	128		是	话题标题

私聊消息信息

表 massage 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
content	text			是	消息内容
to_user_id	int	4		是	发送者 id
from_user_id	int	4		是	接受者 id

标签信息

表 tag 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
name	varchar	128		是	标签名
frequency	int	16		是	标签出现次数

分享信息

表 share 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
user_id	int	4		是	分享者 id
title	varchar	128		是	标题
content	text			是	内容
tags	text			否	标签
route	varchar	128		是	分享文件的存储路径

管理员用户信息

表 adminuser 的结构:

字段名	数据类型	长度	主键	非空	描述
id	int	4	是	否	自增 id
username	varchar	128		是	管理员账号
password_hash	varchar	256		是	密码 hash 值
author_key	varchar	32		是	校验码

权限管理信息

表 auth_assignment 的结构:

字段名	数据类型	长度	主键	非空	描述
name	varchar	64	是	是	权限名
user_id	int	4		是	管理员 id

权限分配信息

表 auth_item 的结构:

字段名	数据类型	长度	主键	非空	描述
name	varchar	64	是	否	权限名
type	int	4		是	权限值
description	text			是	权限描述

权限分配父子关系信息

表 auth_item_child 的结构:

字段名	数据类型	长度	主键	非空	描述
parent	varchar	64	是	否	父权限名
child	varchar	64		是	子权限名

3.3.2 外模式

3.3.3 范式

1.为什么要使用自增 ID 作为主键

①.从业务上来说

在设计数据库时不需要费尽心思去考虑设置哪个字段为主键。然后是这些字段只是理论上是唯一的，例如使用图书编号为主键，这个图书编号只是理论上来说是唯一的，但实践中可能会出现重复的　　情况。所以还是设置一个与业务无关的自增 ID 作为主键，然后增加一个图书编号的唯一性约束。

②.从技术上来说

如果表使用自增主键，那么每次插入新的记录，记录就会顺序添加到当前索引节点的后续位置，当一页写满，就会自动开辟一个新的页。 总的来说就是可以提高查询和插入的性能。

2.对 InnoDB（ InnoDB，是 MySQL 的数据库引擎之一，现为 MySQL 的默认存储引擎 ）来说

①: 主键索引既存储索引值，又在叶子节点中存储行的数据，也就是说数据文件本身就是按照 b+树方式存放数据的。

②: 如果没有定义主键，则会使用非空的 UNIQUE 键做主键 ; 如果没有非空的 UNIQUE 键，则系统生成一个 6 字节的 rowid 做主键;

聚簇索引中，N 行形成一个页（一页通常大小为 16K)。如果碰到不规则数据插入时，为了保持 B+树的平衡，会造成频繁的页分裂和页旋转，插入速度比较慢。所以聚簇索

引的主键值应尽量是连续增长的值，而不是随机值(不要用随机字符串或 UUID)。
故对于 InnoDB 的主键，尽量用整型，而且是递增的整型。这样在存储/查询上都是非常高效的。

数据库设计满足 BCNF 范式：

①post 表：

主键只有 id，不存在主属性对于码的部分函数依赖，故属于 BCNF 范式。

②poststatus 表

主键通过 id 自增显示发布的可能状态，故无需改变表内容，使用自增 id，不存在主属性对于码的部分函数依赖，故属于 BCNF 范式。

③user 表

虽然用户名也是唯一的，为了设计方便，依然采用自增 id 作为主键，不存在主属性对于码的部分函数依赖，故属于 BCNF 范式。

3.4 物理结构设计

存储位置：默认位置

建立系统程序员视图，包括：

a．数据在内存中的安排，包括对索引区、缓冲区的设计；

b．所使用的外存设备及外存空间的组织，包括索引区、数据块的组织与划分

c．访问数据的方式方法。

4　系统安全和健壮性

4.1 输入输出项

注册页面

输入格式:

账号 11 位纯数字手机号

验证码为 4 位数字

输出格式:

不输入账号，提示“请输入账号”

不输入验证码,提示“请输入验证码“

账号不是 11 位手机号，提示“账号格式错误”

验证码不正确，提示”验证码错误”

登录页面

输入格式:

账号 11 位纯数字手机号

密码，6-20 位字符串，限定数字、字母、@!.#*

输出格式:

不输入账号，提示“请输入账号”

不输入密码,提示“请输入密码“

账号不是 11 位手机号，提示“账号格式错误”

密码不正确，提示”密码错误”

找回密码

输入格式:

账号 11 位纯数字手机号

验证码为 4 位数字

输出格式:

不输入账号，提示“请输入账号”

