



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

(深圳)

实验报告

开课学期: 2023 秋季
课程名称: 数据库系统
实验名称: 基于 django 的校园志愿者招募平台系统设计与实现
实验性质: 设计型
实验学时: 8 地点: T2608
学生班级: 计算机 7 班
学生学号: 210810521
学生姓名: 林靖杰
评阅教师:
报告成绩:

实验与创新实践教育中心制

2023 年 9 月

1 实验环境

1.1 实验所用的操作系统

Windows 11 家庭中文版

1.2 主要开发工具

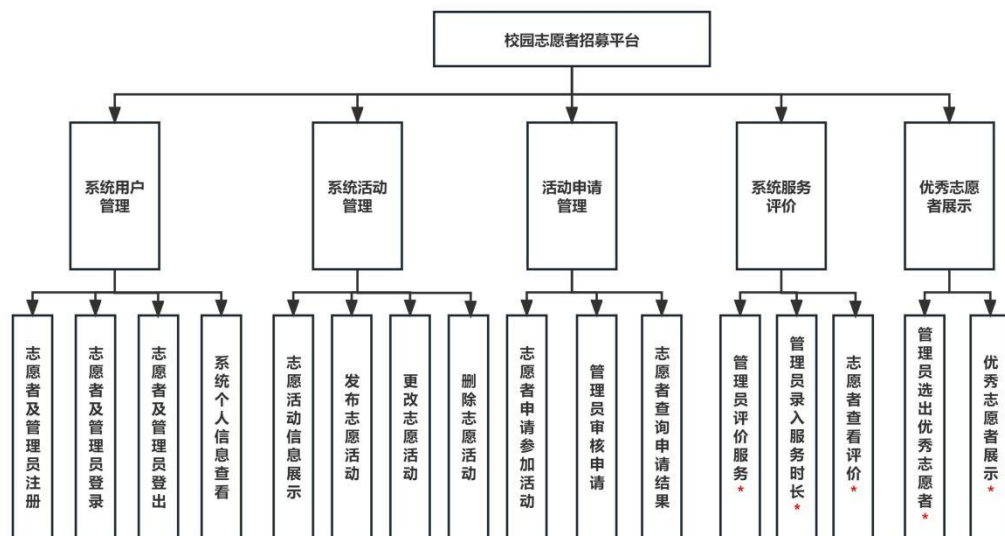
- 数据库 E-R 图设计：PowerDesigner 16.5
- 数据库后台管理系统：MySQL Workbench 8.0CE
- 前后端交互方案：基于 Django 构建的网页展示应用
- 前后端交互编程语言：Python
- 前端开发所用的 IDE：PyCharm 2022.2 (Professional Edition)

2 实验过程

2.1 系统功能

2.1.1 系统设计功能（即设计数据库时所考虑的功能）

根据本次实验所设计出的校园志愿者招募平台，绘制出系统的功能层次图，如下图所示（亮点功能用*标志）。



本次实验设计的校园志愿者招募平台主要有系统用户管理、系统活动管理、活动申请管理、系统服务评价、优秀志愿者展示这 5 大类功能。

（1）系统用户管理

在平台上，有两种用户类型：普通用户（志愿者）和平台管理员。这两种用户可以执行以下操作：

- **注册**：用户可以根据自身的身份进行注册，成为系统的一部分。
- **登录和登出**：用户可以使用已注册的账户登录到系统，同时也可以在不需要时安全登出。
- **查询个人信息**：用户在登录系统后可以查询自己在系统上的个人信息。

（2）系统活动管理

平台在首页展示志愿活动的信息，这些信息包括时间、地点、活动要求和活动人数等。无论用户是否登录，都可以查看发布在平台上的志愿活动信息。此外，平台管理员有以下操作权限：

- **添加活动**：管理员可以填写活动名称、时间、地点等活动信息，将新活动发布在首页供用户浏览。
- **修改活动**：管理员可以对现有的活动信息进行修改，并更新展示在首页。

- **删除活动：**管理员可以将活动信息从平台上移除。

同时，每个活动都有一个状态属性，可以表示活动是否正在报名或已过期。过期的活动会被系统自动关闭从而不可被申请并会被显示为已过期。

（3）活动申请管理

普通用户可以执行以下操作：

- **浏览活动信息：**用户可以在首页查看所有活动信息，无需登录。
- **申请参加活动：**用户登录后可以申请参加正在报名状态的活动。
- **查看申请状态：**用户可以查看自己已提交的活动申请的审核状态，包括待审核、通过或拒绝。

