系统分析与设计

**科技专家资源共享平台**

**14-05-数据设计**

|  |  |  |
| --- | --- | --- |
| **项目组成员信息** | | |
| **小组名称** | **那么队名就只有一个了！** | |
| **小组联系人** | **李光昊** | |
| **学号** | **姓名** | **本文档中主要承担的工作内容** |
| 16211058 | 李光昊 | 文档整合排版 |
| 16211059 | 孟庆博 | 结构化数据库设计——专家信息 |
| 16211061 | 闫晨雨 | 非结构化数据库设计 |
| 16211068 | 曾骏鹏 | 非结构化数据库设计 |
| 16211067 | 陈喆 | 结构化数据库设计——消息 |
| 16211065 | 乔禹 | 非结构化数据库设计 |
| 16211057 | 邹健伦 | 非结构化数据库设计 |
| 16211060 | 张子旺 | 结构化数据库设计——科技成果 |
| 16211072 | 刘铭辉 | 结构化数据库设计——科技成果 |

2019-04

版本变更历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 提交日期 | 主要编制人 | 审核人 | 版本说明 |
| 1.0 | 2019.4.18 | 李光昊 | 李光昊 | 文档整合排版 |

目 录

[引言 1](#_Toc6512937)

[一、架构 1](#_Toc6512938)

[二、数据库设计 1](#_Toc6512939)

[2.1 结构化数据库(mysql)设计 1](#_Toc6512940)

[2.1.1 账户表 1](#_Toc6512941)

[2.1.2 专家表 2](#_Toc6512942)

[2.1.3 消息表 2](#_Toc6512943)

[2.1.4 科技成果表 4](#_Toc6512944)

[2.2 非结构化数据库(MongoDB)设计 6](#_Toc6512945)

[2.2.1 用户的充值记录 6](#_Toc6512946)

[2.2.2 用户的购买记录 6](#_Toc6512947)

[2.2.3 用户登录记录 7](#_Toc6512948)

[2.2.4 专家被访问 7](#_Toc6512949)

[2.2.5 科技成果被浏览 8](#_Toc6512950)

[2.2.6 论文-专家关系 8](#_Toc6512951)

[2.2.7 专利-专家关系 8](#_Toc6512952)

[2.2.8 项目-专家关系 9](#_Toc6512953)

[2.2.9 专家-科技成果关系 9](#_Toc6512954)

[2.3 图数据库 10](#_Toc6512955)

[2.3.1 属性设计 10](#_Toc6512956)

[2.3.2 对关系的设计 11](#_Toc6512957)

[总结 11](#_Toc6512958)

引言

数据设计可分为数据架构、数据库架构和数据库设计，具体涉及数据库技术的使用和管理结构、数据库管理系统(DBMS)的选择与使用、确定存储的持久性数据、设计合理的数据库结构、定义数据库访问的策略等。

一、架构

数据架构和数据库架构的内容在之前的架构设计文档中的部署策略中有所提及。

数据层分为两种数据库，分别是存放论文，专利，专家等基本内容的MySQL和存放日志文件及实体关系（如论文和专家的对应关系）的MongoDB（NoSQL）。科技资源实体存放于文件系统中。

此外后端层还部署有Redis缓存，用来存储热点搜索数据等高频访问的数据，能减少查询数据库耗时，增强服务器性能。

对于缓存这里（两次缓存都是这样），用户如果是查询就不用多说，如果是写入的话，会在数据库写入完毕后立即反向更新各缓存中相关的数据。这听起来可能消耗很大，不过实际上写入操作实际上很少：发送消息时可能涉及到写入，不过相关人员很少，几乎不会涉及到缓存更新；管理员审核通过时可能会修改，不过这修改量很小，而且可以在用户访问量较小的时候进行写入操作。存储论文、专利文章，我们会使用专门的文件服务器来存储，并通过自己的web服务器请求文献数据。

对于数据库的高并发量请求问题，我们会通过消息队列的方式暂存数据库请求。缓存数据库服务器会搭载程序，多进程并发处理消息队列中请求。并且在该服务器上会有缓存数据库，来减少对最终数据库的查询。实际上，我们是采用两级缓存，两级服务器的方式来达到最终数据库的。

之后我们采用数据库集群，并采用合适的管理软件统一管理，来解决存储量庞大的问题。

数据库访问优先从缓存数据库获取数据。缓存数据库服务器的程序如果在redis中没有找到想要的数据，便会从最终的集群数据库中加载数据。集群数据库使用mysql cluster统一管理多个搭载mysql的数据库服务器。

