

哈尔滨工业大学

<<数据库系统>>

实验报告之一

(2021 年度春季学期)

姓名：	郭茁宁
学号：	1183710109
学院：	计算机学院
教师：	高宏

实验一 MySQL 关系数据库管理系统及 SQL 语言的使用

一、实验目的

1. 掌握 MySQL 关系数据库管理系统的基本命令，并熟练使用 SQL 语言管理 MySQL 数据库。
2. 掌握 SQL 语言的使用方法，学会使用 SQL 语言进行关系数据库查询，特别是聚集查询、连接查询和嵌套查询。

二、实验环境

Windows 10 操作系统、MySQL 5.7.19 版本。

三、实验过程及结果

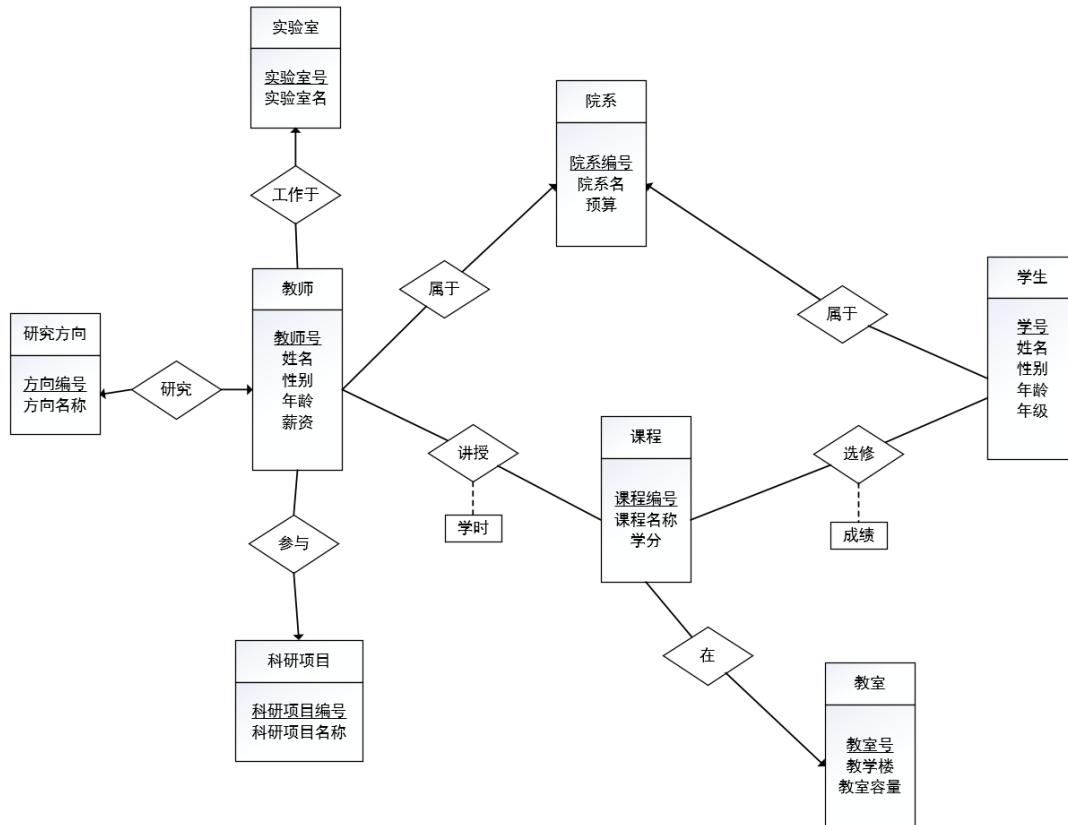
本数据库为一教务系统，基本实体类有教师、学生、课程、教室、院系、实验室、科研项目、研究方向。实体与实体的联系分析如下：

教师、学生均有所在院系，且为一对多联系，每个人只能有 1 个院系，一个院系可以有 multiple 人。

- 教师讲授课程，此为多对多联系，一个教师可以教授多门课程，一门课可以由多个教师教授。
- 学生可以选修课程，此为多对多联系，一个学生可以选修多门课程，一门课可以由多个学生选修。
- 课程需要在教室中进行，此为多对一联系，一门课只能在一个教室，而一个教室在不同时间可以有 multiple 门课。
- 每个教师需要有研究方向，此为多对一联系，一个教师只能有一个研究方向，一个研究方向只能有一个教师。
- 每个老师有所在的实验室，此为多对一联系，一个教师只能在一个实验室，一个实验室可以有 multiple 个教师。
- 每个教师都参与科研项目，此为多对一联系，一个教师只能参与一个科研项

目，一个科研项目可以有多个教师。

E-R 图:



关系表:

研究方向 (方向编号, 方向名称)

科研项目 (科研项目编号, 科研项目名称)

