# **基于 Hadoop 云盘系统数据库设计文档**

目录

[基于 Hadoop 云盘系统数据库设计文档 1](#_Toc31396)

[1. 数据库整体设计概述 2](#_Toc19687)

[2.文档范围 2](#_Toc119)

[3. 术语与缩写解释 2](#_Toc22125)

[4.数据库环境说明 3](#_Toc29574)

[5. 数据库的命名规则 3](#_Toc11390)

[5.1 表名规则： 3](#_Toc18748)

[5.2 字段名规则： 4](#_Toc29376)

[6. 逻辑设计 4](#_Toc13033)

[7. 物理设计 5](#_Toc18559)

[7.1 表汇总 5](#_Toc8444)

[7.2 用户表（system\_users） 6](#_Toc14709)

[7.3 文件表（system\_hadoop\_file） 7](#_Toc296)

[8. 安全性设计 9](#_Toc20968)

[8.1 访问控制策略 9](#_Toc5774)

[8.2 数据加密方案 9](#_Toc22683)

[8.3 角色与权限 9](#_Toc1124)

[9. 性能优化 10](#_Toc1383)

[9.1 索引优化 11](#_Toc9910)

[9.2 数据缓存 11](#_Toc373)

[9.3 数据库配置优化 11](#_Toc29223)

[10. 数据库管理与维护说明 11](#_Toc9840)

[10.1 备份策略 11](#_Toc6891)

[10.2 监控指标 12](#_Toc23595)

[10.3 故障恢复 13](#_Toc1095)

## **数据库整体设计概述**

本数据库设计紧密围绕云盘系统的功能需求，采用MySQL 8.0 作为数据库管理系统，用于存储用户信息、文件元数据、分享记录等关键数据。通过合理的表结构设计与索引优化，确保数据存储的高效性、一致性和可扩展性，同时满足系统对安全性和事务处理的要求。

## **2.文档范围**

涵盖用户管理和文件存储相关模块，详细定义相关表结构及设计要点。

主要包括表结构设计、字段属性、约束条件、安全与性能相关设计等内容，不涉及未在表中体现的业务逻辑及功能模块。

## **3. 术语与缩写解释**

| **缩写、术语** | **解 释** |
| --- | --- |
| HDFS | Hadoop分布式文件系统，用于存储文件实体，数据库存储文件元数据 |
| BIGINT | 数据库中的一种整数类型，占用8字节，可表示较大范围的整数 |
| VARCHAR | 可变长度字符类型，用于存储字符串数据 |
| TINYINT | 小整数类型，占用1字节，常用于表示取值范围较小的整数或布尔值 |
| DATETIME | 用于存储日期和时间的数据类型 |