平台管理员可以执行以下操作：

- **查看申请人清单：**管理员可以查看针对他们发布的活动的申请人清单。
- **审核申请：**管理员可以审核申请人的活动申请，选择通过或拒绝。

（4）系统服务评价（亮点功能，已在功能层次图用*标志）

志愿者完成某次志愿活动的服务后，由发布该活动的管理员对志愿者的实际服务情况进行评价。服务评价包括以下内容：

- **录入实际服务小时数：**管理员记录该名志愿者在本次活动中的实际服务小时数。
- **评分志愿者的整体服务表现：**管理员对志愿者的整体服务表现进行评分，满分为10分，最低不能低于0分。
- **撰写评语：**管理员为志愿者的本次服务撰写评语。

完成服务评价后，志愿者可以查询评价结果。同时，志愿者的累计服务时长、平均服务得分等属性也会随之更新。

（5）优秀志愿者展示（亮点功能，已在功能层次图用*标志）

- **展示优秀志愿者的信息**
- **调用存储函数选择优秀志愿者**

2.1.2 系统实现功能（即实际编程时所实现的功能）

完成了该校园志愿者招募平台的设计后，实际编程实现了如下功能：

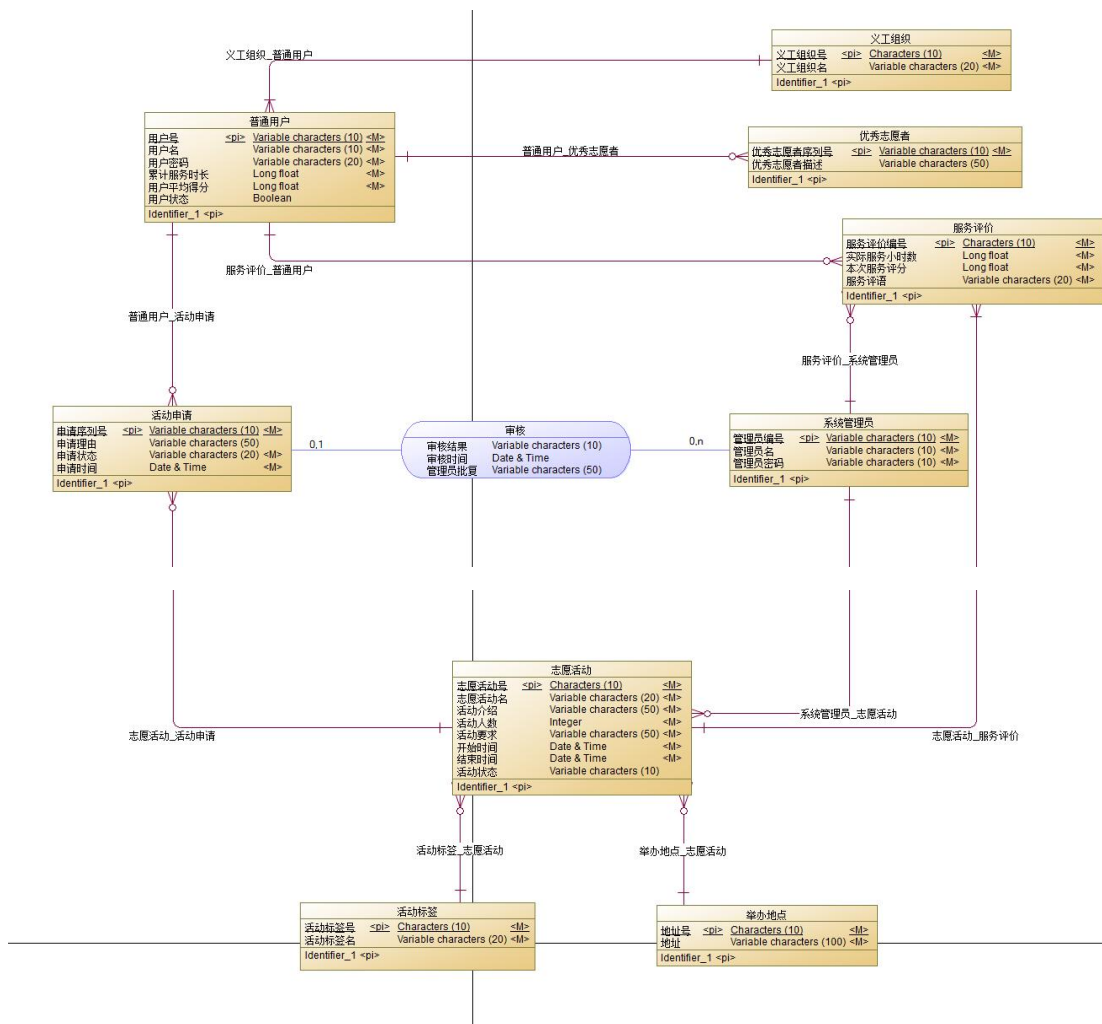
