同济大学

TONGJI UNIVERSITY

数据库课程设计

同济大学数学建模协会

内部工作管理系统

姓　　名：周雨杨

学　　号：1953793

所在院系：计算机科学与技术

学科专业：电子与信息工程学院

指导教师：李文根老师

二〇二二年六月

摘要

本次课程设计中，我从数学建模协会的工作经验出发，针对人员管理、通知管理、文件管理、任务管理四大方面提取出一系列提高社团内部管理效率的需求，并将其规范化、抽取系统工作流程、列写详细用例说明。

然后，针对这些需求，我建立了概念模型，绘制了系统的 ER 图。在概念模型的基础上，我进一步研究系统内各个关系，得到了符合 3NF 范式的一组简单优雅的关系模式。最后，我仔细分析了各关系的各个属性，并选择了合适的索引。

系统实现为下学期内容。

关键词：数据库系统，需求分析，数据库设计，社团管理

目录

第 1 章　选题分析

本章主要阐述项目的背景信息。

1.1　业务背景

同济大学数学建模协会（后称协会）是团委名下的一个社团，主要负责帮助同济大学内的数学建模（后称数模）爱好者们进行数模竞赛的备赛、参赛、领奖。协会在数学学院（后称数院）的帮助下，为同学们提供信息交流的渠道，并开展一系列活动，为同学们提供丰富的数模竞赛资源。

然而，由于历史原因和一些不可抗力因素，协会目前的工作模式基于微信群聊，这会造成一系列不便：

1. **文件混乱，难以查找**。在 qq 群聊中，群文件能以目录的形式较好组织，同学们查找资料快，然而微信群聊中的文件没有层次结构，只能在历史记录中搜索文件名。
2. **通知和文件容易出现缺失**。微信出于节约存储空间的角度考虑，多端聊天记录的同步没有 qq 做得好，经常出现手机看到的通知和文件，电脑上没有的情况。
3. **通知栈过小，只能有一条群公告**。在比赛前后，协会往往会有多个任务并行，但是群公告只能保留一条，这造成了管理上的不便。
4. **任务管理主要靠人力**。在基于社交软件的管理模式下，工作的分配只能靠领导者完成，下达任务、跟进任务时只能通过微信消息通知。这样下来消息链延迟大，多人反而比单人低效。
5. **面向普通同学的通知较为低效**。竞赛报名和活动通知往往由老师、协会工作人员手动完成。在报名季，百人规模的通知交给同学手动完成可能效率不高。
6. **工作的重复性和模式化没有得到利用**。事实上，协会的工作内容和模式较为固定，但是由于管理主要依靠人力，不确定因素大，工作难以形成固定的模式。

基于上述问题，可以开发一个针对社团内部管理的系统，有助于自动化社团管理，提升社团工作效率。

1.2 适用对象

本系统谨针对本人在管理数学建模协会过程中遇到的一系列问题开发。不过这些问题是所有社团都可能遇到的共性问题，可能可以实现在其他社团的复用。

该系统主要面向社团内部管理，仅部分工作内容（如批量发送邮件）会涉及到社团外部成员，非社团成员没有登录系统的权限。系统用户可以分为四个等级：

1. **教师：**具有最高权限。包括：
   (a) 变更人员安排（包括等级和部门）
   (b) 发布、修改和撤销通知
   (c) 上传和删除文件
   (d) 自定义工作流程模板
   (e) 分配任务，建立、更改、删除任务和子任务的 deadline
   (f) 完成任务
2. **社长：**具有和教师同等的最高权限。包括：
   (a) 变更人员安排（包括等级和部门）
   (b) 发布、修改和撤销通知
   (c) 上传和删除文件
   (d) 自定义工作流程模板
   (e) 分配任务，建立、更改、删除任务和子任务的 deadline
   (f) 完成任务
3. **部长：**具有和社长类似的较高权限，但跨部门合作部分权限略低：
   (a) 变更部长及以下的人员安排（包括等级和部门）
   (b) 发布、修改和撤销部门通知
   (c) 上传和删除文件
   (d) 分配本部门任务，建立、更改、删除本部门任务和子任务的 deadline
   (e) 完成任务
4. **部员：**考虑到部员多为新生，误操作可能性较大，权限较低，仅包括：
   (a) 上传文件
   (b) 完成任务
5. **前成员：**这部分用户是已经从社团退休的同学，不受任何人的安排、调度，但是对整个系统只有读权限。保留这部分用户主要是出于联系方式查找的目的。