二、数据库设计

2.1 结构化数据库(mysql)设计

2.1.1 账户表

账户表存储所有用户的账户信息，包括普通用户、专家用户及管理员。

表2-1 account

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键 | user\_id | 唯一标志用户的id | Int | 10 | 否 | 唯一 |  |  |
|  | type | 用户类型，1为普通用户，2为专家用户，3为管理员 | Int | 1 | 否 |  | 1 |  |
|  | user\_name | 用户名 | Varchar | 小于22 | 否 | 唯一 |  |  |
|  | password | 密码 | Varchar | 小于22大于6 | 否 |  |  |  |
|  | real\_name | 真实姓名 | Varchar | 小于22 | 是 |  |  |  |
|  | tel | 电话 | Varchar | 小于22 | 是 |  |  |  |
|  | email | 邮箱 | Varchar | 小于45 | 是 |  |  |  |
|  | basic\_info | 其他基本信息 | Varchar | 小于255 | 是 |  |  |  |
|  | money | 用户积分 | Double | 大于等于0 | 否 |  | 0 |  |

2.1.2 专家表

索引为主键。

表2-2 expert

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键 | Expert\_id | 唯一标识专家 | Int | 10 | 否 | 唯一 |  |  |
|  | Name | 专家的姓名 | Varchar | 小于255 | 否 |  |  |  |
|  | Institute | 专家的所属机构 | Varchar | 小于255 | 是 |  |  |  |
|  | Position | 专家的职位 | Varchar | 小于255 | 是 |  |  |  |
|  | Direction | 专家的研究方向 | Varchar | 小于255 | 是 |  |  |  |
|  | Introduction | 专家简介 | Varchar | 小于1000 | 是 |  |  |  |
|  | Contact | 联系方式 | Varchar | 小于255 | 是 |  |  |  |

2.1.3 消息表

Message表记录消息的基本信息，Message\_txt表记录消息内容，T\_message表记录系统信息。

表2-3 Message

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键值 | Index | 消息的主键，记录唯一一条消息 | Long |  | 否 | 唯一 |  |  |
| 外键 | Send\_id | 发送者id | Int |  | 否 |  |  |  |
| 外键 | Receive\_id | 接受者id | Int |  | 否 |  |  |  |
| 外键 | Text\_id | Message\_txt的id | Long |  | 否 | 唯一 |  |  |
|  | Date | 发送时间 | Date |  | 否 |  |  |  |
|  | State | 查看状态比如说（1：用户能查看，0：用户不能查看） | Int |  | 是 |  | 1 |  |
|  | Type | 消息类型 | Varchar | 255 | 否 |  |  |  |

表2-4 Message\_txt

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键 | Id |  | Long |  | 否 | 唯一 |  |  |
|  | Content | 信息内容 | Varchar | 4096 | 是 |  |  |  |
|  | Date | 发送时间 | Date |  | 否 |  |  |  |

表2-5 T\_message

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键值 | Index | 消息的主键，记录唯一一条消息 | Long |  | 否 | 唯一 |  |  |
| 外键 | Receive\_id | 接受者id | Int |  | 否 |  |  |  |
| 外键 | Request\_id | 成果请求id | Int |  | 否 |  |  |  |
|  | content | 消息内容 | Varchar | 4096 | 是 |  |  |  |
|  | Request\_date | 请求时的时间 | Date |  | 否 |  |  |  |
|  | Date | 发送时间 | Date |  | 否 |  |  |  |
|  | State | 查看状态比如说（1：用户能查看，0：用户不能查看） | Int |  | 是 |  | 1 |  |
|  | Type | 消息类型 | Varchar | 255 | 否 |  | default |  |

2.1.4 科技成果表

科技成果分为论文、专利和科研项目。

论文、专利和科研项目的索引均为主键索引，会默认添加。（其他的标题、摘要等索引在非关系数据库中）

表2-6 essay

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键 | Paper\_id | 唯一标识论文 | Int | 10 | 否 | 唯一 |  |  |
|  | Paper\_name | 论文题目 | Varchar | 小于255 | 否 |  |  |  |
|  | Author\_id | 第一作者id | Int | 10 | 否 |  |  |  |
|  | Author\_name | 第一作者姓名 | Varchar | 小于255 | 否 |  |  |  |
|  | Research\_areas | 研究领域 | Varchar | 小于255 | 是 |  |  |  |
|  | Introduction | 论文简介 | Varchar | 小于1000 | 是 |  |  |  |
|  | Institute | 所属机构 | Varchar | 小于255 | 是 |  |  |  |
|  | Openness | 公开性 | Int | 4 | 否 |  |  |  |
|  | Public\_content | 公开内容 | Varchar | 小于1000 | 是 |  |  |  |
|  | Published\_time | 发表时间 | Date | 3 | 是 | 不超过系统当前日期 |  |  |
|  | Times\_cited | 被引数 | Int |  | 是 | 非负整数 | 0 |  |
|  | Price | 价格 | Int |  | 是 | 非负整数 | 0 |  |
|  | Source | 来源 | Varchar | 小于255 | 是 |  |  |  |
|  | Type | 类型 | Varchar | 8 | 否 |  |  |  |
|  | Download\_link | 下载链接 | Varchar | 小于255 | 是 |  |  |  |
|  | Download\_times | 下载数 | Int |  | 是 | 非负整数 | 0 |  |
|  | Summary | 摘要 | Varchar | 小于1000 | 是 |  |  |  |
|  | Document\_id | 论文文档id | Varchar | 小于255 | 是 |  |  | 文件系统的索引 |

