

# 2020 春《数据库系统》实验报告

## 实验 1：数据库系统开发

姓名：李净峰

学号：1173200214

班级：1703105

### 1. 实验目的

在熟练掌握 MySQL 基本命令、SQL 语言以及用 Python 3.6 编写 MySQL 操作程序的基础上，学习简单数据库系统的设计方法，包括数据库概要设计、逻辑设计。

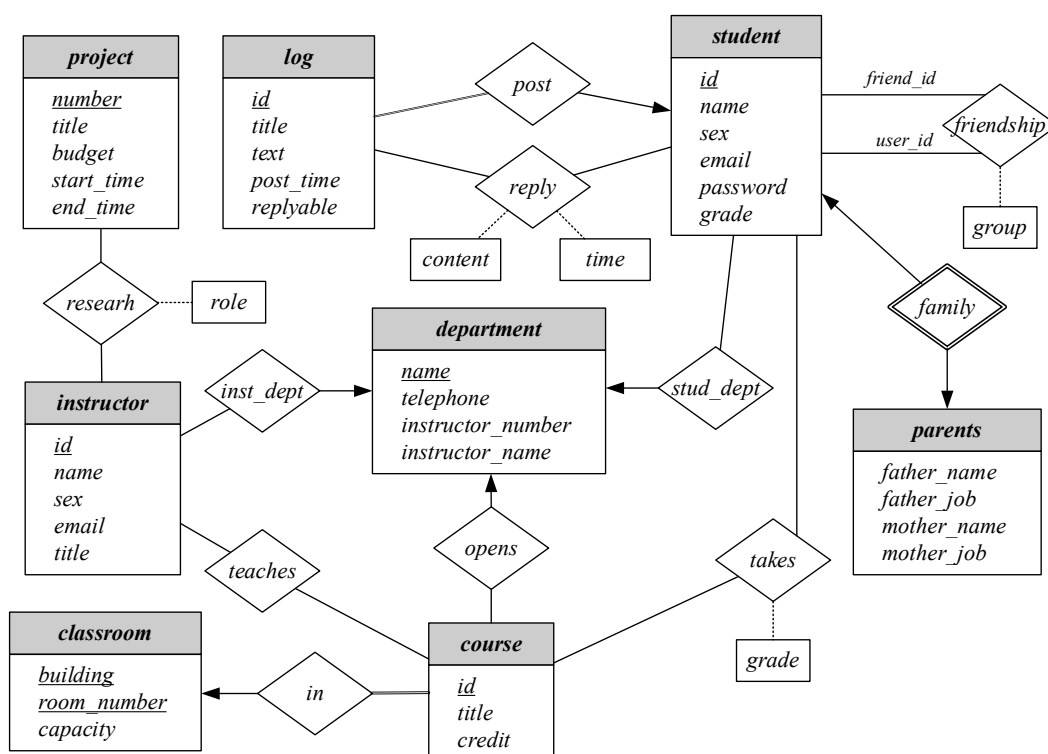
在本次实验中，我设计了一个校园网络系统(Campus Network System)，用来管理学校中学生、老师、课程及校内社交平台中的数据。

### 2. 实验环境

- 操作系统：Windows 10
- 开发环境：PyCharm
- 框架：Django, MySQLdb

### 3. 实验过程及结果

#### 3.1 E-R 图



## 3.2 关系模式

根据转换规则，将 E-R 图中的实体型、弱实体型、关系型转换为对应的关系模式。

department (name, building, telephone)  
instructor (inst\_id, name, sex, email, password, title, dept)  
student (stud\_id, name, sex, email, password, grade, dept)  
student\_family(stud\_id, father\_name, father\_job, mother\_name, mother\_job)  
course (cno, cname, credit, dept, building, room\_number)  
classroom (room\_number, capacity)  
student\_grade (stud\_id, cname, grade)  
teacher\_arrangement (id, cno)  
friendship (user\_id, friend\_id, group)  
log (log\_id, author\_id, title, text, post\_time, replyable)  
reply(user\_id, log\_id, content, time)

由定义知，以上设计的关系模式均满足 BCNF 范式。

(BCNF 定义：如果对于  $R$  的每个函数依赖  $X \rightarrow Y$ ， $X$  必为候选码，则  $R$  是 BCNF 范式)

## 3.3 关系的完整性约束

以关系 student 为例，在 MySQL 中创建表时，设置 stud\_id 为主键，dept 为外键(参照关系 department)。同时，将各属性值声明为 NOT NULL，并检查 sex 属性是否在('M', 'F')中。

```
create table student(  
  stud_id CHAR(10),  
  name VARCHAR(20) NOT NULL,  
  sex ENUM('M', 'F') NOT NULL CHECK (sex IN ('M', 'F')),  
  email VARCHAR(320) NOT NULL,  
  password VARCHAR(16) NOT NULL,  
  grade VARCHAR(10) NOT NULL,  
  dept CHAR(2) NOT NULL,  
  PRIMARY KEY (stud_id),  
  FOREIGN KEY (dept) REFERENCES department(name)  
);
```