下面将教师、社长、部长统称为管理员，部员称为普通用户。

1.3　实现范围

本系统针对社团内部管理的复杂关系与逻辑，设计相应的数据库和 web 应用。推荐用户采用 PC 端登录的主要出于文本类文件的管理需求。考虑到近年来移端应用的普及，时间允许的情况下，可能会支持 app 和微信小程序的前端。

第 2 章　需求分析

本章详细分析系统的需求。

2.1　业务流程

下面介绍系统四大业务的主要流程。

2.1.1　人员管理

用户注册对应的是新成员加入，只有管理员有权限执行。初始化时账号密码和学号相同，新用户需要激活账号、变更密码后使用。

用户登录时，系统能够通过数据库查找到对应的等级和部门，以管理相应权限。

用户登出后返回登录界面。

社长有权限更改所有人（除教师）的等级、部门。部长则有权限更改部门内部所有人的等级、部门，需要注意，如果一个部员分到另一个部门后，只有另一个部长操作才能将这位同学调回。

部门内部，所有成员的联系方式（手机号、邮箱）共享。

2.1.2　通知管理

管理员可以发布通知、取消通知。通知在发布后会通过邮件发送给相关人员。在应用中也有页面可以查看所有通知。

2.1.3　文件管理

管理员可以上传和删除文件。文件系统中有两个根目录，official 和 internal。official 目录下存放面向社团外部成员的官方文件，这些文件可能是竞赛方的官方通知，也可能是社团发放的竞赛事项和活动通知，属于成品性质，修改较少；internal 目录下存放社团内部管理相关的文档，如活动总结、报销规则等资料，属于半成品性质，修改较多。

普通用户只有在 internal 目录下上传、修改文件的权限。只有管理员可以操作 official 目录。

后续可能会考虑将网站对所有人开放，公开 official 目录。

2.1.4　任务管理

任务管理基于工作流程模板实现。工作流程模板可以表现一场活动通常的安排流程，如讲座类活动，首先需要活动部联系讲师、确定场地、策划方案、采购物资；然后学术部的同学审核讲师内容；宣传部的同学组织活动推送。活动过程中还需要活动部的同学主持记录。结束后则是宣传部的同学发推送，活动部的同学整理报销材料。整个流程需要牵扯到多方人员，常常因为消息链延迟导致低效，事实上，活动高度重复，通过 web 应用自动化管理，面向模板快速工作，能够大大提升效率。

管理员可以发布任务，设置任务 deadline，把任务分配给部门、分配给具体的用户。在 deadline 前一天，系统会通过短信提醒社长、任务所属部门的部长和具体的负责人。

用户可以勾选任务已完成。任务完成可能有某些限定条件，如上传活动记录文件等。这些限制条件可以通过模板的修改来指定。

2.2　用例详述

见附件。

第 3 章　概念设计

根据上一章节中提出的需求，系统可以概念建模如下：

对于实体类模型，主要是以 Member 为核心，操作 Task, File, Notification 等一系列实体。而仔细分析可以发现，操作的类型基本相同，以 Create、Update 为主。

在二元关系方面，一个用户可以创建多条通知，但是一条通知只能被一个人创建，所以这里是多对一的关系；类似地，File 和 Task 的 Create 也都是多对一的关系。然而，一条通知可以被很多人修改，也可以 Mention 很多人，所以 Mention 和 Update 都是多对多的关系。

Task 是系统中比较复杂的一个数据结构，每一个 Task 由 TaskTemplate 特化而来，不同之处在于它有自己的 CreateLog，而每一个 Task 或者 TaskTemplate 包含多个 SubTask，为了节约空间和实现多任务交叉，采用类似指针的形式，通过 Task 定位第一个 SubTask，然后每个 Task 存储它的 PreTaskID 和 NextTaskID。Contains 关系也是一个多对多的关系，因为一个任务包含多个子任务，一个子任务有可能同时参与到多个任务中。