## **4.数据库环境说明**

****数据库系统****：MySQL 8.0.32（支持事务、外键约束及高并发场景）

****设计工具****：Navicat Premium 16（ER 图设计、SQL 脚本生成）

****编程工具****：Java + MyBatis-Plus（后端数据持久化）

****部署环境****：服务器：Linux CentOS 7.9

内存：16GB

存储：SSD 512GB（系统盘）+ HDD 4TB（数据盘）

## **数据库的命名规则**

### **表名规则：**

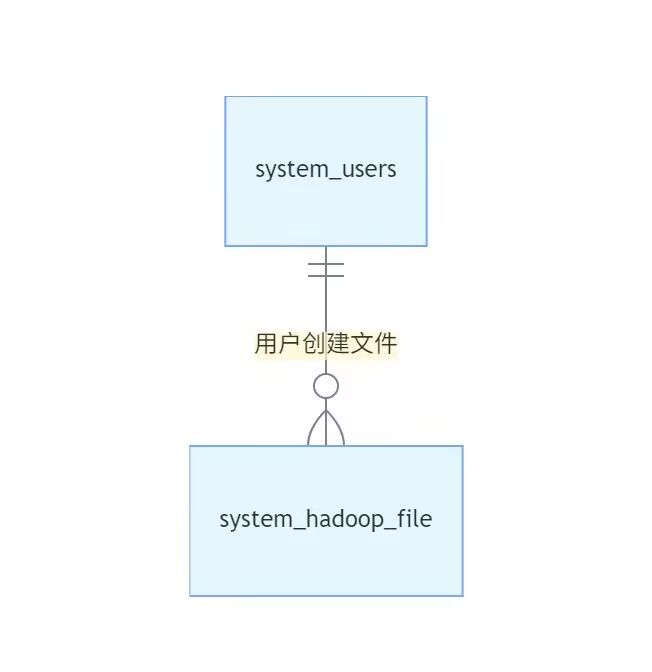
采用小写英文单词组合，必要时使用下划线分隔，如system\_users、system\_hadoop\_file ，体现业务归属及功能特性

### 5.2 字段名规则：

字段名使用小写英文单词，采用驼峰命名法或下划线分隔，如userName或user\_name。主键一般命名为id ，外键命名体现关联关系，如user\_id。时间相关字段采用create\_time（创建时间）、update\_time（更新时间） 等命名方式。

## 逻辑设计

ER关系图



## 7. 物理设计

### 7.1 表汇总

| 表名 | 功能描述 | 主键 | 外键关联 |
| --- | --- | --- | --- |
| system\_users | 用户账号及相关信息存储 | id | - |
| system\_hadoop\_file | Hadoop 云盘文件元数据存储 | id | id（关联 system\_users 表的用户 ID） |

### 7.2 用户表（system\_users）

| 字段名 | 数据类型 | 长度 | 非空 | 虚拟 | 键 | 注释 |
| --- | --- | --- | --- | --- | --- | --- |
| id | bigint | 20 | √ | × | 1 | 用户 ID |
| username | varchar | 30 | √ | × |  | 用户账号 |
| password | varchar | 100 | √ | × |  | 密码 |
| nickname | varchar | 30 | √ | × |  | 用户昵称 |
| email | varchar | 50 | × | × |  | 用户邮箱 |
| mobile | varchar | 11 | × | × |  | 手机号码 |
| sex | tinyint | 4 | × | × |  | 用户性别 |
| avatar | varchar | 512 | × | × |  | 头像地址 |
| status | tinyint | 4 | √ | × |  | 帐号状态（0 正常 1 停用） |
| login\_ip | varchar | 50 | × | × |  | 最后登录 IP |
| login\_date | datetime | - | × | × |  | 最后登录时间 |

补充说明：为username字段添加唯一索引，确保用户名的唯一性，加速用户登录及查询操作。考虑为status字段添加索引，便于快速筛选出正常或停用状态的用户。

### **7.3 文件表（system\_hadoop\_file）**

| 字段名 | 数据类型 | 长度 | 非空 | 虚拟 | 键 | 注释 |
| --- | --- | --- | --- | --- | --- | --- |
| id | bigint | 20 | √ | × | 1 | 用户 ID |
| creator | varchar | 64 | × | × |  | 创建者 |
| create\_time | datetime | - | √ | × |  | 创建时间 |
| updater | varchar | 64 | × | × |  | 更新者 |
| update\_time | datetime | - | √ | × |  | 更新时间 |
| type | tinyint | 1 | × | × |  | 文件类型 |
| name | varchar | 500 | × | × |  | 文件名 |
| catalogue | varchar | 2000 | × | × |  | 父级目录 |
| size | varchar | 255 | × | × |  | 大小 (MB) |

补充说明：为id字段建立主键索引，确保数据的唯一性和快速检索。考虑在type、name、catalogue字段上建立联合索引，提升文件按类型、名称及目录的查询效率。