不输入验证码,提示“请输入验证码“

账号不是 11 位手机号，提示“账号格式错误”

验证码不正确，提示”验证码错误”

个人主页

输入格式:

昵称：1-10 位的字符串，不限数字，字母，符号

头像：jpg 格式的图片

个性签名“：1-50 位的字符串，不限数字，字母，符号

留言：1-200 位的字符串，不限数字，字母，符号

输出格式:

不输入昵称，提示“请输入昵称”

输入昵称过长，提示“昵称请在 10 字以内”

输入头像格式错误，提示“输入的头像图片为 jpg“

输入的个性签名过长，提示“个性签名请在 50 字以内“

不输入留言,提示“请输入留言“

输入留言过长，提示“留言请在 200 字以内“

新建日志

输入格式：

标题：1-20 位的字符串，不限数字，字母，符号

内容：长度不限的字符串

文件：pdf 格式文件

输出格式：

未输入标题，提示“请输入标题”

未输入内容，提示“请输入内容”

输入标题过长，提示“标题应在 20 字以内“

输入文件格式错误，提示“请输入 pdf 文件“

搜索用户

输入格式：

搜索框：1-10 位的字符串，不限数字，字母，符号

输出格式：

未输入搜索框，提示“请输入搜索框”

搜索框输入过长，提示“请输入 10 位以内的用户名“

搜索电影

输入格式：

搜索框：1-20 位的字符串，不限数字，字母

输出格式：

未输入搜索框，提示“请输入搜索框”

搜索框输入过长，提示“请输入 20 位以内的电影名“

搜索框输入符号，提示“输入框中包含非法字符”

搜索音乐

输入格式：

搜索框：1-20 位的字符串，不限数字，字母

输出格式：

未输入搜索框，提示“请输入搜索框”

搜索框输入过长，提示“请输入 20 位以内的音乐名“

搜索框输入符号，提示“输入框中包含非法字符”

搜索读书

输入格式：

搜索框：1-20 位的字符串，不限数字，字母

输出格式：

未输入搜索框，提示“请输入搜索框”

搜索框输入过长，提示“请输入 20 位以内的书名“

搜索框输入符号，提示“输入框中包含非法字符”

新建评论帖子

输入格式：

标题：1-20 位的字符串，不限数字，字母，符号

内容：长度不限的字符串

文件：不限文件格式

输出格式：

未输入标题，提示“请输入标题”
未输入内容，提示“请输入内容”
输入标题过长，提示“标题应在 20 字以内“

回复评论帖子

输入格式：
回复：1-200 位的字符串，不限数字，字母，符号
输出格式：
未输入回复，提示“请输入回复”
输入回复过长，提示“回复请在 200 字以内”

发表动态

输入格式：
回复：1-200 位的字符串，不限数字，字母，符号
输出格式：
未输入回复，提示“请输入动态”
输入动态过长，提示“发表动态请在 200 字以内”

新建资源帖子

输入格式：
标题：1-20 位的字符串，不限数字，字母，符号
内容：长度不限的字符串
文件：不限文件格式
文件可见：选择需要积分，全部可见
积分：当文件可见选择需要积分时，输入积分 1-20 个
输出格式：
未输入标题，提示“请输入标题”
未输入内容，提示“请输入内容”
输入标题过长，提示“标题应在 20 字以内“
未输入积分，提示“请输入查看所需积分”
输入积分过多，提示“积分最多需要 20 分”

回复资源帖子

输入格式：
回复：1-200 位的字符串，不限数字，字母，符号
输出格式：
未输入回复，提示“请输入回复”
输入回复过长，提示“回复请在 200 字以内”

评论话题

输入格式：
回复：1-200 位的字符串，不限数字，字母，符号
输出格式：
未输入评论，提示“请输入评论”
输入评论过长，提示“评论请在 200 字以内”

4.2 对常见 Web 攻击的预防

1.XSS 跨站脚本攻击

处理：用户提交的博客信息进行转义处理。防止被窃取用户的 cookie。

2.sql 注入

处理：使用预编译语句，即使使用 sql 语句伪造成参数，到了服务端的时候，伪造 sql 语句的参数也只是简单的字符，并不能起到攻击的作用。

4.3 数据库表设计遵循法则

1：字段的原子性

解释：保证每列的原子性，不可分解，意思表达要清楚，不能含糊，高度概括字段的含义，能用一个字段表达清楚的绝不使用第二个字段，必须要使用两个字段表达清楚的绝不能使用一个字段

2：主键设计

解释：主键不要与业务逻辑有所关联，最好是毫无意义的一串独立不重复的数字，常见的比如 UUID 或者将主键设置为 Auto_increment;