（亮点功能，已用*标志）

- 在未登录状态、志愿者登录状态或活动管理员登录状态下均能查看在首页查看全部志愿活动的信息；
- 志愿者、活动管理员两种用户进行注册、登录、登出，并查看个人信息；
- 活动管理员输入时间、地点、人数、要求等信息，发布新活动；
- 志愿者申请参加活动；
- 活动管理员查询自己所发布活动的申请人清单，对志愿者所提交的申请进行审核，并通过或拒绝志愿者的申请；
- 活动管理员能够修改已经发布的活动的信息、能够删除已经发布的活动。
- 志愿者查询申请结果，并能观察到自己提交的申请是待审核、通过还是拒绝状态。（*）
- 对于已经过期的活动的添加，系统会自动判断过期并将活动状态改为“已过期”，同时对于过期的活动，无法被普通用户申请。（*）
- 增加系统的超级管理员，形成了超级管理员—系统管理员—普通用户的权限架构。（*）
- 为网页架构增加了逐帧动画效果以及页面美化。（*）

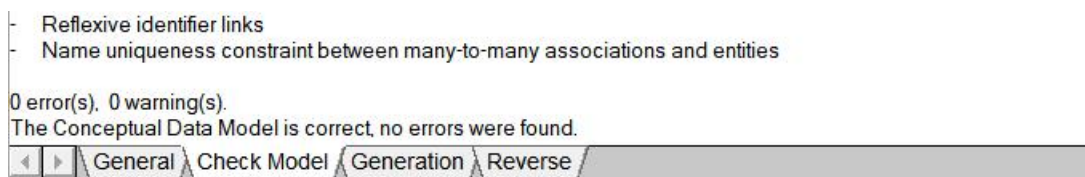
2.2 数据库设计

2.1.1 ER 图

E-R 图如下：



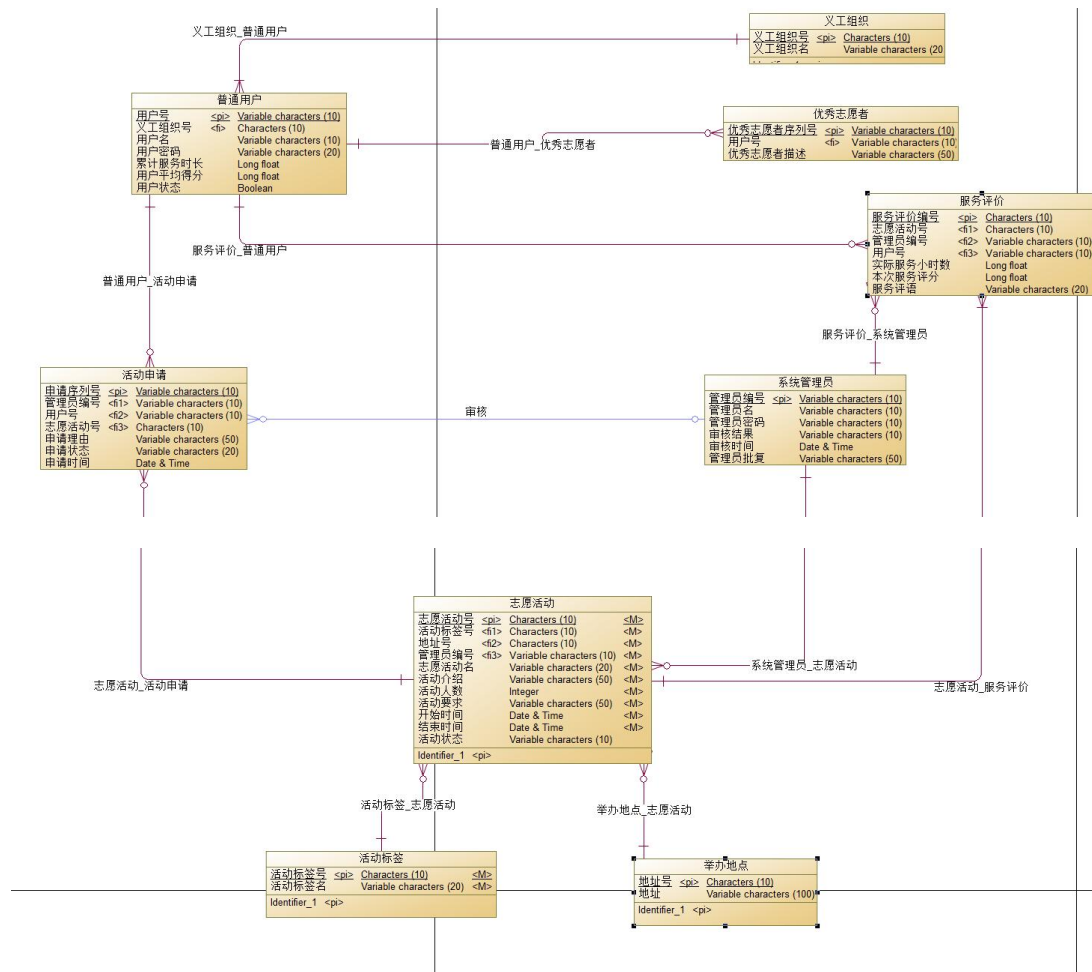
Check model 结果:



所设计的 ER 图总共有地点、活动类型、活动、申请、管理员、义工组织、普通用户、优秀志愿者、服务评价这 9 个实体，并包含有 11 个联系（其中有 1 个是本身包含了属性的关联“审核”）。

2.1.1.2 LDM 图

LDM 图如下：



Check model 结果:

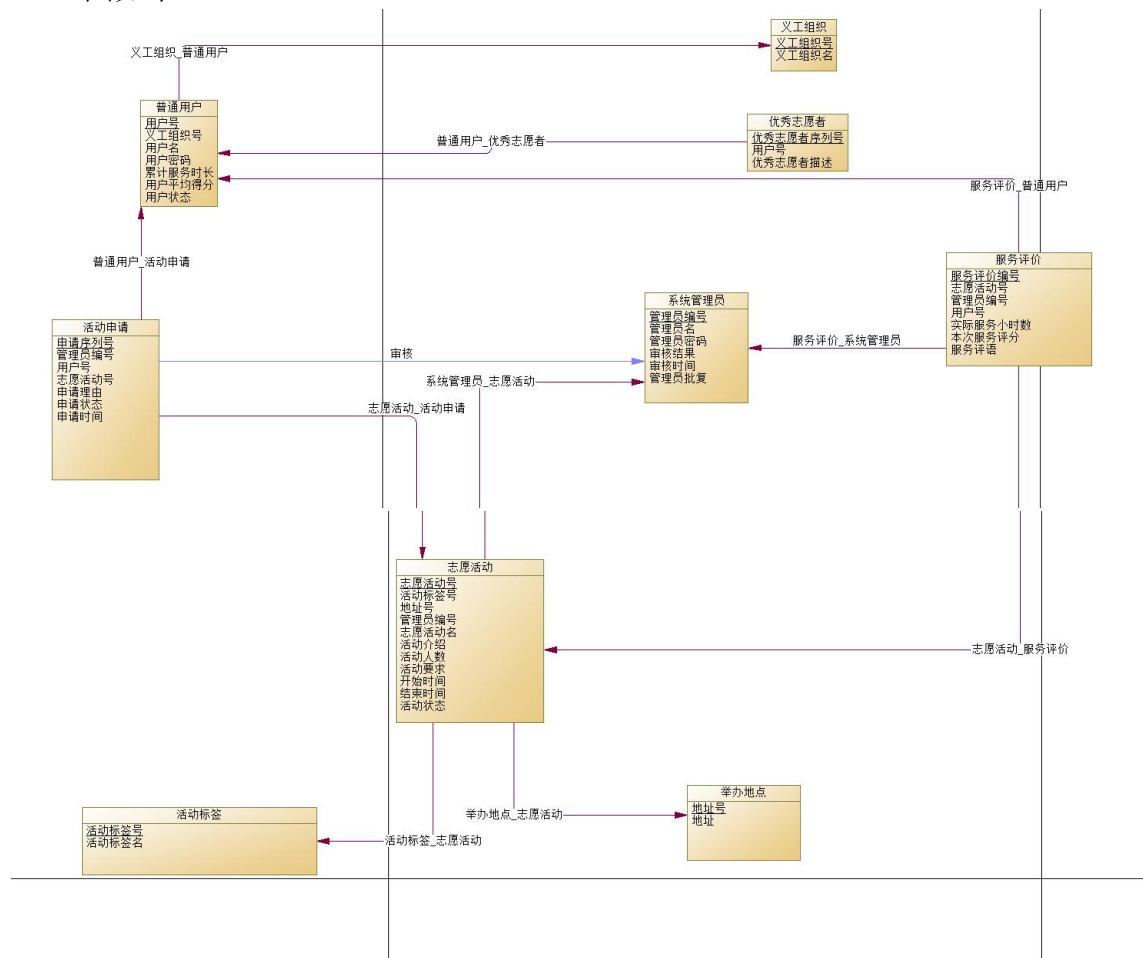
- Reflexive mandatory
- Bijective relationship between two entities
- 'Many - Many' relationships

0 error(s). 0 warning(s).