## 8. 安全性设计

### **8.1 访问控制策略**

网络层面：数据库服务器设置防火墙规则，仅允许授权的后端服务器通过特定内网 IP 进行连接访问，禁止公网直接访问数据库端口。

用户权限层面：为应用程序分配特定权限的数据库用户，仅授予其执行必要操作（如SELECT、INSERT、UPDATE、DELETE）的权限，严格限制危险操作（如DROP、CREATE）。对于数据库管理用户，采用强密码策略，并定期更换密码，同时操作需进行审计和记录。

### **8.2 数据加密方案**

用户密码：采用强加密算法（如SHA - 256）结合盐值（可根据用户ID等信息生成唯一盐值）对用户密码进行加密存储，防止密码明文泄露。

敏感信息：对于用户邮箱、手机号码等敏感信息，可考虑在存储时进行加密处理（如使用AES加密算法 ），在应用层进行解密使用。

### 8.3 角色与权限

| 角色 | 表名 | 操作权限 | 说明 |
| --- | --- | --- | --- |
| 普通用户 | system\_users | SELECT（自身记录）、UPDATE（自身记录部分字段） | 普通用户可查看和修改自身部分信息，如修改密码、邮箱等 |
|  | system\_hadoop\_file | SELECT、INSERT、UPDATE、DELETE（自身创建的文件） | 可对自己创建的文件进行操作，如上传、下载、修改、删除 |
| 管理员 | system\_users | SELECT、INSERT、UPDATE、DELETE | 管理员可管理所有用户信息，包括创建、修改、删除用户等操作 |
|  | system\_hadoop\_file | SELECT、INSERT、UPDATE、DELETE | 可对所有文件进行操作，包括对系统文件的管理 |

## 9. 性能优化

### **9.1 索引优化**

除上述已提及的索引外，定期评估和分析数据库查询语句，根据实际查询需求，适时添加或调整索引，避免过多索引导致写入性能下降。

### **9.2 数据缓存**

考虑在应用层引入缓存机制（如Redis），对高频访问的数据（如热门文件元数据、用户登录信息等）进行缓存，减少数据库的查询压力，提升系统响应速度。

### **9.3 数据库配置优化**

根据服务器硬件资源和业务数据量，合理调整MySQL数据库的配置参数，如缓冲区大小、连接池大小等，以优化数据库的性能表现。

## 10. 数据库管理与维护说明

### 10.1 备份策略

全量备份：定期（如每周一次）在业务低峰期执行全量备份，使用mysqldump命令将数据库数据备份到指定存储位置，保留一定周期（如4周）的备份文件，以便在需要时进行数据恢复。

增量备份：启用MySQL的二进制日志（Binlog ）功能，记录数据库的所有变更操作，每小时或根据实际需求将Binlog文件备份到远程存储，用于在全量备份基础上进行数据的增量恢复。

### 10.2 监控指标

|  |  |  |  |
| --- | --- | --- | --- |
| 指标 | 监控工具 | 阈值 | 告警动作 |
| QPS（每秒查询率） | [具体监控工具，如MySQL 自带监控或第三方监控工具] | 高于 [具体数值，根据业务情况设定] | 检查是否存在异常查询或业务高峰，必要时进行优化或扩容 |
| 慢查询率 | [具体监控工具] | 高于 [具体百分比数值，如 1%] | 分析慢查询语句，检查索引是否缺失或需要优化 SQL |
| 磁盘利用率 | [服务器监控工具] | 高于 [具体百分比数值，如 80%] | 清理无用数据或考虑扩容存储设备 |
| 连接数 | [具体监控工具] | 接近数据库最大连接数 | 检查应用程序的数据库连接管理是否存在问题 |

### 10.3 故障恢复

**数据恢复**：当发生数据丢失或损坏时，根据备份策略，先使用最近的全量备份进行恢复，再通过增量备份（Binlog）将数据恢复到故障发生前的状态。

**服务器故障**：若数据库服务器发生硬件故障，尽快切换到备用服务器，并将备份数据恢复到备用服务器上，确保业务的连续性。在主服务器修复后，重新同步数据并将其重新纳入数据库集群（如有集群部署）。