

本科毕业论文（设计）

数据库设计说明书

|  |  |
| --- | --- |
| **学 生 姓 名** | **唐建国** |
| **学号** | **2020131040** |
| **专业** | **区块链工程** |
| **年级班级** | **2020级201班** |
| **指导教师** | **高琳（讲师）** |
| **所在学院** | **区块链产业学院** |
| **提交日期** | **2024年1月12日** |

2023 年 4 月

成都信息工程大学 区块链产业学院

目录

[本科毕业论文（设计） 1](#_Toc28004)

[数据库设计说明书 1](#_Toc25124)

[1 引言 - 1 -](#_Toc30689)

[1.1 编写目的 - 1 -](#_Toc3106)

[1.2 背景 - 2 -](#_Toc31401)

[1.3 术语 - 2 -](#_Toc29620)

[1.4 参考资料 - 3 -](#_Toc29558)

[2 需求分析 - 4 -](#_Toc29056)

[2.1 数据流图 - 4 -](#_Toc18270)

[2.2 数据字典 - 5 -](#_Toc7293)

[3 E-R模型设计 - 7 -](#_Toc13820)

[4 数据库实现 - 7 -](#_Toc32231)

[4.1 数据库命名约定和环境 - 7 -](#_Toc1630)

[4.1.1 命名约定 - 7 -](#_Toc13318)

[4.1.2 数据库环境 - 8 -](#_Toc26886)

[4.2 数据库关系图 - 9 -](#_Toc31453)

[4.3 数据表信息 - 10 -](#_Toc29553)

[4.3.1 表列表 - 10 -](#_Toc3679)

[4.3.2 表1 - 10 -](#_Toc17091)

[4.3.3 表2 - 11 -](#_Toc7291)

[4.3.4 表3 - 11 -](#_Toc22360)

[4.4 存储过程信息 - 11 -](#_Toc2209)

[5 数据库安全设计 - 12 -](#_Toc24921)

# 引言

## 编写目的

本数据库设计说明书的编写旨在提供面向高校学分认证的区块链应用系统的设计和实现细节。它的目的包括：

1. 提供系统设计的详细说明：该文档将详细描述面向高校学分认证的区块链应用系统的数据库设计。它将涵盖数据库结构、表和字段定义、数据关系和约束等方面的设计细节。

2. 指导系统开发和实施：该文档将为系统开发人员提供指导，以确保能够根据所描述的设计规范正确地开发和实施数据库。

3. 作为技术文档的参考：该文档将作为技术文档的参考，供未来的维护团队、系统管理员和其他相关人员查阅。它将提供数据库结构和设计的全面视图，以便更好地理解系统的数据模型和功能。

预期读者：

本文档的预期读者包括但不限于以下人员：

- 系统架构师：他们负责系统整体设计和架构，需要了解数据库设计的细节以确保其与整体系统设计的一致性。

- 开发人员：他们负责实现系统功能和业务逻辑，需要根据数据库设计进行编码和开发工作。

- 数据库管理员：他们负责数据库的管理和维护工作，需要了解数据库结构和关系，以便进行性能优化、备份恢复等操作。

- 系统测试人员：他们负责验证系统功能和性能，需要了解数据库设计以编写有效的测试用例。

- 项目经理和利益相关者：他们需要了解系统的技术细节和实现方案，以便进行项目管理和决策。

## 背景

本节将描述面向高校学分认证的区块链应用系统的背景，包括以下内容：

1) 需开发的软件系统的名称、英文缩写和项目编号（可选）；

- 软件系统名称：高校学分认证区块链应用系统

- 英文缩写（可选）：UCAS（University Credit Authentication System）

2) 任务提出者和开发者：

- 任务提出者：本系统是由一位论文指导老师提出的项目，其论文指导老师对学生的研究方向和项目提出进行指导和支持。

- 开发者：该系统的开发者为该毕业设计学生自己。学生将负责系统的设计、开发和实施，并在论文指导老师的指导下完成系统的相关工作。

3) 软件系统应用范围和用户：

- 应用范围：该系统旨在解决高校学分认证的问题，使用区块链技术确保学分认证的透明性、安全性和可靠性。

- 用户：主要包括以下几类用户：

- 学生：申请学分认证、查看学分认证状态、管理个人学分记录。

- 高校教务部门：审核学生的学分认证申请、管理学生学分记录。

- 企业或组织：查询学生的学分认证状态和学术成绩，用于招聘和选拔。

- 第三方认证机构：参与学分认证的机构，将认证结果上链。

该系统的目标是提供一个去中心化、透明且不可篡改的学分认证平台，改善传统学分认证过程中的信任问题和操作繁琐性，为学生、高校和用人单位提供更高效、可靠的学分认证服务。同时，该系统也是毕业设计项目的一部分，旨在展示学生在区块链应用系统设计与实现方面的能力和研究成果。