## 3.4 创建视图

- 视图 1：向用户展示学生的信息时，并不需要将学生的密码列出，所以创建视图 student\_info，隐藏学生的密码；
- 视图 2：为了方便查看课程的选择人数，将课程信息、教室信息、学生选课信息连接起来，并做聚集操作，统计出每门课的选课人数。

## 3.4 建立索引

在创建表 student\_grade 时，建立了三种索引：

- 主索引：在声明 PRIMARY KEY 的同时自动创建，这里使用的是代理键；
- 外键索引：在声明 FOREIGN KEY 的同时自动创建；
- 唯一索引：给定学生和课程号，学生的成绩应该是唯一的。

```

student_grade | CREATE TABLE `student_grade` (
  `id` int NOT NULL AUTO_INCREMENT,
  `grade` int DEFAULT NULL,
  `cno` varchar(7) COLLATE utf8mb4_bin NOT NULL,
  `stud_id` varchar(10) COLLATE utf8mb4_bin NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `student_grade_stud_id_cno_38c1cf66_uniq` (`stud_id`,`cno`),
  KEY `student_grade_cno_d307e86d_fk_course_cno` (`cno`),
  CONSTRAINT `student_grade_cno_d307e86d_fk_course_cno` FOREIGN KEY (`cno`) REFERENCES `course` (`cno`),
  CONSTRAINT `student_grade_stud_id_cbc68a5a_fk_student_stud_id` FOREIGN KEY (`stud_id`) REFERENCES `student` (`stud_id`)
) ENGINE=InnoDB AUTO_INCREMENT=14 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin |

```

### 3.5 创建触发器

创建两个触发器，并建立一个新表 department\_history，在表 department 发生增加或删除时触发，并将操作记录到 department\_history 中。

```

create table department_history (
  id INT AUTO_INCREMENT,
  name CHAR(2) NOT NULL,
  operatetype varchar(50) NOT NULL,
  operatetime datetime NOT NULL,
  PRIMARY KEY (id)
);

DELIMITER ;;
CREATE TRIGGER tri_insert_department AFTER INSERT ON department FOR EACH ROW begin
  INSERT INTO department_history(name, operatetype, operatetime) VALUES (new.name, 'add a department', now());
end
;;
DELIMITER ;

DELIMITER ;;
CREATE TRIGGER tri_delete_department AFTER DELETE ON department FOR EACH ROW begin
  INSERT INTO department_history(name, operatetype, operatetime) VALUES (old.name, 'delete a department', now());
end
;;
DELIMITER ;

```

### 3.6 事务管理

事务管理是维护数据库的 ACID 特性(原子性、一致性、隔离性、持久性)。Django 提供了相应的模块进行事务管理。

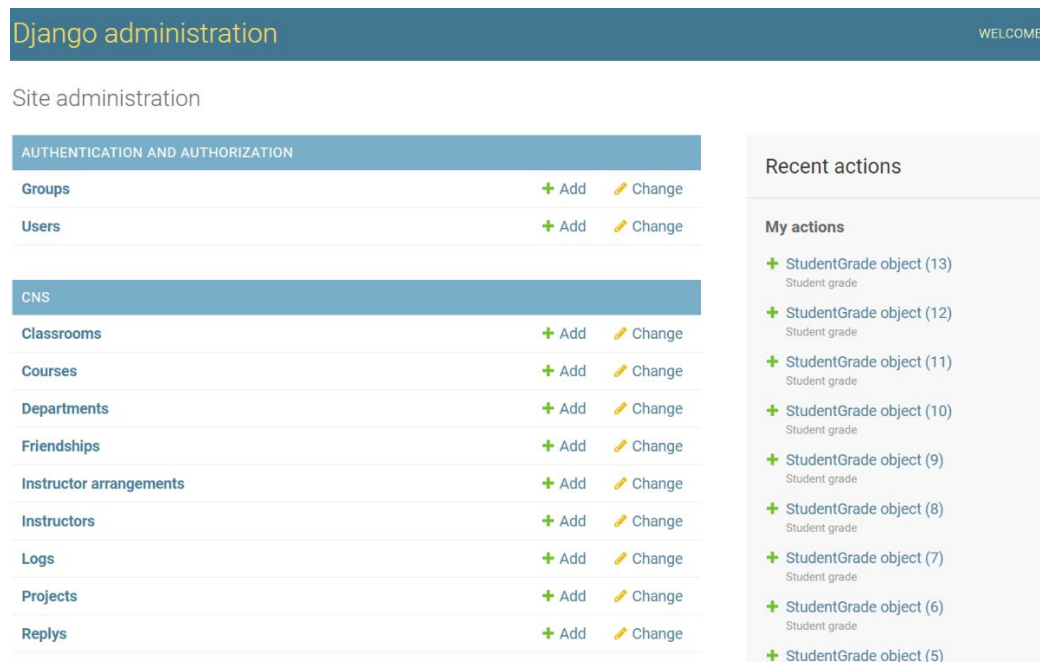
```

@transaction.atomic
def view_func(request):
    with transaction.atomic():
        try:
            save_id = transaction.savepoint()
            # ...
        except Exception as e:
            transaction.savepoint_rollback(save_id)
            # ...
        transaction.savepoint_commit(save_id)