表2-7 Patent

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键 | Patent\_id | 专利唯一标识 | Int | 10 | 否 | 唯一 |  |  |
|  | Patent\_name | 专利题目 | Varchar | 小于255 | 否 |  |  |  |
|  | Author\_id | 第一作者id | Int | 10 | 否 |  |  |  |
|  | Author\_name | 第一作者姓名 | Varchar | 小于255 | 否 |  |  |  |
|  | Research\_areas | 研究领域 | Varchar | 小于255 | 是 |  |  |  |
|  | Introduction | 论文简介 | Varchar | 小于1000 | 是 |  |  |  |
|  | Institute | 所属机构 | Varchar | 小于255 | 是 |  |  |  |
|  | Openness | 公开性 | Int | 4 | 否 |  |  |  |
|  | Public\_content | 公开内容 | Varchar | 小于1000 | 是 |  |  |  |
|  | Published\_time | 发表时间 | Date | 3 | 是 | 不超过系统当前日期 |  |  |
|  | Owner\_id | 所有者id | Int | 10 | 否 |  |  |  |
|  | Owner\_name | 所有者姓名 | Varchar | 小于255 | 否 |  |  |  |
|  | Price | 价格 | Int | 4 | 是 | 非负整数 | 0 |  |
|  | Type | 类型 | Varchar | 8 | 否 |  |  |  |
|  | Download\_link | 下载链接 | Varchar | 小于255 | 是 |  |  |  |
|  | Download\_times | 下载数 | Int | 8 | 是 | 非负整数 | 0 |  |
|  | Document\_id | 专利文档id | Varchar | 小于255 | 是 |  |  | 文件系统的索引 |

表2-8 project

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主键外键 | 字段名 | 字段描述 | 数据类型 | 长度 | 是否可空 | 约束 | 缺省值 | 备注 |
| 主键 | Project\_id | 唯一标识项目 | Int | 4 | 否 | 唯一 |  |  |
|  | Name | 项目的名称 | Varchar | 小于255 | 否 |  |  |  |
|  | Institute | 项目的所属机构 | Varchar | 小于255 | 是 |  |  |  |
|  | Direction | 项目的研究方向 | Varchar | 小于255 | 是 |  |  |  |
|  | Introduction | 项目简介 | Varchar | 小于1000 | 是 |  |  |  |
|  | Experts | 负责专家id | Int | 10 | 是 |  |  |  |

2.2 非结构化数据库(MongoDB)设计

项目中的一些数据用传统的关系型数据库存储并不方便，比如存在复杂的嵌套结构，存在一对多使用数组来存储的情况，这时候使用传统的关系型数据库十分掣肘，而且在查询时效率低下，我们打算使用mongodb数据库来存储这些数据。其中包括：用户的充值记录，用户购买的科技成果（含有一些简要信息），登录记录和访问记录的日志。下面详细说明一下mongodb数据库的设计。

2.2.1 用户的充值记录

为每一个用户生成一个文档，用于存储其充值记录，文档结构如下：

用户文档

{

"\_id": ObjectId(),

"用户id": 32位整数

"充值记录":[

{充值记录1}

{充值记录2}

......

]

}

嵌套在用户文档内部的充值记录文档结构：

{

"充值时间": date

"充值金额": 64位浮点数

}

所有文档的基本结构一致，不存在同一集合或数组内文档结构不一致的现象。用户id就是存储在关系数据库中用于唯一标识用户的用户id。充值记录字段为数组类型，数组中的元素是文档类型，名为充值记录文档，其结构如上。本集合会在用户文档的用户id字段建立索引，增加查找到某个用户的速度。

2.2.2 用户的购买记录

用户购买记录文档，结构：

{

"\_id": ObjectId(),

"用户id": 32位整数

"购买记录":[

{购买记录1}

{购买记录2}

......

]

}

购买记录，文档结构：

{

"购买时间": date

"科技成果id": 32位整数

"花费金额": 64位浮点数

"科技成果简介": 字符串类型

}

同样会按用户id建立索引。其中同一数组内的不同文档科技成果id不会重复（没有则添加）。在一个用户进行专家认证后会自动将自己的科技成果添加到自己账户的购买记录中，花费金额为0。可能会觉得存储简介有些冗余，但其实这是为了实现特定功能提高效率而进行的存储。

2.2.3 用户登录记录

{

"\_id": ObjectId(),

"用户id": 32位整数

"登录记录":[

{登录记录1}

{登录记录2}

......