## 术语

1) 区块链（Blockchain）：一种去中心化的分布式账本技术，通过分布式节点之间的共识机制，实现对交易和数据的不可篡改记录和验证。

2) 学分认证（Credit Authentication）：指高校对学生所修读的课程和取得的成绩进行确认和认可的过程。

3) 高校教务部门（University Academic Affairs Office）：负责学校教务管理和学生学籍管理的部门，负责审核和管理学生的学分认证申请和学分记录。

4) 用户界面（User Interface，UI）：系统与用户之间进行交互和信息传递的界面，包括视觉设计和用户操作方式。

5) 智能合约（Smart Contract）：一种以计算机代码形式编写的自动化合约，基于区块链技术实现交易和合约条款的自动执行和验证。

6) 去中心化（Decentralized）：指系统中的数据和决策权分散在多个节点或参与方之间，而非集中在单一机构或服务器上。

7) 透明性（Transparency）：指系统操作和数据的可见性和可追溯性，用户可以查看和验证系统的运行状态和数据记录。

8) 安全性（Security）：指系统的抵御攻击、保护数据和用户隐私的能力，包括数据加密、身份验证和防止篡改等措施。

9) 可靠性（Reliability）：指系统的稳定性和可信度，能够持续提供准确和可用的服务，并对错误和故障有恢复和容错机制。

10) 操作繁琐性（Operational Complexity）：指传统学分认证过程中，需要学生和教务部门进行大量繁复的操作和文件交换，增加了流程的复杂性和工作量。

## 参考资料

本节列出用得着的参考资料，如：

1. 本项目经核准的计划任务书或合同、上级机关的批文；
2. 属于本项目的其他已发表的文件；
3. 本文件中各处引用的文件、资料、包括所要用到的软件开发标准。
4. 行业标准和规范。
5. 列出这些文件资料的标题、文件编号、发表日期和出版单位。

