# 《数据库系统原理》大作业

## 系统设计报告

题目名称：　航吧-北航学生自己的贴吧


学号及姓名：　孙锐毅（组长）

2237350　黄弘烨

0　尹祺霖

2237143

0

22371157

2024年　　12月　　10日

# 一. 任务分工

| 学生姓名 | 工作内容 | | | 工作量占比 (组内同学总和为 1) |
|---|---|---|---|---|
| | **子任务 1：系统功能设计与数据库设计** | **子任务 2：系统服务器端开发** | **子任务 3：系统客户端开发** | |
| 孙锐毅 | 想法讨论<br>需求分析 | 接口文档敲定<br>前后端连接、数据传输 | 前后端代码测试 | 33.3% |
| 黄弘烨 | 想法讨论<br>需求分析 | / | 前端布局美化<br>前端架构实现<br>网页测试 | 33.3% |
| 尹祺霖 | 想法讨论<br>需求分析 | 接口文档敲定<br>数据库连接<br>数据库存储定义与实现 | / | 33.3% |

# 二. 需求分析

## 1. 需求描述

### 1.1 引言

**我们发现校园里许多同学有交流学术和生活经验的需求，尤其是在学习、就业和社交方面．很多 同学希望能够分享自己的学习资料、实习经历和生活技巧，但目前大家主要依赖于零散的聊天群和社 交媒体，这种方式存在信息难以集中和查找不便的问题．此外，交流的深度和质量也受到限制，难以 形成有效的互动和讨论。**