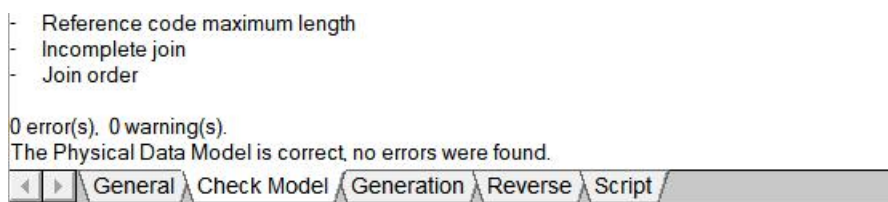
The Logical Data Model is correct, no errors were found.

2.1.3 PDM 图

PDM 图如下：



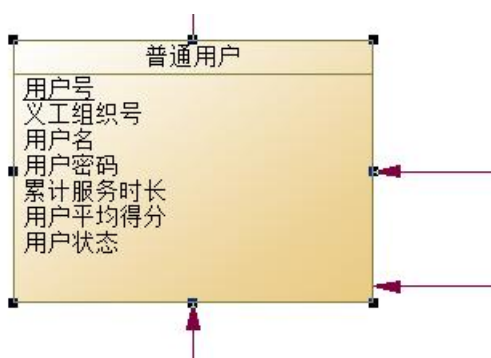
Check model 结果:



2.1.4 数据库表结构

1、表结构

(1) 普通用户 PDM 表:



表中的属性对应的命名为:

Column Name	Datatype	PK	NN	UQ	B
normal_user_id	VARCHAR(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
org_id	CHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
normal_user_name	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
normal_user_code	VARCHAR(20)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
serve_duration	DOUBLE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
average_score	DOUBLE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
normal_user_state	TINYINT(1)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

• 主键约束

普通用户（即志愿者）表的主键为 `normal_user_id`，用于每个志愿者都有一个唯一的用户编号来标识其记录。

• 外键约束

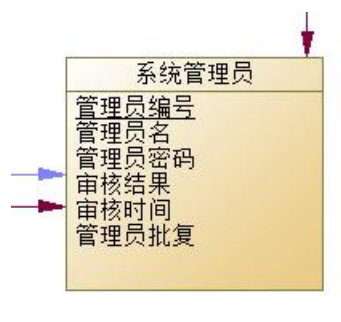
普通用户表包括外键约束，如义工组织编号（`organization_id`）。对于普通用户需要加入对应的义工组织方便进行信息管理，而在义工组织又以唯一的义工组织号进行

标识。且由于义工组织与普通用户之间属于一对多的关系，因而将义工组织（一端）的主键义工组织号加入普通用户中作为外键。







- 非空约束

在普通用户表中，被标记为“NN”选项的属性都被设置为非空属性（not null）。为了保证个人信息的完整性，因而将用户编号、义工组织编号、用户名、密码、累计服务时长、平均服务得分等属性加入非空约束。

（2）系统管理员 PDM 表：



表中的属性对应的命名为：

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 admin_id	VARCHAR(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 admin_name	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 admin_code	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 check_result	VARCHAR(10)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 check_time	DATETIME	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 check_back	VARCHAR(50)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

- 主键约束

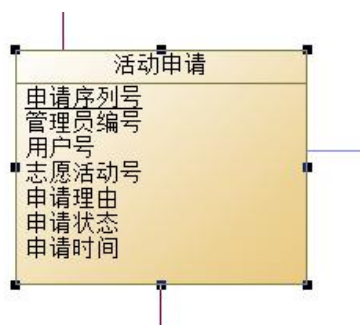
管理员表的主键为 admin_id，每个管理员都有一个唯一的管理员编号来标识其记录。

- 非空约束

在管理员表中，被标记为“NN”选项的属性都被设置为非空属性（not null）。这意味

着管理员编号、管理员名称、密码等属性的取值都不能为空。

(3) 活动申请 PDM 表:



表中的属性对应的命名为:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
application_id	VARCHAR(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
admin_id	VARCHAR(10)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
normal_user_id	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
activity_id	CHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
application_reason	VARCHAR(50)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
application_state	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
application_time	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

- 主键约束

活动申请表的主键为 `application_id`，用于每个活动申请都有一个唯一的序列号来标识其记录。

- 外键约束

活动申请表包括外键约束，如普通用户号（`normal_user_id`）。活动申请需要对应不同的用户，且由于活动申请与普通用户之间属于一对多的关系，因而将普通用户（一端）的主键普通用户号加入活动申请中作为外键。

- 非空约束

在普通用户表中，被标记为“NN”选项的属性都被设置为非空属性（not null）。为了保证个人信息的完整性，因而将活动序列号、普通用户号、活动序列号、申请状态、申请时间等属性加入非空约束。