# 需求分析

## 数据流图

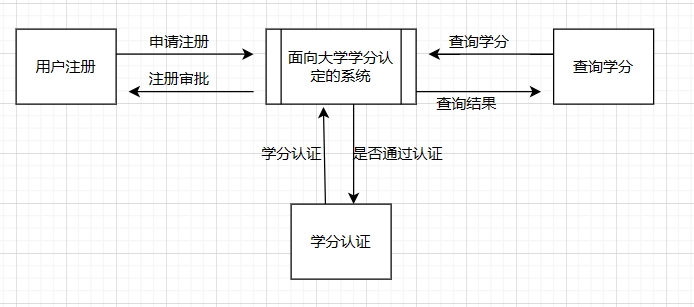


图2‑1 顶层数据流图

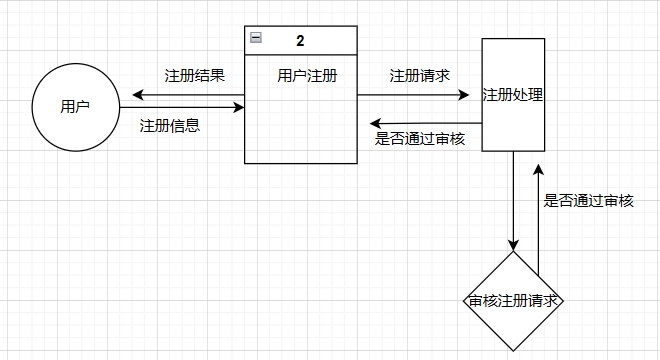


图2‑2 用户注册数据流图

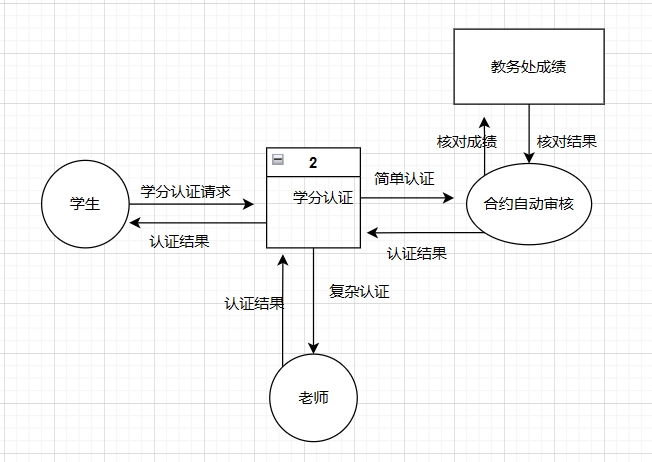


图2‑3 学分认证数据流图

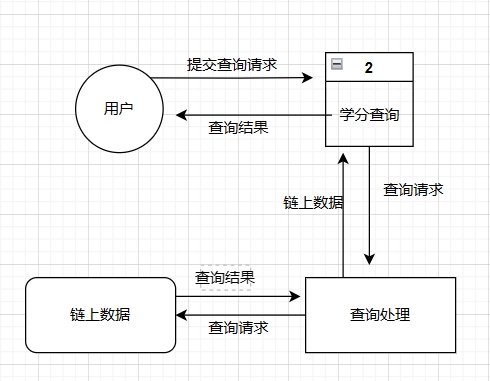


图2‑4 学分查询数据流图

## 数据字典

表2‑1 数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 数据项名 | 含义说明 | 组成 |
| 1 | 用户信息（User） | 学分认证请求中的用户信息 | 用户ID、姓名、学号、院系等 |
| 2 | 学分信息（Credit） | 学分认证请求中的学分信息 | 课程名称、学分数、成绩、学分证明材料等 |
| 3 | 学分认证结果 | 学分认证请求的结果 | 认证状态（通过/不通过）、认证时间、认证机构等 |
| 4 | 区块链数据块 | 区块链上存储学分认证相关数据的数据块 | 区块索引、时间戳、数据哈希值、前一区块哈希值、交易记录等 |
| 5 | 用户认证信息 | 用户认证子系统存储的用户认证相关信息 | 用户ID、用户名、密码哈希值、访问权限等 |

表2‑2 数据项

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 数据项名 | 数据描述 | 数据类型(长度) | 取值范围 | 与其他数据项的逻辑关系 |
| 1 | 用户ID | 用户的唯一标识符 | 字符串（20） |  |  |
| 2 | 姓名 | 用户的姓名 | 字符串（50） |  |  |
| 3 | 学号 | 用户的学号 | 字符串（20） |  |  |
| 4 | 院系 | 用户所在的院系 | 字符串（100） |  |  |
| 5 | 课程名称 | 学分认证请求中的课程名称 | 字符串（100） |  |  |
| 6 | 学分数 | 学分认证请求中的课程学分数 | 浮点数 |  |  |
| 7 | 成绩 | 学分认证请求中的课程成绩 | 字符串（10） |  |  |
| 8 | 学分证明材料 | 学分认证请求中的学分证明材料 | 二进制数据 |  |  |
| 9 | 认证状态 | 学分认证结果的认证状态 | 字符串（20） | 通过/不通过 |  |
| 10 | 认证时间 | 学分认证结果的认证时间 | 日期时间 |  |  |
| 11 | 认证机构 | 学分认证结果的认证机构 | 字符串（100） |  |  |
| 12 | 区块索引 | 区块链数据块的索引号 | 整数 |  |  |
| 13 | 时间戳 | 区块链数据块的时间戳 | 日期时间 |  |  |
| 14 | 数据哈希值 | 区块链数据块中数据的哈希值 | 字符串（64） |  |  |
| 15 | 前一区块哈希值 | 区块链数据块中前一区块的哈希值 | 字符串（64） |  |  |