```

### 3.7 系统功能

Django 提供了 Web 开发的框架，并且封装了许多对 MySQL 的操作，在 admin 中可以对所有的表进行增删改查。但其也提供了调用原生的 MySQL 语句，让开发者自定义开发。下图是 <http://127.0.0.1:8123/admin/> 的 web 页面。



限于时间和精力，我选取了部分的表进行操作，并写了一些没有渲染的 html 文件。以下  
为操作数据库的 web 页面。

- 对 department 表进行增加和删除操作

Department		
name	building	telephone
AS	Main Building	86403166
CS	Zonghe	86403260
EE	Dianji	86418297
ME	Jixie	86413809

#### Add Department

Name:

Building:

Telephone:

#### Delete Department

Name:

Add/Delete History		
name	type	time
AS	add a department	April 2, 2020, 7:24 p.m.
AS	delete a department	April 2, 2020, 7:24 p.m.
AS	add a department	April 2, 2020, 9:59 p.m.

- 查看视图 student\_info

student Information(view)				
stud id	name	sex	email	dept
1160600415	Wei Liu	M	weiliu@stu.hit.edu.cn	EE
1170300101	Wei Zhang	M	wei.zhang@stu.hit.edu.cn	CS
1170800718	Yong Zhang	M	yong.zhang@stu.hit.edu.cn	ME
1180300213	Xiuying Li	F	xiuying.li@stu.edu.cn	CS
1180801008	Qiang Wang	M	qiang.wang@stu.hit.edu.cn	ME
1190610305	Min Zhang	F	min.zhang@stu.hit.edu.cn	EE

- 学生选课界面

Course Enrollment				
#	name	instructors	enroll	capacity
CS31001	C++ Programming	Cheng Yu,Jie Li	4	180
CS32204	Modern Algebra	Cheng Yu	1	60
CS32272	Big Data Analytics	Juan Li	2	80
EE31012	Electric Circuit	Jun Wang	1	60
EE31023	Electronics	Xia Li,Juan Li	2	180
EE31606	Signal and system	Jun Wang,Xia Li	1	60
ME31012	Machinery Theory	Fang Liu	1	50
ME31204	Graphic Design	Fang Liu	1	60

- 查找学分最高的课程

Course				
course#	name	credit	dept	room number
CS31001	C++ Programming	2	CS	B11
CS32204	Modern Algebra	2	CS	A103
CS32272	Big Data Analytics	4	CS	C101
CS33505	Machine Learning	3	CS	B11
EE31012	Electric Circuit	3	EE	A101
EE31023	Electronics	3	EE	B12
EE31606	Signal and system	3	EE	A103
ME31012	Machinery Theory	3	ME	G801
ME31204	Graphic Design	2	ME	A103
ME32407	Quality Management	2	ME	C102

Maximum Credit Course				
course#	name	credit	department	classroom#
CS32272	Big Data Analytics	4	CS	C101

- 查找空闲的教室

Course	
room number	capacity
A101	60
A102	60
A103	60
B11	180
B12	180
C101	80
C102	180
C103	100
G801	50

Spare Classroom	
room number	
A102	
C103	

### 3.8 关键代码

在 view.py 中可以调用接口 `cursor.execute('mysql query')` 让 Django 程序执行原生的 MySQL 语法查询，具体代码如下(以函数 `addDepartment` 为例，其它函数的步骤均与其一致)。程序读入用户在 web 页面中输入的信息后，调用 MySQL 的 `insert` 语句就可以将新的院系信息插入到表 `department` 中。过程中如果违反了数据完整性约束或者违背了 ACID 特性，会抛出异常。

```
@transaction.atomic
def addDepartment(request):
    with transaction.atomic():
        try:
            save_id = transaction.savepoint()
            name = request.POST.get('name')
            building = request.POST.get('building')
            telephone = request.POST.get('telephone')
            cursor = connection.cursor()
            cursor.execute('insert into department (name,building,telephone)
                           values (%s,%s,%s);',[name,building,telephone])
        except Exception as e:
            transaction.savepoint_rollback(save_id)
            return HttpResponse('Oops, an error has occurred!')
        transaction.savepoint_commit(save_id)
    return HttpResponse('Success!')
```

## 4. 实验心得

**问题 1:** Django 不支持复合属性作为主键或者外键。

**解决方法:** 使用代理键作为主键。对于复合属性作为外键的情况，将复合属性中的属性分别设置为外键，并在符合属性上建立唯一(unique)索引，可以起到一致的效果。

**问题 2:** 在 NOT NULL 约束下，在插入元组时属性仍然可以接收空字符串

**解决方法:** 在 MySQL，NULL 和空字符串是两个概念，因此 NOT NULL 并不能拒绝空串的插入，因此，在创建关系表的时候，应该加上 `CHECK(xx!='')`