基于以上考虑，系统的 ER 图如下：

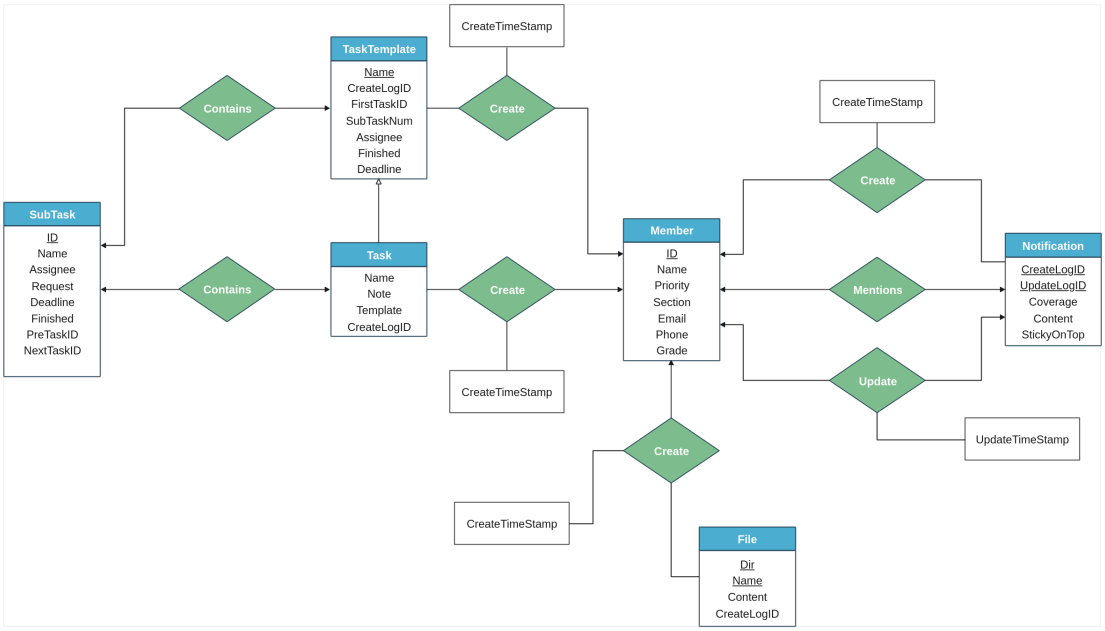

图 3.1　系统 ER 图

第 4 章　逻辑设计

4.1　函数依赖分析

根据上一章节中给出的概念模型，我们可以得到如下关系模式：

$Member(\underline{ID}, Name, Priority, Section, Email, Phone, Grade)$

$CreateLog(\underline{CreateLogID}, Creator, CreateTimestamp)$

$UpdateLog(\underline{UpdateLogID}, Updater, UpdateTimestamp)$

$Notification(\underline{CreateLogId}, \underline{UpdateLogID}, Coverage, Content, StickyOnTop)$

$Mention(\underline{Coverage}, MemberID)$

$File(\underline{Dir}, \underline{Name}, Content, CreateLogID)$

$TaskTemplate(\underline{Name}, CreateLogID, FirstTaskID, SubTaskNum, Assignee, Finished, Deadline)$

$Task(\underline{Name}, Note, Template, CreateLogID)$

$SubTask(\underline{ID}, Name, Assignee, Request, Finished, Deadline, Finished, PreTaskID, NextTaskID)$

系统中存在一系列的函数依赖：

Member 中, 通过一个人的 ID 可以确认其所有的其他信息：

$$ID \rightarrow Name, Priority, Section, Email, Phone, Grade$$

CreateLog 中，根据其 ID 可以确认其所有的其他信息：

$$CreateLogID \rightarrow Creator, CreateTimestamp$$

UpdateLog 中，根据其 ID 可以确认其所有的其他信息：

$$UpdateLogID \rightarrow Updater, UpdateTimestamp$$

Notification 中，根据其 CreateLogID 和 UpdateLogID 可以确认其所有的其他信息：

$$CreateLogID, UpdateLogID \rightarrow Coverage, Content, StickyOnTop$$

Mention 中，根据 Coverage 可以确认 MemberID：

$$Coverage \rightarrow MemberID$$

File 中，根据 Dir 和 Name 可以确定器所有的其他信息：

$$Dir, Name \rightarrow Content, CreateLogID$$

TaskTemplate 中，根据 Name 可以确定器所有的其他信息：

$$Name \rightarrow CreateLogID, FirstTaskID, SubTaskNum, Assignee, Finished, Deadline$$

Task 中，根据 Name 可以确定器所有的其他信息：

$$Name \rightarrow Note, Template, CreateLogID$$

SubTask 中，根据 ID 可以确定其所有的其他信息：

$$ID \rightarrow Name, Assignee, Request, Finished, Deadline, PreTaskID, NextTaskID$$