为了满足这一需求，我们设计了一个大学生论坛平台，旨在为同学们提供一个专属的交流空间。

这个平台将允许大家发帖讨论各种主题，比如学术问题、职业发展、校园活动和生活分享等，方便同学们在一个集中化的地方获取信息和建议。

# 2.功能分析

## 2.1 用户相关功能

- 用户注册

输入学号、用户名、邮箱和密码等信息。如果信息合法，则注册成功，用户将收到确认邮件。反之，将返回相应的错误提示。

- 用户登录

输入用户名和密码，若账户信息正确且未被封禁，则登录成功。若信息不正确，则报错

- 展示个人信息

在个人主页展示自己的信息。

- 更新个人信息

在个人主页修改自己的信息，包括修改头像、用户名、密码及其他个人信息。

- 发帖

用户可以发帖讨论感兴趣的话题，促进交流。

- 回复

用户可以回复其他用户的帖子，促进交流。

- 根据标签分类查看帖子

用户可以通过选择特定的标签，快速获取对应主题的帖子，提升信息检索的效率。

- 查看与管理自己的帖子

用户可以查看自己发布的帖子，删除自己发布的帖子。

## 2.2 经验值功能

- 经验值获取

用户在论坛内发帖、回复等行为都可获得相应的经验值。

不同的行为可获得不同数量的经验值。

- 用户等级系统

根据用户的经验值，系统将用户划分为不同的等级，不同等级能展现出用户的活跃程度

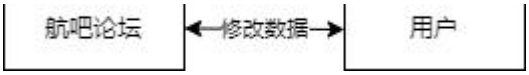
设定等级晋升机制，用户达到特定经验值后自动升级。

- 经验值排行榜

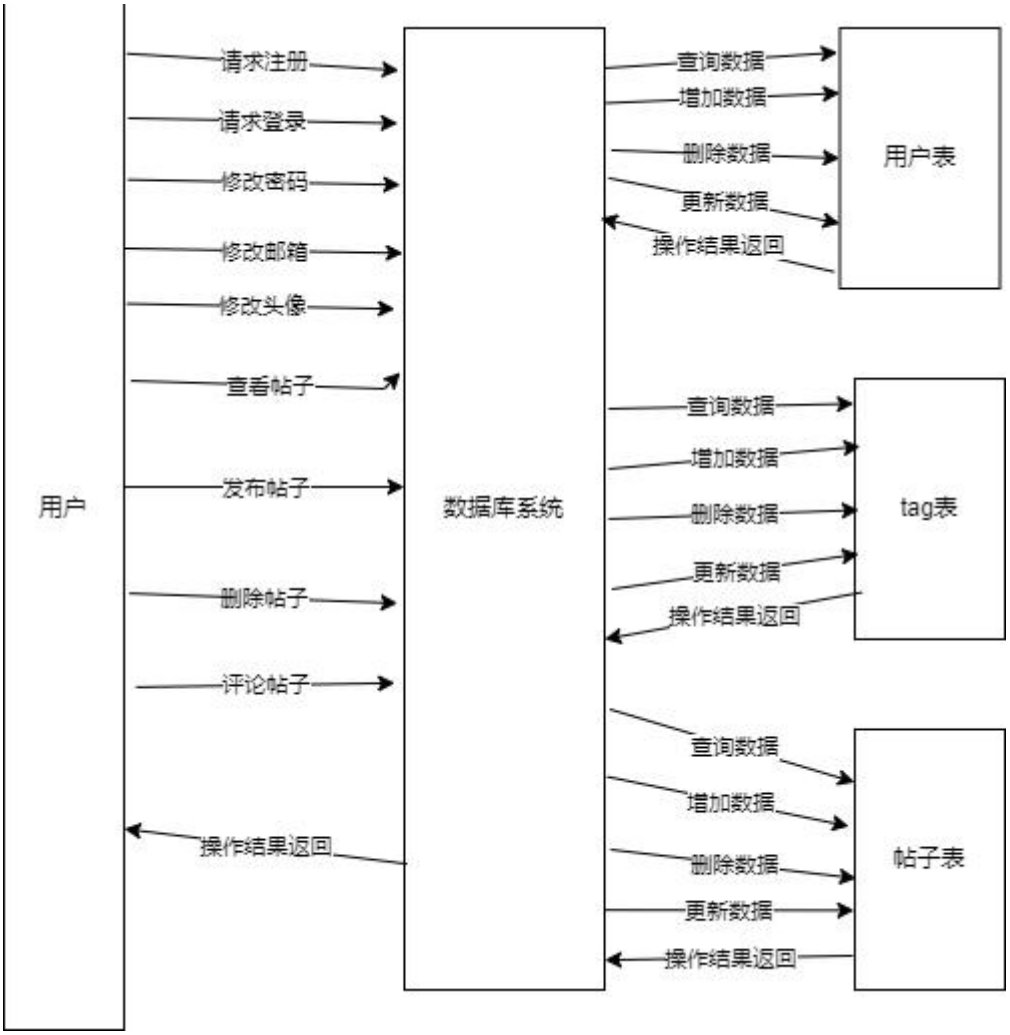
经验值前三的用户会在主页进行展示，激励大家多活跃。

## 三 .数据流图

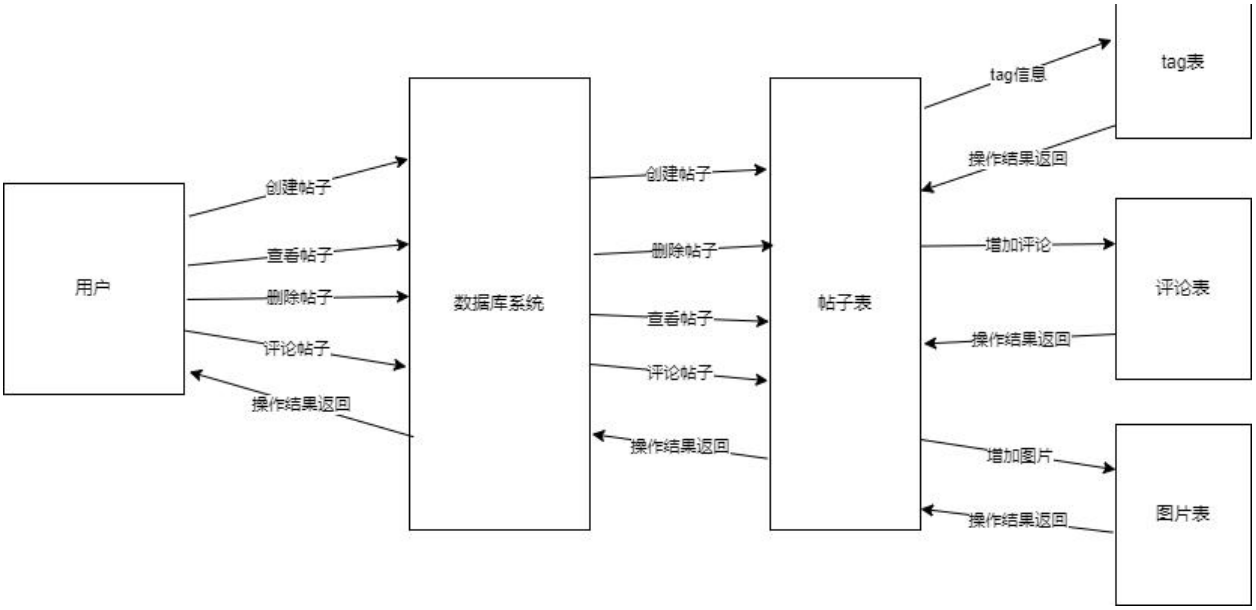
### 1. 顶层数据流图




### 2.用户数据流图



## 3.帖子数据流图


用户
创建帖子
查看帖子
删除帖子
评论帖子
操作结果返回
数据库系统
创建帖子
删除帖子
查看帖子
评论帖子
操作结果返回
帖子表
tag信息
操作结果返回
增加评论
操作结果返回
增加图片
操作结果返回
tag表
评论表
图片表

# 四、数据库基本表定义

## 4.1 用户表

| 属性名 | 中文 | 数据类型 | 备注 |
|---|---|---|---|
| user_id | ID | bigint | 主键 |
| user_name | 用户名 | varchar | |
| user_email | 用户邮箱 | varchar | |