# E-R模型设计

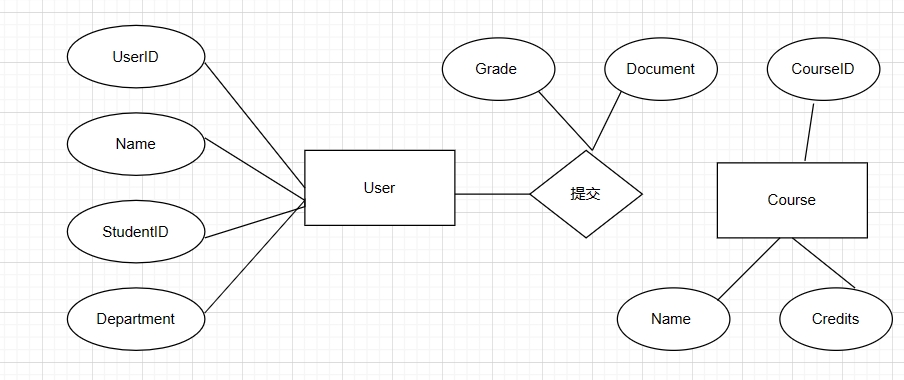


图 3 E-R模型图

# 数据库实现

## 数据库命名约定和环境

### 命名约定

当将数据存储在链上时：

1. 区块链表命名规则：

- 使用小写字母和下划线（\_）。

- 表达数据的实体或概念，可以使用清晰而具有描述性的表名。

- 避免使用过长或复杂的表名，以保持可读性。

2. 交易命名规则：

- 使用小写字母和下划线（\_）。

- 表达交易的目的或操作，可以使用清晰而具有描述性的命名。

- 避免使用过长或复杂的命名，以保持可读性。

3. 智能合约命名规则：

- 使用驼峰命名法，首字母小写。

- 表达智能合约的功能或目的，可以使用清晰而具有描述性的命名。

- 避免使用过长或复杂的命名，以保持可读性。

4. 数据字段命名规则：

- 使用小写字母和下划线（\_）。

- 表达字段的含义或属性，可以使用清晰而具有描述性的命名。

- 避免使用过长或复杂的命名，以保持可读性。

5. 事件命名规则：

- 使用小写字母和下划线（\_）。

- 表达事件的类型或目的，可以使用清晰而具有描述性的命名。

- 避免使用过长或复杂的命名，以保持可读性。

### 数据库环境

1. 数据库软件的名称和版本：由于区块链系统将数据存储在链上，传统的关系型数据库管理系统（DBMS）可能不再使用。而是使用针对区块链的特定软件或协议，如Ethereum、Hyperledger Fabric、Corda等。对于每个软件，选择合适的版本以满足项目需求。

2. 数据库设计工具：传统的数据库设计工具如Navicat可能不再适用于区块链系统，因为数据存储和访问方式与关系型数据库不同。在区块链系统设计中，更重要的是合理设计智能合约和数据结构，以满足学分认定的需求。因此，需要专注于智能合约的编写和区块链网络的部署。

3. 数据库的名称：由于区块链系统使用链式数据结构，数据存储在区块链网络中的多个节点上，而不是传统数据库中的单个数据库实例。因此，没有一个单一的数据库名称，而是使用区块链网络的名称来标识。

4. 数据文件：在区块链系统中，数据以区块的形式存储在链上，每个区块包含一定数量的交易记录。这些数据文件是分布式存储在区块链网络的各个节点上，每个节点都保存了完整的区块链数据副本。

5. 日志文件：区块链系统中的日志信息通常会被记录在区块链上的交易中，以确保数据的不可篡改性和透明性。每个交易都包含必要的日志信息，可以通过区块链浏览器或相应的API进行查询和分析。

6. 数据库其他类型的文件：在区块链系统中，并不需要像传统数据库中的控制文件或其他文件类型。区块链系统的数据存储和管理是通过共识算法和智能合约来实现的，而不是依赖于单一的数据库管理软件。

## 数据库关系图

在链上存储数据的区块链系统中，传统的数据库关系图的概念和工具可能不再适用。区块链的数据存储方式是基于分布式账本和链式数据结构，而不是传统的表格和主外键关系。

在区块链系统中，数据以区块的形式链接在一起，每个区块都包含一系列的交易记录。每个交易记录可以包含特定的数据字段，这些字段根据具体的智能合约和业务逻辑而定。

区块链系统的数据结构是链式的，每个区块都包含了前一个区块的哈希值，从而形成了一个不可篡改的数据链。数据之间的关系是通过交易记录和智能合约的逻辑来定义和维护。

由于区块链系统没有传统数据库中的表格和主外键关系，因此传统的数据库关系图工具（如SQL Server的数据库关系图工具或Navicat的ER图表功能）无法直接显示区块链系统的数据结构和关系。

相反，了解区块链系统的数据结构和智能合约之间的关系需要详细了解系统的设计和实现。可以通过查看智能合约代码、了解交易数据结构以及分析区块链浏览器中的数据来理解数据之间的关系。

## 数据表信息

### 表列表

表4‑1 表清单

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **中文名称** | **物理表名** | **备注** |
| 1 | 用户表 | users | 存储用户相关信息，如姓名、学院、钱包地址等。 |
| 2 | 学分认定申请表 | credit\_applications | 存储学生提交的学分认定申请信息，包括学生ID、申请ID、申请时间、认定状态、学分、成绩、课程id等 |
| 3 | 学分认定记录表 | credit\_records | 存储学分认定的结果记录，包括学生ID、申请ID、课程ID、认定结果、认定时间、认证分数等 |

### 表1

表1作用简要说明。

说明：表一般以ID列作为主键，也可以使用存在一个实际含义上的业务复合主键，例如：月生产计划主表，可以设置ID为（逻辑）主键，也可以将表的“年、序号、计划类型” 设置为（业务）主键为。建议不要同时建立两种类型的主键，应尽量考虑选择其中一种。