实验室 (实验室号, 实验室名)

院系 (院系编号, 院系名, 预算)

教室 (教室号, 教学楼, 教室容量)

课程 (课程编号, 课程名称, 学分, 教室号)

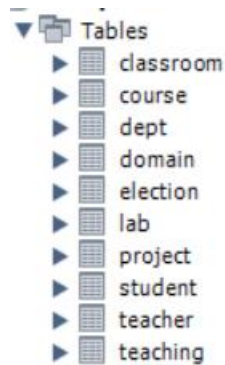
讲授关系 (课程编号, 教师号, 学时)

选修关系 (课程编号, 学号, 成绩)

教师（教师号，姓名，性别，年龄，薪资，方向编号，科研项目编号，实验室号，院系编号）

学生（学号，姓名，性别，年龄，年级，院系编号）

MySQL Workbench 中构建后：



且由于本数据库内的所有关系满足无复合属性多值属性、无非键属性对候选码的部分函数依赖、无非键属性对候选码的传递函数依赖、无键属性对候选码的部分和传递函数依赖，所以它们都满足 BCNF 范式。

关系的完整性约束：

■ 实体完整性约束：

如图，NN 表示非空：






Table Name:

Charset/Collation:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default
 domain_id	INT	<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"/>	
 domain_name	VARCHAR(45)	<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"/>	






Table Name:

Charset/Collation:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default
 project_id	INT	<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"/>	
 project_name	VARCHAR(45)	<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"/>	

等等。

■ 参照完整性约束：

如图，设置外键：

Table Name: Schema: **edusystem**

Charset/Collation: Engine:

Comments:

Foreign Key Name	Referenced Table	Column	Referenced Column
classroom_jd	edusystem`.`classroom`	<input checked="" type="checkbox"/> classroom_id	classroom_id

Columns Indexes **Foreign Keys** Triggers Partitioning Options

等等。

创建的视图：

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
  VIEW `csstudent_view` AS
  SELECT
    `student`.`student_id` AS `student_id`,
    `student`.`student_name` AS `student_name`,
    `student`.`age` AS `age`,
    `student`.`gender` AS `gender`,
    `student`.`grade` AS `grade`,
    `student`.`dept_id` AS `dept_id`
  FROM
    `student`
  WHERE
    (`student`.`dept_id` = 3)
```

最关键代码：

java 中通过 JDBC 来连接 MySQL 数据库：首先要加载 JDBC DRIVER，然后再通过 URL、用户名、密码与数据库进行连接。

```
public class Connect {
    static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver"; //JDBC driver name
    static final String DB_URL = "jdbc:mysql://127.0.0.1:3306/edusystem?user=root&serverTimezone=Asia/Shanghai"; //数据库URL

    static final String USER = "root"; //用户名
    static final String PASS = "chen5211999"; //密码

    public static Connection getConn(Connection conn){
        // Register JDBC driver
        try {
            Class.forName(JDBC_DRIVER);
            System.out.println("成功加载Mysql Driver!");
        } catch (Exception e) {
            System.out.println("加载Mysql Driver失败!");
            e.printStackTrace();
        }
        //连接数据库
        try {
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            conn.setAutoCommit(false);
            System.out.println("成功连接到数据库!");
        } catch (Exception e) {
            System.out.println("连接到数据库失败!");
            e.printStackTrace();
        }
        return conn;
    }
}
```

因为教师与学生的基本信息表示中用名字比较好，所以要得到教师与学生的基本信息需要通过连接查询：

```
//连接查询学生信息
public ResultSet getbase() throws SQLException
{
    ResultSet rs = null;

    String sql = "select * from student,dept where student_id = ? and password = ? and student.dept_id = dept.dept_id";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setInt(1, id);
    pstmt.setString(2, password);
    rs=pstmt.executeQuery();

    return rs;
}

//连接查询教师信息
public ResultSet getbase() throws SQLException
{
    ResultSet rs = null;

    String sql = "select * from teacher,dept,domain,lab,project "
        + "where teacher_id = ? and password = ? and "
        + "teacher.dept_id = dept.dept_id and teacher.lab_id = lab.lab_id and teacher.domain_id = domain.domain_id "
        + "and teacher.project_id = project.project_id";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setInt(1, id);
    pstmt.setString(2, password);
    rs=pstmt.executeQuery();

    return rs;
}
```

教师增加自己教授的课程时需要用到插入：

```
//插入
public void addcourse(int course_id , int hour) throws SQLException
{
    String sql = "insert into teaching(course_id,teacher_id,hour) values(?,?,?);";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setInt(1, course_id);
    pstmt.setInt(2, id);
    pstmt.setInt(3, hour);
    int judge = pstmt.executeUpdate();

    if(judge > 0)
        main.conn.commit();
    else
        main.conn.rollback();
}
```

教师删除自己教授的课程时需要用到删除:

```
//删除
public void dropcourse(int course_id) throws SQLException
{
    String sql = "delete from teaching where course_id=? and teacher_id=?";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setInt(1, course_id);
    pstmt.setInt(2, id);
    int judge = pstmt.executeUpdate();

    if(judge > 0)
        main.conn.commit();
    else
        main.conn.rollback();
}
```

教师和学生更新密码时需要用到更新:

```
//更新
public void updatepassword(String password) throws SQLException
{
    if(password.equals(""))
        password = null;
    String sql = "update teacher set password=? where teacher_id=?";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setString(1, password);
    pstmt.setInt(2, id);
    int judge = pstmt.executeUpdate();

    if(judge > 0)
        main.conn.commit();
    else
        main.conn.rollback();
}
```

财务部门在查询相应纳税信息时,需要选择预算大于某值的系中薪水大于某值的教师,这需要用到嵌套查询:

```
//嵌套查询
public static ResultSet getfinance(float budget , float salary) throws SQLException
{
    ResultSet rs = null;

    String sql = "select * from (select * from dept where budget > ?) as ndept , teacher "
        + "where salary > ? and teacher.dept_id = ndept.dept_id";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setFloat(1, budget);
    pstmt.setFloat(2, salary);
    rs=pstmt.executeQuery();

    return rs;
}
```

财务部门在统计相应纳税信息时，需要按系分组统计人数，并且人数太少不计入统计，所以需要用到分组查询、having 语句。

```
//分组查询
public static ResultSet group(float budget , float salary) throws SQLException
{
    ResultSet rs = null;

    String sql = "select dept_name,count(*) from (select * from dept where budget > ?) as ndept , teacher "
        + "where salary > ? and teacher.dept_id = ndept.dept_id " + "group by dept_name having count(*)>5 "
        + "order by count(*) desc";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setFloat(1, budget);
    pstmt.setFloat(2, salary);
    rs=pstmt.executeQuery();

    return rs;
}
```

事务管理:

```
//插入
public void addcourse(int course_id , int hour) throws SQLException
{
    String sql = "insert into teaching(course_id,teacher_id,hour) values(?,?,?);";
    PreparedStatement pstmt = main.conn.prepareStatement(sql);
    pstmt.setInt(1, course_id);
    pstmt.setInt(2, id);
    pstmt.setInt(3, hour);
    int judge = pstmt.executeUpdate();

    if(judge > 0)
        main.conn.commit();
    else
        main.conn.rollback();
}
```

触发器:

```
CREATE DEFINER='root'@'localhost' TRIGGER `student_BEFORE_INSERT` BEFORE INSERT ON `student` FOR EACH ROW {
    if new.grade > 4 then
        set new.grade = 4;
    elseif new.grade < 1 then
        set new.grade = 1;
    end if;
END
```

运行界面:

登录界面:

登录界面

请登录

id:

password:

☐ Financial Department

☐ Teacher

☐ Student

登录

登录后的个人信息界面：

教师信息

个人信息

教师号: 姓名:

性别: 年龄:

研究方向: 科研项目:

所在系: 实验室名:

薪水: 密码:

修改密码 课程情况

学生信息

个人信息

学号:

姓名:

性别:

年龄:

年级:

所在系:

密码:

修改密码 查看课表

增删课程界面：

课程情况

应教课程

3 C3: 24

4 C4: 24

5 C5: 36

6 C6: 48

7 C7: 60

8 C8: 72

课程修改

课程编号:

讲授学时:

添加

删除

财务部门查询界面：

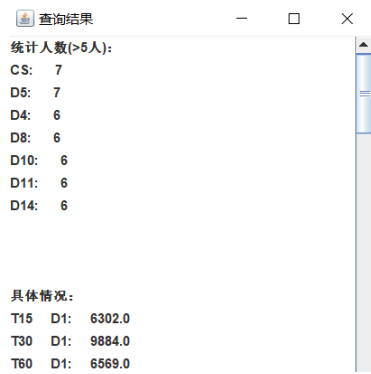
薪金情况查询

系预算大于:

教师薪水大于:

查询

财务部门查询结果统计及细节界面：



统计人数(>5人):	
CS:	7
D5:	7
D4:	6
D8:	6
D10:	6
D11:	6
D14:	6

具体情况:	
T15	D1: 6302.0
T30	D1: 9884.0
T60	D1: 6569.0

四、实验心得

通过本次实验收获如下

1. 掌握了关联数据库系统 MySQL 的使用;
2. 对于 SQL 语句的书写更加了解;
3. 掌握了在高级语言中如何使用嵌入式 SQL 对数据库进行操作;
4. 学会了怎么根据实际需求设计数据库, 绘制 ER 图, 并最终编程实现。