由于所有关系中，所有函数依赖的左半部分都是主码，右半部分都是所有的其他码，并且有且只有着一个约束，若按照 3NF 分解来分解，得到的结果完全不会变化，所以这样的关系模式满足 3NF。

4.2 数据字典

数据字典是每个数据元素的引用和说明，它是数据模型的详细定义和文档。在软件工程当中，需求工程师会在需求规格说明文档（PRD）中给出一个初步的数据字典，向软件工程师进一步说明数据流中的数据元。而软件工程师同样会在设计文档 (Design Doc) 中给出数据库系统更加详细的数据字典，供后续使用系统的人参考。

然而，本次课程设计较为特殊，需求和设计都由一人完成。由于这个系统并不复杂，数据的关系也没有进一步深挖的地方，故只在设计文档部分给出最终的关系模式，以避免冗余。

表 4.1 数据字典

表格	属性	类型	依赖关系
Member	ID	char(8)	primary key
	Name	varchar(15)	
	priority	int	
	Section	varchar(10)	
	Email	varchar(30)	
	Phone	char(12)	
	Grade	int	
CreateLog	CreateLogID	varchar(20)	primary key
	Creator	char(8)	foreign key(Member)

续下页

续表 4.1　数据字典

表格	属性	类型	依赖关系
	CreateTimeStamp	TimeStamp	
UpdateLog	UpdateLogID	varchar(20)	primary key
	Updater	char(8)	foreign key(Member)
	UpdateTimeStamp	TimeStamp	
Notification	CreateLogID	varchar(20)	primary key, foreign key(Member)
	UpdateLogID	varchar(20)	primary key, foreign key(Member)
	Coverage	varchar(10)	foreign key(Mention)
	StickyOnTop	boolean	
Mention	Coverage	varchar(10)	primary key
	MemberID	char(8)	foreign key(Member)
File	Dir	varchar(100)	primary key
	Name	varchar(30)	primary key
	Content	text	
	CreateLogID	varchar(20)	foreign key(CreateLog)
TaskTemplate	Name	varchar(20)	primary key
	CreateLogID	varchar(20)	foreign key(CreateLog)
	FirstTaskID	varchar(20)	foreign key(SubTask)
	SubTaskNum	int	
	Assignee	char(8)	foreign key(Member)
	Finished	boolean	
	Deadline	Date	
Task	Name	varchar(20)	primary key
	Note	varchar(100)	
	Template	varchar(20)	foreign key(TaskTemplate)
	CreateLogID	varchar(20)	foreign key(CreateLog)
SubTask	ID	varchar(20)	primary key
	Name	varchar(20)	
	Assignee	char(8)	foreign key(Member)
	Request	varchar(100)	
	Finished	boolean	
	Deadline	Date	
	PreTaskID	varchar(20)	foreign key(SubTask)
	NextTaskID	varchar(20)	foreign key(SubTask)

第 5 章　物理设计

基于合理的数据库设计，经过深思熟虑后为表建立索引，是获得高性能数据库系统的基础。而未经合理分析便添加索引，则会降低系统的总体性能。索引虽然说提高了数据的访问速度，但同时也增加了插入、更新和删除操作的处理时间。是否要为表增加索引、索引建立在那些字段上，是创建索引前必须要考虑的问题。解决此问题的一个比较好的方法，就是分析应用程序的业务处理、数据使用，为经常被用作查询条件、或者被要求排序的字段建立索引。基于优化器对 SQL 语句的优化处理，我们在创建索引时可以遵循下面的一般性原则：

1. 为经常出现在关键字 order by、group by、distinct 后面的字段，建立索引。在这些字段上建立索引，可以有效地避免排序操作。如果建立的是复合索引，索引的字段顺序要和这些关键字后面的字段顺序一致，否则索引不会被使用。
2. 在 union 等集合操作的结果集字段上，建立索引。其建立索引的目的同上。
3. 为经常用作查询选择的字段，建立索引。
4. 在经常用作表连接的属性上，建立索引。
5. 考虑使用索引覆盖。对数据很少被更新的表，如果用户经常只查询其中的几个字段，可以考虑在这几个字段上建立索引，从而将表的扫描改变为索引的扫描。

