数据库设计文档

|  |  |
| --- | --- |
| 项目名称： | 多平台云同步记事应用 |
| 队长姓名： | 庄耿行13354491 |
| 队长电话： | 13750065626 |
| 队员： | 13354136柯晓鸿 13354364肖福刚 |

## 数据库应用

多平台同步的记事应用，将数据库建于云端，方便用户和团体及时获取最新提醒。

## 数据库设计：

### 1.0汇总表

|  |  |
| --- | --- |
| **表名** | **功能说明** |
| user | 存放用户账户信息 |
| user\_info | 存放用户个人信息 |
| team | 存放团队信息 |
| membership | 个人与团队的归属关系 |
| matter | 存放事项的具体信息 |
| tag | 存放标签信息 |
| matter\_tag | 存放事项和标签的对应关系 |

### 2.0表属性

#### 2.1 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | user | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| user\_id | int | 非空 | 用户id，主键，自增 |
| account | varchar(16) | 非空 | 用户名 |
| password | varchar(128) | 非空 | 加密后的密码 |

#### 2.2个人信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | user\_info | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| user\_id | int | 非空 | 用户id，外键，主键 |
| name | varchar(20) | 非空 | 昵称 |
| sex | Varchar(6) | 非空 | 性别 |

个人信息表用于存储用户的个人信息，可能会在后期动态地添加字段，为了避免字段混乱，和方便以后用户登录信息的迁移，所以单独划分出一张表用于记录用户个人信息。

#### 2.3团队信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | team | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| team\_id | int | 非空 | 团队id，主键, 自增 |
| name | varchar(20) | 非空 | 团队昵称 |
| team\_info | varchar(100) | 可空 | 团队介绍 |

#### 2.4个人与团队的归属关系

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | membership | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| team\_id | int | 非空 | 团队id，外键 |
| user\_id | int | 非空 | 个人id，外键，与team\_id形成联合主键 |
| relation | varchar(20) | 非空 | 用户与团队的关系，如管理员，一般用户等。 |

一个用户可能输入多个团队，一个团队可能包含多个成员，所以要单独用一个表来记录归属关系。

#### 2.5事项表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | matter | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| matter\_id | int | 非空 | 事项id，主键，自增 |
| title | varchar(100) | 非空 | 事项标题 |
| content | varchar(200) | 非空 | 事项内容 |
| time | varchar(50) | 非空 | 到期时间 |
| priority | int | 非空 | 紧急程度 |
| type | int | 非空 | 事项类型，个人事项或团队事项 |
| belong | int | 非空 | 所属的个人或团队id |

#### 2.6 分类标签表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | tag | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| tag\_id | int | 非空 | 标签id，主键, 自增 |
| title | varchar(20) | 非空 | 标签名 |
| type | int | 非空 | 标签类型，个人标签或团队标签 |
| belong | int | 可空 | 所属的个人或团队id |