2、索引

1) 索引截图

索引代码：

```
186  --
187  -- index: idx_label_name 方便用户进行活动类型的检索
188  --
189  • create index idx_label_name on activity_label
190  (
191      activity_label_name
192  );
193
194  --
195  -- index: idx_time 用于集中通过某一时刻的申请，方便管理员对特定时间段的申请进行审核（如批复时间不超过多少小时的规定）
196  --
197  • create index idx_time on application
198  (
199      application_time
200  );
```

插入后的索引如下：

Columns in table			
Column	Type	Nullable	Indexes
◇ activity_label_id	char(10)	NO	PRIMARY
◇ activity_label_name	varchar(20)	NO	idx_label_name

插入后的索引如下：

Columns in table			
Column	Type	Nullable	Indexes
◇ application_id	varchar(10)	NO	PRIMARY
◇ admin_id	varchar(10)	YES	FK_check
◇ normal_user_id	varchar(10)	NO	FK_普通用户_活动申请
◇ activity_id	char(10)	NO	FK_志愿活动_活动申请
◇ application_reason	varchar(50)	YES	
◇ applicaiton_state	varchar(20)	NO	
◇ application_time	datetime	NO	idx_time

2) 使用场景（用途）

索引 idx_label_name

- 考虑到用户索引具有降低搜索代价的效果，且系统在运用场景中存在用户可以通过活动类型的索引来实现查找一类活动类型，进而可以确定某一具体活动类型包含的所有活动的需求。因而为了方便对不同的活动类型进行管理并**加快查找速**

度，建立索引 idx_label_name。

索引 idx_time

•考虑系统上线之后为了提高普通用户对系统的满意度，要求管理员审核批复活动申请的时间不得超过 24 小时的要求（一日之内做出审核批复）这一场景，因而对某一时间段内的申请时间添加索引，使得系统在某一时间段大量申请请求的应用场景下仍然能够检索某一时间段内的申请并**加快查找和排序速度**，并进行集中审核批复。

3、视图

1) 视图截图

创建代码如下：

```
134 • create VIEW information_of_excellent_volunteers
135     as
136     select
137         nu.normal_user_id as id,
138         nu.normal_user_name as user_name,
139         nu.serve_duration as serve_time,
140         nu.average_score as score,
141         org.org_name as organization,
142         ev.excellent_volunteer_dis as discription
143     from
144         normal_user as nu,
145         organization as org,
146         excellent_volunteer as ev
147     where
148         nu.normal_user_id = ev.normal_user_id and
149         nu.org_id = org.org_id;
```

对应视图如下：



2) 使用场景（用途）

为了实现**优秀志愿者的展示功能**，需要额外建立一个视图用于展示优秀志愿者的信息，包括优秀志愿者的用户名、累计服务时长、平均得分、义工组织名和优秀志愿者的描述。同时不能展示出关于优秀志愿者的**隐私信息**，如用户密码(password)等。

4、触发器

1) 触发器截图

【触发器 1】自动设置申请时间为函数 now()的返回值

```
203  --
204  -- Triggers for application
205  --
206
207  delimiter ;;
208 • create trigger Get_Time_Before_Application
209      before insert on application
210      for each row
211      begin
212          set new.application_time = now();
213      end ;;
```

【触发器 2】新增服务评价后自动更新志愿者的累计服务时长、平均服务得分

```
201  delimiter ;;
202 • create trigger Update_time_and_comment after insert on comment
203  for each row
204  begin
205      update normal_user
206      set
207          normal_user.average_score = (normal_user.average_score * normal_user.serve_duration + new.score * new.serve_hour)
208          / (new.serve_hour + normal_user.serve_duration),
209          normal_user.serve_duration = normal_user.serve_duration + new.serve_hour
210      where
211          normal_user.normal_user_id = new.normal_user_id;
212  end ;;
```

【触发器 3】对于新加入的用户自动添加对应的账号状态为激活(false 为激活、true

为被封禁)

```
233  --
234  -- Triggers for user_state
235  --
236
237  delimiter ;;
238  • create trigger Set_User_State
239      before insert on normal_user
240      for each row
241      begin
242          set new.normal_user_state = False;
243      end ;;
```

2) 使用场景 (用途)

【触发器 1】自动设置申请时间为函数 now()的返回值

触发器 1 的功能是自动将志愿者的申请时间设置为当前时间(使用函数 now())。这意味着当创建一条志愿者申请记录时,系统将自动记录下当前的日期和时间作为申请时间。

【触发器 2】新增服务评价后自动更新志愿者的累计服务时长、平均服务得分

触发器 2 的功能是在管理员完成一次服务评价后,自动更新志愿者的累计服务时长(average_hours)和平均服务得分(average_score)。累计服务时长将被更新为旧值加上本次服务的实际时长,而平均服务得分将以单次服务评分按服务时长进行加权平均计算得出。这有助于跟踪志愿者的服务表现并及时更新他们的信息。并根据对应信息评选优秀志愿者。

