数据库设计说明书

1. 引言

1.1 目的

本数据库设计说明书旨在详细描述 AetherNet 校园互助平台所使用的数据库设计方案。通过此文档，开发团队能够理解数据库结构、数据表之间的关系及数据模型，为系统的开发与数据存储提供有效的支持。

1.2 范围

此数据库设计方案涵盖了 AetherNet 平台所需的核心数据模型及表结构，包括用户信息、帖子管理、任务管理、评论互动等各类数据的存储与关系。同时，本说明书也描述了数据库的安全性、优化策略以及数据管理规则。

1.3 定义与缩写

- **AetherNet**：校园互助平台的名称。
- **CRUD**：增删改查（Create, Read, Update, Delete）操作，指对数据表的基本操作。
- **ORM**：对象关系映射（Object-Relational Mapping），将对象与数据库中的记录进行映射。
- **索引**：数据库中一种特殊的查询加速数据结构，用于提升查询性能。

2.数据库设计概述

2.1 设计目标

- 设计易于扩展和维护的数据库结构，支持平台功能的高效运行。
- 保证数据的一致性、完整性和安全性。
- 确保平台在高并发的情况下仍能高效地进行数据存取。

2.2 技术选择

- **数据库管理系统（DBMS）**：MySQL
- **数据库引擎**：InnoDB（支持事务、外键约束）

- **数据存储格式**：支持 JSON 格式存储，如任务描述、评论内容等。
- **数据库优化**：针对常用查询进行索引设计，并考虑查询性能的优化。

---

3. 数据库表设计

以下是数据库中主要表格的设计说明，包括表结构、字段及其功能说明。

3.1 用户表（Users）

- **表描述**：该表存储平台用户的基本信息。
- **字段设计**：
  - `user_id` (INT, PRIMARY KEY, AUTO_INCREMENT): 用户的唯一标识符。
  - `username` (VARCHAR(255), NOT NULL): 用户名。
  - `email` (VARCHAR(255), UNIQUE, NOT NULL): 用户的邮箱地址，必须唯一。
  - `password_hash` (VARCHAR(255), NOT NULL): 存储加密后的密码。
  - `role` (ENUM('student', 'admin'), NOT NULL): 用户角色，区分学生与管理员。
  - `created_at` (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP): 用户的注册时间。

3.2 帖子表（Posts）

- **表描述**：存储用户发布的帖子内容。
- **字段设计**：
  - `post_id` (INT, PRIMARY KEY, AUTO_INCREMENT): 帖子的唯一标识符。
  - `user_id` (INT, FOREIGN KEY): 外键，关联用户表（Users）中的 `user_id`。
  - `title` (VARCHAR(255), NOT NULL): 帖子的标题。
  - `content` (TEXT, NOT NULL): 帖子的内容。
  - `tags` (JSON): 帖子标签，存储与帖子相关的标签数组，如 ["二手", "学习", "代拿"]。
  - `status` (ENUM('pending', 'approved', 'rejected', 'deleted'),

DEFAULT 'pending'): 帖子的审核状态。

- created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP): 帖子发布的时间。

3.3 任务表（Tasks）

- **表描述**：该表存储平台中的任务信息，支持代拿服务等功能。
- **字段设计**：
  - task_id (INT, PRIMARY KEY, AUTO_INCREMENT): 任务的唯一标识符。
  - user_id (INT, FOREIGN KEY): 外键，关联用户表（Users）中的user_id，表示任务的发布者。
  - title (VARCHAR(255), NOT NULL): 任务标题。
  - description (TEXT): 任务描述。
  - reward (DECIMAL(10, 2)): 任务的酬劳。
  - status (ENUM('open', 'in_progress', 'completed', 'cancelled'), DEFAULT 'open'): 任务状态。
  - assigned_to (INT, FOREIGN KEY): 外键，关联任务接单用户的user_id。
  - created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP): 任务发布的时间。

3.4 评论表（Comments）

- **表描述**：该表存储用户对帖子的评论。
- **字段设计**：
  - comment_id (INT, PRIMARY KEY, AUTO_INCREMENT): 评论的唯一标识符。
  - post_id (INT, FOREIGN KEY): 外键，关联帖子表（Posts）中的post_id。
  - user_id (INT, FOREIGN KEY): 外键，关联用户表（Users）中的user_id，表示评论的发布者。
  - content (TEXT): 评论内容。
  - created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP): 评论发布的时间。

3.5 标签表（Tags）

- **表描述**：存储与帖子相关的标签。
- **字段设计**：
  - `tag_id` (INT, PRIMARY KEY, AUTO_INCREMENT): 标签的唯一标识符。
  - `tag_name` (VARCHAR(255), NOT NULL): 标签名称。
  - `post_id` (INT, FOREIGN KEY): 外键，关联帖子表（Posts）中的 `post_id`。

3.6 审核日志表（ModerationLogs）

- **表描述**：存储平台审核内容的日志记录，帮助管理员追踪内容审核历史。
- **字段设计**：
  - `log_id` (INT, PRIMARY KEY, AUTO_INCREMENT): 审核日志的唯一标识符。
  - `post_id` (INT, FOREIGN KEY): 外键，关联帖子表（Posts）中的 `post_id`。
  - `decision` (ENUM('approved', 'rejected', 'pending')): 审核结果，表示帖子是否通过审核。
  - `risk_level` (ENUM('low', 'medium', 'high')): 审核时的风险级别。
  - `reviewer_id` (INT, FOREIGN KEY): 外键，关联管理员（用户表中的 `user_id`），表示审核操作的执行人。
  - `created_at` (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP): 审核日志的创建时间。

3.7 数据模型与关系

以下是系统中的表与表之间的关系：

- 用户与帖子表通过 `user_id` 关联，一个用户可以发布多个帖子（1:N 关系）。
- 帖子与评论表通过 `post_id` 关联，一个帖子可以有多个评论（1:N 关系）。
- 用户与评论表通过 `user_id` 关联，一个用户可以发表评论（1:N 关系）。
- 帖子与标签表通过 `post_id` 关联，一个帖子可以有多个标签（1:N 关

系）。

- 帖子与审核日志表通过 `post_id` 关联，记录每次审核的日志（1:N 关系）。

---

4. 数据库优化与安全设计

4.1 索引设计

为了提高查询效率，以下字段将设置索引：

- `user_id`：在用户表、帖子表、评论表中设置索引，用于快速查找用户发布的帖子和评论。
- `post_id`：在评论表、审核日志表、标签表中设置索引，用于快速查找与帖子相关的评论、审核记录和标签。

4.2 数据安全与加密

- **密码加密**：用户密码使用 AES 加密存储。
- **数据传输加密**：平台所有数据传输采用 HTTPS 协议进行加密，确保信息安全。

---

5. 结语

通过本数据库设计说明书，我们为平台提供了清晰的数据库结构及数据表设计，确保系统数据的高效存储、查询与管理。数据库的设计将支持平台的快速发展和扩展，确保能够满足未来更多功能的需求。