表4‑2 用户表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **中文名称** | | 用户表 | | | | |
| **物理表名** | | users | | | | |
| **主键** | | 钱包地址 | | | | |
| **业务主键** | | 无 | | | | |
| **字段列表** | | | | | | |
| **序号** | **中文名称** | | **列名** | **数据类型** | **非空** | **外键** |
| 1 | 姓名 | | Name | String | 是 |  |
| 2 | 学院 | |  | String | 否 |  |
| 3 | 钱包地址 | | Address | String | 是 |  |

### 表2

表2作用简要说明。

表4‑3 角色表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **中文名称** | | 学分认定申请表 | | | | |
| **物理表名** | | credit\_applications | | | | |
| **主键** | | Address | | | | |
| **业务主键** | | 无 | | | | |
| **字段列表** | | | | | | |
| **序号** | **中文名称** | | **列名** | **数据类型** | **非空** | **外键表** |
| 1 | 学分 | | credit | uint | 是 |  |
| 2 | 成绩 | | scores | uint | 是 |  |
| 3 | 申请时间 | | Application\_date | uint | 是 |  |
| 4 | 认证状态 | | Certification\_status | bool | 是 |  |

### 表3

表3作用简要说明。

表4‑3 角色表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **中文名称** | | 学分认定记录表 | | | | |
| **物理表名** | | credit\_records | | | | |
| **主键** | | Address | | | | |
| **业务主键** | | 无 | | | | |
| **字段列表** | | | | | | |
| **序号** | **中文名称** | | **列名** | **数据类型** | **非空** | **外键表** |
| 1 | 学分 | | credit | uint | 是 |  |
| 2 | 成绩 | | scores | uint | 是 |  |
| 3 | 认证时间 | | Certification\_date | uint | 是 |  |
| 4 | 认证状态 | | Certification\_status | bool | 是 |  |

## 存储过程信息

存储过程/函数名称先给出物理名称，然后在括号中给出功能名称。功能名称就是按照存储过程/函数的功能给出的一个简短名称，能够基本说明存储过程/函数的功能。

功能说明对应概要设计说明书—功能模块设计说明—功能模块中的内容。

如果数据库中没有使用存储过程和函数，则文档中可以不包含本部分内容。

表4‑4 存储过程清单

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **名称** | **参数** | **返回值** | **功能说明** |
| 1 | 用户注册 | 用户信息 | Bool | 用户注册功能，数据上链 |
| 2 | 学分申请请求 | 学分申请信息 | Bool | 学分申请，数据上链 |
| 3 | 学分申请认证 | 审核员信息/无 | Bool | 认证请求是否通过，数据上链 |

# 数据库安全设计

数据库安全设计是确保数据库系统中的数据安全性的重要方面。在数据存储在链上的情况下，以下是一些常见的数据库安全设计措施：

1. 角色分配和权限划分：根据用户的职责和需要，将数据库用户分配到不同的角色，并为每个角色定义相应的权限。角色可以根据功能、数据等进行划分，例如管理员角色、普通用户角色等。权限可以包括读取、写入、更新、删除等操作权限，以及表、视图、存储过程等对象的访问权限。

2. 构建访问矩阵：访问矩阵是一种表示用户和数据库对象之间访问权限的矩阵结构。通过构建访问矩阵，可以清晰地了解每个用户对每个数据库对象的访问权限，从而进行权限管理和控制。

3. 加密数据：对敏感数据进行加密，以保护数据在存储和传输过程中的安全性。在链上存储的情况下，可以使用区块链的加密功能来确保数据的机密性和完整性。

4. 审计和日志记录：启用数据库的审计功能，记录用户的操作和访问行为。审计日志可用于监控和追踪数据库的访问情况，以及检测潜在的安全问题和威胁。日志记录还可以作为法律和合规要求的依据。

5. 强密码策略：要求用户使用强密码，并定期更换密码。强密码策略应包括密码长度要求、复杂性要求（包括大小写字母、数字和特殊字符的组合）、密码失效期限等。

6. 定期备份和恢复：定期备份链上存储的数据库，并测试恢复过程的有效性。备份数据应存储在安全的位置，以防止数据丢失或损坏。

7. 网络安全措施：实施网络安全措施，如防火墙、入侵检测系统和加密通信等，以保护数据库系统免受网络攻击和未经授权的访问。

8. 定期安全审计：定期进行安全审计，评估数据库系统的安全性，并制定相应的改进计划。安全审计可以包括漏洞扫描、渗透测试和安全风险评估等活动。