

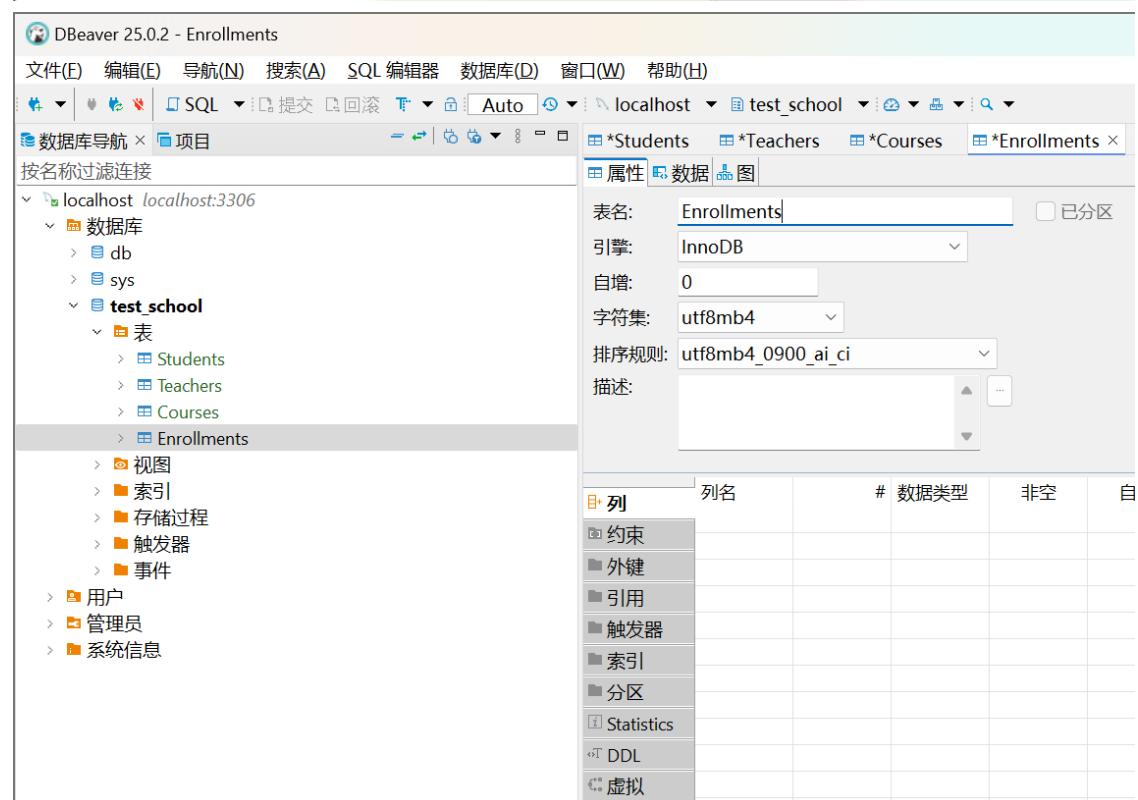
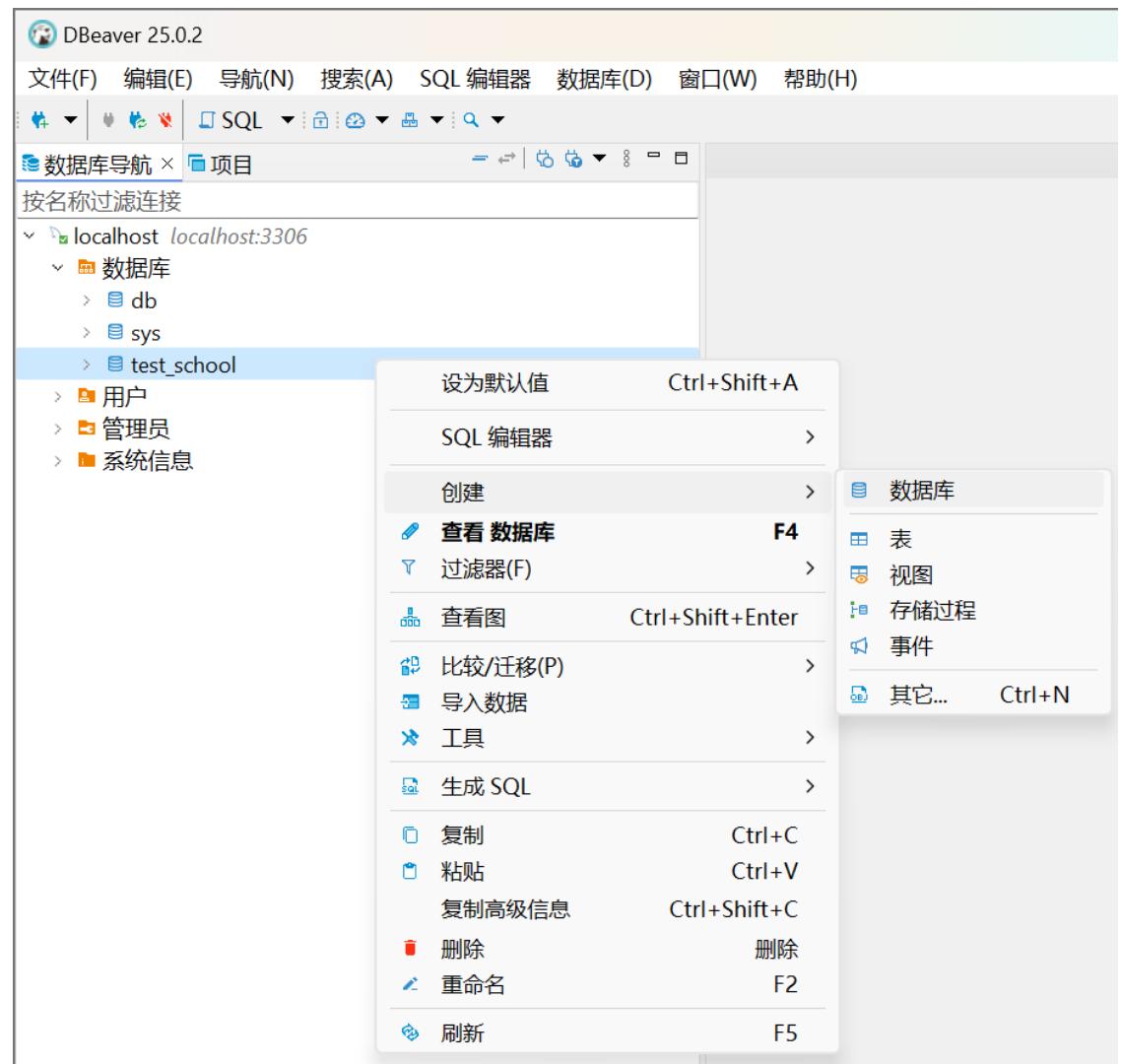
实验环境