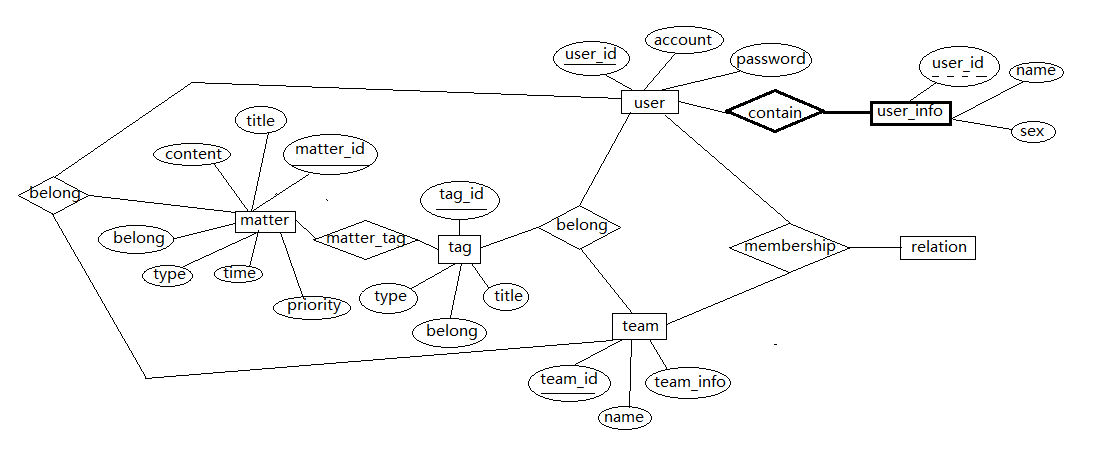
用户和团队可以创建多个标签用于事项的分类。

#### 2.7事项和标签的归属关系

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | matter\_tag | | |
| 列名 | 数据类型(精度) | 空（非空） | 含义 |
| matter\_id | int | 非空 | 事项id，外键 |
| tag\_id | int | 非空 | 标签id，外键，与matter\_id形成联合主键 |

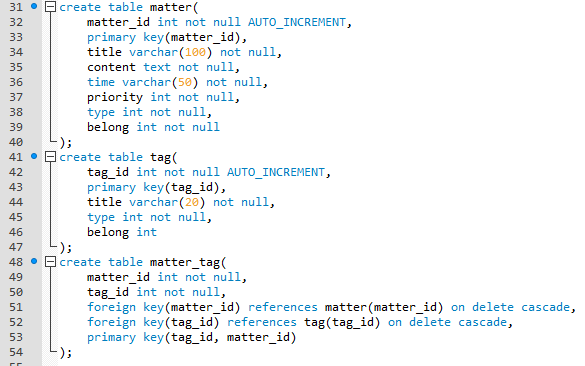
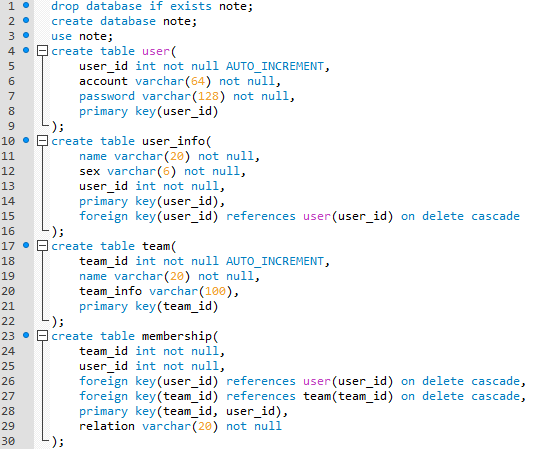
一个标签可以对应多个事项，一个事项可以属于多个标签。需要两个id共同确定从属关系。

### 3.0 ER图



数据库实现：

代码实现：



如何实现的、为什么选MYSQL

我们需要一个MySQL数据库系统，用于存储用户，团队等配置信息和记事条目等应用数据，在客户端请求数据时根据提供的条件查询数据库中的条目返回结果，以及在后台实时更新查询需要进行提醒的用户和对应记事。

MySQL开源免费，用户广泛，开源社区分享互助氛围浓厚，遇到问题时易于解决。此外MySQL数据库使用标准SQL语句，易于学习和维护，即使后期需求发生变化，更换数据库后，程序与数据库交互的逻辑也无需做太大改变。

除此之外，MySQL的表查询效率非常高，且支持内存级缓存，足以应对我们中小型应用的需要。MySQL支持多种表类型，除了常见的几种文件表，也支持如内存表等表类型，可以根据我们应用的功能需要，搭配以不同类型的表，可以减少硬盘IO，大大提升数据库的性能。

实际操作起来/可行性/实用性/复杂度

在做应用的过程中，基本对应我们的实验需求，但是也有些许地方与我们的应用有所差别，主要是在应用实现阶段发现了部分表的更好的设计，但由于之前的表已经定型，所以我们仍然套用了我们原来的设计与实现，但也增加了我们不少的难度，主要在于一些代码要有所侧重地“迎合”数据库的设计。总的来说，可行性和实用性还是不错的，复杂度一般。

功能

基本的文字记录各类事件，如作业、备忘事宜等

可编辑和删除记录的事件等数据内容

备忘、提醒，程序会在事件截止当天发送推送提醒信息

可编辑事件的优先级，程序会根据优先级和截止时间对所有待办事宜排序

多平台操作，用户还可以在网页端进行编辑，一处编辑，多设备查阅

团队协作，多用户可同时操作