除了以上原则，在创建索引时，我们还应当注意以下的限制：

1. 限制表上的索引数目。对一个存在大量更新操作的表，所建索引的数目一般不要超过 3 个，最多不要超过 5 个。索引虽说提高了访问速度，但太多索引会影响数据的更新操作。
2. 不要在有大量相同取值的字段上，建立索引。在这样的字段（例如：性别）上建立索引，字段作为选择条件时将返回大量满足条件的记录，优化器不会使用该索引作为访问路径。
3. 避免在取值朝一个方向增长的字段（例如：日期类型的字段）上，建立索引；对复合索引，避免将这种类型的字段放置在最前面。由于字段的取值总是朝一个方向增长，新记录总是存放在索引的最后一个叶页中，从而不断地引起该叶页的访问竞争、新叶页的分配、中间分支页的拆分。此外，如果所建索引是聚集索引，表中数据按照索引的排列顺序存放，所有的插入操作都集中在最后一个数据页上进行，从而引起插入“热点”。
4. 对复合索引，按照字段在查询条件中出现的频度建立索引。在复合索引中，

记录首先按照第一个字段排序。对于在第一个字段上取值相同的记录，系统再按照第二个字段的取值排序，以此类推。因此只有复合索引的第一个字段出现在查询条件中，该索引才可能被使用。因此将应用频度高的字段，放置在复合索引的前面，会使系统最大可能地使用此索引，发挥索引的作用。

5. 删除不再使用，或者很少被使用的索引。

表中的数据被大量更新，或者数据的使用方式被改变后，原有的一些索引可能不再被需要。数据库管理员应当定期找出这些索引，将它们删除，从而减少索引对更新操作的影响。

基于以上考虑，最终为每个表格建立索引如下：

表 5.1　索引设计

表格	属性	类型	是否作为索引
Member	ID	char(8)	yes
	Name	varchar(15)	yes
	priority	int	yes
	Section	varchar(10)	no
	Email	varchar(30)	no
	Phone	char(12)	no
	Grade	int	no
CreateLog	CreateLogID	varchar(20)	yes
	Creator	char(8)	no
	CreateTimeStamp	TimeStamp	yes
UpdateLog	UpdateLogID	varchar(20)	yes
	Updater	char(8)	no
	UpdateTimeStamp	TimeStamp	yes
Notification	CreateLogID	varchar(20)	yes
	UpdateLogID	varchar(20)	yes
	Coverage	varchar(10)	no
	StickyOnTop	boolean	no
Mention	Coverage	varchar(10)	yes
	MemberID	char(8)	no
File	Dir	varchar(100)	yes
	Name	varchar(30)	yes
	Content	text	no
	CreateLogID	varchar(20)	yes
TaskTemplate	Name	varchar(20)	yes
	CreateLogID	varchar(20)	yes

续下页

续表 5.1　索引设计

表格	属性	类型	是否作为索引
	FirstTaskID	varchar(20)	no
	SubTaskNum	int	no
	Assignee	char(8)	yes
	Finished	boolean	no
	Deadline	Date	yes
Task	Name	varchar(20)	yes
	Note	varchar(100)	no
	Template	varchar(20)	yes
	CreateLogID	varchar(20)	yes
SubTask	ID	varchar(20)	yes
	Name	varchar(20)	yes
	Assignee	char(8)	yes
	Request	varchar(100)	no
	Finished	boolean	no
	Deadline	Date	yes
	PreTaskID	varchar(20)	no
	NextTaskID	varchar(20)	no

第 6 章　系统实现

（下学期）

附录 A　用例详述

表 A.1　新成员注册用例

用例编号	001
用例名称	新成员注册
参与者	管理员
用例说明	管理员将新成员的账号批量录入系统，给予其权限
前置条件	管理员登录
基本事件流	1. 管理员选择自动注册或手动注册。 1a. 若自动注册： 管理员上传指定格式的文本文件，包括所有新成员的必要信息。 1b. 若手动注册： 管理员输入新成员人数 系统给出相应行数的表单 管理员填写表单。 2. 管理员点击注册按钮。 3. 系统检查数据是否符合要求： 学号是否为空或不合法 学号是否已在数据库中存在 学号是否在同济大学数据库中存在 如姓名不为空，学号姓名的对应关系是否准确 4. 数据库系统插入新的数据，密码初始化为学号 5. 如新成员邮箱不为空，发送激活账号的通知。 6. 如新成员电话不为空，发送激活账号的短信。
异常事件流	1. 学号已存在或不合法，注册失败 2. 点击注册按钮时出现网络故障，注册失败 3. 点击注册按钮时出现数据库故障，注册失败
后置条件	1. 新成员的帐号保存到数据库信息表 2. 系统提示操作成功 3. 系统向新成员发送邮件和短信，提示账号激活通知。