- Windows 11 系统
- MySQL 9.2
- DBeaver 25.0.2

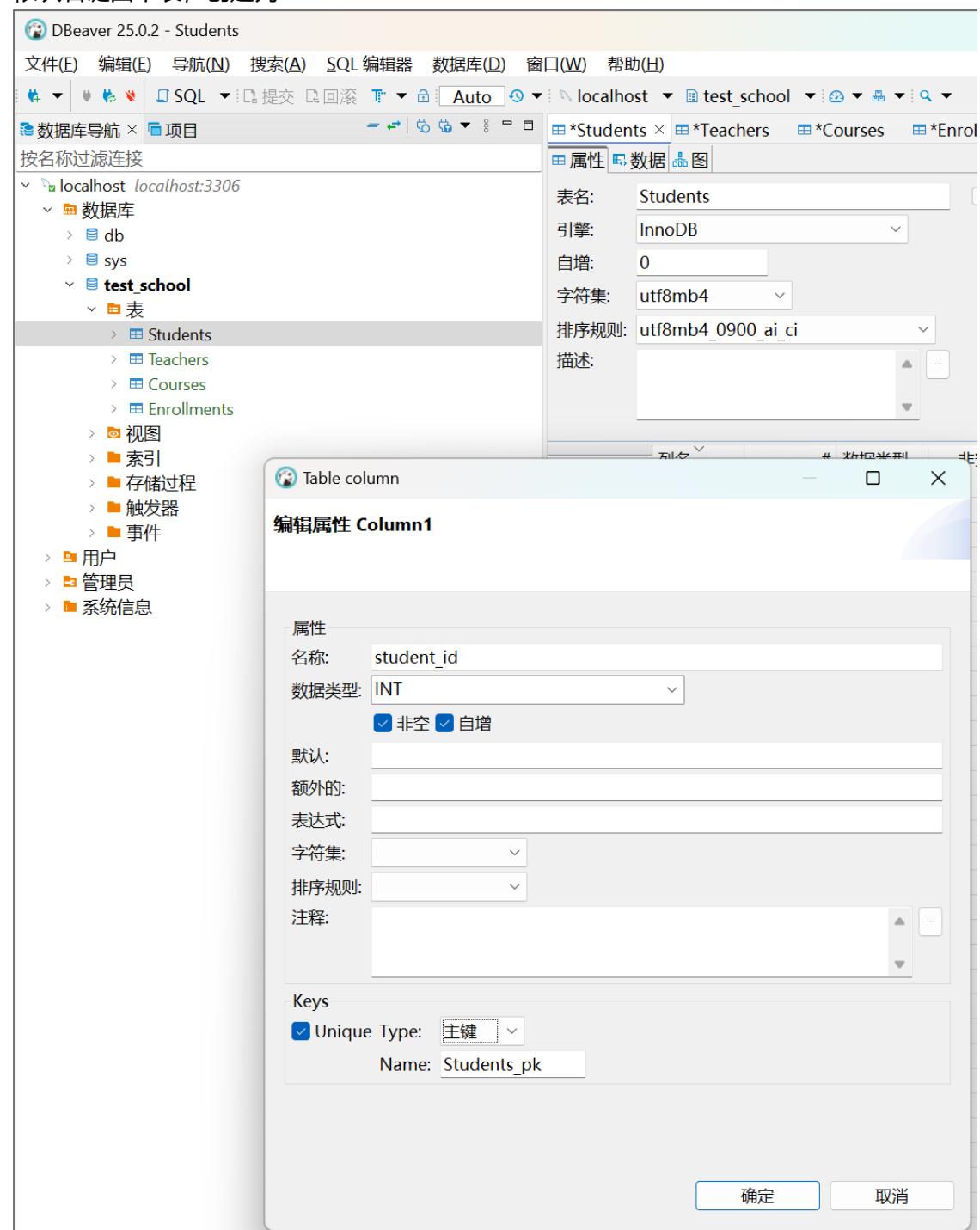
实验内容与完成情况

- 数据库设计 (学生选课)
 - 学生表 Students
 - student_id(primary key)(INT)
 - name(VARCHAR)
 - email(VARCHAR)
 - 教师表 Teachers
 - teacher_id(primary key)(INT)
 - name(VARCHAR)
 - email(VARCHAR)
 - 课程表 Courses
 - course_id(primary key)(INT)
 - name(VARCHAR)
 - description(VARCHAR)
 - teacher_id(foreign key)
 - 选课表 Enrollments
 - student_id(primary key)(foreign key)
 - course_id(primary key)(foreign key)
 - grade(FLOAT)
- 数据定义
 - 表的创建
 1. 连接到 MySQL 后右键数据库，创建 test_school 数据库