| user_password | 用户密码 | varchar | |
| user_student_id | 用户学号 | bigint | |
| user_experience | 用户经验 | bigint | |
| user_sign_date | 注册时间 | date | |
| user_birthday | 生日 | date | |
| user_uploaded | 头像上传状态 | int | |
| user_user_role | 用户角色（管理员等） | int | |
| user_post_cnt | 发表贴数 | int | |
| user_info_cnt | 提示消息数 | int | |
| user_introducntion | 用户简介 | text | |

## 4.2 帖子表

| 属性名 | 中文 | 数据类型 | 备注 |
|---|---|---|---|
| post_id | ID | bigint | 主键 |
| post_title | 帖子标题 | varchar | |
| post_content | 帖子内容 | text | |
| post_tag_id | 帖子标签 | bigint | 外键 |
| post_heat | 帖子热度 | int | |
| post_time | 帖子创建时间 | date | |
| post_user_id | 帖子作者ID | bigint | 外键 |
| post_isTop | 是否置顶 | bool | |

## 4.3 一级评论表

| 属性名 | 中文 | 数据类型 | 备注 |
| --- | --- | --- | --- |
| FLC_id | ID | bigint | 主键 |
| FLC_content | 评论内容 | text | |
| FLC_time | 评论创建时间 | date | |
| FLC_post_id | 评论的帖子ID | bigint | 外键 |
| FLCuser_id | 评论发布者ID | bigint | 外键 |

## 4.4 图片表

| 属性名 | 中文 | 数据类型 | 备注 |
| --- | --- | --- | --- |
| PC_id | ID | bigint | 主键 |
| PC_path | 图片路径 | varchar | |
| PC_category | 类型，贴图/用户图 | int | |
| PC_author_id | 上传者id | bigint | 外键 |