表 A.2　用户登录用例

用例编号	002
用例名称	用户登录
参与者	全体用户
用例说明	用户登录系统
前置条件	无
基本事件流	1. 用户填写表单 2. 用户点击登录按钮 3. 系统检查学号和密码是否匹配

续下页

续表 A.2　用户登录用例

	4. 登录成功，跳转到主页面
异常事件流	1. 学号或密码未填写，登录失败 2. 学号和密码不匹配，登录失败 3. 点击登录按钮时出现网络故障，登录失败
后置条件	系统跳转到主页面

表 A.3　用户登出用例

用例编号	003
用例名称	用户登出
参与者	全体用户
用例说明	用户登出系统
前置条件	无
基本事件流	1. 用户点击登出按钮 2. 登出成功，跳转到登录页面
异常事件流	无
后置条件	系统跳转到登录页面

表 A.4　用户属性修改用例

用例编号	004
用例名称	用户属性修改
参与者	管理员
用例说明	管理员修改权限比自己低的用户的属性（部门、等级等）
前置条件	管理员登录，进入人事管理界面
基本事件流	1. 系统通过树形图展示全体成员姓名。 2. 管理员点击某成员。 3. 检查该成员权限是否低于管理员 3a. 若低于管理员： 系统通过可收缩卡片展示成员信息，相应属性可通过选择框编辑。 管理员点击更改按钮 系统修改该成员属性, 提示修改成功 返回人事管理界面并自动刷新 3b. 若高于管理员： 无事发生
异常事件流	点击修改按钮时网络故障，修改失败
后置条件	若产生修改，系统修改成员属性，提示修改成功，返回原界面并刷新

表 A.5　通知发布用例

用例编号	005
用例名称	通知发布
参与者	管理员

续下页

续表 A.5　通知发布用例

用例说明	管理员给自己管理的用户发送通知
前置条件	管理员登录，进入通知管理界面
基本事件流	1. 管理员通过选择框选择可选通知类型（面向部门、面向全体） 2. 管理员编辑通知内容，上传通知附件 3. 管理员选择是否置顶通知 4. 管理员点击发布按钮 5. 系统内插入新的通知条目 6. 系统给相关人员发送邮件通知 7. 提示用户通知发送成功，返回并刷新通知管理界面
异常事件流	点击发布按钮时网络故障，发布失败
后置条件	系统插入新的通知条目，提示发布成功，返回原界面并刷新

表 A.6　通知修改用例

用例编号	006
用例名称	通知修改
参与者	管理员
用例说明	管理员修改通知
前置条件	管理员登录，进入通知管理界面
基本事件流	1. 管理员点开某条通知 2a. 若管理员对这条通知有修改权限： 管理员可以重新编辑通知、上传文件 管理员选择是否置顶通知 管理员点击修改按钮 系统内修改相应通知条目 发送邮件通知相关人员通知已修改 提示管理员通知修改成功，返回并刷新通知管理界面 2b. 若管理员对这条通知无修改权限： 无事发生
异常事件流	点击修改按钮时网络故障，修改失败
后置条件	系统修改相应通知条目，提示修改成功，返回原界面并刷新

表 A.7　通知删除用例

用例编号	007
用例名称	通知删除
参与者	管理员
用例说明	管理员删除通知
前置条件	管理员登录，进入通知管理界面
基本事件流	1. 管理员点开某条通知 2a. 若管理员对这条通知有删除权限： 管理员点击删除按钮 系统内删除相应通知条目

续下页

续表 A.7　通知删除用例

	发送邮件通知相关人员通知已删除 提示管理员通知删除成功，返回并刷新通知管理界面 2b. 若管理员对这条通知无删除权限： 无事发生
异常事件流	点击删除按钮时网络故障，删除失败
后置条件	系统删除相应通知条目，提示删除成功，返回原界面并刷新

表 A.8　文件发布用例

用例编号	008
用例名称	文件发布
参与者	所有用户
用例说明	用户在系统中上传文件
前置条件	用户登录，进入文件管理界面
基本事件流	1. 用户导航到想要上传文件的目录 2. 用户上传文件与说明 3. 用户点击上传按钮 5. 系统内插入新的文件条目 7. 提示用户文件上传成功，返回并刷新文件管理界面
异常事件流	点击上传按钮时网络故障，shadchan 失败
后置条件	系统插入新的文件条目，提示上传成功，返回原界面并刷新