2. 右键 test_school 数据库，依次创建上述四个表



3. 依次右键四个表，创建列

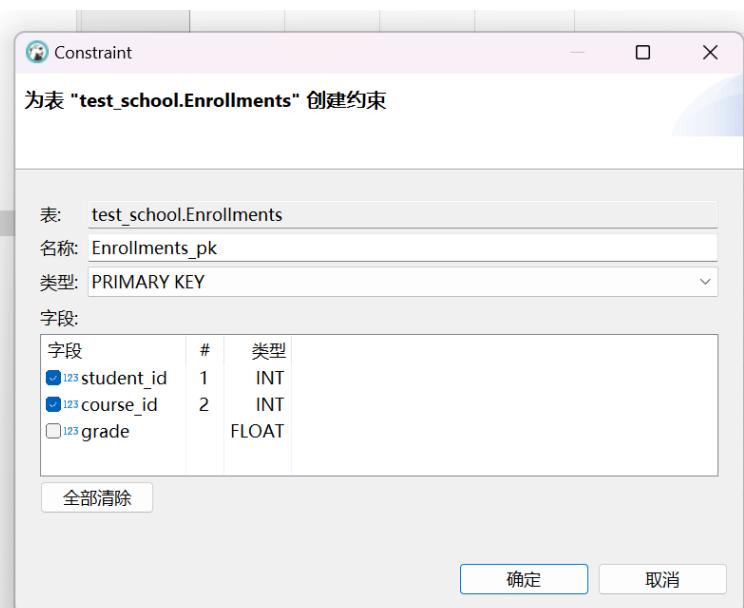


4. 设置 Courses 表和 Enrollments 表的外键

- › **Courses**
 - › **列**
 - › **约束**
 - › **外键**
 - alink Courses_Teachers_FK
 - › **引用**
 - › **触发器**
 - › **索引**
 - › **分区**
- › **Enrollments**
 - › **列**
 - › **约束**
 - › **外键**
 - alink Enrollments_Students_FK
 - alink Enrollments_Courses_FK
 - › **引用**
 - › **触发器**
 - › **索引**
 - › **分区**

5. 设置 Enrollments 表的约束

- › **触发器**
- › **索引**
- › **分区**
- › **Enrollments**
 - › **列**
 - 123 student_id (INT)
 - 123 course_id (INT)
 - 123 grade (FLOAT)
 - › **约束**
 - › **外键**
 - alink Enrollments_Students_FK
 - alink Enrollments_Courses_FK
 - › **引用**
 - › **触发器**
 - › **索引**
 - › **分区**
 - › **视图**
 - › **索引**
 - › **存储过程**
 - › **触发器**
 - › **事件**
- › **信息**