【触发器 3】对于新加入的用户自动添加对应的账号状态为激活(false 为激活、true 为被封禁)

触发器 3 的功能是对于新加入的用户自动添加对应的账号状态为激活(false 为激活、true 为被封禁),这意味着当创建一条志愿者申请记录时,系统将激活

此用户，直到管理员将其封禁。

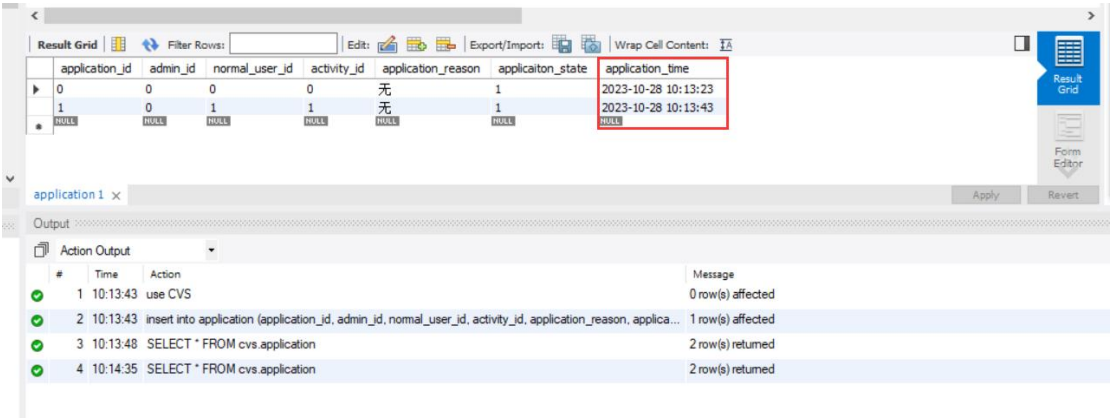
验证触发器

【触发器 1】

运行语句:

```
25 • insert into application
26     (application_id, admin_id, normal_user_id, activity_id, application_reason, applicaiton_state)
27     values(1, 0, 1, 1, '无', 1)
```

运行结果:



分析：发现申请时间自动填充为当前插入的时间，触发器生效;

【触发器 2】

分别运行语句:

```
10 -- insert into comment
11 -- (comment_id, activity_id, admin_id, normal_user_id, serve_hour, score, serve_comment)
12 --     values(0, 0, 0, 1, 1, 10, '很好')
13 -- insert into comment
14 -- (comment_id, activity_id, admin_id, normal_user_id, serve_hour, score, serve_comment)
15 --     values(1, 1, 0, 1, 1, 9, '很好')
```

用于表示管理员对同一志愿者参加两个不同的志愿活动的评价。

运行结果为:

Result Grid							
	normal_user_id	org_id	normal_user_name	normal_user_code	serve_duration	average_score	normal_user_state
▶	0	0	梁浩	lh111	0	0	0
▶	1	0	李梦飞	lmf111	2	9.5	0
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Action Output			Message
#	Time	Action	
1	10:23:41	rt into comment (comment_id, activity_id, admin_id, normal_user_id, serve_hour, score, serve_commen...	Error Code: 1064. You have an error in your SQL syntax; check the manual that
2	10:24:16	use CVS	0 row(s) affected
3	10:24:16	insert into comment (comment_id, activity_id, admin_id, normal_user_id, serve_hour, score, serve_com...	1 row(s) affected
4	10:24:33	SELECT * FROM cvs.normal_user	2 row(s) returned

分析:插入两条评论之后,触发器会自动更新 normal_user 表中的累计服务时间和平均服务分数这两条属性,对应图中红框中的值,触发器生效。

【触发器 3】

运行语句:

```

7 • insert into normal_user
8     (normal_user_id, org_id, normal_user_name, normal_user_code, serve_duration, average_score)
9     values(1, 0, '李梦飞', 'lmf111', 0, 0)

```

运行结果:

Action Output				Message
#	Time	Action		
28	09:59:17	create trigger Set_User_State before insert on normal_user for each row begin set n...	0 row(s) affected	
29	09:59:17	create procedure get_excellent_volunteer() begin select ev.normal_user_id as id, nu.normal_u...	0 row(s) affected	
30	09:59:17	create procedure Frozen_Account(in id varchar(10), in permission boolean) begin update normal_user...	0 row(s) affected	
31	10:00:06	use CVS	0 row(s) affected	
32	10:00:06	insert into organization (org_id, org_name) values(0, '计算机学院义工联合会') -- insert into normal...	1 row(s) affected	
33	10:00:35	use CVS	0 row(s) affected	
34	10:00:35	insert into normal_user (normal_user_id, org_id, normal_user_name, normal_user_code, serve_duration...	1 row(s) affected	
35	10:01:48	use CVS	0 row(s) affected	
36	10:01:48	insert into normal_user (normal_user_id, org_id, normal_user_name, normal_user_code, serve_duration...	1 row(s) affected	