表 A.9　文件删除用例

用例编号	009
用例名称	文件删除
参与者	管理员
用例说明	管理员删除文件
前置条件	管理员登录，进入文件管理界面
基本事件流	1. 管理员选中某个文件 2. 管理员点击删除按钮 3. 系统删除指定文件条目 4. 提示管理员文件删除成功，返回并刷新文件管理界面
异常事件流	点击删除按钮时网络故障，删除失败
后置条件	系统删除相应文件条目，提示删除成功，返回原界面并刷新

表 A.10　自定义任务模板用例

用例编号	010
用例名称	自定义任务模板
参与者	管理员
用例说明	管理员根据实际情况建立新的任务模板
前置条件	管理员登录，进入任务管理界面

续下页

续表 A.10　自定义任务模板用例

基本事件流	1a. 管理员选择指定模板开始新建一个任务模板: 指定原来的任务模板，复制到当前的编辑页面 1b. 管理员选择从头开始新建一个任务模板 2. 在当前模板的基础上，管理员可以增加、修改、删除任何一个子任务 3. 对于每一个子任务，管理员可以设置默认值（如，默认分配给宣传部） 4. 对于每一个子任务，管理员可以设置完成任务的先决条件（如上传活动策划） 5. 管理员点击创建按钮，创建任务模板 6. 系统插入新的任务模板 7. 系统提示创建成功，返回并刷新任务模板界面
异常事件流	点击创建按钮时网络故障，创建失败
后置条件	系统插入相应任务模板条目，提示创建成功，返回原界面并刷新

表 A.11　创建任务用例

用例编号	011
用例名称	创建任务实例
参与者	管理员
用例说明	管理员选择任务模板建立任务实例
前置条件	管理员登录，进入任务管理界面
基本事件流	1. 管理员选择任务模板 2. 管理员对任务模板中的每一个子任务设置 ddl、负责部门、负责人 3. 管理员点击创建按钮，创建任务 4. 对于每一对任务，系统检查前一个任务的 ddl 是否至少比后一个任务早 1 天 5. 系统插入新的任务 6. 系统提示创建成功，返回并刷新任务管理页面
异常事件流	点击创建按钮时网络故障，创建失败
后置条件	系统插入相应任务条目，提示创建成功，返回原界面并刷新

表 A.12　任务修改用例

用例编号	012
用例名称	任务修改
参与者	管理员
用例说明	管理员修改任务
前置条件	管理员登录，进入任务管理界面
基本事件流	1. 管理员导航到某个任务的界面 2. 管理员点击编辑按钮 3. 管理员对 deadline、负责人等信息作出修改 4. 管理员点击修改按钮 5. 对于每一对任务，系统检查前一个任务的 ddl 是否至少比后一个任务早 1 天 6. 系统修改指定任务条目 7. 提示管理员任务修改成功，返回并刷新任务管理界面
异常事件流	点击修改按钮时网络故障，删除失败

续下页

续表 A.12　任务修改用例

后置条件	系统删除相应任务条目，提示删除成功，返回原界面并刷新

表 A.13　任务删除用例

用例编号	013
用例名称	任务删除
参与者	管理员
用例说明	管理员删除任务
前置条件	管理员登录，进入任务管理界面
基本事件流	1. 管理员导航到某个任务的界面 2. 管理员点击删除按钮 3. 系统删除指定任务条目 4. 提示管理员任务删除成功，返回并刷新文件管理界面
异常事件流	点击删除按钮时网络故障，删除失败
后置条件	系统删除相应任务条目，提示删除成功，返回原界面并刷新

表 A.14　完成任务用例

用例编号	012
用例名称	完成任务
参与者	所有用户
用例说明	用户执行指定的任务，并在应用中登记完成
前置条件	用户登录，进入任务管理界面
基本事件流	1. 用户在分配给自己的任务中选择一个特定任务 2. 用户上传相应材料（可选） 3. 用户点击完成按钮 4. 系统检查任务的完成条件是否满足 5. 系统更新任务状态 6. 系统提示提交成功，返回并刷新任务管理页面
异常事件流	点击完成按钮时网络故障，提交失败
后置条件	系统修改相应任务条目的属性，提示提交成功，返回原界面并刷新