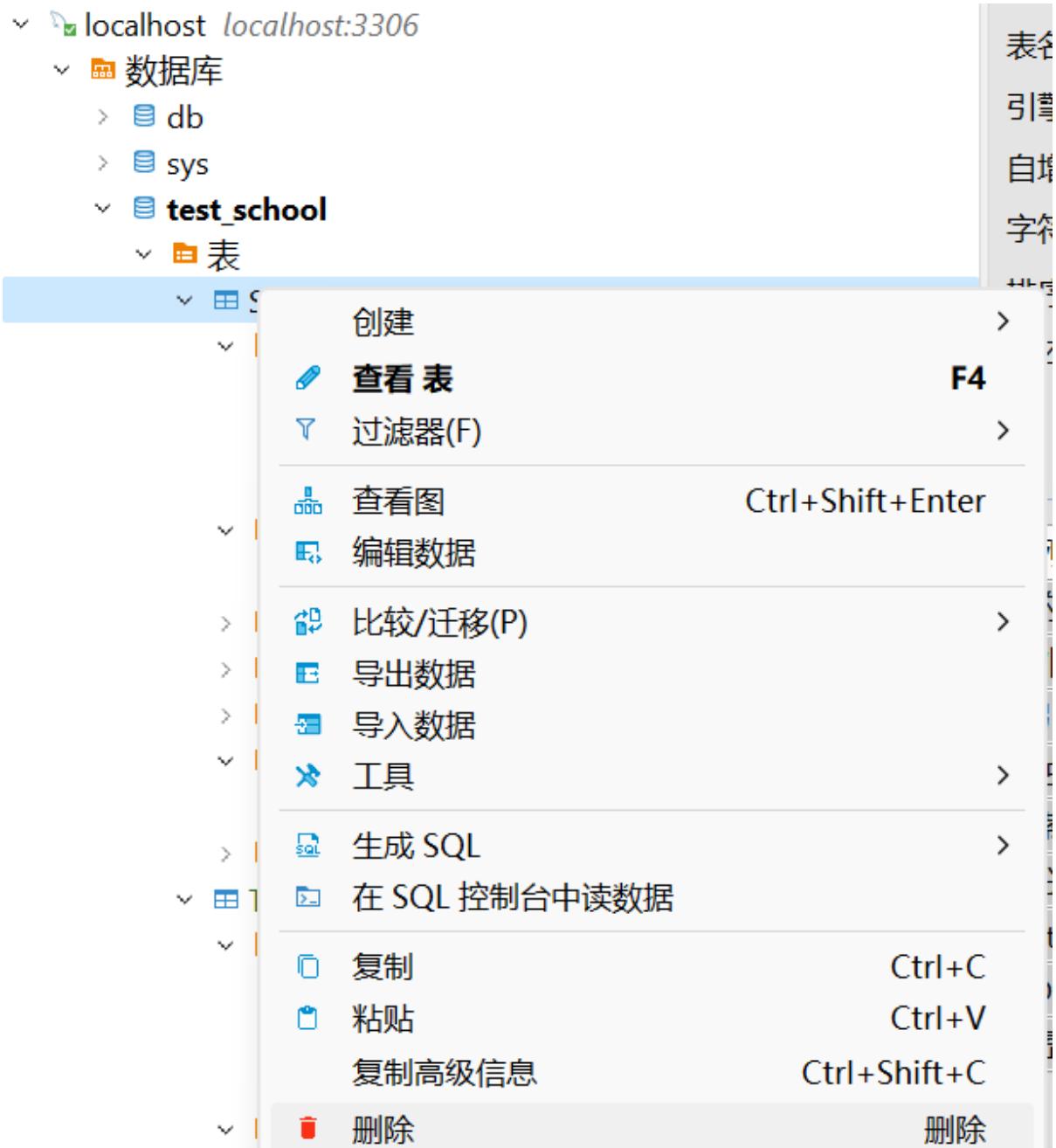


- 表的修改 双击 Students 表的 student_id 列的自增选项 []，成功设置自增属性

The screenshot shows the MySQL Workbench interface with the 'Properties' tab selected for the 'Students' table. The 'Identity' field is set to 1. The 'Columns' table below lists the table structure:

列	列名	#	数据类型	非空	自增
约束	<code>123 studer</code>	1	INT	[v]	[v]
外键	<code>AZ name</code>	2	VARCHAR	[v]	[]
引用	<code>AZ email</code>	3	VARCHAR	[]	[]
触发器					
索引					
分区					
Statistics					
DDL					
虚拟					

- 表的删除 右键表即可删除表



- 索引的创建 创建主键索引

为表 "test_school.Teachers" 创建索引

表: test_school.Teachers
类型: BTree
唯一:

字段:

字段	#	类型	排序	长度
<input checked="" type="checkbox"/> teacher_id	1	INT	ASC	
<input type="checkbox"/> name		VARCHAR	ASC	
<input type="checkbox"/> email		VARCHAR	ASC	

确定 取消

- 索引的删除 右键索引即可删除索引

右键菜单选项:

- 新建索引 Alt+插入
- 查看索引 F4
- 过滤器(F) >
- 比较/迁移(P) >
- 工具 >
- 复制 Ctrl+C
- 粘贴 Ctrl+V
- 复制高级信息 Ctrl+Shift+C
- 删除 (当前被高亮)

- 数据操作

- 插入数据 点击 SQL 按钮，输入对应语句进行插入

```

INSERT INTO Students (student_id, name, email) VALUES
(1, '张三', 'zhangsan@example.com'),
(2, '李四', 'lisi@example.com');
INSERT INTO Teachers (teacher_id, name, email) VALUES