3：字段使用次数

解释：对于频繁修改的字段（一般是指状态类字段）最好用独立的数字或者单个字母去表示，不用使用汉字或长字符的英文

4：字段长度

解释：建表的时候，字段长度尽量要比实际业务的字段大 3-5 个字段左右（考虑到合理性和伸缩性），最好是 2 的 n 次方幂值。不能建比实际业务太大的字段长度（比如订单 id 如果考虑要业务增长的话，一定要使用 Long 型，对应的数据库的数据类型是 bigint），这是因为如果字段长度过大，在进行查询的时候索引在 B-Tree 树上遍历会越耗费时间，从而查询的时间会越久；但是绝对不能建小，否则 mysql 数据会报错，程序会抛出异常；

5：关于外键

解释：尽量不要建立外键，保证每个表的独立性。如果非得保持一定的关系，最好是通过 id 进行关联

6：动静分离

解释：最好做好静态表和动态表的分离。这里解释一下静态表和动态表的含义，静态表：存储着一些固定不变的资源，比如城市/地区名/国家(静态表一定要使用缓存)。动态表：一些频繁修改的表

7：关于 code 值

解释：使用数字码或者字母去代替实际的名字，也就是尽量把 name 转换为 code，因为 name 可能会变（万一变化就会查询处多条数据，从而抛出错误），但是 code 一般是不会变化的.另一方面，code 值存储的字符较少，也能减少数据库的存储空间的压力

8：关于 Null 值

解释：尽量不要有 null 值，有 null 值的话，数据库在进行索引的时候查询的时间更久，从而浪费更多的时间！可以在建表的时候设置一个默认值！

9：关于引擎的选择

解释：关于引擎的选择，innodb 与 myisam，myisam 的实际查询速度要比 innodb 快，因为它不扫面全表，但是 myisam 不支持事务，没办法保证数据的 Acid。选择哪个这就要看自己对于效率和数据稳定性方面的实际业务的取舍了

10：资源存储

解释：数据库不要存储任何资源文件，比如照片/视频/网站等，可以用文件路径/外链用来代替，这样可以在程序中通过路径，链接等来进行索引

11：与主键相关

解释：根据数据库设计三大范式，尽量保证列数据和主键直接相关而不是间接相关

12：关系映射

解释：多对一或者一对多的关系,关联一张表最好通过 id 去建立关系，而不是去做重复数据，这样做最大的好处就是中间的关系表比较清楚明白。

13：预留字段

解释：在设计一张表的时候应该预制一个空白字段，用于以后的扩展，因为你也不是确定这张表以后不会扩展。

14：留下单一字段确定是否可用

解释：通过一个单一字段去控制表是否可用，比如通常起名为 isVaild，预制的含义为 0 为有效，1 为无效，这样便于以后我们去剔除数据或者重整数据，使其成为 boolean 性质的数据　更加便于我们去操控。

15：删除字段

解释：数据库是禁止使用 delete 命令的,一般都不会真正删除数据，都是采用改状态的方式，设置 state 字段，通过修改状态赋予它是否有效的逻辑含义！

4.4 系统安全设计

4.4.1.数据传输安全性设计

使用 ssh2 实现远程上传、下载、执行命令。SSH 可以通过将联机的封包加密的技术进行资料的传递; 使用 SSH 可以把传输的所有数据进行加密，即使有人截获到数据也无法得到有用的信息。同时数据经过压缩，大大地加快了传输的速度。通过 SSH 的使用，可以确保资料传输比较安全并且传输效率较高。

4.4.2.应用系统安全性设计

操作人的操作信息需要提供操作记录。对系统的异常信息需进行记录，已备以后查看。只有授权用户才能登录系统，对于某个操作，需要具有相应权限才能进行操作。

4.4.3.数据存储安全性设计

对于用户的密码等敏感信息采用 MD5 进行加密。

5　权限设计

以角色为基础的权限管理设计（RABC）

5.1 基础表

见表设计 adminuser、auth_assignment、auth_item、auth_item_child 表

auth_assignment 存储管理员 id 和权限

auth_item 存储权限的信息

auth_item_child 存储权限的上下级关系

5.2 权限分配

当前用户只能分配已有权限的子权限。

创建新的管理员只能由唯一的超级管理员创建。

5.3 权限验证

权限控制在控制器里进行拦截。使用 yii2 框架的 ACF（访问控制过滤器）进行过滤。

附录

1、《构建之法现代软件工程》 人民邮电出版社 邹欣著.
2、《软件体系结构》 清华大学出版社 张友生著.
3、《软件工程基础》 北京邮电大学出版社 赵一丁著.