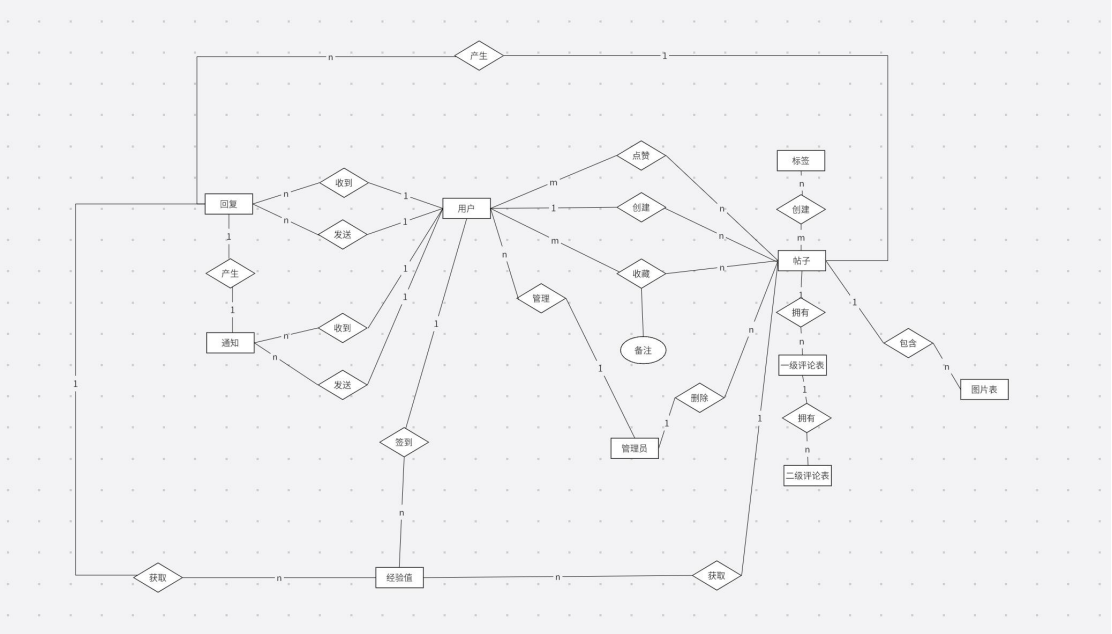
## 4.5 标签表

| 属性名 | 中文 | 数据类型 | 备注 |
| --- | --- | --- | --- |
| LB_id | ID | bigint | 主键 |
| LB_tag_name | 频道名 | varchar | |

## 4.6 通知表

| 属性名 | 中文 | 数据类型 | 备注 |
| --- | --- | --- | --- |
| IF_id | ID | bigint | 主键 |
| IF_content | 通知内容 | text | |
| IF_receiver_id | 收到内容的id | bigint | 外键 |
| IF_sender_id | 发送内容的id | bigint | 外键 |

## 4.7 E-R图




# 4.8 数据库模式设计

## 数据库关系模式

1. 用户表(id, username, email, password, student_id, experience, sign_date, birthday, uploaded, user_role, post_cnt, info_cnt)

2. 帖子表(id, title, content, category, heat, created_at, user_id, isTop)

3. 一级评论表(id, content, created_at, post_id, user_id)

4. 图片表(id, path, category, user_id)

5. 通知表(id, content, receiver_id, sender_id)

6. 回复表(id, content, receiver_id, sender_id, post_id, comment_date)

7. 标签表(id, tag_name)

## 关系模式范式等级的判定与规范化

- 用户表(id，username，email，password，student_id，experience，sign_date,birthday,uploaded,role,post_cnt,user_info_cnt,user_introduction) 主键是用户的id，可以看到都是完全依赖于用户的id的信息，不存在外键，另一个候选码是student_id,这两个都是单一属性，不存在部份依赖，非主属性之间没有依赖，所以肯定是3NF。

- 帖子表(id，title，content，tag_id，heat，time， user_id,isTop) 主键是帖子的id，没有其他候选码，一定是NF，user_id和tag_id是外键，非主属性不传递依赖id，因此一定是BCNF。
- 评论表(id，content，time，post_id，author_id) 每个一级评论有唯一的id作为主键，没有其他候选码，所以是2NF，所有的函数依赖左边都有码，因此是BCNF
- 标签表(id，tag_name) 每个标签有唯一的id作为主键，没有其他候选码，所以是2NF，所有的函数依赖左边都有码，因此是BCNF
- 头像图片表(id，path，category，author_id) 每个头像图片有唯一的id作为主键，没有其他候选码，所以是2NF，所有的函数依赖左边都有码，因此是BCNF
- 通知表(id，content，receiver_id，sender_id) 每个通知有唯一的id作为主键，没有其他候选码，所以是2NF，其中receiver_id和sender_id是外键，所有的函数依赖左边都有码，因此是BCNF

# 五、数据库设计优化与索引

## 5.1 存取方法

存储方法采用了Django中的save方法，取的话是用的objects.filter方法取出

## 5.2 优化

首先我们的数据库包含了范式化的优化，都规范到了3NF，其次，一个用户可以有多个帖子，一个帖子可以有多个评论，一个评论可以有多个图片，这些都是1：n的关系，对这些关系，可以通过在n端实体中加入1端实体的id，避免建立1:n联系表。此外，用户-头像为1:1联系，因此在头像表中加入用户id，避免建立用户-头像联系。

## 5.3 索引的建立

由于本系统中所有关系模式均属于3NF，故而对主键建立索引。