(101, '王老师', 'wanglaoshi@example.com'),
(102, '赵老师', 'zhaolaoshi@example.com');

```

```

INSERT INTO Courses (course_id, name, description, teacher_id) VALUES
(201, '数据库原理', '介绍数据库的基本概念与SQL语言', 101),
(202, '操作系统', '讲解操作系统的工作原理与机制', 102);
INSERT INTO Enrollments (student_id, course_id, grade) VALUES
(1, 201, 88.5),
(2, 202, 92.0);

```

The screenshot shows the MySQL Workbench interface. The main area displays a script editor with the following SQL code:

```

students  teachers  courses  enrollments  *<localhost> Script-2 ×
+-----+
| - e INSERT INTO Students (student_id, name, email) VALUES
|   (1, '张三', 'zhangsan@example.com'),
|   (2, '李四', 'lisi@example.com');
| - e INSERT INTO Teachers (teacher_id, name, email) VALUES
|   (101, '王老师', 'wanglaoshi@example.com'),
|   (102, '赵老师', 'zhaolaoshi@example.com');
| - e -- INSERT INTO Courses (course_id, name, description, teacher_id) VALUES
|   -- (201, '数据库原理', '介绍数据库的基本概念与SQL语言', 101),
|   -- (202, '操作系统', '讲解操作系统的工作原理与机制', 102);
| - e -- INSERT INTO Enrollments (student_id, course_id, grade) VALUES
|   -- (1, 201, 88.5),
|   -- (2, 202, 92.0);