Result Grid							
	normal_user_id	org_id	normal_user_name	normal_user_code	serve_duration	average_score	normal_user_state
▶	0	0	梁浩	lh111	0	0	0
▶	1	0	李梦飞	lmf111	0	0	0
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL

分析:即触发器自动添加了当前创建的用户状态(normal_user_state)为 0,表示激活该用户,触发器生效。

5、存储过程或存储函数

1) 存储过程或存储函数截图

【存储过程 1】根据平均服务得分对志愿者进行排序，选出优秀志愿者

```
231     delimiter ;;
232
233 •   create procedure get_excellent_volunteer()
234     begin
235         select
236             ev.normal_user_id as id,
237             nu.normal_user_name as user_name
238         from
239             excellent_volunteer as ev,
240             normal_user as nu
241         where
242             nu.normal_user_id = ev.normal_user_id
243         order by score desc
244         limit 5;
245     end ;;
```

【存储过程 2】将用户 id 号作为输入参数，冻结或激活对应用户 id 的账号

```
250     delimiter ;;
251
252 •   create procedure Frozen_Account(in id varchar(10), in permission boolean)
253     begin
254         update normal_user as nu
255         set nu.normal_user_state = permission
256         where nu.normal_user_id = id;
257     end ;;
258
259
```

2) 使用场景（用途）

【存储过程 1】根据平均服务得分对志愿者进行排序，选出优秀志愿者

• 为了实现定期选择出优秀志愿者的功能，需要根据平均服务得分对志愿者进行排序，并选择平均服务得分前五的志愿者作为优秀志愿者进行展示。存储过程返回的是对应选出的优秀志愿者的用户 id 和用户名。

【存储过程 2】将用户 id 号作为输入参数，冻结对应参数的用户账号

• 为了实现管理员对用户账号的管理功能，并对一些存在不合规的操作的用户进行冻结，或者将已经被封禁的账号重新激活。因而设置存储过程，使得管理员能够通过输入需要冻结用户对象的 id，将用户账号状态从激活改为冻结、或者从冻结状态重新激活。

3 收获和反思

按照实验老师的要求，首先给出实验过程中出现的问题：

- `not null` 参数设置不合理。在设计 E-R 图的时候，由于将审核这一联系的非空约束设置为 `not null`，而通过生成 `pdm` 将联系的属性放置在了多端的审核中，并在开始的 `sql` 数据库设计的时候并未注意此类问题。然而在前端实现的过程中，在超级管理员架构内创造普通管理员（`admin`）的过程中，出现了要求必须填写审核结果和审核理由的条目。这是不合理的，即在创建管理员的时候，就要求写出审核结果和审核理由，经过排查发现了上述错误，并在 E-R 以及 `sql` 实现中进行了进一步的改正。

- `sql` 迁移至 `django` 后端出现较多问题。本次实验一开始采用的是 `pdm->Mysql->python` 后端的数据库迁移模式，然而在实践过程中发现，`Mysql->python` 的迁移过程存在较大问题（出现外键丢失、非空约束丢失、属性缺乏中文命名表示的错误），因而通过查阅相关资料采用了 `pdm->Mysql` 再再 `python` 中重新建库，重新创建新的 `Mysql` 数据库并从 `Pyhon` 中迁移至其中的架构解决数据库问题。

- 创建表单错误，由于前端的相关函数逻辑实现出错，导致获得数据的 `django` 表单没有出现，即会出现创建网页的输入框没有显示的问题。且此类错误并不会出现在 `django` 的终端反馈之中，只能通过 `print` 方法进行 `debug` 以及错误排除。

- 美化页面加载错误，本次实验采用了 `github` 中的网页美化模板，但是由于美化模板中引入了部分国外网站的 `css` 以及 `json` 文件，导致出现页面长时间无法加

载的情况。此类错误通过终端错误排查以及引进国内镜像 `css` 文件进行解决。

通过本次实验，锻炼了我面对一个相对较大工程的总体设计和实践能力，在这两周付出了巨大的努力与精力将本次实验一步一步的设计并完成。在报告的最后，需要感谢实验老师每节实验课上给我们分享了往届同学易错的问题，确实帮助我避免了很多 `bug`。

最后，再次感谢实验老师的指导和付出，也祝愿数据库系统实验课程越办越好！