]

}

登录记录

{

"登录时间": date

"登录ip": 字符串

"登录设备/mac地址": 字符串

"登录地点": 字符串

}

2.2.4 专家被访问

{

"\_id": ObjectId(),

"专家信息页id": 32位整数

"被访问次数": 64位整数

"访问记录":[

{访问记录1}

{访问记录2}

......

]

}

访问记录

{

"访问时间": date

"访问者id": 32位整数

}

2.2.5 科技成果被浏览

{

"\_id": ObjectId(),

"科技成果id": 32位整数

"被访问次数": 64位整数

"访问记录":[

{访问记录1}

{访问记录2}

......

]

}

访问记录

{

"访问时间": date

"访问者id": 32位整数

}

后几个集合同样以xxxid字段建立索引。

2.2.6 论文-专家关系

{

"\_id": ObjectId(),

"论文id": 32位整数,

"标题": 字符串,

"摘要": 字符串,

"专家id": [

{"专家id1": 32位整数},

{"专家id2": 32位整数}

……

]

}

为标题和摘要设置全文索引

2.2.7 专利-专家关系

{

"\_id": ObjectId(),

"专利id": 32位整数,

"标题": 字符串,

"专家id": [

{"专家id1": 32位整数},

{"专家id2": 32位整数}

……

]

}

为标题设置全文索引

2.2.8 项目-专家关系

{

"\_id": ObjectId(),

"项目id": 32位整数,

"项目名": 字符串,

"专家id": [

{"专家id1": 32位整数},

{"专家id2": 32位整数}

……

]

}

为项目名设置全文索引

2.2.9 专家-科技成果关系

{

"\_id": ObjectId(),

"专家id": 32位整数,

"专家姓名": 字符串,

"论文id": [

{"论文id1": 32位整数},

{"论文id2": 32位整数}

……

],

"专利id": [

{"专利id1": 32位整数},

{"专利id2": 32位整数}

……

],

"项目id": [

{"项目id1": 32位整数},

{"项目id2": 32位整数}

……

]

}

为专家姓名建立全文索引

上面的2.2.6~2.2.9不仅代表了专家和科技成果间的关系，也是之后项目的搜索功能实现的基础。MongoDB对集合字段的修改比较方便，所以这里只列出了可以满足基本搜索功能的字段，之后可根据检索条件添加其他字段并设其为全文索引(通过设置全文索引可以实现对该字段的全文查询)。

再次强调，这些设计是为了具有更高要求的查询速度而进行适当的冗余设计。

以上便是mongodb数据库的存储集合划分和设计。

2.3 图数据库

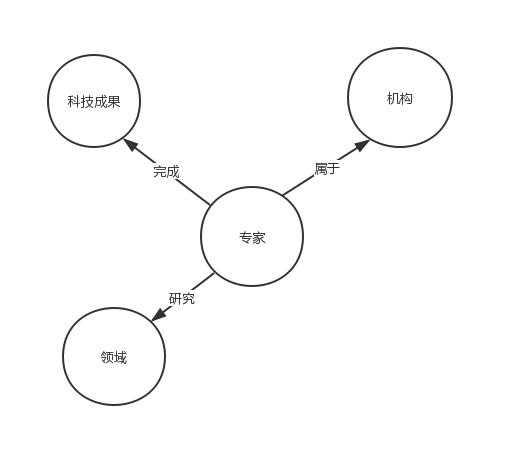


图2-1 专家关系图

图数据库我们打算使用Neo4j，主要就是用来展示专家关系图谱功能。上图节点类的组织关系。下面首先是对不同种类节点的属性设计。

2.3.1 节点设计

2.3.1.1 专家

专家id：int

专家姓名：string

联系方式：string

简介： string

2.3.1.2 科技成果

成果id：int

成果名称：string

成果简介：string

2.3.1.3 机构

机构id：int

机构名称：string

机构简介：string

2.3.1.4 领域

领域名称：string

2.3.2 对关系的设计

2.3.2.1 完成

发表时间：long

2.3.2.2 属于

加入时间：long

离开时间：long

2.3.2.3 研究

（无属性）

通过图数据库的存储我们将实现目标功能，使用户能够以某一专家为中心，通过领域关联、合作关联、同一机构关联的方式查看专家的关系图谱。

总结

数据设计部分在架构设计的基础上对科技资源共享平台中重要的数据库部分进行了设计，确定了数据库技术选择，完成了数据库逻辑结构及物理结构的设计。