```

The bottom right corner of the main window shows a "统计 1" (Statistics 1) window with the following data:

Name	Value
Updated Rows	2
Execute time	0.031s
Start time	Sun May 11 15:08:12 CST 2025
Finish time	Sun May 11 15:08:12 CST 2025
Query	INSERT INTO Teachers (teacher_id, name, email) VALUES (101, '王老师', 'wanglaoshi@example.com'), (102, '赵老师', 'zhaolaoshi@example.com')

- 修改数据 修改学号为 1 的学生的邮箱

```

UPDATE Students
SET email = 'test@example.com'
WHERE student_id = 1;

```

students x teachers courses enrollments *

属性 数据 图

students | 输入一个SQL表达式来过滤结果(使用Ctrl+Space)

	student_id	name	email
1	1	张三	test@example.com
2	2	李四	lisi@example.com

- 删除数据 删除学号为 2 的学生的选课记录

```
DELETE FROM Enrollments
WHERE student_id = 2
```

- 单表查询 查询所有教师的名字与邮箱

```
SELECT name, email
FROM Teachers;
```

teachers 1 x

SELECT name, email FROM Teachers | 输入一个SQL表达式来过滤结果

	name	email
1	王老师	wanglaoshi@example.com
2	赵老师	zhaolaoshi@example.com

- 连接查询
 - 查询每个学生的姓名及其所选课程名称和成绩

```
SELECT S.name AS student_name, C.name AS course_name, E.grade
FROM Students S
JOIN Enrollments E ON S.student_id = E.student_id
JOIN Courses C ON E.course_id = C.course_id;
```

students(+) 1 ×

SELECT S.name AS student_name, C.name AS course_name | 输入一个SQL表达式

	student_name	course_name	grade
1	张三	数据库原理	88.5
2	李四	操作系统	92

2. 查询每

门课程的名称及其授课教师的姓名

```
SELECT C.name AS course_name, T.name AS teacher_name
FROM Courses C
JOIN Teachers T ON C.teacher_id = T.teacher_id;
```

SELECT C.name AS course_name, T.name AS teacher_name | 输入一个SQL表达式

	course_name	teacher_name
1	数据库原理	王老师
2	操作系统	赵老师

- 嵌套查询

1. 查询没有选修任何课程的学生

```
INSERT INTO Students
(student_id, name, email)
VALUES(3, '王五', 'ww@example.com');
SELECT *
FROM Students
WHERE student_id NOT IN (
    SELECT DISTINCT student_id FROM Enrollments
);
```

students 1 ×

SELECT * FROM Students WHERE student_id = 3 | 输入一个SQL表达式

	student_id	name	email
1	3	王五	ww@example.com

2. 查询选修了数

据库原理这门课程的所有学生的姓名

```

SELECT S.name
FROM Students S
WHERE S.student_id IN (
    SELECT E.student_id
    FROM Enrollments E
    JOIN Courses C ON E.course_id = C.course_id
    WHERE C.name = '数据库原理'
);

```

	name
1	张三

- 集合查询 查询选修了数据库原理或者操作系统课程的所有学生姓名

```

SELECT S.name
FROM Students S
JOIN Enrollments E ON S.student_id = E.student_id
JOIN Courses C ON E.course_id = C.course_id
WHERE C.name = '数据库原理'

```

UNION

```

SELECT S.name
FROM Students S
JOIN Enrollments E ON S.student_id = E.student_id
JOIN Courses C ON E.course_id = C.course_id
WHERE C.name = '操作系统';

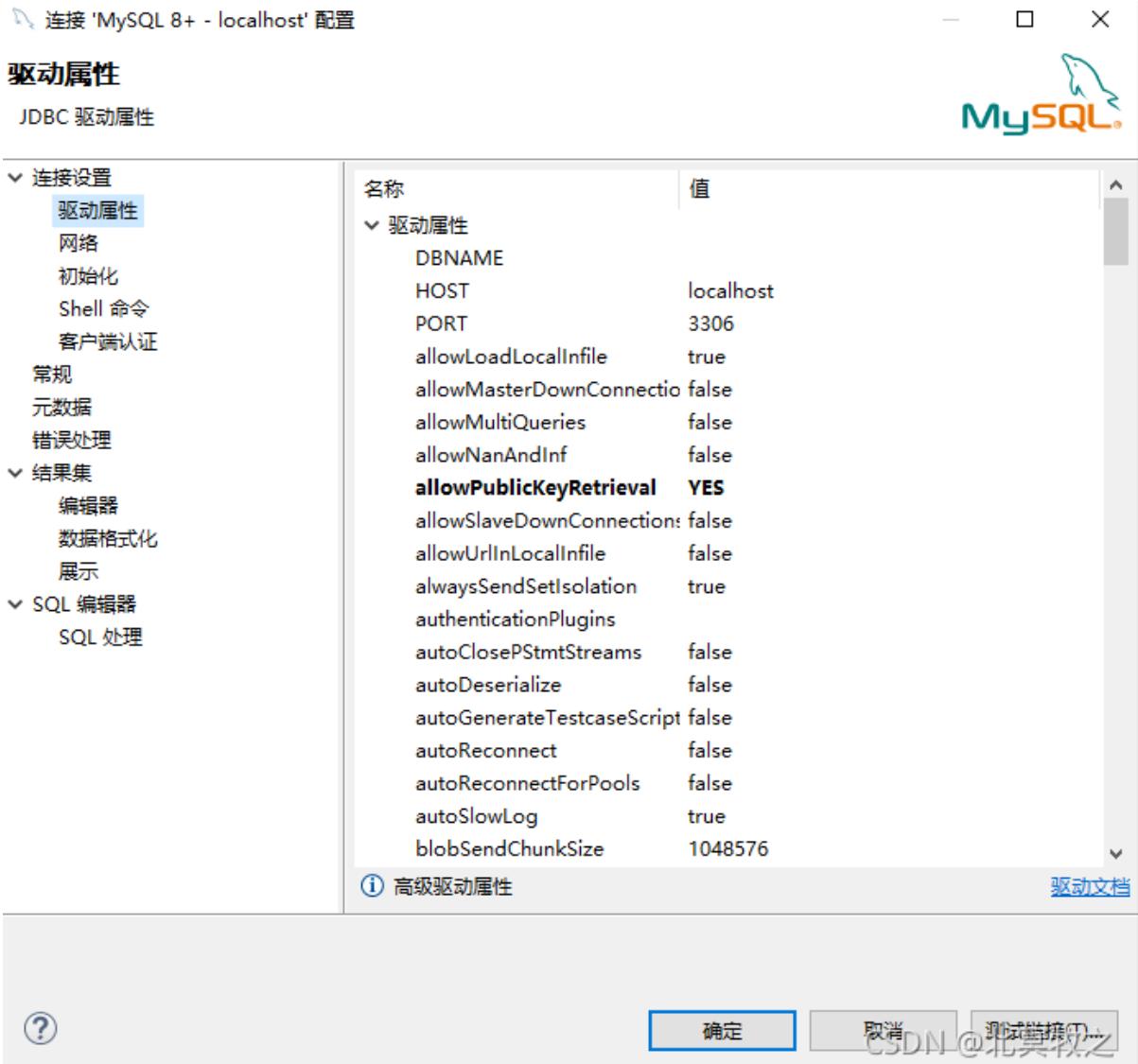
```

	name
1	张三
2	李四

出现的问题及解决方案

- 连接 MySQL 时报错 Public Key Retrieval is not allowed

- 在连接设置中将allowPublicKeyRetrieval设置为true



- 插入数据时报错表不存在

- 在四个表中点击下方保存按钮

- 插入数据时第四个表报错 Cannot add or update a child row: a foreign key constraint fails (`test_school`.`enrollments`, CONSTRAINT `enrollments_students_FK` FOREIGN KEY (`student_id`) REFERENCES `students` (`student_id`))
- 先插入 Students 表和 Teachers 表后再插入其余两个表，不在同一个 SQL 脚本中执行所有插入